## Contents

### I  General
Chapter 1.  What’s New in SAS Visual Statistics 8.4 Procedures ................................................. 3
Chapter 2.  Introduction .................................................................................................................. 5
Chapter 3.  Shared Concepts .......................................................................................................... 9

### II  Statistics
Chapter 4.  The CORRELATION Procedure ................................................................................. 89
Chapter 5.  The FREQTAB Procedure ......................................................................................... 113
Chapter 6.  The GAMMOD Procedure ........................................................................................ 303
Chapter 7.  The GENSELECT Procedure .................................................................................... 363
Chapter 8.  The ICA Procedure .................................................................................................... 427
Chapter 9.  The KCLUS Procedure ............................................................................................... 451
Chapter 10. The LMIXED Procedure ........................................................................................... 487
Chapter 11. The LOGSELECT Procedure ..................................................................................... 547
Chapter 12. The MBC Procedure ................................................................................................ 619
Chapter 13. The MODELMATRIX Procedure ............................................................................. 651
Chapter 14. The NLMOD Procedure ......................................................................................... 673
Chapter 15. The PCA Procedure ................................................................................................ 703
Chapter 16. The PHSELECT Procedure ...................................................................................... 743
Chapter 17. The PLSMOD Procedure ......................................................................................... 787
Chapter 18. The QTRSELECT Procedure ................................................................................... 827
Chapter 19. The REGSELECT Procedure .................................................................................... 881
Chapter 20. The SPC Procedure .................................................................................................. 929
Chapter 21. The TREESPLIT Procedure ..................................................................................... 1001

### III  Utility
Chapter 22. The ASSESS Procedure ............................................................................................ 1073
Chapter 23. The BINNING Procedure ......................................................................................... 1093
Chapter 24. The CARDINALITY Procedure .............................................................................. 1119
Chapter 25. The PARTITION Procedure .................................................................................... 1137
Chapter 26. The VARIMPUTE Procedure ................................................................................. 1153
Chapter 27. The VARREDUCE Procedure ................................................................................ 1163

### Subject Index

### Syntax Index
Part I

General
Overview

SAS Visual Statistics 8.4 includes a number of enhancements to existing procedures.

Enhancements

The BINNING Procedure

The new OUTLEVELBINMAP= option in the OUTPUT statement creates an output bin mapping data set for tree-based binning of class inputs.

The FREQTAB Procedure

The SENSPEC option in the TABLES statement provides estimates and confidence limits for sensitivity, specificity, positive predictive value, and negative predictive value.

The COMMONRISKDIFF(CL=KLINGENBERG) option provides Klingenberg confidence limits for the common risk difference. The COMMON= option for the risk difference plot specifies the confidence limit type for the common value. Available confidence limit types include Mantel-Haenszel, minimum risk, summary score, Newcombe, Newcombe with minimum risk weights, and Klingenberg. The COMMONRISKDIFF(PRINTWTS=SCORE) option includes the summary score weights in the “Stratum Risk Differences and Weights” table.
The CMH(I2) option provides Higgins’s I-square measure of heterogeneity for odds ratios in multiway $2 \times 2$ tables.

The **LMIXED Procedure**

The new REPEATED statement enables you to model repeated measures data by specifying the R matrix in the mixed model.

The **PCA Procedure**

The new PLOTS=PATTERN option in the PROC PCA statement produces pairwise component pattern plots to visualize the correlations between variables and components.
Chapter 2
Introduction

Overview of SAS Visual Statistics Procedures

SAS Visual Statistics procedures include predictive modeling tools that have been specially developed to take advantage of the distributed environment that SAS Viya provides. Methods include linear regression, logistic regression, quantile regression, generalized linear models, generalized additive models, reduced-rank models, nonlinear models, clustering, and decision trees. The procedures provide model selection, dimension reduction, and identification of important variables whenever this identification is appropriate for the analysis.

This book also describes utility procedures, which can be used together with any SAS Viya procedures.

**NOTE:** When you license SAS Visual Statistics, you also have access to SAS/STAT software. For more information about SAS/STAT procedures, see *SAS/STAT User’s Guide*.

About This Book

This book assumes that you are familiar with Base SAS software and with the books *SAS Language Reference: Concepts* and *Base SAS Procedures Guide*. It also assumes that you are familiar with basic SAS System concepts, such as using the DATA step to create SAS data sets and using Base SAS procedures (such as the PRINT and SORT procedures) to manipulate SAS data sets.

Chapter Organization

This book is divided into three major parts. Part I contains the following chapters:
Chapter 2, this chapter, provides an overview of SAS Visual Statistics procedures and summarizes related information, products, and services.

Chapter 3 provides information about topics that are common to multiple procedures. Topics include how to use SAS Cloud Analytic Services (CAS) sessions and how to load a SAS data set onto a CAS server. This chapter also documents the following statements that are used across a number of procedures: CLASS, CODE, EFFECT, PARTITION, and SELECTION.

Part II describes the statistical procedures and Part III describes the utility procedures. In each of these parts, the chapters appear in alphabetical order by procedure name and are organized as follows:

- The “Overview” section briefly describes the analysis provided by the procedure.
- The “Getting Started” section provides a quick introduction to the procedure through a simple example.
- The “Syntax” section describes the SAS statements and options that control the procedure.
- The “Details” section discusses methodology and other topics, such as ODS tables.
- The “Examples” section contains examples that use the procedure.
- The “References” section contains references for the methodology.

**Typographical Conventions**

This book uses several type styles for presenting information. The following list explains the meaning of the typographical conventions used in this book:

- **roman** is the standard type style used for most text.
- **UPPERCASE ROMAN** is used for SAS statements, options, and other SAS language elements when they appear in text. However, you can enter these elements in your own SAS programs in lowercase, uppercase, or a mixture of the two.
- **UPPERCASE BOLD** is used in the “Syntax” sections’ initial lists of SAS statements and options.
- **oblique** is used in the syntax definitions and in text to represent arguments for which you supply a value.
- **VariableName** is used for the names of variables and data sets when they appear in text.
- **bold** is used for matrices and vectors.
- **italic** is used for terms that are defined in text, for emphasis, and for references to publications.
- **monospace** is used for example code. In most cases, this book uses lowercase type for SAS code.
Options Used in Examples

The HTMLBLUE style is used to create the graphs and the HTML tables that appear in the online documentation. The PEARLJ style is used to create the PDF tables that appear in the documentation. A style template controls stylistic elements such as colors, fonts, and presentation attributes. You can specify a style template for an HTML ODS destination as follows:

```
ods html style=HTMLBlue;
```

You can also specify a style template for a PDF ODS destination as follows:

```
ods pdf style=PearlJ;
```

Most of the PDF tables are produced by using the following SAS System option:

```
options papersize=(6.5in 9in);
```

If you run the examples, you might get slightly different output. This is a function of the SAS System options that are used and the precision that your computer uses for floating-point calculations.

Where to Turn for More Information

Online Documentation

You can access the documentation by going to http://support.sas.com/documentation.

SAS Technical Support Services

The SAS Technical Support staff is available to respond to problems and answer technical questions regarding the use of procedures in this book. Go to http://support.sas.com/techsup for more information.
Chapter 3
Shared Concepts

Contents

Introduction to Shared Concepts .............................................. 10
Using CAS Sessions and CAS Engine Librefs .......................... 10
Loading a SAS Data Set onto a CAS Server ......................... 11
Syntax Common to SAS Visual Statistics Procedures ............. 12
CLASS Statement ......................................................... 12
CODE Statement ....................................................... 16
DISPLAY Statement ..................................................... 19
DISPLAYOUT Statement ............................................... 20
EFFECT Statement ....................................................... 21
Collection Effects ....................................................... 23
Multimember Effects .................................................... 23
Polynomial Effects ....................................................... 25
Spline Effects .......................................................... 28
Splines and Spline Bases ............................................... 31
ODS Table Names ....................................................... 35
PARTITION Statement .................................................... 36
SELECTION Statement .................................................. 36
Optimization Options ....................................................... 44
Details for SAS Visual Statistics Procedures ..................... 49
Levelization of Classification Variables ............................... 49
Specification and Parameterization of Model Effects ............. 51
Effect Operators .......................................................... 52
GLM Parameterization of Classification Variables and Effects .......................... 54
Nonsingular Parameterization .......................................... 58
Class Variable Parameterization with Unbalanced Designs ........ 61
Model Selection Methods ............................................... 62
Full Model Fitted .......................................................... 62
Forward Selection ........................................................ 62
Backward Elimination ....................................................... 64
Stepwise Selection ........................................................ 65
Forward-Swap Selection ................................................... 67
Least Angle Regression .................................................... 67
LASSO Selection .......................................................... 68
Adaptive LASSO Selection .............................................. 69
Group LASSO Selection ................................................... 69
SAS Visual Statistics procedures run on SAS Viya. One component of SAS Viya is SAS Cloud Analytic Services (CAS), which is the analytic server and associated cloud services. The following subsections describe how to set up and use CAS sessions.

The section “Syntax Common to SAS Visual Statistics Procedures” on page 12 describes the common syntax elements that are supported by some of the procedures in this book. In some cases, individual procedures implement these common elements in slightly different ways. When this occurs, the differences are described in the respective procedure chapters.

The section “Details for SAS Visual Statistics Procedures” on page 49 provides details that are common to some of the procedures in this book.

### Using CAS Sessions and CAS Engine Librefs

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:
In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:

```sas
proc options option=(CASHOST CASPORT);
run;
```

The CAS statement creates the CAS session named `mysess`, and the LIBNAME statement creates the `mycas` CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the CASHOST and CASPORT of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the `mysess` session, you can terminate it by using the TERMINATE option in the CAS statement as follows:

```sas
cas mysess terminate;
```

For more information about the CAS statement and the LIBNAME statement, see *SAS Cloud Analytic Services: Language Reference*. For general information about CAS and CAS sessions, see *SAS Cloud Analytic Services: Fundamentals*.

---

### Loading a SAS Data Set onto a CAS Server

Procedures in this book require the input data to reside on a CAS server. To work with a SAS data set, you must first load the data set onto the CAS server. Data loaded on the CAS server are called *data tables*. This section lists three methods of loading a SAS data set onto a CAS server. In this section, `mycas` is the name of the caslib that is connected to the `mysess` CAS session.

- You can use a single DATA step to create a data table on the CAS server as follows:

  ```sas
data mycas.Sample;
      input y x @@;
  datalines;
  .46 1 .47 2 .57 3 .61 4 .62 5 .68 6 .69 7
  ;
  ```

  Note that DATA step operations might not work as intended when you perform them on the CAS server instead of the SAS client.

- You can create a SAS data set first, and when it contains exactly what you want, you can use another DATA step to load it onto the CAS server as follows:

  ```sas
data Sample;
      input y x @@;
  datalines;
  .46 1 .47 2 .57 3 .61 4 .62 5 .68 6 .69 7 .78 8
  ;
  ```
Chapter 3: Shared Concepts

```sas
; data mycas.Sample;
  set Sample;
run;
```

- You can use the CASUTIL procedure as follows:

```sas
proc casutil sessref=mysess;
  load data=Sample casout="Sample";
quit;
```

The CASUTIL procedure can load data onto a CAS server more efficiently than the DATA step. For more information about the CASUTIL procedure, see SAS Cloud Analytic Services: Language Reference.

The mycas caslib stores the Sample data table, which can be distributed across many machine nodes. You must use a caslib reference in procedures in this book to enable the SAS client machine to communicate with the CAS session. For example, the following REGSELECT procedure statements use a data table that resides in the mycas caslib:

```sas
proc regselect data = mycas.Sample;
  ...statements...
run;
```

You can delete your data table by using the DELETE procedure as follows:

```sas
proc delete data = mycas.Sample;
run;
```

The Sample data table is accessible only in the mysess session. When you terminate the mysess session, the Sample data table is no longer accessible from the CAS server. If you want your Sample data table to be available to other CAS sessions, then you must promote your data table. For more information about data tables, see SAS Cloud Analytic Services: User’s Guide.

Syntax Common to SAS Visual Statistics Procedures

**CLASS Statement**

```
CLASS variable <(options)>...< variable <(options)></global-options> ;
```

This section applies to the following procedures: GAMMOD, GENSELECT, L MIXED, LOGSELECT, MODELMATRIX, PHSELECT, PLSMOD, QTRSELECT, REGSELECT, and TREESPLIT.

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. These variables enter the analysis not through their values, but through levels to which the unique values
are mapped. For more information about these mappings, see the section “Levelization of Classification Variables” on page 49.

If the procedure permits a classification variable as a response (dependent variable or target), the response does not need to be specified in the CLASS statement.

You can specify options either as individual variable options, by enclosing the options in parentheses after the variable name, or as global-options, by placing them after a slash (/). Global-options are applied to all variables that are specified in the CLASS statement. If you specify more than one CLASS statement, the global-options that are specified in any one CLASS statement apply to all CLASS statements. However, individual CLASS variable options override the global-options.

Table 3.1 summarizes the values you can use for either an option or a global-option. The options are described in detail in the list that follows Table 3.1.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESCENDING</td>
<td>Reverses the sort order</td>
</tr>
<tr>
<td>MISSING</td>
<td>Treats missing values as valid levels</td>
</tr>
<tr>
<td>ORDER=</td>
<td>Specifies the sort order for the levels</td>
</tr>
<tr>
<td>PARAM=</td>
<td>Specifies the parameterization of the variable</td>
</tr>
<tr>
<td>REF=</td>
<td>Specifies the reference level of the variable</td>
</tr>
<tr>
<td>SPLIT</td>
<td>Allows design columns for a variable to enter or leave the model independently</td>
</tr>
</tbody>
</table>

**DESCENDING**

`DESC` reverses the sort order of the classification variable. If both the DESCENDING and ORDER= options are specified, the procedure orders the categories according to the ORDER= option and then reverse that order.

**MISSING**

`MISSING` treats missing values (".", ".A", . . . , ".Z" for numeric variables and blanks for character variables) as valid values for the CLASS variable.

If you do not specify the MISSING option, observations that have missing values for CLASS variables are removed from the analysis.

**ORDER=FORMATTED | FREQ | INTERNAL**

specifies the sort order for the levels of classification variables. This ordering determines which parameters in the model correspond to each level in the data.

The following table shows how values of the ORDER= option are interpreted.
Table 3.2 Value of PARAM=

<table>
<thead>
<tr>
<th>Value of PARAM=</th>
<th>Coding</th>
</tr>
</thead>
<tbody>
<tr>
<td>EFFECT</td>
<td>Effect coding. The REF= option in the CLASS statement determines the reference level.</td>
</tr>
<tr>
<td>GLM</td>
<td>Less-than-full-rank reference cell coding. This keyword can be used only as a global-option and is applied to all CLASS variables; all other individual variable parameterization specifications are ignored. The REF= option in the CLASS statement indirectly determines the reference level through the order of levels.</td>
</tr>
<tr>
<td>ORDINAL</td>
<td>THERMOMETER</td>
</tr>
<tr>
<td>POLYNOMIAL</td>
<td>POLY</td>
</tr>
<tr>
<td>REFERENCE</td>
<td>REF</td>
</tr>
<tr>
<td>ORTHEFFECT</td>
<td></td>
</tr>
<tr>
<td>ORTHORDINAL</td>
<td>ORTHOTHERM</td>
</tr>
<tr>
<td>ORTHPOLY</td>
<td></td>
</tr>
</tbody>
</table>
Table 3.2  continued

<table>
<thead>
<tr>
<th>Value of PARAM=</th>
<th>Coding</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORTHREF</td>
<td>Orthogonalizes PARAM=REFERENCE coding. The REF= option in the CLASS statement determines the reference level.</td>
</tr>
</tbody>
</table>

All parameterizations are full rank, except for the GLM parameterization. If you specify a full rank parameterization for any CLASS variable, then every CLASS variable without a specified coding is given the EFFECT coding.

By default, PARAM=GLM. For more information about how parameterization of classification variables affects the construction and interpretation of model effects, see the section “Specification and Parameterization of Model Effects” on page 51.

\[
\text{REF} = \text{'level' | keyword}
\]

\[
\text{REFERENCE} = \text{'level' | keyword}
\]

specifies the reference level that is used when you specify a nonsingular parameterization. You can specify the following values:

- \text{'level'} specifies the level of the variable to use as the reference level. Specify the formatted value of the variable if a format is assigned. You can specify this value only for an individual variable option.
- \text{FIRST} designates the first ordered level as reference. You can specify this value either for an individual variable option or for a global-option.
- \text{LAST} designates the last ordered level as reference. You can specify this value either for an individual variable option or for a global-option.

By default, REF=LAST.

\[
\text{SPLIT}
\]

specifies that design matrix columns that correspond to any effect that contains a split classification variable can be selected to enter or leave a model independently of the other design columns of that effect.

Suppose that the variable temp has three levels (‘hot’, ‘warm’, and ‘cold’), that the variable gender has two levels (‘M’ and ‘F’), and that the variables are used in a PROC REGSELECT run as follows:

```plaintext
proc regselect data=mycas.data;
   class temp gender / split;
   model y = gender gender*temp;
run;
```

The two effects in the MODEL statement are split into eight independent effects. The effect “gender” is split into two effects that are labeled “gender_M” and “gender_F”. The effect “gender*temp” is split into six effects that are labeled “gender_M*temp_hot”, “gender_F*temp_hot”, “gender_M*temp_warm”, “gender_F*temp_warm”, “gender_M*temp_cold”, and “gender_F*temp_cold”. The previous PROC REGSELECT step is equivalent to the following:
Chapter 3: Shared Concepts

```sas
proc regselect data=mycas.data;
  model y = gender_M gender_F
         gender_M*temp_hot gender_F*temp_hot
         gender_M*temp_warm gender_F*temp_warm
         gender_M*temp_cold gender_F*temp_cold;
run;
```

The SPLIT option can be used on individual classification variables. For example, consider the following PROC REGSELECT step:

```sas
proc regselect data=mycas.data;
  class temp(split) gender;
  model y = gender gender*temp;
run;
```

In this case, the effect “gender” is not split and the effect “gender*temp” is split into three effects, which are labeled “gender*temp_hot”, “gender*temp_warm”, and “gender*temp_cold”. Furthermore, each of these three split effects now has two parameters that correspond to the two levels of “gender.” The PROC REGSELECT step is equivalent to the following:

```sas
proc regselect data=mycas.data;
  class gender;
  model y = gender gender*temp_hot gender*temp_warm gender*temp_cold;
run;
```

---

**CODE Statement**

```sas
CODE <options> ;
```

This section applies to the following procedures: GENSELECT, KCLUS, LOGSELECT, PCA, PHSELECT, QTRSELECT, REGSELECT, TREESPLIT, and VARIMPUTE.

The CODE statement writes SAS DATA step code for computing predicted values of the fitted model to a file, to a catalog entry, or to a CAS table. For survival procedures, the predicted values include survival probabilities and cumulative hazards at specific time points. To score new data, you can then include the file or the catalog entry in a DATA step, or you can specify the CAS table in the `runCodeTable` action in the `dataStep` action set.

Table 3.3 summarizes the `options` that you can specify in the CODE statement.
### Table 3.3  CODE Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMMENT</td>
<td>Adds comments to the generated code</td>
</tr>
<tr>
<td>CUMHAZ</td>
<td>Generates cumulative hazard values at specified time points</td>
</tr>
<tr>
<td>FILE=</td>
<td>Names the file in which to save the generated code</td>
</tr>
<tr>
<td>FORMATWIDTH=</td>
<td>Specifies the numeric format width for the regression coefficients</td>
</tr>
<tr>
<td>INDENTSIZE=</td>
<td>Specifies the number of spaces to indent the generated code</td>
</tr>
<tr>
<td>IPROB</td>
<td>Computes individual predicted probabilities for ordinal response models</td>
</tr>
<tr>
<td>LABELID=</td>
<td>Specifies a number used to construct names and labels</td>
</tr>
<tr>
<td>LINESIZE=</td>
<td>Specifies the line size for the generated code</td>
</tr>
<tr>
<td>NOSURVIVAL</td>
<td>Suppresses the generation of survival probabilities</td>
</tr>
<tr>
<td>NOTRIM</td>
<td>Compares formatted values, including blank padding</td>
</tr>
<tr>
<td>OUT=</td>
<td>Names an output CAS table in which to save the generated code</td>
</tr>
<tr>
<td>PCATALL</td>
<td>Generates probabilities for all levels of categorical response variables</td>
</tr>
<tr>
<td>SHOWTIME</td>
<td>Creates variables that contain the time points at which predictions are made</td>
</tr>
<tr>
<td>TIMEPOINT=</td>
<td>Specifies the time points at which survival probabilities or cumulative hazards are predicted</td>
</tr>
</tbody>
</table>

If you do not specify the FILE= option or the OUT= option and if your SAS client has a default path, then the SAS scoring code is written to an external file named `_code_`. You can specify the following *options* in the CODE statement.

**COMMENT**

adds comments to the generated code.

**CUMHAZ**

generates SAS code to predict the cumulative hazard function at the time points that you specify in the **TIMEPOINT=** option. This option applies to the PHSELECT procedure.

**FILE=** *filename*

names the external file that saves the generated code. When enclosed in a quoted string (for example, `FILE="c:\mydir\scorecode.sas"`), this option specifies the path and filename for writing the code to an external file. If you do not specify a path but your SAS client has a default path, then the code is written to an external file named *filename* at that location. You can also specify an unquoted *filename* of no more than eight characters. If the *filename* is assigned as a fileref in a Base SAS FILENAME statement, the file specified in the FILENAME statement is opened; otherwise, if your SAS client has a default path, an external file named *filename* is created. This option cannot be specified with the **OUT=** option.

**FORMATWIDTH=** *width*

specifies the width to use in formatting derived numbers such as parameter estimates. You can specify a value in the range 12 to 32; the default is 20.

**INDENTSIZE=** *n*

specifies the number of spaces to indent the generated code. You can specify a value in the range 0 to 10; the default is 3.
IPROB
This option applies to the following procedures: GENSELECT and LOGSELECT. Computes the individual predicted probabilities for ordinal response models. For a response variable \( Y \) with three levels, 1, 2, and 3, the individual probabilities are \( \Pr(Y = 1) \), \( \Pr(Y = 2) \), and \( \Pr(Y = 3) \).

LABELID=value
specifies a number used to construct array names and statement labels in the generated code. You can specify a value in the range 0 to 1024; the default is randomly chosen.

LINESIZE=value
LS=value
specifies the line size for the generated code. You can specify a value in the range 64 to 254; the default is 120.

NOSURVIVAL
suppresses SAS code for survival probabilities prediction. This option applies to the PHSELECT procedure. If you do not specify this option, survival procedures generate SAS code to predict survival probabilities at the time points that you specify in the TIMEPOINT= option.

NOTRIM
bases comparisons of formatted values on the full format width, including blank padding. By default, blanks at the beginning and end of strings are ignored.

OUT=CAS-libref.data-table
creates a CAS table that contains the score code. \( \text{CAS-libref.data-table} \) is a two-level name, where \( \text{CAS-libref} \) refers to the caslib and session identifier, and \( \text{data-table} \) specifies the name of the output data table. For more information about this two-level name, see the section “Using CAS Sessions and CAS Engine Librefs” on page 10. You can specify this CAS table in the runCodeTable action in the dataStep action set to score another data table; for more information, see SAS Viya: System Programming Guide. This option cannot be specified with the FILE= option.

PCATALL
generates probabilities for all levels of categorical response variables. This option applies to the following procedures: GENSELECT and LOGSELECT.

SHOWTIME
creates variables that contain the time points at which predictions are made. This option applies to the PHSELECT procedure. If \( T \) is the name of the failure time variable, these time point variables are named \( T_1 \), \( T_2 \), and so on.

TIMEPOINT=list | QUANTILE(probability-list)
TIME=list | QUANTILE(probability-list)
specifies the time points at which survival probabilities or cumulative hazards are predicted. This option applies to the PHSELECT procedure. You can specify a list of numbers that represent exact time points. For example:

\[
\text{code cumhaz timepoint = 40 to 60 by 10;}
\]
If \( T \) is the name of the failure time variable, the preceding specification requests the predicted cumulative hazard and survival probability at the time points \( T=40, T=50, \) and \( T=60 \). If you also specify the \texttt{SHOWTIME} option, these time points are saved in the variables \( T_1, T_2, \) and \( T_3 \), respectively. The predicted variables are named and labeled as shown in Table 3.4.

<table>
<thead>
<tr>
<th>( T )</th>
<th>Name</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>( C_{T_1} )</td>
<td>Cumulative Hazard at ( T_1 )</td>
</tr>
<tr>
<td>50</td>
<td>( C_{T_2} )</td>
<td>Cumulative Hazard at ( T_2 )</td>
</tr>
<tr>
<td>60</td>
<td>( C_{T_3} )</td>
<td>Cumulative Hazard at ( T_3 )</td>
</tr>
<tr>
<td>40</td>
<td>( S_{T_1} )</td>
<td>Survival Probability at ( T_1 )</td>
</tr>
<tr>
<td>50</td>
<td>( S_{T_2} )</td>
<td>Survival Probability at ( T_2 )</td>
</tr>
<tr>
<td>60</td>
<td>( S_{T_3} )</td>
<td>Survival Probability at ( T_3 )</td>
</tr>
</tbody>
</table>

Alternatively, you can use the keyword \texttt{QUANTILE} to specify a list of quantile probabilities. For example:

\[
\texttt{code timepoint = quantile(.2, .5, .8);} 
\]

Survival procedures compute the 20th, 50th, and 80th percentiles from the Kaplan-Meier curve (or the Breslow curve if you also specify the \texttt{ENTRY=} option in the \texttt{MODEL} statement) and use them as the time points for the prediction.

By default, \texttt{TIMEPOINT=}\texttt{QUANTILE(0.25, 0.50, 0.75)}.

### DISPLAY Statement

\[
\texttt{DISPLAY <table-list> < / options> ;} 
\]

This section applies to the following procedures: \texttt{CORRELATION, GAMMOD, GENSELECT, ICA, KCLUS, LMIXED, LOGSELECT, MBC, NLMOD, PARTITION, PCA, PHSELECT, PLSMOD, QTRSELECT, REGSELECT, and VARREDUCE.}

The \texttt{DISPLAY} statement enables you to specify a list of display tables to display or exclude. This statement is similar to the \texttt{ODS SELECT}, \texttt{ODS EXCLUDE}, and \texttt{ODS TRACE} statements. However, the \texttt{DISPLAY} statement can improve performance when a large number of tables could be generated (such as in BY-group processing). The procedure processes the \texttt{DISPLAY} statement on a CAS server and thus sends only a subset of ODS tables to the SAS client. Because ODS statements are processed on a SAS client, first all the generated display tables are sent to the client, and then the client creates a subset.

If you use both \texttt{DISPLAY} and \texttt{ODS} statements together, the \texttt{DISPLAY} statement takes precedence over the \texttt{ODS} statements. Note that the \texttt{ODS EXCLUDE} statement processes tables that are sent to the client after they have been filtered by the \texttt{DISPLAY} statement. In some cases, it might appear that the \texttt{ODS EXCLUDE}
You can specify the table-list as a list of table names, paths, partial pathnames, and regular expressions.

The table names that you can specify are listed in the specific procedure chapters. A path is a table name that is prefixed with dot-separated grouping information. For example, a SelectionSummary table that a procedure produces during a selection routine might have the path `Bygroup1.Summary.SelectionSummary`. A partial pathname does not include all groups; for example, `SelectionSummary` and `Summary.SelectionSummary` are partial pathnames for `Bygroup1.Summary.SelectionSummary`.

When you specify a table name or partial pathname, all display tables whose paths end in the specified name are selected for display or exclusion. For example, both `SelectionSummary` and `Summary.SelectionSummary` select `Bygroup1.Summary.SelectionSummary`.

A regular expression is enclosed in forward slashes (/). For example, specifying “/tions/” selects all pathnames that contain the substring “tions”; in particular, the `Bygroup1.Summary.SelectionSummary` table is selected. Specifying “!/tions/” selects all pathnames that do not contain the substring “tions”; in particular, the `Bygroup1.Summary.SelectionSummary` table is not selected.

You can specify the following options after a slash (/):

- **CASESENSITIVE** performs a case-sensitive comparison of table names in the table-list to display table names when tables are subsetted for display. To preserve case, you must enclose table names in the table-list in quotation marks.

- **EXCLUDE** displays all display tables except those that you specify in the table-list.

- **EXCLUDEALL** suppresses display of all tables. This option takes precedence over the other options.

- **TRACE** displays the display table names, labels, and paths.

---

**DISPLAYOUT Statement**

```
DISPLAYOUT table-spec-list </options> ;
```

This section applies to the following procedures: CORRELATION, GAMMOD, GENSELECT, ICA, KCLUS, LMIXED, LOGSELECT, MBC, NLMOD, PARTITION, PCA, PHSELECT, PLSMOD, QTRSELECT, REGSELECT, and VARREDUCE.

The DISPLAYOUT statement enables you to create CAS output tables from your displayed output. This statement is similar to the ODS OUTPUT statement. For more information about ODS, see *SAS Output Delivery System: Procedures Guide*.

The table-spec-list specifies a list of CAS output tables to create. Each entry in the list has either a key=value format or a key format:
The EFFECT statement enables you to construct special collections of columns for design matrices. These collections are called constructed effects to distinguish them from the usual model effects that are formed from continuous or classification variables, as discussed in the section “GLM Parameterization of Classification Variables and Effects” on page 54. For example, the terms A, B, x, A*x, and A*B in the following statements define fixed effects of the usual type in a generalized linear model:

```plaintext
proc genselect;
    class A B;
    model y = A B x A*x;
run;
```

A constructed effect, on the other hand, is assigned through the EFFECT statement. For example, in the following program, the EFFECT statement defines a constructed effect named spl:

```plaintext
proc genselect;
    class A B;
    effect spl = spline(x);
    model y = A B A*spl;
run;
```
The columns of spl are formed from the data set variable x as a cubic B-spline basis with three equally spaced interior knots.

Each constructed effect corresponds to a collection of columns that are referred to by the name you supply. You can specify multiple EFFECT statements, and all EFFECT statements must precede the MODEL statement.

You must specify the following arguments:

- **effect-name** names the effect. This name can appear in only one EFFECT statement and cannot be the name of a variable in the input data set.

- **effect-type** specifies the type of effect. You can specify the following effect-types:
  - **COLLECTION** specifies a collection effect that defines one or more variables as a single effect that has multiple degrees of freedom. The variables in a collection are considered as a unit for purposes of estimation and inference. For more information, see the section “Collection Effects” on page 23.
  - **MULTIMEMBER | MM** specifies a multimember classification effect whose levels are determined by one or more variables that appear in a CLASS statement. For more information, see the section “Multimember Effects” on page 23.
  - **POLYNOMIAL | POLY** specifies a multivariate polynomial effect in the specified numeric variables. For more information, see the section “Polynomial Effects” on page 25.
  - **SPLINE** specifies a regression spline effect whose columns are univariate spline expansions of one or more variables. A spline expansion replaces the original variable with an expanded or larger set of new variables. For more information, see the section “Spline Effects” on page 28.

- **var-list** specifies a list of variables that are used in constructing the effect. You can also specify any of the effect-options that are shown in Table 3.5 after a slash following the var-list.

### Table 3.5 EFFECT Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DETAILS</strong></td>
<td>Displays the constituents of the collection effect</td>
</tr>
<tr>
<td><strong>NOEFFECT</strong></td>
<td>Specifies that observations whose levels are all missing for the multimember variables should have 0 values in the corresponding design matrix columns</td>
</tr>
<tr>
<td><strong>STDIZE</strong></td>
<td>Standardizes the design matrix entries so that each observation has a sum of 1</td>
</tr>
</tbody>
</table>
**Table 3.5  continued**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WEIGHT=</td>
<td>Specifies the weight variable for the contributions of each classification effect</td>
</tr>
</tbody>
</table>

**Polynomial Effects Options**

- **DEGREE=** Specifies the degree of the polynomial
- **DETAILS** Displays details of the specified polynomial
- **MDEGREE=** Specifies the maximum degree of any variable in a term of the polynomial
- **NOSEPARATE** Treats the polynomial as a single effect with multiple degrees of freedom
- **STANDARDIZE=** Specifies centering and scaling suboptions for the variables that define the polynomial

**Spline Effects Options**

- **BASIS=** Specifies the type of basis (B-spline basis or truncated power function basis) for the spline effect
- **DATABOUNDARY** Uses the extremes of the data as boundary knots for a B-spline basis
- **DEGREE=** Specifies the degree of the spline effect
- **DETAILS** Displays the knots and locations for each spline basis function
- **KNOTMAX=** Requests equally spaced right-side boundary knots starting at the variables’ maximum and ending at the KNOTMAX= value
- **KNOTMETHOD=** Specifies how to construct the knots for the spline effect
- **KNOTMIN=** Requests equally spaced left-side boundary knots starting at the KNOTMIN= value and ending at the variables’ minimum value
- **NATURALCUBIC** Specifies a natural cubic spline basis for the spline effect
- **SEPARATE** Treats the spline basis for each variable as a separate effect when multiple variables are specified
- **SPLIT** Treats each design matrix column as a separate effect for selection methods

**Collection Effects**

**EFFECT**  

`EFFECT effect-name=COLLECTION (var-list < / DETAILS >) ;`

You use a collection effect to define a set of variables that are treated as a single effect that has multiple degrees of freedom. The variables in `var-list` can be continuous or classification variables. Columns in the design matrix that are contributed by a collection effect are the design columns of its constituent variables in the order in which they appear in the definition of the collection effect. If you specify the DETAILS option, then a table that shows the constituents of the collection effect is displayed.

**Multimember Effects**

**EFFECT**  

`EFFECT effect-name=MULTIMEMBER (var-list < / mm-options>) ;`

`EFFECT effect-name=MM (var-list < / mm-options>) ;`
A multimember effect is formed from one or more classification variables in such a way that each observation can be associated with one or more levels of the union of the levels of the classification variables. In other words, a multimember effect is a classification-type effect with possibly more than one nonzero column entry for each observation. Multimember effects are useful, for example, in modeling the following:

- nurses’ effects on patient recovery in hospitals
- teachers’ effects on student scores
- lineage effects in genetic studies

The levels of a multimember effect consist of the union of formatted values of the variables that define this effect. Each such level contributes one column to the design matrix. For each observation, the value that corresponds to each level of the multimember effect in the design matrix is the number of times that this level occurs for the observation.

For example, the following data provide teacher information and end-of-year test scores for students after two semesters:

<table>
<thead>
<tr>
<th>Student</th>
<th>Score</th>
<th>Teacher1</th>
<th>Teacher2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mary</td>
<td>87</td>
<td>Tobias</td>
<td>Cohen</td>
</tr>
<tr>
<td>Tom</td>
<td>89</td>
<td>Rodriguez</td>
<td>Tobias</td>
</tr>
<tr>
<td>Fred</td>
<td>82</td>
<td>Cohen</td>
<td>Cohen</td>
</tr>
<tr>
<td>Jane</td>
<td>88</td>
<td>Tobias</td>
<td>.</td>
</tr>
<tr>
<td>Jack</td>
<td>99</td>
<td>.</td>
<td>.</td>
</tr>
</tbody>
</table>

For example, Mary had different teachers in the two semesters, Fred had the same teacher in both semesters, and Jane received instruction only in the first semester.

You can model the effect of the teachers on student performance by specifying a multimember effect as follows:

```
CLASS teacher1 teacher2;
EFFECT teacher = MM(teacher1 teacher2);
```

The levels of the teacher effect are Cohen, Rodriguez, and Tobias, and the associated design matrix columns are as follows:

<table>
<thead>
<tr>
<th>Student</th>
<th>Cohen</th>
<th>Rodriguez</th>
<th>Tobias</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mary</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Tom</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Fred</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Jane</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Jack</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
</tbody>
</table>

You can specify the following `mm-options` after a slash (`/`):
DETAILS

displays a table that shows the levels of the multimember effect.

NOEFFECT

specifies that, for observations of the multimember variables whose levels are all missing, the values in
the corresponding design matrix columns be set to 0. If, in the preceding example, the teacher effect is
defined by the following statement, then the associated design matrix columns’ values for Jack are all
0:

\[
\text{EFFECT teacher = MM(teacher1 teacher2 / noeffect);}\]

This statement enables you to include Jack in the analysis even though there is no effect of teachers on
his performance.

A situation where it is important to designate observations as having no effect that can be attributed to
a classification variable is the analysis of crossover designs, where lagged treatment levels are used to
model the carryover effects of treatments between periods. Because there is no carryover effect for the
first period, the treatment lag effect in a crossover design can be modeled by using a multimember
effect that consists of a single classification variable and the NOEFFECT option, as in the following
statements:

\[
\text{CLASS Treatment lagTreatment;} \\
\text{EFFECT Carryover = MM(lagTreatment / noeffect);}\]

The lagTreatment variable contains a missing value for the first period. Otherwise, it contains the value
of the treatment variable for the preceding period.

STDIZE

specifies that for each observation, the entries in the design matrix that corresponds to the multimember
effect be scaled to have a sum of 1.

WEIGHT=weight-list

specifies numeric variables used to weigh the contributions of each of the classification effects that
define the constructed multimember effect. The number of variables in weight-list must match the
number of classification variables that define the effect.

Polynomial Effects

\[
\text{EFFECT effect-name=POLYNOMIAL (var-list / polynomial-options);} \]
\[
\text{EFFECT effect-name=POLY (var-list / polynomial-options);}\]

The variables in var-list must be numeric. A design matrix column is generated for each term of the specified
polynomial. By default, each of these terms is treated as a separate effect for the purpose of model building.
For example, the following two PROC genselect,upper() steps yield the identical analysis:

\[
\text{proc genselect;} \\
\text{effect MyPoly = polynomial(x1-x3/degree=2);} \\
\text{model y = MyPoly;} \\
\text{run;}\]
Chapter 3: Shared Concepts

```r
proc genselect;
  model y = x1 x2 x3 x1*x1 x1*x2 x1*x3 x2*x2 x2*x3 x3*x3;
run;
```

You can specify the following *polynomial-options* after a slash (/):

**DEGREE\(=n\)**

specifies the degree of the polynomial, where \(n\) must be a positive integer and is typically a small integer, such as 1, 2, or 3. By default, DEGREE=1.

**DETAILS**

displays a table that shows the details of the specified polynomial, including the number of terms generated. If you also specify the STANDARDIZE option, then a table that shows the standardization details is also produced.

**MDEGREE\(=n\)**

specifies the maximum degree of any variable in a term of the polynomial, where \(n\) must be a positive integer. The default is the degree of the specified polynomial. For example, the following statement generates the terms \(x_1, x_2, x_1^2, x_1x_2, x_2^2, x_1^2x_2, x_1x_2^2\) and \(x_1^2x_2^2\):

```r
EFFECT MyPoly=POLYNOMIAL(x1 x2/degree=4 MDEGREE=2);
```

**NOSEPARATE**

treats the polynomial as a single effect that has multiple degrees of freedom. The *effect-name* that you specify is used as the constructed effect name, and the labels of the terms are used as labels of the corresponding parameters.

**STANDARDIZE \(< (\text{centerscale-opts}) > < = \text{standardize-opt} >\)**

standardizes the variables that define the polynomial. By default, the standardized variables receive prefix “s_” in the variable names.

You can use the following *centerscale-opts* to specify how the center and scale are estimated:

**METHOD=MOMENTS**

estimates the center by the variable mean and the scale by the standard deviation. If a weight variable is specified using a WEIGHT statement, the weights are not used in the computation of the mean and standard deviation, and observations that have invalid weights are ignored. Only observations that are used in performing the analysis are used for the standardization.

**METHOD=RANGE**

estimates the center by the midpoint of the variable range and the scale as half the variable range. Any observation that has a missing value for any regressor used in the model is ignored when the range of variables in a polynomial effect is computed. Observations that have valid regressor values but missing or invalid values of frequency variables, weight variables, or dependent variables are used in computing variable ranges. By default, METHOD=RANGE.

**METHOD=WMOMENTS**

is the same as METHOD=MOMENTS except that weighted means and weighted standard deviations are used.
PREFIX=NONE | quoted-string

specifies the prefix that is appended to standardized variables when forming the term labels. You can specify the following values:

quoted-string specifies the prefix
NONE does not apply a prefix

By default, PREFIX="s_".

Let

\[ n = \text{number of observations used in the analysis} \]
\[ w = \text{weight variable} \]
\[ f = \text{frequency variable} \]
\[ x = \text{variable to be standardized} \]

\[ x_{(n)} = \max_{i=1}^{n} (x_i) \]
\[ x_{(1)} = \min_{i=1}^{n} (x_i) \]
\[ F = \text{sum of frequencies} \]
\[ = \sum_{i=1}^{n} f_i \]
\[ WF = \text{sum of weighted frequencies} \]
\[ = \sum_{i=1}^{n} w_i f_i \]

Table 3.6 shows how the center and scale are computed for each of the supported methods.

<table>
<thead>
<tr>
<th>Method</th>
<th>Center</th>
<th>Scale</th>
</tr>
</thead>
<tbody>
<tr>
<td>RANGE</td>
<td>((x_{(n)} + x_{(1)})/2)</td>
<td>((x_{(n)} - x_{(1)})/2)</td>
</tr>
<tr>
<td>MOMENTS</td>
<td>( \bar{x} = \frac{\sum_{i=1}^{n} f_i x_i}{F} )</td>
<td>( \sqrt{\frac{\sum_{i=1}^{n} f_i (x_i - \bar{x})^2}{(F - 1)}} )</td>
</tr>
<tr>
<td>WMOMENTS</td>
<td>( \bar{x}<em>w = \frac{\sum</em>{i=1}^{n} w_i f_i x_i}{WF} )</td>
<td>( \sqrt{\frac{\sum_{i=1}^{n} w_i f_i (x_i - \bar{x}_w)^2}{(WF - 1)}} )</td>
</tr>
</tbody>
</table>

You can control whether the standardization is to center, scale, or both center and scale by specifying one of the following standardize-opts:

CENTER
centers but does not scale the variables. For a variable \( x \),

\[ s_x = x - \text{center} \]

CENTERSCALE
centers and scales the variables. This is the default if you do not specify a standardization-opt. For a variable \( x \),

\[ s_x = \frac{x - \text{center}}{\text{scale}} \]
NONE
performs no standardization.

SCALE
scales but does not center the variables. For a variable $x$,

$$s_x = \frac{x}{\text{scale}}$$

**Spline Effects**

This section discusses the construction of spline effects through the EFFECT statement. A spline effect expands variables into spline bases whose form depends on the options that you specify. For more information about regression splines and spline bases, see the section “Splines and Spline Bases” on page 31. You request a spline effect with the syntax

```plaintext
EFFECT name=SPLINE (var-list < / spline-options>) ;
```

The variables in `var-list` must be numeric. Design matrix columns are generated separately for each of these variables, and the set of columns is collectively referred to with the specified name. By default, the spline basis that is generated for each variable is a cubic B-spline basis that has three equally spaced knots positioned between the minimum and maximum values of that variable. This yields by default seven design matrix columns for each of the variables in the SPLINE effect.

You can specify the following `spline-options` after a slash (\(/\)):

**BASIS=BSPLINE**
uses a B-spline basis for the spline expansion. For splines of degree $d$ that are defined with $n$ knots, this basis consists of $n + d + 1$ columns. In order to completely specify the B-spline basis, $d$ left-side boundary knots and $\max\{d, 1\}$ right-side boundary knots are also required. For information about how to specify the positions of both the internal and boundary knots, see the `KNOTMETHOD=`, `DATABOUNDARY`, `KNOTMIN=`, and `KNOTMAX=` suboptions.

**BASIS=TPF(options)**
specifies a truncated power function basis for the spline expansion. For splines of degree $d$ that are defined with $n$ knots for a variable $x$, this basis consists of an intercept, polynomials $x, x^2, \ldots, x^d$, and one truncated power function for each of the $n$ knots. Unlike the B-spline basis, no boundary knots are required. For information about how you can specify the position of the internal knots, see the `KNOTMETHOD=` suboption.

You can modify the number of columns when you request `BASIS=TPF` by specifying the following `options`:

**NOINT** excludes the intercept column.

**NOPowers** excludes the intercept and polynomial columns.

By default, `BASIS=BSPLINE`.

**DATABOUNDARY** uses the extremes of the data as boundary knots when building a B-spline basis.
EFFECT Statement

**DEGREE=** <n>
specifies the degree of the spline transformation, where <n> must be a nonnegative integer and is typically a small integer, such as 0, 1, 2, or 3. By default, DEGREE=3.

**DETAILS**
displays tables that show the knot locations and the knots associated with each spline basis function.

**KNOTMAX=** <value>
requests that, for each variable in the EFFECT statement, the right-side boundary knots be equally spaced starting at the maximum of the variable and ending at the specified value. This option is ignored for variables whose maximum value is greater than the specified value or if the DATABOUNDARY option is also specified.

**KNOTMETHOD=** <knot-method>(knot-options)>
specifies how to construct the knots for spline effects. You can choose from the following knot-methods and affect the knot construction further with the method-specific knot-options:

- **EQUAL**(n)>
specifies that <n> equally spaced knots be positioned between the extremes of the data. By default, <n> = 3. For a B-spline basis, any needed boundary knots continue to be equally spaced unless the DATABOUNDARY option has also been specified. By default, KNOTMETHOD=EQUAL(3).

- **LIST**(number-list)
specifies the list of internal knots to be used in forming the spline basis columns. For a B-spline basis, the data extremes are used as boundary knots.

- **LISTWITHBOUNDARY**(number-list)
specifies the list of all knots that are used in forming the spline basis columns. When you use a truncated power function basis, this list is interpreted as the list of internal knots. When you use a B-spline basis of degree <d>, then the first <d> entries are used as left-side boundary knots and the last max(<d>, 1) entries in the list are used as right-side boundary knots.

- **MULTISCALE**(multiscale-options)>
generates multiple B-spline bases, which correspond to sets that have an increasing number of internal knots. As the number of internal knots increases, the spline basis that is generated can approximate features of the data at finer scales. So generating bases at multiple scales facilitates the modeling of both coarse- and fine-grained features of the data. For scale <i>, the spline basis corresponds to 2<sup>i</sup> equally spaced internal knots. By default, the bases for scales 0–7 are generated. For each scale, a separate spline effect is generated. The name of the constructed spline effect at scale <i> is formed by appending _Si to the effect name that you specify in the EFFECT statement. If you specify multiple variables in the EFFECT statement, then spline bases are generated separately for each variable at each scale and the name of the corresponding effect is obtained by appending the variable name followed by _Si to the name in the EFFECT statement. For example, the following statement generates effects named spl_x1_S0, spl_x1_S1, spl_x1_S2, ..., spl_x1_S7 and spl_x2_S1, spl_x2_S2, ..., spl_x2_S7:

```
EFFECT spl = spline(x1 x2 / knotmethod=multiscale);
```

The MULTISCALE option is ignored if you specify the BASIS=TPF spline-option.

The following multiscale-options control which scales are included:
**STARTSCALE=**\(n\) specifies the start scale, where \(n\) is a positive integer. By default, STARTSCALE=0.

**ENDSCALE=**\(n\) specifies the end scale, where \(n\) is a positive integer. By default, ENDSCALE=7.

**PERCENTILES(\(n\))**
requests that internal knots be placed at \(n\) equally spaced percentiles of the variable or variables named in the EFFECT statement. For example, the following statement positions internal knots at the deciles of the variable \(x\). For a B-spline basis, the extremes of the data are used as boundary knots:

```plaintext
EFFECT spl = spline(x / knotmethod=percentiles(9));
```

**RANGEFRACTIONS(\(fraction-list\))**
places internal knots at each fraction of the ranges of the variables in the EFFECT statement. For example, if variable \(x_1\) ranges between 1 and 3 and variable \(x_2\) ranges between 0 and 20, then the following EFFECT statement uses internal knots 1.2, 2, and 2.5 for variable \(x_1\) and internal knots 2, 10, and 15 for variable \(x_2\):

```plaintext
EFFECT spl = spline(x1 x2 / knotmethod=rangefractions(.1 .5 .75));
```

For a B-spline basis, the data extremes are used as boundary knots.

**KNOTMIN=**\(value\)
requests that, for each variable in the EFFECT statement, the left-side boundary knots be equally spaced starting at the specified value and ending at the minimum of the variable. This option is ignored for variables whose minimum value is less than the specified value or if the DATABOUNDARY option is also specified.

**NATURALCUBIC**
uses a natural cubic spline basis for the spline expansion. Natural cubic splines, also known as restricted cubic splines, are cubic splines that are constrained to be linear beyond the extreme knots. The natural cubic spline basis that is produced by the EFFECT statement is obtained by starting from the unrestricted truncated power function cubic spline basis that is defined with \(n\) distinct knots and imposes the linearity constraints beyond the extreme knots. This basis consists of an intercept, the polynomial \(x\), and \(n-2\) functions that are all linear beyond the largest knot. The \(i\)th function, \(i = 1, 2, \ldots, n-2\), is 0 to the left of the \(i\)th knot, which is called the “break knot.” For more information about this basis, see the section “Splines and Spline Bases” on page 31. You can use the NOINT and NOPowers suboptions of the BASIS=TPF option to suppress the intercept and polynomial \(x\) when the columns of the natural cubic spline basis are formed. When you specify the NATURALCUBIC option, the options BASIS=BSPLINE, DATABOUNDARY, DEGREE=, and KNOTMETHOD=MULTISCALE are not applicable.

**SEPARATE**
requests that, when multiple variables are specified in the EFFECT statement, the spline basis for each variable be treated as a separate effect. The names of these separated effects are formed by appending an underscore followed by the name of the variable to the name that you specify in the EFFECT statement. For example, the following statement generates the effect names spl_x1 and spl_x2:
EFFECT spl = spline(x1 x2 / separate);

In procedures that support variable selection, such as the GENSELECT procedure, these two effects can enter or leave the model independently during the selection process.

SPLIT

treats each individual column in the design matrix that corresponds to the spline effect as a separate effect that can enter or leave the model independently. Names for these split effects are generated by appending the variable name and an index for each column to the name that you specify in the EFFECT statement. For example, the effects generated for the spline effect in the following statement are spl_x1:1, spl_x1:2, ..., spl_x1:7 and spl_x2:1, spl_x2:2, ..., spl_x2:7:

EFFECT spl = spline(x1 x2 / split);

Splines and Spline Bases

This section provides details about how the EFFECT statement constructs spline bases. A spline function is a piecewise polynomial function in which the individual polynomials have the same degree and connect smoothly at join points whose abscissa values, called knots, are prespecified. You can use spline functions to fit curves to a wide variety of data.

A spline of degree 0 is a step function with steps located at the knots. A spline of degree 1 is a piecewise linear function where the lines connect at the knots. A spline of degree 2 is a piecewise quadratic curve whose values and slopes coincide at the knots. A spline of degree 3 is a piecewise cubic curve whose values, slopes, and curvature coincide at the knots. Visually, a cubic spline is a smooth curve, and it is the most commonly used spline when a smooth fit is desired. Note that when no knots are used, splines of degree \( d \) are simply polynomials of degree \( d \).

More formally, suppose you specify knots \( k_1 < k_2 < k_3 < \cdots < k_n \). Then a spline of degree \( d \geq 0 \) is a function \( S(x) \) with \( d-1 \) continuous derivatives such that

\[
S(x) = \begin{cases} 
  P_0(x) & x < k_1 \\
  P_1(x) & k_i \leq x < k_{i+1}; \ i = 1, 2, \ldots, n-1 \\
  P_n(x) & x \geq k_n
\end{cases}
\]

where each \( P_i(x) \) is a polynomial of degree \( d \). The requirement that \( S(x) \) has \( d-1 \) continuous derivatives is satisfied by requiring that the function values and all derivatives up to order \( d-1 \) of the adjacent polynomials at each knot match.

A counting argument yields the number of parameters that define a spline with \( n \) knots. There are \( n+1 \) polynomials of degree \( d \), producing \( (n+1)(d+1) \) coefficients. However, there are \( d \) restrictions at each of the \( n \) knots, so the number of free parameters is \( (n+1)(d+1) - nd = n + d + 1 \). In mathematical terminology this says that the dimension of the vector space of splines of degree \( d \) on \( n \) distinct knots is \( n + d + 1 \). If you have \( n + d + 1 \) basis vectors, then you can fit a curve to your data by regressing your dependent variable by using this basis for the corresponding design matrix columns. In this context, such a spline is known as a regression spline. The EFFECT statement provides a simple mechanism for obtaining such a basis.

If you remove the restriction that the knots of a spline must be distinct and allow repeated knots, then you can obtain functions that have less smoothness and even discontinuities at the repeated knot location. For a spline
of degree \( d \) and a repeated knot that has multiplicity \( m \leq d \), the piecewise polynomials that join such a knot are required to have only \( d - m \) matching derivatives. Note that this increases the number of free parameters by \( m - 1 \) but also decreases the number of distinct knots by \( m - 1 \). Hence the dimension of the vector space of splines of degree \( d \) with \( n \) knots is still \( n + d + 1 \), provided that any repeated knot has a multiplicity less than or equal to \( d \).

The EFFECT statement supports the commonly used truncated power function basis and B-spline basis. With exact arithmetic and by using the complete basis, you obtain the same fit with either of these bases. The following subsections provide details about constructing spline bases for the space of splines of degree \( d \) with \( n \) knots that satisfies \( k_1 \leq k_2 \leq k_3 < \cdots \leq k_n \).

**Truncated Power Function Basis**

A truncated power function for a knot \( k_i \) is a function defined by

\[
t_i(x) = \begin{cases} 
0 & x < k_i \\
(x - k_i)^d & x \geq k_i 
\end{cases}
\]

Figure 3.1 shows such functions for \( d = 1 \) and \( d = 3 \) with a knot at \( x = 1 \).

![Figure 3.1 Truncated Power Functions with Knot at x = 1](image)

The name is derived from the fact that these functions are shifted power functions that are truncated to 0 to the left of the knot. These functions are piecewise polynomial functions that have two pieces whose function values and derivatives of all orders up to \( d - 1 \) are 0 at the defining knot. Hence these functions are splines of degree \( d \). It is easy to see that these \( n \) functions are linearly independent. However, they do not form a basis, because such a basis requires \( n + d - 1 \) functions. The usual way to add \( d + 1 \) additional basis functions is to use the polynomials \( 1, x, x^2, \ldots, x^d \). These \( d + 1 \) functions together with the \( n \) truncated power functions \( t_i(x), i = 1, 2, \ldots, n \) form the truncated power basis.

Note that each time a knot is repeated, the associated exponent used in the corresponding basis function is reduced by 1. For example, for splines of degree \( d \) with three repeated knots \( k_i = k_{i+1} = k_{i+2} \), the corresponding basis functions are \( t_i(x) = (x - k_i)^d, t_{i+1}(x) = (x - k_i)^{d-1} \), and \( t_{i+2}(x) = (x - k_i)^{d-2} \). Provided that the multiplicity of each repeated knot is less than or equal to the degree, this construction continues to yield a basis for the associated space of splines.

The main advantage of the truncated power function basis is the simplicity of its construction and the ease
of interpreting the parameters in a model that corresponds to these basis functions. However, there are two weaknesses when you use this basis for regression. These functions grow rapidly without bound as \( x \) increases, resulting in numerical precision problems when the \( x \) data span a wide range. Furthermore, many or even all of these basis functions can be nonzero when evaluated at some \( x \) value, resulting in a design matrix that has few zeros and precludes the use of sparse matrix technology to speed up computation. This weakness can be addressed by using a B-spline basis.

**B-Spline Basis**

A B-spline basis can be built by starting with a set of Haar basis functions, which are functions that are 1 between adjacent knots and 0 elsewhere, and then applying a simple linear recursion relationship \( d \) times, yielding the \( n + d + 1 \) needed basis functions. For the purpose of building the B-spline basis, the \( n \) prespecified knots are called internal knots. This construction requires \( d \) additional knots, known as boundary knots, to be positioned to the left of the internal knots, and \( \max(d, 1) \) boundary knots to be positioned to the right of the internal knots. The actual values of these boundary knots can be arbitrary. The EFFECT statement provides several methods for placing the necessary boundary knots, including the common method of using repeated values of the data extremes as the boundary knots. The boundary knot placement affects the precise form of the basis functions that are generated, but it does not affect the following two desirable properties:

- The B-spline basis functions are nonzero over an interval that spans at most \( d + 2 \) knots. This yields design matrix columns each of whose rows contain at most \( d + 2 \) adjacent nonzero entries.
- The computation of the basis functions at any \( x \) value is numerically stable and does not require evaluating powers of this value.

The following figures show the B-spline bases that are defined on \([0, 1]\) with four equally spaced internal knots at 0.2, 0.4, 0.6, and 0.8.

**Figure 3.2** shows a linear B-spline basis. This basis consists of six functions, each of which is nonzero over an interval that spans at most three knots.

**Figure 3.2** Linear B-Spline Basis with Four Equally Spaced Interior Knots

**Figure 3.3** shows a cubic B-spline basis where the necessary boundary knots are positioned at \( x = 0 \) and \( x = 1 \). This basis consists of eight functions, each of which is nonzero over an interval that spans at most five knots.
Figure 3.3 Cubic B-Spline Basis with Four Equally Spaced Interior Knots

Figure 3.4 shows a different cubic B-spline basis where the necessary left-side boundary knots are positioned at –0.6, –0.4, –0.2, and 0. The right-side boundary knots are positioned at 1, 1.2, 1.4, and 1.6. As in the basis shown in Figure 3.3, this basis consists of eight functions, each of which is nonzero over an interval that spans at most five knots. The different positioning of the boundary knots has merely changed the shape of the individual basis functions.

Figure 3.4 Cubic B-Spline Basis with Equally Spaced Boundary and Interior Knots

For more information about this construction, see Hastie, Tibshirani, and Friedman (2001).
Natural Cubic Spline Basis

Natural cubic splines are cubic splines with the additional restriction that the splines are required to be linear beyond the extreme knots. Some authors prefer the terminology “restricted cubic splines” to “natural cubic splines.” The space of unrestricted cubic splines on $n$ knots has the dimension $n + 4$. Imposing the restrictions that the cubic polynomials beyond the first and last knot reduce to linear polynomials reduces the number of degrees of freedom by 4, so a basis for the natural cubic splines consists of $n$ functions. Starting from the truncated power function basis for the unrestricted cubic splines, you can obtain a reduced basis by imposing linearity constraints. For more information about this construction, see Hastie, Tibshirani, and Friedman (2001). Figure 3.5 shows this natural cubic spline basis defined on $[0, 1]$ with four equally spaced internal knots at 0.2, 0.4, 0.6, and 0.8. This basis consists of four basis functions that are all linear beyond the extreme knots at 0.2 and 0.8.

Figure 3.5 Natural Cubic Spline Basis with Four Equally Spaced Knots

ODS Table Names

Each constructed effect produces one or more informative tables when the DETAILS option is also specified. Each table has a name associated with it. This name is used to refer to the table by procedures that support a DISPLAY or DISPLAYOUT statement. You also use this name to refer to the table in ODS statements. These names are listed in Table 3.7.

Table 3.7 ODS Tables Produced by the EFFECT Statement

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSplineDetails</td>
<td>B-spline basis details</td>
<td>BASIS=BSPLINE</td>
</tr>
<tr>
<td>CollectionLevelInfo</td>
<td>Levels of collection effects</td>
<td></td>
</tr>
<tr>
<td>MMLLevelInfo</td>
<td>Levels of multimember effects</td>
<td></td>
</tr>
<tr>
<td>PolyDetails</td>
<td>Number of variables and columns, polynomial degree, and standardization method</td>
<td></td>
</tr>
<tr>
<td>PolyScaling</td>
<td>Centering and scaling details</td>
<td>STANDARDIZE</td>
</tr>
<tr>
<td>SplineKnots</td>
<td>knot and boundary knot values</td>
<td></td>
</tr>
<tr>
<td>TPFSplineDetails</td>
<td>Truncated power function spline basis details</td>
<td>BASIS=TPF or NATURALCUBIC</td>
</tr>
</tbody>
</table>
PARTITION Statement

PARTITION partition-option ;

This section applies to the following procedures: GENSELECT, LOGSELECT, PHSELECT, QTRSELECT, REGSELECT, and TREESPLIT.

The PARTITION statement specifies how observations in the input data set are logically partitioned into disjoint subsets for model training, validation, and testing. For more information, see the section “Using Validation and Test Data” on page 80. Either you can designate a variable in the input data table and a set of formatted values of that variable to determine the role of each observation, or you can specify proportions to use for randomly assigning observations to each role.

You must specify exactly one of the following partition-options:

FRACTION(<TEST= fraction > <VALIDATE= fraction > <SEED= number >)
randomly assigns specified proportions of the observations in the input data table to the roles. You specify the proportions for testing and validation by using the TEST= and VALIDATE= suboptions. If you specify both the TEST= and VALIDATE= suboptions, then the sum of the specified fractions must be less than 1 and the remaining fraction of the observations are assigned to the training role. The SEED= option specifies an integer that is used to start the pseudorandom number generator for random partitioning of data for training, testing, and validation. If you do not specify SEED=number or if number is less than or equal to 0, the seed is generated by reading the time of day from the computer’s clock.

ROLE=variable (<TEST=’value’> <TRAIN=’value’> <VALIDATE=’value’>)
ROLEVAR=variable (<TEST=’value’> <TRAIN=’value’> <VALIDATE=’value’>)
names the variable in the input data table whose values are used to assign roles to each observation. This variable cannot also appear as an analysis variable in other statements or options. The TEST=, TRAIN=, and VALIDATE= suboptions specify the formatted values of this variable that are used to assign observation roles. If you do not specify the TRAIN= suboption, then all observations whose role is not determined by the TEST= or VALIDATE= suboption are assigned to the training role.

SELECTION Statement

SELECTION <METHOD= method <(method-options)> > <options> ;

This section applies to the following procedures: GENSELECT, LOGSELECT, PHSELECT, QTRSELECT, and REGSELECT.

Procedures in this book that support model selection use the SELECTION statement to control details about the model selection process.

You can specify the following methods in the SELECTION statement:
**METHOD=method < (method-options) >**

specifies the method used to select the model. You can also specify `method-options` that apply to the specified method by enclosing them in parentheses after the `method`.

The following methods are available and are explained in detail in the section “Model Selection Methods” on page 62. By default, `METHOD=STEPWISE`.

- **NONE** specifies no model selection.
- **BACKWARD** specifies backward elimination. This method starts with all effects in the model and deletes effects.
- **FORWARD** specifies forward selection. This method starts with no effects in the model and adds effects.
- **STEPWISE** specifies stepwise regression. This method is similar to the FORWARD method except that effects already in the model do not necessarily stay there.
- **FORWARDSWAP** specifies forward-swap selection, which is an extension of the forward selection method. Before any addition step, the procedure makes all pairwise swaps of one effect in the model and one effect out of the current model that improve the selection criterion. When the selection criterion is R square, this method is the same as the MAXR method in the REG procedure in SAS/STAT software. The only procedure in this book that supports this method is the REGSELECT procedure.
- **LAR** specifies least angle regression. Like forward selection, this method starts by adding effects to an empty model. The parameter estimates at any step are “shrunk” when they are compared to the corresponding least squares estimates. If the model contains classification variables, then these classification variables are split. For more information, see the SPLIT option in the CLASS statement. The only SAS Viya procedure that supports this method is the REGSELECT procedure.
- **LASSO** adds and deletes parameters by using a version of ordinary least squares in which the sum of the absolute regression coefficients is constrained. If the model contains classification variables, then these classification variables are split. For more information, see the SPLIT option in the CLASS statement.

Table 3.8 lists the applicable `method-options` for each of these methods.
Table 3.8  Applicable method-options by method

<table>
<thead>
<tr>
<th>method-option</th>
<th>FORWARD</th>
<th>BACKWARD</th>
<th>STEPWISE</th>
<th>LAR</th>
<th>LASSO</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADAPTIVE</td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHOOSE =</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>COMPETITIVE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>CRITERION =</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FAST</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>LS_COEFS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>MAX_EFFECTS</td>
<td>x</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>MAX_STEPS</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>MIN_EFFECTS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>SELECT =</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLENTRY =</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>SL_STAY =</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>STOP =</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>

The syntax of the method-options that you can specify in parentheses after the SELECTION= option method follows. As described in Table 3.8, not all selection method-options are applicable to every method.

**ADAPTIVE < (GAMMA=nonnegative number) >**

applies adaptive weights to each of the coefficients when METHOD=LASSO. Ordinary least squares estimates of the model parameters are used to form the adaptive weights. You use the GAMMA= option to specify the power transformation that is applied to the parameters in forming the adaptive weights. By default, GAMMA=1.

**CHOOSE=criterion**

chooses from the list of models (at each step of the selection process) the model that yields the best value of the specified criterion. If the optimal value of the specified criterion occurs for models at more than one step, then the model that has the smallest number of parameters is chosen. If you do not specify the CHOOSE= option, then the selected model is the model at the final step in the selection process. The criteria that are supported depend on the type of model that is being fit. For the supported values of criterion, see the chapters for the relevant procedures.

**COMPETITIVE**

is applicable as a method-option only when METHOD=STEPWISE and the SELECT= criterion is not SL. If you specify the COMPETITIVE option, then the SELECT= criterion is evaluated for all models in which an effect currently in the model is dropped or an effect not yet in the model is added. The effect whose removal from or addition to the model yields the maximum improvement to the SELECT= criterion is dropped or added.

**CRITERION=criterion**

is an alias for the SELECT= option.
FAST implements the computational algorithm of Lawless and Singhal (1978) to compute a first-order approximation to the remaining slope estimates for each subsequent elimination of a variable from the model. When applied in backward selection, this option essentially leads to approximating the selection process as the selection process of a linear regression model in which the crossproducts matrix equals the Hessian matrix in the full model under consideration. This option is available only when METHOD=BACKWARD is specified. It is computationally efficient because the model is not fit after removal of each effect.

LSCOEFFS requests a hybrid version of the LAR and LASSO methods, in which the sequence of models is determined by the LAR or LASSO algorithm but the coefficients of the parameters for the model at any step are determined by using ordinary least squares.

MAXEFFECTS=n specifies the maximum number of effects in any model that is considered during the selection process. This option is ignored when METHOD=BACKWARD is specified. If, at some step of the selection process, the model contains the specified maximum number of effects, then no candidates for addition are considered.

MAXSTEPS=n specifies the maximum number of selection steps that are performed. The default value of n is the number of effects in the MODEL statement when METHOD=FORWARD, METHOD=BACKWARD, or METHOD=LAR. The default is three times the number of effects when METHOD=STEPWISE or METHOD=LASSO.

MINEFFECTS=n specifies the minimum number of effects in any model that is considered during backward selection. This option is ignored unless METHOD=BACKWARD is specified. The backward selection process terminates if, at some step of the selection process, the model contains the specified minimum number of effects.

SELECT=SL | criterion specifies the criterion that the procedure uses to determine the order in which effects enter or leave at each step of the selection method. For each step, the effect whose addition to or removal from the current model yields the maximum improvement in the specified criterion is selected. You can use the traditional significance-level approach by specifying the SL criterion; for other supported criteria, see the chapter for the relevant procedure. This option is not valid when METHOD=LAR or METHOD=LASSO.

SLENTRY=value
SLE=value specifies the significance level for entry when STOP=SL or SELECT=SL. By default, SLENTRY=0.05.

SLSTAY=value
SLS=value specifies the significance level for staying in the model when STOP=SL or SELECT=SL. By default, SLSTAY=0.05.
STOP=SL | NONE | criterion

specifies a criterion that is used to stop the selection process. The criteria that are supported depend on the type of model that is being fit. For information about the supported criteria, see the chapter for the relevant procedure.

If you do not specify the STOP= option but do specify the SELECT= option, then the criterion specified in the SELECT= option is also used as the STOP= criterion.

You can specify the following values:

- **NONE** stops the selection process if no suitable add or drop candidates can be found or if a size-based limit is reached. For example, if you specify STOP=NONE MAXEFFECTS=5, then the selection process stops at the first step that produces a model that has five effects.

- **SL** stops the selection process at the step where the significance level of the candidate for entry is greater than the SLENTRY= value for addition steps when METHOD=FORWARD or METHOD=STEPWISE and where the significance level of the candidate for removal is greater than the SLSTAY= value when METHOD=BACKWARD or METHOD=STEPWISE.

- **criterion** stops the selection process if the selection process produces a local extremum of this criterion or if a size-based limit is reached. For example, if you specify STOP=AIC MAXSTEPS=5, then the selection process stops before step 5 if the sequence of models has a local minimum of the AIC criterion before step 5. The determination of whether a local minimum is reached is made on the basis of a stop horizon. The default stop horizon is 3, but you can change it by using the STOPHORIZON= option. If the stop horizon is $n$ and the STOP=criterion at any step is better than the stop criterion at the next $n$ steps, then the selection process terminates.

In addition, you can also specify the following options:

- **DETAILS=NONE | SUMMARY | ALL**
  - **DETAILS=STEPS<(CANDIDATES(ALL | $n$)>**
    - specifies the level of detail to be produced about the selection process. The default is DETAILS=SUMMARY.

  The DETAILS=ALL and DETAILS=STEPS options produce the following output:

  - tables that provide information about the model that is selected at each step of the selection process.
  - entry and removal statistics for inclusion or exclusion candidates at each step. By default, only the top 10 candidates at each step are shown. If you specify STEPS(CANDIDATES($n$)), then the best $n$ candidates are shown. If you specify STEPS(CANDIDATES(ALL)), then all candidates are shown.
  - a selection summary table that shows by step the effect that is added to or removed from the model in addition to the values of the SELECT, STOP, and CHOOSE criteria for the resulting model.
  - a stop reason table that describes why the selection process stopped.
• a selection reason table that describes why the selected model was chosen.
• a selected effects table that lists the effects that are in the selected model.

The DETAILS=SUMMARY option produces only the selection summary, stop reason, selection reason, and selected effects tables.

**HIERARCHY=NONE | SINGLE | SINGLECLASS**
specifies whether and how the model hierarchy requirement is applied. You can specify that only classification effects, or both classification and continuous effects, be subject to the hierarchy requirement. This option is ignored unless you also specify one of the following options: METHOD=FORWARD, METHOD=BACKWARD, or METHOD=STEPWISE.

Model hierarchy refers to the requirement that, for any term to be in the model, all model effects that are contained in the term must be present in the model. For example, in order for the interaction A*B to enter the model, the main effects A and B must be in the model. Likewise, neither effect A nor effect B can leave the model while the interaction A*B is in the model.

You can specify the following values:

- **NONE** specifies that model hierarchy not be maintained. Any single effect can enter or leave the model at any step of the selection process.
- **SINGLE** specifies that only one effect enter or leave the model at one time, subject to the model hierarchy requirement. For example, suppose that the model contains the main effects A and B and the interaction A*B. In the first step of the selection process, either A or B can enter the model. In the second step, the other main effect can enter the model. The interaction effect can enter the model only when both main effects have already entered. Also, before A or B can be removed from the model, the A*B interaction must first be removed. All effects (CLASS and interval) are subject to the hierarchy requirement.
- **SINGLECLASS** is the same as HIERARCHY=SINGLE except that only CLASS effects are subject to the hierarchy requirement.

By default, HIERARCHY=NONE.

**ORDERSELECT** specifies that effects in the selected model are displayed in the order in which they first entered the model. If you do not specify the ORDERSELECT option, then effects in the selected model are displayed in the order in which they appear in the MODEL statement.

**PLOT (<global-plot-options>) < = plot-request < (options) >**
**PLOTS < (global-plot-options) > < = (plot-request < (options) > < … plot-request < (options) > >)** controls the selection process plots that are produced through ODS Graphics.

When you specify only one plot request, you can omit the parentheses around it. Here are some examples:

```
plots=all
plots=coefficients(unpack)
plots(unpack)=(coefficients criteria)
```

ODS Graphics must be enabled before you can request plots. For example:
ods graphics on;
proc regselect;
  model y = x1-x100;
  selection method=forward plots=all;
run;
ods graphics off;

Global Plot Options

Each global-plot-option applies to all plots that the SELECTION statement generates, unless you alter the option by using a specific plot option that follows a plot-request. You can specify the following global-plot-options:

**ENDSTEP=n**
terminates the step ranges shown on the horizontal axes of plots at the specified step. By default, the step range that is shown terminates at the final step of the selection process.

**LOGP | LOGPVALUE**
uses a logarithmic axis to display entry and removal significance levels.

**MAXPARMLABEL=n**
specifies the maximum number of characters beyond which all parameter labels on the coefficient progression panel and the coefficient progression plot are suppressed. By default, MAXPARMLABEL=20.

**MAXSTEP=LABEL=n**
specifies the maximum number of characters of effect labels to display on the horizontal axes of the plots. By default, MAXSTEP=LABEL=20. If the length of any effect label exceeds n, then step numbers are used on the horizontal axes of the plots.

**STARTSTEP=n**
specifies that the step ranges shown on the horizontal axes of plots start at the specified step. By default, the step range that is shown starts at the initial step of the selection process.

**STEPAXIS=EFFECT | NORMB | NUMBER**
specifies the horizontal axis to be used on the plots, where this axis represents the sequence of entering or departing effects. You can specify the following values:

**EFFECT**
requests that each step be labeled by a prefix followed by the name of the effect that enters or leaves at that step. The prefix consists of the step number, followed by a “+” or a “-” sign, depending on whether the effect enters (+) or leaves (-) at that step.

**NORMB**
requests that the horizontal axis value at step i be the L1 norm of the parameters at step i, normalized by the L1 norm of the parameters at the final step. STEPAXIS=NORMB is valid only with the LAR, LASSO, and elastic net selection methods.
NUMBER requests that each step be labeled by the step number.

UNPACK | UNPACKPANEL suppresses paneling. By default, multiple plots can appear in the coefficient progression panel and the criteria panel. Specify UNPACK to see each plot individually. You can also specify UNPACK as a suboption of the CRITERIA and COEFFICIENTS options.

Specific Plot Options

You can specify the following plot-requests and their options:

ALL produces all relevant plots.

COEFFICIENTS | COEFFICIENTPANEL < (UNPACK | UNPACKPANEL)> creates a panel of two plots. The upper plot shows the progression of the parameter values as the selection process proceeds. The lower plot shows the progression of the CHOOSE= criterion. If no CHOOSE= criterion is in effect, then only the coefficient progression is shown. If you specify the UNPACK option, then the coefficient progression and the CHOOSE= criterion progression are shown in separate plots.

CRITERIA | CRITERIONPANEL < (UNPACK | UNPACKPANEL)> creates a panel of model fit criteria. If you specify the UNPACK option, then each criterion progression is shown in a separate plot.

FITBYROLE plots the progression of a fit statistic on the training, test, and validation data. The fit statistic that is displayed depends on the type of model that is being fit. The fit by role plot is not produced if you do not specify a PARTITION statement.

NONE suppress all selection process plots, even if you specify other plot options.

SELECTION=method < (method-options)> is an alias for the METHOD= option.

STOPHORIZON=n specifies the number of consecutive steps at which the STOP= criterion must worsen in order for a local extremum to be detected. For example, suppose that STOP=AIC and the sequence of AIC values at steps 1 to 6 of a selection are 10, 7, 4, 6, 5, 2. If STOPHORIZON=2, then the AIC criterion is deemed to have a local minimum at step 3 because the AIC value at the next two steps are greater than the value 4 that occurs at step 3. However, if STOPHORIZON=3, then the value at step 3 is not deemed to be a local minimum because the AIC value at step 6 is lower than the AIC value at step 3. If you specify STOP=NONE then the stop horizon value is ignored. If you specify STOP=SL, METHOD=LAR, or METHOD=LASSO, then n is ignored and STOPHORIZON=1 is used. By default, STOPHORIZON=3.
Optimization Options

This section describes options that are typically available in the PROC statement of the procedures in this book that perform optimizations. The following notation is used to describe the options. $\beta$ denotes the $p \times 1$ vector of parameters for the optimization and $\beta_i$ is its $i$th element. The objective function being minimized, its $p \times 1$ gradient vector, and its $p \times p$ Hessian matrix are denoted as $f(\beta)$, $g(\beta)$, and $H(\beta)$, respectively. The gradient with respect to the $i$th parameter is denoted as $g_i(\beta)$. Superscripts in parentheses denote the iteration count; for example, $f(\beta)^{(k)}$ is the value of the objective function at iteration $k$.

**ABSCONV=r**

specifies an absolute function convergence criterion. For minimization, termination requires $f(\beta^{(k)}) \leq r$, where $\beta$ is the vector of parameters in the optimization and $f(\cdot)$ is the objective function. The default value of $r$ is the negative square root of the largest double-precision value, which serves only as a protection against overflows.

**ABSFCONV=r < n>**

specifies an absolute function difference convergence criterion. For all techniques except NMSIMP, termination requires a small change of the function value in successive iterations:

$$|f(\beta^{(k-1)}) - f(\beta^{(k)})| \leq r$$

Here, $\beta$ is the vector of parameters in the optimization and $f(\cdot)$ is the objective function. The same formula is used for the NMSIMP technique, but $\beta^{(k)}$ is defined as the vertex that has the lowest function value and $\beta^{(k-1)}$ is defined as the vertex that has the highest function value in the simplex. PROC NLMOD provides an optional integer value $n$, which specifies the number of successive iterations for which the criterion must be satisfied before the process can be terminated. By default, ABSFCONV=0.

**ABSGCONV=r < n>**

specifies an absolute gradient convergence criterion. Termination requires the maximum absolute gradient element to be small:

$$\max_j |g_j(\beta^{(k)})| \leq r$$

Here, $\beta$ is the vector of parameters in the optimization and $g_j(\cdot)$ is the gradient of the objective function with respect to the $j$th parameter. This criterion is not used by the NMSIMP technique. PROC NLMOD provides an optional integer value $n$, which specifies the number of successive iterations for which the criterion must be satisfied before the process can be terminated. By default, ABSGCONV=1E–5.
ABSXCONV=r < n>

ABSXTOL=r < n>

specifies an absolute parameter convergence criterion: For all techniques except NMSIMP, termination requires a small Euclidean distance between successive parameter vectors,

\[ \| \beta^{(k)} - \beta^{(k-1)} \|_2 \leq r \]

For the NMSIMP technique, termination requires either a small length \( \alpha^{(k)} \) of the vertices of a restart simplex,

\[ \alpha^{(k)} \leq r \]

or a small simplex size,

\[ \delta^{(k)} \leq r \]

where the simplex size \( \delta^{(k)} \) is defined as the L1 distance from the simplex vertex \( \xi^{(k)} \) that has the smallest function value to the other \( p \) simplex points \( \beta_l^{(k)} \neq \xi^{(k)} \):

\[ \delta^{(k)} = \sum_{\beta_l \neq \xi} \| \beta_l^{(k)} - \xi^{(k)} \|_1 \]

PROC NLMOD provides an optional integer value \( n \), which specifies the number of successive iterations for which the criterion must be satisfied before the process can terminate. The default is \( r = 1E-8 \) for the NMSIMP technique and \( r = 0 \) otherwise.

FCONV=r < n>

FTOL=r < n>

specifies a relative function difference convergence criterion. For all techniques except NMSIMP, termination requires a small relative change of the function value in successive iterations,

\[ \frac{|f(\beta^{(k)}) - f(\beta^{(k-1)})|}{|f(\beta^{(k-1)})|} \leq r \]

Here, \( \beta \) denotes the vector of parameters that participate in the optimization, and \( f(\cdot) \) is the objective function. The same formula is used for the NMSIMP technique, but \( \beta^{(k)} \) is defined as the vertex that has the lowest function value and \( \beta^{(k-1)} \) is defined as the vertex that has the highest function value in the simplex.

PROC NLMOD provides an optional integer value \( n \), which specifies the number of successive iterations for which the criterion must be satisfied before the process can terminate. The default value is \( r = 2 \times \epsilon \) where \( \epsilon \) is the machine precision, which is the smallest double-precision floating-point number such that \( 1 + \epsilon > 1 \).

FCONV2=r < n>

FTOL2=r < n>

specifies a second function convergence criterion. For all techniques except NMSIMP, termination requires a small predicted reduction of the objective function:

\[ df^{(k)} = f(\beta^{(k)}) - f(\beta^{(k)} + s^{(k)}) \]
The predicted reduction
\[
df^{(k)} = -g^{(k)}' s^{(k)} - \frac{1}{2} s^{(k)}' H^{(k)} s^{(k)}
\]
\[
= -\frac{1}{2} s^{(k)}' g^{(k)} \leq r
\]
is computed by approximating the objective function \( f \) by the first two terms of the Taylor series and substituting the Newton step,
\[
s^{(k)} = -[H^{(k)}]^{-1} g^{(k)}
\]
For the NMSIMP technique, termination requires a small standard deviation of the function values of the \( p + 1 \) simplex vertices \( \beta_i^{(k)} \), \( l = 0, \ldots, p \),
\[
\sqrt{\frac{1}{n + 1} \sum_l \left[ f(\beta_i^{(k)}) - \bar{f}(\beta^{(k)}) \right]^2} \leq r
\]
where \( \bar{f}(\beta^{(k)}) = \frac{1}{p+1} \sum_l f(\beta_i^{(k)}) \). If there are \( p_{act} \) boundary constraints active at \( \beta^{(k)} \), the mean and standard deviation are computed only for the \( n + 1 - p_{act} \) unconstrained vertices.

PROC NLMD provides an optional integer value \( n \), which specifies the number of successive iterations for which the criterion must be satisfied before the process can terminate. The default value is \( r = 1E^{-6} \) for the NMSIMP technique and \( r = 0 \) otherwise.

**GCONV=** \( r < \ n > \)
**GTOL=** \( r < \ n > \)

specifies a relative gradient convergence criterion. For all techniques except CONGRA and NMSIMP, termination requires that the normalized predicted function reduction be small:
\[
\frac{g(\beta^{(k)})' [H^{(k)}]^{-1} g(\beta^{(k)})}{|f(\beta^{(k)})|} \leq r
\]
Here, \( \beta \) denotes the vector of parameters that participate in the optimization, \( f(\cdot) \) is the objective function, and \( g(\cdot) \) is the gradient. For the CONGRA technique (where a reliable Hessian estimate \( H \) is not available), the following criterion is used:
\[
\frac{\| g(\beta^{(k)}) \|_2^2}{\| g(\beta^{(k)}) - g(\beta^{(k-1)}) \|_2} \leq r
\]
This criterion is not used by the NMSIMP technique. PROC NLMD provides an optional integer value \( n \), which specifies the number of successive iterations for which the criterion must be satisfied before the process can terminate. By default, GCONV=1E–8.

**GCONV2=** \( r < \ n > \)
**GTOL2=** \( r < \ n > \)

specifies another relative gradient convergence criterion. For the TRUREG, LEVMAR, NRRIDG, and NEWRAP techniques, the following criterion of Browne (1982) is used:
\[
\max_j \frac{|g_j(\beta^{(k)})|}{\sqrt{f(\beta^{(k)}) H_j^{(k)}}} \leq r
\]
This criterion is not used by the other techniques.

PROC NLMOD provides an optional integer value \( n \), which specifies the number of successive iterations for which the criterion must be satisfied before the process can terminate. By default, GCONV2=0.

**MAXFUNC=**\( n \)

**MAXFU=**\( n \)
specifies the maximum number \( n \) of function calls in the optimization process. The default values are as follows, depending on the optimization technique:

- TRUREG, NRRIDG, and NEWRAP: 125
- QUANEW and DBLDOG: 500
- CONGRA: 1,000
- NMSIMP: 3,000

The optimization can terminate only after completing a full iteration. Therefore, the number of function calls that are actually performed can exceed the number that is specified by this option. You can specify the optimization technique in the `TECHNIQUE=` option.

**MAXITER=**\( n \)

**MAXIT=**\( n \)
specifies the maximum number \( n \) of iterations in the optimization process. The default values are as follows, depending on the optimization technique:

- TRUREG, NRRIDG, and NEWRAP: 50
- QUANEW and DBLDOG: 200
- CONGRA: 400
- NMSIMP: 1,000

These default values also apply when \( n \) is specified as a missing value. You can specify the optimization technique in the `TECHNIQUE=` option.

**MAXTIME=**\( r \)
specifies an upper limit of \( r \) seconds of CPU time for the optimization process. The time specified by \( r \) is checked only once at the end of each iteration. Therefore, the actual running time can be longer than \( r \). The default value is the largest floating-point double representation of your computer.

**MINITER=**\( n \)

**MINIT=**\( n \)
specifies the minimum number of iterations. If you request more iterations than are actually needed for convergence to a stationary point, the optimization algorithms can behave strangely. For example, the effect of rounding errors can prevent the algorithm from continuing for the required number of iterations. By default, MINITER=0.
NORMALIZE=YES | NO

specifies whether the objective function should be normalized during the optimization by the reciprocal of the used frequency count. This option affects the values reported in the “Iteration History” table. The results reported in the “Fit Statistics” are always displayed for the nonnormalized log-likelihood function. The default is to normalize the objective function.

TECHNIQUE=technique

TECH=technique

specifies the optimization technique for obtaining maximum likelihood estimates. You can specify one of the following techniques:

- **CONGRA**: performs a conjugate-gradient optimization.
- **DBLDOG**: performs a version of double-dogleg optimization.
- **LEVMAR**: performs a Levenberg-Marquardt nonlinear least-squares minimization. This technique is available only with PROC NLMOD.
- **NEWRAP**: performs a Newton-Raphson optimization with line search.
- **NMSIMP**: performs a Nelder-Mead simplex optimization.
- **NONE**: performs no optimization.
- **NRRIDG**: performs a Newton-Raphson optimization with ridging.
- **QUANEW**: performs a dual quasi-Newton optimization.
- **TRUREG**: performs a trust-region optimization.

By default, TECHNIQUE=NRRIDG.

For more information, see the section “Choosing an Optimization Algorithm” on page 82.

XCONV=r

XTOL=r

specifies the relative parameter convergence criterion. Convergence requires a small relative parameter change in subsequent iterations,

\[
\max_j |\delta_j^{(i)}| < r
\]

where

\[
\delta_j^{(i)} = \begin{cases} 
\frac{\hat{\beta}_j^{(i)} - \hat{\beta}_j^{(i-1)}}{\hat{\beta}_j^{(i-1)} - \hat{\beta}_j^{(k-1)}} & |\beta_j^{(i-1)}| < 0.01 \\
\frac{\hat{\beta}_j^{(i)} - \hat{\beta}_j^{(k-1)}}{\hat{\beta}_j^{(i-1)}} & \text{otherwise} 
\end{cases}
\]

and \(\hat{\beta}_j^{(i)}\) is the estimate of the \(j\)th parameter at iteration \(i\). For the NMSIMP technique, the same formula is used, but \(\beta_j^{(k)}\) is defined as the vertex that has the lowest function value and \(\beta_j^{(k-1)}\) is defined as the vertex that has the highest function value in the simplex. The default value is \(r = 1E-8\) for the NMSIMP technique and \(r = 0\) otherwise.
Levelization of Classification Variables

A classification variable enters the statistical analysis or model not through its values but through its levels. The process of associating values of a variable with levels is called *levelization*.

During the process of levelization, observations that share the same value are assigned to the same level. The manner in which values are grouped can be affected by the inclusion of formats. The sort order of the levels can be determined by specifying the ORDER= option in the procedure statement. In procedures in this book, you can also control the sorting order separately for each variable in the CLASS statement.

Consider the data on nine observations in Table 3.9. The variable A is integer-valued, and the variable X is a continuous variable that has a missing value for the fourth observation. The fourth and fifth columns of Table 3.9 apply two different formats to the variable X.

<table>
<thead>
<tr>
<th>Obs</th>
<th>A</th>
<th>x</th>
<th>FORMAT x 3.0</th>
<th>FORMAT x 3.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>1.09</td>
<td>1</td>
<td>1.1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1.13</td>
<td>1</td>
<td>1.1</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>1.27</td>
<td>1</td>
<td>1.3</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>2.26</td>
<td>2</td>
<td>2.3</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>2.48</td>
<td>2</td>
<td>2.5</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td>3.34</td>
<td>3</td>
<td>3.3</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>3.34</td>
<td>3</td>
<td>3.3</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>3.14</td>
<td>3</td>
<td>3.1</td>
</tr>
</tbody>
</table>

By default, levelization of the variables groups the observations by the formatted value of the variable, except for numerical variables for which no explicit format is provided. Numerical variables for which no explicit format is provided are sorted by their internal value. The levelization of the four columns in Table 3.9 leads to the level assignment in Table 3.10.

<table>
<thead>
<tr>
<th>Obs</th>
<th>A Value</th>
<th>X Value</th>
<th>FORMAT x 3.0 Value</th>
<th>FORMAT x 3.1 Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>1.09</td>
<td>1</td>
<td>1.1</td>
</tr>
</tbody>
</table>
Chapter 3: Shared Concepts

Table 3.10  continued

<table>
<thead>
<tr>
<th>Obs</th>
<th>A Value Level</th>
<th>X Value Level</th>
<th>FORMAT x 3.0 Value Level</th>
<th>FORMAT x 3.1 Value Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2</td>
<td>1.13</td>
<td>1</td>
<td>1.1</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>1.27</td>
<td>1</td>
<td>1.3</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>2.26</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>2.48</td>
<td>2</td>
<td>2.5</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>3.34</td>
<td>3</td>
<td>3.3</td>
</tr>
<tr>
<td>7</td>
<td>3</td>
<td>3.34</td>
<td>3</td>
<td>3.3</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>3.14</td>
<td>3</td>
<td>3.1</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>3.14</td>
<td>3</td>
<td>3.1</td>
</tr>
</tbody>
</table>

The sort order for the levels of CLASS variables can be specified in the ORDER= option in the CLASS statement.

When ORDER=FORMATTED (which is the default) is in effect for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values. To order numeric classification levels that have no explicit format by their BEST12. formatted values, you can specify the BEST12. format explicitly for the CLASS variables.

Table 3.11 shows how values of the ORDER= option are interpreted.

Table 3.11  Interpretation of Values of ORDER= Option

<table>
<thead>
<tr>
<th>Value of ORDER=</th>
<th>Levels Sorted By</th>
</tr>
</thead>
<tbody>
<tr>
<td>FORMATTED</td>
<td>External formatted value, except for numeric variables that have no explicit format, which are sorted by their unformatted (internal) value. The sort order is machine-dependent.</td>
</tr>
<tr>
<td>FREQ</td>
<td>Descending frequency count (levels that have the most observations come first in the order)</td>
</tr>
<tr>
<td>INTERNAL</td>
<td>Unformatted value. The sort order is machine-dependent.</td>
</tr>
</tbody>
</table>

For more information about sort order, see the chapter about the SORT procedure in the Base SAS Procedures Guide and the discussion of BY-group processing in SAS Language Reference: Concepts.

When the MISSING option is specified in the CLASS statement, the missing values (‘.’ for a numeric variable and blanks for a character variable) are included in the levelization and are assigned a level. Table 3.12 displays the results of levelizing the values in Table 3.9 when the MISSING option is in effect.

Table 3.12  Values and Levels When the MISSING Option Is Specified

<table>
<thead>
<tr>
<th>Obs</th>
<th>A Value Level</th>
<th>X Value Level</th>
<th>FORMAT x 3.0 Value Level</th>
<th>FORMAT x 3.1 Value Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>1.09</td>
<td>1</td>
<td>1.1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1.09</td>
<td>2</td>
<td>1.1</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>1.27</td>
<td>1</td>
<td>1.3</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>2.26</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>2.48</td>
<td>2</td>
<td>2.5</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>3.34</td>
<td>3</td>
<td>3.3</td>
</tr>
<tr>
<td>7</td>
<td>3</td>
<td>3.34</td>
<td>3</td>
<td>3.3</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>3.14</td>
<td>3</td>
<td>3.1</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>3.14</td>
<td>3</td>
<td>3.1</td>
</tr>
</tbody>
</table>
Table 3.12  continued

<table>
<thead>
<tr>
<th>Obs</th>
<th>Value</th>
<th>A Level</th>
<th>X Level</th>
<th>FORMAT x 3.0 Value Level</th>
<th>FORMAT x 3.1 Value Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.13</td>
<td>1</td>
<td>1.1</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1.27</td>
<td>1</td>
<td>1.3</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>2</td>
<td>.</td>
<td>1</td>
<td>.</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>2</td>
<td>2.26</td>
<td>2</td>
<td>2.3</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>2</td>
<td>2.48</td>
<td>2</td>
<td>2.5</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td>3</td>
<td>3.34</td>
<td>3</td>
<td>3.3</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>3</td>
<td>3.34</td>
<td>3</td>
<td>3.3</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>3</td>
<td>3.14</td>
<td>3</td>
<td>3.1</td>
</tr>
</tbody>
</table>

When the MISSING option is not specified, it is important to understand the implications of missing values for your statistical analysis. When a procedure in this book levelizes the CLASS variables, an observation for which any CLASS variable has a missing value is excluded from the analysis.

Procedures in this book print a “Number of Observations” table that shows the number of observations that are read from the data set and the number of observations that are used in the analysis. Pay careful attention to this table—especially when your data table contains missing values—to ensure that no observations are unintentionally excluded from the analysis.

Specification and Parameterization of Model Effects

This section applies to the following procedures: GAMMOD, GENSELECT, LMIXED, LOGSELECT, MODELMATRIX, NLMOD, PHSELECT, PLSMOD, QTRSELECT, and REGSELECT.

Procedures in this book that have a MODEL statement support the formation of effects. An effect is an element in a linear model structure that is formed from one or more variables. At some point the statistical representations of these models involve linear structures such as

\[ X\beta \]

or

\[ X\beta + Z\gamma \]

The model matrices \( X \) and \( Z \) are formed according to effect-construction rules.

Procedures that also have a CLASS statement support the rich set of effects that is discussed in this section.

Procedures that also have an EFFECT statement enable you to construct special constructed effects that are discussed in “EFFECT Statement” on page 21.

In order to correctly interpret the results from a statistical analysis, you need to understand how construction (parameterization) rules apply to regression-type models, whether these are linear models as in the REGSELECT procedure or generalized linear models as in the LOGSELECT and GENSELECT procedures.
Effects are specified by a special notation that uses variable names and operators. There are two types of variables: classification (or CLASS) variables and continuous variables. Classification variables can be either numeric or character and are specified in a CLASS statement. For more information, see the section “Levelization of Classification Variables” on page 49. An independent variable that is not declared in the CLASS statement is assumed to be continuous. For example, the heights and weights of subjects are continuous variables.

Two primary operators (crossing and nesting) are used for combining the variables, and several additional operators are used to simplify effect specification. Operators are discussed in the section “Effect Operators” on page 52.

Procedures in this book that have a CLASS statement support a general linear model (GLM) parameterization and might also support nonsingular parameterizations for the classification variables. The GLM parameterization, commonly called dummy parameterization, is the default for all procedures in this book. For more information, see the sections “GLM Parameterization of Classification Variables and Effects” on page 54 and “Nonsingular Parameterization” on page 58.

**Effect Operators**

Table 3.13 summarizes the operators that are available for selecting and constructing effects. These operators are discussed in the following sections.

<table>
<thead>
<tr>
<th>Operator</th>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interaction</td>
<td>A*B</td>
<td>Crosses the levels of the effects</td>
</tr>
<tr>
<td>Nesting</td>
<td>A(B)</td>
<td>Nests A levels within B levels</td>
</tr>
<tr>
<td>Bar operator</td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>At sign operator</td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>Dash operator</td>
<td>A1-A10</td>
<td>Specifies sequentially numbered variables</td>
</tr>
<tr>
<td>Colon operator</td>
<td>A:</td>
<td>Specifies variables with common prefix</td>
</tr>
<tr>
<td>Double dash operator</td>
<td>A- -C</td>
<td>Specifies sequential variables in data set order</td>
</tr>
</tbody>
</table>

**Bar and At Sign Operators**

You can shorten the specification of a large factorial model by using the bar operator. For example, two ways of writing the model for a full three-way factorial model follow:

```
model Y = A B C A*B A*C B*C A*B*C;
```

```
model Y = A|B|C;
```

When the bar (|) is used, the right and left sides become effects, and the cross of them becomes an effect. Multiple bars are permitted. The expressions are expanded from left to right, using rules 2–4 in Searle (1971, p. 390).

- Multiple bars are evaluated from left to right. For example, A | B | C is evaluated as follows:
Crossed and nested groups of variables are combined. For example, \(A(\text{B)(C)}\) generates \(A^*\text{C(B D)}\), among other terms.

- Duplicate variables are removed. For example, \(A(\text{C)(B)}\) generates \(A^*\text{B(C C)}\), among other terms, and the extra C is removed.

- Effects are discarded if a variable occurs on both the crossed and nested parts of an effect. For example, \(A(\text{B)(D E)}\) generates \(A^*\text{B(D E)}\), but this effect is eliminated immediately.

You can also specify the maximum number of variables involved in any effect that results from bar evaluation by specifying that maximum number, preceded by an at sign (@), at the end of the bar effect. For example, the following specification selects only those effects that contain two or fewer variables:

\[
\text{model}\ Y = A|B|C@2;
\]
The preceding example is equivalent to specifying the following MODEL statement:

\[
\text{model}\ Y = A\ B\ C\ A^*B\ A^*C\ B*C;
\]

More examples of using the bar and at operators follow:

- \(A\ C(B)\) is equivalent to \(A\ C(B)\ A^*C(B)\)
- \(A(\text{B)(C)}\) is equivalent to \(A(\text{B)(C)}\ A^*C(B)\)
- \(A(\text{B)(D E)}\) is equivalent to \(A(\text{B)(D E)}\)
- \(A|\text{B(A)}|C\) is equivalent to \(A\ B(\text{A)}\ C\ A^*C\ B^*C(A)\)
- \(A|\text{B(A)}|C@2\) is equivalent to \(A\ B(\text{A)}\ C\ A^*C\)
- \(A|\text{B(C)(D)}@2\) is equivalent to \(A\ B\ A^*B\ C\ A^*C\ B^*C\ D\ A^*D\ B^*D\ C^*D\)
- \(A^*B(C^*D)\) is equivalent to \(A^*B(C\ D)\)

**Colon, Dash, and Double Dash Operators**

You can simplify the specification of a large model when some of your variables have a common prefix by using the colon (:) operator and the dash (-) operator. The dash operator enables you to list variables that are numbered sequentially, and the colon operator selects all variables with a given prefix. For example, if your data set contains the variables X1 through X9, the following MODEL statements are equivalent:

\[
\text{model}\ Y = X1\ X2\ X3\ X4\ X5\ X6\ X7\ X8\ X9;
\]

\[
\text{model}\ Y = X1\ -X9;
\]

\[
\text{model}\ Y = X:\ ;
\]

If your data set contains only the three covariates X1, X2, and X9, then the colon operator selects all three variables:
model Y = X1-X9;

The double dash (--) operator enables you to select variables that are stored sequentially in the SAS data set, whether or not they have a common prefix. You can use the CONTENTS procedure (see Base SAS Procedures Guide) to determine your variable ordering. For example, if you replace the dash in the preceding MODEL statement with a double dash, as follows, then all three variables are selected:

model Y = X1--X9;

If your data set contains the variables A, B, and C, then you can use the double dash operator to select these variables by specifying the following:

model Y = A--C;

### GLM Parameterization of Classification Variables and Effects

Table 3.14 shows the types of effects that are available in procedures in this book; they are discussed in more detail in the following subsections. Let A, B, and C represent classification variables, and let X and Z represent continuous variables.

<table>
<thead>
<tr>
<th>Effect</th>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>Default</td>
<td>Intercept (unless the NOINT option is specified)</td>
</tr>
<tr>
<td>Regression</td>
<td>X Z</td>
<td>Continuous variables</td>
</tr>
<tr>
<td>Polynomial</td>
<td>X*Z</td>
<td>Interaction of continuous variables</td>
</tr>
<tr>
<td>Main</td>
<td>A B</td>
<td>CLASS variables</td>
</tr>
<tr>
<td>Interaction</td>
<td>A*B</td>
<td>Crossing of CLASS variables</td>
</tr>
<tr>
<td>Nested</td>
<td>A(B)</td>
<td>Main effect A nested within CLASS effect B</td>
</tr>
<tr>
<td>Continuous-by-class</td>
<td>X*A</td>
<td>Crossing of continuous and CLASS variables</td>
</tr>
<tr>
<td>Continuous-nesting-class</td>
<td>X(A)</td>
<td>Continuous variable X1 nested within CLASS variable A</td>
</tr>
<tr>
<td>General</td>
<td>X<em>Z</em>A(B)</td>
<td>Combinations of different types of effects</td>
</tr>
</tbody>
</table>

Table 3.15 shows some examples of MODEL statements that use various types of effects.

<table>
<thead>
<tr>
<th>Specification</th>
<th>Type of Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>model Y=X;</td>
<td>Simple regression</td>
</tr>
<tr>
<td>model Y=X Z;</td>
<td>Multiple regression</td>
</tr>
<tr>
<td>model Y=X X*X;</td>
<td>Polynomial regression</td>
</tr>
<tr>
<td>model Y=A;</td>
<td>One-way analysis of variance (ANOVA)</td>
</tr>
<tr>
<td>model Y=A B C;</td>
<td>Main-effects ANOVA</td>
</tr>
<tr>
<td>model Y=A B A*B;</td>
<td>Factorial ANOVA with interaction</td>
</tr>
<tr>
<td>model y=A B(A) C(B A);</td>
<td>Nested ANOVA</td>
</tr>
</tbody>
</table>
Intercept
By default, linear models that are created by procedures in this book automatically include a column of 1s in \( X \). This column corresponds to an intercept parameter. In many procedures, you can use the NOINT option in the MODEL statement to suppress this intercept. For example, the NOINT option is useful when the MODEL statement contains a classification effect and you want the parameter estimates to be in terms of the mean response for each level of that effect.

Regression Effects
Numeric variables or polynomial terms that involve them can be included in the model as regression effects (covariates). The actual values of such terms are included as columns of the relevant model matrices. You can use the bar operator along with a regression effect to generate polynomial effects. For example, \( X \mid X \mid X \) expands to \( X X X X X X X \), which is a cubic model.

Main Effects
If a classification variable has \( m \) levels, the GLM parameterization generates \( m \) columns for its main effect in the model matrix. Each column is an indicator variable for a particular level. The order of the columns is the sort order of the values of their levels and can be controlled by the ORDER= option in the CLASS statement.

Table 3.16 is an example where \( \beta_0 \) denotes the intercept and A and B are classification variables that have two and three levels, respectively.

的主要内容：
- **Table 3.15 continued**

<table>
<thead>
<tr>
<th>Specification</th>
<th>Type of Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>model Y=A X;</td>
<td>Analysis of covariance (ANCOVA)</td>
</tr>
<tr>
<td>model Y=A X(A);</td>
<td>Separate-slopes regression</td>
</tr>
<tr>
<td>model Y=A X X+A;</td>
<td>Homogeneity-of-slopes regression</td>
</tr>
</tbody>
</table>

**Intercept**
由默认，线性模型在该书中创建的程序自动包括一个1的列在X。此列对应于一个截距参数。在许多程序中，您可以在MODEL语句中使用NOINT选项来抑制此截距。例如，NOINT选项在MODEL语句包含分类效果时很有用，您希望参数估计以每个级别的均值响应来表示。

**回归效果**
数值变量或涉及它们的多项式项可以作为回归效果（协变量）包含在模型中。这些项的实际值作为相关模型矩阵的列。您可以使用bar运算符与回归效果结合使用以生成多项式效果。例如，\( X \mid X \mid X \) 扩展为 \( X X X X X X X \)，这是一个立方模型。

**主效果**
如果分类变量有 \( m \) 个级别，GLM参数化为模型中的主效应生成 \( m \) 列。每一列是一个指示变量，表示特定级别。列的顺序是它们级别值的排序，可以由CLASS语句中的ORDER=选项控制。

表3.16是一个例子，其中 \( \beta_0 \) 表示截距，A和B是分类变量，分别有两个和三个级别。

<table>
<thead>
<tr>
<th>Data</th>
<th>I</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>( \beta_0 )</td>
<td>A1</td>
</tr>
<tr>
<td>1 1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1 2</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1 3</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2 1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2 2</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2 3</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

通常这些效果的列数比估计它们的自由度要多。换句话说，GLM参数化的主效果是*singular*。
**Interaction Effects**

Often a model includes interaction (crossed) effects to account for how the effect of a variable changes along with the values of other variables. With an interaction, the terms are first reordered to correspond to the order of the variables in the **CLASS** statement. Thus, B*A becomes A*B if A precedes B in the **CLASS** statement. Then, the **GLM** parameterization generates columns for all combinations of levels that occur in the data. The order of the columns is such that the rightmost variables in the interaction change faster than the leftmost variables (Table 3.17).

**Table 3.17** Example of Interaction Effects

<table>
<thead>
<tr>
<th>Data</th>
<th>I</th>
<th>A</th>
<th>B</th>
<th>A*B</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>β₀</td>
<td>A1</td>
<td>A2</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

The preceding matrix, main-effects columns are not linearly independent of crossed-effects columns. In fact, the column space for the crossed effects contains the space of the main effect.

When your model contains many interaction effects, you might be able to code them more parsimoniously by using the **bar operator** (|). The bar operator generates all possible interaction effects. For example, A|B|C expands to A B A*B C A*C B*C A*B*C. To eliminate higher-order interaction effects, use the **at sign** (@) in conjunction with the bar operator. For example, A|B|C|D@2 expands to A B A*B C A*C B*C D A*D B*D C*D.

**Nested Effects**

Nested effects are generated in the same manner as crossed effects. Hence, the design columns that are generated by the following two statements are the same (but the ordering of the columns is different):

```plaintext
model Y=A B(A);
model Y=A A*B;
```

The nesting operator in procedures in this book is more of a notational convenience than an operation that is distinct from crossing. Nested effects are typically characterized by the property that the nested variables do not appear as main effects. The order of the variables within nesting parentheses is made to correspond to the order of these variables in the **CLASS** statement. The order of the columns is such that variables outside the parentheses index faster than those inside the parentheses, and the rightmost nested variables index faster than the leftmost variables (Table 3.18).

**Table 3.18** Example of Nested Effects

<table>
<thead>
<tr>
<th>Data</th>
<th>I</th>
<th>A</th>
<th>B(A)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>β₀</td>
<td>A1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Table 3.18 continued

<table>
<thead>
<tr>
<th>Data</th>
<th>I</th>
<th>A</th>
<th>B(A)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Continuous-Nesting-Class Effects

When a continuous variable nests or crosses with a classification variable, the design columns are constructed by multiplying the continuous values into the design columns for the classification effect (Table 3.19).

Table 3.19 Example of Continuous-Nesting-Class Effects

<table>
<thead>
<tr>
<th>Data</th>
<th>I</th>
<th>A</th>
<th>X</th>
<th>β₀</th>
<th>A1</th>
<th>A2</th>
<th>X(A1)</th>
<th>X(A2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>21</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>21</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>24</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>22</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>28</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>23</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This model estimates a separate intercept and a separate slope for X within each level of A.

Continuous-by-Class Effects

Continuous-by-class effects generate the same design columns as continuous-nesting-class effects. Table 3.20 shows the construction of the X*A effect. The two columns for this effect are the same as the columns for the X(A) effect in Table 3.19.

Table 3.20 Example of Continuous-by-Class Effects

<table>
<thead>
<tr>
<th>Data</th>
<th>I</th>
<th>X</th>
<th>A</th>
<th>X*A</th>
</tr>
</thead>
<tbody>
<tr>
<td>21</td>
<td>1</td>
<td>21</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>24</td>
<td>1</td>
<td>24</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>22</td>
<td>1</td>
<td>22</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>28</td>
<td>2</td>
<td>28</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>19</td>
<td>2</td>
<td>19</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>23</td>
<td>2</td>
<td>23</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

You can use continuous-by-class effects together with pure continuous effects to test for homogeneity of slopes.
General Effects
An example that combines all the effects is $X_1 \times X_2 \times A \times B \times C(D E)$. The continuous list comes first, followed by the crossed list, followed by the nested list in parentheses.

Effects might be renamed by the procedure to correspond to ordering rules. For example, $B \times A(E D)$ might be renamed $A \times B(D E)$ to satisfy the following:

- Classification variables that occur outside parentheses (crossed effects) are sorted in the order in which they appear in the CLASS statement.
- Variables within parentheses (nested effects) are sorted in the order in which they appear in the CLASS statement.

The sequencing of the parameters that are generated by an effect is determined by the variables whose levels are indexed faster:

- Variables in the crossed list index faster than variables in the nested list.
- Within a crossed or nested list, variables to the right index faster than variables to the left.

For example, suppose that a model includes four effects—$A$, $B$, $C$, and $D$—each having two levels, 1 and 2, and that the CLASS statement is as follows:

```
class A B C D;
```

Then the order of the parameters for the effect $B \times A(C D)$, which is renamed $A \times B(C D)$, is as follows:

- $A_1 B_1 C_1 D_1 \rightarrow A_1 B_2 C_1 D_1 \rightarrow A_2 B_1 C_1 D_1 \rightarrow A_2 B_2 C_1 D_1 \rightarrow$
- $A_1 B_1 C_2 D_1 \rightarrow A_1 B_2 C_2 D_1 \rightarrow A_2 B_1 C_2 D_1 \rightarrow A_2 B_2 C_2 D_1 \rightarrow$
- $A_1 B_1 C_2 D_2 \rightarrow A_1 B_2 C_2 D_2 \rightarrow A_2 B_1 C_2 D_2 \rightarrow A_2 B_2 C_2 D_2$

Note that first the crossed effects $B$ and $A$ are sorted in the order in which they appear in the CLASS statement so that $A$ precedes $B$ in the parameter list. Then, for each combination of the nested effects in turn, combinations of $A$ and $B$ appear. The $B$ effect changes fastest because it is rightmost in the cross list. Then $A$ changes next fastest, and $D$ changes next fastest. The $C$ effect changes most slowly because it is leftmost in the nested list.

Nonsingular Parameterization
When a parameterization of main effects provides the same number of columns for the effects as there are degrees of freedom to estimate them, the parameterization is called nonsingular. A variety of nonsingular parameterizations for classification effects are available for many procedures in this book. In most of these procedures you use the PARAM= option in the CLASS statement to specify the parameterization.

Consider a model with one CLASS variable $A$ that has four levels, 1, 2, 5, and 7. Details of the possible choices for the PARAM= option follow.
Three columns are created to indicate group membership of the nonreference levels. For the reference level, all three dummy variables have a value of 0. For example, if the reference level is 7 (REF=7), the design matrix columns for A are as follows.

<table>
<thead>
<tr>
<th>Reference Coding</th>
<th>Design Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
</tr>
</tbody>
</table>

Parameter estimates of CLASS main effects that use the reference coding scheme estimate the difference in the effect of each nonreference level compared to the effect of the reference level.

**EFFECT**

Three columns are created to indicate group membership of the nonreference levels. For the reference level, all three dummy variables have a value of –1. For example, if the reference level is 7 (REF=7), the design matrix columns for A are as follows.

<table>
<thead>
<tr>
<th>Effect Coding</th>
<th>Design Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>–1</td>
</tr>
</tbody>
</table>

Parameter estimates of CLASS main effects that use the effect coding scheme estimate the difference in the effect of each nonreference level compared to the average effect over all four levels.

**ORDINAL | THERMOMETER**

Three columns are created to indicate group membership of the higher levels of the effect. For the first level of the effect (which for A is 1), all three dummy variables have a value of 0. The design matrix columns for A are as follows.

<table>
<thead>
<tr>
<th>Ordinal Coding</th>
<th>Design Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
</tr>
</tbody>
</table>

The first level of the effect is a control or baseline level. Parameter estimates of CLASS main effects, using the ORDINAL coding scheme, estimate the differences
between effects of successive levels. When the parameters have the same sign, the effect is monotonic across the levels.

**POLYNOMIAL | POLY**  
Three columns are created. The first represents the linear term \(x\), the second represents the quadratic term \(x^2\), and the third represents the cubic term \(x^3\), where \(x\) is the level value. If the CLASS levels are not numeric, they are translated into 1, 2, 3, \ldots according to their sort order. The design matrix columns for \(A\) are as follows.

<table>
<thead>
<tr>
<th>Polynomial Coding</th>
<th>Design Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>(A)</td>
<td>(A) POLY1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2 4 8</td>
</tr>
<tr>
<td>5</td>
<td>5 25 125</td>
</tr>
<tr>
<td>7</td>
<td>7 49 343</td>
</tr>
</tbody>
</table>

**ORTHEFFECT**  
The columns are obtained by applying the Gram-Schmidt orthogonalization to the columns for PARAM=EFFECT. The design matrix columns for \(A\) are as follows.

<table>
<thead>
<tr>
<th>Orthogonal Effect Coding</th>
<th>Design Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>(A)</td>
<td>AOEFF1 AOEFF2 AOEFF3</td>
</tr>
<tr>
<td>1</td>
<td>1.41421 –0.81650 –0.57735</td>
</tr>
<tr>
<td>2</td>
<td>0 1.63299 –0.57735</td>
</tr>
<tr>
<td>5</td>
<td>0 0 1.73205</td>
</tr>
<tr>
<td>7</td>
<td>–1.41421 –0.81649 –0.57735</td>
</tr>
</tbody>
</table>

**ORTHORDINAL | ORTHOTHERM**  
The columns are obtained by applying the Gram-Schmidt orthogonalization to the columns for PARAM=ORDINAL. The design matrix columns for \(A\) are as follows.

<table>
<thead>
<tr>
<th>Orthogonal Ordinal Coding</th>
<th>Design Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>(A)</td>
<td>AOORD1 AOORD2 AOORD3</td>
</tr>
<tr>
<td>1</td>
<td>–1.73205 0 0</td>
</tr>
<tr>
<td>2</td>
<td>0.57735 –1.63299 0</td>
</tr>
<tr>
<td>5</td>
<td>0.57735 0.81650 –1.41421</td>
</tr>
<tr>
<td>7</td>
<td>0.57735 0.81650 1.41421</td>
</tr>
</tbody>
</table>

**ORTHPOLY**  
The columns are obtained by applying the Gram-Schmidt orthogonalization to the columns for PARAM=POLY. The design matrix columns for \(A\) are as follows.
Orthogonal Polynomial Coding

<table>
<thead>
<tr>
<th>Design Matrix</th>
<th>A</th>
<th>AOPOLY1</th>
<th>AOPOLY2</th>
<th>AOPOLY5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>-1.15311</td>
<td>0.90712</td>
<td>-0.92058</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>-0.73380</td>
<td>-0.54041</td>
<td>1.47292</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>0.52414</td>
<td>-1.37034</td>
<td>-0.92058</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>1.36277</td>
<td>1.00363</td>
<td>0.36823</td>
</tr>
</tbody>
</table>

ORTHREF

The columns are obtained by applying the Gram-Schmidt orthogonalization to the columns for PARAM=REFERENCE. The design matrix columns for A are as follows.

Orthogonal Reference Coding

<table>
<thead>
<tr>
<th>Design Matrix</th>
<th>A</th>
<th>AOREF1</th>
<th>AOREF2</th>
<th>AOREF3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1.73205</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>-0.57735</td>
<td>1.63299</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>-0.57735</td>
<td>-0.81650</td>
<td>1.41421</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>-0.57735</td>
<td>-0.81650</td>
<td>-1.41421</td>
</tr>
</tbody>
</table>

Class Variable Parameterization with Unbalanced Designs

Procedures in this book initially parameterize the CLASS variables by looking at the levels of the variables across the complete data set. If you have an unbalanced replication of levels across variables or BY groups, then the design matrix and the parameter interpretation might be different from what you expect. For example, suppose you have a model that has one CLASS variable A with three levels (1, 2, and 3), and another CLASS variable B with two levels (1 and 2). If the third level of A occurs only with the first level of B, if you use the EFFECT parameterization, and if your model contains the effect A(B) and an intercept, then the design for A within the second level of B is not a differential effect. In particular, the design looks like the following:

<table>
<thead>
<tr>
<th>Design Matrix</th>
<th>A(B=1)</th>
<th>A(B=2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>B A</td>
<td>A1</td>
<td>A2</td>
</tr>
<tr>
<td>1 1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1 2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1 3</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>2 1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2 2</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Procedures in this book detect linear dependency among the last two design variables and set the parameter for A2(B=2) to 0, resulting in an interpretation of these parameters as if they were reference- or dummy-coded. The REFERENCE or GLM parameterization might be more appropriate for such problems.
Model Selection Methods

The model selection methods are specified in the METHOD= option in the SELECTION statement. The following methods are available, although specific procedures might support only a subset of these methods. Furthermore, the examples in this section refer to fit criteria that might not be supported by a specific procedure.

Full Model Fitted

When METHOD=NONE, the complete model that is specified in the MODEL statement is used to fit the model, and no effect selection is done.

Forward Selection

This section applies to the following procedures: GENSELECT, LOGSELECT, PHSELECT, QTRSELECT, and REGSELECT.

METHOD=FORWARD specifies the forward selection technique, which begins with just the intercept and then sequentially adds the effect that most improves the fit. The process terminates when no significant improvement can be obtained by adding any effect.

In the traditional implementation of forward selection, the statistic that is used to determine whether to add an effect is the significance level of a hypothesis test that reflects an effect’s contribution to the model if it is included. At each step, the effect that is most significant is added. The process stops when the significance level for adding any effect is greater than some specified entry significance level.

An alternative approach to address the critical problem of when to stop the selection process is to assess the quality of the models that are produced by the forward selection method and choose the model from this sequence that “best” balances goodness of fit against model complexity. You can use several criteria for this purpose.

It is important to keep in mind that forward selection bases the decision about what effect to add at any step by considering models that differ by one effect from the current model. This search paradigm cannot guarantee reaching a “best” subset model. Furthermore, the add decision is greedy in the sense that the effect that is deemed most significant is the effect that is added. However, if your goal is to find a model that is best in terms of some selection criterion other than the significance level of the entering effect, then even this one step choice might not be optimal. For example, the effect that you would add to get a model that has the smallest value of the Mallows’ $C(p)$ statistic at the next step is not necessarily the same effect that is most significant based on a hypothesis test. You can specify the criterion to optimize at each step by using the SELECT= option. For example, the following statement requests that at each step the effect that is added be the one that produces a model that has the smallest value of the Mallows’ $C(p)$ statistic:

```
  selection method=forward(select=CP);
```

When all effects are variables (that is, effects have one degree of freedom and no hierarchy), using ADJRSQ, AIC, AICC, BIC, CP, RSQUARE, or SBC as the selection criterion for forward selection produces the same sequence of additions. However, if the degrees of freedom contributed by different effects are not constant
or if an out-of-sample prediction-based criterion is used, then different sequences of additions might be obtained.

If you specify only the SELECT= criterion, then this criterion is also used to decide when to stop the selection process. In the previous example, not only do effects enter based on the Mallows’ $C(p)$ statistic, but the selection terminates when the $C(p)$ statistic has a local minimum.

You use the CHOOSE= option to specify the criterion for selecting one model from the sequence of models produced. If you do not specify a CHOOSE= criterion, then the model at the final step is the selected model. For example, if you specify the following statement, then forward selection terminates at the step where no effect can be added at the 0.2 significance level:

```
   selection method=forward(select=SL choose=AIC SLE=0.2);
```

However, the selected model is the first one that has the minimum value of Akaike’s information criterion. In some cases, this minimum value might occur at a step much earlier than the final step. In other cases, the AIC might start increasing only if more steps are performed—that is, a larger value is used for the significance level for entry. If you want to minimize AIC, then too many steps are performed in the former case and too few in the latter case. To address this issue, you can use the STOP= option to specify a stopping criterion. When you specify a stopping criterion, forward selection continues until a local extremum of the stopping criterion in the sequence of models generated is reached. To be deemed a local extremum, a criterion value at a particular step must be better than its value at the next $n$ steps, where $n$ is known as the “stop horizon.” By default, the stop horizon is three steps, but you can change this by specifying the STOPHORIZON= option.

For example, if you specify the following statement, then forward selection terminates at the step where the effect to be added at the next step would produce a model that has an AIC statistic larger than the AIC statistic of the current model:

```
   selection method=forward(select=SL stop=AIC) stophorizon=1;
```

In most cases, provided that the entry significance level is large enough that the local extremum of the named criterion occurs before the final step, specifying either of the following statements selects the same model, but more steps are done in the first case:

```
   selection method=forward(select=SL choose=CRITERION);
   selection method=forward(select=SL stop=CRITERION);
```

In some cases, there might be a better local extremum that cannot be reached if you specify the STOP= option but can be found if you use the CHOOSE= option. Also, you can use the CHOOSE= option in preference to the STOP= option if you want to examine how the named criterion behaves as you move beyond the step where the first local minimum of this criterion occurs.

You can specify both the CHOOSE= and STOP= options. You can also use these options together with options that specify size-based limits on the selected model. You might want to consider models that are generated by forward selection and have at most some fixed number of effects, but select from within this set based on a criterion that you specify. For example, specifying the following statements requests that forward selection continue until there are 20 effects in the final model and chooses among the sequence of models the one that has the largest value of the adjusted R-square statistic:

```
   selection method=forward(stop=none maxeffects=20 choose=ADJRSQ);
```
You can also combine these options to select a model where one of two conditions is met. For example, the following statement chooses whatever occurs first between a local minimum of the sum of squares on validation data and a local minimum of the corrected Akaike’s information criterion (AICC):

```
selection method=forward(stop=AICC choose=VALIDATE);
```

You can find discussion and references to studies about criteria for variable selection in Burnham and Anderson (2002), along with some cautions and recommendations.

**Examples of Forward Selection Specifications**

The following statement adds effects that at each step produce the lowest value of the SBC statistic and stops at the step where adding any effect would increase the SBC statistic:

```
selection method=forward stophorizon=1;
```

The following statement adds effects based on significance level and stops when all candidate effects for entry at a step have a significance level greater than the default entry significance level of 0.05:

```
selection=forward(select=SL);
```

The following statement adds effects based on significance level and stops at a step where adding any effect increases the error sum of squares computed on the validation data:

```
selection=forward(select=SL stop=validation) stophorizon=1;
```

The following statement adds effects that at each step produce the lowest value of the AIC statistic and stops at the first step whose AIC value is smaller than the AIC value at the next three steps:

```
selection=forward(select=AIC);
```

The following statement adds effects that at each step produce the largest value of the adjusted R-square statistic and stops at the step where the significance level that corresponds to the addition of this effect is greater than 0.2:

```
selection=forward(select=ADJRSQ stop=SL SLE=0.2);
```

**Backward Elimination**

This section applies to the following procedures: GENSELECT, LOGSELECT, PHSELECT, QTRSELECT, and REGSELECT.

METHOD=BACKWARD specifies the backward elimination technique. This technique starts from the full model, which includes all independent effects. Then effects are deleted one by one until a stopping condition is satisfied. At each step, the effect that shows the smallest contribution to the model is deleted.

In the traditional implementation of backward selection, the statistic that is used to determine whether to drop an effect is significance level. At any step, the least significant predictor is dropped and the process continues until all effects that remain in the model are significant at a specified stay significance level (SLS).

Just as with forward selection, you can use the SELECT= option to change the criterion that is used to assess effect contributions. You can also specify a stopping criterion in the STOP= option and use a CHOOSE= option to provide a criterion for selecting among the sequence of models produced. For more information, see the discussion in the section “Forward Selection” on page 62.
Examples of Backward Selection Specifications

The following statement removes effects that at each step produce the largest value of the Schwarz Bayesian information criterion (SBC) statistic and stops at the step where removing any effect increases the SBC statistic:

```
selection method=backward stophorizon=1;
```

The following statement bases removal of effects on the corrected Akaike’s information criterion (AICC) and stops at the first step whose AICC value is smaller than the AICC value at the next three steps:

```
selection method=backward(select=AICC);
```

The following statement bases removal of effects on significance level and stops when all candidate effects for removal at a step are significant at the default stay significance level of 0.05:

```
selection method=backward(select=SL);
```

The following statement bases removal of effects on significance level and stops when all effects in the model are significant at the 0.1 level. Finally, from the sequence of models generated, the chosen model is the one that produces the smallest average square error when scored on the validation data:

```
selection method=backward(select=SL choose=VALIDATE SLS=0.1);
```

The following statement applies in logistic regression models the fast backward technique of Lawless and Singhal (1978), a first-order approximation that has greater numerical efficiency than full backward selection:

```
selection method=backward(fast);
```

The fast technique fits an initial full logistic model and a reduced model after the candidate effects have been dropped. On the other hand, full backward selection fits a logistic regression model each time an effect is removed from the model.

Stepwise Selection

This section applies to the following procedures: GENSELECT, LOGSELECT, PHSELECT, QTRSELECT, and REGSELECT.

METHOD=STEPWISE specifies the stepwise method, which modifies the forward selection technique by allowing effects already in the model to be removed.

In the traditional implementation of stepwise selection method, the same entry and removal significance levels for the forward selection and backward elimination methods are used to assess contributions of effects as they are added to or removed from a model. If, at a step of the stepwise method, any effect in the model is not significant at the SLSTAY= level, then the least significant of these effects is removed from the model and the algorithm proceeds to the next step. This ensures that no effect can be added to a model while some effect currently in the model is not deemed significant. Another effect can be added to the model only after all necessary deletions have been accomplished. In this case the effect whose addition is the most significant is added to the model and the algorithm proceeds to the next step. The stepwise process ends when none of the effects outside the model is significant at the SLENTRY= level and every effect in the model is significant at the SLSTAY= level. In some cases, neither of these two conditions for stopping is met and the sequence of models cycles. In this case, the stepwise method terminates at the end of the cycle.
Just as you can in forward selection and backward elimination, you can use the \texttt{SELECT=} option to change the criterion that is used to assess effect contributions. You can also use the \texttt{STOP=} option to specify a stopping criterion and use a \texttt{CHOOSE=} option to provide a criterion for selecting among the sequence of models produced. For more information, see the section “Forward Selection” on page 62.

For selection criteria other than the significance level, further modification to the stepwise method is supported. In the standard stepwise method, no effect can enter the model if removing any effect currently in the model would yield an improved value of the selection criterion. In the modification, you can use the \texttt{COMPETITIVE} option to specify that addition and deletion of effects should be treated competitively. The selection criterion is evaluated for all models that are produced by deleting an effect from the current model or by adding an effect to this model. The action that most improves the selection criterion is the action taken.

\textbf{Examples of Stepwise Selection Specifications}

The following statement requests stepwise selection based on the SBC criterion:

\begin{verbatim}
   selection method=stepwise; 
\end{verbatim}

First, if removing any effect yields a model that has a lower SBC statistic than the current model, then the effect that produces the smallest SBC statistic is removed. If removing any effect increases the SBC statistic, then provided that adding some effect lowers the SBC statistic, the effect that produces the model that has the lowest SBC is added.

The following statement requests the traditional stepwise method:

\begin{verbatim}
   selection=stepwise(select=SL) 
\end{verbatim}

First, if the removal of any effect in the model is not significant at the default stay level of 0.05, then the least significant effect is removed and the algorithm proceeds to the next step. Otherwise, the effect whose addition is the most significant is added, provided that it is significant at the default entry level of 0.05.

The following statement requests the traditional stepwise method, where effects enter and leave based on significance levels, but with the following extra check: if any effect to be added or removed yields a model whose SBC statistic is greater than the SBC statistic of the current model, then the stepwise method terminates at the current model.

\begin{verbatim}
   selection method=stepwise(select=SL stop=SBC) stophorizon=1; 
\end{verbatim}

In this case, the entry and stay significance levels still play a role because they determine whether an effect is deleted from or added to the model. This extra check might result in the selection terminating before a local minimum of the SBC criterion is found.

The following statement selects effects to enter or drop as in the previous example except that the significance level for entry is now 0.1 and the significance level to stay is 0.08. From the sequence of models produced, the selected model is chosen to yield the minimum AIC statistic:

\begin{verbatim}
   selection method=stepwise(select=SL SLE=0.1 SLS=0.08 choose=AIC); 
\end{verbatim}

The following statement requests stepwise selection that is based on the AICC criterion and treats additions and deletions competitively:

\begin{verbatim}
   selection method=stepwise(select=AICC competitive); 
\end{verbatim}

Each step evaluates the AICC statistics that correspond to the removal of any effect in the current model or the addition of any effect to the current model and chooses the addition or removal that produced the minimum value, provided that this minimum is lower than the AICC statistic of the current model.
The following statement requests stepwise selection that is based on the SBC criterion, treats additions and deletions competitively, and stops based on the average square error over the validation data:

```
selection=stepwise(select=SBC competitive stop=VALIDATE);
```

At any step, SBC statistics that correspond to the removal of any effect from the current model or the addition of any effect to the current model are evaluated. The addition or removal that produces the minimum SBC value is made. The average square error on the validation data for the model with this addition or removal is evaluated. The selection stops when the average square error so produced increases for three consecutive steps.

**Forward-Swap Selection**

This section applies to the following procedures: REGSELECT.

METHOD=FORWARDSWAP specifies the forward-swap selection method, which is an extension of the forward selection method. The forward-swap selection method incorporates steps that improve a model by replacing an effect in the model with an effect that is not in the model. When the model selection criterion is R square, this method is the same as the maximum R-square improvement (MAXR) method that is implemented in the REG procedure in SAS/STAT software. You cannot use the effect significance level as the selection criterion for the forward-swap method.

The forward-swap selection method begins by finding the one-effect model that produces the best value of the selection criterion. Then another effect (the one that yields the greatest improvement in the selection criterion) is added. After the two-effect model is obtained, each of the effects in the model is compared to each effect that is not in the model. For each comparison, the forward-swap method determines whether removing one effect and replacing it with the other effect improves the selection criterion. After comparing all possible swaps, the forward-swap method makes the swap that produces the greatest improvement in the selection criterion. Comparisons begin again, and the process continues until the forward-swap method finds that no other swap could improve the selection criterion. Thus, the two-variable model that is produced is considered the “best” two-variable model that the technique can find. Another variable is then added to the model, and the comparing-and-swapping process is repeated to find the “best” three-variable model, and so on.

The difference between the stepwise selection method and the forward-swap selection method is that all swaps are evaluated before any addition is made in the forward-swap method. In the stepwise selection method, the “worst” effect might be removed without considering what adding the “best” remaining effects might accomplish. Because the forward-swap method needs to examine all possible pairwise effect swaps at each step of the selection process, the forward-swap method is much more computationally expensive than the stepwise selection method; it might not be appropriate for models that contain a large number of effects.

**Least Angle Regression**

This section applies to the following procedures: REGSELECT.

METHOD=LAR specifies least angle regression (LAR), which was introduced by Efron et al. (2004). Not only does this algorithm provide a selection method in its own right, but with one additional modification,
it can be used to efficiently produce LASSO solutions. Just like the forward selection method, the LAR algorithm produces a sequence of regression models in which one parameter is added at each step, terminating at the full least squares solution when all parameters have entered the model.

The algorithm starts by centering the covariates and response and scaling the covariates so that they all have the same corrected sum of squares. Initially all coefficients are 0, as is the predicted response. The predictor that is most correlated with the current residual is determined, and a step is taken in the direction of this predictor. The length of this step determines the coefficient of this predictor and is chosen so that some other predictor and the current predicted response have the same correlation with the current residual. At this point, the predicted response moves in the direction that is equiangular between these two predictors. Moving in this direction ensures that these two predictors continue to have a common correlation with the current residual. The predicted response moves in this direction until a third predictor has the same correlation with the current residual as the two predictors already in the model. A new direction is determined that is equiangular among these three predictors, and the predicted response moves in this direction until a fourth predictor, which has the same correlation with the current residual, joins the set. This process continues until all predictors are in the model.

As in other selection methods, the issue of when to stop the selection process is crucial. You can use the CHOOSE= option to specify a criterion for choosing among the models at each step. You can also use the STOP= option to specify a stopping criterion. These formulas use the approximation that at step $k$ of the LAR algorithm, the model has $k$ degrees of freedom. For a detailed discussion of this so-called simple approximation, see Efron et al. (2004).

A modification of LAR selection that is suggested in Efron et al. (2004) uses the LAR algorithm to select the set of covariates in the model at any step, but it uses ordinary least squares regression with just these covariates to obtain the regression coefficients. You can request this hybrid method by specifying the LSCOEFFS suboption of METHOD=LAR.

### LASSO Selection

This section applies to the following procedures: GENSELECT, LOGSELECT, PHSELECT, and REGSELECT.

METHOD=LASSO specifies the least absolute shrinkage and selection operator (LASSO) method. LASSO arises from a constrained form of ordinary least squares regression where the sum of the absolute values of the regression coefficients is constrained to be smaller than a specified parameter. More precisely let $X = (x_1, x_2, \ldots, x_m)$ denote the matrix of covariates and let $y$ denote the response, where the $x_i$ have been centered and scaled to have unit standard deviation and mean 0 and $y$ has mean 0. Then for a particular parameter $t$, the LASSO regression coefficients $\beta = (\beta_1, \beta_2, \ldots, \beta_m)$ are the solution to the following constrained optimization problem:

$$\minimize ||y - X\beta||^2 \quad \text{subject to} \quad \sum_{j=1}^{m} |\beta_j| \leq t$$

Provided that the LASSO parameter $t$ is small enough, some of the regression coefficients are exactly 0. Hence, you can view the LASSO as selecting a subset of the regression coefficients for each LASSO
parameter. By increasing the LASSO parameter in discrete steps, you obtain a sequence of regression coefficients in which the nonzero coefficients at each step correspond to selected parameters.

Early implementations (Tibshirani 1996) of LASSO selection used quadratic programming techniques to solve the constrained least squares problem for each LASSO parameter of interest. Later Osborne, Presnell, and Turlach (2000) developed a “homotopy method” that generates the LASSO solutions for all values of $t$. Efron et al. (2004) derived a variant of their algorithm for least angle regression that can be used to obtain a sequence of LASSO solutions from which all other LASSO solutions can be obtained by linear interpolation. This algorithm for METHOD=LASSO is used in PROC REGSELECT. It can be viewed as a stepwise procedure with a single addition to or deletion from the set of nonzero regression coefficients at any step.

As in the other selection methods, you can use the CHOOSE= option to specify a criterion to choose among the models at each step of the LASSO algorithm. You can also use the STOP= option to specify a stopping criterion. For more information, see the discussion in the section “Forward Selection” on page 62. The model degrees of freedom used at any step of the LASSO are simply the number of nonzero regression coefficients in the model at that step. Efron et al. (2004) cite empirical evidence for doing this but do not give any mathematical justification for this choice.

A modification of LASSO selection suggested in Efron et al. (2004) uses the LASSO algorithm to select the set of covariates in the model at any step, but it uses ordinary least squares regression and just these covariates to obtain the regression coefficients. You can request this hybrid method by specifying the LSCOEFFS suboption of SELECTION=LASSO.

Adaptive LASSO Selection

This section applies to the following procedures: REGSELECT.

Adaptive LASSO selection is a modification of LASSO selection; in adaptive LASSO selection, weights are applied to each of the parameters in forming the LASSO constraint (Zou 2006). More precisely, suppose that the response $y$ has mean 0 and the regressors $x$ are scaled to have mean 0 and common standard deviation. Furthermore, suppose that you can find a suitable estimator $\hat{\beta}$ of the parameters in the true model and you define a weight vector by $w = 1/|\hat{\beta}|^\gamma$, where $\gamma \geq 0$. Then the adaptive LASSO regression coefficients $\hat{\beta} = (\hat{\beta}_1, \hat{\beta}_2, \ldots, \hat{\beta}_m)$ are the solution to the following constrained optimization problem:

$$\text{minimize} \|y - X\beta\|^2 \quad \text{subject to} \quad \sum_{j=1}^m |w_j \beta_j| \leq t$$

The solution to the unconstrained least squares problem is used as the estimator $\hat{\beta}$. This is appropriate unless collinearity is a concern. If the regressors are collinear or nearly collinear, then Zou (2006) suggests using a ridge regression estimate to form the adaptive weights.

Group LASSO Selection

This section applies to the following procedures: GENSELECT, LOGSELECT, and PHSELECT.
The group LASSO method, proposed by Yuan and Lin (2006), is a variant of LASSO that is specifically
designed for models defined in terms of effects that have multiple degrees of freedom, such as the main effects
of CLASS variables and interactions between CLASS variables. If all effects in the model are continuous,
then the group LASSO method is the same as the LASSO method.

Recall that LASSO selection depends on solving a constrained optimization problem of the form

$$\min \{-L(\mu; y)\} \quad \text{subject to} \quad \sum_{j=1}^{m} |\beta_j| \leq t$$

where $L$ is the log-likelihood function. In this formulation, individual parameters can be included or excluded
from the model independently, subject only to the overall constraint. In contrast, the group LASSO method
uses a constraint that forces all parameters that correspond to the same effect to be included or excluded
simultaneously. For a model that has $k$ effects, let $\beta_{G_j}$ be the group of linear coefficients that correspond to
effect $j$ in the model. Then group LASSO depends on solving a constrained optimization problem of the form

$$\min \{-L(\mu; y)\} \quad \text{subject to} \quad \sum_{j=1}^{k} \sqrt{|G_j||\beta_{G_j}|} \leq t$$

where $|G_j|$ is the number of parameters that correspond to effect $j$, and $||\beta_{G_j}||$ denotes the Euclidean norm
of the parameters $\beta_{G_j}$,

$$||\beta_{G_j}|| = \sqrt{\sum_{i=1}^{G_j} \beta_i^2}$$

That is, instead of constraining the sum of the absolute value of individual parameters, group LASSO
constrains the Euclidean norm of groups of parameters, where groups are defined by effects.

You can write the group LASSO method in the equivalent Lagrangian form, which is an example of a
penalized log-likelihood function:

$$\min \{-L(\mu; y)\} + \lambda \sum_{j=1}^{k} \sqrt{|G_j||\beta_{G_j}|}$$

The weight $\sqrt{|G_j|}$ was suggested by Yuan and Lin (2006) in order to take the size of the group into
consideration in group LASSO.

Unlike LASSO for linear models, group LASSO does not allow a piecewise linear constant solution path as
generated by a LAR algorithm. Instead, the method proposed by Nesterov (2013) is adopted to solve the
Lagrangian form of the group LASSO problem that corresponds to a prespecified regularization parameter
$\lambda$. Nesterov’s method is known to have an optimal convergence rate for first-order black-box optimization.
Because the optimal $\lambda$ is usually unknown, a series of regularization parameters $\rho, \rho^2, \rho^3, \ldots$ is used, where
$\rho$ is a positive value less than 1. You can specify $\rho$ by using the LASSORHO= option in the procedure; the
default value is $\rho = 0.8$. In the $i$th step of group LASSO selection, the value that is used for $\lambda$ is $\rho^i$.

A unique feature of the group LASSO method is that it does not necessarily add or remove precisely one
effect at each step of the process. This is different from the forward, stepwise, and backward selection
methods.
As with the other selection methods, you can specify a criterion to choose among the models at each step of the group LASSO algorithm by using the CHOOSE= option in the SELECTION statement. You can also specify a stopping criterion by using the STOP= option in the SELECTION statement. If you do not specify either the CHOOSE= or STOP= option, the model at the last LASSO step is chosen as the selected model and parameter estimates are reported for this model. These parameter estimates are used to compute predicted values for the output data tables.

For more information, see the discussion in the section “SELECTION Statement” on page 36.

The model degrees of freedom at any step of the LASSO are simply the number of nonzero regression coefficients in the model at that step. Efron et al. (2004) cite empirical evidence for doing this but do not give any mathematical justification for this choice.

Some distributions involve a dispersion parameter (the parameter $\phi$ in the expressions for the log likelihood). These parameters are not estimated by the LASSO optimization algorithm, and are set to either the default value or a value that you specify. You can use the PHI= option in the procedure to set the dispersion to a fixed value.

---

**Model Selection Plots**

This section describes the ODS graphical displays that you can request by using the PLOTS= option in the SELECTION statement.

This section applies to the following procedures: GENSELECT, LOGSELECT, PHSELECT, QTRSELECT, and REGSELECT.

The example plots shown in this section are produced by using the REGSELECT procedure.

The following DATA step produces simulated example data that contain a variable that you can use to assign observations to the training, validation, and testing roles. In this case, each role has 5,000 observations. These statements assume that your CAS engine libref is named mycas, as in the section “Using CAS Sessions and CAS Engine Librefs” on page 10, but you can substitute any appropriately defined CAS engine libref.

```plaintext
data mycas.exampleData;
  drop i j;
  array x{20} x1-x20;
  array c{5} c1-c5;
  call streaminit(1);
  do i=1 to 15000;
    do j=1 to dim(x);
      x(j) = rand('NORMAL');
    end;
    do j=1 to dim(c);
      c(j) = 1+ int(rand('UNIFORM')*3);
    end;
  end;
```
The following statements perform forward selection to obtain a parsimonious linear regression model for the response:

```sas
ods graphics on;
proc regselect data=mycas.exampleData;
   partition rolevar=Role(train='TRAIN' validate='VAL' test='TEST');
   class c: ;
   model y = x: c: ;
   selection method=forward(stop=AICC CHOOSE=validate) plots=all;
run;
```

Because you specified the PLOTS=ALL option in the SELECTION statement and have enabled ODS Graphics, the plots that are described in the following sections are produced.

**Coefficient Panel**

Figure 3.6 shows the coefficient panel, which contains two plots that show how the standardized coefficients and the criterion used to choose the final model evolve as the selection progresses.
**Figure 3.6 Coefficient Panel**

The upper plot in the panel displays the standardized coefficients as a function of the step number. To help trace the changes in a parameter, the standardized coefficients for each parameter are connected by lines. Coefficients that correspond to effects not in the selected model at a step are zero and hence not observable. Parameters that are nonzero at the final step of the selection are labeled if their magnitudes are greater than 1% of the range of the magnitudes of all the nonzero parameters at this step. To avoid collision, labels that correspond to parameters that have similar values at the final step might be suppressed.

The lower plot in the panel shows how the criterion used to choose among the examined models progresses. The selected step occurs at the optimal value of this criterion. In this example, this criterion is the average square error (ASE) on the validation data, and it achieves its minimal value at step 6 of the forward selection.

**Criterion Panel**

Figure 3.7 shows the criterion panel, which contains plots that show the progression of fit criteria.
The selected step is displayed as a vertical reference line on the plot of each criterion. The optimal value of each criterion is indicated with the “star” marker. Note that it is possible that a better value of a criterion might have been reached if more steps of the selection process had been done.

**Fit by Role Plot**

Figure 3.8 displays the fit by role plot, which shows the progression of a fit statistic evaluated separately on the training, validation, and test data. If you do not partition your data into roles by using a `PARTITION` statement, then the fit by role plot is not produced.
You use this plot to detect overfitting of the training data. For many types of regression (including the linear regression that is used in the examples in this section), the fit statistic shown in this plot is the average square error (ASE) evaluated separately for each data role. The ASE decreases monotonically on the training data as parameters are added to a model. However, the ASE on test and validation data usually starts increasing when overfitting occurs.

**Using Options to Customize Plots**

In some cases, particularly when the final step contains a large number of parameters, you might be interested in using the coefficient panel only to discern if and when the parameters in the model are essentially unchanged beyond a certain step. In such cases, you might want to suppress the labeling of the parameters and use a numeric axis on the horizontal axis of the plot. You can do this by using the STEPAXIS= and MAXPARMLABEL= global plot options. The following statements provide an example:

```plaintext
proc regselect data=mycas.exampleData;
    partition rolevar=Role(train='TRAIN' validate='VAL' test='TEST');
    class c: ;
    model y = x: c: ;
```
The `selection` method=`forward`(`stop=none` `choose=validate`) plots(`maxparmlabel=1` `stepaxis=number`)=(`coefficients` `criteria`(unpack)); run;

The `STEPAXIS=NUMBER` option requests that a numeric horizontal axis showing step number be used on all plots. The `MAXPAMLABEL=1` option suppresses the labels for the parameters in the coefficient progression panel.

The “Coefficient Progression Panel” is shown in Figure 3.9. You can see that the standardized coefficients of the selected parameters do not vary greatly after the selected step (step 6).

**Figure 3.9** Coefficient Panel

![Coefficient Progression for y](image)

The UNPACK option for the criterion panel specifies that the individual plots in this panel be shown as separate plots. Figure 3.10 shows the plot for the AIC criterion.
Examining Specific Step Ranges

The coefficient panel, criterion panel, and average square error plot display information for all the steps examined in the selection process. In some cases, you might want to focus attention on just a particular step range. For example, it is hard to discern the variation in the criteria displayed in Figure 3.7 near the selected step because the variation in these criteria in the steps close to the selected step is small relative to the variation across all steps. You can request a range of steps to display by using the STARTSTEP= and ENDFSTEP= suboptions of the PLOTS= option. The following statements provide an example:

```
proc regselect data=mycas.exampleData;
   partition rolevar=Role(train='TRAIN' validate='VAL' test='TEST');
   class c: ;
   model y = x: c: ;
   selection method=forward(stop=none choose=validate)
       plots(startstep=3 endstep=8)=criteria;
run;
ods graphics off;
```
Figure 3.11 shows the progression of the fit criteria between steps 3 and 8. Note that if the optimal value of a criterion does not occur in this specified step range, then no optimal marker appears for that criterion.

**Figure 3.11** Criterion Panel for Specified Step Range

---

**Informative Missingness**

This section applies to the following procedures: GENSELECT, LOGSELECT, PHSELECT, QTRSELECT, and REGSELECT.

The concept of informative missingness is one way to account for missing values in statistical analyses and, in particular, statistical modeling. Missing values can be a problem because they reduce the amount of available data. When you work with classification variables (factors, which are levelized variables), you can treat a missing value as an actual level of the variable and allow it to participate in the analysis.

However, when continuous variables have missing values, the observation is removed from the analysis. In data that have many missing values, removing observations can reduce the amount of available data greatly,
and the sets of observations used in one model versus another model can vary based on which variables are included in the model.

Of course, there are many reasons for missing values, and substituting values for missing values has to be done with caution. For example, the famous Framingham Heart study data set contains 5,209 observations on subjects in a longitudinal study that helped understand the relationship between smoking, cholesterol, and coronary heart disease. One of the variables in the data set is $\text{AgeCHDdiag}$. This variable represents the age at which a patient was diagnosed with coronary heart disease (CHD). If you include this variable in a statistical model, only 1,449 observations are available, because the value cannot be observed unless a patient has experienced CHD. Including this variable acts as a filter that reduces the analysis set to the subjects who have CHD. You cannot impute the value for subjects where the variable has a missing value, because you cannot impute an age at which someone who has not had CHD would have contracted coronary heart disease.

With informative missingness, you are not so much substituting imputed values for the missing values as you are modeling the missingness. Consider a simple linear regression model:

$$y = \beta_0 + \beta_1 x + \epsilon$$

Suppose that some of the values for the regressor variable $x$ are missing. The fitted model uses only observations for which $y$ and $x$ have been observed.

In order to predict the outcome $y$ for an observation that has a missing $x$, either you assume that $y$ is missing or you substitute a value (such as the average value, $\bar{x}$) for the missing $x$. Because the estimate for the intercept is in the simple linear regression model, the predicted value would be the average response of the nonmissing values, $\bar{y}$.

With informative missingness, you extend the model by adding extra effects for each effect that contains at least one continuous variable. In the simple linear regression model, you add one column to the model and slightly change the content of the $x$ variable:

$$y = \beta_0 + \beta_1 x^* + \beta_2 x_{miss} + \epsilon_1$$

The variable $x^*$ contains the original values of $x$ if they are not missing, and the average of $x$ otherwise:

$$x^* = \begin{cases} x & \text{if } x \text{ is not missing} \\ \bar{x} & \text{otherwise} \end{cases}$$

The variable $x_{miss}$ is a dummy variable whose value is 1 when $x$ is missing, and 0 otherwise:

$$x_{miss} = \begin{cases} 1 & \text{if } x \text{ is missing} \\ 0 & \text{otherwise} \end{cases}$$

The fitted model is not the same model that results from substituting $\bar{x}$ for the missing values during training, because the model that simply substitutes $\bar{x}$ for the missing values is

$$y = \beta_0 + \beta_1 x^* + \epsilon_2$$

The informative missing model has an extra parameter, and unless all values of $x_{miss}$ are 0 (in which case there are no missing values), the informative missing model has a higher R-square value, because it picks up more variation.

The parameter estimate for $\beta_2$ measures the amount by which the predicted value differs from a predicted value at $\bar{x}$. 

Using Validation and Test Data

This section applies to the following procedures: GENSELECT, LOGSELECT, PHSELECT, QTRSELECT, REGSELECT, and TREESPLIT.

When you have sufficient data, you can divide your data into three parts called the training, validation, and test data. During the selection process, models are fit on the training data, and the prediction errors for the models so obtained are found by using the validation data. This prediction error on the validation data can be used to decide when to terminate the selection process and to decide which model to select. Finally, after a model has been selected, the test set can be used to assess how the selected model generalizes on data that played no role in selecting the model.

In some cases, you might want to use only training and test data. For example, you might decide to use an information criterion to decide which effects to include and when to terminate the selection process. In this case, no validation data are required, but test data can still be useful in assessing the predictive performance of the selected model. In other cases, you might decide to use validation data during the selection process but forgo assessing the selected model on test data. Hastie, Tibshirani, and Friedman (2001) note that it is difficult to provide a general rule for how many observations you should assign to each role. They note that a typical split might be 50% for training and 25% each for validation and testing.

You use a PARTITION statement to logically subdivide the input data table into separate roles. You can specify the fractions of the data that you want to reserve as test data and validation data. For example, the following statements randomly divide the inData data table, reserving 50% for training and 25% each for validation and testing:

```sas
proc logselect data=mycas.inData;
  partition fraction(test=0.25 validate=0.25);
  ...
run;
```

You can specify the SEED= option in the PARTITION statement to create the same partition data tables for a particular number of compute nodes. However, changing the number of compute nodes changes the initial distribution of data, resulting in different partition data tables.

In some cases, you might need to exercise more control over the partitioning of the input data table. You can do this by naming both a variable in the input data table and a formatted value of that variable for each role. For example, the following statements assign roles to the observations in the inData data table that are based on the value of the variable Group in that data table. Observations whose value of Group is 'Group 1' are assigned for testing, and those whose value is 'Group 2' are assigned to training. All other observations are ignored.

```sas
proc logselect data=mycas.inData;
  partition roleVar=Group(test='Group 1' train='Group 2')
  ...
run;
```

When you have reserved observations for training, validation, and testing, a model that is fit on the training data is scored on the validation and test data, and statistics are computed separately for each of these subsets.
Using the Validation Statistic as the CHOOSE= Criterion

When you specify the \texttt{CHOOSE=VALIDATE} suboption of the \texttt{METHOD=} option in the \texttt{SELECTION} statement, the ASE is computed on the validation data for the models at each step of the selection process. The smallest model at any step that yields the smallest validation ASE is selected.

Using the Validation Statistic as the STOP= Criterion

When you specify the \texttt{STOP=VALIDATE} suboption of the \texttt{METHOD=} option in the \texttt{SELECTION} statement, the ASE is computed on the validation data for the models at each step of the selection process. At step $k$ of the selection process, the best candidate effect to enter or leave the current model is determined and the validation ASE for this new model is computed. If this validation ASE is greater than the validation ASE for the model at step $k$, then the selection process terminates at step $k$.

Multithreading

This section applies to the following procedures: \texttt{ASSESS}, \texttt{BINNING}, \texttt{CARDINALITY}, \texttt{CORRELATION}, \texttt{GAMMOD}, \texttt{GENSELECT}, \texttt{ICA}, \texttt{KCLUS}, \texttt{LMIXED}, \texttt{LOGSELECT}, \texttt{MBC}, \texttt{NLMOD}, \texttt{PARTITION}, \texttt{PCA}, \texttt{PHSELECT}, \texttt{PLSMOD}, \texttt{QTRSELECT}, \texttt{REGSELECT}, \texttt{TREESPLIT}, \texttt{VARIMPUTE}, and \texttt{VARREduce}.

Threading refers to the organization of computational work into multiple tasks (processing units that can be scheduled by the operating system). A task is associated with a thread. Multithreading refers to the concurrent execution of threads. When multithreading is possible, substantial performance gains can be realized compared to sequential (single-threaded) execution. The number of threads spawned by a procedure in this book is determined by your installation.

The tasks that are multithreaded by procedures in this book are primarily defined by dividing the data that are processed on a single machine among the threads—that is, the procedures implement multithreading through a data-parallel model. For example, if the input data table has 1,000 observations and the procedure is running on four threads, then 250 observations are associated with each thread. All operations that require access to the data are then multithreaded. These operations include the following (not all operations are required for all procedures):

- variable leveling
- effect leveling
- formation of the initial crossproducts matrix
- formation of approximate Hessian matrices for candidate evaluation during model selection
- objective function calculation
- gradient calculation
- Hessian calculation
- scoring of observations

In addition, operations on matrices such as sweeps can be multithreaded provided that the matrices are of sufficient size to realize performance benefits from managing multiple threads for the particular matrix operation.
Chapter 3: Shared Concepts

Choosing an Optimization Algorithm

This section applies to the following procedures: GAMMOD, GENSELECT, LMIXED, LOGSELECT, NLMOD, and PHSELECT.

First- or Second-Order Algorithms

The factors that go into choosing a particular optimization technique for a particular problem are complex. Trial and error can be involved.

For many optimization problems, computing the gradient takes more computer time than computing the function value. Computing the Hessian sometimes takes much more computer time and memory than computing the gradient, especially when there are many decision variables. Unfortunately, optimization techniques that do not use some kind of Hessian approximation usually require many more iterations than techniques that do use a Hessian matrix, and as a result the total run time of these techniques is often longer. Techniques that do not use the Hessian also tend to be less reliable. For example, they can terminate more easily at stationary points than at global optima.

Table 3.21 shows which derivatives are required for each optimization technique.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>First-Order</th>
<th>Second-Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRUREG</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>NEWRAP</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>NRRIDG</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>QUANEW</td>
<td>x</td>
<td>-</td>
</tr>
<tr>
<td>DBLDOG</td>
<td>x</td>
<td>-</td>
</tr>
<tr>
<td>CONGRA</td>
<td>x</td>
<td>-</td>
</tr>
<tr>
<td>LEVMAR</td>
<td>x</td>
<td>-</td>
</tr>
<tr>
<td>NMSIMP</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

The second-derivative methods TRUREG, NEWRAP, and NRRIDG are best for small problems for which the Hessian matrix is not expensive to compute. Sometimes the NRRIDG algorithm can be faster than the TRUREG algorithm, but TRUREG can be more stable. The NRRIDG algorithm requires only one matrix with \( p(p + 1)/2 \) double words; TRUREG and NEWRAP require two such matrices. Here, \( p \) denotes the number of parameters in the optimization.

The first-derivative methods QUANEW and DBLDOG are best for medium-sized problems for which the objective function and the gradient can be evaluated much faster than the Hessian. In general, the QUANEW and DBLDOG algorithms require more iterations than TRUREG, NRRIDG, and NEWRAP, but each iteration can be much faster. The QUANEW and DBLDOG algorithms require only the gradient to update an approximate Hessian, and they require slightly less memory than TRUREG or NEWRAP.

The first-derivative method CONGRA is best for large problems for which the objective function and the gradient can be computed much faster than the Hessian and for which too much memory is required to store
Choosing an Optimization Algorithm

the (approximate) Hessian. In general, the CONGRA algorithm requires more iterations than QUANEW or DBLDOG, but each iteration can be much faster. Because CONGRA requires only a factor of $p$ double-word memory, many large applications can be solved only by CONGRA.

The no-derivative method NMSIMP is best for small problems for which derivatives are not continuous or are very difficult to compute.

The LEVMAR method is appropriate only for least squares optimization problems.

Each optimization method uses one or more convergence criteria that determine when it has converged. An algorithm is considered to have converged when any one of the convergence criteria is satisfied. For example, under the default settings, the QUANEW algorithm converges if $\text{ABSGCONV} < 1 \times 10^{-5}$, $\text{FCONV} < 2 \times \epsilon$, or $\text{GCONV} < 1 \times 10^{-8}$.

By default, procedures in this book apply the NRRIDG algorithm because it can take advantage of multi-threading in Hessian computations and inversions. If the number of parameters becomes large, specifying the TECHNIQUE=QUANEW option, which is a first-order method with good overall properties, is recommended.

Algorithm Descriptions

The following subsections provide details about each optimization technique and follow the same order as Table 3.21.

Trust Region Optimization (TRUREG)

The trust region method uses the gradient $g(\beta^{(k)})$ and the Hessian matrix $H(\beta^{(k)})$; thus, it requires that the objective function $f(\beta)$ have continuous first- and second-order derivatives inside the feasible region.

The trust region method iteratively optimizes a quadratic approximation to the nonlinear objective function within a hyperelliptic trust region with radius $\Delta$ that constrains the step size that corresponds to the quality of the quadratic approximation. The trust region method is implemented based on Dennis, Gay, and Welsch (1981), Gay (1983), and Moré and Sorensen (1983).

The trust region method performs well for small- to medium-sized problems, and it does not need many function, gradient, and Hessian calls. However, if the computation of the Hessian matrix is computationally expensive, one of the dual quasi-Newton or conjugate gradient algorithms might be more efficient.

Newton-Raphson Optimization with Line Search (NEWRAP)

The NEWRAP technique uses the gradient $g(\beta^{(k)})$ and the Hessian matrix $H(\beta^{(k)})$; thus, it requires that the objective function have continuous first- and second-order derivatives inside the feasible region. If second-order derivatives are computed efficiently and precisely, the NEWRAP method can perform well for medium-sized to large problems, and it does not need many function, gradient, and Hessian calls.

This algorithm uses a pure Newton step when the Hessian is positive-definite and when the Newton step reduces the value of the objective function successfully. Otherwise, a combination of ridging and line search is performed to compute successful steps. If the Hessian is not positive-definite, a multiple of the identity matrix is added to the Hessian matrix to make it positive-definite (Eskow and Schnabel 1991).

In each iteration, a line search is performed along the search direction to find an approximate optimum of the objective function. The line-search method uses quadratic interpolation and cubic extrapolation.
Newton-Raphson Ridge Optimization (NRRIDG)

The NRRIDG technique uses the gradient $g(\beta^{(k)})$ and the Hessian matrix $H(\beta^{(k)})$; thus, it requires that the objective function have continuous first- and second-order derivatives inside the feasible region.

This algorithm uses a pure Newton step when the Hessian is positive-definite and when the Newton step reduces the value of the objective function successfully. If at least one of these two conditions is not satisfied, a multiple of the identity matrix is added to the Hessian matrix.

Because the NRRIDG technique uses an orthogonal decomposition of the approximate Hessian, each iteration of NRRIDG can be slower than that of the NEWRAP technique, which works with a Cholesky decomposition. However, NRRIDG usually requires fewer iterations than NEWRAP.

The NRRIDG method performs well for small- to medium-sized problems, and it does not require many function, gradient, and Hessian calls. However, if the computation of the Hessian matrix is computationally expensive, one of the dual quasi-Newton or conjugate gradient algorithms might be more efficient.

Quasi-Newton Optimization (QUANEW)

The dual quasi-Newton method uses the gradient $g(\beta^{(k)})$, and it does not need to compute second-order derivatives because they are approximated. It works well for medium-sized to moderately large optimization problems, where the objective function and the gradient can be computed much faster than the Hessian. However, in general the QUANEW technique requires more iterations than the TRUREG, NEWRAP, and NRRIDG techniques, which compute second-order derivatives. The QUANEW technique provides an appropriate balance between the speed and stability required for most nonlinear mixed model applications.

The QUANEW technique implemented by procedures in this book is the dual quasi-Newton algorithm, which updates the Cholesky factor of an approximate Hessian.

In each iteration, a line search is performed along the search direction to find an approximate optimum. The line-search method uses quadratic interpolation and cubic extrapolation to obtain a step size $\alpha$ that satisfies the Goldstein conditions (Fletcher 1987). One of the Goldstein conditions can be violated if the feasible region defines an upper limit of the step size. Violating the left-side Goldstein condition can affect the positive-definiteness of the quasi-Newton update. In that case, either the update is skipped or the iterations are restarted with an identity matrix, resulting in the steepest descent or ascent search direction.

Double-Dogleg Optimization (DBLDOG)

The double-dogleg optimization method combines the ideas of the quasi-Newton and trust region methods. In each iteration, the double-dogleg algorithm computes the step $s^{(k)}$ as the linear combination of the steepest descent or ascent search direction $s_1^{(k)}$ and a quasi-Newton search direction $s_2^{(k)}$:

$$s^{(k)} = \alpha_1 s_1^{(k)} + \alpha_2 s_2^{(k)}$$

The step is requested to remain within a prespecified trust region radius (Fletcher 1987, p. 107). Thus, the DBLDOG subroutine uses the dual quasi-Newton update but does not perform a line search.

The double-dogleg optimization technique works well for medium-sized to moderately large optimization problems, where the objective function and the gradient can be computed much faster than the Hessian. The implementation is based on Dennis and Mei (1979) and Gay (1983), but it is extended for dealing with boundary and linear constraints. The DBLDOG technique generally requires more iterations than the TRUREG, NEWRAP, and NRRIDG techniques, which require second-order derivatives; however, each of the DBLDOG iterations is computationally cheap. Furthermore, the DBLDOG technique requires only gradient calls for the update of the Cholesky factor of an approximate Hessian.
Conjugate Gradient Optimization (CONGRA)
Second-order derivatives are not required by the CONGRA algorithm and are not even approximated. The CONGRA algorithm can be expensive in function and gradient calls, but it requires only $O(p)$ memory for unconstrained optimization. In general, many iterations are required to obtain a precise solution, but each of the CONGRA iterations is computationally cheap.

The CONGRA subroutine should be used for optimization problems with large $p$. For the unconstrained or boundary-constrained case, CONGRA requires only $O(p)$ bytes of working memory, whereas all other optimization methods require order $O(p^2)$ bytes of working memory. During $p$ successive iterations, uninterrupted by restarts or changes in the working set, the conjugate gradient algorithm computes a cycle of $p$ conjugate search directions. In each iteration, a line search is performed along the search direction to find an approximate optimum of the objective function. The line-search method uses quadratic interpolation and cubic extrapolation to obtain a step size $\alpha$ that satisfies the Goldstein conditions. One of the Goldstein conditions can be violated if the feasible region defines an upper limit for the step size.

Levenberg-Marquardt Optimization (LEVMAR)
The LEVMAR algorithm performs a highly stable optimization; however, for large problems, it consumes more memory and takes longer than the other techniques. The Levenberg-Marquardt optimization technique is a slightly improved variant of the Moré (1978) implementation.

Nelder-Mead Simplex Optimization (NMSIMP)
The Nelder-Mead simplex method does not use any derivatives and does not assume that the objective function has continuous derivatives. The objective function itself needs to be continuous. This technique is quite expensive in the number of function calls, and it might be unable to generate precise results for $p \gg 40$.

The original Nelder-Mead simplex algorithm is implemented and extended to boundary constraints. This algorithm does not compute the objective for infeasible points, but it changes the shape of the simplex adapting to the nonlinearities of the objective function. This change contributes to an increased speed of convergence and uses a special termination criterion.

OutputCasTables Table
The OutputCasTables table is a special table that has information about each CAS table that is created during a CAS action execution. The information for each CAS table consists of the CAS table name, the caslib in which the table resides, and the number of columns and rows in the CAS table. Because this table is not a typical ODS table that contains analytical results, you cannot include it in the `table-spec-list` in the DISPLAYOUT statement.

References


Part II

Statistics
Chapter 4
The CORRELATION Procedure

Overview: CORRELATION Procedure

The CORRELATION procedure computes Pearson correlation coefficients and the probabilities that are associated with these statistics in SAS Viya.

The Pearson product-moment correlation is a parametric measure of a linear relationship between two variables. When only one set of analysis variables is specified, the default correlation analysis includes
Chapter 4: The CORRELATION Procedure

The CORRELATION procedure offers the following functionality:

- provides a WITH statement for analysis of pairwise Pearson correlation statistics between two sets of variables
- provides a FREQ statement for grouped analysis
- provides a WEIGHT statement for weighted analysis
- produces output data tables that contain descriptive statistics for analysis variables, Pearson correlation statistics, covariances, sums of squares, and corrected sums of squares
- computes Cronbach’s coefficient alpha for estimating reliability

Because the CORRELATION procedure runs on CAS, it also does the following:

- enables you to run on a cluster of machines that distribute the data and the computations
- enables you to run in single-machine mode
- exploits all the available cores and concurrent threads (for information about how PROC CORRELATION uses threads, see the section “Multithreading” on page 81 in Chapter 3, “Shared Concepts”)

PROC CORRELATION Compared with Other SAS Procedures

The CORRELATION procedure provides functionality that is comparable to that of the HPCORR and CORR procedures in Base SAS software.

The functionality of the CORRELATION procedure closely resembles that of the HPCORR procedure, which is a high-performance procedure. The CORRELATION procedure is the next generation of the HPCORR procedure, and it was developed specifically for SAS Viya. Both procedures are designed to run on a cluster of machines that distribute the data and the computations.

The CORRELATION procedure, the HPCORR procedure, and the CORR procedure have the similarities and differences shown in Table 4.1.
Table 4.1 Comparison of PROC CORRELATION, PROC HPCORR, and PROC CORR

<table>
<thead>
<tr>
<th>Feature</th>
<th>CORRELATION</th>
<th>HPCORR</th>
<th>CORR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supports the computation of Pearson correlation coefficients and the probabilities associated with these statistics</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Supports BY statement</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Supports DISPLAY and DISPLAYOUT statements</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Supports the computation of nonparametric measure of association</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Supports OUTP statements</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Supports Cronbach’s alpha coefficient alpha for estimating reliability</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Threading</td>
<td>Specifically designed for CAS; executes on multiple threads</td>
<td>Primarily designed for a distributed environment; executes on multiple threads</td>
<td>Executes on a single thread</td>
</tr>
</tbody>
</table>

Using CAS Sessions and CAS Engine Librefs

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:

```sas
proc options option=(CASHOST CASPORT);
run;
```

In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:
The CAS statement creates the CAS session named mysess, and the LIBNAME statement creates the mycas CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the CASHOST and CASPORT of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the mysess session, you can terminate it by using the TERMINATE option in the CAS statement as follows:

```sas
   cas mysess terminate;
```

For more information about the CAS and LIBNAME statements, see the section “Introduction to Shared Concepts” on page 10 in Chapter 3, “Shared Concepts.”

---

### Getting Started: CORRELATION Procedure

This example creates a data table and then uses PROC CORRELATION to produce Pearson correlations.

The following statements create the Fitness data table in your CAS session. These data have been altered to contain some missing values:

```sas
*----------------- Data on Physical Fitness -----------------*
| These measurements were made on men involved in a physical |
| fitness course at N.C. State University.                 |
| The variables are Age (years), Weight (kg),              |
| Runtime (time to run 1.5 miles in minutes), and          |
| Oxygen (oxygen intake, ml per kg body weight per minute) |
| Certain values were changed to missing for the analysis. |
*------------------------------------------------------------*;

   data mycas.Fitness;
   input Age Weight Oxygen RunTime @@;
   datalines;
    44 89.47 44.609 11.37 40 75.07 45.313 10.07
    44 85.84 54.297 8.65 42 68.15 59.571 8.17
    38 89.02 49.874 . 47 77.45 44.811 11.63
    40 75.98 45.681 11.95 43 81.19 49.091 10.85
    44 81.42 39.442 13.08 38 81.87 60.055 8.63
    44 70.03 50.541 10.13 45 87.66 37.388 14.03
    45 66.45 44.754 11.12 47 79.15 42.73 10.60
    54 83.12 51.855 10.33 49 81.42 49.156 8.95
    51 69.63 40.836 10.95 51 77.91 46.672 10.00
    48 91.33 46.774 10.25 49 73.33  . 10.08
    57 73.37 39.407 12.63 54 79.33 46.080 11.17
    52 76.32 45.441 9.63 50 70.87 54.625 8.92
    51 67.25 45.118 11.08 54 91.63 39.203 12.88
    51 73.71 45.790 10.47 57 59.08 50.545 9.93
    49 76.32  . 10.47 48 61.85 47.920 11.50
    52 82.78 47.467 10.50
;```

---

This example creates a data table and then uses PROC CORRELATION to produce Pearson correlations.
These statements assume that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

The following statements invoke the CORRELATION procedure and request a correlation analysis:

```plaintext
proc correlation data=mycas.Fitness;
run;
```

The “Simple Statistics” table in Figure 4.1 displays univariate statistics for the analysis variables.

**Figure 4.1** Univariate Statistics

<table>
<thead>
<tr>
<th>Variable</th>
<th>N</th>
<th>Mean</th>
<th>Std Dev</th>
<th>Sum</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Age</td>
<td>31</td>
<td>47.67742</td>
<td>5.21144</td>
<td>1478</td>
<td>38.00000</td>
<td>57.00000</td>
</tr>
<tr>
<td>Weight</td>
<td>31</td>
<td>77.44452</td>
<td>8.32857</td>
<td>2401</td>
<td>59.08000</td>
<td>91.63000</td>
</tr>
<tr>
<td>Oxygen</td>
<td>29</td>
<td>47.22721</td>
<td>5.47718</td>
<td>1370</td>
<td>37.38800</td>
<td>60.05500</td>
</tr>
<tr>
<td>RunTime</td>
<td>29</td>
<td>10.67414</td>
<td>1.39194</td>
<td>309.55000</td>
<td>8.17000</td>
<td>14.03000</td>
</tr>
</tbody>
</table>

By default, all numeric variables not listed in other statements are used in the analysis. Observations that have nonmissing values for each variable are used to derive the univariate statistics for that variable.

The “Pearson Correlation Coefficients” table in Figure 4.2 displays the Pearson correlation, the p-value under the null hypothesis of zero correlation, and the number of nonmissing observations for each pair of variables.

**Figure 4.2** Pearson Correlation Coefficients

<table>
<thead>
<tr>
<th>Pearson Correlation Coefficients</th>
<th>Prob &gt;</th>
<th></th>
<th>Number of Observations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Age</td>
</tr>
<tr>
<td>Age</td>
<td>1.0000</td>
<td>-0.2335</td>
<td>-0.3147</td>
</tr>
<tr>
<td>Weight</td>
<td>-0.2335</td>
<td>1.0000</td>
<td>-0.1536</td>
</tr>
<tr>
<td>Oxygen</td>
<td>-0.3147</td>
<td>-0.1536</td>
<td>1.0000</td>
</tr>
<tr>
<td>RunTime</td>
<td>0.1448</td>
<td>0.2007</td>
<td>-0.8684</td>
</tr>
</tbody>
</table>

By default, Pearson correlation statistics are computed from observations that have nonmissing values for each pair of analysis variables. Figure 4.2 displays a correlation of –0.86843 between Runtime and Oxygen, which is significant with a p-value less than 0.0001. That is, an inverse linear relationship exists between these two variables. As Runtime (time in minutes to run 1.5 miles) increases, Oxygen (oxygen intake in milliliters per kilogram body weight per minute) decreases.
Syntax: CORRELATION Procedure

The following statements are available in PROC CORRELATION:

```plaintext
PROC CORRELATION < options > ;
   BY variables ;
   DISPLAY < table-list > < / options > ;
   DISPLAYOUT table-spec-list < / options > ;
   FREQ variable ;
   VAR variables ;
   WEIGHT variable ;
   WITH variables ;
```

The PROC CORRELATION statement is the only required statement for the CORRELATION procedure. The following sections describe the PROC CORRELATION statement and then describe the other statements in alphabetical order.

PROC CORRELATION Statement

```plaintext
PROC CORRELATION < options > ;
```

The PROC CORRELATION statement invokes the procedure. Table 4.2 summarizes the options available in the PROC CORRELATION statement.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Tables</td>
<td></td>
</tr>
<tr>
<td>DATA=</td>
<td>Specifies the CAS input data table</td>
</tr>
<tr>
<td>OUTP=</td>
<td>Specifies the CAS output data table to contain the Pearson correlation statistics</td>
</tr>
<tr>
<td>Statistical Analysis</td>
<td></td>
</tr>
<tr>
<td>EXCLNPWG T</td>
<td>Excludes observations that have nonpositive weight values from the analysis</td>
</tr>
<tr>
<td>NOMISS</td>
<td>Excludes observations that have missing analysis values from the analysis</td>
</tr>
<tr>
<td>Statistics</td>
<td></td>
</tr>
<tr>
<td>ALPHA</td>
<td>Computes Cronbach’s coefficient alpha</td>
</tr>
<tr>
<td>COV</td>
<td>Computes covariances</td>
</tr>
<tr>
<td>CSSCP</td>
<td>Computes corrected sums of squares and crossproducts</td>
</tr>
<tr>
<td>SSCP</td>
<td>Computes sums of squares and crossproducts</td>
</tr>
<tr>
<td>VARDEF=</td>
<td>Specifies the divisor for variance calculations</td>
</tr>
<tr>
<td>Printed Output</td>
<td></td>
</tr>
<tr>
<td>BEST=</td>
<td>Displays the specified number of ordered correlation coefficients</td>
</tr>
<tr>
<td>NOCORR</td>
<td>Suppresses Pearson correlations</td>
</tr>
<tr>
<td>NOPROB</td>
<td>Suppresses p-values</td>
</tr>
<tr>
<td>NOSIMPLE</td>
<td>Suppresses descriptive statistics</td>
</tr>
<tr>
<td>RANK</td>
<td>Displays ordered correlation coefficients</td>
</tr>
</tbody>
</table>
ALPHA

calculates and prints Cronbach’s coefficient alpha. PROC CORRELATION computes separate coefficients by using raw and standardized values (scaling the variables to a unit variance of 1). For each analysis variable, PROC CORRELATION computes the correlation between that variable and all the remaining variables. It also computes Cronbach’s coefficient alpha by using only the remaining variables.

When you specify this option, the Pearson correlations are also displayed. If you also specify the OUTP= option, the output data table also contains observations that have Cronbach’s coefficient alpha. For more information about Cronbach’s coefficient alpha, see the section “Cronbach’s Coefficient Alpha” on page 101.

This option is not valid if a WITH statement is specified.

BEST=n

prints the n highest correlation coefficients for each variable, where n ≥ 1. Correlations are ordered from highest to lowest in absolute value. (If you do not specify this option, PROC CORRELATION prints correlations in a rectangular table, using the variable names as row and column labels.)

COV
displays the variance and covariance matrix. When you specify this option, the Pearson correlations are also displayed. If you also specify the OUTP= option, the output data table also contains the covariance matrix with the corresponding _TYPE_ variable value ‘COV.’

CSSCP
displays a table of the corrected sums of squares and crossproducts. When you specify this option, the Pearson correlations are also displayed. If you also specify the OUTP= option, the output data table also contains a CSSCP matrix with the corresponding _TYPE_ variable value ‘CSSCP.’

DATA=CAS-libref.data-table

names the input data table for PROC CORRELATION to use. The default is the most recently created data table. CAS-libref.data-table is a two-level name, where

CAS-libref refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to the data, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about CAS-libref, see the section “Using CAS Sessions and CAS Engine Librefs” on page 91.

data-table specifies the name of the input data table.

EXCLNPWGT
EXCLNPWGTS

excludes observations that have nonpositive weight values from the analysis. (By default, PROC CORRELATION treats observations that have negative weights like those observations that have zero weights and counts them in the total number of observations.)

NOCORR

suppresses display of Pearson correlations.
NOMISS excludes observations that have missing values from the analysis. (If you do not specify this option, PROC CORRELATION computes correlation statistics by using all the nonmissing pairs of variables.)

NOPROB suppresses display of the probabilities that are associated with each correlation coefficient.

NOSIMPLE suppresses printing of simple descriptive statistics for each variable. However, if you request an output data table, the output data table still contains simple descriptive statistics for the variables.

OUTP=\texttt{CAS-libref.data-table}

creates an output data table that contains Pearson correlation statistics. \texttt{CAS-libref.data-table} is a two-level name, where

\texttt{CAS-libref} refers to a collection of information that is defined in the LIBNAME statement and includes the \texttt{caslib}, which includes a path to where the data table is to be stored, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about \texttt{CAS-libref}, see the section “Using CAS Sessions and CAS Engine Librefs” on page 91.

\texttt{data-table} specifies the name of the output data table.

This data table also includes means, standard deviations, and the number of observations. If you also specify the ALPHA option, the output data table also contains six observations that have Cronbach’s coefficient alpha.

RANK displays the ordered correlation coefficients for each variable. Correlations are ordered from highest to lowest in absolute value.

SSCP displays a table of the sums of squares and crossproducts. When you specify this option, the Pearson correlations are also displayed. If you also specify the OUTP= option, the output data table contains an SSCP matrix and the corresponding \_TYPE\_ variable value is ‘SSCP.’

VARDEF=DF | N | WDF | WEIGHT | WGT

specifies the variance divisor in the calculation of variances and covariances. The default is \texttt{VARDEF}=DF.

Table 4.3 displays available values and associated divisors for the VARDEF= option, where \( n \) is the number of nonmissing observations and \( w_j \) is the weight associated with the \( j \)th nonmissing observation.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
<th>Divisor</th>
</tr>
</thead>
<tbody>
<tr>
<td>DF</td>
<td>Degrees of freedom</td>
<td>( n - 1 )</td>
</tr>
<tr>
<td>N</td>
<td>Number of observations</td>
<td>( n )</td>
</tr>
<tr>
<td>WDF</td>
<td>Sum of weights minus one</td>
<td>( \sum_j^n w_j - 1 )</td>
</tr>
<tr>
<td>WEIGHT</td>
<td>Sum of weights</td>
<td>( \sum_j^n w_j )</td>
</tr>
</tbody>
</table>
BY Statement

BY variables ;

You can specify a BY statement in PROC CORRELATION to obtain separate analyses of observations in groups that are defined by the values of the BY variables. If you specify more than one BY statement, only the last one specified is used. For more information, see the discussion of BY-group processing in SAS Language Reference: Concepts.

DISPLAY Statement

DISPLAY <table-list> < / options> ;

The DISPLAY statement enables you to specify a list of display tables to display or exclude. This statement is similar to the ODS SELECT, ODS EXCLUDE, and ODS TRACE statements. However, the DISPLAY statement can improve performance when a large number of tables could be generated (such as in BY-group processing). The procedure processes the DISPLAY statement on a CAS server and thus sends only a subset of ODS tables to the SAS client. Because ODS statements are processed on a SAS client, first all the generated display tables are sent to the client, and then the client creates a subset.

If you use both DISPLAY and ODS statements together, the DISPLAY statement takes precedence over the ODS statements. Note that the ODS EXCLUDE statement processes tables that are sent to the client after they have been filtered by the DISPLAY statement. In some cases, it might appear that the ODS EXCLUDE statement is taking precedence because it can further filter the tables. For more information about ODS, see SAS Output Delivery System: Procedures Guide.

You can specify the table-list as a list of table names, paths, partial pathnames, and regular expressions.

The table names that you can specify are listed in the section “ODS Table Names” on page 104. A path is a table name that is prefixed with dot-separated grouping information. For example, a SelectionSummary table that a procedure produces during a selection routine might have the path Bygroup1.Summary.SelectionSummary. A partial pathname does not include all groups; for example, SelectionSummary and Summary.SelectionSummary are partial pathnames for Bygroup1.Summary.SelectionSummary.

When you specify a table name or partial pathname, all display tables whose paths end in the specified name are selected for display or exclusion. For example, both SelectionSummary and Summary.SelectionSummary select Bygroup1.Summary.SelectionSummary.

A regular expression is enclosed in forward slashes (/). For example, specifying “/tions/” selects all pathnames that contain the substring “tions”; in particular, the Bygroup1.Summary.SelectionSummary table is selected. Specifying “!/tions/” selects all pathnames that do not contain the substring “tions”; in particular, the Bygroup1.Summary.SelectionSummary table is not selected.

You can specify the following options after a slash (/):

CASESENSITIVE

performs a case-sensitive comparison of table names in the table-list to display table names when tables are subsetted for display. To preserve case, you must enclose table names in the table-list in quotation marks.
EXCLUDE
displays all display tables except those that you specify in the table-list.

EXCLUDEALL
suppresses display of all tables. This option takes precedence over the other options.

TRACE
displays the display table names, labels, and paths.

DISPLAYOUT Statement

DISPLAYOUT table-spec-list < / options > ;

The DISPLAYOUT statement enables you to create CAS output tables from your displayed output. This statement is similar to the ODS OUTPUT statement. For more information about ODS, see SAS Output Delivery System: Procedures Guide.

The table-spec-list specifies a list of CAS output tables to create. Each entry in the list has either a key=value format or a key format:

key=value specifies key as the ODS table name, path, or partial pathname, and specifies value as the CAS output table name.

key specifies key as the ODS table name and also as the CAS output table name.

The ODS table names that you can specify are listed in the section “ODS Table Names” on page 104. You cannot specify the ODS table named OutputCasTables in the table-spec-list.

Table names and partial pathnames are discussed under the DISPLAY statement. The DISPLAYOUT statement does not support regular expressions.

You can specify the following options after a slash (/):

INCLUDEALL
creates output CAS tables for all display tables. The name of the created output CAS table is the same as the corresponding display table name. If you specify this option, the table-spec-list specification is ignored.

NOREPLACE
does not replace any existing CAS output table of the same name.

REPEATED
replicates all CAS output tables on all nodes.

FREQ Statement

FREQ variable ;

The variable in the FREQ statement identifies a numeric variable in the data set that contains the frequency of occurrence of each observation. PROC CORRELATION treats each observation as if it appears f times,
where \( f \) is the value of the FREQ variable for the observation. If \( f \) is not an integer, it is truncated to an integer. If \( f \) is less than 1 or missing, the observation is not used in the analysis. When the FREQ statement is not specified, each observation is assigned a frequency of 1.

The effects of the FREQ and WEIGHT statements are similar except when \( n \) is not an integer and when degrees of freedom are calculated.

---

**VAR Statement**

```
VAR variables ;
```

The VAR statement lists variables for which to compute correlation coefficients. If the VAR statement is not specified, PROC CORRELATION computes correlations for all numeric variables that are not listed in other statements.

---

**WEIGHT Statement**

```
WEIGHT variable ;
```

The variable in the WEIGHT statement is used in the calculation of Pearson weighted product-moment correlations.

Observations that have missing weights are excluded from the analysis. By default, for observations that have nonpositive weights, weights are set to 0 and the observations are included in the analysis. You can use the EXCLNPWGT option to exclude observations that have negative or zero weights from the analysis. If a WEIGHT statement is not included, all observations that are used in the analysis are assigned a weight of 1.

---

**WITH Statement**

```
WITH variables ;
```

The WITH statement lists variables for which to compute correlation coefficients. The WITH statement requests correlations of the form \( r(X_i, Y_j) \), where \( Y_1, \ldots, Y_n \) are the row analysis variables that are specified in the WITH statement and \( X_1, \ldots, X_m \) are the column analysis variables that are specified in the VAR statement; or, if there is no VAR statement, all numeric variables that are not specified in any other statement. The correlation matrix has a rectangular structure of the form

\[
\begin{bmatrix}
  r(Y_1, X_1) & \cdots & r(Y_1, X_m) \\
  \vdots & \ddots & \vdots \\
  r(Y_n, X_1) & \cdots & r(Y_n, X_m)
\end{bmatrix}
\]

For example, the statements

```
proc correlation data=mycas.data;
   var x1 x2;
   with y1 y2 y3;
run;
```
produce correlations for the following combinations:

\[
\begin{bmatrix}
  r(Y_1, X_1) & r(Y_1, X_2) \\
  r(Y_2, X_1) & r(Y_2, X_2) \\
  r(Y_3, X_1) & r(Y_3, X_2)
\end{bmatrix}
\]

### Details: CORRELATION Procedure

#### Pearson Product-Moment Correlation

The Pearson product-moment correlation is a parametric measure of association for two variables. It measures both the strength and the direction of a linear relationship. If one variable \(X\) is an exact linear function of another variable \(Y\), a positive relationship exists if the correlation is 1, and a negative relationship exists if the correlation is \(-1\). If there is no linear predictability between the two variables, the correlation is 0. If the two variables are normal with correlation 0, the two variables are independent. Correlation does not imply causality because an underlying causal relationship might not exist in some cases.

The formula for the population Pearson product-moment correlation, denoted \(\rho_{xy}\), is

\[
\rho_{xy} = \frac{\text{Cov}(x, y)}{\sqrt{\text{V}(x)\text{V}(y)}} = \frac{E((x - E(x))(y - E(y)))}{\sqrt{E(x - E(x))^2 E(y - E(y))^2}}
\]

The sample correlation, such as a Pearson product-moment correlation or weighted product-moment correlation, estimates the population correlation. The formula for the sample Pearson product-moment correlation is as follows, where \(\bar{x}\) is the sample mean of \(x\) and \(\bar{y}\) is the sample mean of \(y\):

\[
r_{xy} = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2 \sum_i (y_i - \bar{y})^2}}
\]

The formula for a weighted Pearson product-moment correlation is as follows, where \(w_i\) is the weight, \(\bar{x}_w\) is the weighted mean of \(x\), and \(\bar{y}_w\) is the weighted mean of \(y\):

\[
r_{xy} = \frac{\sum_i w_i(x_i - \bar{x}_w)(y_i - \bar{y}_w)}{\sqrt{\sum_i w_i(x_i - \bar{x}_w)^2 \sum_i w_i(y_i - \bar{y}_w)^2}}
\]

#### Probability Values

Probability values for the Pearson correlation are computed by treating the following equation as if it came from a \(t\) distribution with \((n - 2)\) degrees of freedom, where \(r\) is the sample correlation:

\[
t = (n - 2)^{1/2} \left( \frac{r^2}{1 - r^2} \right)^{1/2}
\]
Cronbach’s Coefficient Alpha

Analyzing latent constructs such as job satisfaction, motor ability, sensory recognition, or customer satisfaction requires instruments to accurately measure the constructs. Interrelated items can be summed to obtain an overall score for each participant. Cronbach’s coefficient alpha estimates the reliability of this type of scale by determining the internal consistency of the test or the average correlation of items within the test (Cronbach 1951).

When a value is recorded, the observed value contains some degree of measurement error. Two sets of measurements on the same variable for the same individual might not have identical values. However, repeated measurements for a series of individuals do show some consistency. Reliability measures internal consistency from one set of measurements to another. The observed value $Y$ is divided into two components, a true value $T$ and a measurement error $E$. The measurement error is assumed to be independent of the true value:

$$Y = T + E \quad \text{Cov}(T, E) = 0$$

The reliability coefficient of a measurement test is defined as the squared correlation between the observed value $Y$ and the true value $T$:

$$r^2(Y, T) = \frac{\text{Cov}(Y, T)^2}{\text{Var}(Y)\text{Var}(T)} = \frac{\text{Var}(T)^2}{\text{Var}(Y)\text{Var}(T)} = \frac{\text{Var}(T)}{\text{Var}(Y)}$$

This coefficient is the proportion of the observed variance due to true differences among individuals in the sample. If $Y$ is the sum of several observed variables that measure the same feature, you can estimate $\text{Var}(T)$. Cronbach’s coefficient alpha, which is based on a lower bound for $\text{Var}(T)$, is an estimate of the reliability coefficient.

Suppose $p$ variables are used with $Y_j = T_j + E_j$ for $j = 1, 2, \ldots, p$, where $Y_j$ is the observed value, $T_j$ is the true value, and $E_j$ is the measurement error. The measurement errors ($E_j$) are independent of the true values ($T_j$) and are also independent of each other. Let $Y_0 = \sum_j Y_j$ be the total observed score, and let $T_0 = \sum_j T_j$ be the total true score. Because

$$(p - 1) \sum_j \text{Var}(T_j) \geq \sum_{i \neq j} \text{Cov}(T_i, T_j)$$

a lower bound for $\text{Var}(T_0)$ is given by

$$\frac{p}{p-1} \sum_{i \neq j} \text{Cov}(T_i, T_j)$$

With $\text{Cov}(Y_i, Y_j) = \text{Cov}(T_i, T_j)$ for $i \neq j$, a lower bound for the reliability coefficient, $\text{Var}(T_0)/\text{Var}(Y_0)$, is then given by Cronbach’s coefficient alpha:

$$\alpha = \left( \frac{p}{p-1} \right) \frac{\sum_{i \neq j} \text{Cov}(Y_i, Y_j)}{\text{Var}(Y_0)} = \left( \frac{p}{p-1} \right) \left( 1 - \frac{\sum_j \text{Var}(Y_j)}{\text{Var}(Y_0)} \right)$$

If the variances of the items vary widely, you can standardize the items to a standard deviation of 1 before computing the coefficient alpha. If the variables are dichotomous (0,1), the coefficient alpha is equivalent to the Kuder-Richardson 20 (KR-20) reliability measure.
When the correlation between each pair of variables is 1, the coefficient alpha has a maximum value of 1. With negative correlations between some variables, the coefficient alpha can have a value less than 0. The larger the overall alpha coefficient, the more likely that items contribute to a reliable scale. Nunnally and Bernstein (1994) suggest 0.70 as an acceptable reliability coefficient; smaller reliability coefficients are seen as inadequate. However, acceptability varies by discipline.

To determine how each item reflects the reliability of the scale, you calculate a coefficient alpha after deleting each variable independently from the scale. Cronbach’s coefficient alpha from all variables except the $k$th variable is given by

$$\alpha_k = \left(1 - \frac{1}{p - 2} \frac{1}{p} \left(1 - \frac{\sum_{i \neq k} V(Y_i)}{V(\sum_{i \neq k} Y_i)}\right)\right)$$

If the reliability coefficient increases after an item is deleted from the scale, you can assume that the item is not highly correlated with other items in the scale. Conversely, if the reliability coefficient decreases, you can assume that the item is highly correlated with other items in the scale. For more information about how to interpret Cronbach’s coefficient alpha, see Yu (2001).

Listwise deletion of observations that have missing values is necessary to correctly calculate Cronbach’s coefficient alpha. PROC CORRELATION does not automatically use listwise deletion if you specify the ALPHA option. Therefore, you should use the NOMISS option if the data table contains missing values. Otherwise, PROC CORRELATION prints a warning message that indicates that you need to use the NOMISS option with the ALPHA option.

**Missing Values**

Observations that have missing values for the variables in the FREQ or WEIGHT statement are omitted from the analysis.

By default, PROC CORRELATION uses pairwise deletion when observations contain missing values in the analysis variables for which correlation coefficients are computed. PROC CORRELATION includes all nonmissing pairs of values for each pair of variables in the statistical computations. Therefore, the correlation statistics might be based on different numbers of observations.

If you specify the NOMISS option, PROC CORRELATION uses listwise deletion and excludes all observations that have missing values from the analysis. Therefore, the number of observations for each pair of variables is identical.

**OUTP= Data Table**

If you specify the OUTP= option, PROC CORRELATION creates an output data table that contains statistics for Pearson correlation.

The new data table contains the following variables:

- **BY** variables, if any
- **_TYPE_** variable, which identifies the type of observation
OUTP= Data Table

- _NAME_ variable, which identifies the variable that corresponds to a particular row of the correlation matrix
- INTERCEPT variable, which identifies variable sums when the SSCP option is specified
- the column analysis variables that are specified in the VAR statement or, if there is no VAR statement, all numeric variables that are not specified in any other statement

You can use a combination of the _TYPE_ and _NAME_ variables to identify the contents of an observation. The _NAME_ variable indicates which row of the correlation matrix the observation corresponds to. The values of the _TYPE_ variable are as follows:

- SSCP, uncorrected sums of squares and crossproducts
- CSSCP, corrected sums of squares and crossproducts
- COV, covariances
- MEAN, mean of each variable
- STD, standard deviation of each variable
- N, number of nonmissing observations for each variable
- SUMWGT, sum of the weights for each variable when a WEIGHT statement is specified
- CORR, correlation statistics for each variable

If you specify the SSCP option, the OUTP= data table includes an additional observation that contains intercept values. If you specify the ALPHA option, the OUTP= data table also includes observations that have the following _TYPE_ values:

- RAWALPHA, Cronbach’s coefficient alpha for raw variables
- STDALPHA, Cronbach’s coefficient alpha for standardized variables
- RAWALDEL, Cronbach’s coefficient alpha for raw variables after deleting one variable
- STDALDEL, Cronbach’s coefficient alpha for standardized variables after deleting one variable
- RAWCTDEL, the correlation between a raw variable and the total of the remaining raw variables
- STDCTDEL, the correlation between a standardized variable and the total of the remaining standardized variables

The output data table does not have a special type like the OUTP= data table that PROC CORR creates. Therefore, if the OUTP= output data table that PROC CORRELATION creates is provided as input to a SAS/STAT procedure that recognizes special data table types (such as the REG and FACTOR procedures), the input type must also be specified. For example, the following statements produce an output data table mycas.outdata:
proc correlation data=mycas.data outp=mycas.outdata;
run;

proc factor data=mycas.outdata (Type=CORR);
run;

The mycas.outdata data table is then provided to PROC FACTOR as input with the (Type=CORR) option to indicate that the input data table contains a correlation matrix. This might save computational time when the initial data table mycas.data is large and the computation of the correlation matrix can be done in CAS using multiple threads.

**ODS Table Names**

PROC CORRELATION assigns a name to each table it creates. You must use these names to refer to tables when you use the DISPLAY statement, the DISPLAYOUT, or the Output Delivery System (ODS). These names are listed in Table 4.4.

<table>
<thead>
<tr>
<th>ODS Table Name</th>
<th>Description</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cov</td>
<td>Covariances</td>
<td>COV</td>
</tr>
<tr>
<td>CronbachAlpha</td>
<td>Coefficient alpha</td>
<td>ALPHA</td>
</tr>
<tr>
<td>CronbachAlphaDel</td>
<td>Coefficient alpha with deleted variable</td>
<td>ALPHA</td>
</tr>
<tr>
<td>Csscp</td>
<td>Corrected sums of squares and crossproducts</td>
<td>CSSCP</td>
</tr>
<tr>
<td>PearsonCorr</td>
<td>Pearson correlations</td>
<td>Default output</td>
</tr>
<tr>
<td>SimpleStats</td>
<td>Simple descriptive statistics</td>
<td>Default output</td>
</tr>
<tr>
<td>Sscp</td>
<td>Sums of squares and crossproducts</td>
<td>SSCP</td>
</tr>
<tr>
<td>VarInformation</td>
<td>Variable information</td>
<td>Default output</td>
</tr>
</tbody>
</table>

**Examples: CORRELATION Procedure**

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

**Example 4.1: Computing Correlations between Two Sets of Variables**

The following statements create the data table Setosa, which contains measurements for four iris parts from Fisher’s iris data (1936): sepal length, sepal width, petal length, and petal width. The data table has been altered to contain some missing values.
Example 4.1: Computing Correlations between Two Sets of Variables

*------------------- Data on Iris Setosa --------------------*

The data table contains 50 iris specimens from the species Iris Setosa with the following four measurements:

- SepalLength (sepal length)
- SepalWidth (sepal width)
- PetalLength (petal length)
- PetalWidth (petal width)

Certain values were changed to missing for the analysis.

*-----------------------------------------------------------*

data mycas.Setosa;
input SepalLength SepalWidth PetalLength PetalWidth @@;
label sepallength='Sepal Length in mm.';
sepalwidth='Sepal Width in mm.';
petallength='Petal Length in mm.';
petalwidth='Petal Width in mm.';
datalines;
50 33 14 02 46 34 14 03 46 36 . 02
51 33 17 05 55 35 13 02 48 31 16 02
52 34 14 02 49 36 14 01 44 32 13 02
50 35 16 06 44 30 13 02 47 32 16 02
48 30 14 03 51 38 16 02 48 34 19 02
50 30 16 02 50 32 12 02 43 30 11 .
58 40 12 02 51 38 19 04 49 30 14 02
51 35 14 02 50 34 16 04 46 32 14 02
57 44 15 04 50 36 14 02 54 34 15 04
52 41 15 . 55 42 14 02 49 31 15 02
54 39 17 04 50 34 15 02 44 29 14 02
47 32 13 02 46 31 15 02 51 34 15 02
50 35 13 03 49 31 15 01 54 37 15 02
54 39 13 04 51 35 14 03 48 34 16 02
48 30 14 01 45 23 13 03 57 38 17 03
51 38 15 03 54 34 17 02 51 37 15 04
52 35 15 02 53 37 15 02
;

The following statements request a correlation analysis between two sets of variables, the sepal measurements (length and width) and the petal measurements (length and width):

title 'Fisher (1936) Iris Setosa Data';
proc correlation data=mycas.Setosa sscp cov;
  var sepallength sepalwidth;
  with petallength petalwidth;
run;

The “Simple Statistics” table in Output 4.1.1 displays univariate statistics for the analysis variables that are specified in the VAR and WITH statements.
Chapter 4: The CORRELATION Procedure

Output 4.1.1 Simple Statistics

Fisher (1936) Iris Setosa Data

The CORRELATION Procedure

<table>
<thead>
<tr>
<th>Variable</th>
<th>N</th>
<th>Mean</th>
<th>Std Dev</th>
<th>Sum</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>PetalWidth</td>
<td>48</td>
<td>2.52083</td>
<td>1.03121</td>
<td>121.00000</td>
<td>1.00000</td>
<td>6.00000</td>
<td>Petal Width in mm.</td>
</tr>
<tr>
<td>SepalLength</td>
<td>50</td>
<td>50.06000</td>
<td>3.52490</td>
<td>2503</td>
<td>43.00000</td>
<td>58.00000</td>
<td>Sepal Length in mm.</td>
</tr>
<tr>
<td>SepalWidth</td>
<td>50</td>
<td>34.28000</td>
<td>3.79064</td>
<td>1714</td>
<td>23.00000</td>
<td>44.00000</td>
<td>Sepal Width in mm.</td>
</tr>
</tbody>
</table>

When the WITH statement is specified together with the VAR statement, the CORRELATION procedure produces rectangular matrices for statistics such as covariances and correlations. The matrix rows correspond to the variables specified in the WITH statement (PetalLength and PetalWidth), and the matrix columns correspond to the variables specified in the VAR statement (SepalLength and SepalWidth). The CORRELATION procedure uses the WITH variable labels to label the matrix rows.

The SSCP option requests a table of the uncorrected sum of squares and crossproducts matrix, and the COV option requests a table of the covariance matrix.

The sum of squares and crossproducts statistics for each pair of variables are computed by using observations that have nonmissing row and column variable values. The “Sums of Squares and Crossproducts” table in Output 4.1.2 displays the crossproduct, the sum of squares for the row variable, and the sum of squares for the column variable for each pair of variables.

Output 4.1.2 Sums of Squares and Crossproducts

<table>
<thead>
<tr>
<th>Sums of Squares and Crossproducts</th>
<th>SSCP / Row Var SS / Col Var SS</th>
</tr>
</thead>
<tbody>
<tr>
<td>SepalLength</td>
<td>PetalLength</td>
</tr>
<tr>
<td></td>
<td>SepalWidth</td>
</tr>
<tr>
<td>PetalLength</td>
<td>36214.00000</td>
</tr>
<tr>
<td>Petal Length in mm.</td>
<td>10735.00000</td>
</tr>
<tr>
<td></td>
<td>123793.00000</td>
</tr>
<tr>
<td>PetalWidth</td>
<td>6113.00000</td>
</tr>
<tr>
<td>Petal Width in mm.</td>
<td>355.00000</td>
</tr>
<tr>
<td></td>
<td>121356.00000</td>
</tr>
</tbody>
</table>

The variances are computed by using observations that have nonmissing values for the analysis variables. The “Variances and Covariances” table in Output 4.1.3 displays the covariance, variance for the row variable, variance for the column variable, and associated degrees of freedom for each pair of variables.
Example 4.2: Computing Cronbach’s Coefficient Alpha

The following statements create the data table Fish1. The cubic root of the weight (Weight3) is computed as a one-dimensional measure of the size of a fish.

```
*------------------- Fish Measurement Data -----------------------*
| The data table contains 35 fish from the species Bream caught  |
| in Finland's Lake Laengelmavesi with the following measurements:|
| Weight (in grams)                                         |
| Length3 (length from the nose to the end of the tail, in cm)|
| HtPct (max height, as percentage of Length3)               |
| WidthPct (max width, as percentage of Length3)             |
*-----------------------------------------------------------------*;

data mycas.Fish1 (drop=HtPct WidthPct);
  title 'Fish Measurement Data';
  input Weight Length3 HtPct WidthPct @@;
  Weight3= Weight**(1/3);
  Height=HtPct*Length3/100;
  Width=WidthPct*Length3/100;
datalines;
242.0 30.0 38.4 13.4 290.0 31.2 40.0 13.8
340.0 31.1 39.8 15.1 363.0 33.5 38.0 13.3
```
The following statements request a correlation analysis and compute Cronbach’s coefficient alpha for the variables Weight3, Length3, Height, and Width:

```r
   title 'Fish Measurement Data';
   proc correlation data=mycas.fish1 nomiss alpha;
     var Weight3 Length3 Height Width;
   run;
```

The ALPHA option computes Cronbach’s coefficient alpha for the analysis variables.

The “Simple Statistics” table in Output 4.2.1 displays univariate descriptive statistics for each analysis variable.

```
   Output 4.2.1 Simple Statistics

   Fish Measurement Data

   The CORRELATION Procedure

       Simple Statistics
   Variable     N   Mean     Std Dev    Sum   Minimum   Maximum
    Weight3    34   8.44751    0.97574    287.21524   6.23168    10.00000
    Length3    34   38.38529    4.21628   1305.00000   46.50000
    Height    34  15.22057  1.98159  517.49950  11.52000  18.95700
    Width    34   5.43805    0.72967   184.89370   4.02000    6.74970
```

The “Pearson Correlation Coefficients” table in Output 4.2.2 displays Pearson correlation statistics for pairs of analysis variables.
Output 4.2.2  Pearson Correlation Coefficients

<table>
<thead>
<tr>
<th>Pearson Correlation Coefficients N = 34</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prob &gt;</td>
</tr>
<tr>
<td>Weight3</td>
</tr>
<tr>
<td>1.0000</td>
</tr>
<tr>
<td>&lt;.0001</td>
</tr>
<tr>
<td>0.9652</td>
</tr>
<tr>
<td>&lt;.0001</td>
</tr>
<tr>
<td>0.9626</td>
</tr>
<tr>
<td>&lt;.0001</td>
</tr>
<tr>
<td>0.9279</td>
</tr>
<tr>
<td>&lt;.0001</td>
</tr>
</tbody>
</table>

Because the data table contains only one species of fish, all the variables are highly correlated. Using the ALPHA option, the CORRELATION procedure computes Cronbach’s coefficient alpha in Output 4.2.3. The Cronbach’s coefficient alpha is a lower bound for the reliability coefficient for the raw variables and the standardized variables. Positive correlation is needed for the alpha coefficient because variables measure a common entity.

Output 4.2.3  Cronbach’s Coefficient Alpha

<table>
<thead>
<tr>
<th>Cronbach Coefficient Alpha with Deleted Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variables</td>
</tr>
<tr>
<td>Raw</td>
</tr>
<tr>
<td>Standardized</td>
</tr>
</tbody>
</table>

Because the variances of some variables vary widely, you should use the standardized score to estimate reliability. The overall standardized Cronbach’s coefficient alpha of 0.985145 provides an acceptable lower bound for the reliability coefficient. This is much greater than the suggested value of 0.70 given by Nunnally and Bernstein (1994).

The standardized alpha coefficient provides information about how each variable reflects the reliability of the scale with standardized variables. If the standardized alpha decreases after removing a variable from the construct, then this variable is strongly correlated with other variables in the scale. On the other hand, if the standardized alpha increases after removing a variable from the construct, then removing this variable from the scale makes the construct more reliable. The “Cronbach Coefficient Alpha with Deleted Variables” table in Output 4.2.4 does not show a significant increase or decrease in the standardized alpha coefficients. For more information about Cronbach’s alpha, see the section “Cronbach’s Coefficient Alpha” on page 101.

Output 4.2.4  Cronbach’s Coefficient Alpha with Deleted Variables

<table>
<thead>
<tr>
<th>Cronbach Coefficient Alpha with Deleted Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raw Variables</td>
</tr>
<tr>
<td>Correlation with Total</td>
</tr>
<tr>
<td>0.975379</td>
</tr>
<tr>
<td>0.967602</td>
</tr>
<tr>
<td>0.964715</td>
</tr>
<tr>
<td>0.934635</td>
</tr>
</tbody>
</table>
Example 4.3: Saving Correlations in an Output Data Table

The following statements compute Pearson correlations for the Fitness data table that is created in the section “Getting Started: CORRELATION Procedure” on page 92:

```sas
title 'Correlations for a Fitness and Exercise Study';
proc correlation data=mycas.Fitness nomiss outp=mycas.CorrOutp;
  var weight oxygen runtime;
run;
```

The NOMISS option excludes from the analysis observations that have missing values of the variables that are specified in the VAR statement—that is, the same set of 28 observations is used to compute the correlation for each pair of variables. The OUTP= option creates an output data table named CorrOutp to contain the Pearson correlation statistics.

The “Pearson Correlation Coefficients” table in Output 4.3.1 displays the correlation and the $p$-value under the null hypothesis of zero correlation.

**Output 4.3.1** Pearson Correlation Coefficients

**Correlations for a Fitness and Exercise Study**

**The CORRELATION Procedure**

<table>
<thead>
<tr>
<th>Pearson Correlation Coefficients N = 28</th>
<th>Prob &gt;</th>
<th>under H0: Rho=0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Weight</td>
<td>Oxygen</td>
</tr>
<tr>
<td>Weight</td>
<td>1.0000</td>
<td>-0.1842</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3481</td>
</tr>
<tr>
<td>Oxygen</td>
<td>-0.1842</td>
<td>1.0000</td>
</tr>
<tr>
<td></td>
<td>0.3481</td>
<td></td>
</tr>
<tr>
<td>RunTime</td>
<td>0.1950</td>
<td>-0.8684</td>
</tr>
<tr>
<td></td>
<td>0.3199</td>
<td></td>
</tr>
</tbody>
</table>

The following statements display the output data table that is shown in Output 4.3.2:

```sas
title 'Output Data Table from PROC CORRELATION';
proc print data=mycas.CorrOutp noobs;
run;
```

**Output 4.3.2** OUTP= Data Table with Pearson Correlations

**Output Data Table from PROC CORRELATION**

<table>
<thead>
<tr>
<th>TYPE</th>
<th>NAME</th>
<th>Weight</th>
<th>Oxygen</th>
<th>RunTime</th>
</tr>
</thead>
<tbody>
<tr>
<td>MEAN</td>
<td></td>
<td>77.2168</td>
<td>47.1327</td>
<td>10.6954</td>
</tr>
<tr>
<td>STD</td>
<td></td>
<td>8.4495</td>
<td>5.5535</td>
<td>1.4127</td>
</tr>
<tr>
<td>N</td>
<td></td>
<td>28.0000</td>
<td>28.0000</td>
<td>28.0000</td>
</tr>
<tr>
<td>CORR</td>
<td>Weight</td>
<td>1.0000</td>
<td>-0.1842</td>
<td>0.1950</td>
</tr>
<tr>
<td>CORR</td>
<td>Oxygen</td>
<td>-0.1842</td>
<td>1.0000</td>
<td>-0.8684</td>
</tr>
<tr>
<td>CORR</td>
<td>RunTime</td>
<td>0.1950</td>
<td>-0.8684</td>
<td>1.0000</td>
</tr>
</tbody>
</table>
The output data table can be used as an input data table for regression or other statistical procedures. For example, the following statements request a regression analysis that uses CorrOutp instead of reading the original data in the REG procedure:

```sas
title 'Input Type CORR Data Set from PROC REG';
proc reg data=mycas.CorrOutp(type=CORR);
   model runtime= weight oxygen;
run;
```

The following statements read the original data and generate the same results as the preceding statements:

```sas
proc reg data=mycas.Fitness;
   model runtime= weight oxygen;
run;
```

References


Chapter 5
The FREQTAB Procedure

Contents

Overview: FREQTAB Procedure .................................................. 114
PROC FREQTAB Features ......................................................... 115
PROC FREQTAB Compared with the FREQ Procedure ................. 115
Using CAS Sessions and CAS Engine Librefs ............................... 116
Getting Started: FREQTAB Procedure ..................................... 117
Syntax: FREQTAB Procedure ................................................... 123
PROC FREQTAB Statement ...................................................... 123
BY Statement ................................................................. 126
EXACT Statement .......................................................... 126
Statistic Options ........................................................... 127
Computation Options ......................................................... 134
OUTPUT Statement .......................................................... 136
TABLES Statement .......................................................... 147
Without Options ............................................................ 148
Options ................................................................. 149
TEST Statement ........................................................... 193
WEIGHT Statement .......................................................... 197
Details: FREQTAB Procedure ................................................. 198
Inputting Frequency Counts .................................................. 198
Missing Values .............................................................. 198
Statistical Computations ...................................................... 201
Definitions and Notation ....................................................... 201
Chi-Square Tests and Statistics .............................................. 202
Measures of Association ..................................................... 208
Binomial Proportion .......................................................... 217
Sensitivity and Specificity .................................................... 225
Risks and Risk Differences .................................................. 226
Common Risk Difference .................................................... 237
Odds Ratio and Relative Risks for 2 \times 2 Tables ......................... 241
Cochran-Armitage Test for Trend .......................................... 251
Jonckheere-Terpstra Test ...................................................... 252
Tests and Measures of Agreement ......................................... 254
Cochran-Mantel-Haenszel Statistics ........................................ 261
Gail-Simon Test for Qualitative Interactions ............................. 271
Exact Statistics .............................................................. 271
Output Data Sets ............................................................. 276
Overview: FREQTAB Procedure

The FREQTAB procedure produces one-way to n-way frequency and crosstabulation (contingency) tables in SAS Viya. PROC FREQTAB also provides a variety of tests and measures to analyze frequency and crosstabulation tables.

For one-way frequency tables, PROC FREQTAB provides chi-square goodness-of-fit tests for equal proportions and for specified null proportions. It also provides several types of confidence limits for binomial proportions and binomial proportion tests (which include noninferiority and equivalence tests).

For two-way crosstabulation tables, PROC FREQTAB computes a variety of statistics to examine the relationships between the two classification variables. These statistics include the following:

- chi-square tests and measures
- measures and tests of association
- measures and tests of agreement
- tests for trend
- Cochran-Mantel-Haenszel statistics

For 2 x 2 tables, PROC FREQTAB provides several confidence limit types and tests for the following statistics:

- risk (proportion) differences
- odds ratios
- relative risks
For multiway \((n\text{-way})\) tables, PROC FREQTAB performs stratified analysis by computing statistics within strata (two-way tables) and across strata. These statistics include Cochran-Mantel-Haenszel statistics and measures of agreement. For multiway \(2 \times 2\) tables, PROC FREQTAB provides confidence limits and tests for the common (stratified) risk difference, odds ratio, and relative risks.

By default, PROC FREQTAB computes asymptotic standard errors, confidence limits, and tests. Exact tests and confidence limits are also available for many statistics and measures.

**PROC FREQTAB Features**

The FREQTAB procedure offers the following functionality:

- provides TABLES statements to specify frequency and crosstabulation tables
- provides options for statistical analyses of frequency and crosstabulation tables
- provides an EXACT statement for exact tests and confidence limits
- provides a TEST statement for tests of measures of association and agreement
- provides a WEIGHT statement for weighted analyses
- produces ODS tables of tabulation and statistical analysis results
- uses ODS graphics to produce statistical plots as part of the output

Because the FREQTAB procedure constructs the frequency and crosstabulation tables on CAS, it also does the following:

- enables you to construct tables on a cluster of machines that distribute the data and the computations
- enables you to construct tables in single-machine mode on CAS
- exploits all the available cores and concurrent threads for table construction. For information about how PROC FREQTAB uses threads, see the section “Multithreading” on page 81 in Chapter 3, “Shared Concepts.”

**PROC FREQTAB Compared with the FREQ Procedure**

The FREQTAB procedure provides tabulation and statistical analysis capabilities that are very similar to the capabilities of the FREQ procedure in SAS/STAT software. Table 5.1 shows the major functional similarities and differences between these two procedures.
Table 5.1 Comparison of PROC FREQTAB and PROC FREQ

<table>
<thead>
<tr>
<th>Feature</th>
<th>PROC FREQTAB</th>
<th>PROC FREQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>Provides statistical analysis of frequency</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>and crosstabulation tables</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Provides OUT=, OUTPUT, and ODS output data</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>sets</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Threading</td>
<td>Tabulation designed for CAS; executes on multiple threads</td>
<td>Executes on a single thread</td>
</tr>
<tr>
<td>Supports MISSING option</td>
<td>In PROC FREQTAB statement</td>
<td>In TABLES statements</td>
</tr>
<tr>
<td>Supports SPARSE option</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Supports CHISQ(TESTP=SAS-data-set)</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Using CAS Sessions and CAS Engine Librefs

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:

```sas
proc options option=(CASHOST CASPORT);
run;
```

In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:

```sas
cas mysess;
libname mycas cas sessref=mysess;
```

The CAS statement creates the CAS session named `mysess`, and the LIBNAME statement creates the `mycas` CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the CASHOST and CASPORT of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the `mysess` session, you can terminate it by using the TERMINATE option in the CAS statement as follows:

```sas
cas mysess terminate;
```

For more information about the CAS and LIBNAME statements, see the section “Introduction to Shared Concepts” on page 10 in Chapter 3, “Shared Concepts.”
Getting Started: FREQTAB Procedure

This example shows you how to use the FREQTAB procedure to construct crosstabulation tables and produce chi-square statistics and measures of association.

The SAS data table mycas.School_Survey consists of hypothetical data that are based on a customer satisfaction survey for a school information system. This data table includes the following variables for the survey respondents: ID (respondent identification), Response, SchoolType, and State. Figure 5.1 displays the first 20 observations of the data table.

**Figure 5.1 School_Survey Data Table (First 20 Observations)**

<table>
<thead>
<tr>
<th>Obs</th>
<th>ID</th>
<th>Response</th>
<th>SchoolType</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>26629</td>
<td>Satisfied</td>
<td>Middle School (Public)</td>
<td>TN</td>
</tr>
<tr>
<td>2</td>
<td>25834</td>
<td>Neutral</td>
<td>High School (Public)</td>
<td>TN</td>
</tr>
<tr>
<td>3</td>
<td>14821</td>
<td>Very Satisfied</td>
<td>High School (Private)</td>
<td>NC</td>
</tr>
<tr>
<td>4</td>
<td>4153</td>
<td>Neutral</td>
<td>Middle School (Private)</td>
<td>GA</td>
</tr>
<tr>
<td>5</td>
<td>6185</td>
<td>Very Satisfied</td>
<td>High School (Private)</td>
<td>GA</td>
</tr>
<tr>
<td>6</td>
<td>6164</td>
<td>Unsatisfied</td>
<td>High School (Public)</td>
<td>GA</td>
</tr>
<tr>
<td>7</td>
<td>6386</td>
<td>Neutral</td>
<td>Middle School (Public)</td>
<td>GA</td>
</tr>
<tr>
<td>8</td>
<td>1011</td>
<td>Neutral</td>
<td>High School (Private)</td>
<td>GA</td>
</tr>
<tr>
<td>9</td>
<td>28916</td>
<td>Very Satisfied</td>
<td>High School (Private)</td>
<td>TN</td>
</tr>
<tr>
<td>10</td>
<td>10601</td>
<td>Neutral</td>
<td>High School (Public)</td>
<td>NC</td>
</tr>
<tr>
<td>11</td>
<td>17900</td>
<td>Satisfied</td>
<td>High School (Private)</td>
<td>NC</td>
</tr>
<tr>
<td>12</td>
<td>2935</td>
<td>Unsatisfied</td>
<td>High School (Private)</td>
<td>GA</td>
</tr>
<tr>
<td>13</td>
<td>9799</td>
<td>Unsatisfied</td>
<td>High School (Private)</td>
<td>NC</td>
</tr>
<tr>
<td>14</td>
<td>23754</td>
<td>Very Satisfied</td>
<td>Middle School (Public)</td>
<td>TN</td>
</tr>
<tr>
<td>15</td>
<td>9516</td>
<td>Neutral</td>
<td>Middle School (Public)</td>
<td>NC</td>
</tr>
<tr>
<td>16</td>
<td>4614</td>
<td>Very Unsatisfied</td>
<td>Middle School (Private)</td>
<td>GA</td>
</tr>
<tr>
<td>17</td>
<td>12443</td>
<td>Unsatisfied</td>
<td>Middle School (Public)</td>
<td>NC</td>
</tr>
<tr>
<td>18</td>
<td>5356</td>
<td>Very Satisfied</td>
<td>High School (Public)</td>
<td>GA</td>
</tr>
<tr>
<td>19</td>
<td>15189</td>
<td>Unsatisfied</td>
<td>High School (Private)</td>
<td>NC</td>
</tr>
<tr>
<td>20</td>
<td>3309</td>
<td>Satisfied</td>
<td>High School (Public)</td>
<td>GA</td>
</tr>
</tbody>
</table>

**Note:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

The following PROC FREQTAB statements request a two-way crosstabulation of SchoolType and Response. These statements assume that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

```sas
proc freqtab data=mycas.School_Survey;
tables SchoolType * Response / crosslist chisq measures(cl);
run;
```

The PROC FREQTAB statement invokes the procedure and identifies the data table to be analyzed. The TABLES statement requests a two-way crosstabulation of SchoolType and Response. The table request syntax in PROC FREQTAB is identical to the table request syntax in the FREQ procedure. You can
specify one-way, two-way, and multiway table requests. You can specify more than one table request in the same TABLES statement, and you can specify multiple TABLES statements in the same invocation of the procedure.

The CROSSLIST option displays the crosstabulation table in column format; by default, PROC FREQTAB displays crosstabulation tables in table cell format. The CHISQ option requests chi-square tests, and the MEASURES option requests measures of association for this two-way crosstabulation table. The MEASURES(CL) option requests confidence limits for the measures of association.

Figure 5.2 shows the crosstabulation table of SchoolType by Response. You can customize the contents of this table by specifying options in the TABLES statement. For example, the NOCOL option suppresses the column percentages, the EXPECTED option displays the expected cell frequencies (under independence), and the STDRES option displays the standardized residuals.

<table>
<thead>
<tr>
<th>SchoolType</th>
<th>Response</th>
<th>Frequency</th>
<th>Percent</th>
<th>Row Percent</th>
<th>Column Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Middle School (Private)</td>
<td>Very Satisfied</td>
<td>594</td>
<td>1.83</td>
<td>15.77</td>
<td>12.05</td>
</tr>
<tr>
<td></td>
<td>Satisfied</td>
<td>823</td>
<td>2.54</td>
<td>21.85</td>
<td>11.32</td>
</tr>
<tr>
<td></td>
<td>Neutral</td>
<td>1183</td>
<td>3.64</td>
<td>31.41</td>
<td>12.51</td>
</tr>
<tr>
<td></td>
<td>Unsatisfied</td>
<td>846</td>
<td>2.61</td>
<td>22.46</td>
<td>11.16</td>
</tr>
<tr>
<td></td>
<td>Very Unsatisfied</td>
<td>320</td>
<td>0.99</td>
<td>8.50</td>
<td>9.93</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>3766</td>
<td>11.60</td>
<td>100.00</td>
<td></td>
</tr>
<tr>
<td>Middle School (Public)</td>
<td>Very Satisfied</td>
<td>1261</td>
<td>3.88</td>
<td>13.94</td>
<td>25.58</td>
</tr>
<tr>
<td></td>
<td>Satisfied</td>
<td>1811</td>
<td>5.58</td>
<td>20.02</td>
<td>24.91</td>
</tr>
<tr>
<td></td>
<td>Neutral</td>
<td>2494</td>
<td>7.68</td>
<td>27.56</td>
<td>26.37</td>
</tr>
<tr>
<td></td>
<td>Unsatisfied</td>
<td>2364</td>
<td>7.28</td>
<td>26.13</td>
<td>31.18</td>
</tr>
<tr>
<td></td>
<td>Very Unsatisfied</td>
<td>1118</td>
<td>3.44</td>
<td>12.36</td>
<td>34.70</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>9048</td>
<td>27.88</td>
<td>100.00</td>
<td></td>
</tr>
<tr>
<td>High School (Private)</td>
<td>Very Satisfied</td>
<td>1459</td>
<td>4.49</td>
<td>18.23</td>
<td>29.60</td>
</tr>
<tr>
<td></td>
<td>Satisfied</td>
<td>2160</td>
<td>6.65</td>
<td>26.99</td>
<td>29.72</td>
</tr>
<tr>
<td></td>
<td>Neutral</td>
<td>2422</td>
<td>7.46</td>
<td>30.26</td>
<td>25.61</td>
</tr>
<tr>
<td></td>
<td>Unsatisfied</td>
<td>1363</td>
<td>4.20</td>
<td>17.03</td>
<td>17.98</td>
</tr>
<tr>
<td></td>
<td>Very Unsatisfied</td>
<td>599</td>
<td>1.85</td>
<td>7.48</td>
<td>18.59</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>8003</td>
<td>24.66</td>
<td>100.00</td>
<td></td>
</tr>
<tr>
<td>High School (Public)</td>
<td>Very Satisfied</td>
<td>1615</td>
<td>4.98</td>
<td>13.87</td>
<td>32.77</td>
</tr>
<tr>
<td></td>
<td>Satisfied</td>
<td>2475</td>
<td>7.63</td>
<td>21.26</td>
<td>34.05</td>
</tr>
<tr>
<td></td>
<td>Neutral</td>
<td>3359</td>
<td>10.35</td>
<td>28.85</td>
<td>35.51</td>
</tr>
<tr>
<td></td>
<td>Unsatisfied</td>
<td>3008</td>
<td>9.27</td>
<td>25.84</td>
<td>39.68</td>
</tr>
<tr>
<td></td>
<td>Very Unsatisfied</td>
<td>1185</td>
<td>3.65</td>
<td>10.18</td>
<td>36.78</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>11642</td>
<td>35.87</td>
<td>100.00</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>Very Satisfied</td>
<td>4929</td>
<td>15.19</td>
<td>100.00</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Satisfied</td>
<td>7269</td>
<td>22.39</td>
<td>100.00</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Neutral</td>
<td>9458</td>
<td>29.14</td>
<td>100.00</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Unsatisfied</td>
<td>7581</td>
<td>23.36</td>
<td>100.00</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Very Unsatisfied</td>
<td>3222</td>
<td>9.93</td>
<td>100.00</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>32459</td>
<td>100.00</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Figure 5.3 shows the chi-square statistics, and Figure 5.4 shows the measures of association for the table of SchoolType and Response.

**Figure 5.3 Chi-Square Statistics**

Statistics for Table of SchoolType by Response

<table>
<thead>
<tr>
<th>Statistic</th>
<th>DF</th>
<th>Value</th>
<th>Prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chi-Square</td>
<td>12</td>
<td>505.1446</td>
<td>&lt;.0001</td>
</tr>
<tr>
<td>Likelihood Ratio Chi-Square</td>
<td>12</td>
<td>512.2769</td>
<td>&lt;.0001</td>
</tr>
<tr>
<td>Mantel-Haenszel Chi-Square</td>
<td>1</td>
<td>0.0050</td>
<td>0.9438</td>
</tr>
<tr>
<td>Phi Coefficient</td>
<td></td>
<td>0.1247</td>
<td></td>
</tr>
<tr>
<td>Contingency Coefficient</td>
<td></td>
<td>0.1238</td>
<td></td>
</tr>
<tr>
<td>Cramer's V</td>
<td></td>
<td>0.0720</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 5.4 Measures of Association**

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Value</th>
<th>ASE</th>
<th>95% Confidence Limits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gamma</td>
<td>0.0024</td>
<td>0.0061</td>
<td>-0.0097</td>
</tr>
<tr>
<td>Kendall's Tau-b</td>
<td>0.0018</td>
<td>0.0046</td>
<td>-0.0073</td>
</tr>
<tr>
<td>Stuart's Tau-c</td>
<td>0.0018</td>
<td>0.0046</td>
<td>-0.0072</td>
</tr>
<tr>
<td>Somers' D C</td>
<td>R</td>
<td>0.0018</td>
<td>0.0048</td>
</tr>
<tr>
<td>Somers' D R</td>
<td>C</td>
<td>0.0017</td>
<td>0.0044</td>
</tr>
<tr>
<td>Pearson Correlation</td>
<td>-0.0004</td>
<td>0.0055</td>
<td>-0.0112</td>
</tr>
<tr>
<td>Spearman Correlation</td>
<td>0.0018</td>
<td>0.0056</td>
<td>-0.0090</td>
</tr>
<tr>
<td>Lambda Asymmetric C</td>
<td>R</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>Lambda Asymmetric R</td>
<td>C</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>Lambda Symmetric</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>Uncertainty Coefficient C</td>
<td>R</td>
<td>0.0051</td>
<td>0.0004</td>
</tr>
<tr>
<td>Uncertainty Coefficient R</td>
<td>C</td>
<td>0.0060</td>
<td>0.0005</td>
</tr>
<tr>
<td>Uncertainty Coefficient Symmetric</td>
<td>0.0055</td>
<td>0.0005</td>
<td>0.0046</td>
</tr>
</tbody>
</table>

Sample Size = 32459
PROC FREQTAB can produce plots as part of its output by using ODS Graphics. The following statements show how to produce a frequency plot and a mosaic plot for a two-way crosstabulation table. The ODS GRAPHICS ON statement enables ODS Graphics, which is required in order to produce plots.

The PLOTS=FREQPLOT option in the TABLES statement requests a frequency plot of the two-way table of Response by SchoolType. The TWOWAY=CLUSTER plot-option requests a cluster grouped bar chart. The ORIENT=H plot-option specifies horizontal orientation, which places the variable levels on the Y-axis and the percentages on the X-axis. The SCALE=GROUPPERCENT plot-option requests a plot of group (column) percentages (instead of frequencies).

The PLOTS=MOSAICPLOT option requests a mosaic plot of Response by SchoolType. The COLORSTAT=STDRES plot-option colors the mosaic plot tiles according to the standardized residuals of the corresponding table cells.

```sas
ods graphics on;
proc freqtab data=mycas.School_Survey;
  tables Response * SchoolType /
    plots=(freqplot(twoway=cluster orient=h scale=grouppercent)
           mosaicplot(colorstat=stdres));
run;
ods graphics off;
```

Figure 5.5 displays the two-way frequency plot of SchoolType and Response. Figure 5.6 displays the mosaic plot of this crosstabulation.

**Figure 5.5** Frequency Plot (Grouped Bar Chart)
Figure 5.6 Mosaic Plot

The following PROC FREQTAB statements produce a multiway table of State by SchoolType by Response. The TABLES statement requests the multiway table in which SchoolType is the row variable, Response is the column variable, and State is the stratum variable. This request produces a separate two-way table of SchoolType by Response for each level (stratum) of the variable State. The CMH option in the TABLES statement requests stratified analysis, which adjusts for the effect of State.

```plaintext
proc freqtab data=mycas.School_Survey;
   tables State * SchoolType * Response / cmh;
run;
```

Figure 5.7 displays the two-way table of SchoolType and Response for State = ‘GA’. PROC FREQTAB also produces two-way tables for the other states, but these results are not displayed here. Figure 5.8 displays the CMH summary statistics.
**Figure 5.7** Two-Way Table for State=GA

The FREQTAB Procedure

<table>
<thead>
<tr>
<th>SchoolType</th>
<th>Very Satisfied</th>
<th>Satisfied</th>
<th>Neutral</th>
<th>Unsatisfied</th>
<th>Very Unsatisfied</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Middle School (Private)</td>
<td>146</td>
<td>193</td>
<td>293</td>
<td>243</td>
<td>82</td>
<td>957</td>
</tr>
<tr>
<td></td>
<td>1.77</td>
<td>2.34</td>
<td>3.56</td>
<td>2.95</td>
<td>1.00</td>
<td>11.63</td>
</tr>
<tr>
<td>Middle School (Public)</td>
<td>281</td>
<td>406</td>
<td>625</td>
<td>631</td>
<td>295</td>
<td>2238</td>
</tr>
<tr>
<td></td>
<td>3.41</td>
<td>4.93</td>
<td>7.59</td>
<td>7.67</td>
<td>3.58</td>
<td>27.19</td>
</tr>
<tr>
<td>High School (Private)</td>
<td>369</td>
<td>537</td>
<td>651</td>
<td>357</td>
<td>174</td>
<td>2088</td>
</tr>
<tr>
<td></td>
<td>4.48</td>
<td>6.52</td>
<td>7.91</td>
<td>4.34</td>
<td>2.11</td>
<td>25.37</td>
</tr>
<tr>
<td>High School (Public)</td>
<td>404</td>
<td>611</td>
<td>866</td>
<td>762</td>
<td>305</td>
<td>2948</td>
</tr>
<tr>
<td></td>
<td>4.91</td>
<td>7.42</td>
<td>10.52</td>
<td>9.26</td>
<td>3.71</td>
<td>35.82</td>
</tr>
<tr>
<td>Total</td>
<td>1200</td>
<td>1747</td>
<td>2435</td>
<td>1993</td>
<td>856</td>
<td>8231</td>
</tr>
<tr>
<td></td>
<td>14.58</td>
<td>21.22</td>
<td>29.58</td>
<td>24.21</td>
<td>10.40</td>
<td>100.00</td>
</tr>
</tbody>
</table>

**Figure 5.8** CMH Test of No Association

The FREQTAB Procedure

Summary Statistics for SchoolType by Response
Controlling for State

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Alternative Hypothesis</th>
<th>DF</th>
<th>Value</th>
<th>Prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Nonzero Correlation</td>
<td>1</td>
<td>0.0086</td>
<td>0.9262</td>
<td></td>
</tr>
<tr>
<td>2 Row Mean Scores Differ</td>
<td>3</td>
<td>406.7786</td>
<td>&lt;.0001</td>
<td></td>
</tr>
<tr>
<td>3 General Association</td>
<td>12</td>
<td>508.7656</td>
<td>&lt;.0001</td>
<td></td>
</tr>
</tbody>
</table>

Total Sample Size = 32459
Syntax: FREQTAB Procedure

The following statements are available in the FREQTAB procedure:

```
PROC FREQTAB <options> ;
   BY variables ;
   EXACT statistic-options </ computation-options> ;
   OUTPUT <OUT=SAS-data-set> output-options ;
   TABLES requests </ options> ;
   TEST options ;
   WEIGHT variable </ option> ;
```

The PROC FREQTAB statement is the only required statement for the FREQTAB procedure. If you specify the following statements, PROC FREQTAB produces a one-way frequency table for each variable in the most recently created data table.

```
proc freqtab;
   run;
```

Table 5.2 summarizes the basic functions of the procedure statements. The following sections provide detailed syntax information for the BY, EXACT, OUTPUT, TABLES, TEST, and WEIGHT statements in alphabetical order after the description of the PROC FREQTAB statement.

<table>
<thead>
<tr>
<th>Statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BY</td>
<td>Provides separate analyses for each BY group</td>
</tr>
<tr>
<td>EXACT</td>
<td>Requests exact tests</td>
</tr>
<tr>
<td>OUTPUT</td>
<td>Requests an output data set of statistics</td>
</tr>
<tr>
<td>TABLES</td>
<td>Specifies tables and requests analyses</td>
</tr>
<tr>
<td>TEST</td>
<td>Requests tests for measures of association and agreement</td>
</tr>
<tr>
<td>WEIGHT</td>
<td>Identifies a weight variable</td>
</tr>
</tbody>
</table>

PROC FREQTAB Statement

```
PROC FREQTAB <options> ;
```

The PROC FREQTAB statement invokes the FREQTAB procedure. Optionally, it also identifies the input data table. By default, the procedure uses the most recently created SAS data table.

Table 5.3 lists the options available in the PROC FREQTAB statement. Descriptions of the options follow in alphabetical order.
Table 5.3 PROC FREQTAB Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPRESS</td>
<td>Begins the next one-way table on the current page</td>
</tr>
<tr>
<td>DATA=</td>
<td>Names the input data table</td>
</tr>
<tr>
<td>FORMCHAR=</td>
<td>Specifies outline and cell-divider characters for</td>
</tr>
<tr>
<td></td>
<td>cell-format crosstabulation tables</td>
</tr>
<tr>
<td>MISSING</td>
<td>Treats missing values as a valid level</td>
</tr>
<tr>
<td>MISSPRINT</td>
<td>Displays missing levels</td>
</tr>
<tr>
<td>NLEVELS</td>
<td>Displays the number of levels for each TABLES</td>
</tr>
<tr>
<td></td>
<td>statement variable</td>
</tr>
<tr>
<td>NOPRINT</td>
<td>Suppresses all displayed output</td>
</tr>
<tr>
<td>ORDER=</td>
<td>Specifies the order for reporting variable values</td>
</tr>
<tr>
<td>PAGE</td>
<td>Displays one table per page</td>
</tr>
</tbody>
</table>

You can specify the following *options* in the PROC FREQTAB statement.

**COMPRESS**

begins display of the next one-way frequency table on the same page as the preceding one-way table if there is enough space to begin the table. By default, the next one-way table begins on the current page only if the entire table fits on that page. The COMPRESS option is not valid with the PAGE option.

**DATA=** `CAS-libref.data-table`

names the input data table for PROC FREQTAB to use. The default is the most recently created data table. `CAS-libref.data-table` is a two-level name, where

- `CAS-libref` refers to a collection of information that is defined in the LIBNAME statement and includes the `caslib`, which includes a path to the data, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about `CAS-libref`, see the section “Using CAS Sessions and CAS Engine Librefs” on page 116.

- `data-table` specifies the name of the input data table.

**FORMCHAR(1,2,7)=’formchar-string’**

defines the characters to use for constructing the outlines and dividers for the cells in crosstabulation tables that are displayed in the default, table-cell format. The *formchar-string* should be three characters long. The characters are used to draw the vertical separators (1), the horizontal separators (2), and the vertical-horizontal intersections (7). If you do not specify the FORMCHAR= option, PROC FREQTAB uses FORMCHAR(1,2,7)=’|+-’ by default. Table 5.4 summarizes the formatting characters used by PROC FREQTAB.

Table 5.4 Formatting Characters Used by PROC FREQTAB

<table>
<thead>
<tr>
<th>Position</th>
<th>Default</th>
<th>Used to Draw</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>Vertical separators</td>
</tr>
<tr>
<td>2</td>
<td>-</td>
<td>Horizontal separators</td>
</tr>
<tr>
<td>7</td>
<td>+</td>
<td>Intersections of vertical and horizontal separators</td>
</tr>
</tbody>
</table>
The FORMCHAR= option can specify 20 different SAS formatting characters used to display output; however, PROC FREQTAB uses only the first, second, and seventh formatting characters. Therefore, the proper specification for PROC FREQTAB is FORMCHAR(1,2,7)= ‘formchar-string’.

Specifying all blanks for formchar-string produces crosstabulation tables with no outlines or dividers—for example, FORMCHAR(1,2,7)=’ ’. You can use any character in formchar-string, including hexadecimal characters. If you use hexadecimal characters, you must put an x after the closing quote. For information about which hexadecimal codes to use for which characters, see the documentation for your hardware.

MISSING treats missing values as a valid nonmissing level for all classification variables that you specify in the TABLES statements. This option displays missing levels in frequency and crosstabulation tables and includes the frequencies (of missing levels) in the computation of the frequency total, percentages, and statistics.

By default (if you do not specify the MISSING or MISSPRINT option), PROC FREQTAB excludes an observation from a table if the observation contains a missing value for any variable in the table request. For more information, see the section “Missing Values” on page 198.

MISSPRINT displays missing levels in frequency tables but does not include the corresponding frequencies in the computation of totals, percentages, or statistics. This option is available for one-way frequency tables and for tables in list format (which you can request by specifying the LIST option in the TABLES statement).

By default (if you do not specify the MISSING or MISSPRINT option), PROC FREQTAB excludes an observation from a table if the observation contains a missing value for any variable in the table request. For more information, see the section “Missing Values” on page 198.

NLEVELS displays the “Number of Variable Levels” table, which provides the number of levels for each variable that you name in a TABLES statement. PROC FREQTAB determines the variable levels by using the formatted variable values.

NOPRINT suppresses the display of all output. You can use the NOPRINT option when you only want to create an output data set. For more information about the output data sets that PROC FREQTAB produces, see the section “Output Data Sets” on page 276. The NOPRINT option temporarily disables the Output Delivery System (ODS). For more information, see the SAS Output Delivery System: User’s Guide.

NOTE: A NOPRINT option is also available in the TABLES statement. This option suppresses display of the crosstabulation tables but allows display of the statistics tables.

ORDER=FORMATTED | FREQ | INTERNAL specifies the order of the variable levels in the frequency and crosstabulation tables, which you request in the TABLES statement.

The ORDER= option can take the following values:
### Value of ORDER= Levels Ordered By

<table>
<thead>
<tr>
<th>ORDER=</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FORMATTED</td>
<td>External formatted value (in ascending order)</td>
</tr>
<tr>
<td>FREQ</td>
<td>Descending frequency count</td>
</tr>
<tr>
<td>INTERNAL</td>
<td>Unformatted value</td>
</tr>
</tbody>
</table>

By default, ORDER=INTERNAL. The FORMATTED and INTERNAL orders are machine-dependent. When you specify ORDER=FORMATTED, numeric variables that have no explicit format are sorted by their unformatted (internal) value.

When you specify ORDER=FREQ, levels that have larger frequencies (more observations) appear first in the order. ORDER=FREQ is not available when you specify a WEIGHT statement.

The ORDER= option does not apply to missing values, which are always ordered first.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Language Reference: Concepts*.

### PAGE

PAGE displays only one table per page. Otherwise, PROC FREQTAB displays multiple tables per page as space permits. The PAGE option is not valid with the COMPRESS option.

### BY Statement

BY variables ;

You can specify a BY statement in PROC FREQTAB to obtain separate analyses of observations in groups that are defined by the values of the BY variables. If you specify more than one BY statement, only the last one specified is used. For more information, see the discussion of BY-group processing in *SAS Language Reference: Concepts*.

### EXACT Statement

EXACT statistic-options < / computation-options> ;

The EXACT statement requests exact tests and confidence limits for selected statistics. The statistic-options identify which statistics to compute, and the computation-options specify options for computing exact statistics. For more information, see the section “Exact Statistics” on page 271.

**NOTE:** PROC FREQTAB computes exact tests by using fast and efficient algorithms that are superior to direct enumeration. Exact tests are appropriate when a data set is small, sparse, skewed, or heavily tied. For some large problems, computation of exact tests might require a considerable amount of time and memory. Consider using asymptotic tests for such problems. Alternatively, when asymptotic methods might not be sufficient for such large problems, consider using Monte Carlo estimation of exact \( p \)-values. You can request Monte Carlo estimation by specifying the MC computation-option in the EXACT statement. See the section “Computational Resources” on page 274 for more information.
Statistic Options

The statistic-options specify which exact tests and confidence limits to compute. Table 5.5 lists the available statistic-options and the exact statistics that are computed. Descriptions of the statistic-options follow the table in alphabetical order.

For one-way tables, exact $p$-values are available for binomial proportion tests, the chi-square goodness-of-fit test, and the likelihood ratio chi-square test. Exact (Clopper-Pearson) confidence limits are available for the binomial proportion.

For two-way tables, exact $p$-values are available for the following tests: Pearson chi-square test, likelihood ratio chi-square test, Mantel-Haenszel chi-square test, Fisher’s exact test, Jonckheere-Terpstra test, Cochran-Armitage test for trend, and Bowker’s symmetry test. Exact $p$-values are also available for tests of the following statistics: Pearson correlation coefficient, Spearman correlation coefficient, Kendall’s tau-$b$, Stuart’s tau-$c$, Somers’ $D(C|R)$, Somers’ $D(R|C)$, simple kappa coefficient, and weighted kappa coefficient.

For $2 \times 2$ tables, PROC FREQTAB provides the exact McNemar’s test, exact confidence limits for the odds ratio, and Barnard’s unconditional exact test for the risk (proportion) difference. PROC FREQTAB also provides exact unconditional confidence limits for the risk difference and for the relative risk (ratio of proportions). For stratified $2 \times 2$ tables, PROC FREQTAB provides Zelen’s exact test for equal odds ratios, exact confidence limits for the common odds ratio, and an exact test for the common odds ratio.

Most of the statistic-option names listed in Table 5.5 are identical to the corresponding option names in the TABLES and OUTPUT statements. You can request exact computations for groups of statistics by using statistic-options that are identical to the TABLES statement options CHISQ, MEASURES, and AGREE. For example, when you specify the CHISQ statistic-option in the EXACT statement, PROC FREQTAB computes exact $p$-values for the Pearson chi-square, likelihood ratio chi-square, and Mantel-Haenszel chi-square tests for two-way tables. You can request an exact test for an individual statistic by specifying the corresponding statistic-option from the list in Table 5.5.

Using the EXACT Statement with the TABLES Statement

You must use a TABLES statement with the EXACT statement. If you use only one TABLES statement, you do not need to specify the same options in both the TABLES and EXACT statements; when you specify a statistic-option in the EXACT statement, PROC FREQTAB automatically invokes the corresponding TABLES statement option. However, when you use an EXACT statement with multiple TABLES statements, you must specify options in the TABLES statements to request statistics. PROC FREQTAB then provides exact tests or confidence limits for those statistics that you also specify in the EXACT statement.

### Table 5.5 EXACT Statement Statistic Options

<table>
<thead>
<tr>
<th>Statistic Option</th>
<th>Exact Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGREE</td>
<td>McNemar’s test (for $2 \times 2$ tables), simple kappa test, weighted kappa test</td>
</tr>
<tr>
<td>BARNARD</td>
<td>Barnard’s test (for $2 \times 2$ tables)</td>
</tr>
<tr>
<td>BINOMIAL</td>
<td>Binomial proportion tests for one-way tables</td>
</tr>
<tr>
<td>CHISQ</td>
<td>Chi-square goodness-of-fit test for one-way tables; Pearson chi-square, likelihood ratio chi-square, and Mantel-Haenszel chi-square tests for two-way tables</td>
</tr>
<tr>
<td>COMOR</td>
<td>Confidence limits for the common odds ratio, common odds ratio test (for $h \times 2 \times 2$ tables)</td>
</tr>
</tbody>
</table>
You can specify the following **statistic-options**:

**AGREE**

requests McNemar’s exact test, an exact test for the simple kappa coefficient, and an exact test for the weighted kappa coefficient. For more information, see the sections “Tests and Measures of Agreement” on page 254 and “Exact Statistics” on page 271.

For McNemar’s test, you can specify the null hypothesis ratio of discordant proportions by using the \texttt{AGREE(MNULLRATIO=)} option in the \texttt{TABLES} statement; by default, \texttt{MNULLRATIO=1}. For the weighted kappa coefficient, you can request Fleiss-Cohen weights by specifying the \texttt{AGREE(WT=FC)} option in the \texttt{TABLES} statement; by default, \texttt{PROC FREQTAB} computes the weighted kappa coefficient by using Cicchetti-Allison agreement weights.

McNemar’s test is available for $2 \times 2$ tables. Kappa coefficients are defined only for square two-way tables, where the number of rows equals the number of columns. If your table is not square because some observations have weights of 0, you can specify the \texttt{ZEROS} option in the \texttt{WEIGHT} statement to include these observations in the analysis. For more information, see the section “Tables with Zero-Weight Rows or Columns” on page 261.

For $2 \times 2$ tables, the weighted kappa coefficient is equivalent to the simple kappa coefficient, and \texttt{PROC FREQTAB} displays only analyses for the simple kappa coefficient.
BARNARD
requests Barnard’s exact unconditional test for the risk (proportion) difference for $2 \times 2$ tables. For more information, see the section “Barnard’s Unconditional Exact Test” on page 236.

To request exact unconditional confidence limits for the risk difference, you can specify the RISKDIFF option in the EXACT statement. The RISKDIFF option in the TABLES statement provides asymptotic tests and several types of confidence limits for the risk difference. For more information, see the section “Risks and Risk Differences” on page 226.

BINOMIAL
BIN
requests an exact test for the binomial proportion (for one-way tables). For more information, see the section “Binomial Tests” on page 221. You can specify the null hypothesis proportion by using the BINOMIAL(P=) option in the TABLES statement; by default, P=0.5.

The BINOMIAL option in the TABLES statement provides exact (Clopper-Pearson) confidence limits for the binomial proportion by default. You can specify the BINOMIAL(CL=MIDP) option in the TABLES statement to request exact mid-$p$ confidence limits for the binomial proportion. The BINOMIAL option in the TABLES statement also provides asymptotic (Wald) tests and several other confidence limit types for the binomial proportion. For more information, see the section “Binomial Proportion” on page 217.

CHISQ
requests the following exact chi-square tests for two-way tables: Pearson chi-square, likelihood ratio chi-square, and Mantel-Haenszel chi-square. For more information, see the section “Chi-Square Tests and Statistics” on page 202. The CHISQ option in the TABLES statement provides asymptotic tests for these statistics.

For one-way tables, the CHISQ option requests an exact chi-square goodness-of-fit test. You can specify null hypothesis proportions for this test by using the CHISQ(TESTP=) option in the TABLES statement. By default, the one-way chi-square test is based on the null hypothesis of equal proportions. For more information, see the section “Chi-Square Test for One-Way Tables” on page 203.

COMOR
requests an exact test and exact confidence limits for the common odds ratio for multiway $2 \times 2$ tables. For more information, see the section “Exact Confidence Limits for the Common Odds Ratio” on page 269. The CMH option in the TABLES statement provides Mantel-Haenszel and logit estimates of the common odds ratio along with their asymptotic confidence limits.

EQOR
ZELEN
requests Zelen’s exact test for equal odds ratios for multiway $2 \times 2$ tables. For more information, see the section “Zelen’s Exact Test for Equal Odds Ratios” on page 268. The CMH option in the TABLES statement provides an (asymptotic) Breslow-Day test for homogeneity of odds ratios.

FISHER
requests Fisher’s exact test. For more information, see the sections “Fisher’s Exact Test” on page 206 and “Exact Statistics” on page 271. For $2 \times 2$ tables, the CHISQ option in the TABLES statement provides Fisher’s exact test. For general $R \times C$ tables, Fisher’s exact test is also known as the Freeman-Halton test.
JT
requests an exact Jonckheere-Terpstra test. For more information, see the sections “Jonckheere-Terpstra Test” on page 252 and “Exact Statistics” on page 271. The JT option in the TABLES statement provides an asymptotic Jonckheere-Terpstra test.

KAPPA
requests an exact test for the simple kappa coefficient. For more information, see the sections “Simple Kappa Coefficient” on page 255 and “Exact Statistics” on page 271. The AGREE option in the TABLES statement provides the simple kappa estimate, standard error, and confidence limits. The KAPPA option in the TEST statement provides an asymptotic test for the simple kappa coefficient.

Kappa coefficients are defined only for square two-way tables, where the number of rows equals the number of columns. If your table is not square because some observations have weights of 0, you can specify the ZEROS option in the WEIGHT statement to include these observations in the analysis. For more information, see the section “Tables with Zero-Weight Rows or Columns” on page 261.

KENTB
requests an exact test for Kendall’s tau-b. For more information, see the sections “Kendall’s Tau-b” on page 210 and “Exact Statistics” on page 271. The MEASURES option in the TABLES statement provides an estimate and standard error of Kendall’s tau-b. The KENTB option in the TEST statement provides an asymptotic test for Kendall’s tau-b.

LRCHI
requests an exact test for the likelihood ratio chi-square for two-way tables. For more information, see the sections “Likelihood Ratio Chi-Square Test” on page 205 and “Exact Statistics” on page 271. The CHISQ option in the TABLES statement provides an asymptotic likelihood ratio chi-square test for two-way tables.

For one-way tables, the LRCHI option requests an exact likelihood ratio goodness-of-fit test. You can specify null hypothesis proportions by using the CHISQ(TESTP=) option in the TABLES statement. By default, the one-way test is based on the null hypothesis of equal proportions. For more information, see the section “Likelihood Ratio Chi-Square Test for One-Way Tables” on page 204.

MCNEM
requests an exact McNemar’s test. For more information, see the sections “McNemar’s Test” on page 254 and “Exact Statistics” on page 271. You can specify the null hypothesis ratio of discordant proportions by using the AGREE(MNULLRATIO=) option in the TABLES statement; by default, MNULLRATIO=1. The AGREE option in the TABLES statement provides an asymptotic McNemar’s test.

MEASURES
requests exact tests for the Pearson and Spearman correlations. For more information, see the sections “Pearson Correlation Coefficient” on page 211, “Spearman Rank Correlation Coefficient” on page 212, and “Exact Statistics” on page 271. The PCORR and SCORR options in the TEST statement provide asymptotic tests for the Pearson and Spearman correlations, respectively.

The MEASURES option also requests exact confidence limits for the odds ratio for 2 x 2 tables. For more information, see the subsection Exact Confidence Limits in the section “Confidence Limits for the Odds Ratio” on page 242. You can also request exact confidence limits for the odds ratio by specifying the OR option in the EXACT statement.
MHCHI requests an exact test for the Mantel-Haenszel chi-square. For more information, see the sections “Mantel-Haenszel Chi-Square Test” on page 205 and “Exact Statistics” on page 271. The CHISQ option in the TABLES statement provides an asymptotic Mantel-Haenszel chi-square test.

OR

ODDSRATIO requests exact confidence limits for the odds ratio for $2 \times 2$ tables. For more information, see the subsection “Exact Confidence Limits” in the section “Confidence Limits for the Odds Ratio” on page 242.

You can request exact mid-$p$ confidence limits for the odds ratio by specifying the OR(CL=MIDP) option in the TABLES statement. The OR(CL=) option in the TABLES statement also provides other types of confidence limits for the odds ratio. For more information, see the section “Confidence Limits for the Odds Ratio” on page 242.

The ALPHA= option in the TABLES statement determines the confidence level of the exact confidence limits; by default, ALPHA=0.05, which produces 95% confidence limits for the odds ratio.

PCHI requests an exact test for the Pearson chi-square for two-way tables. For more information, see the sections “Pearson Chi-Square Test for Two-Way Tables” on page 203 and “Exact Statistics” on page 271. The CHISQ option in the TABLES statement provides an asymptotic Pearson chi-square test.

For one-way tables, the PCHI option requests an exact chi-square goodness-of-fit test. You can specify null hypothesis proportions by using the CHISQ(TESTP=) option in the TABLES statement. By default, the goodness-of-fit test is based on the null hypothesis of equal proportions. For more information, see the section “Chi-Square Test for One-Way Tables” on page 203.

PCORR requests an exact test for the Pearson correlation coefficient. For more information, see the sections “Pearson Correlation Coefficient” on page 211 and “Exact Statistics” on page 271. The MEASURES option in the TABLES statement provides the estimate and standard error of the Pearson correlation. The PCORR option in the TEST statement provides an asymptotic test for the Pearson correlation.

RELRISK <(options)>

requests exact unconditional confidence limits for the relative risk for $2 \times 2$ tables. By default (beginning in SAS/STAT 14.3), the exact confidence limits are computed by inverting two separate one-sided exact tests that are based on the score statistic (Chan and Zhang 1999). For more information, see the subsection “Exact Unconditional Confidence Limits” in the section “Confidence Limits for the Relative Risk” on page 245.

The RELRISK(CL=) option in the TABLES statement provides additional types of confidence limits for the relative risk. For more information, see the section “Confidence Limits for the Risk Difference” on page 228.

The ALPHA= option in the TABLES statement determines the confidence level; by default, ALPHA=0.05, which produces 95% confidence limits for the relative risk.

You can specify the following options:
Chapter 5: The FREQTAB Procedure

COLUMN=1 | 2 | BOTH
specifies the table column of the relative risk. By default, COLUMN=1, which provides exact confidence limits for the column 1 relative risk. COLUMN=BOTH provides exact confidence limits for both column 1 and column 2 relative risks.

METHOD=NOSCORE | SCORE | SCORE2
specifies the computation method for the exact confidence limits. By default, METHOD=SCORE. You can specify one of the following methods:

NOSCORE
computes the exact confidence limits by inverting two separate one-sided exact tests that are based on the unstandardized relative risk (Santner and Snell 1980). For more information, see the subsection “Exact Unconditional Confidence Limits” in the section “Confidence Limits for the Relative Risk” on page 245. This method is the default in releases before SAS/STAT 14.3.

SCORE
computes the exact confidence limits by inverting two separate one-sided exact tests that are based on the score statistic (Chan and Zhang 1999). For more information, see the subsection “Exact Unconditional Confidence Limits” in the section “Confidence Limits for the Relative Risk” on page 245. This method is the default beginning in SAS/STAT 14.3.

SCORE2
computes the exact confidence limits by inverting a single two-sided exact test that is based on the score statistic (Agresti and Min 2001). For more information, see the subsection “Exact Unconditional Confidence Limits” in the section “Confidence Limits for the Relative Risk” on page 245.

RISKDIFF < (options)>
requests exact unconditional confidence limits for the risk difference for $2 \times 2$ tables. By default (beginning in SAS/STAT 14.3), the exact confidence limits are computed by inverting two separate one-sided exact tests that are based on the score statistic (Chan and Zhang 1999). For more information, see the subsection “Exact Unconditional Confidence Limits” in the section “Confidence Limits for the Risk Difference” on page 228.

The RISKDIFF(CL=) option in the TABLES statement provides additional types of confidence limits for the risk difference. For more information, see the section “Confidence Limits for the Risk Difference” on page 228.

The ALPHA= option in the TABLES statement determines the confidence level; by default, ALPHA=0.05, which produces 95% confidence limits for the risk difference.

You can specify the following options:

COLUMN=1 | 2 | BOTH
specifies the table column of the risk difference. By default, COLUMN=BOTH and the exact confidence limits are displayed in the ‘Risk Estimates’ tables. If you specify the RISKDIFF(NORISKS) option in the TABLES statement to suppress the ‘Risk Estimates’ tables, COLUMN=1 by default and the exact confidence limits are displayed in the ‘Risk Difference Confidence Limits’ table.
METHOD=NOSCORE | SCORE | SCORE2

specifies the computation method for the exact confidence limits. By default, METHOD=SCORE.

You can specify one of the following methods:

NOSCORE

computes the exact confidence limits by inverting two separate one-sided exact tests that are based on the unstandardized risk difference (Santner and Snell 1980). For more information, see the subsection “Exact Unconditional Confidence Limits” in the section “Confidence Limits for the Risk Difference” on page 228. This method is the default in releases before SAS/STAT 14.3.

SCORE

computes the exact confidence limits by inverting two separate one-sided exact tests that are based on the score statistic (Chan and Zhang 1999). For more information, see the subsection “Exact Unconditional Confidence Limits” in the section “Confidence Limits for the Risk Difference” on page 228. This method is the default beginning in SAS/STAT 14.3.

SCORE2

computes the exact confidence limits by inverting a single two-sided exact test that is based on the score statistic (Agresti and Min 2001). For more information, see the subsection “Exact Unconditional Confidence Limits” in the section “Confidence Limits for the Risk Difference” on page 228.

SCORR

requests an exact test for the Spearman correlation coefficient. For more information, see the sections “Spearman Rank Correlation Coefficient” on page 212 and “Exact Statistics” on page 271. The MEASURES option in the TABLES statement provides the estimate and standard error of the Spearman correlation. The SCORR option in the TEST statement provides an asymptotic test for the Spearman correlation.

SMDCR

requests an exact test for Somers’ $D(C | R)$. For more information, see the sections “Somers’ $D$” on page 211 and “Exact Statistics” on page 271. The MEASURES option in the TABLES statement provides the estimate and standard error of Somers’ $D(C | R)$. The SMDCR option in the TEST statement provides an asymptotic test for Somers’ $D(C | R)$.

SMDRC

requests an exact test for Somers’ $D(R | C)$. For more information, see the sections “Somers’ $D$” on page 211 and “Exact Statistics” on page 271. The MEASURES option in the TABLES statement provides the estimate and standard error of Somers’ $D(R | C)$. The SMDRC option in the TEST statement provides an asymptotic test for Somers’ $D(C | R)$.

STUTC

requests an exact test for Stuart’s tau-c. For more information, see the sections “Stuart’s Tau-c” on page 210 and “Exact Statistics” on page 271. The MEASURES option in the TABLES statement provides the estimate and standard error of Stuart’s tau-c. The STUTC option in the TEST statement provides an asymptotic test for Stuart’s tau-c.
Chapter 5: The FREQTAB Procedure

SYMMETRY
BOWKER
requests an exact symmetry test. This test is available for square $R \times R$ two-way tables where the table dimension $R$ is greater than 2. For more information, see the section “Exact Symmetry Test” on page 255. The AGREE option in the TABLES statement provides an asymptotic symmetry test.

TREND
requests the exact Cochran-Armitage test for trend. For more information, see the sections “Cochran-Armitage Test for Trend” on page 251 and “Exact Statistics” on page 271. The TREND option in the TABLES statement provides an asymptotic Cochran-Armitage test for trend. This test is available for tables of dimensions $2 \times C$ or $R \times 2$.

WTKAPPA
WTKAP
requests an exact test for the weighted kappa coefficient. For more information, see the sections “Weighted Kappa Coefficient” on page 257 and “Exact Statistics” on page 271. By default, PROC FREQTAB computes the weighted kappa coefficient by using Cicchetti-Allison agreement weights. You can request Fleiss-Cohen agreement weights by specifying the AGREE(WT=FC) option in the TABLES statement.

Kappa coefficients are defined only for square two-way tables, where the number of rows equals the number of columns. If your table is not square because some observations have weights of 0, you can specify the ZEROS option in the WEIGHT statement to include these observations in the analysis. For more information, see the section “Tables with Zero-Weight Rows or Columns” on page 261.

For $2 \times 2$ tables, the weighted kappa coefficient is equivalent to the simple kappa coefficient, and PROC FREQTAB displays only analyses for the simple kappa coefficient.

Computation Options

The computation-options specify options for computing exact statistics. You can specify the following computation-options in the EXACT statement after a slash (/).

ALPHA=$\alpha$
specifies the level of the confidence limits for Monte Carlo $p$-value estimates. The value of $\alpha$ must be between 0 and 1; a confidence level of $\alpha$ produces $100(1 - \alpha)\%$ confidence limits. By default ALPHA=0.01, which produces 99% confidence limits for the Monte Carlo estimates.

The ALPHA= option invokes the MC option.

MAXTIME=value
specifies the maximum clock time (in seconds) that PROC FREQTAB can use to compute an exact $p$-value. If the procedure does not complete the computation within the specified time, the computation terminates. The MAXTIME= value must be a positive number. This option is available for Monte Carlo estimation of exact $p$-values, in addition to direct exact $p$-value computation. For more information, see the section “Computational Resources” on page 274.

MC
requests Monte Carlo estimation of exact $p$-values instead of direct exact $p$-value computation. Monte Carlo estimation can be useful for large problems that require a considerable amount of time and
memory for exact computations but for which asymptotic approximations might not be sufficient. For more information, see the section “Monte Carlo Estimation” on page 275.

This option is available for all EXACT statistic-options except the BINOMIAL option and the following options that apply only to $2 \times 2$ or $h \times 2 \times 2$ tables: BARNARD, COMOR, EQOR, MCNEM, OR, RELRISK, and RISKDIFF. PROC FREQTAB always computes exact tests or confidence limits (not Monte Carlo estimates) for these statistics.

The ALPHA=, N=, and SEED= options invoke the MC option.

**MIDP**

requests exact mid $p$-values for the exact tests. The exact mid $p$-value is defined as the exact $p$-value minus half the exact point probability. For more information, see the section “Definition of $p$-Values” on page 273.

The MIDP option is available for all EXACT statement statistic-options except the following: BARNARD, EQOR, OR, RELRISK, and RISKDIFF. You cannot specify both the MIDP option and the MC option.

**N=n**

specifies the number of samples for Monte Carlo estimation. The value of $n$ must be a positive integer, and the default is 10,000. Larger values of $n$ produce more precise estimates of exact $p$-values. Because larger values of $n$ generate more samples, the computation time increases.

The N= option invokes the MC option.

**PFORMAT=**format-name | EXACT

specifies the display format for exact $p$-values. PROC FREQTAB applies this format to one- and two-sided exact $p$-values, exact point probabilities, and exact mid $p$-values. By default, PROC FREQTAB displays exact $p$-values in the PVALUE6.4 format.

You can provide a format-name or you can specify PFORMAT=EXACT to control the format of exact $p$-values. The value of format-name can be any standard SAS numeric format or a user-defined format. The format length must not exceed 24. For information about formats, see the FORMAT procedure in the Base SAS Procedures Guide and the FORMAT statement and SAS format in SAS Formats and Informats: Reference.

If you specify PFORMAT=EXACT, PROC FREQTAB uses the 6.4 format to display exact $p$-values that are greater than or equal to 0.001; the procedure uses the E10.3 format to display values that are between 0.000 and 0.001. This is the format that PROC FREQTAB uses to display exact $p$-values in releases before SAS/STAT 12.3. Beginning in SAS/STAT 12.3, by default PROC FREQTAB uses the PVALUE6.4 format to display exact $p$-values.

**POINT**

requests exact point probabilities for the exact tests. The exact point probability is the exact probability that the test statistic equals the observed value. For more information, see the section “Definition of $p$-Values” on page 273.

The POINT option is available for all EXACT statement statistic-options except the following: BARNARD, EQOR, OR, RELRISK, and RISKDIFF. You cannot specify both the POINT option and the MC option.
SEED=number

specifies the initial seed for random number generation for Monte Carlo estimation. The value of the SEED= option must be an integer. If you do not specify the SEED= option or if the SEED= value is negative or 0, PROC FREQTAB uses the time of day from the computer’s clock to obtain the initial seed.

The SEED= option invokes the MC option.

### OUTPUT Statement

**OUTPUT OUT=SAS-data-set > output-options ;**

The OUTPUT statement creates a SAS data set that contains statistics that are computed by PROC FREQTAB. Table 5.6 lists the statistics that can be stored in the output data set. You identify which statistics to include by specifying output-options.

You must use a TABLES statement with the OUTPUT statement. The OUTPUT statement stores statistics for only one table request. If you use multiple TABLES statements, the contents of the output data set correspond to the last TABLES statement. If you use multiple table requests in a single TABLES statement, the contents of the output data set correspond to the last table request. Only one OUTPUT statement is allowed in a single invocation of the procedure.

For a one-way or two-way table, the output data set contains one observation that stores the requested statistics for the table. For a multiway table, the output data set contains an observation for each two-way table (stratum) of the multiway crosstabulation. If you request summary statistics for the multiway table, the output data set also contains an observation that stores the across-strata summary statistics. If you use a BY statement, the output data set contains an observation or set of observations for each BY group. For more information about the contents of the output data set, see the section “Contents of the OUTPUT Statement Output Data Set” on page 277.

The output data set that is created by the OUTPUT statement is not the same as the output data set that is created by the OUT= option in the TABLES statement. The OUTPUT statement creates a data set that contains statistics (such as the Pearson chi-square and its p-value), and the OUT= option in the TABLES statement creates a data set that contains frequency table counts and percentages. See the section “Output Data Sets” on page 276 for more information.

As an alternative to the OUTPUT statement, you can use the Output Delivery System (ODS) to store statistics that PROC FREQTAB computes. ODS can create a SAS data set from any table that PROC FREQTAB produces. See the section “ODS Table Names” on page 288 for more information.

You can specify the following options in the OUTPUT statement:

**OUT=SAS-data-set**

specifies the name of the output data set. When you use an OUTPUT statement but do not use the OUT= option, PROC FREQTAB creates a data set and names it by using the DATA$n$ convention.

**output-options**

specify the statistics to include in the output data set. Table 5.6 lists the output-options that are available in the OUTPUT statement, together with the TABLES statement options that are required to produce the statistics. Descriptions of the output-options follow the table in alphabetical order.
You can specify output-options to request individual statistics, or you can request groups of statistics by using output-options that are identical to the group options in the TABLES statement (for example, the CHISQ, MEASURES, CMH, AGREE, and ALL options).

When you specify an output-option, the output data set includes statistics from the corresponding analysis. In addition to the estimate or test statistic, the output data set includes associated values such as standard errors, confidence limits, p-values, and degrees of freedom. For more information, see the section “Contents of the OUTPUT Statement Output Data Set” on page 277.

To store a statistic in the output data set, you must also request computation of that statistic with the appropriate TABLES, EXACT, or TEST statement option. For example, the PCHI output-option includes the Pearson chi-square in the output data set. You must also request computation of the Pearson chi-square by specifying the CHISQ option in the TABLES statement. Or, if you use only one TABLES statement, you can request computation of the Pearson chi-square by specifying the PCHI or CHISQ option in the EXACT statement. Table 5.6 lists the TABLES statement options that are required to produce the OUTPUT data set statistics.

### Table 5.6 OUTPUT Statement Output Options

<table>
<thead>
<tr>
<th>Output Option</th>
<th>Output Data Set Statistics</th>
<th>Required TABLES Statement Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGREE</td>
<td>McNemar’s test (2 × 2 tables), Bowker’s test, simple and weighted kappas; for multiple strata, overall simple and weighted kappas, tests for equal kappas, and Cochran’s Q (h × 2 × 2 tables)</td>
<td>AGREE</td>
</tr>
<tr>
<td>AJCHI</td>
<td>Continuity-adjusted chi-square (2 × 2 tables)</td>
<td>CHISQ</td>
</tr>
<tr>
<td>ALL</td>
<td>CHISQ, MEASURES, and CMH statistics; N (number of nonmissing observations)</td>
<td>ALL</td>
</tr>
<tr>
<td>BDCHI</td>
<td>Breslow-Day test (h × 2 × 2 tables)</td>
<td>CMH, CMH1, or CMH2</td>
</tr>
<tr>
<td>BINOMIAL</td>
<td>Binomial statistics (one-way tables)</td>
<td>BINOMIAL</td>
</tr>
<tr>
<td>CHISQ</td>
<td>For one-way tables, goodness-of-fit test; for two-way tables, Pearson, likelihood ratio, continuity-adjusted, and Mantel-Haenszel chi-squares, Fisher’s exact test (2 × 2 tables), phi and contingency coefficients, Cramér’s V</td>
<td>CHISQ</td>
</tr>
<tr>
<td>CMH</td>
<td>Cochran-Mantel-Haenszel (CMH) correlation, row mean scores (ANOVA), and general association statistics; for 2 × 2 tables, logit and Mantel-Haenszel common odds ratios and relative risks, Breslow-Day test</td>
<td>CMH</td>
</tr>
<tr>
<td>CMH1</td>
<td>CMH statistics, except row mean scores (ANOVA) and general association statistics</td>
<td>CMH or CMH1</td>
</tr>
<tr>
<td>CMH2</td>
<td>CMH statistics, except general association statistic</td>
<td>CMH or CMH2</td>
</tr>
<tr>
<td>CMHCOR</td>
<td>CMH correlation statistic</td>
<td>CMH, CMH1, or CMH2</td>
</tr>
<tr>
<td>CMHGA</td>
<td>CMH general association statistic</td>
<td>CMH</td>
</tr>
<tr>
<td>CMHRMS</td>
<td>CMH row mean scores (ANOVA) statistic</td>
<td>CMH or CMH2</td>
</tr>
<tr>
<td>COCHQ</td>
<td>Cochran’s Q (h × 2 × 2 tables)</td>
<td>AGREEE</td>
</tr>
<tr>
<td>CONTGY</td>
<td>Contingency coefficient</td>
<td>CHISQ</td>
</tr>
<tr>
<td>Output Option</td>
<td>Output Data Set Statistics</td>
<td>Required TABLES Statement Option</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>CRAMV</td>
<td>Cramér’s $V$</td>
<td>CHISQ</td>
</tr>
<tr>
<td>EQKAP</td>
<td>Test for equal simple kappas</td>
<td>AGREE</td>
</tr>
<tr>
<td>EQOR</td>
<td>ZELEN</td>
<td>Zelen’s test for equal odds ratios ($h \times 2 \times 2$ tables)</td>
</tr>
<tr>
<td>EQWKP</td>
<td>Test for equal weighted kappas</td>
<td>AGREE</td>
</tr>
<tr>
<td>FISHER</td>
<td>Fisher’s exact test</td>
<td>CHISQ or FISHER $^1$</td>
</tr>
<tr>
<td>GAMMA</td>
<td>Gamma</td>
<td>MEASURES</td>
</tr>
<tr>
<td>GS</td>
<td>GAILSIMON</td>
<td>Gail-Simon test</td>
</tr>
<tr>
<td>JT</td>
<td>Jonckheere-Terpstra test</td>
<td>JT</td>
</tr>
<tr>
<td>KAPPA</td>
<td>Simple kappa coefficient</td>
<td>AGREE</td>
</tr>
<tr>
<td>KENTB</td>
<td>TAUB</td>
<td>Kendall’s $tau$-b</td>
</tr>
<tr>
<td>LAMCR</td>
<td>Lambda asymmetric ($C \mid R$)</td>
<td>MEASURES</td>
</tr>
<tr>
<td>LAMDAS</td>
<td>Lambda symmetric</td>
<td>MEASURES</td>
</tr>
<tr>
<td>LAMRC</td>
<td>Lambda asymmetric ($R \mid C$)</td>
<td>MEASURES</td>
</tr>
<tr>
<td>LGOR</td>
<td>Logit common odds ratio</td>
<td>CMH, CMH1, or CMH2</td>
</tr>
<tr>
<td>LGRRC1</td>
<td>Logit common relative risk, column 1</td>
<td>CMH, CMH1, or CMH2</td>
</tr>
<tr>
<td>LGRRC2</td>
<td>Logit common relative risk, column 2</td>
<td>CMH, CMH1, or CMH2</td>
</tr>
<tr>
<td>LRCHI</td>
<td>Likelihood ratio chi-square</td>
<td>CHISQ</td>
</tr>
<tr>
<td>MCNEM</td>
<td>McNemar’s test ($2 \times 2$ tables)</td>
<td>AGREE</td>
</tr>
<tr>
<td>MEASURES</td>
<td>Gamma, Kendall’s $tau$-b, Stuart’s $tau$-c, Somers’ $D(C \mid R)$ and $D(R \mid C)$, Pearson and Spearman correlations, lambda asymmetric ($C \mid R$) and ($R \mid C$), lambda symmetric, uncertainty coefficients ($C \mid R$) and ($R \mid C$), symmetric uncertainty coefficient; odds ratio and relative risks ($2 \times 2$ tables)</td>
<td>MEASURES</td>
</tr>
<tr>
<td>MHCHI</td>
<td>Mantel-Haenszel chi-square</td>
<td>CHISQ</td>
</tr>
<tr>
<td>MHOOR</td>
<td>COMOR</td>
<td>Mantel-Haenszel common odds ratio</td>
</tr>
<tr>
<td>MHRRC1</td>
<td>Mantel-Haenszel common relative risk, column 1</td>
<td>CMH, CMH1, or CMH2</td>
</tr>
<tr>
<td>MHRRC2</td>
<td>Mantel-Haenszel common relative risk, column 2</td>
<td>CMH, CMH1, or CMH2</td>
</tr>
<tr>
<td>N</td>
<td>Number of nonmissing observations</td>
<td></td>
</tr>
<tr>
<td>NMISS</td>
<td>Number of missing observations</td>
<td></td>
</tr>
<tr>
<td>OR</td>
<td>ODDSRATIO</td>
<td>Odds ratio ($2 \times 2$ tables)</td>
</tr>
<tr>
<td>PCHI</td>
<td>Chi-square goodness-of-fit test (one-way tables), Pearson chi-square (two-way tables)</td>
<td>CHISQ</td>
</tr>
<tr>
<td>PCORR</td>
<td>Pearson correlation coefficient</td>
<td>MEASURES</td>
</tr>
<tr>
<td>PHI</td>
<td>Phi coefficient</td>
<td>CHISQ</td>
</tr>
<tr>
<td>PLCORR</td>
<td>Polychoric correlation coefficient</td>
<td>PLCORR</td>
</tr>
<tr>
<td>RDIF1</td>
<td>Column 1 risk difference (row 1 – row 2)</td>
<td>RISKDIFF</td>
</tr>
<tr>
<td>RDIF2</td>
<td>Column 2 risk difference (row 1 – row 2)</td>
<td>RISKDIFF</td>
</tr>
<tr>
<td>RELRISK</td>
<td>Odds ratio and relative risks ($2 \times 2$ tables)</td>
<td>MEASURES or RELRISK</td>
</tr>
</tbody>
</table>

$^1$CHISQ computes Fisher’s exact test for $2 \times 2$ tables. Use the FISHER option to compute Fisher’s exact test for general $r \times c$ tables.
You can specify the following output-options in the OUTPUT statement.

**AGREE**

includes the following tests and measures of agreement in the output data set: McNemar’s test (for 2 × 2 tables), Bowker’s symmetry test, the simple kappa coefficient, and the weighted kappa coefficient. For multiway tables, the AGREE option also includes the following statistics in the output data set: overall simple and weighted kappa coefficients, tests for equal simple and weighted kappa coefficients, and Cochran’s $Q$ test.

The AGREE option in the TABLES statement requests computation of tests and measures of agreement. For more information, see the section “Tests and Measures of Agreement” on page 254.

AGREE statistics are computed only for square tables, where the number of rows equals the number of columns. PROC FREQTAB provides Bowker’s symmetry test and weighted kappa coefficients only for tables larger than 2 × 2. (For 2 × 2 tables, Bowker’s test is identical to McNemar’s test, and the weighted kappa coefficient equals the simple kappa coefficient.) Cochran’s $Q$ is available for multiway 2 × 2 tables.

**AJCHI**

includes the continuity-adjusted chi-square in the output data set. The continuity-adjusted chi-square is available for 2 × 2 tables and is provided by the CHISQ option in the TABLES statement. For more information, see the section “Continuity-Adjusted Chi-Square Test” on page 205.
ALL
includes all statistics that are requested by the CHISQ, MEASURES, and CMH output-options in the output data set. ALL also includes the number of nonmissing observations, which you can request individually by specifying the N output-option.

BDCHI
includes the Breslow-Day test in the output data set. The Breslow-Day test for homogeneity of odds ratios is computed for multiway 2 × 2 tables and is provided by the CMH, CMH1, and CMH2 options in the TABLES statement. For more information, see the section “Breslow-Day Test for Homogeneity of the Odds Ratios” on page 267.

BINOMIAL

BIN
includes the binomial proportion estimate, confidence limits, and tests in the output data set. The BINOMIAL option in the TABLES statement requests computation of binomial statistics, which are available for one-way tables. For more information, see the section “Binomial Proportion” on page 217.

CHISQ
includes the following chi-square tests and measures in the output data set for two-way tables: Pearson chi-square, likelihood ratio chi-square, Mantel-Haenszel chi-square, phi coefficient, contingency coefficient, and Cramér’s V. For 2 × 2 tables, CHISQ also includes Fisher’s exact test and the continuity-adjusted chi-square in the output data set. For more information, see the section “Chi-Square Tests and Statistics” on page 202. For one-way tables, CHISQ includes the chi-square goodness-of-fit test in the output data set. For more information, see the section “Chi-Square Test for One-Way Tables” on page 203. The CHISQ option in the TABLES statement requests computation of these statistics.

If you specify the CHISQ(WARN=OUTPUT) option in the TABLES statement, the CHISQ option also includes the variable WARN_PCHI in the output data set. This variable indicates the validity warning for the asymptotic Pearson chi-square test.

CMH
includes the following Cochran-Mantel-Haenszel statistics in the output data set: correlation, row mean scores (ANOVA), and general association. For 2 × 2 tables, the CMH option also includes the Mantel-Haenszel and logit estimates of the common odds ratio and relative risks. For multiway (stratified) 2 × 2 tables, the CMH option includes the Breslow-Day test for homogeneity of odds ratios. The CMH option in the TABLES statement requests computation of these statistics. For more information, see the section “Cochran-Mantel-Haenszel Statistics” on page 261.

If you specify the CMH(MANTELFLEISS) option in the TABLES statement, the CMH option includes the Mantel-Fleiss analysis in the output data set. The variables MF_CMH and WARN_CMH contain the Mantel-Fleiss criterion and the warning indicator, respectively.

CMH1
includes the CMH statistics in the output data set, with the exception of the row mean scores (ANOVA) statistic and the general association statistic. The CMH1 option in the TABLES statement requests computation of these statistics. For more information, see the section “Cochran-Mantel-Haenszel Statistics” on page 261.
CMH2
includes the CMH statistics in the output data set, with the exception of the general association statistic. The CMH2 option in the TABLES statement requests computation of these statistics. For more information, see the section “Cochran-Mantel-Haenszel Statistics” on page 261.

CMHCOR
includes the Cochran-Mantel-Haenszel correlation statistic in the output data set. The CMH option in the TABLES statement requests computation of this statistic. For more information, see the section “Correlation Statistic” on page 263.

CMHGA
includes the Cochran-Mantel-Haenszel general association statistic in the output data set. The CMH option in the TABLES statement requests computation of this statistic. For more information, see the section “General Association Statistic” on page 264.

CMHRMS
includes the Cochran-Mantel-Haenszel row mean scores (ANOVA) statistic in the output data set. The CMH option in the TABLES statement requests computation of this statistic. For more information, see the section “ANOVA (Row Mean Scores) Statistic” on page 263.

COCHQ
includes Cochran’s $Q$ test in the output data set. The AGREE option in the TABLES statement requests computation of this test, which is available for multiway $2 \times 2$ tables. For more information, see the section “Cochran’s $Q$ Test” on page 261.

CONTGY
includes the contingency coefficient in the output data set. The CHISQ option in the TABLES statement requests computation of the contingency coefficient. For more information, see the section “Contingency Coefficient” on page 207.

CRAMV
includes Cramér’s $V$ in the output data set. The CHISQ option in the TABLES statement requests computation of Cramér’s $V$. For more information, see the section “Cramér’s $V$” on page 207.

EQKAP
includes the test for equal simple kappa coefficients in the output data set. The AGREE option in the TABLES statement requests computation of this test, which is available for multiway, square ($h \times r \times r$) tables. For more information, see the section “Tests for Equal Kappa Coefficients” on page 260.

EQOR
includes Zelen’s exact test for equal odds ratios in the output data set. The EQOR option in the EXACT statement requests computation of this test, which is available for multiway $2 \times 2$ tables. For more information, see the section “Zelen’s Exact Test for Equal Odds Ratios” on page 268.

EQWKP
includes the test for equal weighted kappa coefficients in the output data set. The AGREE option in the TABLES statement requests computation of this test. The test for equal weighted kappas is available for multiway, square ($h \times r \times r$) tables where $r > 2$. For more information, see the section “Tests for Equal Kappa Coefficients” on page 260.
FISHER
includes Fisher’s exact test in the output data set. For $2 \times 2$ tables, the CHISQ option in the TABLES statement provides Fisher’s exact test. For tables larger than $2 \times 2$, the FISHER option in the EXACT statement provides Fisher’s exact test. For more information, see the section “Fisher’s Exact Test” on page 206.

GAMMA
includes the gamma statistic in the output data set. The MEASURES option in the TABLES statement requests computation of the gamma statistic. For more information, see the section “Gamma” on page 209.

GS
includes the Gail-Simon test for qualitative interaction in the output data set. The CMH(GAILSIMON) option in the TABLES statement requests computation of this test. For more information, see the section “Gail-Simon Test for Qualitative Interactions” on page 271.

JT
includes the Jonckheere-Terpstra test in the output data set. The JT option in the TABLES statement requests the Jonckheere-Terpstra test. For more information, see the section “Jonckheere-Terpstra Test” on page 252.

KAPPA
includes the simple kappa coefficient in the output data set. The AGREE option in the TABLES statement requests computation of kappa, which is available for square tables (where the number of rows equals the number of columns). For multiway square tables, the KAPPA option also includes the overall kappa coefficient in the output data set. For more information, see the sections “Simple Kappa Coefficient” on page 255 and “Overall Kappa Coefficient” on page 260.

KENTB
includes Kendall’s tau-$b$ in the output data set. The MEASURES option in the TABLES statement requests computation of Kendall’s tau-$b$. For more information, see the section “Kendall’s Tau-b” on page 210.

LAMCR
includes the asymmetric lambda $\lambda(C|R)$ in the output data set. The MEASURES option in the TABLES statement requests computation of lambda. For more information, see the section “Lambda (Asymmetric)” on page 215.

LAMDAS
includes the symmetric lambda in the output data set. The MEASURES option in the TABLES statement requests computation of lambda. For more information, see the section “Lambda (Symmetric)” on page 216.

LAMRC
includes the asymmetric lambda $\lambda(R|C)$ in the output data set. The MEASURES option in the TABLES statement requests computation of lambda. For more information, see the section “Lambda (Asymmetric)” on page 215.
LGOR
includes the logit estimate of the common odds ratio in the output data set. The CMH option in the TABLES statement requests computation of this statistic, which is available for $2 \times 2$ tables. For more information, see the section “Adjusted Odds Ratio and Relative Risk Estimates” on page 265.

LGRRC1
includes the logit estimate of the common relative risk (column 1) in the output data set. The CMH option in the TABLES statement requests computation of this statistic, which is available for $2 \times 2$ tables. For more information, see the section “Adjusted Odds Ratio and Relative Risk Estimates” on page 265.

LGRRC2
includes the logit estimate of the common relative risk (column 2) in the output data set. The CMH option in the TABLES statement requests computation of this statistic, which is available for $2 \times 2$ tables. For more information, see the section “Adjusted Odds Ratio and Relative Risk Estimates” on page 265.

LRCHI
includes the likelihood ratio chi-square in the output data set. The CHISQ option in the TABLES statement requests computation of the likelihood ratio chi-square. For more information, see the section “Likelihood Ratio Chi-Square Test” on page 205.

MCNEM
includes McNemar’s test (for $2 \times 2$ tables) in the output data set. The AGREE option in the TABLES statement requests computation of McNemar’s test. For more information, see the section “McNemar’s Test” on page 254.

MEASURES
includes the following measures of association in the output data set: gamma, Kendall’s tau-$b$, Stuart’s tau-$c$, Somers’ $D(C|R)$, Somers’ $D(R|C)$, Pearson and Spearman correlation coefficients, lambda (symmetric and asymmetric), and uncertainty coefficients (symmetric and asymmetric). For $2 \times 2$ tables, the MEASURES option also includes the odds ratio, column 1 relative risk, and column 2 relative risk. The MEASURES option in the TABLES statement requests computation of these statistics. For more information, see the section “Measures of Association” on page 208.

MHCHI
includes the Mantel-Haenszel chi-square in the output data set. The CHISQ option in the TABLES statement requests computation of the Mantel-Haenszel chi-square. For more information, see the section “Mantel-Haenszel Chi-Square Test” on page 205.

MHOR
includes the Mantel-Haenszel estimate of the common odds ratio in the output data set. The CMH option in the TABLES statement requests computation of this statistic, which is available for $2 \times 2$ tables. For more information, see the section “Adjusted Odds Ratio and Relative Risk Estimates” on page 265.
MHRRC1
includes the Mantel-Haenszel estimate of the common relative risk (column 1) in the output data set. The CMH option in the TABLES statement requests computation of this statistic, which is available for $2 \times 2$ tables. For more information, see the section “Adjusted Odds Ratio and Relative Risk Estimates” on page 265.

MHRRC2
includes the Mantel-Haenszel estimate of the common relative risk (column 2) in the output data set. The CMH option in the TABLES statement requests computation of this statistic, which is available for $2 \times 2$ tables. For more information, see the section “Adjusted Odds Ratio and Relative Risk Estimates” on page 265.

N
includes the number of nonmissing observations in the output data set.

NMISS
includes the number of missing observations in the output data set. For more information, see the section “Missing Values” on page 198.

OR
ODDSRATIO
RROR
includes the odds ratio (for $2 \times 2$ tables) in the output data set. The MEASURES, OR, and RELRISK options in the TABLES statement request this statistic. For more information, see the section “Odds Ratio” on page 241.

PCHI
includes the Pearson chi-square in the output data set for two-way tables. For more information, see the section “Pearson Chi-Square Test for Two-Way Tables” on page 203. For one-way tables, the PCHI option includes the chi-square goodness-of-fit test in the output data set. For more information, see the section “Chi-Square Test for One-Way Tables” on page 203. The CHISQ option in the TABLES statement requests computation of these statistics.

If you specify the CHISQ(WARN=OUTPUT) option in the TABLES statement, the PCHI option also includes the variable WARN_PCHI in the output data set. This variable indicates the validity warning for the asymptotic Pearson chi-square test.

PCORR
includes the Pearson correlation coefficient in the output data set. The MEASURES option in the TABLES statement requests computation of the Pearson correlation. For more information, see the section “Pearson Correlation Coefficient” on page 211.

PHI
includes the phi coefficient in the output data set. The CHISQ option in the TABLES statement requests computation of the phi coefficient. For more information, see the section “Phi Coefficient” on page 207.

PLCORR
includes the polychoric correlation coefficient in the output data set. For $2 \times 2$ tables, this statistic is known as the tetrachoric correlation coefficient. The PLCORR option in the TABLES statement requests computation of the polychoric correlation. For more information, see the section “Polychoric Correlation” on page 214.
includes the column 1 risk difference (row 1 – row 2) in the output data set. The RISKDIFF option in the TABLES statement requests computation of risks and risk differences, which are available for \(2 \times 2\) tables. For more information, see the section “Risks and Risk Differences” on page 226.

includes the column 2 risk difference (row 1 – row 2) in the output data set. The RISKDIFF option in the TABLES statement requests computation of risks and risk differences, which are available for \(2 \times 2\) tables. For more information, see the section “Risks and Risk Differences” on page 226.

includes the column 1 and column 2 relative risks (for \(2 \times 2\) tables) in the output data set. The MEASURES and RELRISK options in the TABLES statement request these statistics. For more information, see the section “Relative Risks” on page 244.

includes risks (binomial proportions) and risk differences for \(2 \times 2\) tables in the output data set. These statistics include the row 1 risk, row 2 risk, total (overall) risk, and risk difference (row 1 – row 2) for column 1 and column 2. The RISKDIFF option in the TABLES statement requests computation of these statistics. For more information, see the section “Risks and Risk Differences” on page 226.

includes column 1 risks (binomial proportions) and risk differences for \(2 \times 2\) tables in the output data set. These statistics include the row 1 risk, row 2 risk, total (overall) risk, and risk difference (row 1 – row 2). The RISKDIFF option in the TABLES statement requests computation of these statistics. For more information, see the section “Risks and Risk Differences” on page 226.

includes column 2 risks (binomial proportions) and risk differences for \(2 \times 2\) tables in the output data set. These statistics include the row 1 risk, row 2 risk, total (overall) risk, and risk difference (row 1 – row 2). The RISKDIFF option in the TABLES statement requests computation of these statistics. For more information, see the section “Risks and Risk Differences” on page 226.

includes the column 1 relative risk in the output data set. The MEASURES and RELRISK options in the TABLES statement request relative risks, which are available for \(2 \times 2\) tables. For more information, see the section “Odds Ratio and Relative Risks for \(2 \times 2\) Tables” on page 241.

includes the column 2 relative risk in the output data set. The MEASURES and RELRISK options in the TABLES statement request relative risks, which are available for \(2 \times 2\) tables. For more information, see the section “Odds Ratio and Relative Risks for \(2 \times 2\) Tables” on page 241.

includes the overall column 1 risk in the output data set. The RISKDIFF option in the TABLES statement requests computation of risks and risk differences, which are available for \(2 \times 2\) tables. For more information, see the section “Risks and Risk Differences” on page 226.
Chapter 5: The FREQTAB Procedure

**RSK11**
includes the column 1 risk for row 1 in the output data set. The **RISKDIFF** option in the **TABLES** statement requests computation of risks and risk differences, which are available for $2 \times 2$ tables. For more information, see the section “Risks and Risk Differences” on page 226.

**RSK12**
includes the column 2 risk for row 1 in the output data set. The **RISKDIFF** option in the **TABLES** statement requests computation of risks and risk differences, which are available for $2 \times 2$ tables. For more information, see the section “Risks and Risk Differences” on page 226.

**RSK2**
includes the overall column 2 risk in the output data set. The **RISKDIFF** option in the **TABLES** statement requests computation of risks and risk differences. For more information, see the section “Risks and Risk Differences” on page 226.

**RSK21**
includes the column 1 risk for row 2 in the output data set. The **RISKDIFF** option in the **TABLES** statement requests computation of risks and risk differences, which are available for $2 \times 2$ tables. For more information, see the section “Risks and Risk Differences” on page 226.

**RSK22**
includes the column 2 risk for row 2 in the output data set. The **RISKDIFF** option in the **TABLES** statement requests computation of risks and risk differences, which are available for $2 \times 2$ tables. For more information, see the section “Risks and Risk Differences” on page 226.

**SCORR**
includes the Spearman correlation coefficient in the output data set. The **MEASURES** option in the **TABLES** statement requests computation of the Spearman correlation. For more information, see the section “Spearman Rank Correlation Coefficient” on page 212.

**SMDCR**
includes Somers’ $D(C|R)$ in the output data set. The **MEASURES** option in the **TABLES** statement requests computation of Somers’ $D$. For more information, see the section “Somers’ $D$” on page 211.

**SMDRC**
includes Somers’ $D(R|C)$ in the output data set. The **MEASURES** option in the **TABLES** statement requests computation of Somers’ $D$. For more information, see the section “Somers’ $D$” on page 211.

**STUTC**
includes Stuart’s tau-$c$ in the output data set. The **MEASURES** option in the **TABLES** statement requests computation of tau-$c$. For more information, see the section “Stuart’s Tau-$c$” on page 210.
TREND
includes the Cochran-Armitage test for trend in the output data set. The TREND option in the TABLES statement requests computation of the trend test. This test is available for tables of dimension $2 \times C$ or $R \times 2$. For more information, see the section “Cochran-Armitage Test for Trend” on page 251.

TSYMM
BOWKER
includes Bowker’s symmetry test in the output data set. The AGREE option in the TABLES statement requests computation of Bowker’s test. For more information, see the section “Bowker’s Symmetry Test” on page 255.

U
includes the uncertainty coefficient (symmetric) in the output data set. The MEASURES option in the TABLES statement requests computation of the uncertainty coefficient. For more information, see the section “Uncertainty Coefficient (Symmetric)” on page 217.

UCR
includes the asymmetric uncertainty coefficient $U(C|R)$ in the output data set. The MEASURES option in the TABLES statement requests computation of the uncertainty coefficient. For more information, see the section “Uncertainty Coefficients (Asymmetric)” on page 216.

URC
includes the asymmetric uncertainty coefficient $U(R|C)$ in the output data set. The MEASURES option in the TABLES statement requests computation of the uncertainty coefficient. For more information, see the section “Uncertainty Coefficients (Asymmetric)” on page 216.

WTKAPPA
WTKAP
includes the weighted kappa coefficient in the output data set. The AGREE option in the TABLES statement requests computation of weighted kappa, which is available for square tables larger than $2 \times 2$. For multiway tables, the WTKAPPA option also includes the overall weighted kappa coefficient in the output data set. For more information, see the sections “Weighted Kappa Coefficient” on page 257 and “Overall Kappa Coefficient” on page 260.

**TABLES Statement**

```plaintext
TABLES requests </options> ;
```

The TABLES statement requests one-way to $n$-way frequency and crosstabulation tables and statistics for those tables.

If you omit the TABLES statement, PROC FREQTAB generates one-way frequency tables for all data set variables that are not listed in the other statements.

The following argument is required in the TABLES statement.

`requests`

specify the frequency and crosstabulation tables to produce. A request is composed of one variable name or several variable names separated by asterisks. To request a one-way frequency table, use a
single variable. To request a two-way crosstabulation table, use an asterisk between two variables. To request a multiway table (an \( n \)-way table, where \( n > 2 \)), separate the variables with asterisks. The unique values of these variables form the rows, columns, and strata of the table. You can include up to 50 variables in a single multiway table request.

For two-way to multiway tables, the values of the last variable form the crosstabulation table columns, and the values of the next-to-last variable form the rows. Each level (or combination of levels) of the other variables forms one stratum. PROC FREQTAB produces a separate crosstabulation table for each stratum. For example, a specification of \( A*B*C*D \) in a TABLES statement produces \( k \) tables, where \( k \) is the number of different combinations of values for \( A \) and \( B \). Each table lists the values for \( C \) down the side and the values for \( D \) across the top.

You can use multiple TABLES statements in the PROC FREQTAB step. PROC FREQTAB builds all the table requests in one pass of the data, so that there is essentially no loss of efficiency. You can also specify any number of table requests in a single TABLES statement. To specify multiple table requests quickly, use a grouping syntax by placing parentheses around several variables and joining other variables or variable combinations. For example, the statements shown in Table 5.7 illustrate grouping syntax.

<table>
<thead>
<tr>
<th>TABLES Request</th>
<th>Equivalent to</th>
</tr>
</thead>
<tbody>
<tr>
<td>( A*(B\ C) )</td>
<td>( A<em>B\ A</em>C )</td>
</tr>
<tr>
<td>( (A\ B)*(C\ D) )</td>
<td>( A<em>C\ B</em>C\ A<em>D\ B</em>D )</td>
</tr>
<tr>
<td>( (A\ B* C)*D )</td>
<td>( A<em>D\ B</em>D\ C*D )</td>
</tr>
<tr>
<td>( A -- C )</td>
<td>( A\ B\ C )</td>
</tr>
<tr>
<td>( (A -- C)*D )</td>
<td>( A<em>D\ B</em>D\ C*D )</td>
</tr>
</tbody>
</table>

The TABLES statement variables are one or more variables from the DATA= input data table. These variables can be either character or numeric, but the procedure treats them as categorical variables. PROC FREQTAB uses the formatted values of the TABLES variable to determine the categorical variable levels. For more information, see the discussion of the FORMAT procedure in the Base SAS Procedures Guide and the discussion of SAS formats in SAS Formats and Informats: Reference.

By default, the frequency or crosstabulation table lists the values of both character and numeric variables in ascending order based on internal (unformatted) variable values. You can change the order of the values in the table by specifying the ORDER= option in the PROC FREQTAB statement. To list the values in ascending order by formatted value, use ORDER=FORMATTED.

**Without Options**

If you request a one-way frequency table without specifying any options, PROC FREQTAB provides the following for each table level or cell: frequency, cumulative frequency, percentage of the total frequency, and cumulative percentage. If you request a two-way or multiway \( (n\)-way) crosstabulation table without specifying any options, PROC FREQTAB provides the following for each table cell: frequency, percentage of the total frequency, percentage of the row frequency, and percentages of the column frequency. PROC FREQTAB also displays the frequency of missing values after the table. For more information, see the section “Displayed Output” on page 278.
Options

Table 5.8 lists the options available in the TABLES statement. Descriptions of the options follow in alphabetical order.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Control Statistical Analysis</strong></td>
<td></td>
</tr>
<tr>
<td>AGREE</td>
<td>Requests tests and measures of classification agreement</td>
</tr>
<tr>
<td>ALL</td>
<td>Requests tests and measures of association produced by the CHISQ, MEASURES, and CMH options</td>
</tr>
<tr>
<td>ALPHA=</td>
<td>Sets confidence level for confidence limits</td>
</tr>
<tr>
<td>BINOMIAL</td>
<td>Requests binomial proportions, confidence limits, and tests for one-way tables</td>
</tr>
<tr>
<td>CHISQ</td>
<td>Requests chi-square tests and measures based on chi-square</td>
</tr>
<tr>
<td>CL</td>
<td>Requests confidence limits for MEASURES statistics</td>
</tr>
<tr>
<td>CMH</td>
<td>Requests all Cochran-Mantel-Haenszel statistics</td>
</tr>
<tr>
<td>CMH1</td>
<td>Requests CMH correlation statistic, adjusted odds ratios, and adjusted relative risks</td>
</tr>
<tr>
<td>CMH2</td>
<td>Requests CMH correlation and row mean scores (ANOVA) statistics, adjusted odds ratios, and adjusted relative risks</td>
</tr>
<tr>
<td>COMMONRISKDIFF</td>
<td>Requests common risk difference for $h \times 2 \times 2$ tables</td>
</tr>
<tr>
<td>FISHER</td>
<td>Requests Fisher's exact test for tables larger than $2 \times 2$</td>
</tr>
<tr>
<td>GAILSIMON</td>
<td>Requests Gail-Simon test for qualitative interactions</td>
</tr>
<tr>
<td>JT</td>
<td>Requests Jonckheere-Terpstra test</td>
</tr>
<tr>
<td>MEASURES</td>
<td>Requests measures of association</td>
</tr>
<tr>
<td>OR</td>
<td>Requests the odds ratio for $2 \times 2$ tables</td>
</tr>
<tr>
<td>PLCORR</td>
<td>Requests polychoric correlation</td>
</tr>
<tr>
<td>RELRISK</td>
<td>Requests relative risks for $2 \times 2$ tables</td>
</tr>
<tr>
<td>RISKDIFF</td>
<td>Requests risks and risk differences for $2 \times 2$ tables</td>
</tr>
<tr>
<td>SCORES=</td>
<td>Specifies type of row and column scores</td>
</tr>
<tr>
<td>SENSPEC</td>
<td>Requests sensitivity and specificity for $2 \times 2$ tables</td>
</tr>
<tr>
<td>TREND</td>
<td>Requests Cochran-Armitage test for trend</td>
</tr>
<tr>
<td><strong>Control Additional Table Information</strong></td>
<td></td>
</tr>
<tr>
<td>CELLCHI2</td>
<td>Displays cell contributions to the Pearson chi-square statistic</td>
</tr>
<tr>
<td>CUMCOL</td>
<td>Displays cumulative column percentages</td>
</tr>
<tr>
<td>DEVIATION</td>
<td>Displays deviations of cell frequencies from expected values</td>
</tr>
<tr>
<td>EXPECTED</td>
<td>Displays expected cell frequencies</td>
</tr>
<tr>
<td>PEARSONRES</td>
<td>Displays Pearson residuals in the CROSSLIST table</td>
</tr>
<tr>
<td>PRINTKWTS</td>
<td>Displays kappa coefficient weights</td>
</tr>
<tr>
<td>SCOROUT</td>
<td>Displays row and column scores</td>
</tr>
<tr>
<td>STDRES</td>
<td>Displays standardized residuals in the CROSSLIST table</td>
</tr>
<tr>
<td>TOTPCT</td>
<td>Displays percentages of total frequency for $n$-way tables ($n &gt; 2$)</td>
</tr>
<tr>
<td><strong>Control Displayed Output</strong></td>
<td></td>
</tr>
<tr>
<td>CONTENTS=</td>
<td>Specifies contents label for crosstabulation tables</td>
</tr>
<tr>
<td>CROSSLIST</td>
<td>Displays crosstabulation tables in ODS column format</td>
</tr>
</tbody>
</table>
Table 5.8  continued

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FORMAT=</td>
<td>Formats frequencies in crosstabulation tables</td>
</tr>
<tr>
<td>LIST</td>
<td>Displays two-way to n-way tables in list format</td>
</tr>
<tr>
<td>MAXLEVELS=</td>
<td>Specifies maximum number of levels to display in one-way tables</td>
</tr>
<tr>
<td>NOCOL</td>
<td>Suppresses display of column percentages</td>
</tr>
<tr>
<td>NOCUM</td>
<td>Suppresses display of cumulative frequencies and percentages</td>
</tr>
<tr>
<td>NOFREQ</td>
<td>Suppresses display of frequencies</td>
</tr>
<tr>
<td>NOPERCENT</td>
<td>Suppresses display of percentages</td>
</tr>
<tr>
<td>NOPRINT</td>
<td>Suppresses display of crosstabulation tables but displays statistics</td>
</tr>
<tr>
<td>NOROW</td>
<td>Suppresses display of row percentages</td>
</tr>
<tr>
<td>NOSPARSE</td>
<td>Suppresses zero-frequency levels in the CROSSLIST table, LIST table, and OUT= data set</td>
</tr>
<tr>
<td>NOWARN</td>
<td>Suppresses log warning message for the chi-square test</td>
</tr>
</tbody>
</table>

**Produce Statistical Graphics**

PLOTS= Requests plots from ODS Graphics

**Create an Output Data Set**

OUT= Names an output data set to contain frequency counts

OUTCUM Includes cumulative frequencies and percentages in the output data set

OUTEXPECT Includes expected frequencies in the output data set

OUTPCT Includes row, column, and two-way table percentages in the output data set

You can specify the following **options** in a TABLES statement.

**AGREE < (agree-options) >**

requests tests and measures of classification agreement for square tables. This option provides the simple and weighted kappa coefficients along with their standard errors and confidence limits. For multiway tables, this option also produces the overall simple and weighted kappa coefficients (along with their standard errors and confidence limits) and tests for equal kappas among strata. For $2 \times 2$ tables, this option provides McNemar’s test; for square tables that have more than two response categories (levels), this option provides Bowker’s symmetry test. For multiway tables that have two response categories, this option also produces Cochran’s $Q$ test. For more information, see the section “Tests and Measures of Agreement” on page 254.

Measures of agreement can be computed only for square tables, where the number of rows equals the number of columns. If your table is not square because some observations have weights of 0, you can specify the ZEROS option in the WEIGHT statement to include these observations in the analysis. For more information, see the section “Tables with Zero-Weight Rows or Columns” on page 261.

For $2 \times 2$ tables, the weighted kappa coefficient is equivalent to the simple kappa coefficient, and PROC FREQTAB displays only analyses for the simple kappa coefficient.

You can specify the confidence level in the **ALPHA=** option. By default, ALPHA=0.05, which produces 95% confidence limits.
You can specify the **EXACT** statement to request McNemar’s exact test (for $2 \times 2$ tables), an exact symmetry test, and exact tests for the simple and weighted kappa coefficients. For more information, see the section “Exact Statistics” on page 271.

You can specify the following *agree-options*:

**AC1**
requests the AC1 agreement coefficient. For more information, see the section “AC1 Agreement Coefficient” on page 260.

**DFSYM=** \( df \) | **ADJUST**
controls the degrees of freedom for Bowker’s symmetry test. You can specify the value of the degrees of freedom \( df \), or you can specify **DFSYM=ADJUST** to adjust the degrees of freedom for empty table cells. The value of \( df \) must be a positive number. By default, \( df = R(R - 1)/2 \), where \( R \) is the dimension of the two-way table.

When you specify **DFSYM=ADJUST**, the degrees of freedom are reduced by the number of off-diagonal table-cell pairs that have a total frequency of 0. By default, the degrees of freedom count all off-diagonal table-cell pairs. For more information, see the section “Bowker’s Symmetry Test” on page 255.

**KAPPADetails**
**DETAILS**
displays the “Kappa Details” table, which includes the following statistics for the simple kappa coefficient: observed agreement, chance-expected agreement, maximum kappa, and the \( B_n \) measure. If the two-way table is $2 \times 2$, the “Kappa Details” table also includes the prevalence index and the bias index. For more information, see the section “Simple Kappa Coefficient” on page 255.

If the two-way table is larger than $2 \times 2$, this option also displays the “Weighted Kappa Details” table, which includes the observed agreement and chance-expected agreement components of the weighted kappa coefficient. For more information, see the section “Weighted Kappa Coefficient” on page 257.

**MNULLRATIO=** *value*
specifies the null *value* of the ratio of discordant proportions for McNemar’s test. By default, **MNULLRATIO=1**. For more information, see the section “McNemar’s Test” on page 254.

**NULLKAPPA=** *value*
requests the simple kappa coefficient test and specifies the null *value* for the test. The null value must be between –1 and 1. By default, **NULLKAPPA=0**. For more information, see the section “Simple Kappa Coefficient” on page 255.

This option is not available when you specify the **KAPPA** option in the **EXACT** statement, which requests an exact test for the kappa coefficient.

**NULLWTKAPPA=** *value*
requests the weighted kappa coefficient test and specifies the null *value* for the test. The null value must be between –1 and 1. By default, **NULLWTKAPPA=0**. For more information, see the section “Weighted Kappa Coefficient” on page 257.

This option is not available when you specify the **WTKAPPA** option in the **EXACT** statement, which requests an exact test for the weighted kappa coefficient.
PABAK
requests the prevalence-adjusted bias-adjusted kappa coefficient. For more information, see the section “Prevalence-Adjusted Bias-Adjusted Kappa” on page 259.

PRINTKWTS
displays the agreement weights that PROC FREQTAB uses to compute the weighted kappa coefficient. Agreement weights reflect the relative agreement between pairs of variable levels. By default, PROC FREQTAB uses Cicchetti-Allison agreement weights. If you specify the WT=FC option, the procedure uses Fleiss-Cohen agreement weights. For more information, see the section “Weighted Kappa Coefficient” on page 257.

TABLES=RESTORE
displays agreement tables (which are produced by the AGREE option) in factoid (label-value) format, which is the format of these tables in releases before SAS/STAT 14.2. Beginning in SAS/STAT 14.3, PROC FREQTAB displays all agreement tables in tabular format (instead of factoid format) by default.

In SAS/STAT 14.2, PROC FREQTAB displays agreement tables in tabular format (instead of factoid format) by default when you specify any of the following agree-options: AC1, KAPPADETAILS, NULLKAPPA=, NULLWTKAPPA=, PABAK, or WTKAPPADETAILS.

WT=FC
specifies Fleiss-Cohen agreement weights in the computation of the weighted kappa coefficient. Agreement weights reflect the relative agreement between pairs of variable levels. By default, PROC FREQTAB uses Cicchetti-Allison agreement weights to compute the weighted kappa coefficient. For more information, see the section “Weighted Kappa Coefficient” on page 257.

WTKAPPADETAILS
displays the “Weighted Kappa Details” table, which includes the observed agreement and chance-expected agreement components of the weighted kappa coefficient. This information is available for two-way tables that are larger than $2 \times 2$. For more information, see the section “Weighted Kappa Coefficient” on page 257.

ALL
requests all tests and measures that are produced by the CHISQ, MEASURES, and CMH options. You can control the number of CMH statistics to compute by specifying the CMH1 or CMH2 option.

ALPHA=\alpha
specifies the level of confidence limits. The value of $\alpha$ must be between 0 and 1; a confidence level of $\alpha$ produces $100(1 - \alpha)\%$ confidence limits. By default ALPHA=0.05, which produces 95% confidence limits.

This option applies to confidence limits that you request in the TABLES statement. The ALPHA= option in the EXACT statement applies to confidence limits for Monte Carlo estimates of exact $p$-values, which you request by specifying the MC option in the EXACT statement.
requests the binomial proportion for one-way tables. When you specify this option, by default PROC FREQTAB provides the asymptotic standard error, asymptotic Wald and exact (Clopper-Pearson) confidence limits, and the asymptotic equality test for the binomial proportion.

You can specify `binomial-options` in parentheses after the BINOMIAL option. The `LEVEL= binomial-option` identifies the variable level for which to compute the proportion. If you do not specify this option, PROC FREQTAB computes the proportion for the first level that appears in the one-way frequency table. The `P= binomial-option` specifies the null proportion for the binomial tests. If you do not specify this option, PROC FREQTAB uses 0.5 as the null proportion for the binomial tests.

You can also specify `binomial-options` to request additional tests and confidence limits for the binomial proportion. The `EQUIV`, `NONINF`, and `SUP binomial-options` request tests of equivalence, noninferiority, and superiority, respectively. The `CL= binomial-option` requests confidence limits for the binomial proportion.

You can specify the level for the binomial confidence limits in the `ALPHA=` option. By default, `ALPHA=0.05`, which produces 95% confidence limits. As part of the noninferiority, superiority, and equivalence analyses, PROC FREQTAB provides null-based equivalence limits that have a confidence coefficient of \(100(1 - 2\alpha)\%\) (Schuirmann 1999). In these analyses, the default of `ALPHA=0.05` produces 90% equivalence limits. For more information, see the sections “Noninferiority Test” on page 222 and “Equivalence Test” on page 224.

To request exact tests for the binomial proportion, you can specify the `BINOMIAL` option in the EXACT statement. PROC FREQTAB computes exact \(p\)-values for all binomial tests that you request, which can include noninferiority, superiority, and equivalence tests, in addition to the equality test that the BINOMIAL option produces by default.

For more information, see the section “Binomial Proportion” on page 217.

Table 5.9 summarizes the `binomial-options`.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CORRECT</td>
<td>Requests continuity correction</td>
</tr>
<tr>
<td>LEVEL=</td>
<td>Specifies the variable level</td>
</tr>
<tr>
<td>OUTLEVEL</td>
<td>Includes the level in the output data sets</td>
</tr>
<tr>
<td>P=</td>
<td>Specifies the null proportion</td>
</tr>
<tr>
<td><strong>Request Confidence Limits</strong></td>
<td></td>
</tr>
<tr>
<td>`CL=AGRESTICOULL</td>
<td>AC`</td>
</tr>
<tr>
<td><code>CL=BLAKER</code></td>
<td>Requests Blaker confidence limits</td>
</tr>
<tr>
<td>`CL=EXACT</td>
<td>CLOPPERPEARSON`</td>
</tr>
<tr>
<td><code>CL=JEFFREYS</code></td>
<td>Requests Jeffreys confidence limits</td>
</tr>
<tr>
<td>`CL=LIKELIHOODRATIO</td>
<td>LR`</td>
</tr>
<tr>
<td><code>CL=LOGIT</code></td>
<td>Requests logit confidence limits</td>
</tr>
<tr>
<td><code>CL=MIDP</code></td>
<td>Requests exact mid-(p) confidence limits</td>
</tr>
<tr>
<td><code>CL=WALD</code></td>
<td>Requests Wald confidence limits</td>
</tr>
<tr>
<td>`CL=WILSON</td>
<td>SCORE`</td>
</tr>
</tbody>
</table>
Table 5.9 continued

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Request Tests</td>
<td></td>
</tr>
<tr>
<td>EQUIV</td>
<td>EQUIVALENCE</td>
</tr>
<tr>
<td>MARGIN=</td>
<td>Specifies the test margin</td>
</tr>
<tr>
<td>NONINF</td>
<td>NONINFERIORITY</td>
</tr>
<tr>
<td>SUP</td>
<td>SUPERIORITY</td>
</tr>
<tr>
<td>VAR=NULL</td>
<td>SAMPLE</td>
</tr>
</tbody>
</table>

You can specify the following binomial-options:

**CL=type | (types)**

requests confidence limits for the binomial proportion. You can specify one or more types of confidence limits. When you specify only one type, you can omit the parentheses around the request. PROC FREQTAB displays the confidence limits in the “Binomial Confidence Limits” table.

The ALPHA= option determines the level of the confidence limits that the CL= binomial-option provides. By default, ALPHA=0.05, which produces 95% confidence limits for the binomial proportion.

You can specify the CL= binomial-option with or without requests for binomial tests. The confidence limits that CL= produces do not depend on the tests that you request and do not use the value of the test margin (which you can specify in the MARGIN= binomial-option).

If you do not specify the CL= binomial-option, the BINOMIAL option displays Wald and exact (Clopper-Pearson) confidence limits in the “Binomial Proportion” table.

You can specify the following types:

**AGRESTICOULL**

**AC**

requests Agresti-Coull confidence limits for the binomial proportion. For more information, see the section “Agresti-Coull Confidence Limits” on page 219.

**BLAKER**

requests Blaker confidence limits for the binomial proportion. For more information, see the section “Blaker Confidence Limits” on page 219.

**EXACT**

**CLOPPERPEARSON**

requests exact (Clopper-Pearson) confidence limits for the binomial proportion. For more information, see the section “Exact (Clopper-Pearson) Confidence Limits” on page 218.

If you do not specify the CL= binomial-option, PROC FREQTAB displays Wald and exact (Clopper-Pearson) confidence limits in the “Binomial Proportion” table. To request exact tests for the binomial proportion, you can specify the BINOMIAL option in the EXACT statement.
JEFFREYS
requests Jeffreys confidence limits for the binomial proportion. For more information, see the section “Jeffreys Confidence Limits” on page 219.

LIKELIHOODRATIO
LR
requests likelihood ratio confidence limits for the binomial proportion. For more information, see the section “Likelihood Ratio Confidence Limits” on page 219.

LOGIT
requests logit confidence limits for the binomial proportion. For more information, see the section “Logit Confidence Limits” on page 220.

MIDP
requests exact mid-p confidence limits for the binomial proportion. For more information, see the section “Mid-p Confidence Limits” on page 220.

WALD < (CORRECT) >
requests Wald confidence limits for the binomial proportion. For more information, see the section “Wald Confidence Limits” on page 218.

If you specify CL=WALD(CORRECT), the Wald confidence limits include a continuity correction. If you specify the CORRECT binomial-option, both the Wald confidence limits and the Wald tests include continuity corrections.

If you do not specify the CL= binomial-option, PROC FREQTAB displays Wald and exact (Clopper-Pearson) confidence limits in the “Binomial Proportion” table.

WILSON < (CORRECT) >
SCORE < (CORRECT) >
requests Wilson confidence limits for the binomial proportion. These are also known as score confidence limits. For more information, see the section “Wilson (Score) Confidence Limits” on page 220.

If you specify CL=WILSON(CORRECT) or the CORRECT binomial-option, the Wilson confidence limits include a continuity correction.

CORRECT
includes a continuity correction in the Wald confidence limits, Wald tests, and Wilson confidence limits.

You can request continuity corrections individually for Wald or Wilson confidence limits by specifying the CL=WALD(CORRECT) or CL=WILSON(CORRECT) binomial-option, respectively.

EQUIV
EQUIVALENCE
requests a test of equivalence for the binomial proportion. For more information, see the section “Equivalence Test” on page 224. You can specify the equivalence test margins, the null proportion, and the variance type in the MARGIN=, P=, and VAR= binomial-options, respectively. To request an exact equivalence test, you can specify the BINOMIAL option in the EXACT statement.
LEVEL=level-number | 'level-value'
specifies the variable level for the binomial proportion. You can specify the level-number, which is the order in which the level appears in the one-way frequency table. Or you can specify the level-value, which is the formatted value of the variable level. The level-number must be a positive integer. You must enclose the level-value in single quotes.

By default, PROC FREQTAB computes the binomial proportion for the first variable level that appears in the one-way frequency table.

MARGIN=value | (lower, upper)
specifies the margin for the noninferiority, superiority, and equivalence tests, which you can request by specifying the NONINF, SUP, and EQUIV binomial-options, respectively. By default, MARGIN=0.2.

For noninferiority and superiority tests, specify a single value in the MARGIN= option. The MARGIN= value must be a positive number. You can specify value as a number between 0 and 1. Or you can specify value in percentage form as a number between 1 and 100, and PROC FREQTAB converts that number to a proportion. PROC FREQTAB treats the value 1 as 1%.

For noninferiority and superiority tests, the test limits must be between 0 and 1. The limits are determined by the null proportion value (which you can specify in the P= binomial-option) and by the margin value. The noninferiority limit is the null proportion minus the margin. By default, the null proportion is 0.5 and the margin is 0.2, which produces a noninferiority limit of 0.3. The superiority limit is the null proportion plus the margin, which is 0.7 by default.

For an equivalence test, you can specify a single MARGIN= value, or you can specify both lower and upper values. If you specify a single MARGIN= value, it must be a positive number, as described previously. If you specify a single MARGIN= value for an equivalence test, PROC FREQTAB uses –value as the lower margin and value as the upper margin for the test. If you specify both lower and upper values for an equivalence test, you can specify them in proportion form as numbers between –1 and 1. Or you can specify them in percentage form as numbers between –100 and 100, and PROC FREQTAB converts the numbers to proportions. The value of lower must be less than the value of upper.

The equivalence limits must be between 0 and 1. The equivalence limits are determined by the null proportion value (which you can specify in the P= binomial-option) and by the margin values. The lower equivalence limit is the null proportion plus the lower margin. By default, the null proportion is 0.5 and the lower margin is –0.2, which produces a lower equivalence limit of 0.3. The upper equivalence limit is the null proportion plus the upper margin, which is 0.7 by default.

For more information, see the sections “Noninferiority Test” on page 222 and “Equivalence Test” on page 224.

NONINF
NONINFERIORITY
requests a test of noninferiority for the binomial proportion. For more information, see the section “Noninferiority Test” on page 222. You can specify the noninferiority test margin, the null proportion, and the variance type in the MARGIN=, P=, and VAR= binomial-options, respectively. To request an exact noninferiority test, you can specify the BINOMIAL option in the EXACT statement.
OUTLEVEL
includes the variables LevelNumber and LevelValue in all ODS output data sets that PROC FREQTAB produces when you specify the BINOMIAL option in the TABLES statement. The OUTLEVEL option also includes the variables LevelNumber and LevelValue in the statistics output data set that PROC FREQTAB produces when you specify the BINOMIAL option in the OUTPUT statement.

The LevelNumber and LevelValue variables identify the analysis variable level for which PROC FREQTAB computes the binomial proportion. The value of LevelNumber is the order of the level in the one-way frequency table. The value of LevelValue is the formatted value of the level. You can specify the OUTLEVEL binomial-option with or without the LEVEL= binomial-option.

P=value
specifies the null hypothesis proportion for the binomial tests. The null proportion value must be a positive number. You can specify value as a number between 0 and 1. Or you can specify value in percentage form (as a number between 1 and 100), and PROC FREQTAB converts that number to a proportion. PROC FREQTAB treats the value 1 as 1%. By default, P=0.5.

SUP
SUPERIORITY
requests a test of superiority for the binomial proportion. For more information, see the section “Superiority Test” on page 223. You can specify the superiority test margin, the null proportion, and the variance type in the MARGIN=, P=, and VAR= binomial-options, respectively. To request an exact superiority test, you can specify the BINOMIAL option in the EXACT statement.

VAR=NULL | SAMPLE
specifies the type of variance to use in the Wald tests of noninferiority, superiority, and equivalence. If you specify VAR=SAMPLE, PROC FREQTAB computes the variance estimate by using the sample proportion. If you specify VAR=NULL, PROC FREQTAB computes a test-based variance by using the null hypothesis proportion (which you can specify in the P= binomial-option). For more information, see the sections “Noninferiority Test” on page 222 and “Equivalence Test” on page 224. The default is VAR=SAMPLE.

CELLCHI2
displays each table cell’s contribution to the Pearson chi-square statistic in the crosstabulation table. The cell chi-square is computed as \((frequency - expected)^2/expected\), where frequency is the table cell frequency (count) and expected is the expected cell frequency, which is computed under the null hypothesis that the row and column variables are independent. For more information, see the section “Pearson Chi-Square Test for Two-Way Tables” on page 203. This option has no effect for one-way tables or for tables that are displayed in list format (which you can request by specifying the LIST option).

CHISQ < (chisq-options)>
requests chi-square tests of homogeneity or independence and measures of association that are based on the chi-square statistic. For two-way tables, the chi-square tests include the Pearson chi-square, likelihood ratio chi-square, and Mantel-Haenszel chi-square tests. The chi-square measures include the phi coefficient, contingency coefficient, and Cramér’s V. For \(2 \times 2\) tables, the CHISQ option also provides Fisher’s exact test and the continuity-adjusted chi-square test. For more information, see the section “Chi-Square Tests and Statistics” on page 202.
For one-way tables, the CHISQ option provides the Pearson chi-square goodness-of-fit test. You can also request the likelihood ratio goodness-of-fit test for one-way tables by specifying the LRCHI chisq-option in parentheses after the CHISQ option. By default, the one-way chi-square tests are based on the null hypothesis of equal proportions. Alternatively, you can provide null hypothesis proportions or frequencies by specifying the TESTP= or TESTF= chisq-option, respectively. See the section “Chi-Square Test for One-Way Tables” on page 203 for more information.

To request Fisher’s exact test for tables larger than $2 \times 2$, specify the FISHER option in the EXACT statement. Exact $p$-values are also available for the Pearson, likelihood ratio, and Mantel-Haenszel chi-square tests. See the description of the EXACT statement for more information.

You can specify the following chisq-options:

**DF=df**

specifies the degrees of freedom for the chi-square tests. The value of $df$ must not be 0. If the value of $df$ is positive, PROC FREQTAB uses $df$ as the degrees of freedom for the chi-square tests. If the value of $df$ is negative, PROC FREQTAB uses $df$ to adjust the default degrees of freedom for the chi-square tests.

By default for one-way tables, the value of $df$ is $(n - 1)$, where $n$ is the number of variable levels in the table. By default for two-way tables, the value of $df$ is $(r - 1) (c - 1)$, where $r$ is the number of rows in the table and $c$ is the number of columns. See the sections “Chi-Square Test for One-Way Tables” on page 203 and “Chi-Square Tests and Statistics” on page 202 for more information.

If you specify a negative value of $df$, PROC FREQTAB adjusts the default degrees of freedom by adding the (negative) value of $df$ to the default value to produce the adjusted degrees of freedom. The adjusted degrees of freedom must be positive.

The DF= chisq-option specifies or adjusts the degrees of freedom for the following chi-square tests: the Pearson and likelihood ratio goodness-of-fit tests for one-way tables; and the Pearson, likelihood ratio, and Mantel-Haenszel chi-square tests for two-way tables.

**LRCHI**

requests the likelihood ratio goodness-of-fit test for one-way tables. See the section “Likelihood Ratio Chi-Square Test for One-Way Tables” on page 204 for more information.

By default, this test is based on the null hypothesis of equal proportions. You can provide null hypothesis proportions or frequencies by specifying the TESTP= or TESTF= chisq-option, respectively. You can request an exact likelihood ratio goodness-of-fit test by specifying the LRCHI option in the EXACT statement.

**TESTF=(values)**

specifies null hypothesis frequencies for chi-square tests for one-way tables (goodness-of-fit tests). For more information, see the section “Chi-Square Test for One-Way Tables” on page 203.

You can separate the values by using blanks or commas, and you must enclose the list of values in parentheses. The values must be positive numbers. The number of values must equal the number of variable levels in the one-way table. The sum of the values must equal the total frequency in the one-way table. List the values in the same order in which the corresponding variable levels appear in the output.
TESTP=(values)

specifies null hypothesis proportions for chi-square tests for one-way tables (goodness-of-fit tests). For more information, see the section “Chi-Square Test for One-Way Tables” on page 203.

You can separate the values by using blanks or commas, and you must enclose the list of values in parentheses. The values must be positive numbers. The number of values must equal the number of variable levels in the one-way table. List the values in the same order in which the corresponding variable levels appear in the output.

You can specify values in probability form as numbers between 0 and 1, where the proportions sum to 1. Or you can specify values in percentage form as numbers between 0 and 100, where the percentages sum to 100. PROC FREQTAB treats the value 1 as the percentage form 1%.

WARN=type | (types)

controls the warning message for the validity of the asymptotic Pearson chi-square test. By default, PROC FREQTAB displays a warning message when more than 20% of the table cells have expected frequencies that are less than 5. If you specify the NOPRINT option in the PROC FREQTAB statement, the procedure displays the warning in the log; otherwise, the procedure displays the warning as a footnote in the chi-square table. You can use the WARN= option to suppress the warning and to include a warning indicator in the output data set.

You can specify one or more of the following types in the WARN= option. If you specify more than one type value, enclose the values in parentheses after WARN=. For example, warn = (output noprint).

<table>
<thead>
<tr>
<th>Value of WARN=</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUTPUT</td>
<td>Adds a warning indicator variable to the output data set</td>
</tr>
<tr>
<td>NOLOG</td>
<td>Suppresses the chi-square warning message in the log</td>
</tr>
<tr>
<td>NOPRINT</td>
<td>Suppresses the chi-square warning message in the display</td>
</tr>
<tr>
<td>NONE</td>
<td>Suppresses the chi-square warning message entirely</td>
</tr>
</tbody>
</table>

If you specify the WARN=OUTPUT option, the ODS output data set ChiSq contains a variable named Warning that equals 1 for the Pearson chi-square observation when more than 20% of the table cells have expected frequencies that are less than 5 and equals 0 otherwise. If you specify WARN=OUTPUT and also specify the CHISQ option in the OUTPUT statement, the statistics output data set contains a variable named WARN_PCHI that indicates the warning.

The WARN=NOLOG option has the same effect as the NOWARN option in the TABLES statement.

CL

requests confidence limits for the measures of association, which you can request by specifying the MEASURES option. For more information, see the sections “Measures of Association” on page 208 and “Confidence Limits” on page 208. You can set the level of the confidence limits by using the ALPHA= option; by default, ALPHA=0.05, which produces 95% confidence limits.

If you omit the MEASURES option, the CL option invokes MEASURES. The CL option is equivalent to the MEASURES(CL) option.
CMH < (cmh-options) > requests Cochran-Mantel-Haenszel statistics, which test for association between the row and column variables after adjusting for the remaining variables in a multiway table. The Cochran-Mantel-Haenszel statistics include the nonzero correlation statistic, the row mean scores (ANOVA) statistic, and the general association statistic. In addition, for $2 \times 2$ tables, the CMH option provides the adjusted Mantel-Haenszel and logit estimates of the odds ratio and relative risks, together with their confidence limits. For stratified $2 \times 2$ tables, the CMH option provides the Breslow-Day test for homogeneity of odds ratios. (To request Tarone’s adjustment for the Breslow-Day test, specify the BDT cmh-option.) For more information, see the section “Cochran-Mantel-Haenszel Statistics” on page 261.

You can use the CMH1 or CMH2 option to control the number of CMH statistics that PROC FREQTAB computes.

For stratified $2 \times 2$ tables, you can request Zelen’s exact test for equal odds ratios by specifying the EQOR option in the EXACT statement. For more information, see the section “Zelen’s Exact Test for Equal Odds Ratios” on page 268. You can request exact confidence limits for the common odds ratio by specifying the COMOR option in the EXACT statement. This option also provides a common odds ratio test. For more information, see the section “Exact Confidence Limits for the Common Odds Ratio” on page 269.

You can specify the following cmh-options in parentheses after the CMH option. These cmh-options, which apply to stratified $2 \times 2$ tables, are also available with the CMH1 or CMH2 option.

BDT requests Tarone’s adjustment in the Breslow-Day test for homogeneity of odds ratios. For more information, see the section “Breslow-Day Test for Homogeneity of the Odds Ratios” on page 267.

GAILSIMPON < (COLUMN=1 | 2) >

GS < (COLUMN=1 | 2) > requests the Gail-Simon test for qualitative interaction, which applies to stratified $2 \times 2$ tables. For more information, see the section “Gail-Simon Test for Qualitative Interactions” on page 271.

The COLUMN= option specifies the column of the risk differences to use to compute the Gail-Simon test. By default, PROC FREQTAB uses column 1 risk differences. If you specify COLUMN=2, PROC FREQTAB uses column 2 risk differences.

The GAILSIMPON cmh-option has the same effect as the GAILSIMPON option in the TABLES statement.

I2 requests the I-square measure of heterogeneity for stratified $2 \times 2$ tables. I-square is computed from a Q test that is based on odds ratios. The I2 cmh-option invokes the QOR cmh-option. For more information, see the section “I-Square Measure of Heterogeneity” on page 268.

MANTELFLEISS requests the Mantel-Fleiss criterion for the Mantel-Haenszel statistic for stratified $2 \times 2$ tables. For more information, see the section “Mantel-Fleiss Criterion” on page 264.
QOR requests a Q test for heterogeneity of odds ratios for stratified $2 \times 2$ tables. For more information, see the section “Q Test for Homogeneity of Odds Ratios” on page 267.

CMH1 < (cmh-options) > requests the Cochran-Mantel-Haenszel correlation statistic. This option does not provide the CMH row mean scores (ANOVA) statistic or the general association statistic, which are provided by the CMH option. For tables larger than $2 \times 2$, the CMH1 option requires less memory than the CMH option, which can require an enormous amount of memory for large tables.

For $2 \times 2$ tables, the CMH1 option also provides the adjusted Mantel-Haenszel and logit estimates of the odds ratio and relative risks, together with their confidence limits. For stratified $2 \times 2$ tables, the CMH1 option provides the Breslow-Day test for homogeneity of odds ratios.

The cmh-options for CMH1 are the same as the cmh-options that are available with the CMH option. For more information, see the description of the CMH option.

CMH2 < (cmh-options) > requests the Cochran-Mantel-Haenszel correlation statistic and the row mean scores (ANOVA) statistic. This option does not provide the CMH general association statistic, which is provided by the CMH option. For tables larger than $2 \times 2$, the CMH2 option requires less memory than the CMH option, which can require an enormous amount of memory for large tables.

For $2 \times 2$ tables, the CMH1 option also provides the adjusted Mantel-Haenszel and logit estimates of the odds ratio and relative risks, together with their confidence limits. For stratified $2 \times 2$ tables, the CMH1 option provides the Breslow-Day test for homogeneity of odds ratios.

The cmh-options for CMH2 are the same as the cmh-options that are available with the CMH option. For more information, see the description of the CMH option.

COMMONRISKDIFF < options > requests the common (stratified) risk difference for multiway $2 \times 2$ tables, where the risk difference is the difference between the row 1 proportion and the row 2 proportion in a $2 \times 2$ table. By default, this option provides Mantel-Haenszel and summary score estimates of the common risk difference, together with their confidence limits. For more information, see the section “Common Risk Difference” on page 237.

You can specify the following options to request confidence limit types and tests for the common risk difference:

CL=type | (types)
requests confidence limits for the common risk difference. You can specify one or more types of confidence limits. When you specify only one type, you can omit the parentheses. You can specify CL=NONE to suppress the “Confidence Limits for the Common Risk Difference” table.

You can specify the confidence level in the ALPHA= option. By default, ALPHA=0.05, which produces 95% confidence limits for the common risk difference.

You can specify one or more of the following types:
K

**KLINGENBERG**

requests Klingenberg confidence limits for the Mantel-Haenszel common risk difference. For more information, see the section “Klingenberg Confidence Limits” on page 238.

MH

requests Mantel-Haenszel confidence limits, which are computed by using Mantel-Haenszel stratum weights and the Sato variance estimator (Sato 1989). For more information, see the section “Mantel-Haenszel Confidence Limits and Test” on page 237.

MR

**MINRISK**

requests minimum risk confidence limits, which are computed by using minimum risk weights. For more information, see the section “Minimum Risk Confidence Limits and Test” on page 238.

NEWCOMBE

requests stratified Newcombe confidence limits that use Mantel-Haenszel weights to combine the stratum components. For more information, see the section “Stratified Newcombe Confidence Limits” on page 240.

NEWCOMBER

requests stratified Newcombe confidence limits that use minimum risk weights to combine the stratum components. For more information, see the section “Stratified Newcombe Confidence Limits” on page 240.

NONE

suppresses the “Confidence Limits for the Common Risk Difference” table.

SCORE

requests summary score confidence limits. For more information, see the section “Summary Score Confidence Limits” on page 240.

**COLUMN=1 | 2**

specifies the table column for which to compute the common risk difference statistics. If you do not specify this option but you do specify the RISKDIFF(COLUMN=) option, PROC FREQTAB provides the common risk difference statistics for the column that you specify in the RISKDIFF(COLUMN=) option. If you do not specify either of these options, COLUMN=1 by default.

**CORRECT=NO**

removes the continuity correction in the minimum risk confidence limits and in the minimum risk test, which you can request by specifying the CL=MR and TEST=MR options, respectively. For more information, see the section “Minimum Risk Confidence Limits and Test” on page 238.

**PRINTWTS < =type | (types)>**

displays the stratum weights together with the stratum risk differences and frequencies. By default, this option displays the weight type or types for the confidence limits and tests that you request. Optionally, you can specify the weight type to display.

You can specify one or more of the following **types**:
MH  
   displays Mantel-Haenszel stratum weights. For more information, see the section “Mantel-Haenszel Confidence Limits and Test” on page 237.

MR  
   displays minimum risk stratum weights. For more information, see the section “Minimum Risk Confidence Limits and Test” on page 238.

SCORE  
   displays summary score stratum weights. For more information, see the section “Summary Score Confidence Limits” on page 240.

TEST \( \text{<=type | (types)} > \)  
   requests common risk difference tests. You can specify one or more types. When you specify only one type, you can omit the parentheses. If you do not specify types, this option provides tests that correspond to the confidence limit types that you specify in the CL= option.

   You can specify one or more of the following types:

   MH  
      requests a Mantel-Haenszel test, which is computed by using Mantel-Haenszel stratum weights and the Sato variance estimator (Sato 1989). For more information, see the section “Mantel-Haenszel Confidence Limits and Test” on page 237.

   MR \( \text{< (VAR=SAMPLE)} > \)  
      MINRISK \( \text{< (VAR=SAMPLE)} > \)  
      requests the minimum risk test, which is computed by using minimum risk weights. If you specify VAR=SAMPLE, PROC FREQTAB uses the sample (observed) variance estimate instead of a null variance estimate to compute the minimum risk test statistic. For more information, see the section “Minimum Risk Confidence Limits and Test” on page 238.

   SCORE  
      requests the summary score test. For more information, see the section “Summary Score Confidence Limits” on page 240.

CONTENTS=’string’  
   specifies the label to use for crosstabulation tables in the contents file, the Results window, and the trace record. For information about output presentation, see the SAS Output Delivery System: User’s Guide.

   If you omit the CONTENTS= option, the contents label for crosstabulation tables is “Cross-Tabular Freq Table” by default.

   Note that contents labels for all crosstabulation tables that are produced by a single TABLES statement use the same text. To specify different contents labels for different crosstabulation tables, request the tables in separate TABLES statements and use the CONTENTS= option in each TABLES statement.

   To remove the crosstabulation table entry from the contents file, you can specify a null label with CONTENTS=’’.

   The CONTENTS= option affects only contents labels for crosstabulation tables. It does not affect contents labels for other PROC FREQTAB tables.
To specify the contents label for any PROC FREQTAB table, you can use PROC TEMPLATE to create a customized table template. The CONTENTS_LABEL attribute in the DEFINE TABLE statement of PROC TEMPLATE specifies the contents label for the table. See the chapter “The TEMPLATE Procedure” in the *SAS Output Delivery System: User’s Guide* for more information.

**CROSSLIST < (options) >**

displays crosstabulation tables by using an ODS column format instead of the default crosstabulation cell format. In the CROSSLIST table display, the rows correspond to the crosstabulation table cells, and the columns correspond to descriptive statistics such as frequencies and percentages. The CROSSLIST table displays the same information as the default crosstabulation table (but it uses an ODS column format). For more information about the contents of the CROSSLIST table, See the section “Two-Way and Multiway Tables” on page 280.

You can control the contents of a CROSSLIST table by specifying the same options available for the default crosstabulation table. These include the NOFREQ, NOPERCENT, NOROW, and NOCOL options. You can request additional information in a CROSSLIST table by specifying the CELLCHI2, DEVIATION, EXPECTED, MISSPRINT, and TOTPCT options. You can also display standardized residuals or Pearson residuals in a CROSSLIST table by specifying the CROSSLIST(STDRES) or CROSSLIST(PEARSONRES) option, respectively; these options are not available for the default crosstabulation table. The FORMAT= and CUMCOL options have no effect on CROSSLIST tables. You cannot specify both the LIST option and the CROSSLIST option in the same TABLES statement.

You can specify the NOSPARSE option along with the CROSSLIST option to suppress variable levels that have frequencies of 0. By default for CROSSLIST tables, PROC FREQTAB displays all levels of the column variable within each level of the row variable, including any levels that have frequencies of 0. By default for multiway CROSSLIST tables, PROC FREQTAB displays all levels of the row variable within each stratum of the table, including any row levels that have frequencies of 0 in the stratum.

You can specify the following options:

**STDRES**

displays the standardized residuals of the table cells in the CROSSLIST table. The standardized residual is the ratio of \((frequency - expected)\) to its standard error, where \(frequency\) is the table cell frequency (count) and \(expected\) is the expected table cell frequency, which is computed under the null hypothesis that the row and column variables are independent. For more information, see the section “Standardized Residuals” on page 204. You can display the expected values and deviations by specifying the EXPECTED and DEVIATION options, respectively.

**PEARSONRES**

displays the Pearson residuals of the table cells in the CROSSLIST table. The Pearson residual is the square root of the table cell’s contribution to the Pearson chi-square statistic. The Pearson residual is computed as \((frequency - expected)/\sqrt{expected}\), where \(frequency\) is the table cell frequency (count) and \(expected\) is the expected table cell frequency, which is computed under the null hypothesis that the row and column variables are independent. For more information, see the section “Pearson Chi-Square Test for Two-Way Tables” on page 203. You can display the expected values, deviations, and cell chi-squares by specifying the EXPECTED, DEVIATION, and CELLCHI2 options, respectively.
CUMCOL
displays the cumulative column percentages in the cells of the crosstabulation table. The CUMCOL option does not apply to crosstabulation tables produced with the LIST or CROSSLIST option.

DEVIATION
displays the deviations of the frequencies from the expected frequencies (frequency – expected) in the crosstabulation table. The expected frequencies are computed under the null hypothesis that the row and column variables are independent. For more information, see the section “Pearson Chi-Square Test for Two-Way Tables” on page 203. You can display the expected values by specifying the EXPECTED option. This option has no effect for one-way tables or for tables that are displayed in list format (which you can request by specifying the LIST option).

EXPECTED
displays the expected cell frequencies in the crosstabulation table. The expected frequencies are computed under the null hypothesis that the row and column variables are independent. For more information, see the section “Pearson Chi-Square Test for Two-Way Tables” on page 203. This option has no effect for one-way tables or for tables that are displayed in list format (which you can request by specifying the LIST option).

FISHER
requests Fisher’s exact test for tables that are larger than $2 \times 2$. (For $2 \times 2$ tables, the CHISQ option provides Fisher’s exact test.) This test is also known as the Freeman-Halton test. See the sections “Fisher’s Exact Test” on page 206 and “Exact Statistics” on page 271 for more information.

If you omit the CHISQ option in the TABLES statement, the FISHER option invokes CHISQ. You can also request Fisher’s exact test by specifying the FISHER option in the EXACT statement.

NOTE: PROC FREQTAB computes exact tests by using fast and efficient algorithms that are superior to direct enumeration. Exact tests are appropriate when a data set is small, sparse, skewed, or heavily tied. For some large problems, computation of exact tests might require a substantial amount of time and memory. Consider using asymptotic tests for such problems. Alternatively, when asymptotic methods might not be sufficient for such large problems, consider using Monte Carlo estimation of exact $p$-values. You can request Monte Carlo estimation by specifying the MC computation-option in the EXACT statement. See the section “Computational Resources” on page 274 for more information.

FORMAT=format-name
specifies a format for the following crosstabulation table cell values: frequency, expected frequency, and deviation. PROC FREQTAB also uses the specified format to display the row and column total frequencies and the overall total frequency in crosstabulation tables.

You can specify any standard SAS numeric format or a numeric format defined with the FORMAT procedure. The format length must not exceed 24. If you omit the FORMAT= option, by default PROC FREQTAB uses the BEST6. format to display frequencies less than 1E6, and the BEST7. format otherwise.

The FORMAT= option applies only to crosstabulation tables displayed in the default format. It does not apply to crosstabulation tables produced with the LIST or CROSSLIST option.

To change display formats in any table that PROC FREQTAB produces, you can use PROC TEMPLATE. For more information, see the chapter “The TEMPLATE Procedure” in the SAS Output Delivery System: User’s Guide.
GAILSIMON < (COLUMN=1 | 2) >

GS < (COLUMN=1 | 2) >
requests the Gail-Simon test for qualitative interaction, which applies to stratified 2 x 2 tables. For more information, see the section “Gail-Simon Test for Qualitative Interactions” on page 271.

The COLUMN= option specifies the column of the risk differences to use to compute the Gail-Simon test. By default, PROC FREQTAB uses column 1 risk differences. If you specify COLUMN=2, PROC FREQTAB uses column 2 risk differences.

JT
requests the Jonckheere-Terpstra test. For more information, see the section “Jonckheere-Terpstra Test” on page 252. To request exact p-values for the Jonckheere-Terpstra test, specify the JT option in the EXACT statement. See the section “Exact Statistics” on page 271 for more information.

LIST
displays two-way and multiway tables by using a list format instead of the default crosstabulation cell format. This option displays an entire multiway table in a single table instead of separate two-way (stratum) tables. Each row of a table in list format corresponds to a single crosstabulation table cell. For more information, see the section “Two-Way and Multiway Tables” on page 280.

A table in list format does not display table cells for which the total frequency is 0 unless you specify the ZEROS option in the WEIGHT statement.

The LIST option is not available together with statistic options in the same TABLES statement. When you specify statistic options in a TABLES statement, you must display the crosstabulation tables by using the default table cell format or the CROSSLIST format. You can specify the LIST option and statistic options in different TABLES statements in the same invocation of PROC FREQTAB.

MAXLEVELS=n
specifies the maximum number of variable levels to display in one-way frequency tables. The value of n must be a positive integer. PROC FREQTAB displays the first n variable levels, matching the order in which the levels appear in the one-way frequency table. (The ORDER= option controls the order of the variable levels. By default, ORDER=INTERNAL, which orders the variable levels by unformatted value.)

The MAXLEVELS= option also applies to one-way frequency plots, which you can request by specifying the PLOTS=FREQPLOT option when ODS Graphics is enabled.

If you specify the MISSPRINT option to display missing levels in the frequency table, the MAXLEVELS= option displays the first n nonmissing levels.

The MAXLEVELS= option does not apply to the OUT= output data set, which includes all variable levels. The MAXLEVELS= option does not affect the computation of percentages, statistics, or tests for the one-way table; these values are based on the complete table.

MEASURES < (CL) >
requests measures of association and their asymptotic standard errors. This option provides the following measures: gamma, Kendall’s tau-b, Stuart’s tau-c, Somers’ D(C|R), Somers’ D(R|C), Pearson and Spearman correlation coefficients, lambda (symmetric and asymmetric), and uncertainty coefficients (symmetric and asymmetric). If you specify the CL option in parentheses after the MEASURES option, PROC FREQTAB provides confidence limits for the measures of association. For more information, see the section “Measures of Association” on page 208.
For $2 \times 2$ tables, the MEASURES option also provides the odds ratio, column 1 relative risk, column 2 relative risk, and their asymptotic Wald confidence limits. You can request the odds ratio and relative risks separately (without the other measures of association) by specifying the RELRISK option. You can request confidence limits for the odds ratio by specifying the OR(CL=) option.

You can use the TEST statement to request asymptotic tests for the following measures of association: Kendall’s tau-$b$, Stuart’s tau-$c$, Somers’ $D(C \mid R)$, Somers’ $D(R \mid C)$, and Pearson and Spearman correlation coefficients. You can use the EXACT statement to request exact confidence limits for the odds ratio, exact unconditional confidence limits for the relative risks, and exact tests for the following measures of association: Kendall’s tau-$b$, Stuart’s tau-$c$, Somers’ $D(C \mid R)$ and $D(R \mid C)$, and Pearson and Spearman correlation coefficients. For more information, see the descriptions of the TEST and EXACT statements and the section “Exact Statistics” on page 271.

NOCOL
suppresses the display of column percentages in crosstabulation table cells.

NOCUM
suppresses the display of cumulative frequencies and percentages in one-way frequency tables and in list-format crosstabulation tables (which you can request by specifying the LIST option).

NOFREQ
suppresses the display of table cell frequencies in crosstabulation tables. The NOFREQ option also suppresses row total frequencies. This option has no effect for one-way tables or for list-format crosstabulation tables (which you can request by specifying the LIST option).

NOPERCENT
suppresses the display of overall percentages in crosstabulation tables. These percentages include the tables cell percentages, row percentages, and column percentages of the total two-way table frequency. To suppress display of cell percentages of row or column totals, you can specify the NOROW or NOCOL option, respectively.

For one-way frequency tables and crosstabulation tables in list format, the NOPERCENT option suppresses the display of percentages and cumulative percentages.

NOPRINT
suppresses the display of frequency and crosstabulation tables but displays all requested tests and statistics. To suppress the display of all output, including tests and statistics, use the NOPRINT option in the PROC FREQTAB statement.

NOROW
suppresses the display of row percentages in crosstabulation table cells.

NOSPARSE
suppresses zero-frequency cells in the LIST table, CROSSLIST table, and OUT= data set.

The NOSPARSE option is available when you specify the ZEROS option in the WEIGHT statement, which includes observations that have weights of 0. By default, the ZEROS option displays zero-frequency cells in the LIST table and includes them in the OUT= data set; the NOSPARSE option suppresses the zero-frequency cells. For more information, see the description of the ZEROS option.

The NOSPARSE option is also available when you specify the CROSSLIST option. By default for CROSSLIST tables, PROC FREQTAB displays all levels of the column variable within each level
of the row variable, which includes any levels that have frequencies of 0. By default for multiway CROSSLIST tables, PROC FREQTAB displays all levels of the row variable within each stratum of the table, which includes any row levels that have 0 frequencies in the stratum. The NOSPARSE option suppresses the zero-frequency levels in the CROSSLIST table.

**NOWARN**
suppresses the log warning message for the validity of the asymptotic Pearson chi-square test. By default, PROC FREQTAB provides a validity warning for the asymptotic Pearson chi-square test when more than 20% of the table cells have expected frequencies that are less than 5. This warning message appears in the log if you specify the NOPRINT option in the PROC FREQTAB statement.

The NOWARN option is equivalent to the CHISQ(WARN=NOLOG) option. You can also use the CHISQ(WARN=) option to suppress the warning message in the display and to request a warning variable in the chi-square ODS output data set or in the OUTPUT data set.

**OR <(CL=type | (types )>**
requests the odds ratio and confidence limits for 2 × 2 tables. For more information, see the section “Odds Ratio” on page 241.

You can specify one or more *types* of confidence limits. When you specify only one confidence limit *type*, you can omit the parentheses around the request. PROC FREQTAB displays the confidence limits in the “Confidence Limits for the Odds Ratio” table.

Specifying the OR option without the CL= option is equivalent to specifying the RELRISK option, which produces the “Odds Ratio and Relative Risks” table. For more information, see the description of the RELRISK option. When you specify the OR(CL=) option, PROC FREQTAB does not produce the “Odds Ratio and Relative Risks” table unless you also specify the RELRISK or MEASURES option.

The ALPHA= option determines the confidence level; by default, ALPHA=0.05, which produces 95% confidence limits for the odds ratio.

You can specify the following *types*:

**EXACT**
displays exact confidence limits for the odds ratio in the “Confidence Limits for the Odds Ratio” table. (By default, PROC FREQTAB displays the exact confidence limits in a separate table.)
You must also request computation of the exact confidence limits by specifying the OR option in the EXACT statement. For more information, see the subsection “Exact Confidence Limits” in the section “Confidence Limits for the Odds Ratio” on page 242.

**LR**

**LIKELIHOODRATIO**
requests likelihood ratio confidence limits for the odds ratio. For more information, see the subsection “Likelihood Ratio Confidence Limits” in the section “Confidence Limits for the Odds Ratio” on page 242.

**MIDP**
requests exact mid-$p$ confidence limits for the odds ratio. For more information, see the subsection “Exact Mid-$p$ Confidence Limits” in the section “Confidence Limits for the Odds Ratio” on page 242.
SCORE \(<\text{CORRECT=NO}>\>
requests score confidence limits for the odds ratio. For more information, see the subsection “Score Confidence Limits” in the section “Confidence Limits for the Odds Ratio” on page 242. If you specify CORRECT=NO, PROC FREQTAB provides the uncorrected form of the score confidence limits.

WALD
requests asymptotic Wald confidence limits, which are based on a log transformation of the odds ratio. For more information, see the subsection “Wald Confidence Limits” in the section “Confidence Limits for the Odds Ratio” on page 242.

WALDMODIFIED
requests Wald modified confidence limits for the odds ratio. For more information, see the subsection “Wald Modified Confidence Limits” in the section “Confidence Limits for the Odds Ratio” on page 242.

OUT=\text{sas-data-set}
names an output data set that contains frequency or crosstabulation table counts and percentages. If more than one table request appears in the TABLES statement, the contents of the OUT= data set correspond to the last table request in the TABLES statement. The OUT= data set variable COUNT contains the frequencies and the variable PERCENT contains the percentages. For more information, see the section “Contents of the TABLES Statement Output Data Set” on page 276. You can specify the following options to include additional information in the OUT= data set: OUTCUM, OUTEXPECT, and OUTPCT.

OUTCUM
includes cumulative frequencies and cumulative percentages in the OUT= data set. The variable CUM_FREQ contains the cumulative frequencies, and the variable CUM_PCT contains the cumulative percentages. For more information, see the section “Contents of the TABLES Statement Output Data Set” on page 276. This option is available for one-way frequency tables and for tables that you display in list format (which you can request by specifying the LIST option).

OUTEXPECT
includes expected cell frequencies in the OUT= data set for crosstabulation tables. The variable EXPECTED contains the expected cell frequencies. For more information, see the section “Contents of the TABLES Statement Output Data Set” on page 276. This option has no effect for one-way tables or for tables that you display in list format (which you can request by specifying the LIST option).

OUTPCT
includes the following additional variables in the OUT= data set for crosstabulation tables:

\begin{itemize}
  \item PCT_COL percentage of column frequency
  \item PCT_ROW percentage of row frequency
  \item PCT_TABL percentage of stratum (two-way table) frequency, for \(n\)-way tables where \(n > 2\)
\end{itemize}

For more information, see the section “Contents of the TABLES Statement Output Data Set” on page 276. This option has no effect for one-way tables or for tables that you display in list format (which you can request by specifying the LIST option).
PLCORR < (options) >
POLYCHORIC < (options) >
requests the polychoric correlation coefficient and its asymptotic standard error. For 2 × 2 tables, this statistic is more commonly known as the tetrachoric correlation coefficient and is labeled as such in the displayed output. For more information, see the section “Polychoric Correlation” on page 214.

If you also specify the CL or MEASURES(CL) option, PROC FREQTAB provides confidence limits for the polychoric correlation. If you specify the PLCORR option in the TEST statement, the procedure provides Wald and likelihood ratio tests for the polychoric correlation. The PLCORR option invokes the MEASURES option.

You can specify the following options:

ADJUST
replaces a 2 × 2 table cell frequency of 0 by 0.5 before computing the tetrachoric correlation (Brown and Benedetti 1977a, p. 353). To maintain the row and column marginal frequencies, adjacent cell frequencies are decreased by 0.5 and the opposite cell frequency is increased by 0.5.

This option is available for 2 × 2 tables and is applied only when a single cell frequency is 0. It has no effect when both off-diagonal cell frequencies are 0 (and therefore the correlation is 1) or when both diagonal cell frequencies are 1 (and therefore the correlation is −1).

CONVERGE=value
specifies the convergence criterion. The value must be a positive number. By default, CONVERGE=0.0001. Iterative computation of the polychoric correlation stops when the convergence measure falls below value or when the number of iterations exceeds the MAXITER= number, whichever happens first. For parameter values that are less than 0.01, PROC FREQTAB evaluates convergence by using the absolute difference instead of the relative difference. For more information, see the section “Polychoric Correlation” on page 214.

MAXITER=number
specifies the maximum number of iterations. The value of number must be a positive integer. By default, MAXITER=50. Iterative computation of the polychoric correlation stops when the number of iterations exceeds the maximum number or when the convergence measure falls below the CONVERGE= value, whichever happens first. For more information, see the section “Polychoric Correlation” on page 214.

PLOTS < (global-plot-options) > < =plot-request < (plot-options) >>
PLOTS < (global-plot-options) > < =plot-request < (plot-options) >> < ... plot-request < (plot-options) >> >
controls the plots that are produced through ODS Graphics. Plot-requests identify the plots, and plot-options control the appearance and content of the plots. You can specify plot-options in parentheses after a plot-request. A global-plot-option applies to all plots for which it is available unless it is altered by a specific plot-option. You can specify global-plot-options in parentheses after the PLOTS option.

When you specify only one plot-request, you can omit the parentheses around the request. For example:

plots=all
plots=freqplot
plots=(freqplot oddsratioplot)
plots(only)=(cumfreqplot deviationplot)

ODS Graphics must be enabled before plots can be requested. For example:

```plaintext
ods graphics on;
proc freqtab;
   tables treatment*response / chisq plots=freqplot;
   weight wt;
run;
ods graphics off;
```

If ODS Graphics is enabled but you do not specify the PLOTS= option, PROC FREQTAB produces all plots that are associated with the analyses that you request, with the exception of the frequency, cumulative frequency, and mosaic plots. To produce a frequency plot or cumulative frequency plot when ODS Graphics is enabled, you must specify the FREQPLOT or CUMFREQPLOT plot-request, respectively, in the PLOTS= option, or you must specify the PLOTS=ALL option. To produce a mosaic plot when ODS Graphics is enabled, you must specify the MOSAICPLOT plot-request in the PLOTS= option, or you must specify the PLOTS=ALL option.

PROC FREQTAB produces the remaining plots (listed in Table 5.10) by default when you request the corresponding TABLES statement options. You can suppress default plots and request specific plots by using the PLOTS(ONLY)= option; PLOTS(ONLY)=(plot-requests) produces only the plots that are specified as plot-requests. You can suppress all plots by specifying the PLOTS=NONE option. The PLOTS option has no effect when you specify the NOPRINT option in the PROC FREQTAB statement.

**Plot Requests**

Table 5.10 lists the available plot-requests together with their required TABLES statement options. Descriptions of the plot-requests follow the table in alphabetical order.

<table>
<thead>
<tr>
<th>Plot Request</th>
<th>Description</th>
<th>Required TABLES Statement Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGREEPLOT</td>
<td>Agreement plot</td>
<td>AGREE ($r 	imes r$ table)</td>
</tr>
<tr>
<td>ALL</td>
<td>All plots</td>
<td>None</td>
</tr>
<tr>
<td>CUMFREQPLOT</td>
<td>Cumulative frequency plot</td>
<td>One-way table request</td>
</tr>
<tr>
<td>DEVIATIONPLOT</td>
<td>Deviation plot</td>
<td>CHISQ (one-way table)</td>
</tr>
<tr>
<td>FREQPLOT</td>
<td>Frequency plot</td>
<td>Any table request</td>
</tr>
<tr>
<td>KAPPAPLOT</td>
<td>Kappa plot</td>
<td>AGREE ($h 	imes r 	imes r$ table)</td>
</tr>
<tr>
<td>MOSAICPLOT</td>
<td>Mosaic plot</td>
<td>Two-way or multiway table request</td>
</tr>
<tr>
<td>NONE</td>
<td>No plots</td>
<td>None</td>
</tr>
<tr>
<td>ODDSRATIOPLOT</td>
<td>Odds ratio plot</td>
<td>MEASURES, OR, or RELRISK ($h 	imes 2 	imes 2$ table)</td>
</tr>
<tr>
<td>RELRISKPLOT</td>
<td>Relative risk plot</td>
<td>MEASURES or RELRISK ($h 	imes 2 	imes 2$ table)</td>
</tr>
<tr>
<td>RISKDIFFPLOT</td>
<td>Risk difference plot</td>
<td>RISKDIFF ($h 	imes 2 	imes 2$ table)</td>
</tr>
<tr>
<td>WTKAPPAPLOT</td>
<td>Weighted kappa plot</td>
<td>AGREE ($h 	imes r 	imes r$ table, $r &gt; 2$)</td>
</tr>
</tbody>
</table>
You can specify the following plot-requests:

**AGREEPLOT < (plot-options) >**
requests an agreement plot (Bangdiwala and Bryan 1987). An agreement plot displays the strength of agreement in a two-way table, where the row and column variables represent two independent ratings of \( n \) subjects. For information about agreement plots, see Bangdiwala (1988), Bangdiwala et al. (2008), and Friendly (2000, Section 3.7.2).

To produce an agreement plot, you must also specify the AGREE option in the TABLES statement. Agreement statistics and plots are available for two-way square tables, where the number of rows equals the number of columns.

Table 5.11 lists the plot-options that are available for agreement plots. For descriptions of the plot-options, see the subsection “Plot Options” in this section.

<table>
<thead>
<tr>
<th>Plot Option</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>LEGEND=</td>
<td>Legend</td>
<td>NO or YES*</td>
</tr>
<tr>
<td>PARTIAL=</td>
<td>Partial agreement</td>
<td>NO or YES*</td>
</tr>
<tr>
<td>SHOWSCALE=</td>
<td>Frequency scale</td>
<td>NO or YES*</td>
</tr>
<tr>
<td>STATS</td>
<td>Statistics</td>
<td>None</td>
</tr>
</tbody>
</table>

*Default

If you specify the STATS plot-option, the agreement plot displays the values of the kappa coefficient, the weighted kappa coefficient, the \( B_n \) measure (Bangdiwala and Bryan 1987), and the sample size. PROC FREQTAB stores these statistics in an ODS table named \( B_n \text{Measure} \), which is not displayed. For more information, see the section “ODS Table Names” on page 288.

**ALL**
requests all plots that are associated with the specified analyses. Table 5.10 lists the available plot-requests and the corresponding analysis options. If you specify the PLOTS=ALL option, PROC FREQTAB produces the frequency, cumulative frequency, and mosaic plots that are associated with the tables that you request. (These plots are not produced by default when ODS Graphics is enabled.)

**CUMFREQPLOT < (plot-options) >**
requests a plot of cumulative frequencies. Cumulative frequency plots are available for one-way frequency tables.

To produce a cumulative frequency plot, you must specify the CUMFREQPLOT plot-request in the PLOTS= option, or you must specify the PLOTS=ALL option. PROC FREQTAB does not produce cumulative frequency plots by default when ODS Graphics is enabled.

Table 5.12 lists the plot-options that are available for cumulative frequency plots. For descriptions of the plot-options, see the subsection “Plot Options” in this section.
**Table 5.12**  Plot Options for CUMFREQPLOT

<table>
<thead>
<tr>
<th>Plot Option</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORIENT=</td>
<td>Orientation</td>
<td>HORIZONTAL or VERTICAL*</td>
</tr>
<tr>
<td>SCALE=</td>
<td>Scale</td>
<td>FREQ* or PERCENT</td>
</tr>
<tr>
<td>TYPE=</td>
<td>Type</td>
<td>BARCHART* or DOTPLOT</td>
</tr>
</tbody>
</table>

*Default

**DEVIATIONPLOT** <(plot-options)> requests a plot of relative deviations from expected frequencies. Deviation plots are available for chi-square analysis of one-way frequency tables. To produce a deviation plot, you must also specify the CHISQ option in the TABLES statement for a one-way frequency table.

Table 5.13 lists the **plot-options** that are available for deviation plots. For descriptions of the **plot-options**, see the subsection “Plot Options” in this section.

**Table 5.13**  Plot Options for DEVIATIONPLOT

<table>
<thead>
<tr>
<th>Plot Option</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOSTAT</td>
<td>No statistic</td>
<td>None</td>
</tr>
<tr>
<td>ORIENT=</td>
<td>Orientation</td>
<td>HORIZONTAL or VERTICAL*</td>
</tr>
<tr>
<td>TYPE=</td>
<td>Type</td>
<td>BARCHART* or DOTPLOT</td>
</tr>
</tbody>
</table>

*Default

**FREQPLOT** <(plot-options)> requests a frequency plot. Frequency plots are available for frequency and crosstabulation tables. For multiway crosstabulation tables, PROC FREQTAB provides a two-way frequency plot for each stratum (two-way table).

To produce a frequency plot, you must specify the FREQPLOT **plot-request** in the PLTS= option, or you must specify the PLTS=ALL option. PROC FREQTAB does not produce frequency plots by default when ODS Graphics is enabled.

Table 5.14 lists the **plot-options** that are available for frequency plots. For descriptions of the **plot-options**, see the subsection “Plot Options” in this section.
You can specify the following plot-options for all frequency plots: ORIENT=, SCALE=, and TYPE=. You can specify the following plot-options for frequency plots of two-way (and multiway) tables: GROUPBY=, NPANELPOS=, and TWOWAY=. The NPANELPOS= plot-option is not available with the TWOWAY=CLUSTER or TWOWAY=STACKED layout, which is always displayed in a single panel.

By default, PROC FREQTAB displays frequency plots as bar charts. To display frequency plots as dot plots, specify TYPE=DOTPLOT. To plot percentages instead of frequencies, specify SCALE=PERCENT. For two-way tables, there are four frequency plot layouts available, which you can request by specifying the TWOWAY= plot-option. For more information, see the subsection “Plot Options” in this section.

By default, graph cells in a two-way layout are first grouped by column variable levels; row variable levels are then displayed within the column variable levels. To group first by row variable levels, specify GROUPBY=ROW.

**KAPPAPLOT < (plot-options) >**

requests a plot of kappa statistics along with confidence limits. Kappa plots are available for multiway square tables and display the kappa statistic (with confidence limits) for each two-way table (stratum). Kappa plots also display the overall kappa statistic unless you specify the COMMON=NO plot-option. To produce a kappa plot, you must specify the AGREE option in the TABLES statement to compute kappa statistics.

Table 5.15 lists the plot-options that are available for kappa plots. For descriptions of the plot-options, see the subsection “Plot Options” in this section.
MOSAICPLOT <(plot-options)> requests a mosaic plot. Mosaic plots are available for two-way and multiway crosstabulation tables; for multiway tables, PROC FREQTAB provides a mosaic plot for each two-way table (stratum).

To produce a mosaic plot, you must specify the MOSAICPLOT plot-request in the PLOTS= option, or you must specify the PLOTS=ALL option. PROC FREQTAB does not produce mosaic plots by default when ODS Graphics is enabled.

Mosaic plots display tiles that correspond to the crosstabulation table cells. The areas of the tiles are proportional to the frequencies of the table cells. The column variable is displayed on the X axis, and the tile widths are proportional to the relative frequencies of the column variable levels. The row variable is displayed on the Y axis, and the tile heights are proportional to the relative frequencies of the row levels within column levels. For more information, see Friendly (2000).

By default, the colors of the tiles correspond to the row variable levels. If you specify the COLORSTAT= plot-option, the tiles are colored according to the values of the Pearson or standardized residuals.

You can specify the following plot-options:

COLORSTAT < =PEARSONRES | STDRES >

colors the mosaic plot tiles according to the values of residuals. If you specify COLORSTAT=PEARSONRES, the tiles are colored according to the Pearson residuals of the corresponding table cells. For more information, see the section “Pearson Chi-Square Test for Two-Way Tables” on page 203. If you specify COLORSTAT=STDRES, the tiles are colored according to the standardized residuals of the corresponding table cells. For more information, see the section “Standardized Residuals” on page 204. You can display the Pearson or standardized residuals in the CROSSLIST table by specifying the CROSSLIST(PEARSONRES) or CROSSLIST(STDRES) option, respectively.

SQUARE
produces a square mosaic plot, where the height of the Y axis equals the width of the X axis. In a square mosaic plot, the scale of the relative frequencies is the same on both axes. By default, PROC FREQTAB produces a rectangular mosaic plot.

---

Table 5.15 Plot Options for KAPPAPLOT and WTKAPPAPLOT

<table>
<thead>
<tr>
<th>Plot Option</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLDISPLAY=</td>
<td>Error bar type</td>
<td>BAR, LINE, LINEARROW, SERIF*, or SERIFARROW</td>
</tr>
<tr>
<td>COMMON=</td>
<td>Overall kappa</td>
<td>NO or YES*</td>
</tr>
<tr>
<td>NPARLEVELS=</td>
<td>Statistics per graphic</td>
<td>Number (all*)</td>
</tr>
<tr>
<td>ORDER=</td>
<td>Order of two-way levels</td>
<td>ASCENDING or DESCENDING</td>
</tr>
<tr>
<td>RANGE=</td>
<td>Range to display</td>
<td>Values or CLIP</td>
</tr>
<tr>
<td>STATS</td>
<td>Statistic values</td>
<td>None</td>
</tr>
</tbody>
</table>

*Default
NONE
suppresses all plots.

ODDSRATIOPLOT < (plot-options)>
requests a plot of odds ratios along with confidence limits. Odds ratio plots are available for multiway $2 \times 2$ tables and display the odds ratio (with confidence limits) for each $2 \times 2$ table (stratum). To produce an odds ratio plot, you must also specify the MEASURES, OR, or RELRISK option in the TABLES statement to compute the odds ratios.

Table 5.16 lists the plot-options that are available for odds ratio plots. For descriptions of the plot-options, see the subsection “Plot Options” in this section.

Table 5.16 Plot Options for ODDSRATIOPLOT, RELRISKPLOT, and RISKDIFFPLOT

<table>
<thead>
<tr>
<th>Plot Option</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>CL=</td>
<td>Confidence limit type</td>
<td>Type</td>
</tr>
<tr>
<td>CLDISPLAY=</td>
<td>Error bar type</td>
<td>BAR, LINE, LINEARROW, SERIF*, or SERIFARROW</td>
</tr>
<tr>
<td>COLUMN=**</td>
<td>Risk column</td>
<td>1* or 2</td>
</tr>
<tr>
<td>COMMON=</td>
<td>Common value</td>
<td>NO or YES*</td>
</tr>
<tr>
<td>LOGBASE=***</td>
<td>Axis scale</td>
<td>2, E, or 10</td>
</tr>
<tr>
<td>NPARMPOS=</td>
<td>Statistics per graphic</td>
<td>Number (all*)</td>
</tr>
<tr>
<td>ORDER=</td>
<td>Order of two-way levels</td>
<td>ASCENDING or DESCENDING</td>
</tr>
<tr>
<td>RANGE=</td>
<td>Range to display</td>
<td>Values or CLIP</td>
</tr>
<tr>
<td>STATS</td>
<td>Statistic values</td>
<td>None</td>
</tr>
</tbody>
</table>

*Default
**Available for RELRISKPLOT and RISKDIFFPLOT
***Available for ODDSRATIOPLOT and RELRISKPLOT

You can specify one of the following confidence limit types for the odds ratio plot: exact (CL=EXACT), likelihood ratio (CL=LR), exact mid-$p$ (CL=MIDP), score (CL=SCORE), Wald (CL=WALD), or Wald modified (CL=WALDMODIFIED). By default, the odds ratio plot displays Wald confidence limits. For more information, see the descriptions of the CL= plot-option and the OR(CL=) option.

To display exact confidence limits in the odds ratio plot, you must also request their computation by specifying the OR option in the EXACT statement.

When CL=WALD or CL=EXACT, the odds ratio plot displays the common odds ratio by default when it is available. To compute the common odds ratio along with Wald confidence limits, specify the CMH option in the TABLES statement. To compute the common odds ratio along with exact confidence limits, specify the COMOR option in the EXACT statement. To suppress display of the common odds ratio, specify COMMON=NO.

RELRISKPLOT < (plot-options)>
requests a plot of relative risks along with confidence limits. Relative risk plots are available for multiway $2 \times 2$ tables and display the relative risk (with confidence limits) for each $2 \times 2$ table (stratum). To produce a relative risk plot, you must also specify the MEASURES or RELRISK option in the TABLES statement to compute relative risks.
Table 5.16 lists the plot-options that are available for relative risk plots. For descriptions of the plot-options, see the subsection “Plot Options” in this section.

You can specify one of the following confidence limit types for the relative risk plot: exact (CL=EXACT), likelihood ratio (CL=LR), score (CL=SCORE), Wald (CL=WALD), or Wald modified (CL=WALDMODIFIED). By default, the relative risk plot displays Wald confidence limits. For more information, see the descriptions of the CL= plot-option and the RELRISK(CL=) option.

To display exact confidence limits in the relative risk plot, you must also request their computation by specifying the RELRISK option in the EXACT statement. The risk column that you specify for the confidence limits must match the risk column that you specify for the plot.

The relative risk plot displays the common relative risk by default when you specify CL=WALD and the CMH option in the TABLES statement. To suppress display of the common relative risk, specify COMMON=NO.

In addition to the plot-options in Table 5.16, you can specify the following plot-option:

**FOOTNOTE=NO**

suppresses the footnote that identifies the column for which the relative risk is computed.

**RISKDIFFPLOT < (plot-options) >**

requests a plot of risk (proportion) differences along with confidence limits for multiway 2 × 2 tables. The risk difference plot displays the risk difference (with confidence limits) for each 2 × 2 table (stratum). Optionally, the plot also displays the common risk difference.

To produce a risk difference plot, you must also specify the RISKDIFF option in the TABLES statement to compute risk differences.

Table 5.16 lists the plot-options that are available for risk difference plots. For descriptions of the plot-options, see the subsection “Plot Options” in this section.

You can specify the confidence limit type for the stratum risk differences by using the CL= plot-option. You can specify one of the following confidence limit types: Agresti-Caffo (CL=AC), exact (CL=EXACT), Hauck-Anderson (CL=HA), Miettinen-Nurminen (score) (CL=MN), Newcombe (CL=NEWCOMBE), and Wald (CL=WALD). By default, the plot displays Wald confidence limits for the stratum risk differences. For more information, see the descriptions of the CL= plot-option and the RISKDIFF(CL=) option.

To display exact confidence limits in the risk difference plot, you must also request their computation by specifying the RISKDIFF option in the EXACT statement. The risk column that you specify for the confidence limits must match the risk column that you specify for the plot.

By default, the risk difference plot displays the common risk difference when you specify the RISKDIFF(COMMON) or COMMONRISKDIFF option unless you specify the CL=EXACT plot-option. To suppress display of the common risk difference, specify COMMON=NO.

In addition to the plot-options in Table 5.16, you can specify the following plot-options:

**CLNOTE=NO**

suppresses the note that identifies the confidence limit type.
COMMON=type
specifies the type of confidence limits to display for the common risk difference.

The default common confidence limit type depends on the stratum confidence limit type. By default, COMMON=NEWCOMBE if CL=NEWCOMBE, and COMMON=SCORE if CL=SCORE. Otherwise, COMMON=MH by default.

You can specify one of the following types:

K
KLINGENBERG
displays Klingenberg confidence limits. For more information, see the COMMONRISKDIFF(CL=K) option and the section “Klingenberg Confidence Limits” on page 238.

MH
displays Mantel-Haenszel confidence limits. For more information, see the COMMONRISKDIFF(CL=MH) option and the section “Mantel-Haenszel Confidence Limits and Test” on page 237.

MR
MINRISK
displays minimum risk confidence limits. For more information, see the COMMONRISKDIFF(CL=MR) option and the section “Minimum Risk Confidence Limits and Test” on page 238.

NEWCOMBE
displays stratified Newcombe confidence limits that use Mantel-Haenszel weights to combine the stratum components. For more information, see the COMMONRISKDIFF(CL=NEWCOMBE) option and the section “Stratified Newcombe Confidence Limits” on page 240.

NEWCOMBEMR
displays stratified Newcombe confidence limits that use minimum risk weights to combine the stratum components. For more information, see the COMMONRISKDIFF(CL=NEWCOMBEMR) option and the section “Stratified Newcombe Confidence Limits” on page 240.

NONE
suppresses the common risk difference in the risk difference plot.

SCORE
displays summary score confidence limits. For more information, see the COMMONRISKDIFF(CL=SCORE) option and the section “Summary Score Confidence Limits” on page 240.

FOOTNOTE=NO
suppresses the footnote that identifies the column for which the risk difference is computed.
WTKAPPAPLOT < (plot-options)>
requests a plot of weighted kappa coefficients along with confidence limits. Weighted kappa plots are available for multiway square tables and display the weighted kappa coefficient (with confidence limits) for each two-way table (stratum). Weighted kappa plots also display the overall weighted kappa coefficient unless you specify the COMMON=NO plot-option.

To produce a weighted kappa plot, you must specify the AGREE option in the TABLES statement to compute weighted kappa coefficients, and the table dimension must be greater than 1.

Table 5.15 lists the plot-options that are available for weighted kappa plots. For descriptions of the plot-options, see the subsection “Plot Options” in this section.

Global Plot Options

A global-plot-option applies to all plots for which the option is available unless it is altered by an individual plot-option. You can specify global-plot-options in parentheses after the PLOTS option. For example:

```
plots(order=ascending stats)=(riskdiffplot oddsratioplot)
plots(only)=freqplot
```

The following plot-options are available as global-plot-options: CLDISPLAY=, COLUMN=, COMMON=, EXACT, LOGBASE=, NPANELPOS=, ORDER=, ORIENT=, RANGE=, SCALE=, STATS, and TYPE=. For descriptions of these plot-options, see the subsection “Plot Options” in this section.

In addition to these plot-options, you can specify the following global-plot-option:

**ONLY**

suppresses the default plots and requests only the plots that are specified as plot-requests.

Plot Options

You can specify the following plot-options in parentheses after a plot-request:

**CL=type**

specifies the type of confidence limits to display. You can specify the CL= plot-option when you specify any of the following plot-requests: ODDSRATIOPLOT, RELRISKPLOT, and RISKDIFFPLOT.

For odds ratio plots (ODDSRATIOPLOT), the available confidence limit types include the following: exact (CL=EXACT), likelihood ratio (CL=LR), exact mid-$p$ (CL=MIDP), score (CL=SCORE), Wald (CL=WALD), and Wald modified (CL=WALDMODIFIED). For more information, see the description of the OR(CL=) option and the section “Confidence Limits for the Odds Ratio” on page 242. By default, CL=WALD. When you specify CL=EXACT to display exact confidence limits, you must also request computation of exact confidence limits by specifying the OR option in the EXACT statement.

For relative risk plots (RELRISKPLOT), the available confidence limit types include the following: exact (CL=EXACT), likelihood ratio (CL=LR), score (CL=SCORE), Wald (CL=WALD),
and Wald modified (CL=WALDMODIFIED). For more information, see the description of the
RELRISK(CL=) option and the section “Confidence Limits for the Relative Risk” on page 245.
By default, CL=WALD. When you specify CL=EXACT to display exact confidence limits, you
must also request computation of exact confidence limits by specifying the RELRISK option in
the EXACT statement.

For risk difference plots (RISKDIFFPLOT), the available confidence limit types include the
following: Agresti-Caffo (CL=AC), exact (CL=EXACT), Hauck-Anderson (CL=HA), Miettinen-
Nurminen (score) (CL=MN), Newcombe (CL=NEWCOMBE), and Wald (CL=WALD). For
more information, see the description of the RISKDIFF(CL=) option and the section “Confidence
Limits for the Risk Difference” on page 228. By default, CL=WALD. When you specify
CL=EXACT to display exact confidence limits in the plot, you must also request computation of
exact confidence limits by specifying the RISKDIFF option in the EXACT statement.

**CLDISPLAY=BAR < width> | LINE | LINEARROW | SERIF | SERIFARROW**

controls the appearance of the confidence limit error bars. You can specify the CLDISPLAY=
plot-option when you specify the following plot-requests: KAPPAPLOT, ODDSRATIOPLOT,
RELRISKPLOT, RISKDIFFPLOT, and WTKAPPAPLOT.

The default is CLDISPLAY=SERIF, which displays the confidence limits as lines with serifs.
CLDISPLAY=LINE displays the confidence limits as plain lines without serifs. The CLDIS-
PLAY=SERIFARROW and CLDISPLAY=LINEARROW plot-options display arrowheads on
any error bars that are clipped by the RANGE= plot-option; if an entire error bar is cut from the
plot, the plot displays an arrowhead that points toward the statistic.

CLDISPLAY=BAR displays the confidence limits as bars. By default, the width of the bars
equals the size of the marker for the estimate. You can control the width of the bars and the
size of the marker by specifying the value of width as a percentage of the distance between bars,
0 < width ≤ 1. The bar might disappear when the value of width is very small.

**COLUMN=1 | 2**

specifies the table column to use to compute the risks (proportion) for the relative risk plot
(RELRISKPLOT) and the risk difference plot (RISKDIFFPLOT). If you specify COLUMN=1,
the plot displays the column 1 relative risks or the column 1 risk differences. Similarly, if you
specify COLUMN=2, the plot displays the column 2 relative risks or risk differences.

For relative risk plots, the default is COLUMN=1. For risk difference plots, the default is
COLUMN=1 if you request computation of both column 1 and column 2 risk differences by
specifying the RISKDIFF option. If you request computation of only the column 1 (or column
2) risk differences by specifying the RISKDIFF(COLUMN=1) (or RISKDIFF(COLUMN=2))
option, by default the risk difference plot displays the risk differences for the column that you
specify.

**COMMON=NO | YES**

controls the display of the common (overall) statistic in plots that display stratum (two-way table)
statistics for multiway tables. You can specify the COMMON= plot-option when you specify the
following plot-requests: KAPPAPLOT, ODDSRATIOPLOT, RELRISKPLOT, RISKDIFFPLOT,
and WTKAPPAPLOT.

COMMON=NO suppresses display of the common statistic and its confidence limits. By default,
COMMON=YES, which displays the common statistic and its confidence limits when these
values are available. For more information, see the descriptions of the plot-requests.
**EXACT**
requests display of exact confidence limits instead of asymptotic confidence limits. You can specify the EXACT plot-option when you specify the following plot-requests: ODDSRATIOPLOT, RELRISKPLOT, and RISKDIFFPLOT. The EXACT plot-option is equivalent to the CL=EXACT plot-option.

When you specify the EXACT plot-option, you must also request computation of exact confidence limits by specifying the appropriate statistic-option in the EXACT statement.

**GROUPBY=COLUMN | ROW**
specifies the primary grouping for two-way frequency plots, which you can request by specifying the FREQPLOT plot-request. The default is GROUPBY=COLUMN, which groups graph cells first by column variable and displays row variable levels within column variable levels. You can specify GROUPBY=ROW to group first by row variable. In two-way and multiway table requests, the column variable is the last variable specified and forms the columns of the crosstabulation table. The row variable is the next-to-last variable specified and forms the rows of the table.

By default for a bar chart that is displayed in the TWOWAY=STACKED layout, bars correspond to the column variable levels, and row levels are displayed (stacked) within each column bar. By default for a bar chart that is displayed in the TWOWAY=CLUSTER layout, bars are first grouped by column variable levels, and row levels are displayed as adjacent bars within each column-level group. You can reverse the default row and column variable grouping by specifying GROUPBY=ROW.

**LOGBASE=2 | E | 10**
applies to the odds ratio plot (ODDSRATIOPLOT) and the relative risk plot (RELRISKPLOT). This plot-option displays the odds ratio or relative risk axis on the log scale that you specify.

**LEGEND=NO | YES**
applies to the agreement plot (AGREEPLOT). LEGEND=NO suppresses the legend that identifies the areas of exact and partial agreement. The default is LEGEND=YES.

**NOSTAT**
applies to the deviation plot (DEVIATIONPLOT). NOSTAT suppresses the chi-square $p$-value that deviation plot displays by default.

**NPANELPOS=n**
divides the plot into multiple panels that display at most $|n|$ statistics or sections.

If $n$ is positive, the number of statistics or sections per panel is balanced; if $n$ is negative, the number of statistics per panel is not balanced. For example, suppose you want to display 21 odds ratios. NPANELPOS=20 displays two panels, the first with 11 odds ratios and the second with 10 odds ratios; NPANELPOS=−20 displays 20 odds ratios in the first panel but only 1 odds ratio in the second panel. This plot-option is available for all plots except mosaic plots and one-way weighted frequency plots.

For two-way frequency plots (FREQPLOT), NPANELPOS=n requests that panels display at most $|n|$ sections, where sections correspond to row or column variable levels, depending on the type of plot and the grouping. By default, $n=4$ and each panel includes at most four sections. This plot-option applies to two-way plots that are displayed in the TWOWAY=GROUPVERTICAL or TWOWAY=GROUPHORIZONTAL layout. The NPANELPOS= plot-option does not apply to
the TWOWAY=CLUSTER and TWOWAY=STACKED layouts, which are always displayed in a single panel.

For plots that display statistics along with confidence limits, NPANELPOS=n requests that panels display at most $|n|$ statistics. By default, $n=0$ and all statistics are displayed in a single panel. This plot-option applies to the following plots: KAPPAPLOT, ODDSRA TIOPLOT, RELRISKPLOT, RISKDIFFPLOT, and WTKAPPAPLOT.

ORDER=ASCENDING | DESCENDING displays the two-way table (strata) statistics in order of the statistic value. You can specify the ORDER= plot-option when you specify the following plot-requests: KAPPAPLOT, ODDSRA TIOPLOT, RELRISKPLOT, RISKDIFFPLOT, and WTKAPPAPLOT.

If you specify ORDER=ASCENDING or ORDER=DESCENDING, the plot displays the statistics in ascending or descending order, respectively. By default, the order of the statistics in the plot matches the order that the two-way table strata appear in the multiway table display.

ORIENT=HORIZONTAL | VERTICAL controls the orientation of the plot. You can specify the ORIENT= plot-option when you specify the following plot-requests: CUMFREQPLOT, DEVIATIONPLOT, and FREQPLOT.

ORIENT=HORIZONTAL places the variable levels on the Y axis and the frequencies, percentages, or statistic values on the X axis. ORIENT=VERTICAL places the variable levels on the X axis. The default orientation is ORIENT=VERTICAL for bar charts (TYPE=BARCHART) and ORIENT=HORIZONTAL for dot plots (TYPE=DOTPLOT).

PARTIAL=NO | YES controls the display of partial agreement in the agreement plot (AGREEPLOT). PARTIAL=NO suppresses the display of partial agreement. When you specify PARTIAL=NO, the agreement plot displays only exact agreement. Exact agreement includes the diagonal cells of the square table, where the row and column variable levels are the same. Partial agreement includes the adjacent off-diagonal table cells, where the row and column values are within one level of exact agreement. The default is PARTIAL=YES.

RANGE=(< min >, < max > ) | CLIP specifies the range of values to display. You can specify the RANGE= plot-option when you specify the following plot-requests: KAPPAPLOT, ODDSRA TIOPLOT, RELRISKPLOT, RISKDIFFPLOT, and WTKAPPAPLOT.

If you specify RANGE=CLIP, the confidence limits are clipped and the display range is determined by the minimum and maximum values of the statistics. By default, the display range includes all confidence limits.

SCALE=FREQ | GROUPPERCENT | LOG | PERCENT | SQRT specifies the scale of the frequencies to display. This plot-option is available for frequency plots (FREQPLOT) and cumulative frequency plots (CUMFREQPLOT).

The default is SCALE=FREQ, which displays unscaled frequencies. SCALE=PERCENT displays percentages (relative frequencies) of the total frequency. SCALE=LOG displays log (base 10) frequencies. SCALE=SQRT displays square roots of the frequencies, producing a plot known as a rootogram.
SCALE=GROUPPERCENT is available for two-way frequency plots. This option displays the row or column percentages instead of the overall percentages (of the table frequency). By default (or when you specify the GROUPBY=COLUMN plot-option), SCALE=GROUPPERCENT displays the column percentages. If you specify the GROUPBY=ROW plot-option, the primary grouping of graph cells is by row variable level and the plot displays row percentages. For more information, see the description of the GROUPBY= plot-option.

SHOWSCALE=NO | YES

controls the display of the cumulative frequency scale on the right side of the agreement plot (AGREEPLOT). SHOWSCALE=NO suppresses the display of the scale. The default is SHOWSCALE=YES.

STATS

displays statistic values in the plot. For the following plot-requests, the STATS plot-option displays the statistics and their confidence limits on the right side of the plot: KAPPAPLOT, ODDSRATIOPILOT, RELRISKPILOT, RISKDIFFPILOT, and WTAKAPPAPLOT.

For the agreement plot (AGREEPLOT), the STATS plot-option displays the values of the kappa statistic, the weighted kappa statistic, the $B_n$ measure (Bangdiwala and Bryan 1987), and the sample size. PROC FREQUENCIES stores these statistics in an ODS table named BnMeasure, which is not displayed. For more information, see the section “ODS Table Names” on page 288.

If you do not request the STATS plot-option, these plots do not display the statistic values.

TWOWAY=CLUSTER | GROUPOHORIZONTAL | GROUPVERTICAL | STACKED

specifies the layout for two-way frequency plots.

All TWOWAY= layouts are available for bar charts (TYPE=BARCHART). All TWOWAY= layouts except TWOWAY=CLUSTER are available for dot plots (TYPE=DOTPLOT). The ORIENT= and GROUPBY= plot-options are available for all TWOWAY= layouts.

The default two-way layout is TWOWAY=GROUPVERTICAL, which produces a grouped plot that has a vertical common baseline. By default for bar charts (TYPE=BARCHART, ORIENT=VERTICAL), the X axis displays column variable levels, and the Y axis displays frequencies. The plot includes a vertical (Y-axis) block for each row variable level. The relative positions of the graph cells in this plot layout are the same as the relative positions of the table cells in the crosstabulation table. You can reverse the default row and column grouping by specifying the GROUPBY=ROW plot-option.

The TWOWAY=GROUPOHORIZONTAL layout produces a grouped plot that has a horizontal common baseline. By default (GROUPBY=COLUMN), the plot displays a block on the X axis for each column variable level. Within each column-level block, the plot displays row variable levels.

The TWOWAY=STACKED layout produces stacked displays of frequencies. By default (GROUPBY=COLUMN) in a stacked bar chart, the bars correspond to column variable levels, and row levels are stacked within each column level. By default in a stacked dot plot, the dotted lines correspond to column levels, and cell frequencies are plotted as data dots on the corresponding column line. The dot color identifies the row level.

The TWOWAY=CLUSTER layout, which is available only for bar charts, displays groups of adjacent bars. By default, the primary grouping is by column variable level, and row levels are displayed within each column level.
You can reverse the default row and column grouping in any layout by specifying the GROUPBY=ROW plot-option. The default is GROUPBY=COLUMN, which groups first by column variable.

**TYPE=BARCHART | DOTPLOT**
specifies the plot type (format) of the frequency (FREQPLOT), cumulative frequency (CUMFREQPLOT), and deviation plots (DEVIATIONPLOT). TYPE=BARCHART produces a bar chart and TYPE=DOTPLOT produces a dot plot. The default is TYPE=BARCHART.

**PRINTKWTS**
displays the agreement weights that PROC FREQTAB uses to compute the weighted kappa coefficient. Agreement weights reflect the relative agreement between pairs of variable levels. By default, PROC FREQTAB uses the Cicchetti-Allison form of agreement weights. If you specify the AGREE(WT=FC) option, the procedure uses the Fleiss-Cohen form of agreement weights. For more information, see the section “Weighted Kappa Coefficient” on page 257.

This option has no effect unless you also specify the AGREE option to compute the weighted kappa coefficient. The PRINTKWTS option is equivalent to the AGREE(PRINTKWTS) option.

**RELRISK < (relrisk-options) >**
requests relative risk measures and their confidence limits for 2×2 tables. These measures include the odds ratio, the column 1 relative risk, and the column 2 relative risk. For more information, see the section “Odds Ratio and Relative Risks for 2×2 Tables” on page 241. By default, PROC FREQTAB displays the relative risk measures and their asymptotic Wald confidence limits in the “Odds Ratio and Relative Risks” table. You can also obtain this table by specifying the MEASURES option, which produces other measures of association in addition to the relative risks.

You can specify **relrisk-options** in parentheses after the RELRISK option to request tests and additional confidence limits for the column 1 or column 2 relative risk. Table 5.17 summarizes the relrisk-options.

When you request tests or additional confidence limit types for the relative risk, PROC FREQTAB does not display the “Odds Ratio and Relative Risks” table unless you also specify the PRINTALL relrisk-option.

**Table 5.17  RELRISK (Relative Risk) Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COLUMN=1</td>
<td>2</td>
</tr>
<tr>
<td>PRINTALL</td>
<td>Displays “Odds Ratio and Relative Risks” table</td>
</tr>
<tr>
<td>Request Confidence Limits</td>
<td></td>
</tr>
<tr>
<td>CL=EXACT</td>
<td>Displays exact confidence limits</td>
</tr>
<tr>
<td>CL=LR</td>
<td>Requests likelihood ratio confidence limits</td>
</tr>
<tr>
<td>CL=SCORE</td>
<td>Requests score confidence limits</td>
</tr>
<tr>
<td>CL=WALD</td>
<td>Requests Wald confidence limits</td>
</tr>
<tr>
<td>CL=WALDMODIFIED</td>
<td>Requests Wald modified confidence limits</td>
</tr>
<tr>
<td>Request Tests</td>
<td></td>
</tr>
<tr>
<td>EQUAL(NULL=)</td>
<td>Requests an equality test</td>
</tr>
<tr>
<td>EQUIV</td>
<td>EQUIVALENCE</td>
</tr>
</tbody>
</table>
**Table 5.17 continued**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MARGIN=</td>
<td>Specifies the test margin</td>
</tr>
<tr>
<td>METHOD=</td>
<td>Specifies the test method</td>
</tr>
<tr>
<td>NONINF</td>
<td>NONINFERIORITY</td>
</tr>
<tr>
<td>SUP</td>
<td>SUPERIORITY</td>
</tr>
</tbody>
</table>

You can specify the following `relrisk-options`:

**CL=type | (types)**

specifies confidence limit types for the relative risk. You can specify one or more types of confidence limits. When you specify only one type, you can omit the parentheses around the request. When you specify the `CL= relrisk-option`, PROC FREQTAB displays the confidence limits in the “Confidence Limits for the Relative Risk” table.

The `ALPHA= option determines the level of the confidence limits that the CL= relrisk-option provides. By default, ALPHA=0.05, which produces 95% confidence limits for the relative risk.

You can specify the following types:

**EXACT**

 displays exact unconditional confidence limits for the relative risk in the “Confidence Limits for the Relative Risk” table. (By default, PROC FREQTAB displays the exact confidence limits in a separate table.) You must also request computation of the exact confidence limits by specifying the RELRISK option in the EXACT statement. For more information, see the subsection “Exact Unconditional Confidence Limits” in the section “Confidence Limits for the Relative Risk” on page 245.

**LR**

**LIKELIHOOD RATIO**

requests likelihood ratio confidence limits for the relative risk. For more information, see the subsection “Likelihood Ratio Confidence Limits” in the section “Confidence Limits for the Relative Risk” on page 245.

**SCORE <(CORRECT=NO)>**

requests score confidence limits for the relative risk. For more information, see the subsection “Score Confidence Limits” in the section “Confidence Limits for the Relative Risk” on page 245. If you specify CORRECT=NO, PROC FREQTAB provides the uncorrected form of the confidence limits.

**WALD**

requests asymptotic Wald confidence limits, which are based on a log transformation of the relative risk. For more information, see the subsection “Wald Confidence Limits” in the section “Confidence Limits for the Relative Risk” on page 245.

**WALDMODIFIED**

requests Wald modified confidence limits for the odds ratio. For more information, see the subsection “Wald Modified Confidence Limits” in the section “Confidence Limits for the Relative Risk” on page 245.
COLUMN=1 | 2
specifies the table column for which to compute the relative risk confidence limits (which you request by specifying the **CL= relrisk-option**) and the relative risk tests (**EQUAL, EQUIV, NONINF, and SUP**). By default, COLUMN=1.

This option has no effect on the “Odds Ratio and Relative Risks” table, which displays both column 1 and column 2 relative risks.

**EQUAL < (NULL=value )>**
requests an equality test for the relative risk. For more information, see the subsection “Equality Test” in the section “Relative Risk Tests” on page 248. You can specify the test in the **METHOD= relrisk-option**, and you can specify the null hypothesis **value** of the relative risk in the **NULL=** option. The null **value** must be a positive number. By default, METHOD=WALD and NULL=1.

**EQUIV**

**EQUIVALENCE**
requests an equivalence test for the relative risk. For more information, see the subsection “Equivalence Test” in the section “Relative Risk Tests” on page 248. You can specify the test method in the **METHOD= relrisk-option**, and you can specify the test margins in the **MARGIN= relrisk-option**. By default, METHOD=WALD and MARGIN=(0.8,1.25).

**MARGIN=value | (lower, upper)**
specifies the margin for the noninferiority, superiority, and equivalence tests, which you request by specifying the **NONINF, SUP, and EQUIV relrisk-options**, respectively. By default, MARGIN=0.8 for noninferiority tests, MARGIN=1.25 for superiority tests, and MARGIN=(0.8,1.25) for equivalence tests.

For noninferiority and superiority tests, specify a single **value** in the **MARGIN=** option. The **value** must be a positive number. For a noninferiority test, the **value** should be less than 1; for a superiority test, the **value** should be greater than 1.

For an equivalence test, you can specify a single MARGIN= **value**, or you can specify both **lower** and **upper** values. All values must be positive numbers. If you specify a single **value**, PROC FREQTAB uses **value** as the lower margin and the inverse of **value** as the upper margin. If you specify both **lower** and **upper** values, the value of **lower** must be less than the value of **upper**.

**METHOD=method**
specifies the method to be used for the equality, equivalence, noninferiority, and superiority tests, which you request by specifying the **EQUAL, EQUIV, NONINF, and SUP relrisk-options**, respectively. By default, METHOD=WALD.

You can specify one of the following **methods**:

**FM**

**SCORE**
requests Farrington-Manning (score) tests for the equality, equivalence, noninferiority, and superiority analyses of the relative risk. For more information, see the subsection “Farrington-Manning (Score) Test” in the section “Relative Risk Tests” on page 248.
LR
LIKELIHOODRATIO
requests likelihood ratio tests for the equality, equivalence, noninferiority, and superiority analyses of the relative risk. For more information, see the subsection “Likelihood Ratio Test” in the section “Relative Risk Tests” on page 248.

WALD
requests Wald tests for the equality, equivalence, noninferiority, and superiority analyses of the relative risk. For more information, see the subsection “Wald Test” in the section “Relative Risk Tests” on page 248.

WALDMODIFIED
requests Wald modified tests for the equality, equivalence, noninferiority, and superiority analyses of the relative risk. For more information, see the subsection “Wald Modified Test” in the section “Relative Risk Tests” on page 248.

NONINF
NONINFERIORITY
requests a noninferiority test for the relative risk. For more information, see the subsection “Noninferiority Test” in the section “Relative Risk Tests” on page 248. You can specify the test method in the METHOD= relrisk-option, and you can specify the margin in the MARGIN= relrisk-option. By default, METHOD=WALD and MARGIN=0.8.

PRINTALL
displays the “Odds Ratio and Relative Risks” table when you request tests or additional confidence limits by specifying relrisk-options. By default, PROC FREQTAB does not display this table when you request tests or additional confidence limits for the relative risk.

SUP
SUPERIORITY
requests a superiority test for the relative risk. For more information, see the subsection “Superiority Test” in the section “Relative Risk Tests” on page 248. You can specify the test method in the METHOD= relrisk-option, and you can specify the margin in the MARGIN= relrisk-option. By default, METHOD=WALD and MARGIN=1.25.

RISKDIFF < (riskdiff-options) >
requests risks (binomial proportions) and risk differences for $2 \times 2$ tables. By default, this option provides the row 1 risk, row 2 risk, total (overall) risk, and risk difference (row 1 – row 2), together with their asymptotic standard errors and Wald confidence limits; by default, this option also provides exact (Clopper-Pearson) confidence limits for the row 1, row 2, and total risks. You can request exact unconditional confidence limits for the risk difference by specifying the RISKDIFF option in the EXACT statement. PROC FREQTAB displays these results in the column 1 and column 2 “Risk Estimates” tables (which you can suppress by specifying the NORISKS riskdiff-option).

You can specify riskdiff-options in parentheses after the RISKDIFF option to request tests and additional confidence limits for the risk difference, in addition to estimates of the common risk difference for multiway $2 \times 2$ tables. Table 5.18 summarizes the riskdiff-options.

The CL= riskdiff-option requests confidence limits for the risk difference. Available confidence limit types include Agresti-Caffo, exact unconditional, Hauck-Anderson, Miettinen-Nurminen (score),
Newcombe, and Wald. Continuity-corrected Newcombe and Wald confidence limits are also available. You can request more than one type of confidence limits in the same analysis. PROC FREQTAB displays the confidence limits in the “Confidence Limits for the Risk Difference” table.

The CL=EXACT riskdiff-option displays exact unconditional confidence limits in the “Confidence Limits for the Risk Difference” table. When you specify CL=EXACT, you must also request computation of the exact confidence limits by specifying the RISKDIFF option in the EXACT statement.

The EQUAL, EQUIV, NONINF, and SUP riskdiff-options request tests of equality, equivalence, noninferiority, and superiority, respectively, for the risk difference. Available test methods include Farrington-Manning (score), Hauck-Anderson, and Wald. Newcombe (hybrid-score) confidence limits are available for the equivalence, noninferiority, and superiority analyses.

As part of the noninferiority, superiority, and equivalence analyses, PROC FREQTAB provides null-based equivalence limits that have a confidence coefficient of $100(1 - 2\alpha)\%$ (Schuirmann 1999). The ALPHA= option determines the confidence level; by default, ALPHA=0.05, which produces 90% equivalence limits for these analyses. For more information, see the sections “Noninferiority Tests” on page 232 and “Equivalence Test” on page 235.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COLUMN=1</td>
<td>2</td>
</tr>
<tr>
<td>COMMON</td>
<td>Requests common risk difference</td>
</tr>
<tr>
<td>CORRECT</td>
<td>Requests continuity correction</td>
</tr>
<tr>
<td>NORISKS</td>
<td>Suppresses default risk tables</td>
</tr>
</tbody>
</table>

**Request Confidence Limits**

- CL=AC: Requests Agresti-Caffo confidence limits
- CL=EXACT: Displays exact confidence limits
- CL=HA: Requests Hauck-Anderson confidence limits
- CL=MN | SCORE: Requests Miettinen-Nurminen confidence limits
- CL=NEWCOMBE: Requests Newcombe confidence limits
- CL=WALD: Requests Wald confidence limits

**Request Tests**

- EQUAL(NULL=): Requests an equality test
- EQUIV | EQUIVALENCE: Requests an equivalence test
- MARGIN=: Specifies the test margin
- METHOD=: Specifies the test method
- NONINF | NONINFERIORITY: Requests a noninferiority test
- SUP | SUPERIORITY: Requests a superiority test
- VAR=SAMPLE | NULL: Specifies the test variance

You can specify the following riskdiff-options:

**CL=type | (types)**

requests confidence limits for the risk difference. You can specify one or more types of confidence limits. When you specify only one type, you can omit the parentheses around the request. PROC
FREQTAB displays the confidence limits in the “Confidence Limits for the Risk Difference” table.

The ALPHA= option determines the level of the confidence limits. By default, ALPHA=0.05, which produces 95% confidence limits for the risk difference.

You can specify the CL= riskdiff-option with or without requests for risk difference tests. The confidence limits that CL= produces do not depend on the tests that you request and do not use the value of the test margin (which you can specify in the MARGIN= riskdiff-option).

You can specify the following types:

**AC**

**AGRESTICAFFO**
requests Agresti-Caffo confidence limits for the risk difference. For more information, see the subsection “Agresti-Caffo Confidence Limits” in the section “Confidence Limits for the Risk Difference” on page 228.

**EXACT**
displays exact unconditional confidence limits for the risk difference in the “Confidence Limits for the Risk Difference” table. You must also request computation of the exact confidence limits by specifying the RISKDIFF option in the EXACT statement.

By default, PROC FREQTAB computes the exact confidence limits by inverting two separate one-sided exact tests that are based on the score statistic. For more information, see the RISKDIFF option in the EXACT statement and the subsection “Exact Unconditional Confidence Limits” in the section “Confidence Limits for the Risk Difference” on page 228.

By default, PROC FREQTAB also displays these exact confidence limits in the “Risk Estimates” table. You can suppress this table by specifying the NORISKS riskdiff-option.

**HA**
requests Hauck-Anderson confidence limits for the risk difference. For more information, see the subsection “Hauck-Anderson Confidence Limits” in the section “Confidence Limits for the Risk Difference” on page 228.

**MN < (CORRECT=NO | MEE) >**

**SCORE < (CORRECT=NO | MEE) >**
requests Miettinen-Nurminen (score) confidence limits for the risk difference. For more information, see the subsection “Miettinen-Nurminen (Score) Confidence Limits” in the section “Confidence Limits for the Risk Difference” on page 228. By default, the Miettinen-Nurminen confidence limits include a bias correction factor (Miettinen and Nurminen 1985; Newcombe and Nurminen 2011). If you specify CL=MN(CORRECT=NO), PROC FREQTAB provides the uncorrected form of the confidence limits (Mee 1984).

**NEWCOMBE < (CORRECT) >**
requests Newcombe hybrid-score confidence limits for the risk difference. If you specify CL=NEWCOMBE(CORRECT) or the CORRECT riskdiff-option, the Newcombe confidence limits include a continuity correction. For more information, see the subsection “Newcombe Confidence Limits” in the section “Confidence Limits for the Risk Difference” on page 228.
WALD < (CORRECT) >  
requests Wald confidence limits for the risk difference. If you specify CL=WALD(CORRECT) or the CORRECT riskdiff-option, the Wald confidence limits include a continuity correction. For more information, see the subsection “Wald Confidence Limits” in the section “Confidence Limits for the Risk Difference” on page 228.

COLUMN=1 | 2 | BOTH  
specifies the table column for which to compute the risk difference tests (EQUAL, EQUIV, NONINF, and SUP) and the risk difference confidence limits (which you request by specifying the CL= riskdiff-option). By default, COLUMN=1.

This option has no effect on the “Risk Estimates” table, which is produced for both column 1 and column 2. You can suppress the “Risk Estimates” table by specifying the NORISKS riskdiff-option.

COMMON  
requests estimates of the common (overall) risk difference for multiway $2 \times 2$ tables. This option provides Mantel-Haenszel and summary score estimates for the common risk difference, together with their confidence limits. If you specify the RISKDIFF(CL=NEWCOMBE) option, the RISKDIFF(COMMON) option also provides Newcombe confidence limits for the common risk difference. For more information, see the section “Common Risk Difference” on page 237.

You can use the COMMONRISKDIFF option to request additional confidence limits and tests for the common risk difference.

If you do not specify the COLUMN= riskdiff-option, PROC FREQTAB provides the common risk difference for column 1 by default. If you specify COLUMN=2, PROC FREQTAB provides the common risk difference for column 2. COLUMN=BOTH does not apply to the common risk difference.

CORRECT  
includes a continuity correction in the Wald confidence limits, Wald tests, and Newcombe confidence limits. For more information, see the section “Risks and Risk Differences” on page 226.

EQUAL < (NULL=value )>  
requests an equality test for the risk difference. For more information, see the section “Equality Tests” on page 232. You can specify the test method in the METHOD= riskdiff-option, and you can specify the null hypothesis value of the risk difference in the NULL= option. By default, METHOD=WALD and NULL=0. You can specify the null value in proportion form as a number between −1 and 1, or you can specify the null value in percentage form as a number between −100 and 100. When the value is between −100 and −1 or between 1 and 100, PROC FREQTAB converts the number to a proportion. PROC FREQTAB treats the values −1 and 1 as percentages.

EQUIV  
EQUIVALENCE  
requests an equivalence test for the risk difference. For more information, see the section “Equivalence Test” on page 235. You can specify the test method in the METHOD= riskdiff-option, and you can specify the margins in the MARGIN= riskdiff-option. By default, METHOD=WALD and MARGIN=0.2.
**MARGIN=**

Specifies the margin for the noninferiority, superiority, and equivalence tests, which you request by specifying the **NONINF**, **SUP**, and **EQUIV** *riskdiff-options*, respectively. By default, **MARGIN=0.2**.

For noninferiority and superiority tests, specify a single **value** in the **MARGIN=** option. The **value** must be a positive number. You can specify **value** as a number between 0 and 1. Or you can specify **value** in percentage form as a number between 1 and 100, and **PROC FREQTAB** converts that number to a proportion. **PROC FREQTAB** treats the value 1 as 1%.

For an equivalence test, you can specify a single **MARGIN= value**, or you can specify both **lower** and **upper** values. If you specify a single **value**, it must be a positive number, as described previously. If you specify a single **value** for an equivalence test, **PROC FREQTAB** uses –**value** as the lower margin and **value** as the upper margin for the test. If you specify both **lower** and **upper** values for an equivalence test, you can specify them in proportion form as numbers between –1 and 1. Or you can specify them in percentage form as numbers between –100 and 100, and **PROC FREQTAB** converts the numbers to proportions. The value of **lower** must be less than the value of **upper**.

**METHOD=**

Specifies the method to be used for the equality, equivalence, noninferiority, and superiority tests, which you request by specifying the **EQUAL**, **EQUIV**, **NONINF**, and **SUP** *riskdiff-options*, respectively. By default, **METHOD=WALD**.

You can specify the following **methods**:

**FM**

**SCORE**

Requests Farrington-Manning (score) tests for the equality, equivalence, noninferiority, and superiority analyses. For more information, see the subsection “Farrington-Manning (Score) Test” in the section “Noninferiority Tests” on page 232.

**HA**

Requests Hauck-Anderson tests for the equality, equivalence, noninferiority, and superiority analyses. For more information, see the subsection “Hauck-Anderson Test” in the section “Noninferiority Tests” on page 232.

**NEWCOMBE**

Requests Newcombe (hybrid-score) confidence limits for the equivalence, noninferiority, and superiority analyses. If you specify the **CORRECT riskdiff-option**, the Newcombe confidence limits include a continuity correction. For more information, see the subsection “Newcombe Noninferiority Analysis” in the section “Noninferiority Tests” on page 232.

**WALD**

Requests Wald tests for the equality, equivalence, noninferiority, and superiority analyses. If you specify the **CORRECT riskdiff-option**, the Wald tests and confidence limits include a continuity correction. If you specify the **VAR=NULL riskdiff-option**, the tests use the null (test-based) variance instead of the sample variance. For more information, see the subsection “Wald Test” in the section “Noninferiority Tests” on page 232.
NONINF

NONINFERIORITY

requests a noninferiority test for the risk difference. For more information, see the section “Noninferiority Tests” on page 232. You can specify the test method in the METHOD= riskdiff-option, and you can specify the margin in the MARGIN= riskdiff-option. By default, METHOD=WALD and MARGIN=0.2.

NORISKS

suppresses display of the “Risk Estimates” tables, which the RISKDIFF option produces by default for column 1 and column 2. The “Risk Estimates” tables contain the risks and risk differences, together with their asymptotic standard errors, Wald confidence limits, and exact confidence limits.

SUP

SUPERIORITY

requests a superiority test for the risk difference. For more information, see the section “Superiority Test” on page 235. You can specify the test method in the METHOD= riskdiff-option, and you can specify the margin in the MARGIN= riskdiff-option. By default, METHOD=WALD and MARGIN=0.2.

VAR=NULL | SAMPLE

specifies the type of variance to use in the Wald tests of equality, equivalence, noninferiority, and superiority. If you specify VAR=SAMPLE, PROC FREQTAB uses the sample variance. If you specify VAR=NULL, PROC FREQTAB uses a test-based variance that is computed by using the null hypothesis value of the risk difference. For more information, see the sections “Equality Tests” on page 232 and “Noninferiority Tests” on page 232. The default is VAR=SAMPLE.

SCORES=type

specifies the type of row and column scores that PROC FREQTAB uses to compute the following statistics: Mantel-Haenszel chi-square, Pearson correlation, Cochran-Armitage test for trend, weighted kappa coefficient, and Cochran-Mantel-Haenszel statistics. The value of type can be one of the following:

- MODRIDIT
- RANK
- RIDIT
- TABLE

See the section “Scores” on page 201 for descriptions of these score types.

If you do not specify the SCORES= option, PROC FREQTAB uses SCORES=TABLE by default. For character variables, the row and column TABLE scores are the row and column numbers. That is, the TABLE score is 1 for row 1, 2 for row 2, and so on. For numeric variables, the row and column TABLE scores equal the variable values. For more information, see the section “Scores” on page 201. Using MODRIDIT, RANK, or RIDIT scores yields nonparametric analyses.

You can use the SCOROUT option to display the row and column scores.
SCOROUT

displays the row and column scores that PROC FREQTAB uses to compute score-based tests and statistics. You can specify the score type by using the SCORES= option. For more information, see the section “Scores” on page 201.

The scores are computed and displayed only when PROC FREQTAB computes statistics for two-way tables. You can use ODS to store the scores in an output data set. See the section “ODS Table Names” on page 288 for more information.

SENSPEC

requests estimates of sensitivity, specificity, positive predictive value, and negative predictive value for 2×2 tables. The “Sensitivity and Specificity” table provides these estimates together with their standard errors and confidence limits. For more information, see the section “Sensitivity and Specificity” on page 225.

You can specify the confidence level in the ALPHA= option. By default, ALPHA=0.05, which produces 95% confidence limits.

TOTPCT

displays the percentage of the total multiway table frequency in multiway crosstabulation tables (n-way tables where n > 2). By default, PROC FREQTAB displays the percentage of the individual two-way table (stratum) frequency but does not display the percentage of the overall total frequency. For more information, see the section “Two-Way and Multiway Tables” on page 280.

By default, tables in list format (which you can request by specifying the LIST option) display the percentage of the total multiway table frequency. The variable PERCENT in the OUT= output data set also provides the percentage of the total multiway table frequency.

TREND

requests the Cochran-Armitage test for trend. The table must be 2×C or R×2 to compute the trend test. For more information, see the section “Cochran-Armitage Test for Trend” on page 251. To request exact p-values for the trend test, specify the TREND option in the EXACT statement. See the section “Exact Statistics” on page 271 for more information.

TEST Statement

TEST test-options ;

The TEST statement requests asymptotic tests for measures of association and measures of agreement. The test-options identify which tests to compute. Table 5.19 lists the available test-options, together with their corresponding TABLES statement options. Descriptions of the test-options follow the table in alphabetical order.

For each measure of association or agreement that you request in the TEST statement, PROC FREQTAB provides an asymptotic test that the measure is 0. The procedure displays the asymptotic standard error under the null hypothesis, the test statistic, and the one-sided and two-sided p-values. PROC FREQTAB also provides confidence limits for the measure. The ALPHA= option in the TABLES statement determines the confidence level; by default, ALPHA=0.05, which provides 95% confidence limits. For more information, see the sections “Asymptotic Tests” on page 208 and “Confidence Limits” on page 208. For information about
the individual measures, see the sections “Measures of Association” on page 208 and “Tests and Measures of Agreement” on page 254.

You can also request exact tests for selected measures of association and agreement by using the EXACT statement. For more information, see the section “Exact Statistics” on page 271.

Using the TEST Statement with the TABLES Statement

You must use a TABLES statement with the TEST statement. If you use only one TABLES statement, you do not need to specify the same options in both the TABLES and TEST statements; when you specify an option in the TEST statement, PROC FREQTAB automatically invokes the corresponding TABLES statement option. However, when you use the TEST statement with multiple TABLES statements, you must specify options in the TABLES statements to request statistics; PROC FREQTAB then provides asymptotic tests for those statistics that you specify in the TEST statement.

Test Options

Table 5.19  TEST Statement Options

<table>
<thead>
<tr>
<th>Test Option</th>
<th>Asymptotic Tests</th>
<th>Required TABLES Statement Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGREE</td>
<td>Simple and weighted kappa coefficients</td>
<td>AGREE</td>
</tr>
<tr>
<td>GAMMA</td>
<td>Gamma</td>
<td>ALL or MEASURES</td>
</tr>
<tr>
<td>KAPPA</td>
<td>Simple kappa coefficient</td>
<td>AGREE</td>
</tr>
<tr>
<td>KENTB</td>
<td>TAUB</td>
<td>Kendall’s tau-(b)</td>
</tr>
<tr>
<td>MEASURES</td>
<td>Gamma, Kendall’s tau-(b), Stuart’s tau-(c), Somers’ (D(C \mid R)), Somers’ (D(R \mid C)), Pearson and Spearman correlations</td>
<td>ALL or MEASURES</td>
</tr>
<tr>
<td>PCORR</td>
<td>Pearson correlation coefficient</td>
<td>ALL or MEASURES</td>
</tr>
<tr>
<td>PLCORR</td>
<td>Polychoric correlation</td>
<td>PLCORR</td>
</tr>
<tr>
<td>SCORR</td>
<td>Spearman correlation coefficient</td>
<td>ALL or MEASURES</td>
</tr>
<tr>
<td>SMDCR</td>
<td>Somers’ (D(C \mid R))</td>
<td>ALL or MEASURES</td>
</tr>
<tr>
<td>SMDRC</td>
<td>Somers’ (D(R \mid C))</td>
<td>ALL or MEASURES</td>
</tr>
<tr>
<td>STUTC</td>
<td>TAUC</td>
<td>Stuart’s tau-(c)</td>
</tr>
<tr>
<td>WTKAPPA</td>
<td>WTKAP</td>
<td>Weighted kappa coefficient</td>
</tr>
</tbody>
</table>

You can specify the following test-options in the TEST statement.

AGREE

requests asymptotic tests for the simple kappa coefficient and the weighted kappa coefficient. For more information, see the sections “Simple Kappa Coefficient” on page 255 and “Weighted Kappa Coefficient” on page 257.

By default, these tests are based on null values of 0; you can specify nonzero null values for the simple kappa and weighted kappa tests by using the AGREE(NULLKAPPA=) and AGREE(NULLWTKAPPA=) options, respectively, in the TABLES statement.

The AGREE option in the TABLES statement provides estimates, standard errors, and confidence limits for kappa coefficients. You can request exact tests for kappa coefficients by using the EXACT statement.
Kappa coefficients are defined only for square tables, where the number of rows equals the number of columns. Kappa coefficients are not computed for tables that are not square. For $2 \times 2$ tables, the weighted kappa coefficient is identical to the simple kappa coefficient, and PROC FREQTAB presents only the simple kappa coefficient.

**GAMMA**
requests an asymptotic test for the gamma statistic. For more information, see the section “Gamma” on page 209. The MEASURES option in the TABLES statement provides the gamma statistic and its asymptotic standard error.

**KAPPA**
requests an asymptotic test for the simple kappa coefficient. For more information, see the section “Simple Kappa Coefficient” on page 255.

By default, the null value of kappa for this test is 0; you can specify a nonzero null value by using the AGREE(NULLKAPPA=) option in the TABLES statement.

The AGREE option in the TABLES statement provides the kappa statistic, its standard error, and its confidence limits. You can request an exact test for the simple kappa coefficient by specifying the KAPPA option in the EXACT statement.

Kappa coefficients are defined only for square tables, where the number of rows equals the number of columns. PROC FREQTAB does not compute kappa coefficients for tables that are not square.

**KENTB**
requests an asymptotic test for Kendall’s tau-$b$. For more information, see the section “Kendall’s Tau-$b$” on page 210.

The MEASURES option in the TABLES statement provides Kendall’s tau-$b$ and its standard error. You can request an exact test for Kendall’s tau-$b$ by specifying the KENTB option in the EXACT statement.

**MEASURES**
requests asymptotic tests for the following measures of association: gamma, Kendall’s tau-$b$, Pearson correlation coefficient, Somers’ $D(C|R)$, Somers’ $D(R|C)$, Spearman correlation coefficient, and Stuart’s tau-$c$. For more information, see the section “Measures of Association” on page 208.

The MEASURES option in the TABLES statement provides measures of association and their asymptotic standard errors. You can request exact tests for selected measures by using the EXACT statement.

**PCORR**
requests an asymptotic test for the Pearson correlation coefficient. For more information, see the section “Pearson Correlation Coefficient” on page 211.

The MEASURES option in the TABLES statement provides the Pearson correlation and its standard error. You can request an exact test for the Pearson correlation by specifying the PCORR option in the EXACT statement.

**PLCORR**
requests Wald and likelihood ratio tests for the polychoric correlation coefficient. For more information, see the section “Polychoric Correlation” on page 214.

The PLCORR option in the TABLES statement provides the polychoric correlation and its standard error.
SCORR requests an asymptotic test for the Spearman correlation coefficient. For more information, see the section “Spearman Rank Correlation Coefficient” on page 212.

The MEASURES option in the TABLES statement provides the Spearman correlation and its standard error. You can request an exact test for the Spearman correlation by specifying the SCORR option in the EXACT statement.

SMDCR requests an asymptotic test for Somers’ $D(C \mid R)$. For more information, see the section “Somers’ $D$” on page 211.

The MEASURES option in the TABLES statement provides Somers’ $D(C \mid R)$ and its standard error. You can request an exact test for Somers’ $D(C \mid R)$ by specifying the SMDCR option in the EXACT statement.

SMDRC requests an asymptotic test for Somers’ $D(R \mid C)$. For more information, see the section “Somers’ $D$” on page 211.

The MEASURES option in the TABLES statement provides Somers’ $D(R \mid C)$ and its standard error. You can request an exact test for Somers’ $D(R \mid C)$ by specifying the SMDRC option in the EXACT statement.

STUTC TAUC requests an asymptotic test for Stuart’s tau-c. For more information, see the section “Stuart’s Tau-c” on page 210.

The MEASURES option in the TABLES statement provides Stuart’s tau-c and its standard error. You can request an exact test for Stuart’s tau-c by specifying the STUTC option in the EXACT statement.

WTKAPPA WTKAP requests an asymptotic test for the weighted kappa coefficient. For more information, see the section “Weighted Kappa Coefficient” on page 257.

By default, the null value of weighted kappa for this test is 0; you can specify a nonzero null value by using the AGREE(NULLWTKAPPA=) option in the TABLES statement.

The AGREE option in the TABLES statement provides the weighted kappa coefficient, its standard error, and confidence limits. You can request an exact test for the weighted kappa by specifying the WTKAPPA option in the EXACT statement.

Kappa coefficients are defined only for square tables, where the number of rows equals the number of columns. PROC FREQTAB does not compute kappa coefficients for tables that are not square. For $2 \times 2$ tables, the weighted kappa coefficient is identical to the simple kappa coefficient, and PROC FREQTAB presents only the simple kappa coefficient.
The WEIGHT statement names a numeric variable that provides a weight for each observation in the input data table. The WEIGHT statement is most commonly used to input cell count data. See the section “Inputting Frequency Counts” on page 198 for more information. If you use a WEIGHT statement, PROC FREQTAB assumes that an observation represents \( n \) observations, where \( n \) is the value of `variable`. The value of the WEIGHT variable is not required to be an integer.

If the value of the WEIGHT variable is missing, PROC FREQTAB does not use that observation in the analysis. If the value of the WEIGHT variable is 0, PROC FREQTAB ignores the observation unless you specify the ZEROS option, which includes observations that have weights of 0. If you do not specify a WEIGHT statement, PROC FREQTAB assigns a weight of 1 to each observation. The sum of the WEIGHT variable values represents the total number of observations.

If any value of the WEIGHT variable is negative, PROC FREQTAB displays the frequencies computed from the weighted values but does not compute percentages and statistics. If you create an output data set by using the OUT= option in the TABLES statement, PROC FREQTAB assigns missing values to the PERCENT variable. PROC FREQTAB also assigns missing values to the variables that the OUTEXPECT and OUTPCT options provide. If any value of the WEIGHT variable is negative, you cannot create an output data set by using the OUTPUT statement because statistics are not computed when there are negative weights.

You can specify the following option in the WEIGHT statement:

**ZEROS**

includes observations that have weights of 0. By default, PROC FREQTAB ignores observations that have weights of 0.

If you specify the ZEROS option, frequency and crosstabulation tables display levels that contain only zero-weight observations. If you do not specify the ZEROS option, PROC FREQTAB does not process observations that have weights of 0 and therefore does not display levels that contain only zero-weight observations.

When you specify the ZEROS option, PROC FREQTAB includes zero-weight levels in chi-square tests and binomial computations for one-way tables. This makes it possible to compute binomial tests and estimates for a reference level that contains no observations with positive weights.

For two-way tables, the ZEROS option enables computation of kappa statistics when there are levels that contain no observations with positive weights. For more information, see the section “Tables with Zero-Weight Rows or Columns” on page 261.

Even when you specify the ZEROS option, PROC FREQTAB does not compute CHISQ or MEASURES statistics for two-way tables that contain a zero-weight row or column because most of these statistics are undefined in this case.

By default, the ZEROS option includes zero-weight table cells in the LIST table and the OUT= data set. To suppress zero-weight cells, you can specify the NOSPARSE option in the TABLES statement.
Chapter 5: The FREQTAB Procedure

Details: FREQTAB Procedure

Inputting Frequency Counts

PROC FREQTAB can use either raw data or cell count data to produce frequency and crosstabulation tables. Raw data, also known as case-record data, report the data as one record for each subject or sample member. Cell count data report the data as a table, listing all possible combinations of data values along with the frequency counts. This way of presenting data often appears in published results.

The following DATA step statements store raw data in a SAS data table:

```sas
data mycas.Raw;
  input Subject $ R C @@;
datalines;
  01 1 1 02 1 1 03 1 1 04 1 1 05 1 1
  06 1 2 07 1 2 08 1 2 09 2 1 10 2 1
  11 2 1 12 2 1 13 2 2 14 2 2 14 2 2
;
```

You can store the same data as cell counts by using the following DATA step statements:

```sas
data mycas.CellCounts;
  input R C Count @@;
datalines;
  1 1 5 1 2 3
  2 1 4 2 2 3
;
```

The variable R contains the values for the rows, and the variable C contains the values for the columns. The variable Count contains the cell count for each row and column combination.

Both the mycas.Raw data table and the mycas.CellCounts data table produce identical frequency counts, two-way tables, and statistics. When you use the mycas.CellCounts data table, you must include a WEIGHT statement to specify that the variable Count contains cell counts. For example, the following PROC FREQTAB statements create a two-way crosstabulation table by using the mycas.CellCounts data table:

```sas
proc freqtab data=mycas.CellCounts;
  tables R*C;
  weight Count;
run;
```

Missing Values

If an observation has a missing value for the WEIGHT variable, PROC FREQTAB excludes that observation from the analysis.

If an observation has a missing value for a TABLES statement variable, by default PROC FREQTAB does not include that observation in the frequency or crosstabulation table (or in the frequency total, percentages, and
statistics). PROC FREQTAB displays the frequency of missing values after the frequency or crosstabulation table.

PROC FREQTAB includes the frequency of missing values in the output data sets. The OUT= data set that you specify in the TABLES statement includes an observation that contains the frequency of missing values. The OUT= data set that you specify in the OUTPUT statement includes the frequency of missing values in the variable NMISS when you specify the NMISS option in the OUTPUT statement.

The following options change the way that PROC FREQTAB handles missing values of TABLES statement variables:

MISSPRINT    displays missing value frequencies in frequency tables but does not include them in computing percentages or statistics. This option is available for one-way frequency tables and for tables in list format (which you can request by specifying the LIST option in the TABLES statement).

MISSING      treats missing values as a valid nonmissing level for all TABLES statement variables. This option displays missing levels in frequency and crosstabulation tables and includes them in computing percentages and statistics.

This example shows the three ways that PROC FREQTAB can handle missing values of TABLES statement variables. The following DATA step statements create a data table that contains a missing value for the variable A:

```plaintext
data mycas.One;
  input A Freq;
datalines;
  1 2
  2 2
  . 2
;
```

The following PROC FREQTAB statements request a one-way frequency table for the variable A. The first request does not specify a missing value option. The second request specifies the MISSPRINT option in the PROC FREQTAB statement. The third request specifies the MISSING option in the PROC FREQTAB statement.

```plaintext
proc freqtab data=mycas.One;
  tables A;
  weight Freq;
  title 'Default';
run;
proc freqtab data=mycas.One missprint;
  tables A;
  weight Freq;
  title 'MISSPRINT Option';
run;
proc freqtab data=mycas.One missing;
  tables A;
  weight Freq;
  title 'MISSING Option';
run;
```
Figure 5.9 displays the frequency tables produced by this example. The first table shows PROC FREQTAB’s default behavior for handling missing values. The frequency table does not include the missing level of the TABLES statement variable A. PROC FREQTAB displays the frequency of missing values after the table.

The second table, for which the MISSPRINT option is specified, displays the missing level of A but does not include its frequency in computing the total frequency and percentages.

The third table, for which the MISSING option is specified, treats the missing level as a valid nonmissing level. The frequency table displays the missing level of A and includes its frequency in computing the total frequency and percentages.

**Figure 5.9 Missing Values in Frequency Tables**

**Default**

The FREQTAB Procedure

<table>
<thead>
<tr>
<th></th>
<th>Frequency</th>
<th>Percent</th>
<th>Cumulative Frequency</th>
<th>Cumulative Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>50.00</td>
<td>2</td>
<td>50.00</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>50.00</td>
<td>4</td>
<td>100.00</td>
</tr>
</tbody>
</table>

Frequency Missing = 2

**MISSPRINT Option**

The FREQTAB Procedure

<table>
<thead>
<tr>
<th></th>
<th>Frequency</th>
<th>Percent</th>
<th>Cumulative Frequency</th>
<th>Cumulative Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.</td>
<td>2</td>
<td></td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>50.00</td>
<td>2</td>
<td>50.00</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>50.00</td>
<td>4</td>
<td>100.00</td>
</tr>
</tbody>
</table>

Frequency Missing = 2

**MISSING Option**

The FREQTAB Procedure

<table>
<thead>
<tr>
<th></th>
<th>Frequency</th>
<th>Percent</th>
<th>Cumulative Frequency</th>
<th>Cumulative Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.</td>
<td>2</td>
<td>33.33</td>
<td>2</td>
<td>33.33</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>33.33</td>
<td>4</td>
<td>66.67</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>33.33</td>
<td>6</td>
<td>100.00</td>
</tr>
</tbody>
</table>

When a combination of variable levels in a two-way table is missing, PROC FREQTAB assigns 0 to the frequency count of the corresponding table cell. The default crosstabulation table and the CROSSLIST crosstabulation table display table cells that have frequencies of 0.

By default, the LIST crosstabulation table and the OUT= output data set do not include table cells that have frequencies of 0. If you specify the ZEROS option in the WEIGHT statement to include observations that have weights of 0, the LIST table and the OUT= output data set include zero-frequency table cells by default unless you specify the NOSPARSE option in the TABLES statement.
Statistical Computations

Definitions and Notation

A two-way table represents the crosstabulation of row variable $X$ and column variable $Y$. Let the table row values or levels be denoted by $X_i, i = 1, 2, \ldots, R$, and the column values by $Y_j, j = 1, 2, \ldots, C$. Let $n_{ij}$ denote the frequency of the table cell in the $i$th row and $j$th column and define the following notation:

$$n_i = \sum_j n_{ij} \quad \text{(row totals)}$$

$$n_j = \sum_i n_{ij} \quad \text{(column totals)}$$

$$n = \sum_i \sum_j n_{ij} \quad \text{(overall total)}$$

$$p_{ij} = \frac{n_{ij}}{n} \quad \text{(cell percentages)}$$

$$p_i = \frac{n_i}{n} \quad \text{(row percentages of total)}$$

$$p_j = \frac{n_j}{n} \quad \text{(column percentages of total)}$$

$R_i = \text{score for row } i$

$C_j = \text{score for column } j$

$$\bar{R} = \frac{\sum_i n_i R_i}{n} \quad \text{(average row score)}$$

$$\bar{C} = \frac{\sum_j n_j C_j}{n} \quad \text{(average column score)}$$

$$A_{ij} = \sum_{k>i} \sum_{l>j} n_{kl} + \sum_{k<i} \sum_{l<j} n_{kl}$$

$$D_{ij} = \sum_{k>i} \sum_{l<j} n_{kl} + \sum_{k<i} \sum_{l>j} n_{kl}$$

$$P = \sum_i \sum_j n_{ij} A_{ij} \quad \text{(twice the number of concordances)}$$

$$Q = \sum_i \sum_j n_{ij} D_{ij} \quad \text{(twice the number of discordances)}$$

Scores

PROC FREQTAB uses scores of the variable values to compute the Mantel-Haenszel chi-square, Pearson correlation, Cochran-Armitage test for trend, weighted kappa coefficient, and Cochran-Mantel-Haenszel statistics. The SCORES= option in the TABLES statement specifies the score type that PROC FREQTAB uses. The available score types are TABLE, RANK, RIDIT, and MODRIDIT scores. The default score type is TABLE. Using MODRIDIT, RANK, or RIDIT scores yields nonparametric analyses.

For numeric variables, table scores are the values of the row and column levels. If the row or column variable is formatted, then the table score is the internal numeric value corresponding to that level. If two or more numeric values are classified into the same formatted level, then the internal numeric value for that level is
the smallest of these values. For character variables, table scores are defined as the row numbers and column numbers (that is, 1 for the first row, 2 for the second row, and so on).

Rank scores, which you request with the SCORES=RANK option, are defined as

\[
R_i^1 = \sum_{k < i} n_{k.} + (n_{i.} + 1)/2 \quad i = 1, 2, \ldots, R
\]

\[
C_j^1 = \sum_{l < j} n_{.l} + (n_{.j} + 1)/2 \quad j = 1, 2, \ldots, C
\]

where \( R_i^1 \) is the rank score of row \( i \), and \( C_j^1 \) is the rank score of column \( j \). Note that rank scores yield midranks for tied values.

Ridit scores, which you request with the SCORES=RIDIT option, are defined as rank scores standardized by the sample size (Bross 1958; Mack and Skillings 1980). Ridit scores are derived from the rank scores as

\[
R_i^2 = R_i^1 / n \quad i = 1, 2, \ldots, R
\]

\[
C_j^2 = C_j^1 / n \quad j = 1, 2, \ldots, C
\]

Modified ridit scores (SCORES=MODRIDIT) represent the expected values of the order statistics of the uniform distribution on (0,1) (Van Elteren 1960; Lehmann and D’Abrera 2006). Modified ridit scores are derived from rank scores as

\[
R_i^3 = R_i^1 / (n + 1) \quad i = 1, 2, \ldots, R
\]

\[
C_j^3 = C_j^1 / (n + 1) \quad j = 1, 2, \ldots, C
\]

**Chi-Square Tests and Statistics**

The CHISQ option provides chi-square tests of homogeneity or independence and measures of association that are based on the chi-square statistic. When you specify the CHISQ option in the TABLES statement, PROC FREQTAB computes the following chi-square tests for each two-way table: Pearson chi-square, likelihood ratio chi-square, and Mantel-Haenszel chi-square tests. PROC FREQTAB provides the following measures of association that are based on the Pearson chi-square statistic: phi coefficient, contingency coefficient, and Cramér’s \( V \). For \( 2 \times 2 \) tables, the CHISQ option also provides Fisher’s exact test and the continuity-adjusted chi-square statistic. You can request Fisher’s exact test for general \( R \times C \) tables by specifying the FISHER option in the TABLES or EXACT statement.

If you specify the CHISQ option for one-way tables, PROC FREQTAB provides a one-way Pearson chi-square goodness-of-fit test. If you specify the CHISQ(LRCHI) option for one-way tables, PROC FREQTAB also provides a one-way likelihood ratio chi-square test. The other tests and statistics that the CHISQ option produces are available only for two-way tables.

For two-way tables, the null hypothesis for the chi-square tests is no association between the row variable and the column variable. When the sample size \( n \) is large, the test statistics have asymptotic chi-square distributions under the null hypothesis. When the sample size is not large, or when the data set is sparse or heavily tied, exact tests might be more appropriate than asymptotic tests. PROC FREQTAB provides exact \( p \)-values for the Pearson chi-square, likelihood ratio chi-square, and Mantel-Haenszel chi-square tests, in addition to Fisher’s exact test. For one-way tables, PROC FREQTAB provides exact \( p \)-values for the Pearson and likelihood ratio chi-square goodness-of-fit tests. You can request these exact tests by specifying the
corresponding options in the EXACT statement. See the section “Exact Statistics” on page 271 for more information.

The Mantel-Haenszel chi-square statistic is appropriate only when both variables lie on an ordinal scale. The other chi-square tests and statistics in this section are appropriate for either nominal or ordinal variables. The following sections give the formulas that PROC FREQTAB uses to compute the chi-square tests and statistics. For more information about these statistics, see Agresti (2007) and Stokes, Davis, and Koch (2012), and the other references cited.

**Chi-Square Test for One-Way Tables**
For one-way frequency tables, the CHISQ option in the TABLES statement provides a chi-square goodness-of-fit test. Let \( C \) denote the number of classes, or levels, in the one-way table. Let \( f_i \) denote the frequency of class \( i \) (or the number of observations in class \( i \)) for \( i = 1, 2, \ldots, C \). Then PROC FREQTAB computes the one-way chi-square statistic as

\[
Q_P = \sum_{i=1}^{C} \frac{(f_i - e_i)^2}{e_i}
\]

where \( e_i \) is the expected frequency for class \( i \) under the null hypothesis.

In the test for equal proportions, which is the default for the CHISQ option, the null hypothesis specifies equal proportions of the total sample size for each class. Under this null hypothesis, the expected frequency for each class equals the total sample size divided by the number of classes,

\[
e_i = \frac{n}{C} \quad \text{for} \quad i = 1, 2, \ldots, C
\]

In the test for specified frequencies, which PROC FREQTAB computes when you input null hypothesis frequencies by using the TESTF= option, the expected frequencies are the TESTF= values that you specify. In the test for specified proportions, which PROC FREQTAB computes when you input null hypothesis proportions by using the TESTP= option, the expected frequencies are determined from the specified TESTP= proportions \( p_i \) as

\[
e_i = p_i \times n \quad \text{for} \quad i = 1, 2, \ldots, C
\]

Under the null hypothesis (of equal proportions, specified frequencies, or specified proportions), \( Q_P \) has an asymptotic chi-square distribution with \( C-1 \) degrees of freedom.

In addition to the asymptotic test, you can request an exact one-way chi-square test by specifying the CHISQ option in the EXACT statement. See the section “Exact Statistics” on page 271 for more information.

**Pearson Chi-Square Test for Two-Way Tables**
The Pearson chi-square for two-way tables involves the differences between the observed and expected frequencies, where the expected frequencies are computed under the null hypothesis of independence. The Pearson chi-square statistic is computed as

\[
Q_P = \sum_{i} \sum_{j} \frac{(n_{ij} - e_{ij})^2}{e_{ij}}
\]

where \( n_{ij} \) is the observed frequency in table cell \((i, j)\) and \( e_{ij} \) is the expected frequency in table cell \((i, j)\). The expected frequency is computed under the null hypothesis that the row and column variables are independent,

\[
e_{ij} = \frac{(n_i \times n_j)}{n}
\]
When the row and column variables are independent, $Q_P$ has an asymptotic chi-square distribution with $(R-1)(C-1)$ degrees of freedom. For large values of $Q_P$, this test rejects the null hypothesis in favor of the alternative hypothesis of general association.

In addition to the asymptotic test, you can request an exact Pearson chi-square test by specifying the PCHI or CHISQ option in the EXACT statement. See the section “Exact Statistics” on page 271 for more information.

For $2 \times 2$ tables, the Pearson chi-square is also appropriate for testing the equality of two binomial proportions. For $R \times 2$ and $2 \times C$ tables, the Pearson chi-square tests the homogeneity of proportions. For more information, see Fienberg (1980).

**Standardized Residuals**

When you specify the CROSSLIST(STDRES) option in the TABLES statement for two-way or multiway tables, PROC FREQTAB displays the standardized residuals in the CROSSLIST table.

The standardized residual of a crosstabulation table cell is the ratio of \((frequency – expected)\) to its standard error, where \(frequency\) is the table cell frequency and \(expected\) is the estimated expected cell frequency. The expected frequency is computed under the null hypothesis that the row and column variables are independent. See the section “Pearson Chi-Square Test for Two-Way Tables” on page 203 for more information.

PROC FREQTAB computes the standardized residual of table cell \((i, j)\) as

$$
\frac{(n_{ij} - e_{ij})}{\sqrt{e_{ij}(1 - p_i)(1 - p_j)}}
$$

where \(n_{ij}\) is the observed frequency of table cell \((i, j)\), \(e_{ij}\) is the expected frequency of the table cell, \(p_i\) is the proportion in row \(i\) \((n_i/n)\), and \(p_j\) is the proportion in column \(j\) \((n_j/n)\). The expected frequency of table cell \((i, j)\) is computed as

$$
e_{ij} = (n_i \times n_j) / n
$$

Under the null hypothesis of independence, each standardized residual has an asymptotic standard normal distribution. See section 2.4.5 of Agresti (2007) for more information.

**Likelihood Ratio Chi-Square Test for One-Way Tables**

For one-way frequency tables, the CHISQ(LRCHI) option in the TABLES statement provides a likelihood ratio chi-square goodness-of-fit test. By default, the likelihood ratio test is based on the null hypothesis of equal proportions in the \(C\) classes (levels) of the one-way table. If you specify null hypothesis proportions or frequencies by using the CHISQ(TESTP=) or CHISQ(TESTF=) option, respectively, the likelihood ratio test is based on the null hypothesis values that you specify.

PROC FREQTAB computes the one-way likelihood ratio test as

$$
G^2 = 2 \sum_{i=1}^{C} f_i \log \left( \frac{f_i}{e_i} \right)
$$

where \(f_i\) is the observed frequency of class \(i\), and \(e_i\) is the expected frequency of class \(i\) under the null hypothesis.

For the null hypothesis of equal proportions, the expected frequency of each class is the total sample size divided by the number of classes,

$$
e_i = n / C \quad \text{for} \quad i = 1, 2, \ldots, C
$$
If you provide null hypothesis frequencies by specifying the CHISQ(TESTF=) option in the TABLES statement, the expected frequencies are the TESTF= values that you specify. If you provide null hypothesis proportions by specifying the CHISQ(TESTP=) option in the TABLES statement, PROC FREQTAB computes the expected frequencies as

$$e_i = p_i \times n \quad \text{for} \quad i = 1, 2, \ldots, C$$

where the proportions $p_i$ are the TESTP= values that you specify.

Under the null hypothesis (of equal proportions, specified frequencies, or specified proportions), the likelihood ratio statistic $G^2$ has an asymptotic chi-square distribution with $C-1$ degrees of freedom.

In addition to the asymptotic test, you can request an exact one-way likelihood ratio chi-square test by specifying the LRCHI option in the EXACT statement. See the section “Exact Statistics” on page 271 for more information.

**Likelihood Ratio Chi-Square Test**

The likelihood ratio chi-square involves the ratios between the observed and expected frequencies. The likelihood ratio chi-square statistic is computed as

$$G^2 = 2 \sum_i \sum_j n_{ij} \log \left( \frac{n_{ij}}{e_{ij}} \right)$$

where $n_{ij}$ is the observed frequency in table cell $(i, j)$ and $e_{ij}$ is the expected frequency for table cell $(i, j)$.

When the row and column variables are independent, $G^2$ has an asymptotic chi-square distribution with $(R-1)(C-1)$ degrees of freedom.

In addition to the asymptotic test, you can request an exact likelihood ratio chi-square test by specifying the LRCHI or CHISQ option in the EXACT statement. See the section “Exact Statistics” on page 271 for more information.

**Continuity-Adjusted Chi-Square Test**

The continuity-adjusted chi-square for $2 \times 2$ tables is similar to the Pearson chi-square, but it is adjusted for the continuity of the chi-square distribution. The continuity-adjusted chi-square is most useful for small sample sizes. The use of the continuity adjustment is somewhat controversial; this chi-square test is more conservative (and more like Fisher’s exact test) when the sample size is small. As the sample size increases, the continuity-adjusted chi-square becomes more like the Pearson chi-square.

The continuity-adjusted chi-square statistic is computed as

$$Q_C = \sum_i \sum_j \left( \max(0, |n_{ij} - e_{ij}| - 0.5) \right)^2 / e_{ij}$$

Under the null hypothesis of independence, $Q_C$ has an asymptotic chi-square distribution with $(R-1)(C-1)$ degrees of freedom.

**Mantel-Haenszel Chi-Square Test**

The Mantel-Haenszel chi-square statistic tests the alternative hypothesis that there is a linear association between the row variable and the column variable. Both variables must lie on an ordinal scale. The Mantel-Haenszel chi-square statistic is computed as

$$Q_{MH} = (n - 1)r^2$$
where \( r \) is the Pearson correlation between the row variable and the column variable. For a description of the Pearson correlation, see the “Pearson Correlation Coefficient” on page 211. The Pearson correlation and thus the Mantel-Haenszel chi-square statistic use the scores that you specify in the SCORES= option in the TABLES statement. See Mantel and Haenszel (1959) and Landis, Heyman, and Koch (1978) for more information.

Under the null hypothesis of no association, \( Q_{MH} \) has an asymptotic chi-square distribution with 1 degree of freedom.

In addition to the asymptotic test, you can request an exact Mantel-Haenszel chi-square test by specifying the MHCHI or CHISQ option in the EXACT statement. See the section “Exact Statistics” on page 271 for more information.

**Fisher’s Exact Test**

Fisher’s exact test is another test of association between the row and column variables. This test assumes that the row and column totals are fixed and uses the hypergeometric distribution to compute probabilities of possible tables conditional on the observed row and column totals. Fisher’s exact test does not depend on any large-sample distribution assumptions, and so it is appropriate even for small sample sizes and for sparse tables.

### 2 × 2 Tables

For 2 × 2 tables, PROC FREQTAB gives the following information for Fisher’s exact test: table probability, two-sided \( p \)-value, left-sided \( p \)-value, and right-sided \( p \)-value. The table probability is the hypergeometric probability of the observed table, and is in fact the value of the test statistic for Fisher’s exact test.

Where \( p \) is the hypergeometric probability of a specific table with the observed row and column totals, Fisher’s exact \( p \)-values are computed by summing probabilities \( p \) over defined sets of tables,

\[
\text{Prob} = \sum_A p
\]

The two-sided \( p \)-value is the sum of all possible table probabilities (conditional on the observed row and column totals) that are less than or equal to the observed table probability. For the two-sided \( p \)-value, the set \( A \) includes all possible tables with hypergeometric probabilities less than or equal to the probability of the observed table. A small two-sided \( p \)-value supports the alternative hypothesis of association between the row and column variables.

For 2 × 2 tables, one-sided \( p \)-values for Fisher’s exact test are defined in terms of the frequency of the cell in the first row and first column of the table, the (1,1) cell. Denoting the observed (1,1) cell frequency by \( n_{11} \), the left-sided \( p \)-value for Fisher’s exact test is the probability that the (1,1) cell frequency is less than or equal to \( n_{11} \). For the left-sided \( p \)-value, the set \( A \) includes those tables with a (1,1) cell frequency less than or equal to \( n_{11} \). A small left-sided \( p \)-value supports the alternative hypothesis that the probability of an observation being in the first cell is actually less than expected under the null hypothesis of independent row and column variables.

Similarly, for a right-sided alternative hypothesis, \( A \) is the set of tables where the frequency of the (1,1) cell is greater than or equal to that in the observed table. A small right-sided \( p \)-value supports the alternative that the probability of the first cell is actually greater than that expected under the null hypothesis.

Because the (1,1) cell frequency completely determines the 2 × 2 table when the marginal row and column sums are fixed, these one-sided alternatives can be stated equivalently in terms of other cell probabilities or
ratios of cell probabilities. The left-sided alternative is equivalent to an odds ratio less than 1, where the odds ratio is \((n_{11}n_{22} / n_{12}n_{21})\). The left-sided alternative is also equivalent to the column 1 risk for row 1 being less than the column 1 risk for row 2, \(p_{1|1} < p_{1|2}\). Similarly, the right-sided alternative is equivalent to the column 1 risk for row 1 being greater than the column 1 risk for row 2, \(p_{1|1} > p_{1|2}\). For more information, see Agresti (2007).

**R × C Tables**

Fisher’s exact test was extended to general \(R \times C\) tables by Freeman and Halton (1951), and this test is also known as the Freeman-Halton test. For \(R \times C\) tables, the two-sided \(p\)-value definition is the same as for \(2 \times 2\) tables. The set \(A\) contains all tables with \(p\) less than or equal to the probability of the observed table. A small \(p\)-value supports the alternative hypothesis of association between the row and column variables. For \(R \times C\) tables, Fisher’s exact test is inherently two-sided. The alternative hypothesis is defined only in terms of general, and not linear, association. Therefore, Fisher’s exact test does not have right-sided or left-sided \(p\)-values for general \(R \times C\) tables.

For \(R \times C\) tables, PROC FREQTAB computes Fisher’s exact test by using the network algorithm of Mehta and Patel (1983), which provides a faster and more efficient solution than direct enumeration. See the section “Exact Statistics” on page 271 for more details.

**Phi Coefficient**

The phi coefficient is a measure of association derived from the Pearson chi-square. The range of the phi coefficient is \(-1 \leq \phi \leq 1\) for \(2 \times 2\) tables. For tables larger than \(2 \times 2\), the range is \(0 \leq \phi \leq \min(\sqrt{R-1}, \sqrt{C-1})\) (Liebetrau 1983). The phi coefficient is computed as

\[
\phi = (n_{11}n_{22} - n_{12}n_{21}) / \sqrt{n_{11}n_{22}n_{12}n_{21}}
\]

for \(2 \times 2\) tables

\[
\phi = \sqrt{Q_P/n}
\]

for \(2 \times 2\) tables

otherwise

See Fleiss, Levin, and Paik (2003, pp. 98–99) for more information.

**Contingency Coefficient**

The contingency coefficient is a measure of association derived from the Pearson chi-square. The range of the contingency coefficient is \(0 \leq P \leq \sqrt{(m-1)/m}\), where \(m = \min(R, C)\) (Liebetrau 1983). The contingency coefficient is computed as

\[
P = \sqrt{Q_P / (Q_P + n)}
\]

See Kendall and Stuart (1979, pp. 587–588) for more information.

**Cramér’s \(V\)**

Cramér’s \(V\) is a measure of association derived from the Pearson chi-square. It is designed so that the attainable upper bound is always 1. The range of Cramér’s \(V\) is \(-1 \leq V \leq 1\) for \(2 \times 2\) tables; for tables larger than \(2 \times 2\), the range is \(0 \leq V \leq 1\). Cramér’s \(V\) is computed as

\[
V = \phi
\]

for \(2 \times 2\) tables

\[
V = \sqrt{\frac{Q_P/n}{\min(R-1, C-1)}}
\]

otherwise

See Kendall and Stuart (1979, p. 588) for more information.
Measures of Association

When you specify the MEASURES option in the TABLES statement, PROC FREQTAB computes several statistics that describe the association between the row and column variables of the contingency table. The following are measures of ordinal association that consider whether the column variable \( Y \) tends to increase as the row variable \( X \) increases: gamma, Kendall’s tau-\( b \), Stuart’s tau-\( c \), and Somers’ \( D \). These measures are appropriate for ordinal variables, and they classify pairs of observations as concordant or discordant. A pair is concordant if the observation with the larger value of \( X \) also has the larger value of \( Y \). A pair is discordant if the observation with the larger value of \( X \) has the smaller value of \( Y \). See Agresti (2007) and the other references cited for the individual measures of association.

The Pearson correlation coefficient and the Spearman rank correlation coefficient are also appropriate for ordinal variables. The Pearson correlation describes the strength of the linear association between the row and column variables, and it is computed by using the row and column scores specified by the SCORES= option in the TABLES statement. The Spearman correlation is computed with rank scores. The polychoric correlation (requested by the PLCORR option) also requires ordinal variables and assumes that the variables have an underlying bivariate normal distribution. The following measures of association do not require ordinal variables and are appropriate for nominal variables: lambda asymmetric, lambda symmetric, and the uncertainty coefficients.

PROC FREQTAB computes estimates of the measures according to the formulas given in the following sections. For each measure, PROC FREQTAB computes an asymptotic standard error (ASE), which is the square root of the asymptotic variance denoted by \( \text{Var} \) in the following sections.

Confidence Limits

If you specify the CL option in the TABLES statement, PROC FREQTAB computes asymptotic confidence limits for all MEASURES statistics. The confidence coefficient is determined according to the value of the ALPHA= option, which, by default, is 0.05 and produces 95% confidence limits.

The confidence limits are computed as

\[
\text{Est} \pm (z_{\alpha/2} \times \text{ASE})
\]

where \( \text{Est} \) is the estimate of the measure, \( z_{\alpha/2} \) is the 100(1 - \( \alpha \)/2)th percentile of the standard normal distribution, and ASE is the asymptotic standard error of the estimate.

Asymptotic Tests

For each measure that you specify in the TEST statement, PROC FREQTAB computes an asymptotic test of the null hypothesis that the measure is 0. Asymptotic tests are available for the following measures of association: gamma, Kendall’s tau-\( b \), Stuart’s tau-\( c \), Somers’ \( D(C|R) \), Somers’ \( D(R|C) \), the Pearson correlation coefficient, and the Spearman rank correlation coefficient. To compute an asymptotic test, PROC FREQTAB uses a standardized test statistic \( z \), which has an asymptotic standard normal distribution under the null hypothesis. The test statistic is computed as

\[
z = \frac{\text{Est}}{\sqrt{\text{Var}_0(\text{Est})}}
\]

where \( \text{Est} \) is the estimate of the measure and \( \text{Var}_0(\text{Est}) \) is the variance of the estimate under the null hypothesis. Formulas for \( \text{Var}_0(\text{Est}) \) for the individual measures of association are given in the following sections.
Note that the ratio of Est to $\sqrt{\text{Var}(\text{Est})}$ is the same for the following measures: gamma, Kendall’s tau-$b$, Stuart’s tau-$c$, Somers’ $D(C|R)$, and Somers’ $D(R|C)$. Therefore, the tests for these measures are identical. For example, the $p$-values for the test of $H_0: \text{gamma} = 0$ equal the $p$-values for the test of $H_0: \tau = b = 0$.

PROC FREQTAB computes one-sided and two-sided $p$-values for each of these tests. When the test statistic $z$ is greater than its null hypothesis expected value of 0, PROC FREQTAB displays the right-sided $p$-value, which is the probability of a larger value of the statistic occurring under the null hypothesis. A small right-sided $p$-value supports the alternative hypothesis that the true value of the measure is greater than 0. When the test statistic is less than or equal to 0, PROC FREQTAB displays the left-sided $p$-value, which is the probability of a smaller value of the statistic occurring under the null hypothesis. A small left-sided $p$-value supports the alternative hypothesis that the true value of the measure is less than 0. The one-sided $p$-value $P_1$ can be expressed as

$$P_1 = \begin{cases} 
\text{Prob}(Z > z) & \text{if } z > 0 \\
\text{Prob}(Z < z) & \text{if } z \leq 0
\end{cases}$$

where $Z$ has a standard normal distribution. The two-sided $p$-value $P_2$ is computed as

$$P_2 = \text{Prob}(|Z| > |z|)$$

**Exact Tests**

Exact tests are available for the following measures of association: Kendall’s tau-$b$, Stuart’s tau-$c$, Somers’ $D(C|R)$ and $D(R|C)$, the Pearson correlation coefficient, and the Spearman rank correlation coefficient. If you request an exact test for a measure of association in the EXACT statement, PROC FREQTAB computes the exact test of the hypothesis that the measure is 0. For more information, see the section “Exact Statistics” on page 271.

**Gamma**

The gamma ($\Gamma$) statistic is based only on the number of concordant and discordant pairs of observations. It ignores tied pairs (that is, pairs of observations that have equal values of $X$ or equal values of $Y$). Gamma is appropriate only when both variables lie on an ordinal scale. The range of gamma is $-1 \leq \Gamma \leq 1$. If the row and column variables are independent, gamma tends to be close to 0. Gamma is computed as

$$G = (P - Q) / (P + Q)$$

and the asymptotic variance is

$$\text{Var}(G) = \frac{16}{(P + Q)^4} \sum_i \sum_j n_{ij} (Q A_{ij} - P D_{ij})^2$$

For $2 \times 2$ tables, gamma is equivalent to Yule’s $Q$. See Goodman and Kruskal (1979) and Agresti (2002) for more information.

The variance under the null hypothesis that gamma equals 0 is computed as

$$\text{Var}_0(G) = \frac{4}{(P + Q)^2} \left( \sum_i \sum_j n_{ij} (A_{ij} - D_{ij})^2 - (P - Q)^2 / n \right)$$

For more information, see Brown and Benedetti (1977b).
**Kendall’s Tau-b**

Kendall’s tau-b ($\tau_b$) is similar to gamma except that tau-b uses a correction for ties. Tau-b is appropriate only when both variables lie on an ordinal scale. The range of tau-b is $-1 \leq \tau_b \leq 1$. Kendall’s tau-b is computed as

$$t_b = (P - Q) / \sqrt{w_r w_c}$$

and the asymptotic variance is

$$\text{Var}(t_b) = \frac{1}{w^4} \left( \sum_i \sum_j n_{ij} (2w d_{ij} + t_b v_{ij})^2 - n^3 t_b^2 (w_r + w_c)^2 \right)$$

where

- $w = \sqrt{w_r w_c}$
- $w_r = n^2 - \sum_i n_i^2$
- $w_c = n^2 - \sum_j n_j^2$
- $d_{ij} = A_{ij} - D_{ij}$
- $v_{ij} = n_i w_c + n_j w_r$

See Kendall (1955) for more information.

The variance under the null hypothesis that tau-b equals 0 is computed as

$$\text{Var}_0(t_b) = \frac{4}{w_r w_c} \left( \sum_i \sum_j n_{ij} (A_{ij} - D_{ij})^2 - (P - Q)^2 / n \right)$$

For more information, see Brown and Benedetti (1977b).

PROC FREQTAB also provides an exact test for the Kendall’s tau-b. You can request this test by specifying the KENTB option in the EXACT statement. See the section “Exact Statistics” on page 271 for more information.

**Stuart’s Tau-c**

Stuart’s tau-c ($\tau_c$) makes an adjustment for table size in addition to a correction for ties. Tau-c is appropriate only when both variables lie on an ordinal scale. The range of tau-c is $-1 \leq \tau_c \leq 1$. Stuart’s tau-c is computed as

$$t_c = m(P - Q) / n^2 (m - 1)$$

and the asymptotic variance is

$$\text{Var}(t_c) = \frac{4m^2}{(m - 1)^2 n^4} \left( \sum_i \sum_j n_{ij} d_{ij}^2 - (P - Q)^2 / n \right)$$
where \( m = \min(R, C) \) and \( d_{ij} = A_{ij} - D_{ij} \). The variance under the null hypothesis that \( \text{tau-c} \) equals 0 is the same as the asymptotic variance

\[
\text{Var}_0(t_c) = \text{Var}(t_c)
\]

For more information, see Brown and Benedetti (1977b).

PROC FREQTAB also provides an exact test for the Stuart’s \( \text{tau-c} \). You can request this test by specifying the \( \text{STUTC} \) option in the EXACT statement. See the section “Exact Statistics” on page 271 for more information.

**Somers’ D**

Somers’ \( D(C|R) \) and Somers’ \( D(R|C) \) are asymmetric modifications of \( \text{tau-b} \). \( C|R \) indicates that the row variable \( X \) is regarded as the independent variable and the column variable \( Y \) is regarded as dependent. Similarly, \( R|C \) indicates that the column variable \( Y \) is regarded as the independent variable and the row variable \( X \) is regarded as dependent. Somers’ \( D \) differs from \( \text{tau-b} \) in that it uses a correction only for pairs that are tied on the independent variable. Somers’ \( D \) is appropriate only when both variables lie on an ordinal scale. The range of Somers’ \( D \) is \(-1 \leq D \leq 1\). Somers’ \( D(C|R) \) is computed as

\[
D(C|R) = \frac{(P - Q)}{w_r}
\]

and its asymptotic variance is

\[
\text{Var}(D(C|R)) = 4 \frac{w_r^2}{w_r^4} \sum_i \sum_j n_{ij} (w_r d_{ij} - (P - Q)(n - n_i))^2
\]

where \( d_{ij} = A_{ij} - D_{ij} \) and

\[
w_r = n^2 - \sum_i n_i^2.
\]

For more information, see Somers (1962); Goodman and Kruskal (1979); Liebetrau (1983).

The variance under the null hypothesis that \( D(C|R) \) equals 0 is computed as

\[
\text{Var}_0(D(C|R)) = 4 \frac{w_r^2}{w_r^4} \left( \sum_i \sum_j n_{ij} (A_{ij} - D_{ij})^2 - (P - Q)^2/n \right)
\]

For more information, see Brown and Benedetti (1977b).

Formulas for Somers’ \( D(R|C) \) are obtained by interchanging the indices.

PROC FREQTAB also provides exact tests for Somers’ \( D(C|R) \) and \( (R|C) \). You can request these tests by specifying the \( \text{SMDCR} \) and \( \text{SMDCR} \) options in the EXACT statement. See the section “Exact Statistics” on page 271 for more information.

**Pearson Correlation Coefficient**

The Pearson correlation coefficient (\( \rho \)) is computed by using the scores specified in the SCORES= option. This measure is appropriate only when both variables lie on an ordinal scale. The range of the Pearson correlation is \(-1 \leq \rho \leq 1\). The Pearson correlation coefficient is computed as

\[
r = \frac{v}{w} = s_{rc}/\sqrt{s_r s_c}
\]
and its asymptotic variance is

$$\text{Var}(r) = \frac{1}{w^4} \sum_i \sum_j n_{ij} \left( w(R_i - \bar{R})(C_j - \bar{C}) - \frac{b_{ij}v}{2w} \right)^2$$

where $R_i$ and $C_j$ are the row and column scores and

$$s_r = \sum_i \sum_j n_{ij} (R_i - \bar{R})^2$$

$$s_c = \sum_i \sum_j n_{ij} (C_j - \bar{C})^2$$

$$s_{rc} = \sum_i \sum_j n_{ij} (R_i - \bar{R})(C_j - \bar{C})$$

$$b_{ij} = (R_i - \bar{R})^2 s_c + (C_j - \bar{C})^2 s_r$$

$$v = s_{rc}$$

$$w = \sqrt{s_r s_c}$$

For more information, see Snedecor and Cochran (1989).

The \texttt{SCORES} option in the \texttt{TABLES} statement determines the type of row and column scores used to compute the Pearson correlation (and other score-based statistics). The default is \texttt{SCORES=TABLE}. See the section “Scores” on page 201 for details about the available score types and how they are computed.

The variance under the null hypothesis that the correlation equals 0 is computed as

$$\text{Var}_0(r) = \left( \sum_i \sum_j n_{ij} (R_i - \bar{R})^2 (C_j - \bar{C})^2 - s_{rc}^2 / n \right) / s_r s_c s_c$$

This expression for the variance is derived for multinomial sampling in a contingency table framework, and it differs from the form obtained under the assumption that both variables are continuous and normally distributed. For more information, see Brown and Benedetti (1977b).

PROC \texttt{FREQTAB} also provides an exact test for the Pearson correlation coefficient. You can request this test by specifying the \texttt{PCORR} option in the \texttt{EXACT} statement. See the section “Exact Statistics” on page 271 for more information.

**Spearman Rank Correlation Coefficient**

The Spearman correlation coefficient ($\rho_s$) is computed by using rank scores, which are defined in the section “Scores” on page 201. This measure is appropriate only when both variables lie on an ordinal scale. The range of the Spearman correlation is $-1 \leq \rho_s \leq 1$. The Spearman correlation coefficient is computed as

$$r_s = v / w$$
and its asymptotic variance is

$$\text{Var}(r_x) = \frac{1}{n^2 w^2} \sum_i \sum_j n_{ij} (z_{ij} - \bar{z})^2$$

where $R_i$ and $C_j$ are the row and column rank scores and

$$v = \sum_i \sum_j n_{ij} R(i) C(j)$$

$$w = \frac{1}{12} \sqrt{FG}$$

$$F = n^3 - \sum_i n_i^3$$

$$G = n^3 - \sum_j n_j^3$$

$$R(i) = R_i - n/2$$

$$C(j) = C_j - n/2$$

$$z = \frac{1}{n} \sum_i \sum_j n_{ij} z_{ij}$$

$$z_{ij} = w v_{ij} - v w_{ij}$$

$$v_{ij} = n \left( R(i) C(j) + \frac{1}{2} \sum_l n_{ij} C(l) + \frac{1}{2} \sum_k n_{kj} R(k) + \sum_l \sum_{k>i} n_{kl} C(l) + \sum_k \sum_{l>j} n_{kl} R(k) \right)$$

$$w_{ij} = -\frac{n}{96w} \left( F n_j^2 + G n_i^2 \right)$$

For more information, see Snedecor and Cochran (1989).

The variance under the null hypothesis that the correlation equals 0 is computed as

$$\text{Var}_0(r_x) = \frac{1}{n^2 w^2} \sum_i \sum_j n_{ij} (v_{ij} - \bar{v})^2$$

where

$$\bar{v} = \sum_i \sum_j n_{ij} v_{ij} / n$$
This expression for the variance is derived for multinomial sampling in a contingency table framework, and it differs from the form obtained under the assumption that both variables are continuous and normally distributed. For more information, see Brown and Benedetti (1977b).

PROC FREQTAB also provides an exact test for the Spearman correlation coefficient. You can request this test by specifying the SCORR option in the EXACT statement. For more information, see the section “Exact Statistics” on page 271.

Polychoric Correlation
When you specify the PLCORR option in the TABLES statement, PROC FREQTAB computes the polychoric correlation and its standard error. The polychoric correlation is based on the assumption that the two ordinal, categorical variables of the frequency table have an underlying bivariate normal distribution. The polychoric correlation coefficient is the maximum likelihood estimate of the product-moment correlation between the underlying normal variables. The range of the polychoric correlation is from –1 to 1. For 2 × 2 tables, the polychoric correlation is also known as the tetrachoric correlation (and it is labeled as such in the displayed output). See Drasgow (1986) for an overview of polychoric correlation coefficient.

Olsson (1979) gives the likelihood equations and the asymptotic standard errors for estimating the polychoric correlation. The underlying continuous variables relate to the observed crosstabulation table through thresholds, which define a range of numeric values that correspond to each categorical (table) level. PROC FREQTAB uses Olsson’s maximum likelihood method for simultaneous estimation of the polychoric correlation and the thresholds. (Olsson also presents a two-step method that estimates the thresholds first.)

PROC FREQTAB iteratively solves the likelihood equations by using a Newton-Raphson algorithm. The initial estimates of the thresholds are computed from the inverse of the normal distribution function at the cumulative marginal proportions of the table. Iterative computation of the polychoric correlation stops when the convergence measure falls below the convergence criterion or when the maximum number of iterations is reached, whichever occurs first. For parameter values that are less than 0.01, the procedure evaluates convergence by using the absolute difference instead of the relative difference. The PLCORR(CONVERGE=) option specifies the convergence criterion, which is 0.0001 by default. The PLCORR(MAXITER=) option specifies the maximum number of iterations, which is 20 by default.

If you specify the CL option in the TABLES statement, PROC FREQTAB provides confidence limits for the polychoric correlation. The confidence limits are computed as

\[ \hat{\rho} \pm (z_{\alpha/2} \times se(\hat{\rho})) \]

where \( \hat{\rho} \) is the estimate of the polychoric correlation, \( z_{\alpha/2} \) is the \( 100(1 - \alpha/2) \)th percentile of the standard normal distribution, and \( se(\hat{\rho}) \) is the standard error of the polychoric correlation estimate.

If you specify the PLCORR option in the TEST statement, PROC FREQTAB provides Wald and likelihood ratio tests of the null hypothesis that the polychoric correlation is 0. The Wald test statistic is computed as

\[ z = \frac{\hat{\rho}}{se(\hat{\rho})} \]

which has a standard normal distribution under the null hypothesis. PROC FREQTAB computes one-sided and two-sided \( p \)-values for the Wald test. When the test statistic \( z \) is greater than its null expected value of 0, PROC FREQTAB displays the right-sided \( p \)-value. When the test statistic is less than or equal to 0, PROC FREQTAB displays the left-sided \( p \)-value.
The likelihood ratio statistic for the polychoric correlation is computed as

\[ G^2 = -2 \log \left( \frac{L_0}{L_1} \right) \]

where \( L_0 \) is the value of the likelihood function (Olsson 1979) when the polychoric correlation is 0, and \( L_1 \) is the value of the likelihood function at the maximum (where all parameters are replaced by their maximum likelihood estimates). Under the null hypothesis, the likelihood ratio statistic has an asymptotic chi-square distribution with 1 degree of freedom.

**Lambda (Asymmetric)**

Asymmetric lambda, \( \lambda(C \mid R) \), is interpreted as the probable improvement in predicting the column variable \( Y \) given knowledge of the row variable \( X \). The range of asymmetric lambda is \( 0 \leq \lambda(C \mid R) \leq 1 \). Asymmetric lambda \( (C \mid R) \) is computed as

\[ \lambda(C \mid R) = \frac{\sum_i r_i - r}{n - r} \]

and its asymptotic variance is

\[ \text{Var}(\lambda(C \mid R)) = \frac{n - \sum_i r_i}{(n - r)^3} \left( \sum_i r_i + r - 2 \sum_i (r_i \mid l_i = l) \right) \]

where

\[ r_i = \max_j (n_{ij}) \]
\[ r = \max_j (n_{.j}) \]
\[ c_j = \max_i (n_{ij}) \]
\[ c = \max_i (n_{.i}) \]

The values of \( l_i \) and \( l \) are determined as follows. Denote by \( l_i \) the unique value of \( j \) such that \( r_i = n_{ij} \), and let \( l \) be the unique value of \( j \) such that \( r = n_{.j} \). Because of the uniqueness assumptions, ties in the frequencies or in the marginal totals must be broken in an arbitrary but consistent manner. In case of ties, \( l_i \) is defined as the smallest value of \( j \) such that \( r_i = n_{ij} \).

For those columns containing a cell \((i, j)\) for which \( n_{ij} = r_i = c_j \), \( cs_j \) records the row in which \( c_j \) is assumed to occur. Initially \( cs_j \) is set equal to 1 for all \( j \). Beginning with \( i = 1 \), if there is at least one value \( j \) such that \( n_{ij} = r_i = c_j \), and if \( cs_j = -1 \), \( l_i \) is defined to be the smallest such value of \( j \), and \( cs_j \) is set equal to \( i \). Otherwise, if \( n_{ij} = r_i, l_i \) is defined to be equal to \( l \). If neither condition is true, \( l_i \) is taken to be the smallest value of \( j \) such that \( n_{ij} = r_i \).

The formulas for lambda asymmetric \( (R \mid C) \) can be obtained by interchanging the indices.

For more information, see Goodman and Kruskal (1979).
**Lambda (Symmetric)**

The nondirectional lambda is the average of the two asymmetric lambdas, \( \lambda(C|R) \) and \( \lambda(R|C) \). Its range is \( 0 \leq \lambda \leq 1 \). Lambda symmetric is computed as

\[
\lambda = \frac{\sum_i r_i + \sum_j c_j - r - c}{2n - r - c} = \frac{w - v}{w}
\]

and its asymptotic variance is computed as

\[
\text{Var}(\lambda) = \frac{1}{w^4} \left( wvy - 2w^2(n - \sum_i \sum_j (n_{ij} | j = l_i, i = k_j) - 2v^2(n - n_{kl})) \right)
\]

where

\[
\begin{align*}
    r_i & = \max_j (n_{ij}) \\
    r & = \max_j (n_{.j}) \\
    c_j & = \max_i (n_{ij}) \\
    c & = \max_i (n_{i.}) \\
    w & = 2n - r - c \\
    v & = 2n - \sum_i r_i - \sum_j c_j \\
    x & = \sum_i (r_i | l_i = l) + \sum_j (c_j | k_j = k) + r_k + c_l \\
    y & = 8n - w - v - 2x
\end{align*}
\]

The definitions of \( l_i \) and \( l \) are given in the previous section. The values \( k_j \) and \( k \) are defined in a similar way for lambda asymmetric \( (R|C) \).

For more information, see Goodman and Kruskal (1979).

**Uncertainty Coefficients (Asymmetric)**

The uncertainty coefficient \( U(C|R) \) measures the proportion of uncertainty (entropy) in the column variable \( Y \) that is explained by the row variable \( X \). Its range is \( 0 \leq U(C|R) \leq 1 \). The uncertainty coefficient is computed as

\[
U(C|R) = (H(X) + H(Y) - H(XY)) / H(Y) = v/w
\]

and its asymptotic variance is

\[
\text{Var}(U(C|R)) = \frac{1}{n^2 w^4} \sum_i \sum_j n_{ij} (H(Y) \log \left( \frac{n_{ij}}{n_i} \right) + (H(X) - H(XY)) \log \left( \frac{n_{ij}}{n} \right))^2
\]
where

\[ v = H(X) + H(Y) - H(XY) \]

\[ w = H(Y) \]

\[ H(X) = -\sum_i \left( \frac{n_{i.}}{n} \right) \log \left( \frac{n_{i.}}{n} \right) \]

\[ H(Y) = -\sum_j \left( \frac{n_{.j}}{n} \right) \log \left( \frac{n_{.j}}{n} \right) \]

\[ H(XY) = -\sum_i \sum_j \left( \frac{n_{ij}}{n} \right) \ln \left( \frac{n_{ij}}{n} \right) \]

The formulas for the uncertainty coefficient \( U(R|C) \) can be obtained by interchanging the indices.

For more information, see Theil (1972, pp. 115–120) and Goodman and Kruskal (1979).

**Uncertainty Coefficient (Symmetric)**

The uncertainty coefficient \( U \) is the symmetric version of the two asymmetric uncertainty coefficients. Its range is \( 0 \leq U \leq 1 \). The uncertainty coefficient is computed as

\[ U = 2 \frac{(H(X) + H(Y) - H(XY))}{(H(X) + H(Y))} \]

and its asymptotic variance is

\[ \text{Var}(U) = 4 \sum_i \sum_j \frac{n_{ij}}{n} \left( H(XY) \log \left( \frac{n_{ij}}{n^2} \right) - (H(X) + H(Y)) \log \left( \frac{n_{ij}}{n} \right) \right)^2 \]

\[ \frac{n^2 (H(X) + H(Y))^4}{n^2 (H(X) + H(Y))^4} \]

where \( H(X) \), \( H(Y) \), and \( H(XY) \) are defined in the previous section. For more information, see Goodman and Kruskal (1979).

**Binomial Proportion**

If you specify the BINOMIAL option in the TABLES statement, PROC FREQTAB computes the binomial proportion for one-way tables. By default, this is the proportion of observations in the first variable level that appears in the output. (You can use the LEVEL= option to specify a different level for the proportion.) The binomial proportion is computed as

\[ \hat{p} = \frac{n_1}{n} \]

where \( n_1 \) is the frequency of the first (or designated) level and \( n \) is the total frequency of the one-way table. The standard error of the binomial proportion is computed as

\[ \text{se}(\hat{p}) = \sqrt{\hat{p} (1 - \hat{p}) / n} \]
Binomial Confidence Limits
PROC FREQTAB provides Wald and exact (Clopper-Pearson) confidence limits for the binomial proportion. You can also request the following binomial confidence limit types by specifying the BINOMIAL(CL=) option: Agresti-Coull, Blaker, Jeffreys, exact mid-$p$, likelihood ratio, logit, and Wilson (score). For more information, see Brown, Cai, and DasGupta (2001), Agresti and Coull (1998), and Newcombe (1998b), in addition to the references cited for each confidence limit type.

Wald Confidence Limits
Wald asymptotic confidence limits are based on the normal approximation to the binomial distribution. PROC FREQTAB computes the Wald confidence limits for the binomial proportion as

$$\hat{p} \pm (z_{\alpha/2} \times \text{se}(\hat{p}))$$

where $z_{\alpha/2}$ is the 100$(1 - \alpha/2)$th percentile of the standard normal distribution. The confidence level $\alpha$ is determined by the ALPHA= option; by default, ALPHA=0.05, which produces 95% confidence limits.

If you specify CL=WALD(CORRECT) or the CORRECT binomial-option, PROC FREQTAB includes a continuity correction of $1/(2n)$ in the Wald asymptotic confidence limits. The purpose of this correction is to adjust for the difference between the normal approximation and the discrete binomial distribution. See Fleiss, Levin, and Paik (2003) for more information. The continuity-corrected Wald confidence limits for the binomial proportion are computed as

$$\hat{p} \pm (z_{\alpha/2} \times \text{se}(\hat{p}) + (1/2n))$$

Exact (Clopper-Pearson) Confidence Limits
Exact (Clopper-Pearson) confidence limits for the binomial proportion are constructed by inverting the equal-tailed test based on the binomial distribution. This method is attributed to Clopper and Pearson (1934). The exact confidence limits $P_L$ and $P_U$ satisfy the following equations, for $n_1 = 1, 2, \ldots, n - 1$:

$$\sum_{x=n_1}^{n} \binom{n}{x} P_L^x (1 - P_L)^{n-x} = \alpha/2$$

$$\sum_{x=0}^{n_1} \binom{n}{x} P_U^x (1 - P_U)^{n-x} = \alpha/2$$

The lower confidence limit is 0 when $n_1 = 0$, and the upper confidence limit is 1 when $n_1 = n$.

PROC FREQTAB computes the exact (Clopper-Pearson) confidence limits by using the $F$ distribution as

$$P_L = \left(1 + \frac{n - n_1 + 1}{n_1 F(\alpha/2, 2n_1, 2(n - n_1 + 1))}\right)^{-1}$$

$$P_U = \left(1 + \frac{n - n_1}{(n_1 + 1) F(1 - \alpha/2, 2(n_1 + 1), 2(n - n_1))}\right)^{-1}$$

where $F(\alpha/2, b, c)$ is the $(\alpha/2)$th percentile of the $F$ distribution with $b$ and $c$ degrees of freedom. See Leemis and Trivedi (1996) for a derivation of this expression. Also see Collett (1991) for more information about exact binomial confidence limits.

Because this is a discrete problem, the confidence coefficient (coverage probability) of the exact (Clopper-Pearson) interval is not exactly $(1 - \alpha)$ but is at least $(1 - \alpha)$. Thus, this confidence interval is conservative. Unless the sample size is large, the actual coverage probability can be much larger than the target value. For more information about the performance of these confidence limits, see Agresti and Coull (1998), Brown, Cai, and DasGupta (2001), and Leemis and Trivedi (1996).
**Agresti-Coull Confidence Limits** If you specify the CL=AGRESTICOULL binomial-option, PROC FREQTAB computes Agresti-Coull confidence limits for the binomial proportion as

\[ \hat{p} \pm (z_{\alpha/2} \times \sqrt{\frac{\hat{p}(1-\hat{p})}{\tilde{n}}}) \]

where

\[
\begin{align*}
\tilde{n}_1 &= n_1 + \frac{z_{\alpha/2}^2}{2} \\
\tilde{n} &= n + \frac{z_{\alpha/2}^2}{2} \\
\hat{p} &= \frac{\tilde{n}_1}{\tilde{n}}
\end{align*}
\]

The Agresti-Coull confidence interval has the same general form as the standard Wald interval but uses \( \hat{p} \) in place of \( \hat{\hat{p}} \). For \( \alpha = 0.05 \), the value of \( z_{\alpha/2} \) is close to 2, and this interval is the “add 2 successes and 2 failures” adjusted Wald interval of Agresti and Coull (1998).

**Blaker Confidence Limits** If you specify the CL=BLAKER binomial-option, PROC FREQTAB computes Blaker confidence limits for the binomial proportion, which are constructed by inverting the two-sided exact Blaker test (Blaker 2000). The 100(1-\( \alpha \))% Blaker confidence interval consists of all values of the proportion \( p_0 \) for which the test statistic \( B(p_0, n_1) \) falls in the acceptance region,

\[ \{p_0 : B(p_0, n_1) > \alpha\} \]

where

\[ B(p_0, n_1) = \text{Prob}( \gamma(p_0, X) \leq \gamma(p_0, n_1) \mid p_0) \]

\[ \gamma(p_0, n_1) = \min( \text{Prob}( X \geq n_1 \mid p_0), \text{Prob}( X \leq n_1 \mid p_0) ) \]

and \( X \) is a binomial random variable. For more information, see Blaker (2000).

**Jeffreys Confidence Limits** If you specify the CL=JEFFREYS binomial-option, PROC FREQTAB computes Jeffreys confidence limits for the binomial proportion as

\[ \left( \beta(\alpha/2, n_1 + 1/2, n - n_1 + 1/2), \beta(1 - \alpha/2, n_1 + 1/2, n - n_1 + 1/2) \right) \]

where \( \beta(\alpha, b, c) \) is the \( \alpha \)th percentile of the beta distribution with shape parameters \( b \) and \( c \). The lower confidence limit is set to 0 when \( n_1 = 0 \), and the upper confidence limit is set to 1 when \( n_1 = n \). This is an equal-tailed interval based on the noninformative Jeffreys prior for a binomial proportion. For more information, see Brown, Cai, and DasGupta (2001). For information about using beta priors for inference on the binomial proportion, see Berger (1985).

**Likelihood Ratio Confidence Limits** If you specify the CL=LIKELIHOODRATIO binomial-option, PROC FREQTAB computes likelihood ratio confidence limits for the binomial proportion by inverting the likelihood ratio test. The likelihood ratio test statistic for the null hypothesis that the proportion equals \( p_0 \) can be expressed as

\[ L(p_0) = -2(n_1 \log(\hat{p}/p_0) + (n - n_1) \log((1-\hat{p})/(1-p_0))) \]
The 100(1 – α)% likelihood ratio confidence interval consists of all values of \( p_0 \) for which the test statistic \( L(p_0) \) falls in the acceptance region,

\[
\{ p_0 : L(p_0) < \chi^2_{1, \alpha} \}
\]

where \( \chi^2_{1, \alpha} \) is the 100(1 – α)th percentile of the chi-square distribution with 1 degree of freedom. PROC FREQTAB finds the confidence limits by iterative computation. For more information, see Fleiss, Levin, and Paik (2003), Brown, Cai, and DasGupta (2001), Agresti (2013), and Newcombe (1998b).

**Logit Confidence Limits** If you specify the CL=LOGIT binomial-option, PROC FREQTAB computes logit confidence limits for the binomial proportion, which are based on the logit transformation \( Y = \log(\hat{p}/(1-\hat{p})) \).

Approximate confidence limits for \( Y \) are computed as

\[
Y_L = \log(\hat{p}/(1-\hat{p})) - z_{\alpha/2} \sqrt{n/(n_1(n-n_1))}
\]

\[
Y_U = \log(\hat{p}/(1-\hat{p})) + z_{\alpha/2} \sqrt{n/(n_1(n-n_1))}
\]

The confidence limits for \( Y \) are inverted to produce 100(1 – α)% logit confidence limits \( P_L \) and \( P_U \) for the binomial proportion \( p \) as

\[
P_L = \exp(Y_L/(1 + \exp(Y_L)))
\]

\[
P_U = \exp(Y_U/(1 + \exp(Y_U)))
\]

For more information, see Brown, Cai, and DasGupta (2001) and Korn and Graubard (1998).

**Mid-p Confidence Limits** If you specify the CL=MIDP binomial-option, PROC FREQTAB computes exact mid-p confidence limits for the binomial proportion by inverting two one-sided binomial tests that include mid-p tail areas. The mid-p approach replaces the probability of the observed frequency by half of that probability in the Clopper-Pearson sum, which is described in the section “Exact (Clopper-Pearson) Confidence Limits” on page 218. The exact mid-p confidence limits \( P_L \) and \( P_U \) are the solutions to the equations

\[
\sum_{x=n_1+1}^{n} \binom{n}{x} P_L^x (1-P_L)^{n-x} + \frac{1}{2} \binom{n}{n_1} P_L^{n_1} (1-P_L)^{n-n_1} = \alpha/2
\]

\[
\sum_{x=0}^{n_1-1} \binom{n}{x} P_U^x (1-P_U)^{n-x} + \frac{1}{2} \binom{n}{n_1} P_U^{n_1} (1-P_U)^{n-n_1} = \alpha/2
\]

For more information, see Agresti and Gottard (2007), Agresti (2013), Newcombe (1998b), and Brown, Cai, and DasGupta (2001).

**Wilson (Score) Confidence Limits** If you specify the CL=WILSON binomial-option, PROC FREQTAB computes Wilson confidence limits for the binomial proportion. These are also known as score confidence limits (Wilson 1927). The confidence limits are based on inverting the normal test that uses the null proportion in the variance (the score test). Wilson confidence limits are the roots of

\[
|p - \hat{p}| = z_{\alpha/2} \sqrt{p(1-p)/n}
\]
and are computed as
\[
\left( \hat{p} + \frac{z_{\alpha/2}^2}{2n} \pm z_{\alpha/2} \sqrt{\left( \hat{p}(1-\hat{p}) + \frac{z_{\alpha/2}^2}{4n} \right) / n} \right) / \left( 1 + \frac{z_{\alpha/2}^2}{n} \right)
\]

If you specify CL=WILSON(CORRECT) or the CORRECT binomial-option, PROC FREQTAB provides continuity-corrected Wilson confidence limits, which are computed as the roots of
\[
|p - \hat{p}| - 1/2n = z_{\alpha/2} \sqrt{p(1-p)/n}
\]

The Wilson interval has been shown to have better performance than the Wald interval and the exact (Clopper-Pearson) interval. For more information, see Agresti and Coull (1998), Brown, Cai, and DasGupta (2001), and Newcombe (1998b).

**Binomial Tests**

The BINOMIAL option provides an asymptotic equality test for the binomial proportion by default. You can also specify binomial-options to request tests of noninferiority, superiority, and equivalence for the binomial proportion. If you specify the BINOMIAL option in the EXACT statement, PROC FREQTAB also computes exact p-values for the tests that you request with the binomial-options.

**Equality Test**

PROC FREQTAB computes an asymptotic test of the hypothesis that the binomial proportion equals \( p_0 \), where you can specify the value of \( p_0 \) with the P= binomial-option. If you do not specify a null value with P=, PROC FREQTAB uses \( p_0 = 0.5 \) by default. The binomial test statistic is computed as
\[
z = (\hat{p} - p_0)/se
\]

By default, the standard error is based on the null hypothesis proportion as
\[
se = \sqrt{p_0(1-p_0)/n}
\]

If you specify the VAR=SAMPLE binomial-option, the standard error is computed from the sample proportion as
\[
se = \sqrt{\hat{p}(1-\hat{p})/n}
\]

If you specify the CORRECT binomial-option, PROC FREQTAB includes a continuity correction in the asymptotic test statistic, towards adjusting for the difference between the normal approximation and the discrete binomial distribution. For more information, see Fleiss, Levin, and Paik (2003). The continuity correction of \( (1/2n) \) is subtracted from the numerator of the test statistic if \((\hat{p} - p_0)\) is positive; otherwise, the continuity correction is added to the numerator.

PROC FREQTAB computes one-sided and two-sided p-values for this test. When the test statistic \( z \) is greater than 0 (its expected value under the null hypothesis), PROC FREQTAB computes the right-sided p-value, which is the probability of a larger value of the statistic occurring under the null hypothesis. A small right-sided p-value supports the alternative hypothesis that the true value of the proportion is greater than \( p_0 \). When the test statistic is less than or equal to 0, PROC FREQTAB computes the left-sided p-value, which is the probability of a smaller value of the statistic occurring under the null hypothesis. A small left-sided
Chapter 5: The FREQTABLE Procedure

The p-value supports the alternative hypothesis that the true value of the proportion is less than \( p_0 \). The one-sided p-value \( P_1 \) can be expressed as

\[
P_1 = \begin{cases} 
\text{Prob}(Z > z) & \text{if } z > 0 \\
\text{Prob}(Z < z) & \text{if } z \leq 0 
\end{cases}
\]

where \( Z \) has a standard normal distribution. The two-sided p-value is computed as \( P_2 = 2 \times P_1 \).

If you specify the BINOMIAL option in the EXACT statement, PROC FREQTABLE also computes an exact test of the null hypothesis \( H_0: p = p_0 \). To compute the exact test, PROC FREQTABLE uses the binomial probability function,

\[
\text{Prob}(X = x \mid p_0) = \binom{n}{x} p_0^x (1 - p_0)^{(n-x)} \quad \text{for } x = 0, 1, \ldots, n
\]

where the variable \( X \) has a binomial distribution with parameters \( n \) and \( p_0 \). To compute the left-sided p-value, \( \text{Prob}(X \leq n_1) \), PROC FREQTABLE sums the binomial probabilities over \( x \) from 0 to \( n_1 \). To compute the right-sided p-value, \( \text{Prob}(X \geq n_1) \), PROC FREQTABLE sums the binomial probabilities over \( x \) from \( n_1 \) to \( n \). The exact one-sided p-value is the minimum of the left-sided and right-sided p-values,

\[
P_1 = \min \left( \text{Prob}(X \leq n_1 \mid p_0), \text{Prob}(X \geq n_1 \mid p_0) \right)
\]

and the exact two-sided p-value is computed as \( P_2 = 2 \times P_1 \).

**Noninferiority Test** If you specify the NONINF binomial-option, PROC FREQTABLE provides a noninferiority test for the binomial proportion. The null hypothesis for the noninferiority test is

\[
H_0: p - p_0 \leq -\delta
\]

versus the alternative

\[
H_a: p - p_0 > -\delta
\]

where \( \delta \) is the noninferiority margin and \( p_0 \) is the null proportion. Rejection of the null hypothesis indicates that the binomial proportion is not inferior to the null value. See Chow, Shao, and Wang (2003) for more information.

You can specify the value of \( \delta \) with the MARGIN= binomial-option, and you can specify \( p_0 \) with the P= binomial-option. By default, \( \delta = 0.2 \) and \( p_0 = 0.5 \).

PROC FREQTABLE provides an asymptotic Wald test for noninferiority. The test statistic is computed as

\[
z = (\hat{p} - p_0^*) / \text{se}
\]

where \( p_0^* \) is the noninferiority limit,

\[
p_0^* = p_0 - \delta
\]

By default, the standard error is computed from the sample proportion as

\[
\text{se} = \sqrt{\hat{p}(1-\hat{p})/n}
\]
If you specify the VAR=NULL binomial-option, the standard error is based on the noninferiority limit (determined by the null proportion and the margin) as

$$se = \sqrt{p_0^*(1 - p_0^*)/n}$$

If you specify the CORRECT binomial-option, PROC FREQTAB includes a continuity correction in the asymptotic test statistic $z$. The continuity correction of $(1/2n)$ is subtracted from the numerator of the test statistic if $(\hat{p} - p_0^*)$ is positive; otherwise, the continuity correction is added to the numerator.

The $p$-value for the noninferiority test is

$$P_z = \text{Prob}(Z > z)$$

where $Z$ has a standard normal distribution.

As part of the noninferiority analysis, PROC FREQTAB provides asymptotic Wald confidence limits for the binomial proportion. These confidence limits are computed as described in the section “Wald Confidence Limits” on page 218 but use the same standard error (VAR=NULL or VAR=SAMPLE) as the noninferiority test statistic $z$. The confidence coefficient is $100(1 - 2\alpha)%$ (Schuirmann 1999). By default, if you do not specify the ALPHA= option, the noninferiority confidence limits are 90% confidence limits. You can compare the confidence limits to the noninferiority limit, $p_0^* = p_0 - \delta$.

If you specify the BINOMIAL option in the EXACT statement, PROC FREQTAB provides an exact noninferiority test for the binomial proportion. The exact $p$-value is computed by using the binomial probability function with parameters $p_0^*$ and $n$,

$$P_x = \sum_{k=n_1}^{k=n} \binom{n}{k} (p_0^*)^k (1 - p_0^*)^{n-k}$$

For more information, see Chow, Shao, and Wang (2003, p. 116). If you request exact binomial statistics, PROC FREQTAB also includes exact (Clopper-Pearson) confidence limits for the binomial proportion in the equivalence analysis display. For more information, see the section “Exact (Clopper-Pearson) Confidence Limits” on page 218.

**Superiority Test** If you specify the SUP binomial-option, PROC FREQTAB provides a superiority test for the binomial proportion. The null hypothesis for the superiority test is

$$H_0: p - p_0 \leq \delta$$

versus the alternative

$$H_a: p - p_0 > \delta$$

where $\delta$ is the superiority margin and $p_0$ is the null proportion. Rejection of the null hypothesis indicates that the binomial proportion is superior to the null value. You can specify the value of $\delta$ with the MARGIN= binomial-option, and you can specify the value of $p_0$ with the P= binomial-option. By default, $\delta = 0.2$ and $p_0 = 0.5$.

The superiority analysis is identical to the noninferiority analysis but uses a positive value of the margin $\delta$ in the null hypothesis. The superiority limit equals $p_0 + \delta$. The superiority computations follow those in the section “Noninferiority Test” on page 222 but replace $-\delta$ with $\delta$. See Chow, Shao, and Wang (2003) for more information.
Equivalence Test  If you specify the EQUIV binomial-option, PROC FREQTAB provides an equivalence test for the binomial proportion. The null hypothesis for the equivalence test is

\[ H_0: p - p_0 \leq \delta_L \text{ or } p - p_0 \geq \delta_U \]

versus the alternative

\[ H_a: \delta_L < p - p_0 < \delta_U \]

where \( \delta_L \) is the lower margin, \( \delta_U \) is the upper margin, and \( p_0 \) is the null proportion. Rejection of the null hypothesis indicates that the binomial proportion is equivalent to the null value. See Chow, Shao, and Wang (2003) for more information.

You can specify the value of the margins \( \delta_L \) and \( \delta_U \) with the MARGIN= binomial-option. If you do not specify MARGIN=, PROC FREQTAB uses lower and upper margins of –0.2 and 0.2 by default. If you specify a single margin value \( \delta \), PROC FREQTAB uses lower and upper margins of –\( \delta \) and \( \delta \). You can specify the null proportion \( p_0 \) with the P= binomial-option. By default, \( p_0 = 0.5 \).

PROC FREQTAB computes two one-sided tests (TOST) for equivalence analysis (Schuirmann 1987). The TOST approach includes a right-sided test for the lower margin and a left-sided test for the upper margin. The overall \( p \)-value is taken to be the larger of the two \( p \)-values from the lower and upper tests.

For the lower margin, the asymptotic Wald test statistic is computed as

\[ z_L = \frac{\hat{p} - p_L^*}{se} \]

where the lower equivalence limit is

\[ p_L^* = p_0 + \delta_L \]

By default, the standard error is computed from the sample proportion as

\[ se = \sqrt{\hat{p}(1 - \hat{p})/n} \]

If you specify the VAR=NULL binomial-option, the standard error is based on the lower equivalence limit (determined by the null proportion and the lower margin) as

\[ se = \sqrt{p_L^*(1 - p_L^*)/n} \]

If you specify the CORRECT binomial-option, PROC FREQTAB includes a continuity correction in the asymptotic test statistic \( z_L \). The continuity correction of \( (1/2n) \) is subtracted from the numerator of the test statistic \( (\hat{p} - p_L^*) \) if the numerator is positive; otherwise, the continuity correction is added to the numerator.

The \( p \)-value for the lower margin test is

\[ P_{z,L} = \text{Prob}(Z > z_L) \]

The asymptotic test for the upper margin is computed similarly. The Wald test statistic is

\[ z_U = \frac{\hat{p} - p_U^*}{se} \]

where the upper equivalence limit is

\[ p_U^* = p_0 + \delta_U \]
By default, the standard error is computed from the sample proportion. If you specify the VAR=NULL binomial-option, the standard error is based on the upper equivalence limit as

$$se = \sqrt{p_U^*(1 - p_U^*)/n}$$

If you specify the CORRECT binomial-option, PROC FREQTAB includes a continuity correction of \((1/2n)\) in the asymptotic test statistic \(z_U\).

The \(p\)-value for the upper margin test is

$$P_{z,U} = \text{Prob}(Z < z_U)$$

Based on the two one-sided tests (TOST), the overall \(p\)-value for the test of equivalence equals the larger \(p\)-value from the lower and upper margin tests, which can be expressed as

$$P_z = \max(P_{z,L}, P_{z,U})$$

As part of the equivalence analysis, PROC FREQTAB provides asymptotic Wald confidence limits for the binomial proportion. These confidence limits are computed as described in the section “Wald Confidence Limits” on page 218, but use the same standard error (VAR=NULL or VAR=SAMPLE) as the equivalence test statistics and have a confidence coefficient of 100(1 – 2\(\alpha\))% (Schuirmann 1999). By default, if you do not specify the ALPHA= option, the equivalence confidence limits are 90% limits. If you specify VAR=NULL, separate standard errors are computed for the lower and upper margin tests, each based on the null proportion and the corresponding (lower or upper) margin. The confidence limits are computed by using the maximum of these two standard errors. You can compare the confidence limits to the equivalence limits, \((p_0 + \delta_L, p_0 + \delta_U)\).

If you specify the BINOMIAL option in the EXACT statement, PROC FREQTAB also provides an exact equivalence test by using two one-sided exact tests (TOST). The procedure computes lower and upper margin exact tests by using the binomial probability function as described in the section “Noninferiority Test” on page 222. The overall exact \(p\)-value for the equivalence test is taken to be the larger \(p\)-value from the lower and upper margin exact tests. If you request exact statistics, PROC FREQTAB also includes exact (Clopper-Pearson) confidence limits in the equivalence analysis display. The confidence coefficient is 100(1 – 2\(\alpha\))% (Schuirmann 1999). For more information, see the section “Exact (Clopper-Pearson) Confidence Limits” on page 218.

**Sensitivity and Specificity**

The SENSPEC option in the TABLES statement provides estimates of sensitivity, specificity, positive predictive value, and negative predictive value for 2 × 2 tables. These measures are conditional (row and column) proportions in the 2 × 2 table crosstabulation. In sensitivity analysis, the row variable might represent the presence or absence of a condition, and the column variable might represent a positive or negative diagnostic test result. For more information, see Fleiss, Levin, and Paik (2003).

Sensitivity is defined as the column proportion for table cell (1,1), which is the ratio of the total in table cell (1,1) to the total in column 1 of the 2 × 2 table. The estimate of sensitivity is computed as

$$SN = n_{11} / n_{-1}$$

Specificity is defined as the column proportion for table cell (2,2), which is the ratio of the total in table cell (2,2) to the total in column 2. The estimate of specificity is computed as

$$SP = n_{22} / n_{-2}$$
The positive predictive value is the row proportion for table cell (1,1), which is computed as

$$PPV = \frac{n_{11}}{n_1}.$$  

The negative predictive value is the row proportion for table cell (2,2), which is computed as

$$NPV = \frac{n_{22}}{n_2}.$$  

The “Sensitivity and Specificity” table provides the estimates together with their standard errors and Wald confidence limits. PROC FREQTAB computes the standard errors and Wald confidence limits for these proportions as described in the section “Risks and Risk Differences” on page 226. The value of the confidence coefficient $\alpha$ is determined by the ALPHA= option; by default, ALPHA=0.05, which produces 95% confidence limits.

**Risks and Risk Differences**

The RISKDIFF option in the TABLES statement provides estimates of risks (binomial proportions) and risk differences for $2 \times 2$ tables. This analysis might be appropriate when comparing the proportion of some characteristic for two groups, where row 1 and row 2 correspond to the two groups, and the columns correspond to two possible characteristics or outcomes. For example, the row variable might be a treatment or dose, and the column variable might be the response. For more information, see Collett (1991); Fleiss, Levin, and Paik (2003); Stokes, Davis, and Koch (2012).

Let the frequencies of the $2 \times 2$ table be represented as follows:

<table>
<thead>
<tr>
<th></th>
<th>Column 1</th>
<th>Column 2</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Row 1</td>
<td>$n_{11}$</td>
<td>$n_{12}$</td>
<td>$n_1$</td>
</tr>
<tr>
<td>Row 2</td>
<td>$n_{21}$</td>
<td>$n_{22}$</td>
<td>$n_2$</td>
</tr>
<tr>
<td>Total</td>
<td>$n_1$</td>
<td>$n_2$</td>
<td>$n$</td>
</tr>
</tbody>
</table>

By default when you specify the RISKDIFF option, PROC FREQTAB provides estimates of the row 1 risk (proportion), the row 2 risk, the overall risk, and the risk difference for column 1 and for column 2 of the $2 \times 2$ table. The risk difference is defined as the row 1 risk minus the row 2 risk. The risks are binomial proportions of their rows (row 1, row 2, or overall), and the computation of their standard errors and Wald confidence limits follow the binomial proportion computations, which are described in the section “Binomial Proportion” on page 217.

The column 1 risk for row 1 is the proportion of row 1 observations classified in column 1,

$$\hat{p}_1 = \frac{n_{11}}{n_1},$$

which estimates the conditional probability of the column 1 response, given the first level of the row variable. The column 1 risk for row 2 is the proportion of row 2 observations classified in column 1,

$$\hat{p}_2 = \frac{n_{21}}{n_2}.$$  

The overall column 1 risk is the proportion of all observations classified in column 1,

$$\hat{p} = \frac{n_1}{n}.$$
The column 1 risk difference compares the risks for the two rows, and it is computed as the column 1 risk for row 1 minus the column 1 risk for row 2,

$$\hat{d} = \hat{p}_1 - \hat{p}_2$$

The standard error of the column 1 risk for row \(i\) is computed as

$$se(\hat{p}_i) = \sqrt{\hat{p}_i (1 - \hat{p}_i) / n_i}.$$  

The standard error of the overall column 1 risk is computed as

$$se(\hat{p}) = \sqrt{\hat{p} (1 - \hat{p}) / n}$$

Where the two rows represent independent binomial samples, the standard error of the column 1 risk difference is computed as

$$se(\hat{d}) = \sqrt{\hat{p}_1 (1 - \hat{p}_1)/n_1} + \hat{p}_2 (1 - \hat{p}_2)/n_2.$$  

The computations are similar for the column 2 risks and risk difference.

**Confidence Limits**

By default, the RISKDIFF option provides Wald asymptotic confidence limits for the risks (row 1, row 2, and overall) and the risk difference. By default, the RISKDIFF option also provides exact (Clopper-Pearson) confidence limits for the risks. You can suppress the display of this information by specifying the NORISKS riskdiff-option. You can specify riskdiff-options to request tests and other types of confidence limits for the risk difference. For more information, see the sections “Confidence Limits for the Risk Difference” on page 228 and “Risk Difference Tests” on page 232.

The risks are equivalent to the binomial proportions of their corresponding rows. This section describes the Wald confidence limits that are provided by default when you specify the RISKDIFF option. The BINOMIAL option provides additional confidence limit types and tests for risks (binomial proportions). For more information, see the sections “Binomial Confidence Limits” on page 218 and “Binomial Tests” on page 221.

The Wald confidence limits are based on the normal approximation to the binomial distribution. PROC FREQTAB computes the Wald confidence limits for the risks and risk differences as

$$\text{Est} \pm (z_{\alpha/2} \times se(\text{Est}))$$

where Est is the estimate, \(z_{\alpha/2}\) is the 100(1 - \(\alpha/2\))th percentile of the standard normal distribution, and \(se(\text{Est})\) is the standard error of the estimate. The confidence level \(\alpha\) is determined by the value of the ALPHA= option; by default, ALPHA=0.05, which produces 95% confidence limits.

If you specify the CORRECT riskdiff-option, PROC FREQTAB includes continuity corrections in the Wald confidence limits for the risks and risk differences. The purpose of a continuity correction is to adjust for the difference between the normal approximation and the binomial distribution, which is discrete. See Fleiss, Levin, and Paik (2003) for more information. The continuity-corrected Wald confidence limits are computed as

$$\text{Est} \pm (z_{\alpha/2} \times se(\text{Est}) + c)$$
where \( c \) is the continuity correction. For the row 1 risk, \( c = (1/2n_1) \); for the row 2 risk, \( c = (1/2n_2) \); for the overall risk, \( c = (1/2n) \); and for the risk difference, \( c = ((1/n_1 + 1/n_2)/2) \). The column 1 and column 2 risks use the same continuity correction.

By default when you specify the RISKDIFF option, PROC FREQTAB also provides exact (Clopper-Pearson) confidence limits for the column 1, column 2, and overall risks. These confidence limits are constructed by inverting the equal-tailed test that is based on the binomial distribution. For more information, see the section “Exact (Clopper-Pearson) Confidence Limits” on page 218.

Confidence Limits for the Risk Difference

PROC FREQTAB provides the following confidence limit types for the risk difference: Agresti-Caffo, exact unconditional, Hauck-Anderson, Miettinen-Nurmininen (score), Newcombe (hybrid-score), and Wald confidence limits. Continuity-corrected forms of Newcombe and Wald confidence limits are also available.

The confidence coefficient for the confidence limits produced by the CL= riskdiff-option is \( 100(1 - \alpha)\% \), where the value of \( \alpha \) is determined by the ALPHA= option. By default, ALPHA=0.05, which produces 95% confidence limits. This differs from the test-based confidence limits that are provided with the equivalence, noninferiority, and superiority tests, which have a confidence coefficient of \( 100(1 - 2\alpha)\% \) (Schuirmann 1999). For more information, see the section “Risk Difference Tests” on page 232.

Agresti-Caffo Confidence Limits

Agresti-Caffo confidence limits for the risk difference are computed as

\[
\hat{d} \pm (z_{\alpha/2} \times se(\hat{d}))
\]

where \( \hat{d} = \hat{\rho}_1 - \hat{\rho}_2 \), \( \hat{\rho}_i = (n_{i1} + 1)/(n_i + 2) \),

\[
se(\hat{d}) = \sqrt{\hat{\rho}_1(1 - \hat{\rho}_2)/(n_1 + 2) + \hat{\rho}_2(1 - \hat{\rho}_2)/(n_2 + 2)}
\]

and \( z_{\alpha/2} \) is the \( 100(1 - \alpha/2) \)th percentile of the standard normal distribution.

The Agresti-Caffo interval adjusts the Wald interval for the risk difference by adding a pseudo-observation of each type (success and failure) to each sample. See Agresti and Caffo (2000) and Agresti and Coull (1998) for more information.

Hauck-Anderson Confidence Limits

Hauck-Anderson confidence limits for the risk difference are computed as

\[
\hat{d} \pm (c + z_{\alpha/2} \times se(\hat{d}))
\]

where \( \hat{d} = \hat{\rho}_1 - \hat{\rho}_2 \) and \( z_{\alpha/2} \) is the \( 100(1 - \alpha/2) \)th percentile of the standard normal distribution. The standard error is computed from the sample proportions as

\[
se(\hat{d}) = \sqrt{\hat{\rho}_1(1 - \hat{\rho}_1)/(n_1 - 1) + \hat{\rho}_2(1 - \hat{\rho}_2)/(n_2 - 1)}
\]

The Hauck-Anderson continuity correction \( c \) is computed as

\[
c = 1 / (2 \min(n_1, n_2))
\]

For more information, see Hauck and Anderson (1986).
The subsection “Hauck-Anderson Test” in the section “Noninferiority Tests” on page 232 describes the corresponding noninferiority test.

**Miettinen-Nurminen (Score) Confidence Limits**

Miettinen-Nurminen (score) confidence limits for the risk difference (Miettinen and Nurminen 1985) are computed by inverting score tests for the risk difference. A score-based test statistic for the null hypothesis that the risk difference equals \( \delta \) can be expressed as

\[
T(\delta) = (\hat{d} - \delta) / \sqrt{\text{Var}(\delta)}
\]

where \( \hat{d} \) is the observed value of the risk difference (\( \hat{p}_1 - \hat{p}_2 \)),

\[
\text{Var}(\delta) = (n/(n-1)) \left((\hat{p}_1(\delta)(1 - \hat{p}_1(\delta))/n_1 + \hat{p}_2(\delta)(1 - \hat{p}_2(\delta))/n_2\right)
\]

and \( \hat{p}_1(\delta) \) and \( \hat{p}_2(\delta) \) are the maximum likelihood estimates of the row 1 and row 2 risks (proportions) under the restriction that the risk difference is \( \delta \). For more information, see Miettinen and Nurminen (1985, pp. 215–216) and Miettinen (1985, chapter 12).

The 100(1 - \( \alpha \))% confidence interval for the risk difference consists of all values of \( \delta \) for which the score test statistic \( T(\delta) \) falls in the acceptance region,

\[
\{\delta : T(\delta) < z_{\alpha/2}\}
\]

where \( z_{\alpha/2} \) is the 100(1 - \( \alpha/2 \))th percentile of the standard normal distribution. PROC FREQTAB finds the confidence limits by iterative computation, which stops when the iteration increment falls below the convergence criterion or when the maximum number of iterations is reached, whichever occurs first. By default, the convergence criterion is 0.00000001 and the maximum number of iterations is 100.

By default, the Miettinen-Nurminen confidence limits include the bias correction factor \( n/(n-1) \) in the computation of \( \text{Var}(\delta) \) (Miettinen and Nurminen 1985, p. 216). For more information, see Newcombe and Nurminen (2011). If you specify the CL=MN(CORRECT=NO) riskdiff-option, PROC FREQTAB does not include the bias correction factor in this computation (Mee 1984). See also Agresti (2002, p. 77). The uncorrected confidence limits are labeled as “Miettinen-Nurminen-Mee” confidence limits in the displayed output.

The maximum likelihood estimates of \( p_1 \) and \( p_2 \), subject to the constraint that the risk difference is \( \delta \), are computed as

\[
\hat{p}_1 = 2u \cos(w) - b/3a \quad \text{and} \quad \hat{p}_2 = \hat{p}_1 - \delta
\]

where

\[
w = (\pi + \cos^{-1}(v/u^3))/3
\]

\[
v = b^3/(3a)^3 - bc/6a^2 + d/2a
\]

\[
u = \text{sign}(v) \sqrt{b^2/(3a)^2 - c/3a}
\]

\[
a = 1 + \theta
\]

\[
b = -(1 + \theta + \hat{p}_1 + \theta \hat{p}_2 + \delta(\theta + 2))
\]

\[
c = \delta^2 + \delta(2\hat{p}_1 + \theta + 1) + \hat{p}_1 + \theta \hat{p}_2
\]

\[
d = -\hat{p}_1\delta(1 + \delta)
\]

\[
\theta = n_2/n_1.
\]
For more information, see Farrington and Manning (1990, p. 1453).

**Newcombe Confidence Limits**

Newcombe (hybrid-score) confidence limits for the risk difference are constructed from the Wilson score confidence limits for each of the two individual proportions. The confidence limits for the individual proportions are used in the standard error terms of the Wald confidence limits for the proportion difference. See Newcombe (1998a) and Barker et al. (2001) for more information.

Wilson score confidence limits for \( p_1 \) and \( p_2 \) are the roots of
\[
|p_i - \hat{p}_i| = z_{a/2}\sqrt{\frac{\hat{p}_i(1 - \hat{p}_i)}{n_i}}.
\]
for \( i = 1, 2 \). The confidence limits are computed as
\[
\left( \hat{p}_i + \frac{z^2_{a/2}}{2n_i} \pm z_{a/2}\sqrt{\left(\hat{p}_i(1 - \hat{p}_i) + \frac{z^2_{a/2}}{4n_i}\right) / n_i} \right) / \left(1 + \frac{z^2_{a/2}}{n_i}\right).
\]

For more information, see the section “Wilson (Score) Confidence Limits” on page 220.

Denote the lower and upper Wilson score confidence limits for \( p_1 \) as \( L_1 \) and \( U_1 \), and denote the lower and upper confidence limits for \( p_2 \) as \( L_2 \) and \( U_2 \). The Newcombe confidence limits for the proportion difference \( d = p_1 - p_2 \) are computed as
\[
d_L = (\hat{p}_1 - \hat{p}_2) - \sqrt{(\hat{p}_1 - L_1)^2 + (U_2 - \hat{p}_2)^2}
\]
\[
d_U = (\hat{p}_1 - \hat{p}_2) + \sqrt{(U_1 - \hat{p}_1)^2 + (\hat{p}_2 - L_2)^2}
\]

If you specify the CORRECT riskdiff-option, PROC FREQTAB provides continuity-corrected Newcombe confidence limits. By including a continuity correction of \( 1/2n_i \), the Wilson score confidence limits for the individual proportions are computed as the roots of
\[
|p_i - \hat{p}_i| - 1/2n_i = z_{a/2}\sqrt{\frac{\hat{p}_i(1 - \hat{p}_i)}{n_i}}.
\]

The continuity-corrected confidence limits for the individual proportions are then used to compute the proportion difference confidence limits \( d_L \) and \( d_U \).

**Wald Confidence Limits**

Wald confidence limits for the risk difference are computed as
\[
\hat{d} \pm ( z_{a/2} \times se(\hat{d}) )
\]
where \( \hat{d} = \hat{p}_1 - \hat{p}_2 \), \( z_{a/2} \) is the \( 100(1 - \alpha/2) \)th percentile of the standard normal distribution, and the standard error is computed from the sample proportions as
\[
se(\hat{d}) = \sqrt{\hat{p}_1(1 - \hat{p}_1)/n_1} + \hat{p}_2(1 - \hat{p}_2)/n_2.
\]

If you specify the CORRECT riskdiff-option, the Wald confidence limits include a continuity correction \( cc \),
\[
\hat{d} \pm ( c + z_{a/2} \times se(\hat{d}) )
\]
where \(c = (1/n_1 + 1/n_2)/2\).

The subsection “Wald Test” in the section “Noninferiority Tests” on page 232 describes the corresponding noninferiority test.

**Exact Unconditional Confidence Limits**

If you specify the RISKDIFF option in the EXACT statement, PROC FREQTAB provides exact unconditional confidence limits for the risk difference \((d = p_1 - p_2)\). The exact unconditional approach fixes the row margins of the \(2 \times 2\) table and eliminates the nuisance parameter \(p_2\) by using the maximum \(p\)-value (worst-case scenario) over all possible values of \(p_2\) (Santner and Snell 1980). The conditional approach, which is described in the section “Exact Statistics” on page 271, does not apply to the risk difference because of the nuisance parameter (Agresti 1992).

By default, PROC FREQTAB computes the confidence limits by the tail method, which inverts two separate one-sided exact tests of the risk difference, where the tests are based on the score statistic (Chan and Zhang 1999). The size of each one-sided exact test is at most \(\alpha/2\), and the confidence coefficient is at least \((1 - \alpha)\). If you specify the RISKDIFF(METHOD=NOSCORE) option in the EXACT statement, PROC FREQTAB computes the confidence limits by inverting two separate one-sided exact tests that are based on the unstandardized risk difference. If you specify the RISKDIFF(METHOD=SCORE2) option in the EXACT statement, PROC FREQTAB computes the confidence limits by inverting a single two-sided exact test that is based on the score statistic (Agresti and Min 2001).

The score statistic is a less discrete statistic than the unstandardized risk difference and produces less conservative confidence limits (Agresti and Min 2001). For more information, see Santner et al. (2007). The section “Miettinen-Nurminen (Score) Confidence Limits” describe computation of the risk difference score statistic. For more information, see Miettinen and Nurminen (1985) and Farrington and Manning (1990).

PROC FREQTAB computes the exact unconditional confidence limits as follows. The risk difference is defined as the difference between the row 1 and row 2 risks (proportions), \(d = p_1 - p_2\), and \(n_1\) and \(n_2\) denote the row totals of the \(2 \times 2\) table. The joint probability function for the table can be expressed in terms of the table cell frequencies, the risk difference, and the nuisance parameter \(p_2\) as

\[
f(n_{11}, n_{21}; n_1, n_2, d, p_2) = \binom{n_1}{n_{11}}(d + p_2)^{n_{11}}(1 - d - p_2)^{n_1 - n_{11}} \times \binom{n_2}{n_{21}}p_2^{n_{21}}(1 - p_2)^{n_2 - n_{21}}
\]

For the tail method (which inverts two separate one-sided exact tests), the 100\((1 - \alpha/2)\)% confidence limits for the risk difference are computed as

\[
d_L = \sup (d_* : P_U(d_*) > \alpha/2) \\
d_U = \inf (d_* : P_L(d_*) > \alpha/2)
\]

where

\[
P_U(d_*) = \sup_{p_2} \left( \sum_{A,T(a) \geq t_0} f(n_{11}, n_{21}; n_1, n_2, d_*, p_2) \right) \\
P_L(d_*) = \sup_{p_2} \left( \sum_{A,T(a) \leq t_0} f(n_{11}, n_{21}; n_1, n_2, d_*, p_2) \right)
\]
Chapter 5: The FREQTAB Procedure

The set $A$ includes all $2 \times 2$ tables in which the row sums are $n_1$ and $n_2$, $T(a)$ denotes the value of the test statistic for table $a$ in $A$, and $t_0$ is the value of the test statistic for the observed table. The test statistic is either the score statistic (by default) or the unstandardized risk difference. To compute $P_U(d_*)$, the sum includes probabilities of those tables for which $(T(a) \geq t_0)$. For a fixed value of $d_*$, $P_U(d_*)$ is defined as the maximum sum over all possible values of $p_2$.

The two-sided score method evaluates the $p$-values $P_U(d_*)$ and $P_L(d_*)$ by comparing $|T(a)|$ to $|t_0|$. To compute the confidence limits $d_L$ and $d_u$, the two-sided method compares the $p$-values to $\alpha$. For more information, see Agresti and Min (2001) and Santner et al. (2007).

**Risk Difference Tests**

PROC FREQTAB provides tests of equality, noninferiority, superiority, and equivalence for the risk (proportion) difference. The following analysis methods are available: Wald (with and without continuity correction), Hauck-Anderson, Farrington-Manning (score), and Newcombe (with and without continuity correction). You can specify the method by using the `METHOD=` riskdiff-option; by default, PROC FREQTAB provides Wald tests.

**Equality Tests**  The equality test for the risk difference tests the null hypothesis that the risk difference equals the null value. You can specify a null value by using the `EQUAL(NULL=)` riskdiff-option; by default, the null value is 0. This test can be expressed as $H_0: d = d_0$ versus the alternative $H_a: d \neq d_0$, where $d = p_1 - p_2$ denotes the risk difference (for column 1 or column 2) and $d_0$ denotes the null value.

The test statistic is computed as

$$z = \frac{\hat{d} - d_0}{se(\hat{d})}$$

where the standard error $se(\hat{d})$ is computed by using the method that you specify. Available methods for the equality test include Wald (with and without continuity correction), Hauck-Anderson, and Farrington-Manning (score). For a description of the standard error computation, see the subsections “Wald Test,” “Hauck-Anderson Test,” and “Farrington-Manning (Score) Test,” respectively, in the section “Noninferiority Tests” on page 232.

PROC FREQTAB computes one-sided and two-sided $p$-values for equality tests. When the test statistic $z$ is greater than 0, PROC FREQTAB displays the right-sided $p$-value, which is the probability of a larger value occurring under the null hypothesis. The one-sided $p$-value can be expressed as

$$P_1 = \begin{cases} \text{Prob}(Z > z) & \text{if } z > 0 \\ \text{Prob}(Z < z) & \text{if } z \leq 0 \end{cases}$$

where $Z$ has a standard normal distribution. The two-sided $p$-value is computed as $P_2 = 2 \times P_1$.

**Noninferiority Tests**  If you specify the NONINF riskdiff-option, PROC FREQTAB provides a noninferiority test for the risk difference, or the difference between two proportions. The null hypothesis for the noninferiority test is

$$H_0: p_1 - p_2 \leq -\delta$$

versus the alternative

$$H_a: p_1 - p_2 > -\delta$$
where $\delta$ is the noninferiority margin. Rejection of the null hypothesis indicates that the row 1 risk is not inferior to the row 2 risk. See Chow, Shao, and Wang (2003) for more information.

You can specify the value of $\delta$ with the MARGIN= riskdiff-option. By default, $\delta = 0.2$. You can specify the test method with the METHOD= riskdiff-option. The following methods are available for the risk difference noninferiority analysis: Wald (with and without continuity correction), Hauck-Anderson, Farrington-Manning (score), and Newcombe (with and without continuity correction). The Wald, Hauck-Anderson, and Farrington-Manning methods provide tests and corresponding test-based confidence limits; the Newcombe method provides only confidence limits. If you do not specify METHOD=, PROC FREQTAB uses the Wald test by default.

The confidence coefficient for the test-based confidence limits is $100(1-2\alpha)$% (Schuirmann 1999). By default, if you do not specify the ALPHA= option, these are 90% confidence limits. You can compare the confidence limits to the noninferiority limit, $-\delta$.

The following sections describe the noninferiority analysis methods for the risk difference.

**Wald Test**

If you specify the METHOD=WALD riskdiff-option, PROC FREQTAB provides an asymptotic Wald test of noninferiority for the risk difference. This is also the default method. The Wald test statistic is computed as

$$z = (\hat{d} + \delta) / \text{se}(\hat{d})$$

where $(\hat{d} = \hat{p}_1 - \hat{p}_2)$ estimates the risk difference and $\delta$ is the noninferiority margin.

By default, the standard error for the Wald test is computed from the sample proportions as

$$\text{se}(\hat{d}) = \sqrt{\hat{p}_1(1-\hat{p}_1)/n_1 + \hat{p}_2(1-\hat{p}_2)/n_2}.$$  

If you specify the VAR=NULL riskdiff-option, the standard error is based on the null hypothesis that the risk difference equals $-\delta$ (Dunnett and Gent 1977). The standard error is computed as

$$\text{se}(\hat{d}) = \sqrt{\bar{p}(1-\bar{p})/n} + (\bar{p} - \delta)(1-\bar{p} + \delta)/n_1.$$  

where

$$\bar{p} = (n_{11} + n_{21} + \delta n_1)/n$$

If you specify the CORRECT riskdiff-option, the test statistic includes a continuity correction. The continuity correction is subtracted from the numerator of the test statistic if the numerator is greater than 0; otherwise, the continuity correction is added to the numerator. The value of the continuity correction is $(1/n_1 + 1/n_2)/2$.

The $p$-value for the Wald noninferiority test is $P_z = \text{Prob}(Z > z)$, where $Z$ has a standard normal distribution.

**Hauck-Anderson Test**

If you specify the METHOD=HA riskdiff-option, PROC FREQTAB provides the Hauck-Anderson test for noninferiority. The Hauck-Anderson test statistic is computed as

$$z = (\hat{d} + \delta \pm c) / \text{se}(\hat{d})$$

where $c = \sqrt{2/n}$. The test statistic $z$ is compared to the normal distribution to compute the $p$-value.
where $\hat{d} = \hat{p}_1 - \hat{p}_2$ and the standard error is computed from the sample proportions as

$$se(\hat{d}) = \sqrt{\hat{p}_1(1-\hat{p}_1)/(n_1-1) + \hat{p}_2(1-\hat{p}_2)/(n_2-1)}$$

The Hauck-Anderson continuity correction $c$ is computed as

$$c = 1 / \left(2 \min(n_1, n_2)\right)$$

The $p$-value for the Hauck-Anderson noninferiority test is $P_z = \text{Prob}(Z > z)$, where $Z$ has a standard normal distribution. For more information, see Hauck and Anderson (1986) and Schuirmann (1999).

**Farrington-Manning (Score) Test**

If you specify the METHOD=FM riskdiff-option, PROC FREQTAB provides the Farrington-Manning (score) test of noninferiority for the risk difference. A score test statistic for the null hypothesis that the risk difference equals $-\delta$ can be expressed as

$$z = (\hat{d} + \delta) / se(\hat{d})$$

where $\hat{d}$ is the observed value of the risk difference $(\hat{p}_1 - \hat{p}_2)$,

$$se(\hat{d}) = \sqrt{\hat{p}_1(1-\hat{p}_1)/n_1 + \hat{p}_2(1-\hat{p}_2)/n_2}$$

and $\tilde{p}_1$ and $\tilde{p}_2$ are the maximum likelihood estimates of the row 1 and row 2 risks (proportions) under the restriction that the risk difference is $-\delta$. The $p$-value for the noninferiority test is $P_z = \text{Prob}(Z > z)$, where $Z$ has a standard normal distribution. For more information, see Miettinen and Nurminen (1985); Miettinen (1985); Farrington and Manning (1990); Dann and Koch (2005).

The maximum likelihood estimates of $p_1$ and $p_1$, subject to the constraint that the risk difference is $-\delta$, are computed as

$$\tilde{p}_1 = 2u \cos(w) - b/3a \quad \text{and} \quad \tilde{p}_2 = \tilde{p}_1 + \delta$$

where

$$w = (\pi + \cos^{-1}(v/u^3))/3$$
$$v = b^3/(3a)^3 - bc/6a^2 + d/2a$$
$$u = \text{sign}(v)\sqrt{b^2/(3a)^2 - c/3a}$$
$$a = 1 + \theta$$
$$b = -(1 + \theta + \tilde{p}_1 + \theta \tilde{p}_2 - \delta(\theta + 2))$$
$$c = \delta^2 - \delta(2\tilde{p}_1 + \theta + 1) + \tilde{p}_1 + \theta \tilde{p}_2$$
$$d = \tilde{p}_1(1-\delta)$$
$$\theta = n_2/n_1.$$ 

For more information, see Farrington and Manning (1990, p. 1453).

**Newcombe Noninferiority Analysis**

If you specify the METHOD=NEWCOMBE riskdiff-option, PROC FREQTAB provides a noninferiority analysis that is based on Newcombe hybrid-score confidence limits for the risk difference. The confidence
coefficient for the confidence limits is \(100(1 - 2\alpha)\%\) (Schuirmann 1999). By default, if you do not specify the ALPHA= option, these are 90\% confidence limits. You can compare the confidence limits with the noninferiority limit, \(-\delta\). If you specify the CORRECT riskdiff-option, the confidence limits includes a continuity correction. See the subsection “Newcombe Confidence Limits” in the section “Confidence Limits for the Risk Difference” on page 228 for more information.

**Superiority Test** If you specify the SUP riskdiff-option, PROC FREQTAB provides a superiority test for the risk difference. The null hypothesis is

\[H_0: p_1 - p_2 \leq \delta\]

versus the alternative

\[H_a: p_1 - p_2 > \delta\]

where \(\delta\) is the superiority margin. Rejection of the null hypothesis indicates that the row 1 proportion is superior to the row 2 proportion. You can specify the value of \(\delta\) with the MARGIN= riskdiff-option. By default, \(\delta = 0.2\).

The superiority analysis is identical to the noninferiority analysis but uses a positive value of the margin \(\delta\) in the null hypothesis. The superiority computations follow those in the section “Noninferiority Tests” on page 232 by replacing \(-\delta\) by \(\delta\). See Chow, Shao, and Wang (2003) for more information.

**Equivalence Test** If you specify the EQUIV riskdiff-option, PROC FREQTAB provides an equivalence test for the risk difference, or the difference between two proportions. The null hypothesis for the equivalence test is

\[H_0: p_1 - p_2 \leq -\delta_L \quad \text{or} \quad p_1 - p_2 \geq \delta_U\]

versus the alternative

\[H_a: \delta_L < p_1 - p_2 < \delta_U\]

where \(\delta_L\) is the lower margin and \(\delta_U\) is the upper margin. Rejection of the null hypothesis indicates that the two binomial proportions are equivalent. See Chow, Shao, and Wang (2003) for more information.

You can specify the value of the margins \(\delta_L\) and \(\delta_U\) with the MARGIN= riskdiff-option. If you do not specify MARGIN=, PROC FREQTAB uses lower and upper margins of \(-0.2\) and \(0.2\) by default. If you specify a single margin value \(\delta\), PROC FREQTAB uses lower and upper margins of \(-\delta\) and \(\delta\). You can specify the test method with the METHOD= riskdiff-option. The following methods are available for the risk difference equivalence analysis: Wald (with and without continuity correction), Hauck-Anderson, Farrington-Manning (score), and Newcombe (with and without continuity correction). The Wald, Hauck-Anderson, and Farrington-Manning methods provide tests and corresponding test-based confidence limits; the Newcombe method provides only confidence limits. If you do not specify METHOD=, PROC FREQTAB uses the Wald test by default.

PROC FREQTAB computes two one-sided tests (TOST) for equivalence analysis (Schuirmann 1987). The TOST approach includes a right-sided test for the lower margin \(\delta_L\) and a left-sided test for the upper margin \(\delta_U\). The overall \(p\)-value is taken to be the larger of the two \(p\)-values from the lower and upper tests.

The section “Noninferiority Tests” on page 232 gives details about the Wald, Hauck-Anderson, Farrington-Manning (score), and Newcombe methods for the risk difference. The lower margin equivalence test statistic
takes the same form as the noninferiority test statistic but uses the lower margin value $\delta_L$ in place of $-\delta$. The upper margin equivalence test statistic take the same form as the noninferiority test statistic but uses the upper margin value $\delta_U$ in place of $-\delta$.

The test-based confidence limits for the risk difference are computed according to the equivalence test method that you select. If you specify METHOD=WALD with VAR=NULL, or METHOD=FM, separate standard errors are computed for the lower and upper margin tests. In this case, the test-based confidence limits are computed by using the maximum of these two standard errors. These confidence limits have a confidence coefficient of $100(1 - 2\alpha)\%$ (Schuirmann 1999). By default, if you do not specify the ALPHA= option, these are 90% confidence limits. You can compare the test-based confidence limits to the equivalence limits, $(\delta_L, \delta_U)$.

**Barnard’s Unconditional Exact Test**

The BARNARD option in the EXACT statement provides an unconditional exact test for the risk (proportion) difference for $2 \times 2$ tables. The reference set for the unconditional exact test consists of all $2 \times 2$ tables that have the same row sums as the observed table (Barnard 1945, 1947, 1949). This differs from the reference set for exact conditional inference, which is restricted to the set of tables that have the same row sums and the same column sums as the observed table. See the sections “Fisher’s Exact Test” on page 206 and “Exact Statistics” on page 271 for more information.

The test statistic is the standardized risk difference, which is computed as

$$T = \frac{d}{\sqrt{p_1(1 - p_1)(1/n_1 + 1/n_2)}}$$

where the risk difference $d$ is defined as the difference between the row 1 and row 2 risks (proportions), $d = (n_1/n_1 - n_{21}/n_2)$; $n_1$ and $n_2$ are the row 1 and row 2 totals, respectively; and $p_1$ is the overall proportion in column 1, $(n_1 + n_{21})/n$.

Under the null hypothesis that the risk difference is 0, the joint probability function for a table can be expressed in terms of the table cell frequencies, the row totals, and the unknown parameter $\pi$ as

$$f(n_{11}, n_{21}; n_1, n_2, \pi) = \binom{n_1}{n_{11}} \binom{n_2}{n_{21}} \pi^{n_{11} + n_{21}} (1 - \pi)^{n - n_{11} - n_{21}}$$

where $\pi$ is the common value of the risk (proportion).

PROC FREQTAB sums the table probabilities over the reference set for those tables where the test statistic is greater than or equal to the observed value of the test statistic. This sum can be expressed as

$$\text{Prob}(\pi) = \sum_{A, T(a) \geq t_0} f(n_{11}, n_{21}; n_1, n_2, \pi)$$

where the set $A$ contains all $2 \times 2$ tables with row sums equal to $n_1$ and $n_2$, and $T(a)$ denotes the value of the test statistic for table $a$ in $A$. The sum includes probabilities of those tables for which $(T(a) \geq t_0)$, where $t_0$ is the value of the test statistic for the observed table.

The sum $\text{Prob}(\pi)$ depends on the unknown value of $\pi$. To compute the exact $p$-value, PROC FREQTAB eliminates the nuisance parameter $\pi$ by taking the maximum value of $\text{Prob}(\pi)$ over all possible values of $\pi$,

$$\text{Prob} = \sup_{0 \leq \pi \leq 1} (\text{Prob}(\pi))$$

Common Risk Difference

If you specify the COMMONRISKDIFF option in the TABLES statement, PROC FREQTAB provides estimates, confidence limits, and tests for the common (overall) risk difference for multiway 2 x 2 tables.

Mantel-Haenszel Confidence Limits and Test

PROC FREQTAB computes the Mantel-Haenszel estimate, confidence limits, and test for the common risk difference by using Mantel-Haenszel stratum weights (Mantel and Haenszel 1959) and the Sato variance estimator (Sato 1989). The Mantel-Haenszel estimate of the common risk difference is

\[ \hat{d}_{\text{MH}} = \sum_h \hat{d}_h w_h \]

where \( \hat{d}_h \) is the risk difference in stratum \( h \) and

\[ w_h = \frac{n_{h1}n_{h2}}{n_h} / \sum_i \frac{n_{i1}n_{i2}}{n_i} \]

is the Mantel-Haenszel weight of stratum \( h \). The column 1 risk difference in stratum (2 x 2 table) \( h \) is computed as

\[ \hat{d}_h = \hat{p}_{h1} - \hat{p}_{h2} = (n_{h11}/n_{h1}) - (n_{h21}/n_{h2}) \]

where \( \hat{p}_{h1} \) is the proportion of row 1 observations that are classified in column 1 and \( \hat{p}_{h2} \) is the proportion or row 2 observations that are classified in column 1. The column 2 risk is computed in the same way. For more information, see Agresti (2013, p. 231).

PROC FREQTAB computes the variance of \( \hat{d}_{\text{MH}} \) (Sato 1989) as

\[ \hat{\sigma}^2(\hat{d}_{\text{MH}}) = \left( \hat{d}_{\text{MH}} \sum_h P_h + \sum_h Q_h / \left( \sum_h n_{h1}n_{h2}/n_h \right)^2 \right) \]

where

\[ P_h = \left( n_{h11}^2 - n_{h21}^2 n_{h1} - n_{h1} n_{h2} n_{h21} n_{h11} + n_{h21} n_{h11} n_{h2} / n_h^2 \right) \]

\[ Q_h = \left( n_{h11} n_{h2} n_{h21} + n_{h21} (n_{h11} - n_{h12}) / 2 n_h \right) \]

The 100(1 - \alpha) % confidence limits for the common risk difference are

\[ \hat{d}_{\text{MH}} \pm \left( z_{\alpha/2} \times \hat{\sigma}(\hat{d}_{\text{MH}}) \right) \]

If you specify the COMMONRISKDIFF(TEST=MH) option, PROC FREQTAB provides a Mantel-Haenszel test of the null hypothesis that the common risk difference is 0, which is computed as \( Z_{\text{MH}} = \hat{d}_{\text{MH}} / \hat{\sigma}(\hat{d}_{\text{MH}}) \). The two-sided p-value is \( \text{Prob}(|Z| > |Z_{\text{MH}}|) \), where \( Z \) has a standard normal distribution.
**Klingenberg Confidence Limits**

Klingenberg confidence limits (Klingenberg 2014) for the Mantel-Haenszel common risk difference are based on inverting a test of homogeneity that uses the null form of the Sato variance estimator (Sato 1989). For performance evaluation of Klingenberg confidence limits, see Fisher (2015) and Klingenberg (2014).

The $100(1 - \alpha)\%$ Klingenberg confidence limits for the common risk difference are

$$\hat{d}_{\text{Mid}} \pm M_{\alpha/2}$$

where $M$ (margin of error) is computed as

$$M_{\alpha/2} = \sqrt{\hat{d}_{\text{Mid}}^2 - \hat{d}_{\text{MH}}^2 + z_{\alpha/2}^2 \left(\frac{Q}{W^2}\right)}$$

and the confidence interval midpoint is computed as

$$\hat{d}_{\text{Mid}} = \hat{d}_{\text{MH}} + 0.5 z_{\alpha/2}^2 \left(\frac{P}{W^2}\right)$$

The values $P$, $Q$, and $W$ are computed as

$$P = \sum_h P_h$$

$$Q = \sum_h Q_h$$

$$W = \sum_h n_{h1} \cdot n_{h2} / n_h$$

where $h$ denotes the stratum, and $P_h$ and $Q_h$ are defined in the section “Mantel-Haenszel Confidence Limits and Test” on page 237.

**Minimum Risk Confidence Limits and Test**

PROC FREQTAB computes the minimum risk estimate, confidence limits, and test for the common risk difference by using the method of Mehrotra and Railkar (2000). The stratum estimates are weighted by minimum risk weights, which minimize the mean square error of the estimate of the common risk difference. Minimum risk weights are designed to improve precision and reduce bias (compared to other weighting strategies) and can minimize the power loss that can occur when underlying assumptions are not met. For more information, see Mehrotra (2001) and Dmitrienko et al. (2005, section 1.3.3).

The minimum risk estimate of the common risk difference is

$$\hat{d}_{\text{MR}} = \sum_h \hat{d}_h w_h^*$$

where $\hat{d}_h$ is the risk difference in stratum $h$ and $w_h^*$ is the minimum risk weight of stratum $h$ (which is described in the section “Minimum Risk Weights” on page 239). The variance of $\hat{d}_{\text{MR}}$ is estimated by

$$\hat{V}(\hat{d}_{\text{MR}}) = \sum_h w_h^* \hat{V}_h$$

where $\hat{V}_h$ (the variance estimate of the stratum $h$ risk difference) is computed as

$$\hat{V}_h = \hat{p}_{h1}(1 - \hat{p}_{h1})/n_{h1} + \hat{p}_{h2}(1 - \hat{p}_{h2})/n_{h2}.$$
The 100(1 − α)% minimum risk confidence limits for the common risk difference are
\[
\hat{d}_{MR} \pm \left( c + \frac{z_{\alpha/2}}{\sqrt{\hat{V}(\hat{d}_{MR})}} \right)
\]
where the continuity correction is
\[
c = 0.1875 / \sum_h (n_{h1}n_{h2}/n_h)
\]
The continuity correction is applied only when \(c < |\hat{d}_{MR}|\) (Fleiss, Levin, and Paik 2003). You can remove the continuity correction by specifying the \texttt{COMMONRISKDIFF(CORRECT=NO)} option.

By default, the minimum risk test is computed as
\[
z_{MR} = \frac{(\hat{d}_{MR} \pm c)}{\sqrt{\hat{V}_0(\hat{d}_{MR})}}
\]
The continuity correction \(c\) is subtracted from \(\hat{d}_{MR}\) if \(\hat{d}_{MR} > 0\) and added to \(\hat{d}_{MR}\) if \(\hat{d}_{MR} < 0\). The null variance of the common risk difference is estimated by
\[
\hat{V}_0(\hat{d}_{MR}) = \sum_h w_h^2 \hat{V}_0(\hat{d}_h)
\]
where \(\hat{V}_0(\hat{d}_h)\) (an estimate of the variance of the stratum \(h\) risk difference under the null hypothesis) is
\[
\hat{V}_0(\hat{d}_h) = \hat{p}_h(1 - \hat{p}_h) (1/n_{h1} + 1/n_{h2})
\]
and
\[
\hat{p}_h = (n_{h1}\hat{p}_{h1} + n_{h2}\hat{p}_{h2}) / (n_{h1} + n_{h2})
\]
The two-sided \(p\)-value is \(\text{Prob}(|Z| > |z_{MR}|)\), where \(Z\) has a standard normal distribution.

If you specify the \texttt{VAR=SAMPLE} option for \texttt{COMMONRISKDIFF(\texttt{TEST=MR})}, \texttt{PROC FREQTAB} uses the sample variance estimate \(\hat{V}(\hat{d}_{MR})\) instead of the null variance estimate \(\hat{V}_0(\hat{d}_{MR})\) in the denominator of the test statistic \(z_{MR}\). If you specify the \texttt{COMMONRISKDIFF(CORRECT=NO)} option, the continuity correction is not included in the test statistic.

**Minimum Risk Weights** The estimate of the minimum risk weight for stratum \(h\) is defined by Mehrotra and Railkar (2000), as
\[
w_h^* = \frac{\beta_h}{\sum_i \hat{V}_i^{-1}} \left( \frac{\alpha_h \hat{V}_h^{-1}}{\sum_i \alpha_i \hat{V}_i^{-1} + \sum_k \alpha_i \hat{d}_i \hat{V}_i^{-1}} \right) \left( \frac{\sum_i \hat{d}_i \beta_i}{\sum_i \hat{V}_i^{-1}} \right)
\]
where
\[
\alpha_h = \sum_i \hat{V}_i^{-1} - \sum_i \hat{d}_i \hat{V}_i^{-1}
\]
\[
\beta_h = \hat{V}_h \left( 1 + \alpha_h \sum_i f_i \hat{d}_i \right)
\]
and \(f_h\) is the fraction in stratum \(h\)
\[
f_h = n_h / \sum_i n_h
\]
All sums are over the \(s\) strata (2 × 2 tables) in the multiway table request, \(\hat{d}_i\) denotes the risk difference estimate in stratum \(i\), and \(\hat{V}_i\) denotes the sample variance estimate of the risk difference in stratum \(i\).
Summary Score Confidence Limits
PROC FREQTAB computes the summary score estimate of the common risk difference (Agresti 2013, p. 231) by using inverse-variance stratum weights and Miettinen-Nurminen (score) confidence limits for the stratum risk differences. For more information, see the section “Miettinen-Nurminen (Score) Confidence Limits.”

The score confidence interval for the risk difference in stratum $h$ can be expressed as

$$
\bar{d}_h' \pm z_{\alpha/2} s_h',
$$

where $\bar{d}_h'$ is the midpoint of the score confidence interval and $s_h'$ is the width of the confidence interval divided by $2z_{\alpha/2}$. The summary score estimate of the common risk difference is computed as

$$
\bar{d}_S = \sum_h \bar{d}_h' w_h'
$$

where

$$
w_h' = (1/s_h'^2) / \sum_i (1/s_i'^2)
$$

The variance of $\bar{d}_S$ is computed as

$$
\hat{\sigma}^2(\bar{d}_S) = 1 / \sum_h (1/s_h'^2)
$$

The $100(1 - \alpha)%$ summary score confidence limits for the common risk difference are

$$
\bar{d}_S \pm \left( z_{\alpha/2} \times \hat{\sigma}(\bar{d}_S) \right)
$$

If you specify the COMMONRISKDIFF(TEST=SCORE) option, PROC FREQTAB provides a summary score test of the null hypothesis that the common risk difference is 0. The test statistic is $z_S = \bar{d}_S / \hat{\sigma}(\bar{d}_S)$

The two-sided $p$-value is $\text{Prob}(|Z| > |z_S|)$ where $Z$ has a standard normal distribution.

Stratified Newcombe Confidence Limits
PROC FREQTAB computes stratified Newcombe confidence limits for the common risk (proportion) difference by using the method of Yan and Su (2010). The stratified Newcombe confidence limits are constructed from stratified Wilson confidence limits for the common (overall) row proportions. By default, the strata are weighted by Mantel-Haenszel weights; if you specify the COMMONRISKDIFF(CL=NEWCOMBEMR) option, the strata are weighted by minimum risk weights.

PROC FREQTAB first computes individual Wilson confidence limits for the row proportions in each $2 \times 2$ table (stratum), as described in the section “Wilson (Score) Confidence Limits” on page 220. These stratum Wilson confidence limits are then combined to form stratified Wilson confidence limits for the overall row proportions by using stratum weights (either Mantel-Haenszel or minimum risk). The confidence levels of the stratum Wilson confidence limits are chosen so that the overall confidence coefficient (for the stratified Wilson confidence limits) is $100(1 - \alpha)%$ (Yan and Su 2010).

Denote the lower and upper stratified Wilson score confidence limits for the common row 1 proportion as $L_1$ and $U_1$, respectively, and denote the lower and upper stratified Wilson confidence limits for the common row 2 proportion as $L_2$ and $U_2$, respectively. The $100(1 - \alpha)%$ stratified Newcombe confidence limits for the common risk (proportion) difference are

$$
L = \hat{d} - z_{\alpha/2} \sqrt{\lambda_1 L_1 (1 - L_1) + \lambda_2 U_2 (1 - U_2)}
$$

$$
U = \hat{d} + z_{\alpha/2} \sqrt{\lambda_2 L_2 (1 - L_2) + \lambda_1 U_1 (1 - U_1)}
$$

where

$$
\lambda_1 = \frac{s_1^2}{\sum_i s_i^2}
$$

$$
\lambda_2 = \frac{s_2^2}{\sum_i s_i^2}
$$

$\lambda_1$ and $\lambda_2$ weight the stratum confidence limits. The parameter $\lambda_1$ is the relative weight for the lower proportion, and $\lambda_2$ is the relative weight for the upper proportion.
where \( \hat{d} \) is the weighted estimate of the common risk difference and

\[
\begin{align*}
\lambda_1 &= \sum_h w_h^2 / n_{h1}, \\
\lambda_2 &= \sum_h w_h^2 / n_{h2}.
\end{align*}
\]

By default, the strata are weighted by Mantel-Haenszel weights, which are defined as

\[
w_h = \frac{n_{h1} n_{h2}}{n_h} / \sum_i \frac{n_{i1} n_{i2}}{n_i}
\]

and the weighted estimate of the common risk difference is \( \hat{d}_{MH} \). For more information, see the section “Mantel-Haenszel Confidence Limits and Test” on page 237. Optionally, the strata are weighted by minimum risk weights, and the weighted estimate of the common risk difference is \( \hat{d}_{MR} \). For more information, see the section “Minimum Risk Confidence Limits and Test” on page 238.

When there is a single stratum, the stratified Newcombe confidence interval is equivalent to the (unstratified) Newcombe confidence interval. For more information, see the subsection “Newcombe Confidence Limits” in the section “Confidence Limits for the Risk Difference” on page 228. See also Kim and Won (2013).

### Odds Ratio and Relative Risks for 2 × 2 Tables

#### Odds Ratio

The odds ratio is a useful measure of association for a variety of study designs. For a retrospective design called a *case-control study*, the odds ratio can be used to estimate the relative risk when the probability of positive response is small (Agresti 2002). In a case-control study, two independent samples are identified based on a binary (yes-no) response variable, and the conditional distribution of a binary explanatory variable is examined within fixed levels of the response variable. For more information, see Stokes, Davis, and Koch (2012), Agresti (2013), and Agresti (2007).

The odds of a positive response (column 1) in row 1 is \( n_{11} / n_{12} \). Similarly, the odds of a positive response in row 2 is \( n_{21} / n_{22} \). The odds ratio is formed as the ratio of the row 1 odds to the row 2 odds. The odds ratio for a 2 × 2 table is defined as

\[
OR = \frac{n_{11} / n_{12}}{n_{21} / n_{22}} = \frac{n_{11} n_{22}}{n_{12} n_{21}}
\]

The odds ratio can be any nonnegative number. When the row and column variables are independent, the true value of the odds ratio is 1. An odds ratio greater than 1 indicates that the odds of a positive response are higher in row 1 than in row 2. An odds ratio less than 1 indicates that the odds of a positive response are higher in row 2. The strength of association increases as the deviation from 1 increases.

The transformation \( G = (OR - 1)/(OR + 1) \) transforms the odds ratio to the range \((-1,1)\), where \( G = 0 \) when \( OR = 1 \); \( G = -1 \) when \( OR = 0 \); and \( G \) approaches 1 as \( OR \) approaches infinity. \( G \) is the gamma statistic, which PROC FREQTAB computes when you specify the MEASURES option.
**Confidence Limits for the Odds Ratio**  The following types of confidence limits are available for the odds ratio: exact, exact mid-$p$, likelihood ratio, score, Wald, and Wald modified.

**Wald Confidence Limits**
The asymptotic Wald confidence limits are based on a log transformation of the odds ratio (Woolf 1955; Haldane 1955). PROC FREQTAB computes the Wald confidence limits as

\[
\text{OR} \times \exp(-z\sqrt{v}), \quad \text{OR} \times \exp(z\sqrt{v})
\]

where

\[
v = \text{Var}(\log \text{OR}) = 1/n_{11} + 1/n_{12} + 1/n_{21} + 1/n_{22}
\]

and $z$ is the 100$(1 - \alpha/2)$th percentile of the standard normal distribution. The confidence level $\alpha$ is determined by the ALPHAN option in the TABLES statement; by default, ALPHAN=0.05, which produces 95% confidence limits for the odds ratio. If any of the four cell frequencies are 0, $v$ is undefined and the Wald confidence limits cannot be computed. For more information, see Agresti (2013, p. 70).

**Wald Modified Confidence Limits**
PROC FREQTAB computes Wald modified confidence limits (Haldane 1955) for the odds ratio by replacing the $n_{ij}$ by $(n_{ij} + 0.5)$ in the estimator OR and the variance $v$ as follows:

\[
\text{OR} = \frac{(n_{11} + 0.5)(n_{22} + 0.5)}{(n_{12} + 0.5)(n_{21} + 0.5)}
\]

\[
v = \text{Var}(\log \text{OR}) = 1/(n_{11} + 0.5) + 1/(n_{12} + 0.5) + 1/(n_{21} + 0.5) + 1/(n_{22} + 0.5)
\]

The modified confidence limits are then computed as

\[
(\text{OR} \times \exp(-z\sqrt{v}), \quad \text{OR} \times \exp(z\sqrt{v}))
\]

where $z$ is the 100$(1 - \alpha/2)$th percentile of the standard normal distribution. For more information, see Fleiss, Levin, and Paik (2003) and Agresti (2013).

**Score Confidence Limits**
Score confidence limits for the odds ratio (Miettinen and Nurminen 1985) are computed by inverting score tests for the odds ratio. A score-based chi-square test statistic for the null hypothesis that the odds ratio is $\theta$ can be expressed as

\[
Q(\theta) = \left\{ n_1, (\hat{p}_1 - \tilde{p}_1) \right\}^2 / \left\{ n/(n - 1) \right\} \{1/(n_1, \hat{p}_1(1 - \hat{p}_1)) + 1/(n_2, \tilde{p}_2(1 - \tilde{p}_2))\}^{-1}
\]

where $\hat{p}_1$ is the observed row 1 risk ($n_{11}/n_1$), and $\hat{p}_1$ and $\tilde{p}_2$ are the maximum likelihood estimates of the row 1 and row 2 risks under the restriction that the odds ratio $(n_{11}n_{22}/n_{12}n_{21})$ is $\theta$. For more information, see Miettinen and Nurminen (1985) and Miettinen (1985, chapter 14).

The 100$(1 - \alpha)$% score confidence interval for the odds ratio consists of all values of $\theta$ for which the test statistic $Q(\theta)$ falls in the acceptance region,

\[
\{ \theta : Q(\theta) < \chi^2_{1,\alpha} \} \]
where $\chi^2_{1,\alpha}$ is the 100$(1 - \alpha)$th percentile of the chi-square distribution with 1 degree of freedom. For more information about score confidence limits, see Agresti (2013).

By default, the score confidence limits include the bias correction factor $n/(n - 1)$ in the denominator of $Q(\theta)$ (Miettinen and Nurminen 1985, p. 217). If you specify the CL=SCORE(CORRECT=NO) option, PROC FREQTAB does not include this factor in the computation.

The maximum likelihood estimates of $p_1$ and $p_2$, subject to the constraint that the odds ratio is $\theta$, are computed as

$$
\hat{p}_2 = \left(-b + \sqrt{b^2 - 4ac}\right)/2a \quad \text{and} \quad \hat{p}_1 = \hat{p}_2\theta/(1 + \hat{p}_2(\theta - 1))
$$

where

$$
a = n_2.(\theta - 1) \\
b = n_1.\theta + n_2. - \hat{p}_1(\theta - 1) \\
c = -\hat{p}_1
$$

For more information, see Miettinen and Nurminen (1985, pp. 217–218) and Miettinen (1985, chapter 14).

**Likelihood Ratio Confidence Limits**

Likelihood ratio (profile likelihood) confidence limits for the odds ratio are computed by inverting likelihood ratio tests. The likelihood ratio test statistic for the null hypothesis that the odds ratio is $\theta$ can be expressed as

$$
G^2(\theta) = 2 \left(n_{11} \log(\hat{p}_1/\bar{p}_1) + n_{12} \log((1-\hat{p}_1)/(1-\bar{p}_1)) + n_{21} \log(\hat{p}_2/\bar{p}_2) + n_{22} \log((1-\hat{p}_2)/(1-\bar{p}_2))\right)
$$

where $\hat{p}_i$ is the observed row $i$ risk ($n_{1i}/n_{.1}$) and $\bar{p}_i$ is the maximum likelihood estimate of the row $i$ risk under the restriction that the odds ratio is $\theta$. The computation of the maximum likelihood estimates is described in the subsection “Score Confidence Limits” in this section. For more information, see Agresti (2013), Miettinen and Nurminen (1985), and Miettinen (1985, chapter 14).

The 100$(1 - \alpha)$% likelihood ratio confidence interval for the odds ratio consists of all values of $\theta$ for which the test statistic $G^2(\theta)$ falls in the acceptance region,

$$\{\theta : G^2(\theta) < \chi^2_{1,\alpha}\}$$

where $\chi^2_{1,\alpha}$ is the 100$(1 - \alpha)$th percentile of the chi-square distribution with 1 degree of freedom.

**Exact Confidence Limits**

PROC FREQTAB computes exact confidence limits for the odds ratio by inverting two one-sided (equal-tail) exact tests that are based on the noncentral hypergeometric distribution, where the distribution is conditional on the observed marginal totals of the $2 \times 2$ table. The exact confidence limits $\phi_1$ and $\phi_2$ are the solutions to the equations

$$
\sum_{i=n_{11}}^{n_1} f(i : n_{1.}, n_{1.}. n_{2.}, \phi_1) = \alpha/2 \\
\sum_{i=0}^{n_{11}} f(i : n_{1.}, n_{1.}. n_{2.}, \phi_2) = \alpha/2
$$
where

\[ f(i : n_1, n_1, n_2, \phi) = \binom{n_1}{i} \binom{n_2}{n_1 - i} \phi^i / \sum_{i=0}^{n_1} \binom{n_1}{i} \binom{n_2}{n_1 - i} \phi^i \]

For more information, see Fleiss, Levin, and Paik (2003), Thomas (1971), and Gart (1971).

Because this is a discrete problem, the confidence coefficient for the exact confidence interval is not exactly \((1 - \alpha)\) but is at least \((1 - \alpha)\); thus, these confidence limits are conservative. For more information, see Agresti (1992).

When the odds ratio is 0, which occurs when either \(n_{11} = 0\) or \(n_{22} = 0\), PROC FREQTAB sets the lower exact confidence limit to 0 and determines the upper limit by using the level \(\alpha\) (instead of \(\alpha/2\)). Similarly, when the odds ratio is infinity, which occurs when either \(n_{12} = 0\) or \(n_{21} = 0\), PROC FREQTAB sets the upper exact confidence limit to infinity and determines the lower limit by using level \(\alpha\).

**Exact Mid-p Confidence Limits**

PROC FREQTAB computes exact mid-\(p\) confidence limits for the odds ratio by inverting two one-sided hypergeometric tests that include mid-\(p\) tail areas. The mid-\(p\) approach replaces the probability of the observed table by half of that probability in the hypergeometric probability sums, which are described in the subsection “Exact Confidence Limits” in this section. The exact mid-\(p\) confidence limits \(\phi_1\) and \(\phi_2\) are the solutions to the equations

\[
\sum_{i=n_{11}+1}^{n_1} \left( f(i : n_1, n_1, n_2, \phi_1) \right) + (1/2) f(n_{11} : n_1, n_1, n_2, \phi_1) = \alpha/2
\]

\[
\sum_{i=0}^{n_{11}-1} \left( f(i : n_1, n_1, n_2, \phi_2) \right) + (1/2) f(n_{11} : n_1, n_1, n_2, \phi_2) = \alpha/2
\]

where

\[ f(i : n_1, n_1, n_2, \phi) = \binom{n_1}{i} \binom{n_2}{n_1 - i} \phi^i / \sum_{i=0}^{n_1} \binom{n_1}{i} \binom{n_2}{n_1 - i} \phi^i \]

For more information, see Agresti (2013).

Relative risks are useful measures in cohort (prospective) study designs, where two samples are identified based on the presence or absence of an explanatory factor. The two samples are observed in future time for the binary (yes-no) response variable under study. Relative risks are also useful in cross-sectional studies, where two variables are observed simultaneously. For more information, see Stokes, Davis, and Koch (2012) and Agresti (2007).
The relative risk is the ratio of the row 1 risk to the row 2 risk in a $2 \times 2$ table. The column 1 risk in row 1 is the proportion of row 1 observations that are classified in column 1, which can be expressed as

$$p_1 = \frac{n_{11}}{n_1}.$$

Similarly, the column 1 risk in row 2 is

$$p_2 = \frac{n_{21}}{n_2}.$$

The column 1 relative risk is computed as

$$R = \frac{p_1}{p_2}.$$

A relative risk greater than 1 indicates that the probability of positive response is greater in row 1 than in row 2. Similarly, a relative risk less than 1 indicates that the probability of positive response is less in row 1 than in row 2. The strength of association increases as the deviation from 1 increases.

**Confidence Limits for the Relative Risk** PROC FREQTAB provides the following types of confidence limits for the relative risk: exact unconditional, likelihood ratio, score, Wald, and Wald modified.

**Wald Confidence Limits**

The asymptotic Wald confidence limits are based on a log transformation of the relative risk. PROC FREQTAB computes the Wald confidence limits for the column 1 relative risk as

$$\left( \hat{R} \times \exp(-z \sqrt{v}), \hat{R} \times \exp(z \sqrt{v}) \right)$$

where $\hat{R}$ is the observed value of the relative risk, $\hat{p}_1/\hat{p}_2$, and

$$v = \text{Var}(\log(\hat{R})) = \left( (1 - \hat{p}_1)/n_{11} \right) + \left( (1 - \hat{p}_2)/n_{21} \right)$$

and $z$ is the 100(1 - $\alpha$/2)th percentile of the standard normal distribution. The confidence level $\alpha$ is determined by the ALPHA= option in the TABLES statement; by default, ALPHA=0.05, which produces 95% confidence limits. If either cell frequency $n_{11}$ or $n_{21}$ is 0, then $v$ is undefined and the Wald confidence limits cannot be computed.

PROC FREQTAB computes the confidence limits for the column 2 relative risk in the same way.

**Wald Modified Confidence Limits**

PROC FREQTAB computes Wald modified confidence limits (Haldane 1955) for the relative risk by replacing the $n_{ij}$ with $(n_{ij} + 0.5)$ and the $n_i$, with $(n_i + 0.5)$ in the estimator $R$ and the variance $v$ as follows:

$$\hat{r}_m = \frac{\hat{p}_1}{\hat{p}_2} = \frac{(n_{11} + 0.5)/(n_1 + 0.5)}{(n_{21} + 0.5)/(n_2 + 0.5)}$$

$$v = \text{Var}(\log(\hat{r}_m)) = 1/(n_{11} + 0.5) + 1/(n_{21} + 0.5) - 1/(n_1 + 0.5) - 1/(n_2 + 0.5)$$

The confidence limits are computed as

$$\left( \hat{r}_m \times \exp(-z \sqrt{v}), \hat{r}_m \times \exp(z \sqrt{v}) \right)$$

where $z$ is the 100(1 - $\alpha$/2)th percentile of the standard normal distribution. For more information, see Fleiss, Levin, and Paik (2003) and Agresti (2013).
Score Confidence Limits

Score confidence limits (Miettinen and Nurminen 1985; Farrington and Manning 1990) are computed by inverting score tests for the relative risk. A score-based chi-square test statistic for the null hypothesis that the relative risk is $r_0$ can be expressed as

$$Q(r_0) = \frac{(\hat{p}_1 - r_0 \hat{p}_2)^2}{\text{Var}(r_0)}$$

where $\hat{p}_1$ and $\hat{p}_2$ are the observed row 1 and row 2 risks (proportions), respectively,

$$\text{Var}(r_0) = \left( \frac{n}{n-1} \right) \left( \frac{\hat{p}_1(1-\hat{p}_1)/n_1 + r_0^2 \hat{p}_2(1-\hat{p}_2)/n_2}{2} \right)$$

where $\hat{p}_1$ and $\hat{p}_2$ are the maximum likelihood estimates of $p_1$ and $p_2$, respectively, under the null hypothesis that the relative risk is $r_0$. For more information, see Miettinen and Nurminen (1985) and Miettinen (1985, chapter 13).

The $100(1-\alpha)\%$ score confidence interval for the relative risk consists of all values of $r_0$ for which the test statistic $Q(r_0)$ falls in the acceptance region,

$$\{r_0 : Q(r_0) < \chi^2_{1,\alpha} \}$$

where $\chi^2_{1,\alpha}$ is the $100(1-\alpha)$th percentile of the chi-square distribution with 1 degree of freedom. For more information, see Agresti (2013).

By default, the score confidence limits include the bias correction factor $n/(n-1)$ in the denominator of $Q(r_0)$ (Miettinen and Nurminen 1985, p. 217). If you specify the CL=SCORE(CORRECT=NO) option, PROC FREQTAB does not include this factor in the computation.

The maximum likelihood estimates of $p_1$ and $p_2$, subject to the constraint that the relative risk is $r_0$, are computed as

$$\hat{p}_1 = \left( -b - \sqrt{b^2 - 4ac} \right) / 2a \quad \text{and} \quad \hat{p}_2 = \hat{p}_1 / r_0$$

where

$$a = 1 + \theta$$
$$b = -r_0(1 + \theta \hat{p}_2 + \theta + \hat{p}_1)$$
$$c = r_0(\hat{p}_1 + \theta \hat{p}_2)$$
$$\theta = n_2 / n_1.$$

For more information, see Farrington and Manning (1990, p. 1454) and Miettinen and Nurminen (1985, p. 217).

Likelihood Ratio Confidence Limits

Likelihood ratio (profile likelihood) confidence limits for the relative risk are computed by inverting likelihood ratio tests. The likelihood ratio test statistic for the null hypothesis that the relative risk ratio is $r_0$ can be expressed as

$$G^2(r_0) = 2 \left( n_{11} \log(\hat{p}_1/\hat{p}_1) + n_{12} \log((1-\hat{p}_1)/(1-\hat{p}_1)) + n_{21} \log(\hat{p}_2/\hat{p}_2) + n_{22} \log((1-\hat{p}_2)/(1-\hat{p}_2)) \right)$$
where \( \hat{p}_i \) is the observed row \( i \) risk \( (n_{i1}/n_i) \) and \( \hat{p}_i \) is the maximum likelihood estimate of the row \( i \) risk under the restriction that the relative risk is \( r_0 \). Expressions for the maximum likelihood estimates \( \hat{p}_1 \) and \( \hat{p}_2 \) are given in the subsection “Score Confidence Limits” in this section. For more information, see Miettinen and Nurminen (1985) and Miettinen (1985, chapter 13).

The 100(1 – \( \alpha \))% likelihood ratio confidence interval for the relative risk consists of all values of \( r_0 \) for which the test statistic \( G^2(r_0) \) falls in the acceptance region,

\[
\{ \theta : G^2(r_0) < \chi^2_{1,\alpha} \}
\]

where \( \chi^2_{1,\alpha} \) is the 100(1 – \( \alpha \))th percentile of the chi-square distribution with 1 degree of freedom.

**Exact Unconditional Confidence Limits**

If you specify the RELRISK option in the EXACT statement, PROC FREQTAB provides exact unconditional confidence limits for the relative risk. The exact unconditional approach fixes the row margins of the 2 \( \times \) 2 table and eliminates the nuisance parameter \( p_2 \) by using the maximum \( p \)-value (worst-case scenario) over all possible values of \( p_2 \) (Santner and Snell 1980). The conditional approach, which is described in the section “Exact Statistics” on page 271, does not apply to the relative risk because of the nuisance parameter (Agresti 1992).

By default, PROC FREQTAB computes the confidence limits by the tail method, which inverts two separate one-sided exact tests of the relative risk, where the tests are based on the score statistic (Chan and Zhang 1999). The size of each one-sided exact test is at most \( \alpha/2 \), and the confidence coefficient is at least \((1 - \alpha)\). If you specify the RELRISK(METHOD=NOSCORE) option in the EXACT statement, PROC FREQTAB computes the confidence limits by inverting two separate one-sided exact tests that are based on the unstandardized relative risk. If you specify the RELRISK(METHOD=SCORE2) option in the EXACT statement, PROC FREQTAB computes the confidence limits by inverting a single two-sided exact test that is based on the score statistic (Agresti and Min 2001).

PROC FREQTAB uses the relative risk score statistic (or the modified form of the unstandardized relative risk) to compute the exact confidence limits as described in the subsection “Exact Unconditional Confidence Limits” in the section “Confidence Limits for the Risk Difference” on page 228.

The score statistic is a less discrete statistic than the unstandardized risk difference and produces less conservative confidence limits (Agresti and Min 2001). For more information, see Santner et al. (2007). The relative risk score statistic (Miettinen and Nurminen 1985; Farrington and Manning 1990) is computed as

\[
z(r_0) = \frac{(\hat{p}_1 - r_0 \hat{p}_2)}{se(r_0)}
\]

where

\[
se(r_0) = \sqrt{\frac{\hat{p}_1(1 - \hat{p}_1) / n_1 + r_0^2 \hat{p}_2(1 - \hat{p}_2) / n_2}{}}.
\]

where \( \hat{p}_1 \) and \( \hat{p}_2 \) are the maximum likelihood estimates of \( p_1 \) and \( p_2 \) under the restriction that the relative risk is \( r_0 \). Expressions for the maximum likelihood estimates \( \hat{p}_1 \) and \( \hat{p}_2 \) are given in the subsection “Score Confidence Limits” in this section. For more information, see Farrington and Manning (1990, p. 1454) and Miettinen and Nurminen (1985, p. 217).

When the confidence limits are computed by using the unstandardized relative risk as the test statistic (METHOD=NOSCORE), PROC FREQTAB uses a modified form of the relative risk to ensure that the
statistic is defined when there are zero-frequency table cells. The modified form adds 0.5 to the table cell and row frequencies (Gart and Nam 1988) and is computed as:

\[
\hat{r} = \frac{(n_{11} + 0.5) / (n_{1-} + 0.5)}{(n_{21} + 0.5) / (n_{2-} + 0.5)}
\]

For more information, see the subsection “Wald Modified Confidence Limits” in this section.

**Relative Risk Tests** PROC FREQTAB provides tests of equality, noninferiority, superiority, and equivalence for the relative risk. The following analysis methods are available: Wald (which is based on a log transformation), Wald modified, score, and likelihood ratio. You can specify the method by using the METHOD= relrisk-option; by default, PROC FREQTAB provides Wald tests.

**Equality Test** An equality test for the relative risk can be expressed as

\[H_0: R = r_0\]

versus the alternative

\[H_a: R \neq r_0\]

where \(R = p_1 / p_2\) denotes the relative risk (for column 1 or column 2) and \(r_0\) denotes the null value. You can specify a null value by using the EQUAL(NULL=) relrisk-option; by default, the null value is 1.

The test statistic is computed by the method that you specify; by default, PROC FREQTAB uses the Wald test. For information about test statistic computation, see the subsections “Wald Test,” “Wald Modified Test,” “Farrington-Manning (Score) Test,” and “Likelihood Ratio Test” in this section.

For the Wald and score methods, the test statistics \(z\) have standard normal distributions under the null hypothesis. For the likelihood ratio test, the test statistic \(G^2\) has a chi-square distribution with 1 degree of freedom under the null hypothesis.

When the test statistic \(z\) is greater than 0, PROC FREQTAB displays the right-sided \(p\)-value, which is the probability of a larger value occurring under the null hypothesis. The one-sided \(p\)-value can be expressed as

\[P_1 = \begin{cases} \text{Prob}(Z > z) & \text{if } z > 0 \\ \text{Prob}(Z < z) & \text{if } z \leq 0 \end{cases}\]

where \(Z\) has a standard normal distribution. The two-sided \(p\)-value is computed as \(P_2 = 2 \times P_1\).

**Noninferiority Test** A noninferiority test for the relative risk can be expressed as

\[H_0: R \leq \delta\]

versus the alternative

\[H_a: R > \delta\]

where \(R = p_1 / p_2\) denotes the relative risk (for column 1 or column 2) and \(\delta\) denotes the noninferiority margin (limit). You can specify the margin by using the MARGIN= relrisk-option; by default, the noninferiority
margin is 0.8. The noninferiority margin for a relative risk test should be less than 1. Rejection of the null hypothesis indicates that the row 1 risk is not inferior to the row 2 risk. For more information, see Chow, Shao, and Wang (2008).

The test statistic $z$ is computed by the method that you specify. For information about test statistic computation, see the subsections “Wald Test,” “Wald Modified Test,” “Farrington-Manning (Score) Test,” and “Likelihood Ratio Test” in this section. The test statistic $z$ is computed by using the noninferiority margin (limit) as the null value of the relative risk. Under the null hypothesis, the test statistic has a standard normal distribution. The $p$-value for the noninferiority test is the right-sided $p$-value (the probability that $Z > z$).

As part of the noninferiority analysis, PROC FREQTAB also provides confidence limits for the relative risk. The confidence coefficient is $100(1 - 2\alpha)\%$ (Schuirmann 1999). The confidence level $\alpha$ is determined by the ALPHA= option in the TABLES statement; by default, ALPHA=0.05, which produces 90% confidence limits for the noninferiority analysis. You can compare the confidence limits to the value of the noninferiority limit $\delta$.

**Superiority Test**

A superiority test for the relative risk can be expressed as

$$H_0: R \leq \delta$$

versus the alternative

$$H_a: R > \delta$$

where $R = p_1 / p_2$ denotes the relative risk (for column 1 or column 2) and $\delta$ denotes the superiority margin (limit). You can specify the margin by using the MARGIN= relrisk-option; by default, the superiority margin is 1.25. The superiority margin for a relative risk test should be greater than 1. Rejection of the null hypothesis indicates that the row 1 risk is superior to the row 2 risk. For more information, see Chow, Shao, and Wang (2008).

The test statistic $z$ is computed by using the superiority margin (limit) as the null value of the relative risk. Under the null hypothesis, the test statistic has a standard normal distribution. The $p$-value for the superiority test is the right-sided $p$-value (the probability that $Z > z$).

The computations for the superiority analysis are the same as the computations for the noninferiority analysis, which are described in the subsection “Noninferiority Test” in this section.

**Equivalence Test**

An equivalence test for the relative risk can be expressed as

$$H_0: R \leq \delta_L \text{ or } R \geq \delta_U$$

versus the alternative

$$H_a: \delta_L < R < \delta_U$$

where $\delta_L$ is the lower margin and $\delta_U$ is the upper margin. Rejection of the null hypothesis indicates that the two risks are equivalent. For more information, see Chow, Shao, and Wang (2008).

You can specify the margins by using the MARGIN= relrisk-option; by default, the lower margin is 0.8 and the upper margin is 1.25. If you specify a single margin value, PROC FREQTAB uses this value as the lower margin for the equivalence test and computes the upper margin as the inverse of the lower margin.
PROC FREQTAB computes two one-sided tests (TOST) for equivalence analysis (Schuirmann 1987), which include a right-sided test for the lower margin $\delta_L$ and a left-sided test for the upper margin $\delta_U$. The lower test statistic uses the lower margin as the null relative risk value, and the $p$-value is the right-sided probability ($Z > z_L$). The upper test statistic uses the upper margin as the null value, and the $p$-value is the left-sided probability ($Z < z_U$). The overall $p$-value is taken to be the larger of the two $p$-values for the lower and upper tests.

The test statistics are computed by the method that you specify. For more information about the test statistic computation, see the subsections “Wald Test,” “Wald Modified Test,” “Farrington-Manning (Score) Test,” and “Likelihood Ratio Test” in this section.

As part of the equivalence analysis, PROC FREQTAB also provides confidence limits for the relative risk. The confidence coefficient is $100(1 - 2\alpha)$% (Schuirmann 1999). The confidence level $\alpha$ is determined by the ALPHA= option in the TABLES statement; by default, ALPHA=0.05, which produces 90% confidence limits for the equivalence analysis. You can compare the confidence limits to the equivalence limits, which are $\delta_L$ and $\delta_U$.

**Wald Test**

The Wald test statistic (which is based on a log transformation of the relative risk) is computed as $z(r_0) = (\log(\hat{r}) - \log(r_0))/\sqrt{v}$, where $\hat{r}$ is the relative risk estimate ($\hat{p}_1/\hat{p}_2$), $r_0$ is the null value of the relative risk, and

$$v = \text{Var}(\log(\hat{r})) = 1/n_{11} + 1/(n_{21} - 1/n_1) - 1/n_2.$$  

The null value is determined by the type of test (equality, noninferiority, superiority, or equivalence) and the null or margin values that you specify. The side of the $p$-value and the interpretation of the test are also determined by the type of test; for more information, see the subsections “Equality Test,” “Noninferiority Test,” “Superiority Test,” and “Equivalence Test” in this section.

**Wald Modified Test**

The Wald modified test statistic is computed by replacing the $n_{ij}$ with $(n_{ij} + 0.5)$ and the $n_i$ with $(n_i + 0.5)$ in the relative risk estimator $\hat{R}$ and the variance $v$. The test statistic is computed as $z(r_0) = (\log(\hat{r}) - \log(r_0))/\sqrt{v}$, where $r_0$ is the null value of the relative risk,

$$\hat{r} = \hat{p}_1/\hat{p}_2 = \frac{(n_{11} + 0.5)/(n_1 + 0.5)}{(n_{21} + 0.5)/(n_2 + 0.5)}$$

$$v = \text{Var}(\log(\hat{r})) = 1/(n_{11} + 0.5) + 1/(n_{21} + 0.5) - 1/(n_1 + 0.5) - 1/(n_2 + 0.5)$$

The null value is determined by the type of test (equality, noninferiority, superiority, or equivalence) and the null or margin values that you specify. The side of the $p$-value and the interpretation of the test are also determined by the type of test; for more information, see the subsections “Equality Test,” “Noninferiority Test,” “Superiority Test,” and “Equivalence Test” in this section.

**Farrington-Manning (Score) Test**

The relative risk score test statistic (Miettinen and Nurminen 1985; Farrington and Manning 1990) for the null value $r_0$ is computed as

$$z(r_0) = (\hat{p}_1 - r_0 \hat{p}_2) / \text{se}(r_0)$$
where
\[ \text{se}(r_0) = \sqrt{\frac{\hat{p}_1(1 - \hat{p}_1)}{n_1} + \frac{r_0^2 \hat{p}_2(1 - \hat{p}_2)}{n_2}}. \]

where \( \hat{p}_1 \) and \( \hat{p}_2 \) are the maximum likelihood estimates of \( p_1 \) and \( p_2 \) under the null value \( r_0 \). Expressions for the maximum likelihood estimates \( \hat{p}_1 \) and \( \hat{p}_2 \) are given in the subsection “Score Confidence Limits” in this section.

The null value is determined by the type of test (equality, noninferiority, superiority, or equivalence) and the null or margin values that you specify. The side of the \( p \)-value and the interpretation of the test are also determined by the type of test; for more information, see the subsections “Equality Test,” “Noninferiority Test,” “Superiority Test,” and “Equivalence Test” in this section.

Likelihood Ratio Test

The likelihood ratio statistic for the null relative risk value \( r_0 \) is computed as
\[
G^2(r_0) = 2 \left( n_{11} \log(\hat{p}_1/\hat{p}_1) + n_{12} \log((1-\hat{p}_1)/(1-\hat{p}_1)) + n_{21} \log(\hat{p}_2/\hat{p}_2) + n_{22} \log((1-\hat{p}_2)/(1-\hat{p}_2)) \right)
\]
where \( \hat{p}_1 \) and \( \hat{p}_2 \) are the maximum likelihood estimates of \( p_1 \) and \( p_2 \) under the null value \( r_0 \). Expressions for the maximum likelihood estimates \( \hat{p}_1 \) and \( \hat{p}_2 \) are given in the subsection “Score Confidence Limits” in this section. For more information, see Miettinen and Nurminen (1985) and Miettinen (1985, chapter 13).

PROC FREQTAB computes the likelihood ratio test statistic \( z(r_0) \) for the noninferiority, superiority, and equivalence tests as \( \sqrt{G^2(r_0)} \), where the sign is positive if the estimate is greater than the null value (\( \hat{r} \geq r_0 \)) and negative otherwise (\( \hat{r} < r_0 \)).

The null value is determined by the type of test (equality, noninferiority, superiority, or equivalence) and the null or margin values that you specify. The side of the \( p \)-value and the interpretation of the test are also determined by the type of test; for more information, see the subsections “Equality Test,” “Noninferiority Test,” “Superiority Test,” and “Equivalence Test” in this section.

Cochran-Armitage Test for Trend

The TREND option in the TABLES statement provides the Cochran-Armitage test for trend, which tests for trend in binomial proportions across levels of a single factor or covariate. This test is appropriate for a two-way table where one variable has two levels and the other variable is ordinal. The two-level variable represents the response, and the other variable represents an explanatory variable with ordered levels. When the two-way has two columns and \( R \) rows, PROC FREQTAB tests for trend across the \( R \) levels of the row variable, and the binomial proportion is computed as the proportion of observations in the first column. When the table has two rows and \( C \) columns, PROC FREQTAB tests for trend across the \( C \) levels of the column variable, and the binomial proportion is computed as the proportion of observations in the first row.

The trend test is based on the regression coefficient for the weighted linear regression of the binomial proportions on the scores of the explanatory variable levels. For more information, see Margolin (1988) and Agresti (2002). If the table has two columns and \( R \) rows, the trend test statistic is computed as
\[
T = \sum_{i=1}^{R} n_{i1} (R_i - \bar{R}) / \sqrt{p_1 (1 - p_1) s^2}
\]
where \( R_i \) is the score of row \( i \), \( \bar{R} \) is the average row score, and
\[
s^2 = \sum_{i=1}^{R} n_{i1} (R_i - \bar{R})^2
\]
The`SCORES`= option in the TABLES statement determines the type of row scores used in computing the trend test (and other score-based statistics). The default is `SCORES=TABLE`. For more information, see the section “Scores” on page 201. For character variables, the table scores for the row variable are the row numbers (for example, 1 for the first row, 2 for the second row, and so on). For numeric variables, the table score for each row is the numeric value of the row level. When you perform the trend test, the explanatory variable might be numeric (for example, dose of a test substance), and the variable values might be appropriate scores. If the explanatory variable has ordinal levels that are not numeric, you can assign meaningful scores to the variable levels. Sometimes equidistant scores, such as the table scores for a character variable, might be appropriate. For more information on choosing scores for the trend test, see Margolin (1988).

The null hypothesis for the Cochran-Armitage test is no trend, which means that the binomial proportion \( p_{i1} = n_{i1}/n_i \) is the same for all levels of the explanatory variable. Under the null hypothesis, the trend statistic has an asymptotic standard normal distribution.

PROC FREQTAB computes one-sided and two-sided \( p \)-values for the trend test. When the test statistic is greater than its null hypothesis expected value of 0, PROC FREQTAB displays the right-sided \( p \)-value, which is the probability of a larger value of the statistic occurring under the null hypothesis. A small right-sided \( p \)-value supports the alternative hypothesis of increasing trend in proportions from row 1 to row \( R \). When the test statistic is less than or equal to 0, PROC FREQTAB displays the left-sided \( p \)-value. A small left-sided \( p \)-value supports the alternative of decreasing trend.

The one-sided \( p \)-value for the trend test is computed as

\[
P_1 = \begin{cases} 
\Pr(Z > T) & \text{if } T > 0 \\
\Pr(Z < T) & \text{if } T \leq 0
\end{cases}
\]

where \( Z \) has a standard normal distribution. The two-sided \( p \)-value is computed as

\[
P_2 = \Pr(|Z| > |T|)
\]

PROC FREQTAB also provides exact \( p \)-values for the Cochran-Armitage trend test. You can request the exact test by specifying the TREND option in the EXACT statement. See the section “Exact Statistics” on page 271 for more information.

**Jonckheere-Terpstra Test**

The `JT` option in the TABLES statement provides the Jonckheere-Terpstra test, which is a nonparametric test for ordered differences among classes. It tests the null hypothesis that the distribution of the response variable does not differ among classes. It is designed to detect alternatives of ordered class differences, which can be expressed as \( \tau_1 \leq \tau_2 \leq \cdots \leq \tau_R \) (or \( \tau_1 \geq \tau_2 \geq \cdots \geq \tau_R \)), with at least one of the inequalities being strict, where \( \tau_i \) denotes the effect of class \( i \). For such ordered alternatives, the Jonckheere-Terpstra test can be preferable to tests of more general class difference alternatives, such as the Kruskal–Wallis test (produced by the WILCOXON option in the NPAR1WAY procedure). See Pirie (1983) and Hollander and Wolfe (1999) for more information about the Jonckheere-Terpstra test.

The Jonckheere-Terpstra test is appropriate for a two-way table in which an ordinal column variable represents the response. The row variable, which can be nominal or ordinal, represents the classification variable. The levels of the row variable should be ordered according to the ordering you want the test to detect. The order of variable levels is determined by the ORDER= option in the PROC FREQTAB statement. The default
is ORDER=INTERNAL, which orders by unformatted values. For more information about how to order variable levels, see the ORDER= option.

The Jonckheere-Terpstra test statistic is computed by first forming $R(R-1)/2$ Mann-Whitney counts $M_{i,j'}$, where $i < i'$, for pairs of rows in the contingency table,

$$M_{i,j'} = \begin{cases} \text{number of times } X_{i,j} < X_{i',j'}, & j = 1, \ldots, n_i; \ j' = 1, \ldots, n_{i'} \end{cases} + \frac{1}{2} \begin{cases} \text{number of times } X_{i,j} = X_{i',j'}, & j = 1, \ldots, n_i; \ j' = 1, \ldots, n_{i'} \end{cases}$$

where $X_{i,j}$ is response $j$ in row $i$. The Jonckheere-Terpstra test statistic is computed as

$$J = \sum_{1 \leq i < i' \leq R} M_{i,i'}$$

This test rejects the null hypothesis of no difference among classes for large values of $J$. Asymptotic $p$-values for the Jonckheere-Terpstra test are obtained by using the normal approximation for the distribution of the standardized test statistic. The standardized test statistic is computed as

$$J^* = (J - E_0(J)) / \sqrt{\text{Var}_0(J)}$$

where $E_0(J)$ and $\text{Var}_0(J)$ are the expected value and variance of the test statistic under the null hypothesis,

$$E_0(J) = \left(n^2 - \sum_i n_i^2\right)/4$$

$$\text{Var}_0(J) = A/72 + B / (36n(n-1)(n-2)) + C / (8n(n-1))$$

where

$$A = n(n-1)(2n+5) - \sum_i n_i(n_i-1)(2n_i+5) - \sum_j n_j(n_j-1)(2n_j+5)$$

$$B = \left(\sum_i n_i(n_i-1)(n_i-2)\right)\left(\sum_j n_j(n_j-1)(n_j-2)\right)$$

$$C = \left(\sum_i n_i(n_i-1)\right)\left(\sum_j n_j(n_j-1)\right)$$

PROC FREQTAB computes one-sided and two-sided $p$-values for the Jonckheere-Terpstra test. When the standardized test statistic is greater than its null hypothesis expected value of 0, PROC FREQTAB displays the right-sided $p$-value, which is the probability of a larger value of the statistic occurring under the null hypothesis. A small right-sided $p$-value supports the alternative hypothesis of increasing order from row 1 to row $R$. When the standardized test statistic is less than or equal to 0, PROC FREQTAB displays the left-sided $p$-value. A small left-sided $p$-value supports the alternative of decreasing order from row 1 to row $R$.

The one-sided $p$-value for the Jonckheere-Terpstra test, $P_1$, is computed as

$$P_1 = \begin{cases} \text{Prob}(Z > J^*) & \text{if } J^* > 0 \\ \text{Prob}(Z < J^*) & \text{if } J^* \leq 0 \end{cases}$$
where $Z$ has a standard normal distribution. The two-sided $p$-value, $P_2$, is computed as

$$P_2 = \text{Prob}(|Z| > |J^*|)$$

PROC FREQTAB also provides exact $p$-values for the Jonckheere-Terpstra test. You can request the exact test by specifying the JT option in the EXACT statement. See the section “Exact Statistics” on page 271 for more information.

**Tests and Measures of Agreement**

When you specify the AGREE option in the TABLES statement, PROC FREQTAB computes tests and measures of agreement for square tables (for which the number of rows equals the number of columns). By default, these statistics include McNemar’s test for $2 \times 2$ tables, Bowker’s symmetry test, the simple kappa coefficient, and the weighted kappa coefficient. For multiple strata ($n$-way tables, where $n > 2$), the AGREE option provides the overall simple and weighted kappa coefficients, in addition to tests for equal kappas (simple and weighted) among strata. For multiple strata of $2 \times 2$ tables, the AGREE option provides Cochran’s $Q$ test.

Optionally, PROC FREQTAB provides kappa tests and other agreement statistics. In addition to the asymptotic tests described in this section, PROC FREQTAB provides exact $p$-values for McNemar’s test, the simple kappa coefficient test, and the weighted kappa coefficient test. You can request these exact tests by specifying the corresponding options in the EXACT statement. For more information, see the section “Exact Statistics” on page 271.

The following sections provide the formulas that PROC FREQTAB uses to compute agreement statistics. For information about the use and interpretation of these statistics, see Agresti (2002, 2007); Fleiss, Levin, and Paik (2003), and the other references cited for each statistic.

**McNemar’s Test**

PROC FREQTAB computes McNemar’s test (McNemar 1947) for $2 \times 2$ tables when you specify the AGREE option. This test is appropriate when you are analyzing data from matched pairs of subjects with a dichotomous (yes-no) response. By default, the null hypothesis for McNemar’s test is marginal homogeneity, which can be expressed as $p_{1.} = p_{.1}$; this is equivalent to a discordant proportion ratio ($p_{12}/p_{21}$) of 1. The corresponding test statistic is computed as

$$Q_M = (n_{12} - n_{21})^2 / (n_{12} + n_{21})$$

Under the null hypothesis, $Q_M$ has an asymptotic chi-square distribution with 1 degree of freedom.

Optionally, you can specify the null ratio of discordant proportions ($p_{12}/p_{21}$) by using the AGREE(MNULLRATIO=) option. When the null ratio is $r$, McNemar’s test is computed as

$$Q_M(r) = (n_{12} - e_{12})^2 / e_{12} + (n_{21} - e_{21})^2 / e_{21}$$

where $e_{12} = D/(1 + 1/r)$, $e_{21} = D/(1 + r)$, and $D$ is the number of discordant pairs, $(n_{12} + n_{21})$. Under the null hypothesis, $Q_M(r)$ has an asymptotic chi-square distribution with 1 degree of freedom.

PROC FREQTAB also computes an exact $p$-value for McNemar’s test when you specify the MCNEM option in the EXACT statement.
**Bowker’s Symmetry Test**

The null hypothesis for Bowker’s symmetry test (Bowker 1948) is symmetric table-cell proportions, which can be expressed as \( p_{ij} = p_{ji} \) for all off-diagonal pairs of table cells. For \( 2 \times 2 \) tables, Bowker’s test is identical to McNemar’s test; therefore, PROC FREQTAB provides Bowker’s test only for square tables that are larger than \( 2 \times 2 \).

Bowker’s symmetry test is computed as

\[
Q_B = \sum_{i<j} \frac{(n_{ij} - n_{ji})^2}{n_{ij} + n_{ji}}
\]

For large samples, \( Q_B \) has an asymptotic chi-square distribution with \( R(R-1)/2 \) degrees of freedom under the null hypothesis of symmetry, where \( R \) is the dimension of the square, two-way table.

By default, the number of degrees of freedom for this test \( (R(R-1)/2) \) is the number of off-diagonal table-cell comparisons. You can specify the number of degrees of freedom in the `AGREE(DFSYM=)` option. Alternatively, you can specify the `AGREE(DFSYM=ADJUST)` option, which reduces the degrees of freedom by the number of off-diagonal table-cell pairs that have a total frequency of 0. For more information, see Hoenig, Morgan, and Brown (1995).

**Exact Symmetry Test**  When you specify the `SYMMETRY` option in the EXACT statement, PROC FREQTAB provides an exact symmetry test by using the method of Krauth (1973). This exact test is computed by conditioning on the observed frequency sums of the complementary off-diagonal table-cell pairs \( (n_{ij} + n_{ji}) \). PROC FREQTAB evaluates the symmetry test statistic for all tables in the reference set, which includes all possible tables in which the frequency sums of the off-diagonal table-cell pairs match the corresponding frequency sums in the observed table. The exact \( p \)-value is then computed as the sum of the table probabilities for those tables for which the symmetry test statistic is greater than or equal to the observed test statistic. The table probabilities are computed as products of \( R(R-1)/2 \) binomial probabilities (which correspond to the off-diagonal table-cell pairs in tables of dimension \( R \)) by using the binomial proportion 0.5 under the null hypothesis of symmetry. For more information, see the section “Exact Statistics” on page 271.

Alternatively, you can request a Monte Carlo estimate of the exact \( p \)-value by specifying the `SYMMETRY` option together with the `MC computation-option` in the EXACT statement. The Monte Carlo computation for the exact symmetry test is conditional on the same reference set that the exact test uses (tables in which the frequency sums of the off-diagonal table-cell pairs match the corresponding sums in the observed table). For more information, see the section “Monte Carlo Estimation” on page 275.

**Simple Kappa Coefficient**

The simple kappa coefficient (Cohen 1960) is a measure of interrater agreement. PROC FREQTAB computes the simple kappa coefficient as

\[
\hat{\kappa} = \frac{(P_o - P_e)}{(1 - P_e)}
\]

where \( P_o = \sum_i p_{ii} \) and \( P_e = \sum_i p_i \cdot p_j \). The component \( P_o \) is the proportion of observed agreement, and the component \( P_e \) represents the proportion of chance-expected agreement.

If the two response variables are viewed as two independent ratings of the \( n \) subjects, the kappa coefficient is +1 when there is complete agreement of the raters. When the observed agreement exceeds the chance-expected agreement, the kappa coefficient is positive, and its magnitude reflects the strength of agreement. When the observed agreement is less than the chance-expected agreement, the kappa coefficient is negative. The minimum value of kappa is between –1 and 0, depending on the marginal proportions of the table.
PROC FREQTAB computes the asymptotic variance of the simple kappa coefficient as

\[ \text{Var}(\hat{\kappa}) = \frac{(A + B - C)}{(1 - P_e)^2 n} \]

where

\[ A = \sum_i p_{ii} (1 - (p_{i.} + p_{.i})(1 - \hat{\kappa}))^2 \]

\[ B = (1 - \hat{\kappa})^2 \sum_{i \neq j} p_{ij} (p_{i.} + p_{.j})^2 \]

\[ C = (\hat{\kappa} - P_e(1 - \hat{\kappa}))^2 \]

For more information, see Fleiss, Cohen, and Everitt (1969).

Confidence limits for the simple kappa coefficient are computed as

\[ \hat{\kappa} \pm \left( z_{\alpha/2} \times \sqrt{\text{Var}(\hat{\kappa})} \right) \]

where \( z_{\alpha/2} \) is the 100\((1 - \alpha/2)\)th percentile of the standard normal distribution. The value of \( \alpha \) is determined by the ALPHA= option; by default \( \text{ALPHA}=0.05 \), which produces 95\% confidence limits.

PROC FREQTAB provides an asymptotic test for the simple kappa coefficient. By default, the null hypothesis value of kappa is 0; alternatively, you can specify a nonzero null value of kappa (by using the AGREE(NULLKAPPA=) option in the TABLES statement). When the null value of kappa is nonzero, PROC FREQTAB computes the test statistic as

\[ z = \frac{(\hat{\kappa} - \kappa_0)}{\sqrt{\text{Var}(\hat{\kappa})}} \]

where \( \kappa_0 \) is the null value that you specify and \( \text{Var}(\hat{\kappa}) \) is the variance of the kappa coefficient.

When the null value of kappa is 0, PROC FREQTAB computes the test statistic as

\[ z = \frac{\hat{\kappa}}{\sqrt{\text{Var}_0(\hat{\kappa})}} \]

where \( \text{Var}_0(\hat{\kappa}) \) is the variance of the kappa coefficient under the null hypothesis (that kappa is 0) and is computed as

\[ \text{Var}_0(\hat{\kappa}) = \left( P_e + P_e^2 - \sum_i p_{i.} p_{.i} (p_{i.} + p_{.i}) \right) / (1 - P_e)^2 n \]

This test statistic has an asymptotic standard normal distribution under the null hypothesis. For more information, see Fleiss, Levin, and Paik (2003).

PROC FREQTAB also provides an exact test for the simple kappa coefficient. You can request the exact test by specifying the KAPPA or AGREE option in the EXACT statement. For more information, see the section “Exact Statistics” on page 271.
Kappa Details When you specify the AGREE(KAPPADETAILS) option, PROC FREQTAB displays the “Kappa Details” table, which includes the observed agreement $P_o$, chance-expected agreement $P_e$, maximum kappa, and $B_n$ measure.

The maximum kappa, which is the maximum possible value of the kappa coefficient given the marginal proportions of the two-way table, is computed as

$$\max(\kappa) = \frac{(\max(P_o) - P_e)}{(1 - P_e)}$$

where

$$\max(P_o) = \left( \sum_i \min(n_i, n_{..}) \right) / n$$

The $B_n$ measure (Bangdiwala 1988; Bangdiwala et al. 2008) is computed as

$$B_n = \left( \frac{\sum_i n_i^2}{\left( \sum_i \sum_j n_{ij} \right) \left( \sum_i n_i \right) \left( \sum_j n_{..j} \right)} \right)$$

For $2 \times 2$ tables, the “Kappa Details” table also includes the prevalence index and the bias index. The prevalence index is the absolute difference between the agreement proportions, $|p_{11} - p_{22}|$. The bias index is the absolute difference between the disagreement proportions, $|p_{12} - p_{21}|$. For more information, see Sim and Wright (2005) and Byrt, Bishop, and Carlin (1993).

Weighted Kappa Coefficient

The weighted kappa coefficient is a generalization of the simple kappa coefficient that uses weights to quantify the relative differences between categories. For $2 \times 2$ tables, the weighted kappa coefficient is equivalent to the simple kappa coefficient; therefore, PROC FREQTAB displays the weighted kappa coefficient only for tables larger than $2 \times 2$. PROC FREQTAB computes the kappa weights from the column scores, by using either Cicchetti-Allison weights or Fleiss-Cohen weights, both of which are described in the section “Kappa Weights” on page 259. The kappa weights $w_{ij}$ are constructed so that $0 \leq w_{ij} < 1$ for all $i \neq j$, $w_{ii} = 1$ for all $i$, and $w_{ij} = w_{ji}$. The weighted kappa coefficient is computed as

$$\hat{\kappa}_w = \frac{P_o(w) - P_e(w)}{(1 - P_e(w))}$$

where

$$P_o(w) = \sum_i \sum_j w_{ij} p_{ij}$$

$$P_e(w) = \sum_i \sum_j w_{ij} p_{i..} p_{..j}$$

The component $P_o(w)$ is the proportion of observed (weighted) agreement, and the component $P_e(w)$ represents the proportion of chance-expected (weighted) agreement. When you specify the AGREE(WTKAPDETAILS) option, PROC FREQTAB displays these components in the “Weighted Kappa Details” table.
PROC FREQTAB computes the asymptotic variance of the weighted kappa coefficient as

$$\text{Var}(\hat{k}_w) = \left( \sum_i \sum_j p_{ij} \left( w_{ij} - (\bar{w}_i + \bar{w}_j)(1 - \hat{k}_w) \right)^2 - (\hat{k}_w - P_{e(w)}(1 - \hat{k}_w))^2 \right) / (1 - P_{e(w)})^2 n$$

where

$$\bar{w}_i = \sum_j p_{j \cdot} w_{ij}$$

$$\bar{w}_j = \sum_i p_{\cdot i} w_{ij}$$

For more information, see Fleiss, Cohen, and Everitt (1969).

Confidence limits for the weighted kappa coefficient are computed as

$$\hat{k}_w \pm \left( z_{\alpha/2} \times \sqrt{\text{Var}(\hat{k}_w)} \right)$$

where $z_{\alpha/2}$ is the 100$(1 - \alpha/2)$th percentile of the standard normal distribution. The value of $\alpha$ is determined by the ALPHA= option; by default ALPHA=0.05, which produces 95% confidence limits.

PROC FREQTAB provides an asymptotic test for the weighted kappa coefficient. By default, the null hypothesis value of weighted kappa is 0; alternatively, you can specify a nonzero null value of weighted kappa (by using the AGREE(NULLWTKAPPA=) option in the TABLES statement). When the null value of weighted kappa is nonzero, PROC FREQTAB computes the test statistic as

$$z = (\hat{k}_w - \kappa_{w(0)}) / \sqrt{\text{Var}(\hat{k}_w)}$$

where $\kappa_{w(0)}$ is the null value that you specify and $\text{Var}(\hat{k}_w)$ is the variance of the weighted kappa coefficient.

When the null value of weighted kappa is 0, PROC FREQTAB computes the test statistic as

$$z = \hat{k}_w / \sqrt{\text{Var}_0(\hat{k}_w)}$$

where $\text{Var}_0(\hat{k}_w)$ is the variance of the weighted kappa coefficient under the null hypothesis (that weighted kappa is 0) and is computed as

$$\text{Var}_0(\hat{k}_w) = \left( \sum_i \sum_j p_{i \cdot} p_{\cdot j} \left( w_{ij} - (\bar{w}_i + \bar{w}_j)^2 - P_{e(w)}^2 \right) \right) / (1 - P_{e(w)})^2 n$$

This test statistic has an asymptotic standard normal distribution under the null hypothesis. For more information, see Fleiss, Levin, and Paik (2003).

PROC FREQTAB also provides an exact test for the weighted kappa coefficient. You can request the exact test by specifying the KAPPA or AGREE option in the EXACT statement. For more information, see the section “Exact Statistics” on page 271.
**Kappa Weights** PROC FREQTAB computes kappa coefficient weights by using the column scores and one of the two available weight types. The column scores are determined by the `SCORES=` option in the TABLES statement. The two available types of kappa weights are Cicchetti-Allison and Fleiss-Cohen weights. By default, PROC FREQTAB uses Cicchetti-Allison weights. If you specify the AGREE(WT=FC) option, PROC FREQTAB uses Fleiss-Cohen weights to compute the weighted kappa coefficient.

PROC FREQTAB computes Cicchetti-Allison kappa coefficient weights as

\[ w_{ij} = 1 - \frac{|C_i - C_j|}{C_C - C_1} \]

where \( C_i \) is the score for column \( i \) and \( C \) is the number of categories or columns. For more information, see Cicchetti and Allison (1971).

The `SCORES=` option in the TABLES statement determines the type of column scores used to compute the kappa weights (and other score-based statistics). The default is `SCORES=TABLE`. For more information, see the section “Scores” on page 201. For numeric variables, table scores are the values of the variable levels. You can assign numeric values to the levels in a way that reflects their level of similarity. For example, suppose you have four levels and order them according to similarity. If you assign them values of 0, 2, 4, and 10, the Cicchetti-Allison kappa weights take the following values: \( w_{12} = 0.8, w_{13} = 0.6, w_{14} = 0, w_{23} = 0.8, w_{24} = 0.2, \) and \( w_{34} = 0.4 \). Note that when there are only two categories (that is, \( C = 2 \)), the weighted kappa coefficient is identical to the simple kappa coefficient.

If you specify the AGREE(WT=FC) option in the TABLES statement, PROC FREQTAB computes Fleiss-Cohen kappa coefficient weights as

\[ w_{ij} = 1 - \frac{(C_i - C_j)^2}{(C_C - C_1)^2} \]

For more information, see Fleiss and Cohen (1973).

For the preceding example, the Fleiss-Cohen kappa weights are \( w_{12} = 0.96, w_{13} = 0.84, w_{14} = 0, w_{23} = 0.96, w_{24} = 0.36, \) and \( w_{34} = 0.64 \).

**Prevalence-Adjusted Bias-Adjusted Kappa** When you specify the AGREE(PABAK) option, PROC FREQTAB provides the prevalence-adjusted bias-adjusted kappa coefficient (PABAK) (Byrt, Bishop, and Carlin 1993). This coefficient is computed as

\[ \hat{\kappa}_a = \left( P_o - 1/R \right) / (1 - 1/R) \]

where \( P_o = \sum_i p_{ii} \) and \( R \) is the dimension of the square, two-way table. The component \( P_o \) is the proportion of observed agreement, and the component \( 1/R \) represents the chance-expected agreement. When the table is \( 2 \times 2 \), \( \hat{\kappa}_a = 2P_o - 1 \). For more information, see Sim and Wright (2005), Xie (2013), and Holley and Guilford (1964).

PROC FREQTAB computes the variance of the prevalence-adjusted bias-adjusted kappa as

\[ \text{Var}(\hat{\kappa}_a) = \left( R/(R - 1) \right)^2 (P_o(1 - P_o)/n) \]

Confidence limits are computed as

\[ \hat{\kappa}_a \pm (z_{\alpha/2} \times \sqrt{\text{Var}(\hat{\kappa}_a)}) \]

where \( z_{\alpha/2} \) is the \( 100(1 - \alpha/2) \)th percentile of the standard normal distribution. The value of \( \alpha \) is determined by the `ALPHA=` option; by default `ALPHA=0.05`, which produces 95% confidence limits.
AC1 Agreement Coefficient
When you specify the AGREE(AC1) option, PROC FREQTAB provides Gwet’s first-order agreement coefficient, AC1 (Gwet 2008). This coefficient is computed as

\[
\hat{\gamma} = (P_o - P_{e(\gamma)}) / (1 - P_{e(\gamma)})
\]

where \( P_o = \sum_i p_{ii} \), \( P_e = \sum_i e_i (1 - e_i) / (R - 1) \), and \( e_i = (p_{i.} + p_{.i}) / 2 \) The component \( P_o \) is the proportion of observed agreement, and the component \( P_{e(\gamma)} \) represents the proportion of chance-expected agreement. For more information, see Xie (2013) and Blood and Spratt (2007).

PROC FREQTAB computes the variance of AC1 as

\[
\text{Var}(\hat{\gamma}) = \left( P_o (1 - P_o) - 4(1 - \hat{\gamma}) A + 4(1 - \hat{\gamma}^2) B \right) / n (1 - P_{e(\gamma)})^2
\]

where

\[
A = \sum_i p_{ii} (1 - e_i) / (R - 1) - P_o P_{e(\gamma)}
\]

\[
B = \sum_i \sum_j p_{ij} (1 - (e_i + e_j) / 2)^2 / (R - 1)^2 - P_{e(\gamma)}^2
\]

Confidence limits for AC1 are computed as

\[
\hat{\gamma} \pm \left( z_{\alpha/2} \times \sqrt{\text{Var}(\hat{\gamma})} \right)
\]

where \( z_{\alpha/2} \) is the 100\( (1 - \alpha/2) \)th percentile of the standard normal distribution. The value of \( \alpha \) is determined by the ALPHA= option; by default ALPHA=0.05, which produces 95% confidence limits.

Overall Kappa Coefficient
When there are multiple strata, PROC FREQTAB combines the stratum-level estimates of kappa into an overall estimate of the supposed common value of kappa. Assume there are \( q \) strata, indexed by \( h = 1, 2, \ldots, q \), and let \( \text{Var}(\hat{k}_h) \) denote the variance of \( \hat{k}_h \). The estimate of the overall kappa coefficient is computed as

\[
\hat{k}_T = \frac{\sum_{h=1}^q \hat{k}_h / \text{Var}(\hat{k}_h)}{\sum_{h=1}^q 1 / \text{Var}(\hat{k}_h)}
\]

For more information, see Fleiss, Levin, and Paik (2003).

PROC FREQTAB computes an estimate of the overall weighted kappa in the same way.

Tests for Equal Kappa Coefficients
When there are multiple strata, the following chi-square statistic tests whether the stratum-level values of kappa are equal:

\[
Q_K = \sum_{h=1}^q (\hat{k}_h - \hat{k}_T)^2 / \text{Var}(\hat{k}_h)
\]

Under the null hypothesis of equal kappas for the \( q \) strata, \( Q_K \) has an asymptotic chi-square distribution with \( q - 1 \) degrees of freedom. See Fleiss, Levin, and Paik (2003) for more information. PROC FREQTAB computes a test for equal weighted kappa coefficients in the same way.
Cochran’s Q Test

Cochran’s Q is computed for multiway tables when each variable has two levels, that is, for $2 \times 2 \cdots \times 2$ tables. Cochran’s Q statistic is used to test the homogeneity of the one-dimensional margins. Let $m$ denote the number of variables and $N$ denote the total number of subjects. Cochran’s Q statistic is computed as

$$Q_C = (m - 1) \left( \frac{m \sum_{j=1}^{m} T_j^2 - T^2}{mT - \sum_{k=1}^{N} S_k^2} \right)$$

where $T_j$ is the number of positive responses for variable $j$, $T$ is the total number of positive responses over all variables, and $S_k$ is the number of positive responses for subject $k$. Under the null hypothesis, Cochran’s Q has an asymptotic chi-square distribution with $m-1$ degrees of freedom. For more information, see Cochran (1950). When there are only two binary response variables ($m=2$), Cochran’s Q simplifies to McNemar’s test. When there are more than two response categories, you can test for marginal homogeneity by using the repeated measures capabilities of the CATMOD procedure.

Tables with Zero-Weight Rows or Columns

The AGREE statistics are defined only for square tables, where the number of rows equals the number of columns; if a table is not square, PROC FREQTAB does not compute AGREE statistics for the table. In the kappa statistic framework, where two independent raters assign ratings to each of $n$ subjects, suppose one of the raters does not use all possible $r$ rating levels. If the corresponding table contains $r$ rows but only $r-1$ columns, the table is not square and PROC FREQTAB does not compute AGREE statistics. To create a square table in this situation, you can use the ZEROS option in the WEIGHT statement, which includes zero-weight observations in the analysis. You can include zero-weight observations in the input data table to represent any rating levels that are not used by a rater, so that the input data table has at least one observation for each possible rater and rating combination. When you use this input data table and specify the ZEROS option, the analysis includes all rating levels (even when all levels are not actually assigned by both raters). The resulting table (of rater 1 by rater 2) is a square table, and AGREE statistics can be computed.

For more information, see the description of the ZEROS option in the WEIGHT statement. By default, PROC FREQTAB does not process observations that have weights of 0 because these observations do not contribute to the total frequency count, and because many of the tests and measures of association are undefined for tables that contain zero-weight rows or columns. However, kappa statistics are defined for tables that contain zero-weight rows or columns, and the ZEROS option enables you to input zero-weight observations and construct the tables needed to compute kappa statistics.

Cochran-Mantel-Haenszel Statistics

The CMH option in the TABLES statement gives a stratified statistical analysis of the relationship between the row and column variables after controlling for the strata variables in a multiway table. For example, for the table request A*B*C*D, the CMH option provides an analysis of the relationship between C and D, after controlling for A and B. The stratified analysis provides a way to adjust for the possible confounding effects of A and B without being forced to estimate parameters for them.

The CMH analysis produces Cochran-Mantel-Haenszel statistics, which include the correlation statistic, the ANOVA (row mean scores) statistic, and the general association statistic. For $2 \times 2$ tables, the CMH option also provides Mantel-Haenszel and logit estimates of the common odds ratio and the common relative risks, in addition to the Breslow-Day test for homogeneity of the odds ratios.

Exact statistics are also available for stratified $2 \times 2$ tables. If you specify the EQOR option in the EXACT statement, PROC FREQTAB provides Zelen’s exact test for equal odds ratios. If you specify the COMOR
option in the EXACT statement, PROC FREQTAB provides exact confidence limits for the common odds ratio and an exact test that the common odds ratio equals one.

Let the number of strata be denoted by \( q \), indexing the strata by \( h = 1, 2, \ldots, q \). Each stratum contains a contingency table with \( X \) representing the row variable and \( Y \) representing the column variable. For table \( h \), denote the cell frequency in row \( i \) and column \( j \) by \( n_{hij} \), with corresponding row and column marginal totals denoted by \( n_{hi} \) and \( n_{hj} \), and the overall stratum total by \( n_h \).

Because the formulas for the Cochran-Mantel-Haenszel statistics are more easily defined in terms of matrices, the following notation is used. Vectors are presumed to be column vectors unless they are transposed (\(^t\)).

\[
\begin{align*}
n'_{hi} &= (n'_{h1}, n'_{h2}, \ldots, n'_{hC}) \quad (1 \times C) \\
n'_h &= (n'_{h1}, n'_{h2}, \ldots, n'_{hR}) \quad (1 \times RC) \\
p_{hi} &= n_{hi} / n_h \quad (1 \times 1) \\
p_{h-j} &= n_{h-j} / n_h \quad (1 \times 1) \\
P'_h &= (p_{h1}, p_{h2}, \ldots, p_{hC}) \quad (1 \times R) \\
P'_{h-} &= (p_{h1}, p_{h2}, \ldots, p_{hR}) \quad (1 \times C)
\end{align*}
\]

Assume that the strata are independent and that the marginal totals of each stratum are fixed. The null hypothesis, \( H_0 \), is that there is no association between \( X \) and \( Y \) in any of the strata. The corresponding model is the multiple hypergeometric; this implies that, under \( H_0 \), the expected value and covariance matrix of the frequencies are, respectively,

\[
\begin{align*}
m_h &= E[n_h \mid H_0] = n_h (P_{h-} \otimes P_{-*}) \\
\text{Var}[n_h \mid H_0] &= c \left( (D_{P_{h-}} - P_{h-}P_{-*}) \otimes (D_{P_{h*}} - P_{h*}P'_{h*}) \right)
\end{align*}
\]

where

\[
c = n_h^2 / (n_h - 1)
\]

and where \( \otimes \) denotes Kronecker product multiplication and \( D_a \) is a diagonal matrix with the elements of \( a \) on the main diagonal.

The generalized CMH statistic (Landis, Heyman, and Koch 1978) is defined as

\[
Q_{CMH} = G'V_G^{-1}G
\]

where

\[
G = \sum_h B_h(n_h - m_h)
\]

\[
V_G = \sum_h B_h \left( \text{Var}[n_h \mid H_0] \right) B'_h
\]

and where

\[
B_h = C_h \otimes R_h
\]
is a matrix of fixed constants based on column scores \( C_h \) and row scores \( R_h \). When the null hypothesis is true, the CMH statistic has an asymptotic chi-square distribution with degrees of freedom equal to the rank of \( B_h \). If \( V_G \) is found to be singular, PROC FREQTAB prints a message and sets the value of the CMH statistic to missing.

PROC FREQTAB computes three CMH statistics by using this formula for the generalized CMH statistic, with different row and column score definitions for each statistic. The CMH statistics that PROC FREQTAB computes are the correlation statistic, the ANOVA (row mean scores) statistic, and the general association statistic. These statistics test the null hypothesis of no association against different alternative hypotheses. The following sections describe the computation of these CMH statistics.

**CAUTION:** The CMH statistics have low power for detecting an association in which the patterns of association for some of the strata are in the opposite direction of the patterns displayed by other strata. Thus, a nonsignificant CMH statistic suggests either that there is no association or that no pattern of association has enough strength or consistency to dominate any other pattern.

**Correlation Statistic**
The correlation statistic, popularized by Mantel and Haenszel, has 1 degree of freedom and is known as the Mantel-Haenszel statistic (Mantel and Haenszel 1959; Mantel 1963).

The alternative hypothesis for the correlation statistic is that there is a linear association between \( X \) and \( Y \) in at least one stratum. If either \( X \) or \( Y \) does not lie on an ordinal (or interval) scale, this statistic is not meaningful.

To compute the correlation statistic, PROC FREQTAB uses the formula for the generalized CMH statistic with the row and column scores determined by the SCORES= option in the TABLES statement. See the section “Scores” on page 201 for more information about the available score types. The matrix of row scores \( R_h \) has dimension \( 1 \times R \), and the matrix of column scores \( C_h \) has dimension \( 1 \times C \).

When there is only one stratum, this CMH statistic reduces to \((n - 1)r^2\), where \( r \) is the Pearson correlation coefficient between \( X \) and \( Y \). When nonparametric (RANK or RIDIT) scores are specified, the statistic reduces to \((n - 1)r_s^2\), where \( r_s \) is the Spearman rank correlation coefficient between \( X \) and \( Y \). When there is more than one stratum, this CMH statistic becomes a stratum-adjusted correlation statistic.

**ANOVA (Row Mean Scores) Statistic**
The ANOVA statistic can be used only when the column variable \( Y \) lies on an ordinal (or interval) scale so that the mean score of \( Y \) is meaningful. For the ANOVA statistic, the mean score is computed for each row of the table, and the alternative hypothesis is that, for at least one stratum, the mean scores of the \( R \) rows are unequal. In other words, the statistic is sensitive to location differences among the \( R \) distributions of \( Y \).

The matrix of column scores \( C_h \) has dimension \( 1 \times C \), and the column scores are determined by the SCORES= option.

The matrix of row scores \( R_h \) has dimension \((R - 1) \times R \) and is created internally by PROC FREQTAB as

\[
R_h = [I_{R-1}, -J_{R-1}]
\]

where \( I_{R-1} \) is an identity matrix of rank \( R - 1 \) and \( J_{R-1} \) is an \((R - 1) \times 1\) vector of ones. This matrix has the effect of forming \( R - 1 \) independent contrasts of the \( R \) mean scores.

When there is only one stratum, this CMH statistic is essentially an analysis of variance (ANOVA) statistic in the sense that it is a function of the variance ratio \( F \) statistic that would be obtained from a one-way ANOVA on the dependent variable \( Y \). If nonparametric scores are specified in this case, the ANOVA statistic is a Kruskal-Wallis test.
When there is more than one stratum, this CMH statistic corresponds to a stratum-adjusted ANOVA or
Kruskal-Wallis test. In the special case where there is one subject per row and one subject per column in the
contingency table of each stratum, this CMH statistic is identical to Friedman’s chi-square.

**General Association Statistic**
The alternative hypothesis for the general association statistic is that, for at least one stratum, there is some
kind of association between \(X\) and \(Y\). This statistic is always interpretable because it does not require an
ordinal scale for either \(X\) or \(Y\).

For the general association statistic, the matrix \(R_h\) is the same as the one used for the ANOVA statistic. The
matrix \(C_h\) is defined similarly as

\[
C_h = [I_{C-1}, -J_{C-1}]
\]

PROC FREQTAB generates both score matrices internally. When there is only one stratum, the general
association CMH statistic reduces to \(Q_P (n - 1)/n\), where \(Q_P\) is the Pearson chi-square statistic. When
there is more than one stratum, the CMH statistic becomes a stratum-adjusted Pearson chi-square statistic.
Note that a similar adjustment can be made by summing the Pearson chi-squares across the strata. However,
the latter statistic requires a large sample size in each stratum to support the resulting chi-square distribution
with \(q(R-1)(C-1)\) degrees of freedom. The CMH statistic requires only a large overall sample size because it
has only \((R-1)(C-1)\) degrees of freedom.

See Cochran (1954); Mantel and Haenszel (1959); Mantel (1963); Birch (1965); Landis, Heyman, and Koch
(1978).

**Mantel-Fleiss Criterion**
If you specify the CMH(MANTELFLEISS) option in the TABLES statement, PROC FREQTAB computes
the Mantel-Fleiss criterion for stratified \(2 \times 2\) tables. The Mantel-Fleiss criterion can be used to assess the
validity of the chi-square approximation for the distribution of the Mantel-Haenszel statistic for \(2 \times 2\) tables.
For more information, see Mantel and Fleiss (1980); Mantel and Haenszel (1959); Stokes, Davis, and Koch
(2012); Dmitrienko et al. (2005).

The Mantel-Fleiss criterion is computed as

\[
MF = \min \left( \left[ \sum_h m_{h11} - \sum_h (n_{h11})_L \right], \left[ \sum_h (n_{h11})_U - \sum_h m_{h11} \right] \right)
\]

where \(m_{h11}\) is the expected value of \(n_{h11}\) under the hypothesis of no association between the row and column
variables in table \(h\), \((n_{h11})_L\) is the minimum possible value of the table cell frequency, and \((n_{h11})_U\) is the
maximum possible value,

\[
m_{h11} = n_{h1} \cdot n_{h1} / n_h
\]

\[
(n_{h11})_L = \max (0, n_{h1} - n_{h2})
\]

\[
(n_{h11})_U = \min (n_{h1}, n_{h2})
\]

The Mantel-Fleiss guideline accepts the validity of the Mantel-Haenszel approximation when the value of the
criterion is at least 5. When the criterion is less than 5, PROC FREQTAB displays a warning.
Adjusted Odds Ratio and Relative Risk Estimates

The CMH option provides adjusted odds ratio and relative risk estimates for stratified $2 \times 2$ tables. For each of these measures, PROC FREQTAB computes a Mantel-Haenszel estimate and a logit estimate. These estimates apply to $n$-way table requests in the TABLES statement, when the row and column variables both have two levels.

For example, for the table request $A*B*C*D$, if the row and column variables $C$ and $D$ both have two levels, PROC FREQTAB provides odds ratio and relative risk estimates, adjusting for the confounding variables $A$ and $B$.

The choice of an appropriate measure depends on the study design. For case-control (retrospective) studies, the odds ratio is appropriate. For cohort (prospective) or cross-sectional studies, the relative risk is appropriate. See the section “Odds Ratio and Relative Risks for $2 \times 2$ Tables” on page 241 for more information on these measures.

Throughout this section, $z$ denotes the $100(1 - \alpha/2)$th percentile of the standard normal distribution.

Odds Ratio, Case-Control Studies

PROC FREQTAB provides Mantel-Haenszel and logit estimates for the common odds ratio for stratified $2 \times 2$ tables.

Mantel-Haenszel Estimator

The Mantel-Haenszel estimate of the common odds ratio is computed as

$$\text{OR}_{\text{MH}} = \left( \frac{\sum_h n_{h11} n_{h22} / n_h}{\sum_h n_{h12} n_{h21} / n_h} \right) / \left( \frac{\sum_h n_{h12} n_{h21} / n_h}{\sum_h n_{h11} n_{h22} / n_h} \right)$$

It is always computed unless the denominator is 0. For more information, see Mantel and Haenszel (1959) and Agresti (2002).

To compute confidence limits for the common odds ratio, PROC FREQTAB uses the Robins, Breslow, and Greenland (1986) variance estimate for $\log(\text{OR}_{\text{MH}})$. The $100(1 - \alpha/2)$% confidence limits for the common odds ratio are

$$\left( \text{OR}_{\text{MH}} \times \exp(-z \hat{\sigma}), \text{OR}_{\text{MH}} \times \exp(z \hat{\sigma}) \right)$$

where

$$\hat{\sigma}^2 = \sqrt{\text{var}(\log(\text{OR}_{\text{MH}}))}$$

$$= \frac{\sum_h (n_{h11} + n_{h22})(n_{h11} n_{h22})/n_h^2}{2 (\sum_h n_{h11} n_{h22} / n_h)^2}$$

$$+ \frac{\sum_h [(n_{h11} + n_{h22})(n_{h12} n_{h21}) + (n_{h12} + n_{h21})(n_{h11} n_{h22})]/n_h^2}{2 (\sum_h n_{h11} n_{h22} / n_h) (\sum_h n_{h12} n_{h21} / n_h)}$$

$$+ \frac{\sum_h (n_{h12} + n_{h21})(n_{h12} n_{h21})/n_h^2}{2 (\sum_h n_{h12} n_{h21} / n_h)^2}$$

Note that the Mantel-Haenszel odds ratio estimator is less sensitive to small $n_h$ than the logit estimator.
Logit Estimator

The adjusted logit estimate of the common odds ratio (Woolf 1955) is computed as

\[
\text{OR}_L = \exp \left( \sum_h w_h \log(\text{OR}_h) / \sum_h w_h \right)
\]

and the corresponding 100(1 - \(\alpha\))% confidence limits are

\[
\left( \text{OR}_L \times \exp \left( -z / \sqrt{\sum_h w_h} \right), \ \text{OR}_L \times \exp \left( z / \sqrt{\sum_h w_h} \right) \right)
\]

where \(\text{OR}_h\) is the odds ratio for stratum \(h\), and

\[w_h = 1/\text{Var}(\log(\text{OR}_h))\]

If any table cell frequency in a stratum \(h\) is 0, PROC FREQTAB adds 0.5 to each cell of the stratum before computing \(\text{OR}_h\) and \(w_h\) (Haldane 1955) for the logit estimate. The procedure provides a warning when this occurs.

Relative Risks, Cohort Studies

PROC FREQTAB provides Mantel-Haenszel and logit estimates of the common relative risks for stratified 2 \(\times\) 2 tables.

Mantel-Haenszel Estimator

The Mantel-Haenszel estimate of the common relative risk for column 1 is computed as

\[
\text{RR}_{MH} = \left( \sum_h n_{h11} n_{h22} / n_h \right) / \left( \sum_h n_{h21} n_{h1} / n_h \right)
\]

It is always computed unless the denominator is 0. For more information, see Mantel and Haenszel (1959) and Agresti (2002).

To compute confidence limits for the common relative risk, PROC FREQTAB uses the Greenland and Robins (1985) variance estimate for \(\log(\text{RR}_{MH})\). The 100(1 - \(\alpha\)/2)% confidence limits for the common relative risk are

\[
\left( \text{RR}_{MH} \times \exp(-z\hat{\sigma}), \ \text{RR}_{MH} \times \exp(z\hat{\sigma}) \right)
\]

where

\[
\hat{\sigma}^2 = \text{Var}(\log(\text{RR}_{MH})) = \frac{\sum_h (n_{h11} n_{h22} n_h - n_{h11} n_{h21} n_{h1}) / n_h^2}{(\sum_h n_{h11} n_{h22} / n_h) (\sum_h n_{h21} n_{h1} / n_h)}
\]

Logit Estimator

The adjusted logit estimate of the common relative risk for column 1 is computed as

\[
\text{RR}_L = \exp \left( \sum_h w_h \log(\text{RR}_h) / \sum_h w_h \right)
\]
and the corresponding $100(1 - \alpha)$% confidence limits are

$$\left( \text{RR}_L \times \exp \left( -z \sqrt{\sum_{h} w_h} \right), \text{RR}_L \times \exp \left( z \sqrt{\sum_{h} w_h} \right) \right)$$

where $\text{RR}_h$ is the column 1 relative risk estimate for stratum $h$ and

$$w_h = \frac{1}{\text{Var}(\log(\text{RR}_h))}$$

If $n_{h11} \,$ or $n_{h21} \,$ is 0, PROC FREQTAB adds 0.5 to each cell of the stratum before computing $\text{RR}_h$ and $w_h$ for the logit estimate. The procedure prints a warning when this occurs. For more information, see Kleinbaum, Kupper, and Morgenstern (1982, Sections 17.4 and 17.5).

**Breslow-Day Test for Homogeneity of the Odds Ratios**

When you specify the CMH option, PROC FREQTAB computes the Breslow-Day test for stratified $2 \times 2$ tables. It tests the null hypothesis that the odds ratios for the $q$ strata are equal. When the null hypothesis is true, the statistic has approximately a chi-square distribution with $q-1$ degrees of freedom. See Breslow and Day (1980) and Agresti (2007) for more information.

The Breslow-Day statistic is computed as

$$Q_{BD} = \sum_{h} \left( n_{h11} - E(n_{h11} \mid \text{OR}_{\text{MH}}) \right)^2 / \text{Var}(n_{h11} \mid \text{OR}_{\text{MH}})$$

where E and Var denote expected value and variance, respectively. The summation does not include any table that contains a row or column that has a total frequency of 0. If OR$_{\text{MH}}$ is 0 or undefined, PROC FREQTAB does not compute the statistic and prints a warning message.

For the Breslow-Day test to be valid, the sample size should be relatively large in each stratum, and at least 80% of the expected cell counts should be greater than 5. Note that this is a stricter sample size requirement than the requirement for the Cochran-Mantel-Haenszel test for $q \times 2 \times 2$ tables, in that each stratum sample size (not just the overall sample size) must be relatively large. Even when the Breslow-Day test is valid, it might not be very powerful against certain alternatives, as discussed in Breslow and Day (1980).

If you specify the BDT option, PROC FREQTAB computes the Breslow-Day test with Tarone’s adjustment, which subtracts an adjustment factor from $Q_{BD}$ to make the resulting statistic asymptotically chi-square. The Breslow-Day-Tarone statistic is computed as

$$Q_{BDT} = Q_{BD} - \left( \sum_{h} \left( n_{h11} - E(n_{h11} \mid \text{OR}_{\text{MH}}) \right) \right)^2 / \sum_{h} \text{Var}(n_{h11} \mid \text{OR}_{\text{MH}})$$

For more information, see Tarone (1985); Jones et al. (1989); Breslow (1996).

**Q Test for Homogeneity of Odds Ratios**

PROC FREQTAB computes a Q test for homogeneity of odds ratios as

$$Q = \sum_{h} w_h (\theta_h - \bar{\theta})^2$$

where $\theta_h$ is the log odds ratio in stratum $h$ and $\bar{\theta}$ is the logit estimate of the common log odds ratio. The stratum weights $w_h$ are

$$w_h = \frac{1}{\text{Var}(\theta_h)}$$
where

\[ \text{Var}(\theta_h) = \frac{1}{n_{h11}} + \frac{1}{n_{h12}} + \frac{1}{n_{h21}} + \frac{1}{n_{h22}} \]

If any table cell frequency in a stratum is 0, PROC FREQTAB adds 0.5 to each cell frequency in the stratum before computing \( \theta_h \) and \( w_h \). For more information, see the sections “Odds Ratio” on page 241 and “Adjusted Odds Ratio and Relative Risk Estimates” on page 265.

Under the null hypothesis of homogeneity, the Q statistic has approximately a chi-square distribution with \( k-1 \) degrees of freedom, where \( k \) is the number of strata.

**I-Square Measure of Heterogeneity**

The I-square statistic (Higgins and Thompson 2002) is a measure of heterogeneity among strata for stratified 2 \( \times \) 2 tables. I-square is expressed in percentage form and can be interpreted as the proportion of total variability that is due to between-strata variability. For more information, see Higgins et al. (2003) and Thorlund et al. (2012).

PROC FREQTAB computes I-square for the Q test for odds ratios as

\[ I^2 = \max \left( 100\% \times \frac{(Q - (k - 1))}{Q}, 0 \right) \]

where \( k \) is the number of strata and \( Q \) is described in the section “Q Test for Homogeneity of Odds Ratios” on page 267.

PROC FREQTAB computes uncertainty limits for I-square by using the test-based method of Higgins and Thompson (2002). This method constructs confidence limits for \( H \), where \( H^2 = Q / (k - 1) \). When \( Q > k \) or \( k = 2 \), the standard error of \( \log(H) \) is computed as

\[ \text{se}_1(\log(H)) = \left( \log(Q) - \log(k - 1) \right) / 2 \left( \sqrt{2Q} - \sqrt{2k - 3} \right) \]

When \( Q \leq k \) and \( k > 2 \), the standard error of \( \log(H) \) is computed as

\[ \text{se}_0(\log(H)) = \sqrt{\left( 1 - \frac{1}{3(k - 2)} \right) / (2(k - 2))} \]

The 100(1 - \( \alpha \))\% confidence limits for \( H \) are

\[ \left( H \times \exp(-z_{\alpha/2} \times \text{se}(\log(H))), \ H \times \exp(z_{\alpha/2} \times \text{se}(\log(H))) \right) \]

The uncertainty limits for \( I^2 \) are computed by transforming the confidence limits for \( H \), where \( I^2 = 1 - (1/H^2) \).

When \( I^2 \) is 0, PROC FREQTAB sets the lower confidence limit to 0 and determines the upper limit by using the level \( \alpha \) (instead of \( \alpha/2 \)).

**Zelen’s Exact Test for Equal Odds Ratios**

If you specify the EQOR option in the EXACT statement, PROC FREQTAB computes Zelen’s exact test for equal odds ratios for stratified 2 \( \times \) 2 tables. Zelen’s test is an exact counterpart to the Breslow-Day asymptotic test for equal odds ratios. The reference set for Zelen’s test includes all possible \( q \times 2 \times 2 \) tables with the same row, column, and stratum totals as the observed multiway table and with the same sum of cell (1,1) frequencies as the observed table. The test statistic is the probability of the observed \( q \times 2 \times 2 \) table conditional on the fixed margins, which is a product of hypergeometric probabilities.
The \( p \)-value for Zelen’s test is the sum of all table probabilities that are less than or equal to the observed table probability, where the sum is computed over all tables in the reference set determined by the fixed margins and the observed sum of cell (1,1) frequencies. This test is similar to Fisher’s exact test for two-way tables. For more information, see Zelen (1971); Hirji (2006); Agresti (1992). PROC FREQTAB computes Zelen’s exact test by using the polynomial multiplication algorithm of Hirji et al. (1996).

**Exact Confidence Limits for the Common Odds Ratio**

If you specify the COMOR option in the EXACT statement, PROC FREQTAB computes exact confidence limits for the common odds ratio for stratified \( 2 \times 2 \) tables. This computation assumes that the odds ratio is constant over all the \( 2 \times 2 \) tables. Exact confidence limits are constructed from the distribution of \( S = \sum_h n_{h11} \), conditional on the marginal totals of the \( 2 \times 2 \) tables.

Because this is a discrete problem, the confidence coefficient for these exact confidence limits is not exactly \( (1 - \alpha) \) but is at least \( (1 - \alpha) \). Thus, these confidence limits are conservative. See Agresti (1992) for more information.


Conditional on the marginal totals of \( 2 \times 2 \) table \( h \), let the random variable \( S_h \) denote the frequency of table cell (1,1). Given the row totals \( n_{h1.} \) and \( n_{h2.} \) and column totals \( n_{h.1} \) and \( n_{h.2} \), the lower and upper bounds for \( S_h \) are \( l_h \) and \( u_h \),

\[
\begin{align*}
l_h &= \max (0, \ n_{h1.} - n_{h2.}) \\
u_h &= \min (n_{h1.}, \ n_{h2.})
\end{align*}
\]

Let \( C_{s_h} \) denote the hypergeometric coefficient,

\[
C_{s_h} = \binom{n_{h1.}}{s_h} \binom{n_{h2.}}{n_{h1.} - s_h}
\]

and let \( \phi \) denote the common odds ratio. Then the conditional distribution of \( S_h \) is

\[
P( S_h = s_h | n_{1.}, n_{-1}, n_{-2} ) = C_{s_h} \phi^{s_h} / \sum_{x = l_h}^{x = u_h} C_x \phi^x
\]

Summing over all the \( 2 \times 2 \) tables, \( S = \sum_h S_h \), and the lower and upper bounds of \( S \) are \( l \) and \( u \),

\[
l = \sum_h l_h \quad \text{and} \quad u = \sum_h u_h
\]

The conditional distribution of the sum \( S \) is

\[
P( S = s | n_{h1.}, n_{h2.}; \ h = 1, \ldots, q ) = C_s \phi^s / \sum_{x = l}^{x = u} C_x \phi^x
\]

where

\[
C_s = \sum_{s_1 + \cdots + s_q = s} \left( \prod_h C_{s_h} \right)
\]
Let $s_0$ denote the observed sum of cell $(1,1)$ frequencies over the $q$ tables. The following two equations are solved iteratively for lower and upper confidence limits for the common odds ratio, $\phi_1$ and $\phi_2$:

\[
\sum_{x=s_0}^{u} C_x \phi_1^x / \sum_{x=l}^{u} C_x \phi_1^x = \alpha/2
\]

\[
\sum_{x=s_0}^{l} C_x \phi_2^x / \sum_{x=l}^{u} C_x \phi_2^x = \alpha/2
\]

When the observed sum $s_0$ equals the lower bound $l$, PROC FREQTAB sets the lower confidence limit to 0 and determines the upper limit with level $\alpha$. Similarly, when the observed sum $s_0$ equals the upper bound $u$, PROC FREQTAB sets the upper confidence limit to infinity and determines the lower limit with level $\alpha$.

When you specify the COMOR option in the EXACT statement, PROC FREQTAB also computes the exact test that the common odds ratio equals one. Setting $D_1$, the conditional distribution of the sum $S$ under the null hypothesis becomes

\[
P_0(S = s | n_{h1}, n_{h1}, n_{h2}; h = 1, \ldots, q) = C_s / \sum_{x=l}^{u} C_x
\]

The point probability for this exact test is the probability of the observed sum $s_0$ under the null hypothesis, conditional on the marginals of the stratified $2 \times 2$ tables, and is denoted by $P_0(s_0)$. The expected value of $S$ under the null hypothesis is

\[
E_0(S) = \sum_{x=l}^{u} x C_x / \sum_{x=l}^{u} C_x
\]

The one-sided exact $p$-value is computed from the conditional distribution as $P_0(S \geq s_0)$ or $P_0(S \leq s_0)$, depending on whether the observed sum $s_0$ is greater or less than $E_0(S)$,

\[
P_1 = P_0(S \geq s_0) = \sum_{x=s_0}^{u} C_x / \sum_{x=l}^{u} C_x \text{ if } s_0 > E_0(S)
\]

\[
P_1 = P_0(S \leq s_0) = \sum_{x=l}^{s_0} C_x / \sum_{x=l}^{u} C_x \text{ if } s_0 \leq E_0(S)
\]

PROC FREQTAB computes two-sided $p$-values for this test according to three different definitions. A two-sided $p$-value is computed as twice the one-sided $p$-value, setting the result equal to one if it exceeds one,

\[
P_2^a = 2 \times P_1
\]

In addition, a two-sided $p$-value is computed as the sum of all probabilities less than or equal to the point probability of the observed sum $s_0$, summing over all possible values of $s$, $l \leq s \leq u$,

\[
P_2^b = \sum_{l \leq s \leq u: P_0(s) \leq P_0(s_0)} P_0(s)
\]

Also, a two-sided $p$-value is computed as the sum of the one-sided $p$-value and the corresponding area in the opposite tail of the distribution, equidistant from the expected value,

\[
P_2^c = P_0(|S - E_0(S)| \geq |s_0 - E_0(S)|)
\]
Gail-Simon Test for Qualitative Interactions

The GAILSIMON option in the TABLES statement provides the Gail-Simon test for qualitative interaction for stratified $2 \times 2$ tables. For more information, see Gail and Simon (1985); Silvapulle (2001); Dmitrienko et al. (2005).

The Gail-Simon test is based on the risk differences in stratified $2 \times 2$ tables, where the risk difference is defined as the row 1 risk (proportion in column 1) minus the row 2 risk. For more information, see the section “Risks and Risk Differences” on page 226. By default, PROC FREQTAB uses column 1 risks to compute the Gail-Simon test. If you specify the GAILSIMON(COLUMN=2) option, PROC FREQTAB uses column 2 risks.

PROC FREQTAB computes the Gail-Simon test statistics as described in Gail and Simon (1985),

$$Q^- = \sum_h (d_h/s_h)^2 I(d_h > 0)$$

$$Q^+ = \sum_h (d_h/s_h)^2 I(d_h < 0)$$

$$Q = \min (Q^-, Q^+)$$

where $d_h$ is the risk difference in table $h$, $s_h$ is the standard error of the risk difference, and $I(d_h > 0)$ equals 1 if $d_h > 0$ and 0 otherwise. Similarly, $I(d_h < 0)$ equals 1 if $d_h < 0$ and 0 otherwise. The $q$ $2 \times 2$ tables (strata) are indexed by $h = 1, 2, \ldots, q$.

The $p$-values for the Gail-Simon statistics are computed as

$$P(Q^-) = \sum_h (1 - F_h(Q^-)) B(h; n = q, p = 0.5)$$

$$P(Q^+) = \sum_h (1 - F_h(Q^+)) B(h; n = q, p = 0.5)$$

$$P(Q) = \sum_{h=1}^{q-1} (1 - F_h(Q)) B(h; n = (q - 1), p = 0.5)$$

where $F_h(\cdot)$ is the cumulative chi-square distribution function with $h$ degrees of freedom and $B(h; n, p)$ is the binomial probability function with parameters $n$ and $p$. The statistic $Q$ tests the null hypothesis of no qualitative interaction. The statistic $Q^-$ tests the null hypothesis of positive risk differences. A small $p$-value for $Q^-$ indicates negative differences; similarly, a small $p$-value for $Q^+$ indicates positive risk differences.

Exact Statistics

Exact statistics can be useful in situations where the asymptotic assumptions are not met, and so the asymptotic $p$-values are not close approximations for the true $p$-values. Standard asymptotic methods involve the assumption that the test statistic follows a particular distribution when the sample size is sufficiently large. When the sample size is not large, asymptotic results might not be valid, with the asymptotic $p$-values
differing perhaps substantially from the exact \( p \)-values. Asymptotic results might also be unreliable when the distribution of the data is sparse, skewed, or heavily tied. See Agresti (2007) and Bishop, Fienberg, and Holland (1975) for more information. Exact computations are based on the statistical theory of exact conditional inference for contingency tables, reviewed by Agresti (1992).

In addition to computation of exact \( p \)-values, PROC FREQTAB provides the option of estimating exact \( p \)-values by Monte Carlo simulation. This can be useful for problems that are so large that exact computations require a great amount of time and memory, but for which asymptotic approximations might not be sufficient.

Exact statistics are available for many PROC FREQTAB tests. For one-way tables, PROC FREQTAB provides exact \( p \)-values for the binomial proportion tests and the chi-square goodness-of-fit test. Exact (Clopper-Pearson) confidence limits are available for the binomial proportion. For two-way tables, PROC FREQTAB provides exact \( p \)-values for the following tests: Pearson chi-square test, likelihood ratio chi-square test, Mantel-Haenszel chi-square test, Fisher’s exact test, Jonckheere-Terpstra test, Cochran-Armitage test for trend, and the symmetry test. PROC FREQTAB also computes exact \( p \)-values for tests of the following statistics: Kendall’s tau-\( b \), Stuart’s tau-\( c \), Somers’ \( D(C \mid R) \), Somers’ \( D(R \mid C) \), Pearson correlation coefficient, Spearman correlation coefficient, simple kappa coefficient, and weighted kappa coefficient. For \( 2 \times 2 \) tables, PROC FREQTAB provides McNemar’s exact test and exact confidence limits for the odds ratio. PROC FREQTAB also provides exact unconditional confidence limits for the proportion (risk) difference and for the relative risk. For stratified \( 2 \times 2 \) tables, PROC FREQTAB provides Zelen’s exact test for equal odds ratios, exact confidence limits for the common odds ratio, and an exact test for the common odds ratio.

The following sections summarize the exact computational algorithms, define the exact \( p \)-values that PROC FREQTAB computes, discuss the computational resource requirements, and describe the Monte Carlo estimation option.

**Computational Algorithms**

PROC FREQTAB computes exact \( p \)-values for general \( R \times C \) tables by using the network algorithm developed by Mehta and Patel (1983). This algorithm provides a substantial advantage over direct enumeration, which can be very time-consuming and feasible only for small problems. See Agresti (1992) for a review of algorithms for computation of exact \( p \)-values, and see Mehta, Patel, and Tsai (1984) and Mehta, Patel, and Senchaudhuri (1991) for information about the performance of the network algorithm.

The reference set for a given contingency table is the set of all contingency tables with the observed marginal row and column sums. Corresponding to this reference set, the network algorithm forms a directed acyclic network consisting of nodes in a number of stages. A path through the network corresponds to a distinct table in the reference set. The distances between nodes are defined so that the total distance of a path through the network is the corresponding value of the test statistic. At each node, the algorithm computes the shortest and longest path distances for all the paths that pass through that node. For statistics that can be expressed as a linear combination of cell frequencies multiplied by increasing row and column scores, PROC FREQTAB computes shortest and longest path distances by using the algorithm of Agresti, Mehta, and Patel (1990). For statistics of other forms, PROC FREQTAB computes an upper bound for the longest path and a lower bound for the shortest path by following the approach of Valz and Thompson (1994).

The longest and shortest path distances or bounds for a node are compared to the value of the test statistic to determine whether all paths through the node contribute to the \( p \)-value, none of the paths through the node contribute to the \( p \)-value, or neither of these situations occurs. If all paths through the node contribute, the \( p \)-value is incremented accordingly, and these paths are eliminated from further analysis. If no paths contribute, these paths are eliminated from the analysis. Otherwise, the algorithm continues, still processing this node and the associated paths. The algorithm finishes when all nodes have been accounted for.
In applying the network algorithm, PROC FREQTAB uses full numerical precision to represent all statistics, row and column scores, and other quantities involved in the computations. Although it is possible to use rounding to improve the speed and memory requirements of the algorithm, PROC FREQTAB does not do this because it can result in reduced accuracy of the \( p \)-values.

For one-way tables, PROC FREQTAB computes the exact chi-square goodness-of-fit test by the method of Radlow and Alf (1975). PROC FREQTAB generates all possible one-way tables with the observed total sample size and number of categories. For each possible table, PROC FREQTAB compares its chi-square value with the value for the observed table. If the table’s chi-square value is greater than or equal to the observed chi-square, PROC FREQTAB increments the exact \( p \)-value by the probability of that table, which is calculated under the null hypothesis by using the multinomial frequency distribution. By default, the null hypothesis states that all categories have equal proportions. If you specify null hypothesis proportions or frequencies by using the TESTP= or TESTF= option in the TABLES statement, PROC FREQTAB calculates the exact chi-square test based on that null hypothesis.

Other exact computations are described in sections about the individual statistics. For information about the computation of exact confidence limits and tests for the binomial proportion, see the section “Binomial Proportion” on page 217. For information about computation of exact confidence limits for the odds ratio, see the subsection “Exact Confidence Limits” in the section “Confidence Limits for the Odds Ratio” on page 242. For information about other exact computations, see the subsection “Exact Unconditional Confidence Limits” in the section “Confidence Limits for the Relative Risk” on page 245, and the sections “Exact Symmetry Test” on page 255, “Exact Confidence Limits for the Common Odds Ratio” on page 269 and “Zelen’s Exact Test for Equal Odds Ratios” on page 268.

**Definition of \( p \)-Values**

For several tests in PROC FREQTAB, the test statistic is nonnegative, and large values of the test statistic indicate a departure from the null hypothesis. Such nondirectional tests include the Pearson chi-square, the likelihood ratio chi-square, the Mantel-Haenszel chi-square, Fisher’s exact test for tables larger than \( 2 \times 2 \), McNemar’s test, the symmetry test, and the one-way chi-square goodness-of-fit test. The exact \( p \)-value for a nondirectional test is the sum of probabilities for those tables having a test statistic greater than or equal to the value of the observed test statistic.

There are other tests where it might be appropriate to test against either a one-sided or a two-sided alternative hypothesis. For example, when you test the null hypothesis that the true parameter value equals 0 (\( T = 0 \)), the alternative of interest might be one-sided (\( T \leq 0 \), or \( T \geq 0 \)) or two-sided (\( T \neq 0 \)). Such tests include the Pearson correlation coefficient, Spearman correlation coefficient, Jonckheere-Terpstra test, Cochran-Armitage test for trend, simple kappa coefficient, and weighted kappa coefficient. For these tests, PROC FREQTAB displays the right-sided \( p \)-value when the observed value of the test statistic is greater than its expected value. The right-sided \( p \)-value is the sum of probabilities for those tables for which the test statistic is greater than or equal to the observed test statistic. Otherwise, when the observed test statistic is less than or equal to the expected value, PROC FREQTAB displays the left-sided \( p \)-value. The left-sided \( p \)-value is the sum of probabilities for those tables for which the test statistic is less than or equal to the one observed. The one-sided \( p \)-value \( P_1 \) can be expressed as

\[
P_1 = \begin{cases} 
\text{Prob}( \text{Test Statistic} \geq t ) & \text{if } t > E_0(T) \\
\text{Prob}( \text{Test Statistic} \leq t ) & \text{if } t \leq E_0(T) 
\end{cases}
\]

where \( t \) is the observed value of the test statistic and \( E_0(T) \) is the expected value of the test statistic under the null hypothesis. PROC FREQTAB computes the two-sided \( p \)-value as the sum of the one-sided \( p \)-value and
the corresponding area in the opposite tail of the distribution of the statistic, equidistant from the expected value. The two-sided \( p \)-value \( P_2 \) can be expressed as

\[
P_2 = \text{Prob}( |\text{Test Statistic} - E_0(T)| \geq |t - E_0(T)|)
\]

If you specify the POINT option in the EXACT statement, PROC FREQTAB provides exact point probabilities for the exact tests. The exact point probability is the exact probability that the test statistic equals the observed value.

If you specify the MIDP option in the EXACT statement, PROC FREQTAB provides exact mid-\( p \)-values. The exact mid \( p \)-value is defined as the exact \( p \)-value minus half the exact point probability, which equals the average of \( \text{Prob}(\text{Test Statistic} \geq t) \) and \( \text{Prob}(\text{Test Statistic} > t) \) for a right-sided test. The exact mid \( p \)-value is smaller and less conservative than the non-adjusted exact \( p \)-value. For more information, see Agresti (2013, section 1.1.4) and Hirji (2006, sections 2.5 and 2.11.1).

**Computational Resources**

PROC FREQTAB uses relatively fast and efficient algorithms for exact computations. These recently developed algorithms, together with improvements in computer power, now make it feasible to perform exact computations for data sets where previously only asymptotic methods could be applied. Nevertheless, there are still large problems that might require a prohibitive amount of time and memory for exact computations, depending on the speed and memory available on your computer. For large problems, consider whether exact methods are really needed or whether asymptotic methods might give results quite close to the exact results and require much less computer time and memory. When asymptotic methods might not be sufficient for such large problems, consider using Monte Carlo estimation of exact \( p \)-values, as described in the section “Monte Carlo Estimation” on page 275.

A formula does not exist that can predict in advance how much time and memory are needed to compute an exact \( p \)-value for a certain problem. The time and memory required depend on several factors, including which test is being performed, the total sample size, the number of rows and columns, and the specific arrangement of the observations into table cells. Generally, larger problems (in terms of total sample size, number of rows, and number of columns) tend to require more time and memory. For a fixed total sample size, time and memory requirements tend to increase as the number of rows and number of columns increase because the number of tables in the reference set increases. Also for a fixed sample size, time and memory requirements tend to increase as the marginal row and column totals become more homogeneous. For more information, see Agresti, Mehta, and Patel (1990) and Gail and Mantel (1977).

When PROC FREQTAB is computing exact \( p \)-values, you can terminate the computations by pressing the system interrupt key sequence (see the SAS Companion for your system) and choosing to stop computations. After you terminate exact computations, PROC FREQTAB completes all other remaining tasks. The procedure produces the requested output and reports missing values for any exact \( p \)-values that were not computed by the time of termination.

You can also use the MAXTIME= option in the EXACT statement to limit the amount of time PROC FREQTAB uses for exact computations. You specify a MAXTIME= value that is the maximum amount of clock time (in seconds) that PROC FREQTAB can use to compute an exact \( p \)-value. If PROC FREQTAB does not finish computing an exact \( p \)-value within that time, it terminates the computation and completes all other remaining tasks.
**Monte Carlo Estimation**

If you specify the option MC in the EXACT statement, PROC FREQTAB computes Monte Carlo estimates of the exact \( p \)-values instead of directly computing the exact \( p \)-values. Monte Carlo estimation can be useful for large problems that require a great amount of time and memory for exact computations but for which asymptotic approximations might not be sufficient. To describe the precision of each Monte Carlo estimate, PROC FREQTAB provides the asymptotic standard error and 100(1 – \( \alpha \))\% confidence limits. The ALPHA= option in the EXACT statement determines the confidence level \( \alpha \); by default, ALPHA=0.01, which produces 99% confidence limits. The N=n option in the EXACT statement specifies the number of samples that PROC FREQTAB uses for Monte Carlo estimation; the default is 10000 samples. You can specify a larger value for \( n \) to improve the precision of the Monte Carlo estimates. Because larger values of \( n \) generate more samples, the computation time increases. Alternatively, you can specify a smaller value of \( n \) to reduce the computation time.

To compute a Monte Carlo estimate of an exact \( p \)-value, PROC FREQTAB generates a random sample of tables with the same total sample size, row totals, and column totals as the observed table. PROC FREQTAB uses the algorithm of Agresti, Wackerly, and Boyett (1979), which generates tables in proportion to their hypergeometric probabilities conditional on the marginal frequencies. For each sample table, PROC FREQTAB computes the value of the test statistic and compares it to the value for the observed table. When estimating a right-sided \( p \)-value, PROC FREQTAB counts all sample tables for which the test statistic is greater than or equal to the observed test statistic. Then the \( p \)-value estimate equals the number of these tables divided by the total number of tables sampled.

\[
\hat{P}_{MC} = \frac{M}{N}
\]

- \( M \) = number of samples with (Test Statistic \( \geq t \))
- \( N \) = total number of samples
- \( t \) = observed Test Statistic

PROC FREQTAB computes left-sided and two-sided \( p \)-value estimates in a similar manner. For left-sided \( p \)-values, PROC FREQTAB evaluates whether the test statistic for each sampled table is less than or equal to the observed test statistic. For two-sided \( p \)-values, PROC FREQTAB examines the sample test statistics according to the expression for \( P_2 \) given in the section “Definition of \( p \)-Values” on page 273.

The variable \( M \) is a binomially distributed variable with \( N \) trials and success probability \( p \). It follows that the asymptotic standard error of the Monte Carlo estimate is

\[
se(\hat{P}_{MC}) = \sqrt{\frac{\hat{P}_{MC} (1 - \hat{P}_{MC})}{(N - 1)}}
\]

PROC FREQTAB constructs asymptotic confidence limits for the \( p \)-values according to

\[
\hat{P}_{MC} \pm (z_{\alpha/2} \times se(\hat{P}_{MC}))
\]

where \( z_{\alpha/2} \) is the 100(1 – \( \alpha/2 \))\% percentile of the standard normal distribution and the confidence level \( \alpha \) is determined by the ALPHA= option in the EXACT statement.

When the Monte Carlo estimate \( \hat{P}_{MC} \) is 0, PROC FREQTAB computes the confidence limits for the \( p \)-value as

\[
(0, 1 - \alpha^{(1/N)})
\]

When the Monte Carlo estimate \( \hat{P}_{MC} \) is 1, PROC FREQTAB computes the confidence limits as

\[
(\alpha^{(1/N)}, 1)
\]
Output Data Sets

PROC FREQTAB produces two types of output data sets that you can use with other statistical and reporting procedures. You can request these data sets as follows:

- Specify the OUT= option in a TABLES statement. This creates an output data set that contains frequency or crosstabulation table counts and percentages
- Specify an OUTPUT statement. This creates an output data set that contains statistics.

PROC FREQTAB does not display the output data sets. Use PROC PRINT, PROC REPORT, or any other SAS reporting tool to display an output data set.

In addition to these two output data sets, you can create a SAS data set from any piece of PROC FREQTAB output by using the Output Delivery System. See the section “ODS Table Names” on page 288 for more information.

Contents of the TABLES Statement Output Data Set

The OUT= option in the TABLES statement creates an output data set that contains one observation for each combination of variable values (or table cell) in the last table request. By default, each observation contains the frequency and percentage for the table cell. The output data set includes the following variables:

- BY variables
- table request variables, such as A, B, C, and D in the table request A*B*C*D
- COUNT, which contains the table cell frequency
- PERCENT, which contains the table cell percentage

If you specify the OUTEXPECT option in the TABLES statement for a two-way or multiway table, the output data set also includes expected cell frequencies. If you specify the OUTPCT option for a two-way or multiway table, the output data set also includes row, column, and table percentages. The additional variables are as follows:

- EXPECTED, which contains the expected frequency
- PCT_TABL, which contains the percentage of two-way table frequency, for n-way tables where n > 2
- PCT_ROW, which contains the percentage of row frequency
- PCT_COL, which contains the percentage of column frequency

The OUTEXPECT and OUTPCT options have no effect for one-way tables or for tables that you display in list format (which you can request by specifying the LIST option).

If you specify the OUTCUM option in the TABLES statement for a one-way table or for a table in list format, the output data set also includes cumulative frequencies and cumulative percentages. The additional variables are as follows:
• CUM_FREQ, which contains the cumulative frequency
• CUM_PCT, which contains the cumulative percentage

The following PROC FREQTAB statements create an output data set of frequencies and percentages:

```plaintext
proc freqtab;
  tables A A*B / out=D;
run;
```

The output data set D contains frequencies and percentages for the table of A by B, which is the last table request listed in the TABLES statement. If A has two levels (1 and 2), B has three levels (1,2, and 3), and no table cell count is 0 or missing, the output data set D includes six observations, one for each combination of A and B levels. The first observation corresponds to A=1 and B=1; the second observation corresponds to A=1 and B=2; and so on. The data set includes the variables COUNT and PERCENT. The value of COUNT is the number of observations with the given combination of A and B levels. The value of PERCENT is the percentage of the total number of observations with that A and B combination.

When PROC FREQTAB combines different variable values into the same formatted level, the output data set contains the smallest internal value for the formatted level. For example, suppose a variable X has the values 1.1, 1.4, 1.7, 2.1, and 2.3. When you submit the statement

```plaintext
format X 1.;
```

in a PROC FREQTAB step, the formatted levels listed in the frequency table for X are 1 and 2. If you create an output data set with the frequency counts, the internal values of the levels of X are 1.1 and 1.7. To report the internal values of X when you display the output data set, use a format of 3.1 for X.

**Contents of the OUTPUT Statement Output Data Set**

The OUTPUT statement creates a SAS data set that contains statistics computed by PROC FREQTAB. Table 5.6 lists the statistics that can be stored in the output data set. You identify which statistics to include by specifying output-options. For more information, see the description of the OUTPUT statement.

If you specify multiple TABLES statements or multiple table requests in a single TABLES statement, the contents of the output data set correspond to the last table request.

For a one-way table or a two-way table, the output data set contains one observation that stores the requested statistics for the table. For a multiway table, the output data set contains an observation for each two-way table (stratum) of the multiway crosstabulation. If you request summary statistics for the multiway table, the output data set also contains an observation that stores the across-strata summary statistics. If you use a BY statement, the output data set contains an observation (for one-way or two-way tables) or set of observations (for multiway tables) for each BY group.

The OUTPUT data set can include the following variables:

• BY variables
• Variables that identify the stratum for multiway tables, such as A and B in the table request A*B*C*D
• Variables that contain the specified statistics
In addition to the specified estimate or test statistic, the output data set includes associated values such as standard errors, confidence limits, \(p\)-values, and degrees of freedom.

PROC FREQTAB constructs variable names for the statistics in the output data set by enclosing the `output-option` names in underscores. Variable names for the corresponding standard errors, confidence limits, \(p\)-values, and degrees of freedom are formed by combining the `output-option` names with prefixes that identify the associated values. Table 5.20 lists the prefixes and their descriptions.

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>E_</td>
<td>Asymptotic standard error (ASE)</td>
</tr>
<tr>
<td>L_</td>
<td>Lower confidence limit</td>
</tr>
<tr>
<td>U_</td>
<td>Upper confidence limit</td>
</tr>
<tr>
<td>E0_</td>
<td>Null hypothesis ASE</td>
</tr>
<tr>
<td>Z_</td>
<td>Standardized value</td>
</tr>
<tr>
<td>DF_</td>
<td>Degrees of freedom</td>
</tr>
<tr>
<td>P_</td>
<td>(p)-value</td>
</tr>
<tr>
<td>P2_</td>
<td>Two-sided (p)-value</td>
</tr>
<tr>
<td>PL_</td>
<td>Left-sided (p)-value</td>
</tr>
<tr>
<td>PR_</td>
<td>Right-sided (p)-value</td>
</tr>
<tr>
<td>XP_</td>
<td>Exact (p)-value</td>
</tr>
<tr>
<td>XP2_</td>
<td>Exact two-sided (p)-value</td>
</tr>
<tr>
<td>XPL_</td>
<td>Exact left-sided (p)-value</td>
</tr>
<tr>
<td>XPR_</td>
<td>Exact right-sided (p)-value</td>
</tr>
<tr>
<td>XPT_</td>
<td>Exact point probability</td>
</tr>
<tr>
<td>XMP_</td>
<td>Exact mid (p)-value</td>
</tr>
<tr>
<td>XL_</td>
<td>Exact lower confidence limit</td>
</tr>
<tr>
<td>XU_</td>
<td>Exact upper confidence limit</td>
</tr>
</tbody>
</table>

For example, the `PCHI` `output-option` in the `OUTPUT` statement includes the Pearson chi-square test in the output data set. The variable names for the Pearson chi-square statistic, its degrees of freedom, and the corresponding \(p\)-value are `_PCHI_`, `DF_PCHI`, and `P_PCHI`, respectively. For variables that were added to the output data set before SAS/STAT 8.2, PROC FREQTAB truncates the variable name to eight characters when the length of the prefix plus the `output-option` name exceeds eight characters.

---

### Displayed Output

#### Number of Variable Levels Table

If you specify the `NLEVELS` option in the PROC FREQTAB statement, PROC FREQTAB displays the “Number of Variable Levels” table. This table provides the number of levels for all variables named in the `TABLES` statements. PROC FREQTAB determines the variable levels by using the formatted variable values. The “Number of Variable Levels” table contains the following information:

- Variable name
• Levels, which is the total number of levels of the variable

**One-Way Frequency Tables**

PROC FREQTAB displays one-way frequency tables for all one-way table requests in the TABLES statements, unless you specify the NOPRINT option in the PROC FREQTAB statement or the NOPRINT option in the TABLES statement. For a one-way table showing the frequency distribution of a single variable, PROC FREQTAB displays the name of the variable and its values. For each variable value or level, PROC FREQTAB displays the following information:

• Frequency count, which is the number of observations in the level

• Test Frequency count, if you specify the CHISQ and TESTF= options to request a chi-square goodness-of-fit test for specified frequencies

• Percent, which is the percentage of the total number of observations. (The NOPERCENT option suppresses this information.)

• Test Percent, if you specify the CHISQ and TESTP= options to request a chi-square goodness-of-fit test for specified percents. (The NOPERCENT option suppresses this information.)

• Cumulative Frequency count, which is the sum of the frequency counts for that level and all other levels listed above it in the table. The last cumulative frequency is the total number of nonmissing observations. (The NOCUM option suppresses this information.)

• Cumulative Percent, which is the percentage of the total number of observations in that level and in all other levels listed above it in the table. (The NOCUM or the NOPERCENT option suppresses this information.)

The one-way table also displays the Frequency Missing, which is total frequency of missing values.

**Statistics for One-Way Frequency Tables**

For one-way tables, two statistical options are available in the TABLES statement. The CHISQ option provides a chi-square goodness-of-fit test, and the BINOMIAL option provides binomial proportion statistics and tests. PROC FREQTAB displays the following information, unless you specify the NOPRINT option in the PROC FREQTAB statement:

• If you specify the CHISQ option for a one-way table, PROC FREQTAB provides a chi-square goodness-of-fit test, displaying the Chi-Square statistic, the degrees of freedom (DF), and the probability value (Pr > ChiSq). If you specify the CHISQ option in the EXACT statement, PROC FREQTAB also displays the exact probability value for this test. If you specify the POINT option with the CHISQ option in the EXACT statement, PROC FREQTAB displays the exact point probability for the test statistic. If you specify the MIDP option in the EXACT statement, PROC FREQTAB displays the exact mid $p$-value for the chi-square test.

• If you specify the BINOMIAL option for a one-way table, PROC FREQTAB displays the estimate of the binomial Proportion, which is the proportion of observations in the first class listed in the one-way table. PROC FREQTAB also displays the asymptotic standard error (ASE) and the asymptotic (Wald)
Chapter 5: The FREQTAB Procedure

and exact (Clopper-Pearson) confidence limits by default. For the binomial proportion test, PROC FREQTAB displays the asymptotic standard error under the null hypothesis (ASE Under H0), the standardized test statistic (Z), and the one-sided and two-sided probability values.

If you specify the BINOMIAL option in the EXACT statement, PROC FREQTAB also displays the exact one-sided and two-sided probability values for this test. If you specify the POINT option with the BINOMIAL option in the EXACT statement, PROC FREQTAB displays the exact point probability for the test. If you specify the MIDP option in the EXACT statement, PROC FREQTAB displays the exact mid p-value for the binomial proportion test.

- If you request binomial confidence limits by specifying the BINOMIAL(CL=) option, PROC FREQTAB displays the “Binomial Confidence Limits” table, which includes the Lower and Upper Confidence Limits for each confidence limit Type that you request. In addition to Wald and Clopper-Pearson (Exact) confidence limits, you can request the following confidence limit types for the binomial proportion: Agresti-Coull, Blaker, Jeffreys, Likelihood Ratio, Logit, Mid-p, and Wilson (score).

- If you request a binomial noninferiority or superiority test by specifying the NONINF or SUP binomial-option, PROC FREQTAB displays a Noninferiority Analysis or Superiority Analysis table that contains the following information: the binomial Proportion, the test ASE (under H0 or Sample), the test statistic Z, the probability value, the noninferiority or superiority limit, and the test confidence limits. If you specify the BINOMIAL option in the EXACT statement, PROC FREQTAB also provides the exact probability value for the test, and exact test confidence limits.

- If you request a binomial equivalence test by specifying the EQUIV binomial-option, PROC FREQTAB displays an Equivalence Analysis table that contains the following information: binomial Proportion and the test ASE (under H0 or Sample). PROC FREQTAB displays two one-sided tests (TOST) for equivalence, which include test statistics (Z) and probability values for the Lower and Upper tests, together with the Overall probability value. PROC FREQTAB also displays the equivalence limits and the test-based confidence limits. If you specify the BINOMIAL option in the EXACT statement, PROC FREQTAB provides exact probability values for the TOST and exact test-based confidence limits.

Two-Way and Multiway Tables

PROC FREQTAB displays all multiway table requests in the TABLES statements, unless you specify the NOPRINT option in the PROC FREQTAB statement or the NOPRINT option in the TABLES statement.

For two-way to multiway crosstabulation tables, the values of the last variable in the table request form the table columns. The values of the next-to-last variable form the rows. Each level (or combination of levels) of the other variables forms one stratum.

There are three ways to display multiway tables in PROC FREQTAB. By default, PROC FREQTAB displays multiway tables as separate two-way crosstabulation tables for each stratum of the multiway table. Also by default, PROC FREQTAB displays these two-way crosstabulation tables in table cell format. Alternatively, if you specify the CROSSLIST option, PROC FREQTAB displays the two-way crosstabulation tables in ODS column format. If you specify the LIST option, PROC FREQTAB displays multiway tables in list format, which presents the entire multiway crosstabulation in a single table.
Crosstabulation Tables
By default, PROC FREQTAB displays two-way crosstabulation tables in table cell format. The row variable values are listed down the side of the table, the column variable values are listed across the top of the table, and each row and column variable level combination forms a table cell.

Each cell of a crosstabulation table can contain the following information:

- Frequency, which is the number of observations in the table cell. (The NOFREQ option suppresses this information.)
- Expected frequency under the hypothesis of independence, if you specify the EXPECTED option
- Deviation of the cell frequency from the expected value, if you specify the DEVIATION option
- Cell Chi-Square, which is the cell’s contribution to the total chi-square statistic, if you specify the CELLCHI2 option
- Tot Pct, which is the cell’s percentage of the total multiway table frequency, for \( n \)-way tables when \( n > 2 \), if you specify the TOTPCT option
- Percent, which is the cell’s percentage of the total (two-way table) frequency. (The NOPERCENT option suppresses this information.)
- Row Pct, or the row percentage, which is the cell’s percentage of the total frequency for its row. (The NOROW option suppresses this information.)
- Col Pct, or column percentage, which is the cell’s percentage of the total frequency for its column. (The NOCOL option suppresses this information.)
- Cumulative Col%, or cumulative column percentage, if you specify the CUMCOL option

The table also displays the Frequency Missing, which is the total frequency of missing values.

CROSSLIST Tables
If you specify the CROSSLIST option, PROC FREQTAB displays two-way crosstabulation tables in ODS column format. The CROSSLIST column format is different from the default crosstabulation table cell format, but the CROSSLIST table provides the same information (frequencies, percentages, and other statistics) as the default crosstabulation table.

In the CROSSLIST table format, the rows of the display correspond to the crosstabulation table cells, and the columns of the display correspond to descriptive statistics such as frequencies and percentages. Each table cell is identified by the values of its TABLES row and column variable levels, with all column variable levels listed within each row variable level. The CROSSLIST table also provides row totals, column totals, and overall table totals.

For a crosstabulation table in CROSSLIST format, PROC FREQTAB displays the following information:

- the row variable name and values
- the column variable name and values
- Frequency, which is the number of observations in the table cell. (The NOFREQ option suppresses this information.)
- Expected cell frequency under the hypothesis of independence, if you specify the `EXPECTED` option
- Deviation of the cell frequency from the expected value, if you specify the `DEVIATION` option
- Standardized Residual, if you specify the `CROSSLIST(STDRES)` option
- Pearson Residual, if you specify the `CROSSLIST(PEARSONRES)` option
- Cell Chi-Square, which is the cell’s contribution to the total chi-square statistic, if you specify the `CELLCHI2` option
- Total Percent, which is the cell’s percentage of the total multiway table frequency, for \( n \)-way tables when \( n > 2 \), if you specify the `TOTPCT` option
- Percent, which is the cell’s percentage of the total (two-way table) frequency. (The `NOPERCENT` option suppresses this information.)
- Row Percent, which is the cell’s percentage of the total frequency for its row. (The `NOROW` option suppresses this information.)
- Column Percent, the cell’s percentage of the total frequency for its column. (The `NOCOL` option suppresses this information.)

The table also displays the Frequency Missing, which is the total frequency of missing values.

**LIST Tables**

If you specify the `LIST` option in the `TABLES` statement, PROC FREQTAB displays two-way and multiway tables by using a list format instead of the default crosstabulation cell format. The list format displays an entire multiway table in a single table instead of separate two-way (stratum) tables. Each row of a table in list format corresponds to a single crosstabulation cell. Tables that are displayed in list format do not include row percentages, column percentages, or optional two-way table information such as expected frequencies and cell chi-squares.

PROC FREQTAB displays the following information in a list table:

- the variable names and values
- Frequency, which is the number of observations in the level (with the indicated variable values)
- Percent, which is the level’s percentage of the total number of observations. (The `NOPERCENT` option suppresses this information.)
- Cumulative Frequency, which is the accumulated frequency of the level and all other levels listed above it in the table. The last cumulative frequency in the table is the total number of nonmissing observations. (The `NOCUM` option suppresses this information.)
- Cumulative Percent, which is the accumulated percentage of the level and all other levels listed above it in the table. (The `NOCUM` or the `NOPERCENT` option suppresses this information.)

The table also displays the Frequency Missing, which is the total frequency of missing values.
Statistics for Two-Way and Multiway Tables

PROC FREQTAB computes statistical tests and measures for crosstabulation tables, depending on which statements and options you specify. You can suppress the display of these results by specifying the NOPRINT option in the PROC FREQTAB statement. With any of the following information, PROC FREQTAB also displays the Sample Size and the Frequency Missing.

- If you specify the SCOROUT option in the TABLES statement, PROC FREQTAB displays the Row Scores and Column Scores that it uses for statistical computations. The Row Scores table displays the row variable values and the Score corresponding to each value. The Column Scores table displays the column variable values and the corresponding Scores. PROC FREQTAB also identifies the score type used to compute the row and column scores. You can specify the score type with the SCORES= option in the TABLES statement.

- If you specify the CHISQ option, PROC FREQTAB displays the following statistics for each two-way table: Pearson Chi-Square, Likelihood Ratio Chi-Square, Continuity-Adjusted Chi-Square (for $2 \times 2$ tables), Mantel-Haenszel Chi-Square, the Phi Coefficient, the Contingency Coefficient, and Cramér’s V. For each test statistic, PROC FREQTAB also displays the degrees of freedom (DF) and the probability value (Prob).

- If you specify the FISHER option in the TABLES statement (or, equivalently, the FISHER option in the EXACT statement), PROC FREQTAB displays Fisher’s exact test. The test output includes the cell (1,1) frequency (F), the exact left-sided and right-sided probability values, the table probability (P), and the exact two-sided probability value. If you specify the POINT option in the EXACT statement, PROC FREQTAB displays the exact point probability for Fisher’s exact test. If you specify the MIDP option in the EXACT statement, PROC FREQTAB displays the Mid $p$-Value for the test.

- If you specify the PCHI, LRCHI, or MHCHI option in the EXACT statement, PROC FREQTAB displays the corresponding exact test: Pearson Chi-Square, Likelihood Ratio Chi-Square, or Mantel-Haenszel Chi-Square, respectively. The test output includes the test statistic, the degrees of freedom (DF), and the asymptotic and exact probability values. If you also specify the POINT option in the EXACT statement, PROC FREQTAB displays the point probability for each exact test requested. If you specify the MIDP option in the EXACT statement, PROC FREQTAB displays the exact mid $p$-value for each test. If you specify the CHISQ option in the EXACT statement, PROC FREQTAB displays exact probability values for all three of these chi-square tests.

- If you specify the MC option in the EXACT statement, PROC FREQTAB displays Monte Carlo estimates for all exact $p$-values that you request in the EXACT statement. The Monte Carlo output includes the $p$-value Estimate, its Confidence Limits, the Number of Samples used to compute the Monte Carlo estimate, and the Initial Seed for random number generation.

- If you specify the MEASURES option, PROC FREQTAB displays the following statistics and their asymptotic standard errors (ASE) for each two-way table: Gamma, Kendall’s Tau-\$b\$, Stuart’s Tau-\$c\$,
Chapter 5: The FREQTAB Procedure

- If you specify the PLCORR option, PROC FREQTAB displays the polychoric correlation and its asymptotic standard error (ASE). For $2 \times 2$ tables, this statistic is known as the tetrachoric correlation (and is labeled as such in the displayed output). If you specify the CL option, PROC FREQTAB also displays confidence limits for the polychoric correlation. If you specify the PLCORR option in the TEST statement, PROC FREQTAB displays the polychoric correlation, asymptotic standard error (ASE), confidence limits, and the following: the standardized test statistic ($Z$), the corresponding one-sided and two-sided probability values, the likelihood ratio (LR) chi-square, and the probability value ($Pr > ChiSq$).

- If you specify the GAMMA, KENTB, STUTC, SMDCR, SMDRC, PCORR, or SCORR option in the TEST statement, PROC FREQTAB displays asymptotic tests for Gamma, Kendall’s Tau-$b$, Stuart’s Tau-$c$, Somers’ $D(C \mid R)$, Somers’ $D(R \mid C)$, the Pearson Correlation, or the Spearman Correlation, respectively. If you specify the MEASURES option in the TEST statement, PROC FREQTAB displays all these asymptotic tests. The test output includes the statistic, its asymptotic standard error (ASE), Confidence Limits, the ASE under the null hypothesis $H_0$, the standardized test statistic ($Z$), and the one-sided and two-sided probability values.

- If you specify the KENTB, STUTC, SMDCR, SMDRC, PCORR, or SCORR option in the EXACT statement, PROC FREQTAB displays asymptotic and exact tests for the corresponding measure of association: Kendall’s Tau-$b$, Stuart’s Tau-$c$, Somers’ $D(C \mid R)$, Somers’ $D(R \mid C)$, the Pearson Correlation, or the Spearman Correlation, respectively. The test output includes the correlation, its asymptotic standard error (ASE), Confidence Limits, the ASE under the null hypothesis $H_0$, the standardized test statistic ($Z$), and the asymptotic and exact one-sided and two-sided probability values. If you also specify the POINT option in the EXACT statement, PROC FREQTAB displays the point probability for each exact test requested. If you specify the MIDP option in the EXACT statement, PROC FREQTAB displays the exact Mid $p$-Value for each test.

- If you specify the SENSPEC option for $2 \times 2$ tables, PROC FREQTAB displays the “Sensitivity and Specificity” table. This table displays the Estimate, Standard Error, and Confidence Limits for the following statistics: Sensitivity, Specificity, Positive Predictive Value, and Negative Predictive Value.

- If you specify the RISKDIFF option for $2 \times 2$ tables, PROC FREQTAB displays the “Confidence Limits for the Proportion (Risk) Difference” table, which includes the Lower and Upper Confidence Limits for each confidence limit Type that you request (Agresti-Caffo, Exact, Hauck-Anderson, Miettinen-Nurminen, Newcombe, or Wald). If you specify the RISKDIFF option in the EXACT statement, PROC FREQTAB provides unconditional Exact Confidence Limits for the Risk Difference. You can suppress this table by specifying the RISKDIFF(NORISKS) option.

- If you specify the RISKDIFF(CL=) option for $2 \times 2$ tables, PROC FREQTAB displays the “Confidence Limits for the Proportion (Risk) Difference” table, which includes the Lower and Upper Confidence Limits for each confidence limit Type that you request (Agresti-Caffo, Exact, Hauck-Anderson, Miettinen-Nurminen, Newcombe, or Wald). If you specify the RISKDIFF(NONINF) option for $2 \times 2$ tables, PROC FREQTAB displays the “Noninferiority Analysis for the Risk Difference” table, which includes the Risk Difference, test
ASE, standardized test statistic Z, probability value (Pr > Z), Noninferiority Limit, and (test-based) Confidence Limits.

- If you specify the RISKDIFF(SUP) option for 2 × 2 tables, PROC FREQTAB displays the “Superiority Analysis for the Risk Difference” table, which includes the Risk Difference, test ASE, standardized test statistic Z, probability value (Pr > Z), Superiority Limit, and (test-based) Confidence Limits.

- If you specify the RISKDIFF(EQUIV) option for 2 × 2 tables, PROC FREQTAB displays the “Equivalence Analysis for the Risk Difference” table, which includes the Risk Difference, test ASE, Equivalence Limits, and (test-based) Confidence Limits. PROC FREQTAB also displays the “Two One-Sided Tests (TOST)” table, which includes test statistics (Z) and P-Values for the Lower Margin and Upper Margin tests, along with the Overall P-Value.

- If you specify the RISKDIFF(EQUAL) option for 2 × 2 tables, PROC FREQTAB displays the “Risk Difference Test” table, which includes the Risk Difference, test ASE, standardized test statistic Z, One-sided probability value (Pr > Z or Pr < Z), and Two-sided probability value (Pr > |Z|).

- If you specify the MEASURES option or the RELRISK option for 2 × 2 tables, PROC FREQTAB displays the “Odds Ratio and Relative Risks” table, which includes the following statistics with their confidence limits: Odds Ratio, Relative Risk (Column 1), and Relative Risk (Column 2). If you specify the OR option in the EXACT statement, PROC FREQTAB also displays the “Exact Confidence Limits for the Odds Ratio” table. If you specify the RELRISK option in the EXACT statement, PROC FREQTAB displays the “Exact Confidence Limits for the Relative Risk” table.

- If you specify the OR(CL=) option for 2 × 2 tables, PROC FREQTAB displays the “Confidence Limits for the Odds Ratio” table, which includes the Lower and Upper Confidence Limits for each confidence limit Type that you request (Exact, Mid-p, Likelihood Ratio, Score, Wald, or Wald Modified).

- If you specify the RELRISK(CL=) option for 2 × 2 tables, PROC FREQTAB displays the “Confidence Limits for the Relative Risk” table, which includes the Lower and Upper Confidence Limits for each confidence limit Type that you request (Exact, Likelihood Ratio, Score, Wald, or Wald Modified).

- If you specify the RELRISK(NONINF) option, PROC FREQTAB displays the “Noninferiority Analysis for the Relative Risk” table, which includes the Relative Risk, standardized test statistic Z, probability value (Pr > Z), Noninferiority Limit, and Confidence Limits.

- If you specify the RELRISK(SUP) option, PROC FREQTAB displays the “Superiority Analysis for the Relative Risk” table, which includes the Relative Risk, standardized test statistic Z, probability value (Pr > Z), Superiority Limit, and Confidence Limits.

- If you specify the RELRISK(EQUIV) option, PROC FREQTAB displays the “Equivalence Analysis for the Relative Risk” table, which includes the Relative Risk, Equivalence Limits, and Confidence Limits. PROC FREQTAB also displays the “Two One-Sided Tests (TOST)” table, which includes test statistics (Z) and P-Values for the Lower Margin and Upper Margin tests, along with the Overall P-Value.

- If you specify the RELRISK(EQUAL) option, PROC FREQTAB displays the “Relative Risk Test” table, which includes the Relative Risk, standardized test statistic Z, One-sided probability value (Pr > Z or Pr < Z), and Two-sided probability value (Pr > |Z|).
• If you specify the TREND option, PROC FREQTAB displays the Cochran-Armitage Trend Test for tables that are $2 \times C$ or $R \times 2$. For this test, PROC FREQTAB gives the Statistic ($Z$) and the one-sided and two-sided probability values. If you specify the TREND option in the EXACT statement, PROC FREQTAB also displays the exact one-sided and two-sided probability values for this test. If you specify the POINT option with the TREND option in the EXACT statement, PROC FREQTAB displays the exact point probability for the test statistic. If you specify the MIDP option in the EXACT statement, PROC FREQTAB displays the exact Mid p-Value for the trend test.

• If you specify the JT option, PROC FREQTAB displays the Jonckheere-Terpstra Test, showing the Statistic (JT), the standardized test statistic (Z), and the one-sided and two-sided probability values. If you specify the JT option in the EXACT statement, PROC FREQTAB also displays the exact one-sided and two-sided probability values for this test. If you specify the POINT option with the JT option in the EXACT statement, PROC FREQTAB displays the exact point probability for the test statistic. If you specify the MIDP option in the EXACT statement, PROC FREQTAB displays the exact Mid p-Value for the Jonckheere-Terpstra test.

• If you specify the AGREE option for a $2 \times 2$ table, PROC FREQTAB displays the “McNemar’s Test” table. This table includes the McNemar test statistic (chi-square), the degrees of freedom, and the $p$-value. If you specify the MCNEM option in the EXACT statement, this table also includes the exact $p$-value. If you specify the POINT option or the MIDP option in the EXACT statement, the “McNemar’s Test” table includes the exact point probability or the exact mid $p$-value, respectively.

• If you specify the AGREE option for a square table of dimension greater than 2, PROC FREQTAB displays the “Symmetry Test” table. This table displays Bowker’s symmetry test statistic (chi-square), the degrees of freedom, and the $p$-value. If you specify the SYMMETRY option in the EXACT statement, this table also includes the exact $p$-value. If you specify the POINT option or the MIDP option in the EXACT statement, the “Symmetry Test” table includes the exact point probability or the exact mid $p$-value, respectively.

• The AGREE option also produces the “Kappa Statistics” table, which displays the simple kappa coefficient. If the dimension of the two-way table is greater than 2, the “Kappa Statistics” table includes the weighted kappa coefficient. If you specify the AGREE(AC1) option or the AGREE(PABAK) option, this table includes the AC1 agreement coefficient or the prevalence-adjusted bias-adjusted kappa (PABAK), respectively. The “Kappa Statistics” table displays the standard error and confidence limits for each agreement statistic.

• If you specify the AGREE(KAPPADETAILS) option, PROC FREQTAB displays the “Kappa Details” table, which includes the observed agreement, the chance-expected agreement, the maximum kappa, and the B_N measure. For $2 \times 2$ tables, the “Kappa Details” table also includes the prevalence index and the bias index.

• If you specify the AGREE(WTKAPPADETAILS) or AGREE(KAPPADETAILS) option for a square table of dimension greater than 2, PROC FREQTAB produces the “Weighted Kappa Details” table, which displays the observed agreement and the chance-expected agreement components of the weighted kappa coefficient.

• If you specify the AGREE(PRINTKWTS) option for a square table of dimension greater than 2, PROC FREQTAB displays the matrix of agreement weights in the “Kappa Coefficient Weights” table.

• If you request a simple kappa coefficient test, PROC FREQTAB produces the “Kappa Test” table. You can request this test by specifying the KAPPA option in the TEST statement, the KAPPA option in the
EXACT statement, or the AGREE(NULLKAPPA=) option in the TABLES statement. The “Kappa Test” table displays the kappa coefficient, null test value, standard error (when the null value is 0), standardized test statistic (Z), and one-sided and two-sided p-values.

If you request an exact test (by specifying the KAPPA option in the EXACT statement), the “Kappa Test” table also includes the exact one-sided and two-sided p-values. If you specify the POINT option or the MIDP option in the EXACT statement, the “Kappa Test” table includes the point probability or the exact mid p-value, respectively.

- If you request a weighted kappa coefficient test for a square table of dimension greater than 2, PROC FREQTAB produces the “Weighted Kappa Test” table. You can request this test by specifying the WTKAPPA option in the TEST statement, the WTKAPPA option in the EXACT statement, or the AGREE(NULLWTKAPPA=) option in the TABLES statement. The “Weighted Kappa Test” table displays the weighted kappa coefficient, null test value, standard error (when the null value is 0), standardized test statistic (Z), and one-sided and two-sided p-values.

If you request an exact test (by specifying the WTKAPPA option in the EXACT statement), the “Weighted Kappa Test” table also includes the exact one-sided and two-sided p-values. If you specify the POINT option or the MIDP option in the EXACT statement, the “Weighted Kappa Test” table includes the point probability or the exact mid p-value, respectively.

- If you specify the AGREE option for a multiway square table, PROC FREQTAB displays the “Overall Kappa Coefficients” table, which includes the overall simple kappa coefficient together with its standard error and confidence limits. This table also includes the overall weighted kappa coefficient if the two-way table dimension is greater than 2.

- For multiway square tables, the AGREE option also produces the “Tests for Equal Kappa Coefficients” table. This table includes the chi-square statistic, degrees of freedom, and p-value for the test of equal simple kappa coefficients (over all strata). If the two-way table dimension is greater than 2, this table also includes the test for equal weighted kappa coefficients.

- For multiway $2 \times 2$ tables, the AGREE option displays the “Cochran’s Q” table, which includes Cochran’s Q statistic (to test for marginal homogeneity), the degrees of freedom, and the p-value.

- If you specify the COMMONRISKDIFF option for a multiway $2 \times 2$ table, PROC FREQTAB displays the “Confidence Limits for the Common Risk Difference” table, which includes the Method, Value of the common risk difference, Standard Error, and Confidence Limits for each confidence limit type that you request (Klingenberg, Mantel-Haenszel, Minimum Risk, Newcombe, Newcombe MR, or Summary Score).

- If you specify the COMMONRISKDIFF(TEST) option for a multiway $2 \times 2$ table, PROC FREQTAB displays the “Common Risk Difference Tests” table, which includes Method, Risk Difference, Z, and Pr > |Z| for each test that you request (Mantel-Haenszel, Minimum Risk, or Summary Score).

- If you specify the COMMONRISKDIFF(PRINTWTS) option for a multiway $2 \times 2$ table, PROC FREQTAB displays the “Stratum Weights” table, which includes the following information for each stratum ($2 \times 2$ table): Stratum index, variable levels, Risk Difference, Frequency, Fraction, and the stratum weights that you request (Mantel-Haenszel, Minimum Risk, or Summary Score Weights).

- If you specify the CMH option, PROC FREQTAB displays Cochran-Mantel-Haenszel Statistics for the following three alternative hypotheses: Nonzero Correlation, Row Mean Scores Differ (ANOVA Statistic), and General Association. For each of these statistics, PROC FREQTAB gives the degrees of
freedom (DF) and the probability value (Prob). If you specify the **MANTELFLEISS** option, PROC FREQTAB displays the Mantel-Fleiss Criterion for $2 \times 2$ tables. For $2 \times 2$ tables, PROC FREQTAB also displays Estimates of the Common Relative Risk for Case-Control and Cohort studies, together with their confidence limits. These include both Mantel-Haenszel and Logit stratum-adjusted estimates of the common Odds Ratio, Column 1 Relative Risk, and Column 2 Relative Risk. Also for $2 \times 2$ tables, PROC FREQTAB displays the Breslow-Day Test for Homogeneity of the Odds Ratios. For this test, PROC FREQTAB gives the Chi-Square, the degrees of freedom (DF), and the probability value (Pr > ChiSq).

- If you specify the **CMH(QOR)** option for a stratified $2 \times 2$ table, PROC FREQTAB displays the “Q Test for Homogeneity of Odds Ratios” table, which includes the Chi-Square, the degrees of freedom (DF), and the probability value (Pr > ChiSq).

- If you specify the **CMH(I2)** option for a stratified $2 \times 2$ table, PROC FREQTAB displays the “I-Square Measure of Heterogeneity” table, which includes the I-Square, the degrees of freedom (DF), and the Confidence Limits.

- If you specify the **CMH** option in the TABLES statement and also specify the COMOR option in the EXACT statement for a multiway $2 \times 2$ table, PROC FREQTAB displays exact confidence limits for the Common Odds Ratio. PROC FREQTAB also displays the Exact Test of H$_0$: Common Odds Ratio = 1. The test output includes the Cell (1,1) Sum (S), Mean of S Under H$_0$, One-sided Pr $<=$ S, and Point Pr $=$ S. PROC FREQTAB also provides exact two-sided probability values for the test, computed according to the following three methods: 2 * One-sided, Sum of probabilities $<=$ Point probability, and Pr $>= |S - Mean|$. If you specify the MIDP option in the EXACT statement, PROC FREQTAB provides the exact Mid $p$-Value for the common odds ratio test.

- If you specify the **CMH** option in the TABLES statement and also specify the EQOR option in the EXACT statement for a multiway $2 \times 2$ table, PROC FREQTAB computes Zelen’s exact test for equal odds ratios. PROC FREQTAB displays Zelen’s exact test statistic, Zelen’s Exact Test (P), and the probability value, Exact Pr $<=$ P.

- If you specify the **GAILSIMON** option in the TABLES statement for a multiway $2 \times 2$ tables, PROC FREQTAB displays the Gail-Simon test for qualitative interactions. The display include the following statistics and their $p$-values: Q+ (Positive Risk Differences), Q- (Negative Risk Differences), and Q (Two-Sided).

**ODS Table Names**

PROC FREQTAB assigns a name to each table that it creates. You can use these names to refer to tables when you use the Output Delivery System (ODS) to select tables and create output data sets. For more information about ODS, see the *SAS Output Delivery System: User’s Guide*.

Table 5.21 lists the ODS table names together with their descriptions and the options that are required to produce the tables.
<table>
<thead>
<tr>
<th>ODS Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>BarnardsTest</td>
<td>Barnard’s exact test</td>
<td>EXACT</td>
<td>BARNARD</td>
</tr>
<tr>
<td>Binomial</td>
<td>Binomial proportion</td>
<td>TABLES</td>
<td>BINOMIAL</td>
</tr>
<tr>
<td>BinomialCLs</td>
<td>Binomial confidence limits</td>
<td>TABLES</td>
<td>BINOMIAL(CL=)</td>
</tr>
<tr>
<td>BinomialEquiv</td>
<td>Binomial equivalence analysis</td>
<td>TABLES</td>
<td>BINOMIAL(EQUIV)</td>
</tr>
<tr>
<td>BinomialEquivLimits</td>
<td>Binomial equivalence limits</td>
<td>TABLES</td>
<td>BINOMIAL(EQUIV)</td>
</tr>
<tr>
<td>BinomialEquivTest</td>
<td>Binomial equivalence test</td>
<td>TABLES</td>
<td>BINOMIAL(EQUIV)</td>
</tr>
<tr>
<td>BinomialNoninf</td>
<td>Binomial noninferiority test</td>
<td>TABLES</td>
<td>BINOMIAL(NONINF)</td>
</tr>
<tr>
<td>BinomialSup</td>
<td>Binomial superiority test</td>
<td>TABLES</td>
<td>BINOMIAL(SUP)</td>
</tr>
<tr>
<td>BinomialTest</td>
<td>Binomial proportion test</td>
<td>TABLES</td>
<td>BINOMIAL</td>
</tr>
<tr>
<td>BnMeasure</td>
<td>Agreement measures</td>
<td>TABLES</td>
<td>PLOTS=AGREEPLOT(STATS)</td>
</tr>
<tr>
<td>BreslowDayTest</td>
<td>Breslow-Day test</td>
<td>TABLES</td>
<td>CMH ((h \times 2 \times 2) table)</td>
</tr>
<tr>
<td>ChiSq</td>
<td>Chi-square tests</td>
<td>TABLES</td>
<td>CHISQ</td>
</tr>
<tr>
<td>CMH</td>
<td>Cochran-Mantel-Haenszel statistics</td>
<td>TABLES</td>
<td>CMH</td>
</tr>
<tr>
<td>CochransQ</td>
<td>Cochran’s (Q)</td>
<td>TABLES</td>
<td>AGREE ((h \times 2 \times 2) table)</td>
</tr>
<tr>
<td>ColScores</td>
<td>Column scores</td>
<td>TABLES</td>
<td>SCOROUT</td>
</tr>
<tr>
<td>CommonOdds-</td>
<td>Exact confidence limits</td>
<td>EXACT</td>
<td>COMOR</td>
</tr>
<tr>
<td>RatioCl</td>
<td>for the common odds ratio</td>
<td></td>
<td>((h \times 2 \times 2) table)</td>
</tr>
<tr>
<td>CommonOdds-</td>
<td>Common odds ratio exact test</td>
<td>EXACT</td>
<td>COMOR</td>
</tr>
<tr>
<td>RatioTest</td>
<td>((h \times 2 \times 2) table)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CommonPdiff</td>
<td>Common risk difference</td>
<td>TABLES</td>
<td>COMMONRISKDIFF</td>
</tr>
<tr>
<td>CommonPdiffTests</td>
<td>confidence limits</td>
<td></td>
<td>((h \times 2 \times 2) table)</td>
</tr>
<tr>
<td>CommonRelRisks</td>
<td>Common relative risks</td>
<td>TABLES</td>
<td>CMH ((h \times 2 \times 2) table)</td>
</tr>
<tr>
<td>CrossList</td>
<td>Crosstabulation table</td>
<td>TABLES</td>
<td>CROSSLIST</td>
</tr>
<tr>
<td>CrossTabFreqs</td>
<td>in column format</td>
<td></td>
<td>((n\text{-way table, } n &gt; 1))</td>
</tr>
<tr>
<td>EqualKappaTest</td>
<td>Test for equal simple kappas</td>
<td>TABLES</td>
<td>AGREE</td>
</tr>
<tr>
<td>EqualKappaTests</td>
<td>Tests for equal kappas</td>
<td>TABLES</td>
<td>AGREE</td>
</tr>
<tr>
<td>EqualOddsRatios</td>
<td>Tests for equal odds ratios</td>
<td>EXACT</td>
<td>EQOR</td>
</tr>
<tr>
<td>GailSimonTest</td>
<td>Gail-Simon test</td>
<td>TABLES</td>
<td>GAILSIMON</td>
</tr>
<tr>
<td>FishersExact</td>
<td>Fisher’s exact test</td>
<td>EXACT</td>
<td>FISHER</td>
</tr>
<tr>
<td>FishersExactMC</td>
<td>Monte Carlo estimates for Fisher’s exact test</td>
<td>EXACT</td>
<td>FISHER / MC</td>
</tr>
<tr>
<td>Gamma</td>
<td>Gamma</td>
<td>TEST</td>
<td>GAMMA</td>
</tr>
<tr>
<td>GammaTest</td>
<td>Gamma test</td>
<td>TEST</td>
<td>GAMMA</td>
</tr>
</tbody>
</table>
### Table 5.21 continued

<table>
<thead>
<tr>
<th>ODS Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>I2</td>
<td>I-square measure</td>
<td>TABLES</td>
<td>CMH(I2) (h x 2 x 2 table)</td>
</tr>
<tr>
<td>JTTTest</td>
<td>Jonckheere-Terpstra test</td>
<td>TABLES</td>
<td>JT</td>
</tr>
<tr>
<td>JTTTestMC</td>
<td>Monte Carlo estimates for Jonckheere-Terpstra exact test</td>
<td>EXACT</td>
<td>JT / MC</td>
</tr>
<tr>
<td>KappaDetails</td>
<td>Kappa details</td>
<td>TABLES</td>
<td>AGREE(KPAPADETAILS)</td>
</tr>
<tr>
<td>KappaMC</td>
<td>Monte Carlo exact test for simple kappa coefficient</td>
<td>EXACT</td>
<td>KAPPA / MC</td>
</tr>
<tr>
<td>KappaStatistics</td>
<td>Kappa statistics</td>
<td>TABLES</td>
<td>AGREE</td>
</tr>
<tr>
<td>KappaTest</td>
<td>Simple kappa test</td>
<td>TEST</td>
<td>KAPPA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>or EXACT</td>
<td>KAPPA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>or TABLES</td>
<td>AGREE(NULLKAPPA=)</td>
</tr>
<tr>
<td>KappaWeights</td>
<td>Kappa weights</td>
<td>TABLES</td>
<td>AGREE(PRINTWKTS)</td>
</tr>
<tr>
<td>List</td>
<td>List format multiway table</td>
<td>TABLES</td>
<td>LIST</td>
</tr>
<tr>
<td>LRChiSq</td>
<td>Likelihood ratio chi-square exact test</td>
<td>EXACT</td>
<td>LRCHI</td>
</tr>
<tr>
<td>LRChiSqMC</td>
<td>Monte Carlo exact test for likelihood ratio chi-square</td>
<td>EXACT</td>
<td>LRCHI / MC</td>
</tr>
<tr>
<td>MantelFleiss</td>
<td>Mantel-Fleiss criterion</td>
<td>TABLES</td>
<td>CMH(MANTELFEISS) (h x 2 x 2 table)</td>
</tr>
<tr>
<td>McNemarsTest</td>
<td>McNemar’s test</td>
<td>TABLES</td>
<td>AGREE (2 x 2 table)</td>
</tr>
<tr>
<td>Measures</td>
<td>Measures of association</td>
<td>TABLES</td>
<td>MEASURES</td>
</tr>
<tr>
<td>MHChiSq</td>
<td>Mantel-Haenszel chi-square exact test</td>
<td>EXACT</td>
<td>MHCHI</td>
</tr>
<tr>
<td>MHChiSqMC</td>
<td>Monte Carlo exact test for Mantel-Haenszel chi-square</td>
<td>EXACT</td>
<td>MHCHI / MC</td>
</tr>
<tr>
<td>NLevels</td>
<td>Number of variable levels</td>
<td>PROC</td>
<td>NLEVELS</td>
</tr>
<tr>
<td>OddsRatioCLs</td>
<td>Odds ratio confidence limits</td>
<td>TABLES</td>
<td>OR(CL=) (2 x 2 table)</td>
</tr>
<tr>
<td>OddsRatioExactCL</td>
<td>Exact confidence limits for the odds ratio</td>
<td>EXACT</td>
<td>OR (2 x 2 table)</td>
</tr>
<tr>
<td>OneWayChiSq</td>
<td>One-way chi-square test</td>
<td>TABLES</td>
<td>CHISQ (one-way table)</td>
</tr>
<tr>
<td>OneWayChiSqMC</td>
<td>Monte Carlo exact test for one-way chi-square</td>
<td>EXACT</td>
<td>CHISQ / MC (one-way table)</td>
</tr>
<tr>
<td>OneWayFreqs</td>
<td>One-way frequencies</td>
<td>PROC</td>
<td>(no TABLES stmt)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>or TABLES</td>
<td>(one-way table)</td>
</tr>
<tr>
<td>OneWayLRChiSq</td>
<td>One-way likelihood ratio chi-square test</td>
<td>TABLES</td>
<td>CHISQ(LRCHI) (one-way table)</td>
</tr>
<tr>
<td>OverallKappa</td>
<td>Overall simple kappa</td>
<td>TABLES</td>
<td>AGREE (h x 2 x 2 table)</td>
</tr>
<tr>
<td>OverallKappas</td>
<td>Overall kappa coefficients</td>
<td>TABLES</td>
<td>AGREE (h x r x r table, r &gt; 2)</td>
</tr>
<tr>
<td>PdiffCLs</td>
<td>Risk difference confidence limits</td>
<td>TABLES</td>
<td>RISKDIFF(CL=) (2 x 2 table)</td>
</tr>
<tr>
<td>ODS Table Name</td>
<td>Description</td>
<td>Statement</td>
<td>Option</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------------------------</td>
<td>-------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>PdiffEquiv</td>
<td>Equivalence analysis for the risk difference</td>
<td>TABLES</td>
<td>RISKDIFF(EQUIV)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(2 × 2 table)</td>
</tr>
<tr>
<td>PdiffEquivTest</td>
<td>Equivalence test for the risk difference</td>
<td>TABLES</td>
<td>RISKDIFF(EQUIV)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(2 × 2 table)</td>
</tr>
<tr>
<td>PdiffNoninf</td>
<td>Noninferiority test for the risk difference</td>
<td>TABLES</td>
<td>RISKDIFF(NONINF)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(2 × 2 table)</td>
</tr>
<tr>
<td>PdiffSup</td>
<td>Superiority test for the risk difference</td>
<td>TABLES</td>
<td>RISKDIFF(SUP)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(2 × 2 table)</td>
</tr>
<tr>
<td>PdiffTest</td>
<td>Risk difference test</td>
<td>TABLES</td>
<td>RISKDIFF(EQUAL)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(2 × 2 table)</td>
</tr>
<tr>
<td>PearsonChiSq</td>
<td>Pearson chi-square exact test</td>
<td>EXACT</td>
<td>PCHI</td>
</tr>
<tr>
<td>PearsonChiSqMC</td>
<td>Monte Carlo exact test for Pearson chi-square</td>
<td>EXACT</td>
<td>PCHI / MC</td>
</tr>
<tr>
<td>PearsonCorr</td>
<td>Pearson correlation</td>
<td>TEST</td>
<td>PCORR</td>
</tr>
<tr>
<td>PearsonCorrMC</td>
<td>Monte Carlo exact test for Pearson correlation</td>
<td>EXACT</td>
<td>PCORR / MC</td>
</tr>
<tr>
<td>PearsonCorrTest</td>
<td>Pearson correlation test</td>
<td>TEST or EXACT</td>
<td>PCORR</td>
</tr>
<tr>
<td>PLCorr</td>
<td>Polychoric correlation</td>
<td>TEST</td>
<td>PLCORR</td>
</tr>
<tr>
<td>PLCorrTest</td>
<td>Polychoric correlation test</td>
<td>TEST</td>
<td>PLCORR</td>
</tr>
<tr>
<td>QOR</td>
<td>Q test for odds ratios</td>
<td>TABLES</td>
<td>CMH(QOR) or CMH(I2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(h × 2 × 2 table)</td>
</tr>
<tr>
<td>RelativeRiskCLs</td>
<td>Relative risk confidence limits</td>
<td>TABLES</td>
<td>RELRISK(CL=)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(2 × 2 table)</td>
</tr>
<tr>
<td>RelativeRisks</td>
<td>Relative risk estimates</td>
<td>TABLES</td>
<td>RELRISK or MEASURES</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(2 × 2 table)</td>
</tr>
<tr>
<td>RelRisk1ExactCL</td>
<td>Exact confidence limits for column 1 relative risk</td>
<td>EXACT</td>
<td>RELRISK</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(2 × 2 table)</td>
</tr>
<tr>
<td>RelRisk2ExactCL</td>
<td>Exact confidence limits for column 2 relative risk</td>
<td>EXACT</td>
<td>RELRISK</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(2 × 2 table)</td>
</tr>
<tr>
<td>RelriskEquiv</td>
<td>Equivalence analysis for the relative risk</td>
<td>TABLES</td>
<td>RELRISK(EQUIV)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(2 × 2 table)</td>
</tr>
<tr>
<td>RelriskEquivTest</td>
<td>Equivalence test for the relative risk</td>
<td>TABLES</td>
<td>RELRISK(EQUIV)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(2 × 2 table)</td>
</tr>
<tr>
<td>RelriskNoninf</td>
<td>Noninferiority test for the relative risk</td>
<td>TABLES</td>
<td>RELRISK(NONINF)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(2 × 2 table)</td>
</tr>
<tr>
<td>RelriskSup</td>
<td>Superiority test for the relative risk</td>
<td>TABLES</td>
<td>RELRISK(SUP)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(2 × 2 table)</td>
</tr>
<tr>
<td>RelriskTest</td>
<td>Relative risk test</td>
<td>TABLES</td>
<td>RELRISK(EQUAL)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(2 × 2 table)</td>
</tr>
<tr>
<td>RiskDiffCol1</td>
<td>Column 1 risk estimates</td>
<td>TABLES</td>
<td>RISKDIFF</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(2 × 2 table)</td>
</tr>
<tr>
<td>RiskDiffCol2</td>
<td>Column 2 risk estimates</td>
<td>TABLES</td>
<td>RISKDIFF</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(2 × 2 table)</td>
</tr>
<tr>
<td>RowScores</td>
<td>Row scores</td>
<td>TABLES</td>
<td>SCOROUT</td>
</tr>
<tr>
<td>ODS Table Name</td>
<td>Description</td>
<td>Statement</td>
<td>Option</td>
</tr>
<tr>
<td>------------------</td>
<td>--------------------------------------------</td>
<td>-----------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>SenSpec</td>
<td>Sensitivity and specificity</td>
<td>TABLES</td>
<td>SENSPEC (2 x 2 table)</td>
</tr>
<tr>
<td>SomersDCR</td>
<td>Somers’ $D(C</td>
<td>R)$</td>
<td>TEST</td>
</tr>
<tr>
<td></td>
<td>or EXACT SMDCR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SomersDCRMC</td>
<td>Monte Carlo exact test for Somers’ $D(C</td>
<td>R)$</td>
<td>EXACT</td>
</tr>
<tr>
<td>SomersDCRTest</td>
<td>Somers’ $D(C</td>
<td>R)$ test</td>
<td>TEST</td>
</tr>
<tr>
<td></td>
<td>or EXACT SMDCR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SomersDRC</td>
<td>Somers’ $D(R</td>
<td>C)$</td>
<td>TEST</td>
</tr>
<tr>
<td></td>
<td>or EXACT SMDRC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SomersDRCMC</td>
<td>Monte Carlo exact test for Somers’ $D(R</td>
<td>C)$</td>
<td>EXACT</td>
</tr>
<tr>
<td>SomersDRCTest</td>
<td>Somers’ $D(R</td>
<td>C)$ test</td>
<td>TEST</td>
</tr>
<tr>
<td></td>
<td>or EXACT SMDRC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SpearmanCorr</td>
<td>Spearman correlation</td>
<td>TEST</td>
<td>SCORR</td>
</tr>
<tr>
<td></td>
<td>or EXACT SCORR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SpearmanCorrMC</td>
<td>Monte Carlo exact test for Spearman correlation</td>
<td>EXACT</td>
<td>SCORR / MC</td>
</tr>
<tr>
<td>SpearmanCorrTest</td>
<td>Spearman correlation test</td>
<td>TEST</td>
<td>SCORR</td>
</tr>
<tr>
<td></td>
<td>or EXACT SCORR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>StratumWeights</td>
<td>Stratum weights and risk differences</td>
<td>TABLES</td>
<td>COMMONRISKDIFF (h x 2 x 2 table)</td>
</tr>
<tr>
<td>SymmetryMC</td>
<td>Monte Carlo exact symmetry test</td>
<td>EXACT</td>
<td>SYMMETRY / MC</td>
</tr>
<tr>
<td>SymmetryTest</td>
<td>Symmetry test</td>
<td>TABLES</td>
<td>AGREE</td>
</tr>
<tr>
<td></td>
<td>or EXACT AGREE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TauB</td>
<td>Kendall’s tau-b</td>
<td>TEST</td>
<td>KENTB</td>
</tr>
<tr>
<td></td>
<td>or EXACT KENTB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TauBMC</td>
<td>Monte Carlo exact test for Kendall’s tau-b</td>
<td>EXACT</td>
<td>KENTB / MC</td>
</tr>
<tr>
<td>TauBTest</td>
<td>Kendall’s tau-b test</td>
<td>TEST</td>
<td>KENTB</td>
</tr>
<tr>
<td></td>
<td>or EXACT KENTB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TauC</td>
<td>Stuart’s tau-c</td>
<td>TEST</td>
<td>STUTC</td>
</tr>
<tr>
<td></td>
<td>or EXACT STUTC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TauCMC</td>
<td>Monte Carlo exact test for Stuart’s tau-c</td>
<td>EXACT</td>
<td>STUTC / MC</td>
</tr>
<tr>
<td>TauCTest</td>
<td>Stuart’s tau-c test</td>
<td>TEST</td>
<td>STUTC</td>
</tr>
<tr>
<td></td>
<td>or EXACT STUTC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TrendTest</td>
<td>Cochran-Armitage trend test</td>
<td>TABLES</td>
<td>TREND</td>
</tr>
<tr>
<td>TrendTestMC</td>
<td>Monte Carlo exact test for trend</td>
<td>EXACT</td>
<td>TREND / MC</td>
</tr>
<tr>
<td>WtKappaDetails</td>
<td>Weighted kappa details</td>
<td>TABLES</td>
<td>AGREE(WTKAPPADetails)</td>
</tr>
<tr>
<td>WtKappaMC</td>
<td>Monte Carlo exact test for weighted kappa coefficient</td>
<td>EXACT</td>
<td>WTKAPPA / MC</td>
</tr>
</tbody>
</table>
Table 5.21  continued

<table>
<thead>
<tr>
<th>ODS Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>WtKappaTest</td>
<td>Weighted kappa test</td>
<td>TEST</td>
<td>WTKAPPA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>or EXACT</td>
<td>WTKAPPA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>or TABLES</td>
<td>AGREE(NULLWTKAPPA=)</td>
</tr>
</tbody>
</table>

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in the “Statistical Graphics Using ODS” chapter in SAS/STAT User’s Guide.

Before you create graphs, ODS Graphics must be enabled (for example, with the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section “Enabling and Disabling ODS Graphics” in that chapter.

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section “A Primer on ODS Statistical Graphics” in that chapter.

When ODS Graphics is enabled, you can request specific plots by specifying the PLOTS= option in the TABLES statement. To produce a frequency plot or cumulative frequency plot, you must specify the FREQPLOT or CUMFREQPLOT plot-request, respectively, in the PLOTS= option. To produce a mosaic plot, you must specify the MOSAICPLOT plot-request in the PLOTS= option. You can also produce frequency, cumulative frequency, and mosaic plots by specifying the PLOTS=ALL option. By default, PROC FREQTAB produces all other plots that are associated with the analyses that you request in the TABLES statement. You can suppress the default plots and request specific plots by using the PLOTS(ONLY)= option. For more information, see the description of the PLOTS= option.

PROC FREQTAB assigns a name to each graph that it creates by using ODS Graphics. You can use these names to refer to the graphs. Table 5.22 lists the names of the graphs that PROC FREQTAB generates together with their descriptions, their PLOTS= options (plot-requests), and the TABLES statement options that are required to produce the graphs.

Table 5.22  Graphs Produced by PROC FREQTAB

<table>
<thead>
<tr>
<th>ODS Graph Name</th>
<th>Description</th>
<th>PLOTS= Option</th>
<th>TABLES Statement Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>AgreePlot</td>
<td>Agreement plot</td>
<td>AGREEPLOT</td>
<td>AGREE (r x r table)</td>
</tr>
<tr>
<td>CumFreqPlot</td>
<td>Cumulative frequency plot</td>
<td>CUMFREQPLOT</td>
<td>One-way table request</td>
</tr>
<tr>
<td>DeviationPlot</td>
<td>Deviation plot</td>
<td>DEVIATIONPLOT</td>
<td>CHISQ (one-way table)</td>
</tr>
<tr>
<td>FreqPlot</td>
<td>Frequency plot</td>
<td>FREQPLOT</td>
<td>Any table request</td>
</tr>
<tr>
<td>KappaPlot</td>
<td>Kappa plot</td>
<td>KAPPAPLOT</td>
<td>AGREE (h x r x r table)</td>
</tr>
<tr>
<td>MosaicPlot</td>
<td>Mosaic plot</td>
<td>MOSAICPLOT</td>
<td>Two-way or multiway table request</td>
</tr>
<tr>
<td>ORPlot</td>
<td>Odds ratio plot</td>
<td>ODDSRATIOPLOT</td>
<td>MEASURES, OR,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>or RELRISK (h x 2 x 2 table)</td>
</tr>
<tr>
<td>RelRiskPlot</td>
<td>Relative risk plot</td>
<td>RELRISKPLOT</td>
<td>MEASURES or RELRISK</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(h x 2 x 2 table)</td>
</tr>
<tr>
<td>RiskDiffPlot</td>
<td>Risk difference plot</td>
<td>RISKDIFFPLOT</td>
<td>RISKDIFF (h x 2 x 2 table)</td>
</tr>
<tr>
<td>WtKappaPlot</td>
<td>Weighted kappa plot</td>
<td>WTKAPPAPLOT</td>
<td>AGREE (h x r x r table, r &gt; 2)</td>
</tr>
</tbody>
</table>
References


References


Chapter 5: The FREQTAB Procedure


References


# Chapter 6
The GAMMOD Procedure

## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview: GAMMOD Procedure</td>
<td>304</td>
</tr>
<tr>
<td>PROC GAMMOD Features</td>
<td>304</td>
</tr>
<tr>
<td>PROC GAMMOD Contrasted with Other SAS Procedures</td>
<td>305</td>
</tr>
<tr>
<td>Using CAS Sessions and CAS Engine Librefs</td>
<td>307</td>
</tr>
<tr>
<td>Getting Started: GAMMOD Procedure</td>
<td>307</td>
</tr>
<tr>
<td>Syntax: GAMMOD Procedure</td>
<td>314</td>
</tr>
<tr>
<td>PROC GAMMOD Statement</td>
<td>314</td>
</tr>
<tr>
<td>BY Statement</td>
<td>317</td>
</tr>
<tr>
<td>CLASS Statement</td>
<td>318</td>
</tr>
<tr>
<td>DISPLAY Statement</td>
<td>318</td>
</tr>
<tr>
<td>DISPLAYOUT Statement</td>
<td>319</td>
</tr>
<tr>
<td>FREQ Statement</td>
<td>320</td>
</tr>
<tr>
<td>MODEL Statement</td>
<td>320</td>
</tr>
<tr>
<td>OUTPUT Statement</td>
<td>329</td>
</tr>
<tr>
<td>WEIGHT Statement</td>
<td>330</td>
</tr>
<tr>
<td>Details: GAMMOD Procedure</td>
<td>330</td>
</tr>
<tr>
<td>Missing Values</td>
<td>330</td>
</tr>
<tr>
<td>Thin-Plate Regression Splines</td>
<td>331</td>
</tr>
<tr>
<td>Generalized Additive Models</td>
<td>333</td>
</tr>
<tr>
<td>Model Evaluation Criteria</td>
<td>335</td>
</tr>
<tr>
<td>Fitting Algorithms</td>
<td>336</td>
</tr>
<tr>
<td>Degrees of Freedom</td>
<td>337</td>
</tr>
<tr>
<td>Model Inference</td>
<td>338</td>
</tr>
<tr>
<td>Dispersion Parameter</td>
<td>338</td>
</tr>
<tr>
<td>Tests for Smoothing Components</td>
<td>339</td>
</tr>
<tr>
<td>Multithreading</td>
<td>340</td>
</tr>
<tr>
<td>Optimization Algorithms</td>
<td>340</td>
</tr>
<tr>
<td>Displayed Output</td>
<td>341</td>
</tr>
<tr>
<td>ODS Table Names</td>
<td>343</td>
</tr>
<tr>
<td>ODS Graphics</td>
<td>344</td>
</tr>
<tr>
<td>Examples: GAMMOD Procedure</td>
<td>345</td>
</tr>
<tr>
<td>Example 6.1: Scatter Plot Smoothing</td>
<td>345</td>
</tr>
<tr>
<td>Example 6.2: Nonparametric Logistic Regression</td>
<td>350</td>
</tr>
<tr>
<td>Example 6.3: Nonparametric Tweedie Regression</td>
<td>356</td>
</tr>
<tr>
<td>References</td>
<td>361</td>
</tr>
</tbody>
</table>
Overview: GAMMOD Procedure

The GAMMOD procedure fits generalized additive models that are based on low-rank regression splines (Wood 2006) in SAS Viya.

Generalized additive models are extensions of generalized linear models. They relax the generalized linear models’ assumption of linearity by allowing spline terms that characterize nonlinear dependency structures. Each spline term is constructed by the thin-plate regression spline technique (Wood 2003). A roughness penalty is applied to each spline term by a smoothing parameter that controls the balance between goodness of fit and the roughness of the spline curve. PROC GAMMOD fits models for standard distributions in the exponential family, such as the normal, Poisson, gamma, and Tweedie distributions.

PROC GAMMOD Features

PROC GAMMOD offers the following basic features:

- estimates the regression parameters of a generalized additive model that has fixed smoothing parameters by using penalized likelihood estimation
- estimates the smoothing parameters of a generalized additive model by using either the performance iteration method or the outer iteration method
- estimates the regression parameters of a generalized linear model by using maximum likelihood techniques
- tests the total contribution of each spline term based on the Wald statistic
- provides model-building syntax in the CLASS statement and effect-based parametric effects in the MODEL statement, which are used in other SAS/STAT analytic procedures (in particular, the GLM, LOGISTIC, GLIMMIX, and MIXED procedures)
- provides response-variable options
- enables you to construct a spline term by using multiple variables
- provides control options for constructing a spline term, such as fixed degrees of freedom, initial smoothing parameter, fixed smoothing parameter, smoothing parameter search range, user-supplied knot values, and so on
- provides multiple link functions for any distribution
- provides a WEIGHT statement for weighted analysis
- provides a FREQ statement for grouped analysis
- provides an OUTPUT statement to produce a data table that has predicted values and other observationwise statistics
- produces graphs via ODS Graphics
Because the GAMMOD procedure runs on SAS Cloud Analytic Services (CAS), it also does the following:

- enables you to run on a cluster of machines that distribute the data and the computations
- enables you to run in single-machine mode
- exploits all the available cores and concurrent threads (for information about how PROC GAMMOD uses threads, see the section “Multithreading” on page 81 in Chapter 3, “Shared Concepts”)

### PROC GAMMOD Compared with Other SAS Procedures

The GAMMOD procedure provides generalized additive modeling functionality that is comparable to that of GAMPL and GAM procedures in SAS/STAT software.

### PROC GAMMOD Compared with the GAMPL Procedure

The functionality of the GAMMOD procedure closely resembles that of the GAMPL procedure, which is a high-performance SAS/STAT procedure. The GAMMOD procedure is the next generation of the GAMPL procedure, and it was developed specifically for SAS Viya. Both procedures are designed to run on a cluster of machines that distribute the data and the computations. Both procedures perform computations in multiple threads.

PROC GAMMOD provides all the features and options that PROC GAMPL supports; in addition, it supports BY processing. The default link functions for two distributions differ between the two procedures, as shown in Table 6.1.

<table>
<thead>
<tr>
<th>Distribution</th>
<th>GAMMOD</th>
<th>GAMPL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gamma</td>
<td>Log</td>
<td>Reciprocal</td>
</tr>
<tr>
<td>Inverse Gaussian</td>
<td>Log</td>
<td>Reciprocal square</td>
</tr>
</tbody>
</table>

You should expect close results between these two procedures, although exactly equivalent models are not guaranteed because their implementations are different.

### PROC GAMMOD Compared with the GAM Procedure

Both the GAMMOD procedure and the GAM procedure in SAS/STAT software fit generalized additive models. However, the GAMMOD procedure uses different approaches for constructing spline basis expansions, fitting generalized additive models, and testing smoothing components. The GAMMOD procedure focuses on automatic smoothing parameter selection by using global model-evaluation criteria to find optimal models. The GAM procedure focuses on constructing models by fitting partial residuals against each smoothing term with fixed degrees of freedom. In general, you should not expect similar results from these two procedures. Table 6.2 summarizes the main differences.
Table 6.2  PROC GAMMOD Compared with PROC GAM

<table>
<thead>
<tr>
<th>Functionality</th>
<th>GAMMOD</th>
<th>GAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constructing spline basis</td>
<td>Uses thin-plate regression splines to construct basis expansions for</td>
<td>Uses univariate or bivariate smoothing splines to construct basis</td>
</tr>
<tr>
<td>expansions</td>
<td>each spline term, and each term allows multiple variables.</td>
<td>expansions, and each term allows only one or two variables. Also</td>
</tr>
<tr>
<td></td>
<td></td>
<td>allows loess smoothers.</td>
</tr>
<tr>
<td>Fitting generalized</td>
<td>Fits models that have fixed smoothness terms by optimizing penalized</td>
<td>Fits models that have fixed smoothness terms by fitting partial</td>
</tr>
<tr>
<td>additive models</td>
<td>likelihood. For models that have varying smoothness terms, PROC</td>
<td>residuals against each smoothing term. For models that have varying</td>
</tr>
<tr>
<td></td>
<td>GAMMOD estimates smoothing parameters simultaneously by optimizing</td>
<td>smoothness terms, PROC GAM estimates each smoothing parameter by</td>
</tr>
<tr>
<td></td>
<td>global model fit criteria such as generalized cross validation (GCV).</td>
<td>optimizing the local GCV criterion for one spline term at a time.</td>
</tr>
<tr>
<td>Distribution families and link</td>
<td>Supports more distributions, including the negative binomial and the</td>
<td>Supports major distribution families and the canonical link function</td>
</tr>
<tr>
<td>link functions</td>
<td>Tweedie families. Also supports any applicable link functions for each</td>
<td>for each distribution.</td>
</tr>
<tr>
<td></td>
<td>distribution.</td>
<td></td>
</tr>
<tr>
<td>Testing smoothing components</td>
<td>Tests the total contribution for a spline term.</td>
<td>Tests the existence of nonlinearity for a spline term beyond the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>linear trend.</td>
</tr>
<tr>
<td>Model inference</td>
<td>A global Bayesian posterior covariance matrix is available. The</td>
<td>A local Bayesian posterior covariance matrix is available for each</td>
</tr>
<tr>
<td></td>
<td>confidence limits for each observation’s prediction is available, in</td>
<td>spline term. Only the componentwise confidence limits are available.</td>
</tr>
<tr>
<td></td>
<td>addition to componentwise confidence limits.</td>
<td></td>
</tr>
<tr>
<td>Model degrees of freedom</td>
<td>Uses the trace of the global degrees-of-freedom matrix.</td>
<td>Uses the summation of the traces of individual smoothing matrix for</td>
</tr>
<tr>
<td></td>
<td></td>
<td>each smoothing term.</td>
</tr>
<tr>
<td>Multithreading scheme</td>
<td>Specifically designed to operate in SAS Viya and performs computations</td>
<td>Executes in a single thread on a single machine.</td>
</tr>
<tr>
<td></td>
<td>in multiple threads.</td>
<td></td>
</tr>
</tbody>
</table>
Using CAS Sessions and CAS Engine Librefs

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:

```
proc options option=(CASHOST CASPORT);
run;
```

In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:

```
cas mysess;
libname mycas cas sessref=mysess;
```

The CAS statement creates the CAS session named `mysess`, and the LIBNAME statement creates the `mycas` CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the CASHOST and CASPORT of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the `mysess` session, you can terminate it by using the TERMINATE option in the CAS statement as follows:

```
cas mysess terminate;
```

For more information about the CAS and LIBNAME statements, see the section “Introduction to Shared Concepts” on page 10 in Chapter 3, “Shared Concepts.”

Getting Started: GAMMOD Procedure

This example concerns the proportions and demographic and geographic characteristics of votes that were cast in 3,107 counties in the United States in the 1980 presidential election. You can use the data set `sashelp.Vote1980` directly from the `Sashelp` library or download it from the StatLib data sets archive (Vlachos 1998). For more information about the data set, see Pace and Barry (1997).

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”
You can load the sashelp.Vote1980 data set into your CAS session by using your CAS engine libref with the following DATA step:

```sas
data mycas.Vote1980;
   set sashelp.Vote1980;
run;
```

These statements assume that your CAS engine libref is named mycas, as in the section “Using CAS Sessions and CAS Engine Librefs” on page 307, but you can substitute any appropriately defined CAS engine libref.

The data table contains 3,107 observations and seven variables. The dependent variable LogVoteRate is the logarithm transformation of the proportion of the county population who voted for any candidate. The six explanatory variables are the number of people in the county 18 years of age or older (Pop), the number of people in the county who have a 12th-grade or higher education (Edu), the number of owned housing units (Houses), the aggregate income (Income), and the scaled longitude and latitude of geographic centroids (Longitude and Latitude).

The following statements produce a plot of LogVoteRate with respect to the geographic locations Longitude and Latitude:

```sas
%let off0 = offsetmin=0 offsetmax=0
   linearopts=(thresholdmin=0 thresholdmax=0);
proc template;
   define statgraph surface;
      dynamic _title _z;
      begingraph / designwidth=defaultDesignHeight;
         entrytitle _title;
         layout overlay / xaxisopts=(&off0) yaxisopts=(&off0);
            contourplotparm z=_z y=Latitude x=Longitude / gridded=FALSE;
         endlayout;
      endgraph;
   end;
run;

proc sgrender data=mycas.Vote1980 template=surface;
   dynamic _title = 'US County Vote Proportion in the 1980 Election'
      _z = 'LogVoteRate';
run;
```

Figure 6.1 shows the map of the logarithm transformation of the proportion of the county population who voted for any candidate in the 1980 US presidential election.
The objective is to explore the nonlinear dependency structure between the dependent variable and the demographic variables (Pop, Edu, Houses, and Income), in addition to the nonlinear dependency structure between the response and the Longitude and Latitude variables. The following statements use thin-plate regression splines to fit a generalized additive model:

```plaintext
ods graphics on;
proc gammod data=mycas.Vote1980 plots seed=12345;
   model LogVoteRate = spline(Pop ) spline(Edu) spline(Houses)
                      spline(Income) spline(Longitude Latitude);
   output out=mycas.VotePred copyvars=(Longitude Latitude);
run;
```

With ODS Graphics enabled by the first statement, the PLOTS option in the PROC GAMMOD statement requests a smoothing component panel of fitted spline terms. The SEED option specifies the random seed so that you can reproduce the analysis.

The default output from this analysis is presented in Figure 6.2 through Figure 6.9.

Figure 6.2 displays the “Model Information” table. The response variable LogVoteRate is modeled by using a normal distribution whose mean is modeled by an identity link function. The GAMMOD procedure uses
the performance iteration method and the generalized cross validation (GCV) criterion as the fitting criterion. PROC GAMMOD searches for the optimum smoothing parameters by using the Newton-Raphson algorithm to optimize the fitting criterion. The random number seed is set to 12,345. Random number generation is used for sampling from observations to form spline knots and truncated eigendecomposition. Changing the random number seed might yield slightly different model fits.

**Figure 6.2** Model Information

<table>
<thead>
<tr>
<th>The GAMMOD Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Model Information</strong></td>
</tr>
<tr>
<td><strong>Data Source</strong></td>
</tr>
<tr>
<td><strong>Response Variable</strong></td>
</tr>
<tr>
<td><strong>Distribution</strong></td>
</tr>
<tr>
<td><strong>Link Function</strong></td>
</tr>
<tr>
<td><strong>Fitting Method</strong></td>
</tr>
<tr>
<td><strong>Fitting Criterion</strong></td>
</tr>
<tr>
<td><strong>Optimization Technique for Smoothing</strong></td>
</tr>
<tr>
<td><strong>Random Number Seed</strong></td>
</tr>
</tbody>
</table>

**Figure 6.3** displays the “Number of Observations” table. All 3,107 observations in the data table are used in the analysis. For data tables that have missing or invalid values, the number of used observations might be less than the number of observations read.

**Figure 6.3** Number of Observations

| **Number of Observations Read** | 3107 |
| **Number of Observations Used** | 3107 |

**Figure 6.4** displays the convergence status of the performance iteration method.

**Figure 6.4** Convergence Status

- The performance iteration converged after 2 steps.

**Figure 6.5** shows the “Fit Statistics” table. The penalized log likelihood and the roughness penalty are displayed. You can use effective degrees of freedom to compare generalized additive models with generalized linear models that do not have spline transformations. Information criteria such as Akaike’s information criterion (AIC), Akaike’s bias-corrected information criterion (AICC), and Schwarz Bayesian information criterion (BIC) can also be used for comparisons. These criteria penalize the –2 log likelihood for effective degrees of freedom. The GCV criterion is used to compare against other generalized additive models or models that are penalized.
The “Parameter Estimates” table in Figure 6.6 shows the regression parameter and dispersion parameter estimates. In this model, the intercept is the only regression parameter because (1) all variables are characterized by spline terms and no parametric effects are present and (2) the intercept absorbs the constant effect that is extracted from each spline term to make fitted splines identifiable. The dispersion parameter is estimated by maximizing the likelihood, given other model parameters.

The “Estimates for Smoothing Components” table is shown in Figure 6.7. For each spline term, the effective degrees of freedom, the estimated smoothing parameter, and the corresponding roughness penalty are displayed. The table also displays additional information about spline terms, such as the number of parameters, penalty matrix rank, and number of spline knots.

Figure 6.8 displays the hypothesis testing results for each smoothing component. The null hypothesis for each spline term is whether the total dependency on each variable is 0. The effective degrees of freedom for fit and the effective degrees of freedom for test are displayed.
Figure 6.8 Tests for Smoothing Components

<table>
<thead>
<tr>
<th>Component</th>
<th>Effective DF</th>
<th>Effective DF for Test</th>
<th>F Value</th>
<th>Pr &gt; F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spline(Pop)</td>
<td>7.80559</td>
<td>8</td>
<td>1443.64</td>
<td>&lt;.0001</td>
</tr>
<tr>
<td>Spline(Edu)</td>
<td>7.12453</td>
<td>8</td>
<td>153.94</td>
<td>&lt;.0001</td>
</tr>
<tr>
<td>Spline(Houses)</td>
<td>7.20940</td>
<td>8</td>
<td>1213.94</td>
<td>&lt;.0001</td>
</tr>
<tr>
<td>Spline(Income)</td>
<td>5.92854</td>
<td>7</td>
<td>43.17</td>
<td>&lt;.0001</td>
</tr>
<tr>
<td>Spline(Longitude Latitude)</td>
<td>18.64138</td>
<td>19</td>
<td>1619.15</td>
<td>&lt;.0001</td>
</tr>
</tbody>
</table>

Figure 6.9 displays the “Smoothing Component Panel” for all the spline terms used in the model. It displays predicted spline curves and 95% Bayesian posterior confidence bands for each univariate spline term.

The preceding program requests the prediction for each observation by default, and the OUTPUT statement saves the results in the data table mycas.VotePred. The COPYVARS= option in the OUTPUT statement copies the specified variables (Longitude and Latitude) from the input data table, mycas.Vote1980, to the output data table, mycas.VotePred. The following run of the SGRENDER procedure produces the fitted surface of the log vote proportion in the 1980 presidential election:
proc sgrender data=mycas.VotePred template=surface;
    dynamic _title='Predicted US County Vote Proportion in the 1980 Election'
                   _z='Pred';
run;

Figure 6.10 shows the map of predictions of the logarithm transformation of the proportion of county population who voted for any candidates in the 1980 US presidential election from the fitted generalized additive model.

**Figure 6.10** Predicted US County Vote Proportion in the 1980 Election

Compared to the map of the logarithm transformations of the proportion of votes cast shown in Figure 6.1, the map of the predictions of the logarithm transformations of the proportion of votes cast has a smoother surface.
Syntax: GAMMOD Procedure

The following statements are available in the GAMMOD procedure:

```plaintext
PROC GAMMOD < options > ;
   BY variables ;
   CLASS variable <(options)> . . . < variable <(options)> > < / global-options > ;
   DISPLAY < table-list > < / options > ;
   DISPLAYOUT table-spec-list < / options > ;
   FREQ variable ;
   MODEL response <(response-options)> = < PARAM(effects) >
      < spline-effects > < / model-options > ;
   MODEL events / trials = < PARAM(effects) > < spline-effects > < / model-options > ;
   OUTPUT OUT = CAS-libref.data-table < keyword < = name > > . . . < keyword < = name > > < / options >
   ;
   WEIGHT variable ;
```

The PROC GAMMOD statement and at least one MODEL statement are required. The CLASS statement can appear multiple times. If a CLASS statement is specified, it must precede the MODEL statements. The following sections describe the PROC GAMMOD statement and then describe the other statements in alphabetical order.

PROC GAMMOD Statement

```plaintext
PROC GAMMOD < options > ;
```

The PROC GAMMOD statement invokes the procedure. Table 6.3 summarizes the available options in the PROC GAMMOD statement by function. The options are then described fully in alphabetical order.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Basic Options</strong></td>
<td></td>
</tr>
<tr>
<td>ALPHA=</td>
<td>Specifies a global significance level</td>
</tr>
<tr>
<td>DATA=</td>
<td>Specifies the CAS input data table</td>
</tr>
<tr>
<td>SEED=</td>
<td>Sets the seed for pseudorandom number generation</td>
</tr>
<tr>
<td><strong>Display Options</strong></td>
<td></td>
</tr>
<tr>
<td>ITDETAILS</td>
<td>Displays the “Iteration History” table</td>
</tr>
<tr>
<td>NOCLPRINT</td>
<td>Limits or suppresses the display of classification variable levels</td>
</tr>
<tr>
<td>NOPRINT</td>
<td>Suppresses ODS output</td>
</tr>
<tr>
<td>PLOTS=</td>
<td>Controls plots that are produced through ODS Graphics</td>
</tr>
<tr>
<td><strong>Optimization Subject Options</strong></td>
<td></td>
</tr>
<tr>
<td>PLIKEOPTIONS</td>
<td>Sets optimization parameters for likelihood estimation</td>
</tr>
<tr>
<td>SMOOTHOPTIONS</td>
<td>Sets optimization parameters for smoothing parameter estimation</td>
</tr>
</tbody>
</table>
Table 6.3  continued

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Tolerance Options</strong></td>
<td></td>
</tr>
<tr>
<td>SINGCHOL=</td>
<td>Tunes the singularity criterion for Cholesky decompositions</td>
</tr>
<tr>
<td>SINGULAR=</td>
<td>Tunes the general singularity criterion</td>
</tr>
</tbody>
</table>

You can specify the following *options* in the PROC GAMMOD statement.

**ALPHA=**<var number>

specifies a global significance level for the hypothesis testing of smoothing components and the construction of Bayesian confidence bands of predictions. The confidence level is \(1 - \text{number}\). The value of \(\text{number}\) must be between 0 and 1; the default is 0.05. You can override this global significance level for Bayesian confidence bands of predictions by specifying the **ALPHA=** option in the **OUTPUT** statement.

**DATA=**<var CAS-libref.data-table>

names the input data table for PROC GAMMOD to use. The default is the most recently created data table. `<var CAS-libref.data-table>` is a two-level name, where

- `<var CAS-libref>` refers to a collection of information that is defined in the LIBNAME statement and includes the `caslib`, which includes a path to the data, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about `<var CAS-libref>`, see the section “Using CAS Sessions and CAS Engine Librefs” on page 307.

- `<var data-table>` specifies the name of the input data table.

**ITDETAILS**

adds to the “Iteration History” table the current values of the parameter estimates and their gradients. If the optimization algorithm is used to determine at least one smoothing parameter, the table lists values for smoothing parameters. If all smoothing parameters are fixed or a parametric generalized linear model is specified, the table lists values for regression parameters. These quantities are reported only for parameters that participate in the optimization.

**NOCLPRINT< =**<var number> >

suppresses the display of the “Class Level Information” table if you do not specify `number`. If you specify `number`, the values of the classification variables are displayed for only those variables whose number of levels is less than `number`. Specifying a `number` helps reduce the size of the “Class Level Information” table if some classification variables have a large number of levels.

**NOPRINT**

suppresses the generation of ODS output.

**PLIKEOPTIONS(optimization-parameters)**

specifies optimization parameters for either maximum or penalized likelihood estimation. For more information about which `optimization-parameters` you can specify, see the section “Optimization Parameters” on page 317. For the performance iteration method, only the ABSFCONV=, FCONV=,
and `MAXITER=` options are effective for determining when the performance iteration should have converged. If the distribution family involves a dispersion parameter that needs to be estimated by the maximum likelihood method, all the optimization parameters in this option are applicable.

```
PLOTS <(global-plot-option)> < plot-requests <(option)>>
```

controls the plots that are produced through ODS Graphics. When ODS Graphics is enabled, PROC GAMMOD produces by default a panel of plots of partial prediction curves or surfaces of smoothing components.

```
ods graphics on;
proc gammod plots;
  model y=spline(x1) spline(x2);
run;
ods graphics off;
```

You can specify the following `global-plot-option`, which applies to the smoothing component plots that the GAMMOD procedure generates:

**UNPACK | UNPACKPANEL**

suppresses paneling. By default, multiple smoothing component plots can appear in some output panels. Specify UNPACK to get each plot individually.

You can specify the following `plot-requests`:

**ALL**

requests that all default plots be produced.

**COMPONENTS <(component-option)>**

plots a panel of smoothing components of the fitted model. You can specify the following `component-option`:

**COMMONAXES**

requests that the smoothing component plots use a common vertical axis except for bivariate contour plots. This option enables you to visually judge the relative effect size.

**NONE**

suppresses all plots.

**SEED=number**

specifies an integer that is used to start the pseudorandom number generator for truncated eigendecomposition and for subset sampling from observations to form knots if necessary. If you do not specify this option or if `number` ≤ 0, the seed is generated from the time of day, which is read from the computer’s clock.

**SINGCHOL=number**

tunes the singularity criterion in Cholesky decomposition and matrix inversion operations. The default is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.
SINGULAR=number
tunes the singularity criterion in truncated eigendecomposition to determine its convergence. The
default is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

SMOOTHOPTIONS(optimization-parameters)
specifies optimization parameters for smoothing parameter estimation. For more information about
which optimization-parameters you can specify, see the next section. For parametric generalized
linear models or generalized additive models that have fixed smoothing parameters, any optimization
parameters that you specify for this option are ignored.

Optimization Parameters
You can specify optimization-parameters for both the PLIKEOPTIONS and SMOOTHOPTIONS options.
Depending on the modeling context, some optimization parameters might have no effect. The optimization
algorithm is considered to have converged when any one of the convergence criteria that are specified
in optimization-parameters is satisfied. Table 6.4 lists the available optimization parameters for both the
PLIKEOPTIONS and SMOOTHOPTIONS options.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSCONV=</td>
<td>Tunes the absolute function convergence criterion</td>
</tr>
<tr>
<td>ABSFCONV=</td>
<td>Tunes the absolute function difference convergence criterion</td>
</tr>
<tr>
<td>ABSGCONV=</td>
<td>Tunes the absolute gradient convergence criterion</td>
</tr>
<tr>
<td>FCONV=</td>
<td>Tunes the relative function difference convergence criterion</td>
</tr>
<tr>
<td>GCONV=</td>
<td>Tunes the relative gradient convergence criterion</td>
</tr>
<tr>
<td>MAXFUNC=</td>
<td>Specifies the maximum number of function evaluations in any optimization</td>
</tr>
<tr>
<td>MAXITER=</td>
<td>Chooses the maximum number of iterations in any optimization</td>
</tr>
<tr>
<td>MAXTIME=</td>
<td>Specifies the upper limit of CPU time (in seconds) for any optimization</td>
</tr>
<tr>
<td>MINITER=</td>
<td>Specifies the minimum number of iterations in any optimization</td>
</tr>
<tr>
<td>TECHNIQUE=</td>
<td>Selects the optimization technique</td>
</tr>
</tbody>
</table>

The optimization-parameters are fully described in the section “Optimization Options” on page 44 in
Chapter 3, “Shared Concepts.”

NOTE: By default, TECHNIQUE=NEWRAP for the performance iteration (METHOD=PERFORMANCE),
and TECHNIQUE=QUANEW for the outer iteration (METHOD=OUTER).

BY Statement

BY variables ;

You can specify a BY statement in PROC GAMMOD to obtain separate analyses of observations in groups
that are defined by the values of the BY variables. If you specify more than one BY statement, only the last
one specified is used. For more information, see the discussion of BY-group processing in SAS Language
Reference: Concepts.
CLASS Statement

```
CLASS variable <(options)> . . . < variable <(options)> > < / global-options > ;
```

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. You can list the response variable for binary models in the CLASS statement, but this is not required. Table 6.5 summarizes the values that you can use for either an option or a global-option. The options are fully documented in the section “CLASS Statement” on page 12 in Chapter 3, “Shared Concepts.”

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESCENDING</td>
<td>Reverses the sort order</td>
</tr>
<tr>
<td>MISSING</td>
<td>Treats missing values as valid levels</td>
</tr>
<tr>
<td>ORDER=</td>
<td>Specifies the sort order for the levels</td>
</tr>
<tr>
<td>PARAM=</td>
<td>Specifies the parameterization of the variable</td>
</tr>
<tr>
<td>REF=</td>
<td>Specifies the reference level of the variable</td>
</tr>
</tbody>
</table>

DISPLAY Statement

```
DISPLAY < table-list > < / options > ;
```

The DISPLAY statement enables you to specify a list of display tables to display or exclude. This statement is similar to the ODS SELECT, ODS EXCLUDE, and ODS TRACE statements. However, the DISPLAY statement can improve performance when a large number of tables could be generated (such as in BY-group processing). The procedure processes the DISPLAY statement on a CAS server and thus sends only a subset of ODS tables to the SAS client. Because ODS statements are processed on a SAS client, first all the generated display tables are sent to the client, and then the client creates a subset.

If you use both DISPLAY and ODS statements together, the DISPLAY statement takes precedence over the ODS statements. Note that the ODS EXCLUDE statement processes tables that are sent to the client after they have been filtered by the DISPLAY statement. In some cases, it might appear that the ODS EXCLUDE statement is taking precedence because it can further filter the tables. For more information about ODS, see SAS Output Delivery System: Procedures Guide.

You can specify the `table-list` as a list of table names, paths, partial pathnames, and regular expressions.

The table names that you can specify are listed in the section “ODS Table Names” on page 343. A path is a table name that is prefixed with dot-separated grouping information. For example, a SelectionSummary table that a procedure produces during a selection routine might have the path `Bygroup1.Summary:SelectionSummary`. A partial pathname does not include all groups; for example, `SelectionSummary` and `Summary:SelectionSummary` are partial pathnames for `Bygroup1.Summary:SelectionSummary`.

When you specify a table name or partial pathname, all display tables whose paths end in the specified name are selected for display or exclusion. For example, both `SelectionSummary` and `Summary:SelectionSummary` select `Bygroup1.Summary:SelectionSummary`.
A regular expression is enclosed in forward slashes (/). For example, specifying “/tions/” selects all pathnames that contain the substring “tions”; in particular, the Bygroup1.Summary_SelectionSummary table is selected. Specifying “!/tions/” selects all pathnames that do not contain the substring “tions”; in particular, the Bygroup1.Summary_SelectionSummary table is not selected.

You can specify the following options after a slash (/):

**CASESENSITIVE**
performs a case-sensitive comparison of table names in the table-list to display table names when tables are subsetted for display. To preserve case, you must enclose table names in the table-list in quotation marks.

**EXCLUDE**
displays all display tables except those that you specify in the table-list.

**EXCLUDEALL**
suppresses display of all tables. This option takes precedence over the other options.

**TRACE**
displays the display table names, labels, and paths.

---

**DISPLAYOUT Statement**

**DISPLAYOUT table-spec-list */ options */ ;

The DISPLAYOUT statement enables you to create CAS output tables from your displayed output. This statement is similar to the ODS OUTPUT statement. For more information about ODS, see SAS Output Delivery System: Procedures Guide.

The table-spec-list specifies a list of CAS output tables to create. Each entry in the list has either a key=value format or a key format:

- **key=value** specifies key as the ODS table name, path, or partial pathname, and specifies value as the CAS output table name.
- **key** specifies key as the ODS table name and also as the CAS output table name.

The ODS table names that you can specify are listed in the section “ODS Table Names” on page 343. You cannot specify the ODS table named OutputCasTables in the table-spec-list.

Table names and partial pathnames are discussed under the DISPLAY statement. The DISPLAYOUT statement does not support regular expressions.

You can specify the following options after a slash (/):

**INCLUDEALL**
creates output CAS tables for all display tables. The name of the created output CAS table is the same as the corresponding display table name. If you specify this option, the table-spec-list specification is ignored.
Chapter 6: The GAMMOD Procedure

NOREPLACE

does not replace any existing CAS output table of the same name.

REPEATED

replicates all CAS output tables on all nodes.

FREQ Statement

FREQ variable ;

The variable in the FREQ statement identifies a numeric variable in the data set that contains the frequency of occurrence of each observation. PROC GAMMOD treats each observation as if it appears \( f \) times, where \( f \) is the value of the FREQ variable for the observation. If \( f \) is not an integer, it is truncated to an integer. If \( f \) is less than 1 or missing, the observation is not used in the analysis. When the FREQ statement is not specified, each observation is assigned a frequency of 1.

MODEL Statement

MODEL response < (response-options) > = < PARAM(effects) > < spline-effects > </ model-options> ;

MODEL events / trials = < PARAM(effects) > < spline-effects > </ model-options> ;

The MODEL statement specifies the response (dependent or target) variable and the predictor (independent or explanatory) effects of the model. You can specify the response in the form of a single variable or in the form of a ratio of two variables, which are denoted events/trials. The first form applies to all distribution families; the second form applies only to summarized binomial response data. When you have binomial data, the events variable contains the number of positive responses (or events) and the trials variable contains the number of trials. The values of both events and (trials – events) must be nonnegative, and the value of trials must be positive. If you specify a single response variable that is in a CLASS statement, then the response is assumed to be binary.

You can specify parametric effects that are constructed from variables in the input data and include the effects in the parentheses of a PARAM( ) option, which can appear multiple times. For information about constructing the model effects, see the section “Specification and Parameterization of Model Effects” on page 51 in Chapter 3, “Shared Concepts.”

You can specify spline-effects by including independent variables inside the parentheses of the SPLINE( ) option. Only continuous variables (not classification variables) can be specified in spline-effects. Each spline-effect can have at least one variable and optionally some spline-options. You can specify any number of spline-effects. The following table shows some examples.

<table>
<thead>
<tr>
<th>Spline Effect Specification</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>spline(x)</td>
<td>Constructs a univariate spline by using ( x ) and uses the observed data points as knots. The maximum degrees of freedom is 10. PROC GAMMOD uses an optimization algorithm to determine the optimal smoothing parameter.</td>
</tr>
</tbody>
</table>
Table 6.5  continued

<table>
<thead>
<tr>
<th>Spline Effect Specification</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>spline(x1/knots=list(1 to 10))</td>
<td>Constructs a univariate spline by using x1 and a supplied list of knots from 1 to 10. PROC GAMMOD uses an optimization algorithm to determine the optimal smoothing parameter.</td>
</tr>
<tr>
<td>spline(x2 x3/smooth=0.3)</td>
<td>Constructs a bivariate spline by using x2 and x3 and a fixed smoothing parameter 0.3.</td>
</tr>
<tr>
<td>spline(x4 x5 x6/maxdf=40)</td>
<td>Constructs a trivariate spline by using x4, x5, and x6 and a maximum of 40 degrees of freedom. PROC GAMMOD uses an optimization algorithm to determine the optimal smoothing parameter.</td>
</tr>
</tbody>
</table>

Both parametric effects and spline effects are optional. If none are specified, a model that contains only an intercept is fitted. If only parametric effects are present, PROC GAMMOD fits a parametric generalized linear model by using the terms inside the parentheses of all PARAM( ) terms. If only spline effects are present, PROC GAMMOD fits a nonparametric additive model. If both types of effects are present, PROC GAMMOD fits a semiparametric model by using the parametric effects as the linear part of the model.

There are three sets of options in the MODEL statement. The response-options determine how the GAMMOD procedure models probabilities for binary data. The spline-options control how each spline term forms basis expansions. The model-options control other aspects of model formation and inference. Table 6.6 summarizes these options, and subsequent sections describe them in detail.

Table 6.6  MODEL Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Response Variable Options</strong></td>
<td></td>
</tr>
<tr>
<td>DESCENDING</td>
<td>Reverses the response categories</td>
</tr>
<tr>
<td>EVENT=</td>
<td>Specifies the event category</td>
</tr>
<tr>
<td>ORDER=</td>
<td>Specifies the sort order</td>
</tr>
<tr>
<td>REF=</td>
<td>Specifies the reference category</td>
</tr>
<tr>
<td><strong>Spline Effect Options</strong></td>
<td></td>
</tr>
<tr>
<td>DETAILS</td>
<td>Requests detailed spline information</td>
</tr>
<tr>
<td>DF=</td>
<td>Specifies the fixed degrees of freedom</td>
</tr>
<tr>
<td>INITSMOOTH=</td>
<td>Specifies the starting value for the smoothing parameter</td>
</tr>
<tr>
<td>KNOTS=</td>
<td>Specifies the knots to use for constructing the spline</td>
</tr>
<tr>
<td>M=</td>
<td>Specifies polynomial orders for constructing the spline</td>
</tr>
<tr>
<td>MAXDF=</td>
<td>Specifies the maximum degrees of freedom</td>
</tr>
<tr>
<td>MAXKNOTS=</td>
<td>Specifies the maximum number of knots to use for constructing the spline</td>
</tr>
<tr>
<td>MAXSMOOTH=</td>
<td>Specifies the upper bound for the smoothing parameter</td>
</tr>
<tr>
<td>MINSMOOTH=</td>
<td>Specifies the lower bound for the smoothing parameter</td>
</tr>
<tr>
<td>SMOOTH=</td>
<td>Specifies a fixed smoothing parameter</td>
</tr>
</tbody>
</table>
### Chapter 6: The GAMMOD Procedure

#### Table 6.6 continued

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Model Options</strong></td>
<td></td>
</tr>
<tr>
<td>ALLOBS</td>
<td>Requests all nonmissing values of spline variables for constructing spline</td>
</tr>
<tr>
<td></td>
<td>basis functions regardless of other model variables</td>
</tr>
<tr>
<td>CRITERION=</td>
<td>Specifies the model evaluation criterion</td>
</tr>
<tr>
<td>DISPERSION</td>
<td>PHI=</td>
</tr>
<tr>
<td>DISTRIBUTION</td>
<td>DIST=</td>
</tr>
<tr>
<td>FDHESSIAN</td>
<td>Requests a finite-difference Hessian for smoothing parameter selection</td>
</tr>
<tr>
<td>INITIALPHI=</td>
<td>Specifies the starting value of the dispersion parameter</td>
</tr>
<tr>
<td>LINK=</td>
<td>Specifies the link function</td>
</tr>
<tr>
<td>NORMALIZE</td>
<td>Requests normalized spline basis functions for model fitting</td>
</tr>
<tr>
<td>MAXPHI=</td>
<td>Specifies the upper bound for searching the dispersion parameter</td>
</tr>
<tr>
<td>METHOD=</td>
<td>Specifies the algorithm for selecting smoothing parameters</td>
</tr>
<tr>
<td>MINPHI=</td>
<td>Specifies the lower bound for searching the dispersion parameter</td>
</tr>
<tr>
<td>OFFSET=</td>
<td>Specifies the offset variable</td>
</tr>
<tr>
<td>RIDGE=</td>
<td>Specifies the ridge parameter</td>
</tr>
<tr>
<td>SCALE=</td>
<td>Specifies the method for estimating the dispersion parameter</td>
</tr>
</tbody>
</table>

#### Response Variable Options

Response variable options determine how the GAMMOD procedure models probabilities for binary data.

You can specify the following response-options by enclosing them in parentheses after the response variable.

**DESCENDING**

- DESC
  
  reverses the order of the response categories. If you specify both the DESCENDING and ORDER= options, PROC GAMMOD orders the response categories according to the ORDER= option and then reverses that order.

**EVENT='category' | FIRST | LAST**

- specifies the event category for the binary response model. PROC GAMMOD models the probability of the event category. This option has no effect when there are more than two response categories.

You can specify any of the following:

- ‘category’

  specifies that observations whose value matches category (formatted, if a format is applied) in quotation marks represent events in the data. For example, the following statements specify that observations that have a formatted value of ‘1’ represent events in the data. The probability that is modeled by the GAMMOD procedure is thus the probability that the variable def takes on the (formatted) value ‘1’.

```plaintext
proc gammod data=mycas.MyData;
   class A B C;
   model def(event = '1') = param(A B C) spline(x1 x2 x3);
run;
```
**MODEL Statement**

**FIRST** designates the first ordered category as the event.

**LAST** designates the last ordered category as the event.

By default, EVENT=FIRST.

**ORDER=FORMATTED | FREQ | INTERNAL**

specifies the sort order for the levels of the *response* variable. You can specify the following values:

**FORMATTED** sorts the levels by external formatted value, except for numeric variables that have no explicit format, which are sorted by their unformatted (internal) value. For numeric variables for which you have supplied no explicit format (that is, for which there is no corresponding FORMAT statement in the current PROC GAMMOD run or in the DATA step that created the data table), the levels are ordered by their internal (numeric) value. The sort order is machine-dependent.

**FREQ** sorts the levels by descending frequency count (levels that have the most observations come first in the order).

**INTERNAL** sorts the levels by unformatted value. The sort order is machine-dependent.

By default, ORDER=FORMATTED.

For more information about sort order, see the chapter about the SORT procedure in *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Language Reference: Concepts*.

**REF=’category’ | FIRST | LAST**

specifies the reference category for the binary response model. Specifying one response category as the reference is the same as specifying the other response category as the event category. You can specify any of the following:

‘*category*’ specifies that observations whose value matches *category* (formatted, if a format is applied) are designated as the reference.

**FIRST** designates the first ordered category as the reference

**LAST** designates the last ordered category as the reference.

By default, REF=LAST.

**Spline Effect Options**

Spline effect options control how each spline term forms basis expansions.

You can specify the following *spline-effects* inside the parentheses of a SPLINE( ) term:

**DETAILS** requests a detailed spline specification information table.
Chapter 6: The GAMMOD Procedure

**DF=** number

specifies a fixed degrees of freedom. When you specify this option, no smoothing parameter selection is performed on the spline term. If *number* is not an integer, then *number* is truncated to an integer.

**INITSMOOTH=** number

specifies the starting value for a smoothing parameter. The *number* must be nonnegative.

**KNOTS=** method

specifies the method for supplying user-defined knot values instead of using data values for constructing basis expansions. You can use the following methods for supplying the knots:

**LIST(list-of-values)**

specifies a list of values as knots for the spline construction. For a multivariate spline term, the listed values are taken as multiple row vectors, where each vector has values that are ordered by specified variables. If the last row vector of knots contains fewer values than the number of variables, then the last row vector is ignored. For example, the following specification of a spline term produces two actual knot vectors ($k_1$ and $k_2$) and the value 5 is ignored.

```plaintext
spline(x1 x2/knots=list(1 2 3 4 5))
```

**Table 6.7** Knot Values for a Bivariate Spline with a Supplied List

<table>
<thead>
<tr>
<th>x1</th>
<th>x2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k_1$</td>
<td>1 2</td>
</tr>
<tr>
<td>$k_2$</td>
<td>3 4</td>
</tr>
</tbody>
</table>

**EQUAL(n)**

specifies the number of equally spaced interior knots for every variable in a spline term. Two boundary knots are automatically added to the knot list for each variable such that the total number of knots is $(n + 2)^d$, where $d$ is the number of variables in the spline term. For a multivariate spline term, knot values for each variable are determined independently from the corresponding boundary values. For example, if the boundary points for $x_1$ are 1 and 5 and the boundary points for $x_2$ are 2 and 6, then the following specification of a spline term produces nine actual knots ($k_1$ through $k_9$), which consist of two boundary knots and one interior knot for each variable.

```plaintext
spline(x1 x2/knots=equal(1))
```

**Table 6.8** Knot Values for a Bivariate Spline with One Interior Knot

<table>
<thead>
<tr>
<th>x1</th>
<th>x2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k_1$</td>
<td>1 2</td>
</tr>
<tr>
<td>$k_2$</td>
<td>1 4</td>
</tr>
<tr>
<td>$k_3$</td>
<td>1 6</td>
</tr>
<tr>
<td>$k_4$</td>
<td>3 2</td>
</tr>
<tr>
<td>$k_5$</td>
<td>3 4</td>
</tr>
<tr>
<td>$k_6$</td>
<td>3 6</td>
</tr>
<tr>
<td>$k_7$</td>
<td>5 2</td>
</tr>
<tr>
<td>$k_8$</td>
<td>5 4</td>
</tr>
<tr>
<td>$k_9$</td>
<td>5 6</td>
</tr>
</tbody>
</table>
MODEL Statement  ♦  325

\textbf{M=number}

specifies the order of the derivative in the penalty term. The \textit{number} must be a positive integer. The default is \( \max(2, \text{int}(d/2) + 1) \), where \( d \) is the number of variables in the spline term.

\textbf{MAXDF=number}

specifies the maximum \textit{number} of degrees of freedom. When a thin-plate regression spline is formed, the specified \textit{number} is used for constructing a low-rank penalty matrix to approximate the penalty matrix via the truncated eigendecomposition. The \textit{number} must be greater than \( \left( \frac{m + d - 1}{d} \right) \), where \( m \) is the derivative order that is specified in the \textbf{M=} option, and \( d \) is the number of variables in the spline term. The default is \( 10 \times d \). For more information, see the section “Thin-Plate Regression Splines” on page 331.

\textbf{MAXKNOTS=number}

specifies the maximum \textit{number} of knots if data points are used to form knots. If \textbf{KNOTS=LIST(list-of-values)} is not specified, PROC GAMMOD forms knots from unique data points. If the number of unique data points is greater than \textit{number}, a subset of size \textit{number} is formed by random sampling from all unique data points. The \textit{number} cannot exceed the largest integer that can be stored on the CAS server. By default, \textbf{MAXKNOTS=}2000.

\textbf{MAXSMOOTH=number}

specifies the upper bound for the smoothing parameter. The default is the largest double-precision value.

\textbf{MINSMOOTH=number}

specifies the lower bound for the smoothing parameter. By default, \textbf{MINSMOOTH=}0.

\textbf{SMOOTH=number}

specifies a fixed smoothing parameter. When you specify this option, no smoothing parameter selection is performed on the spline term.

\textbf{Model Options}

You can specify the following \textit{model-options} in the MODEL statement after a slash (/):

\textbf{ALLOBs}

requests that all nonmissing values of the variables be used in a spline term for constructing the spline basis functions, regardless of whether other model variables are missing.

\textbf{CRITERION=criterion}

specifies the model evaluation criterion for selecting smoothing parameters for \textit{spline-effects}. You can specify the following values:

\textbf{GACV<(FACTOR=number | GAMMA=number)>}

uses the generalized approximate cross validation (GACV) criterion to evaluate models.

\textbf{GCV<(FACTOR=number | GAMMA=number)>}

uses the generalized cross validation (GCV) criterion to evaluate models.
UBRE<(FACTOR=number | GAMMA=number)>
uses the unbiased risk estimator (UBRE) criterion to evaluate models.

The default criterion depends on the value of the DISTRIBUTION= option. For distributions that involve dispersion parameters, GCV is the default. For distributions without dispersion parameters, UBRE is the default. For all three criteria, you can optionally use the FACTOR= option to specify an extra tuning parameter in order to penalize more for model roughness. The value of number must be greater than or equal to 1. For more information about the model evaluation criteria, see the section “Model Evaluation Criteria” on page 335.

DISPERSION=number
PHI=number
specifies a fixed dispersion parameter for distributions that have a dispersion parameter. The dispersion parameter that is used in all computations is fixed at number; it is not estimated.

DISTRIBUTION=keyword
specifies the response distribution for the model. The keywords and their associated distributions are shown in Table 6.9.

Table 6.9 Built-In Distribution Functions

<table>
<thead>
<tr>
<th>DISTRIBUTION=</th>
<th>Distribution Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>BINARY</td>
<td>Binary</td>
</tr>
<tr>
<td>BINOMIAL</td>
<td>Binary or binomial</td>
</tr>
<tr>
<td>GAMMA</td>
<td>Gamma</td>
</tr>
<tr>
<td>INVERSEGAUSSIAN</td>
<td>IG</td>
</tr>
<tr>
<td>NEGATIVEBINOMIAL</td>
<td>NB</td>
</tr>
<tr>
<td>NORMAL</td>
<td>GAUSSIAN</td>
</tr>
<tr>
<td>POISSON</td>
<td>Poisson</td>
</tr>
<tr>
<td>TWEEDIE&lt;(Tweedie-options)&gt;</td>
<td>Tweedie</td>
</tr>
</tbody>
</table>

When DISTRIBUTION=TWEEDIE, you can specify the following Tweedie-options:

INITIALP=
specifies a starting value for iterative estimation of the Tweedie power parameter.

P=
requests a fixed Tweedie power parameter.

If you do not specify a link function in the LINK= option, a default link function is used. The default link function for each distribution is shown in Table 6.10. You can use any link function shown in Table 6.11 by specifying the LINK= option. Other commonly used link functions for each distribution are shown in Table 6.10.
Table 6.10 Default and Commonly Used Link Functions

<table>
<thead>
<tr>
<th>DISTRIBUTION=</th>
<th>Default Link Function</th>
<th>Other Commonly Used Link Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>BINARY</td>
<td>Logit</td>
<td>Probit, complementary log-log, log-log</td>
</tr>
<tr>
<td>BINOMIAL</td>
<td>Logit</td>
<td>Probit, complementary log-log, log-log</td>
</tr>
<tr>
<td>GAMMA</td>
<td>Log</td>
<td>Reciprocal</td>
</tr>
<tr>
<td>INVERSEGAUSSIAN</td>
<td>Log</td>
<td>Reciprocal square</td>
</tr>
<tr>
<td>NEGATIVEBINOMIAL</td>
<td>Log</td>
<td>Identity</td>
</tr>
<tr>
<td>NORMAL</td>
<td>Identity</td>
<td>Log</td>
</tr>
<tr>
<td>POISSON</td>
<td>Log</td>
<td></td>
</tr>
<tr>
<td>TWEEDIE</td>
<td>Log</td>
<td></td>
</tr>
</tbody>
</table>

FDHESSIAN
requests that the second-order derivatives (Hessian) be computed using finite-difference approximations based on evaluation of the first-order derivatives (gradient). This option might be useful if the analytical Hessian takes a long time to compute.

INITIALPHI=number
specifies a starting value for iterative maximum likelihood estimation of the dispersion parameter for distributions that have a dispersion parameter.

LINK=keyword
specifies the link function for the model. The keywords and the associated link functions are shown in Table 6.11. Default and commonly used link functions for the available distributions are shown in Table 6.10.

Table 6.11 Built-In Link Functions

<table>
<thead>
<tr>
<th>LINK=</th>
<th>Link Function</th>
<th>$g(\mu) = \eta =$</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLOGLOG</td>
<td>Complementary log-log</td>
<td>$\log(-\log(1 - \mu))$</td>
</tr>
<tr>
<td>IDENTITY</td>
<td>Identity</td>
<td>$\mu$</td>
</tr>
<tr>
<td>INV</td>
<td>Reciprocal</td>
<td>$1/\mu$</td>
</tr>
<tr>
<td>INV2</td>
<td>Reciprocal square</td>
<td>$1/\mu^2$</td>
</tr>
<tr>
<td>LOG</td>
<td>Logarithm</td>
<td>$\log(\mu)$</td>
</tr>
<tr>
<td>LOGIT</td>
<td>Logit</td>
<td>$\log(\mu/(1 - \mu))$</td>
</tr>
<tr>
<td>LOGLOG</td>
<td>Log-log</td>
<td>$-\log(-\log(\mu))$</td>
</tr>
<tr>
<td>PROBIT</td>
<td>Probit</td>
<td>$\Phi^{-1}(\mu)$</td>
</tr>
</tbody>
</table>

$\Phi^{-1}(\cdot)$ denotes the quantile function of the standard normal distribution.

MAXPHI=number
specifies an upper bound for maximum likelihood estimation of the dispersion parameter for distributions that have a dispersion parameter.
Chapter 6: The GAMMOD Procedure

METHOD=OUTER | PERFORMANCE
specifies the algorithm for selecting smoothing parameters for spline-effects. You can specify the following values:

OUTER specifies the outer iteration method for selecting smoothing parameters. For more information about this method, see the section “Outer Iteration” on page 336.

PERFORMANCE specifies the performance iteration method for selecting smoothing parameters. For more information about this method, see the section “Performance Iteration” on page 337.

By default, METHOD=PERFORMANCE.

MINPHI=number
specifies a lower bound for maximum likelihood estimation of the dispersion parameter for distributions that have a dispersion parameter.

NORMALIZE requests normalized spline basis functions for model fitting. After the regression spline basis functions are computed, each column is standardized to have a unit standard error. The corresponding penalty matrix is also scaled accordingly. This option might be helpful when you have badly scaled data.

OFFSET=variable
specifies a variable to be used as an offset to the linear predictor. An offset plays the role of an effect whose coefficient is known to be 1. The offset variable cannot appear in the CLASS statement or elsewhere in the MODEL statement. Observations that have missing values for the offset variable are excluded from the analysis.

RIDGE=number
allows a ridge parameter such that a diagonal matrix \( H_i \) is added to the optimization problem with respect to regression parameters:

\[
\min (y - X\beta)'(y - X\beta) + \beta' S\beta + \beta' H \beta
\]

with respect to \( \beta \)

By default, RIDGE=0. Specifying a small ridge parameter might be helpful if the model matrix \( X'X + S \) is close to singular.

SCALE=DEVIANCE | MLE | PEARSON
specifies the method for estimating the scale and dispersion parameters. You can specify the following values:

DEVIANCE estimates the dispersion parameter by using the deviance statistic.

MLE computes the dispersion parameter by maximizing the likelihood or penalized likelihood.

PEARSON estimates the dispersion parameter by using Pearson’s statistic.

By default, SCALE=MLE.
The OUTPUT statement creates a data table that contains observationwise statistics that PROC GAMMOD computes after fitting the model. In order to avoid data duplication for large data tables, the variables in the input data table are not included in the output data table unless you specify them in the COPYVARS= option.

The computation of the output statistics is based on the final parameter estimates. If the model fit does not converge, missing values are produced for the quantities that depend on the estimates.

You must specify the following option:

```
OUT=CAS-libref.data-table
```

names the output data table for PROC GAMMOD to use. You must specify this option before any other options. CAS-libref.data-table is a two-level name, where

- **CAS-libref** refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to where the data table is to be stored, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about CAS-libref, see the section “Using CAS Sessions and CAS Engine Librefs” on page 307.

- **data-table** specifies the name of the output data table.

You can also specify the following syntax elements:

- **ALPHA=number**
  specifies the significance level for the construction of confidence intervals in the output data table. The confidence level is $1 - number$. The value of number must be between 0 and 1. By default, number is equal to the value of the ALPHA= option in the PROC GAMMOD statement, or 0.05 if that option is not specified.

- **COMPONENT**
  requests componentwise statistics for all spline terms if LINP, LOWER, STD, or UPPER is specified as a keyword.

- **COPYVAR=variable**
  - **COPYVARS=(variables)**
    transfers one or more variables from the input data table to the output data table.

- **keyword <=name>**
  specifies a statistic to include in the output data table and optionally assigns a name to the variable. If you do not provide a name, the GAMMOD procedure assigns a default name based on the keyword.

You can specify the following keywords for adding statistics to the OUTPUT data table:
Chapter 6: The GAMMOD Procedure

LINP | XBETA
requests the linear predictor $\eta = x^T \beta$. For observations in which only the response variable is missing, values of the linear predictor are computed even though these observations do not affect the model fit. The default name is Xbeta.

LOWER
requests a lower Bayesian confidence band value for the predicted value. The default name is Lower.

PEARSON | PEARLS | RESCHI
requests the Pearson residual, $(y - \mu) / \sqrt{V(\mu)}$, where $\mu$ is the estimate of the predicted response mean and $V(\mu)$ is the response distribution variance function. The default name is Pearson.

PREDICTED | PRED | P
requests predicted values for the response variable. For observations in which only the response variable is missing, the predicted values are computed even though these observations do not affect the model fit. The default name is Pred.

RESIDUAL | RESID | R
requests the raw residual, $y - \mu$, where $\mu$ is the estimate of the predicted mean. The default name is Residual.

STD
requests a standard error for the linear predictor. The default name is Std.

UPPER
requests an upper Bayesian confidence band value for the predicted value. The default name is Upper.

WEIGHT Statement

WEIGHT variable;

The variable in the WEIGHT statement is used as a weight to perform a weighted analysis of the data. Observations that have nonpositive or missing weights are not included in the analysis. If a WEIGHT statement is not included, then all observations that are used in the analysis are assigned a weight of 1.

Details: GAMMOD Procedure

Missing Values

Any observation that has missing values for the response, frequency, weight, offset, or explanatory variables is excluded from the analysis; however, missing values are valid for response and explanatory variables if you specify the MISSING option in the CLASS statement. Observations that have a nonpositive weight or a frequency less than 1 are also excluded. For Poisson, negative binomial, and Tweedie distributions,
observations that have a negative response value are excluded. For gamma and inverse Gaussian distributions, observations that have a nonpositive response value are excluded.

The estimated linear predictor and the fitted means are not computed for any observation that has missing offset or explanatory variables. However, if only the response value is missing, the linear predictor and the fitted means can be computed and output to a data table by using the OUTPUT statement.

**Thin-Plate Regression Splines**

The GAMMOD procedure uses thin-plate regression splines (Wood 2003) to construct spline basis expansions. The thin-plate regression splines are based on thin-plate smoothing splines (Duchon 1976, 1977). Compared to thin-plate smoothing splines, thin-plate regression splines produce fewer basis expansions and thus make direct fitting of generalized additive models possible.

**Thin-Plate Smoothing Splines**

Consider the problem of estimating a smoothing function \( f \) of \( x \) with \( d \) covariates from \( n \) observations. The model assumes

\[
y_i = f(x_i) + \epsilon_i, \quad i = 1, \ldots, n
\]

Then the thin-plate smoothing splines estimate the smoothing function \( f \) by minimizing the penalized least squares function:

\[
\sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda J_{m,d}(f)
\]

The penalty term \( \lambda J_{m,d}(f) \) includes the function that measures roughness on the \( f \) estimate:

\[
J_{m,d}(f) = \int \cdots \int \sum_{\alpha_1 + \cdots + \alpha_d = m}^{m!} \frac{\alpha_1! \cdots \alpha_d! m!}{\partial x_1^{\alpha_1} \cdots \partial x_d^{\alpha_d}} \left( \frac{\partial^m f}{\partial x_1^{\alpha_1} \cdots \partial x_d^{\alpha_d}} \right)^2 \, dx_1 \cdots dx_d
\]

The parameter \( m \) (which corresponds to the M= option for a spline effect) specifies how the penalty is applied to the function roughness. Function derivatives whose order is less than \( m \) are not penalized. The relation \( 2m > d \) must be satisfied.

The penalty term also includes the smoothing parameter \( \lambda \in [0, \infty) \), which controls the trade-off between the model’s fidelity to the data and the function smoothness of \( f \). When \( \lambda = 0 \), the function estimate corresponds to an interpolation. When \( \lambda \rightarrow \infty \), the function estimate becomes the least squares fit. By using the defined penalized least squares criterion and a fixed \( \lambda \) value, you can explicitly express the estimate of the smooth function \( f \) in the following form:

\[
f_\lambda(x) = \sum_{j=1}^{M} \theta_j \phi_j(x) + \sum_{i}^{n} \delta_i \eta_{m,d}(\|x - x_i\|)
\]

In the expression of \( f_\lambda(x) \), \( \delta_i \) and \( \theta_j \) are coefficients to be estimated. The functions \( \phi_j(x) \) correspond to unpenalized polynomials of \( x \) with degrees up to \( m - 1 \). The total number of these polynomials is
Chapter 6: The GAMMOD Procedure

The function \( \eta_{m,d} \) models the extra nonlinearity besides the polynomials and is a function of the Euclidean distance \( r \) between any \( x \) value and an observed \( x_i \) value:

\[
\eta_{m,d}(r) = \begin{cases} 
\frac{(-1)^{m+1+d/2}}{2^{m-1} \pi^{d/2} (m-1)! (m-d/2)!} r^{2m-d} \log(r) & \text{if } d \text{ is even} \\
\frac{\Gamma(d/2-m)}{2^{m} \pi^{d/2} (m-1)!} r^{2m-d} & \text{if } d \text{ is odd}
\end{cases}
\]

Define the penalty matrix \( E \) such that each entry \( E_{ij} = \eta_{m,d}(\|x_i - x_j\|) \), let \( y \) be the vector of the response, let \( T \) be the matrix where each row is formed by \( \phi_j(x) \), and let \( \theta \) and \( \delta \) be vectors of coefficients \( \theta_j \) and \( \delta_j \).

Then you can obtain the function estimate \( f \) from the following minimization problem:

\[
\min \| y - T\theta - E\delta \|^2 + \lambda \delta' E \delta \quad \text{subject to} \quad T'\delta = 0
\]

Low-Rank Approximation

Given the representation of the thin-plate smoothing spline, the estimate of \( f \) involves as many parameters as the number of unique data points. Solving \((\theta, \delta)\) with an optimum \( \lambda \) becomes difficult for large problems.

Because the matrix \( E \) is symmetric and nonnegative definite, the eigendecomposition can be taken as \( E = U D U' \), where \( D \) is the diagonal matrix of eigenvalues \( d_i \) of \( E \), and \( U \) is the matrix of eigenvectors that corresponds to \( d_i \). The truncated eigendecomposition forms \( \tilde{E}_k \), which is an approximation to \( E \) such that

\[
\tilde{E}_k = U_k D_k U_k'
\]

where \( D_k \) is a diagonal matrix that contains the \( k \) most extreme eigenvalues in descending order of absolute values: \( |\tilde{d}_1| > \cdots > |\tilde{d}_k| \). \( U_k \) is the matrix that is formed by columns of eigenvectors that correspond to the eigenvalues in \( D_k \).

The approximation \( \tilde{E}_k \) not only reduces the dimension from \( n \times n \) of \( E \) to \( n \times k \) but also is optimal in two senses. First, \( \tilde{E}_k \) minimizes the spectral norm \( \|E - F_k\|_2 \) between \( E \) and all rank \( k \) matrices \( F_k \). Second, \( \tilde{E}_k \) also minimizes the worst possible change that is introduced by the eigenspace truncation as defined by

\[
\max_{\delta \neq 0} \frac{\delta'(E - G_k)\delta}{\|\delta\|^2}
\]

where \( G_k \) is formed by any \( k \) eigenvalues and corresponding eigenvectors. For more information, see Wood (2003).

Now given \( E \approx \tilde{E}_k \) and \( \tilde{E}_k = U_k D_k U_k' \), and letting \( \delta_k = U_k' \delta \), the minimization problem becomes

\[
\min \| y - T\theta - U_k D_k \delta_k \|^2 + \lambda \delta_k' D_k \delta_k \quad \text{subject to} \quad T'U_k \delta_k = 0
\]

You can turn the constrained optimization problem into an unconstrained one by using any orthogonal column basis \( Z \). One way to form \( Z \) is via the QR decomposition of \( U_k' T \):

\[
U_k' T = [Q_1 Q_2] \begin{bmatrix} R \\ 0 \end{bmatrix}
\]

Let \( Z = Q_2 \). Then it is verified that

\[
T' U_k Z = R'Q_1 Q_2 = 0
\]
So for $\delta_k$ such that $T^T U_k \delta_k = 0$, it is true that $\delta_k = Z \tilde{\delta}$. Now the problem becomes the unconstrained optimization,

$$\min \| y - T \theta - U_k D_k Z \tilde{\delta} \|^2 + \lambda \tilde{\delta}' D_k Z \tilde{\delta}$$

Let

$$\beta = \begin{bmatrix} \theta \\ \delta \end{bmatrix}, \quad X = [T : U_k D_k Z], \quad \text{and} \quad S = \begin{bmatrix} 0 & 0 \\ 0 & D_k Z \end{bmatrix}$$

The optimization is simplified as

$$\min \| y - X \beta \|^2 + \lambda \beta' S \beta$$

with respect to $\beta$

---

**Generalized Additive Models**

**Generalized Linear Models**

All probability distributions that the GAMMOD procedure fits are members of an exponential family of distributions. For the specification of an exponential family, see the section “Exponential Family Distributions” in Chapter 7, “The GENSELECT Procedure.”

Table 6.12 lists and defines some common notation that is used in the context of generalized linear models and generalized additive models.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\ell$</td>
<td>Log-likelihood</td>
</tr>
<tr>
<td>$\ell_p$</td>
<td>Penalized log-likelihood</td>
</tr>
<tr>
<td>$D$</td>
<td>Deviance</td>
</tr>
<tr>
<td>$D_p$</td>
<td>Penalized deviance</td>
</tr>
<tr>
<td>$g$</td>
<td>Link function</td>
</tr>
<tr>
<td>$g^{-1}$</td>
<td>Inverse link function</td>
</tr>
<tr>
<td>$\mu$</td>
<td>Response mean $\mu = g^{-1}(\eta)$</td>
</tr>
<tr>
<td>$\eta$</td>
<td>Linear predictor $\eta = X \beta$</td>
</tr>
<tr>
<td>$\phi$</td>
<td>Dispersion parameter</td>
</tr>
<tr>
<td>$z$</td>
<td>Column of adjusted response variable</td>
</tr>
<tr>
<td>$v$</td>
<td>Column of response variance</td>
</tr>
<tr>
<td>$\omega_i$</td>
<td>Prior weight for each observation</td>
</tr>
<tr>
<td>$w_i$</td>
<td>Adjusted weight for each observation</td>
</tr>
<tr>
<td>$W$</td>
<td>Diagonal matrix of adjusted weights</td>
</tr>
</tbody>
</table>

The GAMMOD procedure supports the following distributions: binary, binomial, gamma, inverse Gaussian, negative binomial, normal (Gaussian), Poisson, and Tweedie.

For forms of log likelihood functions for each of the probability distributions, see the section “Log-Likelihood Functions” in Chapter 7, “The GENSELECT Procedure.”
**Generalized Additive Models**

Generalized additive models are extensions of generalized linear models (Nelder and Wedderburn 1972). For each observation that has a response $Y_i$ and a row vector of the model matrix $x_i$, both generalized additive models and generalized linear models assume the model additivity,

$$g(\mu_i) = f_1(x_{i1}) + \cdots + f_p(x_{ip})$$

where $\mu_i = E(Y_i)$ and $Y_i$ is independently distributed in some exponential family. Generalized linear models further assume model linearity by $f_j(x_{ij}) = x_{ij}\beta_j$ for $j = 1, \ldots, p$. Generalized additive models relax the linearity assumption by allowing some smoothing functions $f_j$ to characterize the dependency. The GAMMOD procedure constructs the smoothing functions by using thin-plate regression splines.

Consider a generalized additive model that has some linear terms $X_L$ with coefficients $\beta_L$ and $p$ smoothing functions $f_j$. Each smoothing function can be constructed by thin-plate regression splines with a smoothing parameter $\lambda_j$. Using the notations described in the section “Low-Rank Approximation” on page 332, you can characterize each smoothing function by

$$\beta_j = \begin{bmatrix} \theta_j \\ \delta_j \end{bmatrix}, \quad X_j = [T_1 : U_{kj}D_{kj}Z_j], \quad \text{and} \quad S_j = \begin{bmatrix} 0 & 0 \\ 0 & Z_j'Z_j \end{bmatrix}$$

Notice that each smoothing function representation contains a zero-degree polynomial that corresponds to a constant. Having multiple constant terms makes the smoothing functions unidentifiable. The solution is to include a global constant term (that is, the intercept) in the model and enforce the centering constraint to each smoothing function. You can write the constraint as

$$1'X_j\beta_j = 0$$

By using a similar approach as the linear constraint for thin-plate regression splines, you obtain the orthogonal column basis $V_j$ via the QR decomposition of $X_j1$ such that $1'X_jV_j = 0$. Each smoothing function can be reparameterized as $\hat{X}_j = X_jV_j$.

Let $X = [X_L : \hat{X}_1 : \cdots : \hat{X}_p]$ and $\beta' = [\beta_L' : \beta_1' : \cdots : \beta_p']$. Then the generalized additive model can be represented as $g(\mu) = X\beta$. The roughness penalty matrix is represented as a block diagonal matrix:

$$S_\lambda = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & \lambda_1S_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_pS_p \end{bmatrix}$$

Then the roughness penalty is measured in the quadratic form $\beta'S_\lambda \beta$.

**Penalized Likelihood Estimation**

Given a fixed vector of smoothing parameters, $\lambda = [\lambda_1 \cdots \lambda_p]'$, you can fit the generalized additive models by the penalized likelihood estimation. In contrast to the maximum likelihood estimation, penalized likelihood estimation obtains an estimate for $\beta$ by maximizing the penalized log likelihood,

$$\ell_p(\beta) = \ell(\beta) - \frac{1}{2} \beta'S_\lambda \beta$$
Any optimization technique that you can use for maximum likelihood estimation can also be used for penalized likelihood estimation. If first-order derivatives are required for the optimization technique, you can compute the gradient as

\[
\frac{\partial \ell_p}{\partial \beta} = \frac{\partial \ell}{\partial \beta} - S_\lambda \beta
\]

If second-order derivatives are required for the optimization technique, you can compute the Hessian as

\[
\frac{\partial^2 \ell_p}{\partial \beta \partial \beta'} = \frac{\partial^2 \ell}{\partial \beta \partial \beta'} - S_\lambda
\]

In the gradient and Hessian forms, \(\partial \ell / \partial \beta\) and \(\partial^2 \ell / (\partial \beta \partial \beta')\) are the corresponding gradient and Hessian, respectively, for the log likelihood for generalized linear models. For more information about optimization techniques, see the section “Optimization Options” on page 44 in Chapter 3, “Shared Concepts.”

### Model Evaluation Criteria

Given a fixed set of smoothing parameters \(\lambda\) in which each \(\lambda_i\) controls the smoothness of each spline term, you can fit a generalized additive model by penalized likelihood estimation. There are infinitely many sets of smoothing parameters. In order to search optimum models, some model evaluation criteria need to be defined to quantify the model’s goodness of fit. The GAMMOD procedure uses the following model evaluation criteria:

- generalized cross validation (GCV), \(V_g\) (Craven and Wahba 1979)
- unbiased risk estimator (UBRE), \(V_u\) (Craven and Wahba 1979)
- generalized approximate cross validation (GACV), \(V_a\) (Xiang and Wahba 1996)

Consider the optimization problem

\[
\min (y - X\beta)'(y - X\beta) + \beta'S_\lambda \beta \quad \text{with respect to} \quad \beta
\]

The parameter estimate for \(\beta\) can be represented as

\[
\hat{\beta} = (X'X + S_\lambda)^{-1}X'y
\]

And the smoothing matrix (also called the influence matrix or hat matrix) is thus represented as

\[
H_\lambda = X(X'X + S_\lambda)^{-1}X'
\]

With the defined smoothing matrix, you can form the model evaluation criteria as follows:

\[
V_{g}(\lambda) = \frac{n \| y - H_\lambda y \|^2}{\text{tr}(I - y H_\lambda)^2}
\]

\[
V_{u}(\lambda) = \frac{1}{n} \| y - H_\lambda y \|^2 - \frac{2}{n} \sigma^2 \text{tr}(I - y H_\lambda) + \sigma^2
\]

\[
V_{a}(\lambda) = \frac{1}{n} \| y - H_\lambda y \|^2 \left( 1 + 2\gamma \frac{\text{tr}(H_\lambda)}{\text{tr}(I - H_\lambda)} \right)
\]
In the equations, \( \gamma \geq 1 \) (which corresponds to the GAMMA= suboption of the CRITERION= option) is the tuning parameter that is sometimes used to enforce smoother models.

The GAMMOD procedure uses fitting algorithms that involve minimizing the model evaluation criterion with respect to unknown smoothing parameters \( \lambda \).

### Fitting Algorithms

For models that assume a normally distributed response variable, you can minimize the model evaluation criteria directly by searching for optimal smoothing parameters. For models that have nonnormal distributions, you cannot use the model evaluation criteria directly because the involved statistics keep changing between iterations. The GAMMOD procedure enables you to use either of two fitting approaches to search for optimum models: the outer iteration method and the performance iteration method. The outer iteration method modifies the model evaluation criteria so that a global objective function can be minimized in order to find the best smoothing parameters. The performance iteration method minimizes a series of objective functions in an iterative fashion and then obtains the optimum smoothing parameters at convergence. For large data tables, the performance iteration method usually converges faster than the outer iteration method.

#### Outer Iteration

The outer iteration method is outlined in Wood (2006). The method uses modified model evaluation criteria, which are defined as follows:

\[
\mathcal{V}_{g}(\lambda) = \frac{n D_{\lambda}(\mu)}{(n - \gamma \text{tr}(H_{\lambda}))^2}
\]

\[
\mathcal{V}_{o}(\lambda) = \frac{D_{\lambda}(\mu)}{n} - \frac{2 - \sigma^2}{n} \text{tr}(I - \gamma H_{\lambda}) + \sigma^2
\]

\[
\mathcal{V}_{d}(\lambda) = \frac{D_{\lambda}(\mu)}{n} + \frac{2 \gamma \text{tr}(H_{\lambda}) P_{\lambda}}{n \text{tr}(I - H_{\lambda})}
\]

By replacing \( \|y - H_{\lambda}y\|^2 \) with model deviance \( D_{\lambda}(\mu) \), the modified model evaluation criteria relate to the smoothing parameter \( \lambda \) in a direct way such that the analytic gradient and Hessian are available in explicit forms. Pearson’s statistic \( P_{\lambda} \) is used in the GACV criterion \( \mathcal{V}_{o}(\lambda) \) (Wood 2008). The algorithm for the outer iteration is thus as follows:

1. Initialize smoothing parameters by taking one step of performance iteration based on adjusted response and adjusted weight except for spline terms whose initial values are specified in the INITSMOOTH= option.

2. Search for the best smoothing parameters by minimizing the modified model evaluation criteria. The optimization process stops when any of the convergence criteria that are specified in the SMOOTHOPTIONS option is met. At each optimization step:
   a) Initialize by setting initial regression parameters \( \beta = \{g(\hat{y}), 0, \ldots, 0\}' \). Set the initial dispersion parameter if necessary.
   b) Search for the best regression parameters \( \beta \) by minimizing the penalized deviance \( D_{\rho} \) (or maximizing the penalized likelihood \( \ell_{\rho} \) for negative binomial distribution). The optimization
process stops when any of the convergence criteria that are specified in the PLIKEOPTIONS option is met.

c) At convergence, evaluate derivatives of the model evaluation criteria with respect to \( \lambda \) by using

\[
\frac{\partial D_p}{\partial \beta}, \frac{\partial^2 D_p}{\partial \beta \partial \beta'}, \frac{\partial \beta}{\partial \lambda_j}, \text{ and } \frac{\partial^2 \beta}{\partial \lambda_j \partial \lambda_k}.
\]

Step 2b usually converges quickly because it is essentially penalized likelihood estimation given that \( D_p = 2\phi(\ell_{\text{max}} - \ell) + \beta S_\lambda \beta' \). Step 2c contains involved computation by using the chain rule of derivatives. For more information about computing derivatives of \( V_g^p \) and \( V_u^p \), see Wood (2008, 2011).

**Performance Iteration**

The performance iteration method is proposed by Gu and Wahba (1991). Wood (2004) modifies and stabilizes the algorithm for fitting generalized additive models. The algorithm for the performance iteration method is as follows:

1. Initialize smoothing parameters \( \lambda = \{1, \ldots, 1\} \), except for spline terms whose initial values are specified in the INITSMOOTH= option. Set the initial regression parameters \( \beta = \{g(\bar{y}), 0, \ldots, 0\}' \). Set the initial dispersion parameter if necessary.

2. Repeat the following steps until any of these three conditions is met: (1) the absolute function change in penalized likelihood is sufficiently small; (2) the absolute relative function change in penalized likelihood is sufficiently small; (3) the number of iterations exceeds the maximum iteration limit.

   a) Form the adjusted response and adjusted weight from \( \mu = g^{-1}(\eta) \). For each observation,

\[
z_i = \eta_i + (y_i - \mu_i)/\mu_i', \quad w_i = \omega_i \mu_i'^2/v_i
\]

   b) Search for the best smoothing parameters for the current iteration based on valid adjusted response values and adjusted weight values.

   c) Use the smoothing parameters to construct the linear predictor and the predicted means.

   d) Obtain an estimate for the dispersion parameter if necessary.

In step 2b, you can use different optimization techniques to search for the best smoothing parameters. The Newton-Raphson optimization is efficient in finding the optimum \( \lambda \) where the first- and second-order derivatives are available. For more information about computing derivatives of \( V_g^p \) and \( V_u^p \) with respect to \( \lambda \), see Wood (2004).

**Degrees of Freedom**

Let \( W \) be the adjusted weight matrix at convergence, and let \( S_\lambda \) be the roughness penalty matrix with selected smoothing parameters. The degrees of freedom matrix is defined as in Wood (2006):

\[
F = (X'WX + S_\lambda)^{-1}X'WX
\]

Given the adjusted response \( z \), the parameter estimate is shown to be \( \tilde{\beta} = (X'WX)^{-1}X'Wz \) for the model without penalization, and the parameter estimate is \( \hat{\beta} = (X'WX + S_\lambda)^{-1}X'Wz = F\tilde{\beta} \) with penalization. \( F \) is thus the matrix that projects or maps the unpenalized parameter estimates to the penalized ones.
The model degrees of freedom is given as
\[ df = \text{tr}(F) \]
And the degrees of freedom for error is given as
\[ df_r = n - 2df + \text{tr}(FF) \]
For the \( j \)th spline term, the degrees of freedom for the component is defined to be the trace of the submatrix of \( F \) that corresponds to parameter estimates \( \beta_j \):
\[ df_j = \text{tr}(F_j) \]
The degrees of freedom for the smoothing component test of the \( j \)th term is defined similarly as
\[ df_t^j = 2df_j - \text{tr}((FF)_j) \]

**Model Inference**

Wahba (1983) proposes a Bayesian covariance matrix for parameter estimates \( \beta \) by interpreting a smoothing spline as a posterior mean. Nychka (1988) shows that the derived Bayesian posterior confidence limits work well from frequentist viewpoints. The Bayesian posterior covariance matrix for the parameters is
\[ V_\beta = (X'WX + S_\lambda)^{-1} \sigma^2 \]
The posterior distribution for \( \beta \) is thus
\[ \beta | y \sim N(\hat{\beta}, V_\beta) \]
For a particular point whose design row is vector \( x \), the prediction is \( x\hat{\beta} \) and the standard error is \( \sqrt{xV_\beta x'} \). The Bayesian posterior confidence limits are thus
\[ \left( x\hat{\beta} \pm z_{\alpha/2} \sqrt{xV_\beta x'} \right) \]
where \( z_{\alpha/2} \) is the \( 1 - \alpha/2 \) quantile of the standard normal distribution.
For the \( j \)th spline term, the prediction for the component is formed by \( x_j\hat{\beta} \), where \( x_j \) is a row vector of zeros except for columns that correspond to basis expansions of the \( j \)th spline term. And the standard error for the component is \( \sqrt{x_jV_\beta x_j'} \).

**Dispersion Parameter**

Some distribution families (Gaussian, gamma, inverse Gaussian, negative binomial, and Tweedie) have a dispersion parameter that you can specify in the DISPERSION= option in the MODEL statement or that you can estimate from the data. The following three suboptions for the SCALE= option in the MODEL statement correspond to three ways to estimate the dispersion parameter:
DEVIANCE estimates the dispersion parameter by the deviance, given the regression parameter estimates:
\[ \hat{\phi} = \frac{\sum_i D_i(y_i, \mu_i)}{n - df} \]

MLE estimates the dispersion parameter by maximizing the penalized likelihood, given the regression parameter estimates:
\[ \hat{\phi} = \arg\max_{\phi} \ell_p(\hat{\beta}, \phi) \]
The MLE option is the only option that you can use to estimate the dispersion parameter for the negative binomial and Tweedie distributions.

PEARSON estimates the dispersion parameter by Pearson’s statistic, given the regression parameter estimates:
\[ \hat{\phi} = \frac{\sum_i \omega_i(y_i - \mu_i)^2/v_i}{n - df} \]

If the dispersion parameter is estimated, it contributes one additional degree of freedom to the fitted model.

**Tests for Smoothing Components**

The GAMMOD procedure performs a smoothing component test on the null hypothesis \( f_j = 0 \) for the \( j \)th component. In contrast to the analysis of deviance that is used in the GAM procedure (which tests existence of nonlinearity for each smoothing component), the smoothing component test used in PROC GAMMOD tests for the existence of a contribution for each smoothing component.

The hypothesis test is based on the Wald statistic. Define \( X_j \) as the matrix of all zeros except for columns that correspond to basis expansions of the \( j \)th spline term. Then the column vector of predictions is \( \hat{f}_j = X_j\hat{\beta} \), and the covariance matrix for the predictions is \( V_j = X_jV_{\beta}X_j' \). The Wald statistic for testing is
\[ T_r = \hat{f}_j'V_j^{-1}\hat{f}_j = \hat{\beta}'X_j'(X_jV_{\beta}X_j)'^{-1}X_j\hat{\beta} \]
where \( V_j^{-1} \) is the rank-\( r \) pseudo-inverse of \( V_j \). If \( R_j \) is the Cholesky root for \( X_j'X_j \) such that \( R_j'\hat{R}_j = X_j'X_j \), then the test statistic can be written as
\[ T_r = \hat{\beta}'R_j'(R_jV_{\beta}R_j)'^{-1}R_j\hat{\beta} \]

Wood (2012) proposes using the \( df_j \) degrees of freedom for test (which is defined in the section “Degrees of Freedom” on page 337) as the rank \( r \). Because spline terms in fitted models often have noninteger degrees of freedom, the GAMMOD procedure uses a rounded value of \( df_j \) as the rank:
\[ r = \begin{cases} \lfloor df_j \rfloor & \text{if } df_j - \lfloor df_j \rfloor \leq 0.05 \text{ or } df_j < 1 \\ \lfloor df_j \rfloor & \text{otherwise} \end{cases} \]

Let \( K \) be a symmetric and nonnegative definite matrix, and let its eigenvalues be sorted as \( d_1 > d_2 > \cdots \); then the rank-\( r \) pseudo-inverse of \( K \) is formed by
\[ K^{r-} = U_k \begin{bmatrix} d_1^{-1} & \cdots & \cdot & d_r^{-1} \\ \cdot & \ddots & \cdot & \cdot \\ \cdot & \cdots & \ddots & \cdot \\ \cdot & \cdots & \cdots & d_r^{-1} \end{bmatrix} U_k^T \]

where \( U_k \) are formed by columns of eigenvectors that correspond to the \( r \) eigenvalues.

Under the null hypothesis, the Wald statistic \( T_r \) approximately follows the chi-square distribution \( T_r \sim \chi^2_r \).

For an observed test statistic \( t_r \), the \( p \)-value for rejecting the null hypothesis is computed as \( P(\chi^2_r > t_r) \) if the dispersion parameter is constant, or \( P(F_{r,df_r} > t_r) \) with \( df_r \) error degrees of freedom if the dispersion parameter is estimated.

Be cautious when you interpret the results of the smoothing component test because \( p \)-values are computed by approximation and the test does not take the smoothing parameter selection process into account.

### Multithreading

The GAMMOD procedure allocates data to different threads and calculates the likelihood function, gradient, and Hessian by accumulating the values from all threads. For more information about how PROC GAMMOD uses threads, see the section “Multithreading” on page 81 in Chapter 3, “Shared Concepts.”

### Optimization Algorithms

Several optimization techniques are available in PROC GAMMOD. You can choose a particular optimizer by using the TECHNIQUE= option in the PROC GAMMOD statement. Table 6.13 summarizes the optimization techniques available in PROC GAMMOD.

<table>
<thead>
<tr>
<th>TECHNIQUE=</th>
<th>Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRUREG</td>
<td>Trust region method</td>
</tr>
<tr>
<td>NEWRAP</td>
<td>Newton-Raphson method with line search</td>
</tr>
<tr>
<td>NRRIDG</td>
<td>Newton-Raphson method with ridging</td>
</tr>
<tr>
<td>QUANEW</td>
<td>Quasi-Newton methods</td>
</tr>
<tr>
<td>DBLDOG</td>
<td>Double-dogleg method</td>
</tr>
<tr>
<td>CONGRA</td>
<td>Conjugate gradient methods</td>
</tr>
<tr>
<td>NMSIMP</td>
<td>Nelder-Mead simplex method</td>
</tr>
</tbody>
</table>

There is no algorithm for optimizing general nonlinear functions that always finds the global optimum for a general nonlinear optimization problem in a reasonable amount of time. Because no single optimization technique is always superior to others, PROC GAMMOD provides a variety of optimization techniques that work well in various circumstances. However, you can devise problems for which none of the techniques in PROC GAMMOD can find the correct solution. Moreover, nonlinear optimization can be computationally expensive in terms of time and memory, so you must be careful when matching an algorithm to a problem. The section “Choosing an Optimization Algorithm” on page 82 in Chapter 3, “Shared Concepts,” is helpful in choosing a suitable optimization algorithm.
Displayed Output

The following sections describe the output that the GAMMOD procedure produces by default. The output is organized into various tables, which are discussed in the order of their appearance.

Model Information

The “Model Information” table displays basic information about the model, such as the response variable, frequency variable, link function, and model category that the GAMMOD procedure determines based on your input and options. The “Model Information” table also displays the distribution of the data that PROC GAMMOD assumes. For information about the supported response distributions, see the section “DISTRIBUTION=keyword” on page 326.

Number of Observations

The “Number of Observations” table displays the number of observations that are read from the input data table and the number of observations that are used in the analysis. If a FREQ statement is present, the sum of the frequencies read and used is displayed. If the events/trials syntax is used, the number of events and trials is also displayed.

Response Profile

The “Response Profile” table displays the ordered values from which the GAMMOD procedure determines the probability that is modeled as an event in binary models. For each response category level, the frequency that is used in the analysis is reported. You can affect the ordering of the response values by specifying response-options in the MODEL statement. For binary models, the note that follows the “Response Profile” table indicates which outcome is modeled as the event in binary models and which value serves as the reference category.

The “Response Profile” table is not produced for binomial (events/trials) data. You can find information about the number of events and trials in the “Number of Observations” table.

Class Level Information

The “Class Level Information” table lists the levels of every variable that is specified in the CLASS statement. You should check this information to make sure that the data are correct. You can adjust the order of the CLASS variable levels by specifying the ORDER= option in the CLASS statement. You can suppress the “Class Level Information” table completely or partially by specifying the NOCLPRINT= option in the PROC GAMMOD statement.

If the classification variables use reference parameterization, the “Class Level Information” table also displays the reference value for each variable.

Specifications for Spline(spline-variables)

The “Specifications for Spline(spline-variables)” table displays basic information (such as the number of variables, specified degrees of freedom, search range for the smoothing parameter, and so on) about how to construct a spline term that the GAMMOD procedure uses to construct basis expansions and search for the smoothing parameter. PROC GAMMOD generates the “Specifications for Spline(spline-variables)” table only when you specify the DETAILS option for a spline term.
Optimization Iteration History

For each iteration of the optimization, the “Iteration History” table displays the number of function evaluations (including gradient and Hessian evaluations), the value of the objective function, the change in the objective function from the previous iteration, and the absolute value of the largest (projected) gradient element. The objective function that PROC GAMMOD uses in the optimization is normalized by default to enable comparisons across data tables that have different sampling intensity.

If you specify the ITDETAILS option in the PROC GAMMOD statement, information about the parameter estimates and gradients in the course of the optimization is added to the “Iteration History” table.

For a parametric generalized linear model or for a generalized additive model that has fixed smoothing parameters, the “Iteration History” table displays information about regression parameter estimates and gradients. For a generalized additive model that has unknown smoothing parameters, the “Iteration History” table displays information about smoothing parameter estimates and gradients. If the performance iteration method is used, a column of performance iteration steps is added to the table.

Convergence Status

The convergence status table is a small ODS table that follows the “Iteration History” table in the default output. In the listing, this table appears as a message that indicates whether the optimization succeeded and which convergence criterion was met. If the optimization fails, the message indicates the reason for the failure. If you save the convergence status table to an output data table, a numeric Status variable is added that enables you to programmatically assess convergence. The values of the Status variable encode the following:

0 Convergence was achieved, or an optimization was not performed because TECHNIQUE=NONE was specified.
1 The objective function could not be improved.
2 Convergence was not achieved because of a user interrupt or because a limit (such as the maximum number of iterations or the maximum number of function evaluations) was reached. To modify these limits, see the MAXITER=, MAXFUNC=, and MAXTIME= options in the PROC GAMMOD statement.
3 Optimization failed to converge because function or derivative evaluations failed at the starting values or during the iterations or because a feasible point that satisfies the parameter constraints could not be found in the parameter space.

Fit Statistics

The “Fit Statistics” table displays a variety of likelihood-based measures of fit in addition to the model roughness measurement. All information criteria are presented in “smaller is better” form.

The calculation of the information criteria uses the following formulas, where \( \text{df} \) denotes the model degrees of freedom, \( f \) denotes the number of frequencies used, and \( \ell \) is the log likelihood that is evaluated at the converged estimates:

\[
\begin{align*}
\text{AIC} & = -2\ell + 2\text{df} \\
\text{AICC} & = \begin{cases} 
-2\ell + 2\text{df} f/(f - \text{df} - 1) & \text{when } f > \text{df} + 2 \\
-2\ell + 2\text{df}(\text{df} + 2) & \text{otherwise}
\end{cases} \\
\text{BIC} & = -2\ell + \text{df} \log(f)
\end{align*}
\]

If no FREQ statement is specified, then \( f \) equals \( n \), the number of observations that are used.
**Parameter Estimates**

The “Parameter Estimates” table displays the regression parameter estimates, their estimated (asymptotic) standard errors, chi-square statistics, and $p$-values for the hypothesis that the parameter is 0.

**Estimates for Smoothing Components**

The “Estimates for Smoothing Components” table displays a summary of the fitted spline terms, including effective degrees of freedom, smoothing parameters, roughness penalty values, number of parameters, rank of penalty matrix, and number of knots.

**Tests for Smoothing Components**

The “Tests for Smoothing Components” table displays effective degrees of freedom, effective degrees of freedom for test, $F/\chi^2$ statistics, and $p$-values for rejecting the hypothesis that the smoothing component has zero contribution.

**Timing**

The “Timing” table displays the amount of time (in seconds) that PROC GAMMOD required to perform different tasks in the analysis.

**OutputCasTables Table**

The OutputCasTables table is a special table that has information about each CAS table that is created during a CAS action execution. The information for each CAS table consists of the CAS table name, the caslib in which the table resides, and the number of columns and rows in the CAS table. Because this table is not a typical ODS table that contains analytical results, you cannot include it in the `table-spec-list` in the DISPLAYOUT statement.

---

**ODS Table Names**

Each table that the GAMMOD procedure creates has a name that is associated with it, and you must use this name to refer to the table when you use ODS statements. These names are listed in Table 6.14.

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClassInfo</td>
<td>Level information from the CLASS statement</td>
<td>CLASS</td>
<td></td>
</tr>
<tr>
<td>ConvergenceStatus</td>
<td>Status of optimization at conclusion of optimization</td>
<td>Default output</td>
<td></td>
</tr>
<tr>
<td>FitStatistics</td>
<td>Fit statistics</td>
<td>Default output</td>
<td></td>
</tr>
<tr>
<td>LikelihoodHist</td>
<td>Iteration history for maximum likelihood estimation or penalized likelihood estimation</td>
<td>Default output</td>
<td></td>
</tr>
<tr>
<td>ModelInfo</td>
<td>Information about the modeling environment</td>
<td>Default output</td>
<td></td>
</tr>
<tr>
<td>Table Name</td>
<td>Description</td>
<td>Statement</td>
<td>Option</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td>NObs</td>
<td>Number of observations read and used, and number of events and trials, if applicable</td>
<td>Default output</td>
<td></td>
</tr>
<tr>
<td>OutputCasTables</td>
<td>See the section “OutputCasTables Table” on page 343</td>
<td>OUTPUT</td>
<td>OUT=</td>
</tr>
<tr>
<td>ParameterEstimates</td>
<td>Solutions for the parameter estimates that are associated with effects in MODEL statements</td>
<td>Default output</td>
<td></td>
</tr>
<tr>
<td>ResponseProfile</td>
<td>Response categories and the category that is modeled in models for binary and multinomial data</td>
<td>Default output</td>
<td></td>
</tr>
<tr>
<td>SmoothingEstimates</td>
<td>Information for spline terms after model fitting</td>
<td>Default output</td>
<td></td>
</tr>
<tr>
<td>SmoothingHist</td>
<td>Iteration history for smoothing parameter estimation</td>
<td>PROC</td>
<td>ITDETAILS</td>
</tr>
<tr>
<td>SmoothingTests</td>
<td>Smoothing components test result</td>
<td>Default output</td>
<td></td>
</tr>
<tr>
<td>SplineDetails</td>
<td>Information about spline construction and smoothing parameter search</td>
<td>MODEL</td>
<td>SPLINE(</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/ DETAILS)</td>
</tr>
<tr>
<td>Timing</td>
<td>Absolute and relative times for tasks performed by the procedure</td>
<td>Default output</td>
<td></td>
</tr>
</tbody>
</table>

**ODS Graphics**

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in the “Statistical Graphics Using ODS” chapter in SAS/STAT User’s Guide.

Before you create graphs, ODS Graphics must be enabled (for example, with the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section “Enabling and Disabling ODS Graphics” in that chapter.

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section “A Primer on ODS Statistical Graphics” in that chapter.

When ODS Graphics is enabled, the GAMMOD procedure by default produces plots of the partial predictions for each spline term in the model. Use the PLOTS option in the PROC GAMMOD statement to control aspects of these plots.

PROC GAMMOD assigns a name to each graph that it creates by using ODS. You can use these names to refer to the graphs when using ODS. The names are listed in Table 6.15.
Table 6.15  Graphs Produced by PROC GAMMOD

<table>
<thead>
<tr>
<th>ODS Graph Name</th>
<th>Plot Description</th>
<th>PLOTS= Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>SmoothingComponentPanel</td>
<td>Panel of multiple prediction curves</td>
<td>Default COMPONENTS</td>
</tr>
<tr>
<td>SmoothingComponentPlot</td>
<td>Unpacked prediction curves</td>
<td>PLOTS(UNPACK) COMPONENTS(UNPACK)</td>
</tr>
</tbody>
</table>

By default, prediction plots for each spline component are displayed in panels that contain at most six plots. If you specify more than six smoothing components, multiple panels are used. Use the PLOTS(UNPACK) option in the PROC GAMMOD statement to display these plots individually.

Examples: GAMMOD Procedure

Example 6.1: Scatter Plot Smoothing

This example shows how you can use PROC GAMMOD to perform scatter plot smoothing.

The example uses the LIDAR data set (Ruppert, Wand, and Carroll 2003). This data set is used in many books and journals to illustrate different smoothing techniques. Scientists use a technique known as LIDAR (light detection and ranging), which uses laser reflections to detect chemical compounds in the atmosphere. The following DATA step creates the data table Lidar. This DATA step assumes that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

```sas
   title 'Scatter Plot Smoothing';
   data mycas.Lidar;
      input Range LogRatio @@;
   datalines;
   390 -0.05035573 391 -0.06009706 393 -0.04190091 394 -0.0509847
   396 -0.05991345 397 -0.02842392 399 -0.05958421 400 -0.03988881
   402 -0.02939582 403 -0.03949445 405 -0.04764749 406 -0.06038
   408 -0.03123034 409 -0.03816584 411 -0.07562269 412 -0.05001751
   414 -0.0457295 415 -0.07766966 417 -0.02460641 418 -0.07133184
   ... more lines ...
   702 -0.4716702 703 -0.7801088 705 -0.6668431 706 -0.5783479
   708 -0.7874522 709 -0.6156956 711 -0.8967602 712 -0.7077379
   714 -0.672567 715 -0.6218413 717 -0.8657611 718 -0.557754
   720 -0.8026684 ;
```

In this data table, Range records the distance that light travels before it is reflected back to the source. LogRatio is the logarithm of the ratio of light that is received from two laser sources. The objective is to use scatter plot smoothing to discover the nonlinear pattern in the data. SAS provides different methods (for
example, local regression) for scatter plot smoothing. You can perform scatter plot smoothing by using the SGPLOT procedure, as shown in the following statements:

```sas
proc sgplot data=mycas.Lidar;
  scatter x=Range y=LogRatio;
  loess x=Range y=LogRatio / nomarkers;
  pbspline x=Range y=LogRatio / nomarkers;
run;
```

Output 6.1.1 shows the scatter plot of Range and LogRatio and the smoothing curves that are fitted by the local regression and penalized B-splines smoothing techniques.

**Output 6.1.1** Scatter Plot Smoothing

Both scatter plot smoothing techniques show a significant nonlinear structure between Range and LogRatio that cannot be easily modeled by ordinary polynomials. You can also use the GAMMOD procedure to perform scatter plot smoothing on these data, as in the following statements:

```sas
proc gammod data=mycas.Lidar seed=12345;
  model LogRatio = spline(Range/details);
  output out=mycas.LidarOut pred=p;
run;
```
Example 6.1: Scatter Plot Smoothing

The “Specifications for Spline(Range)” table in Output 6.1.2 displays the specifications for constructing the spline term for Range. The maximum degrees of freedom is 10, which sets the upper limit of effective degrees of freedom for the spline term to be 9 after one degree of freedom is absorbed in the intercept. The order of the derivative in the penalty is 2, which means that the unpenalized portion of the spline term involves polynomials with degrees up to 2.

**Output 6.1.2 Spline Specification**

**Scatter Plot Smoothing**

**The GAMMOD Procedure**

<table>
<thead>
<tr>
<th>Specifications for Spline(Range)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Variables</td>
</tr>
<tr>
<td>Rank of Penalty Approximation</td>
</tr>
<tr>
<td>Order of Derivative in the Penalty</td>
</tr>
<tr>
<td>Maximum Number of Knots</td>
</tr>
</tbody>
</table>

The “Fit Statistics” table in Output 6.1.3 shows the summary statistics for the fitted model.

**Output 6.1.3 Fit Statistics**

<table>
<thead>
<tr>
<th>Fit Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Penalized Log Likelihood</td>
</tr>
<tr>
<td>Roughness Penalty</td>
</tr>
<tr>
<td>Effective Degrees of Freedom</td>
</tr>
<tr>
<td>Effective Degrees of Freedom for Error</td>
</tr>
<tr>
<td>AIC (smaller is better)</td>
</tr>
<tr>
<td>AICC (smaller is better)</td>
</tr>
<tr>
<td>BIC (smaller is better)</td>
</tr>
<tr>
<td>GCV (smaller is better)</td>
</tr>
</tbody>
</table>

The “Estimates for Smoothing Components” table in Output 6.1.4 shows that the effective degrees of freedom for the spline term of Range is approximately 8 after the GCV criterion is optimized with respect to the smoothing parameter. The roughness penalty is small, suggesting that there is an important contribution from the penalized part of thin-plate regression splines beyond nonpenalized polynomials.

**Output 6.1.4 Estimates for Smoothing Components**

<table>
<thead>
<tr>
<th>Estimates for Smoothing Components</th>
</tr>
</thead>
<tbody>
<tr>
<td>Component</td>
</tr>
<tr>
<td>----------</td>
</tr>
<tr>
<td>Spline(Range)</td>
</tr>
</tbody>
</table>

Because the optimal model is obtained by searching in a functional space that is constrained by the maximum degrees of freedom for a spline term, you might wonder whether PROC GAMMOD produces a much different model if you increase the value. The following statements fit another model in which the maximum degrees of freedom is increased to 20:
Chapter 6: The GAMMOD Procedure

```
proc gammod data=mycas.Lidar seed=12345;
  model LogRatio = spline(Range/maxdf=20);
  output out=mycas.LidarOut2 pred=p2;
run;
```

Output 6.1.5 displays fit summary statistics for the second model. The model fit statistics from the second model are very close to the ones from the first model, indicating that the second model is not much different from the first model.

```
Output 6.1.5  Fit Statistics

Scatter Plot Smoothing

The GAMMOD Procedure

<table>
<thead>
<tr>
<th>Fit Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Penalized Log Likelihood: 250.96488</td>
</tr>
<tr>
<td>Roughness Penalty: 0.05315</td>
</tr>
<tr>
<td>Effective Degrees of Freedom: 10.06143</td>
</tr>
<tr>
<td>Effective Degrees of Freedom for Error: 209.02182</td>
</tr>
<tr>
<td>AIC (smaller is better): -481.86006</td>
</tr>
<tr>
<td>AICC (smaller is better): -480.79981</td>
</tr>
<tr>
<td>BIC (smaller is better): -447.66969</td>
</tr>
<tr>
<td>GCV (smaller is better): 0.00657</td>
</tr>
</tbody>
</table>
```

Output 6.1.6 shows that the effective degrees of freedom for the spline term of Range is slightly larger than 8, which is understandable because increasing the maximum degrees of freedom expands the functional space for model searching. Functions in the expanded space can provide a better fit to the data, but they are also penalized more because the roughness penalty value for the second model is much larger than the one for the first model. This suggests that functions in the expanded space do not help much, given the nonlinear relationship between Range and LogRatio.

```
Output 6.1.6  Estimates for Smoothing Components

<table>
<thead>
<tr>
<th>Component</th>
<th>Effective DF</th>
<th>Smoothing Parameter</th>
<th>Roughness Penalty</th>
<th>Number of Parameters</th>
<th>Rank of Penalty Matrix</th>
<th>Number of Knots</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spline(Range)</td>
<td>8.06143</td>
<td>23032.9</td>
<td>0.0532</td>
<td>19</td>
<td>20</td>
<td>221</td>
</tr>
</tbody>
</table>
```

The two fitted models are both based on thin-plate regression splines, in which polynomials that have degrees higher than 2 are penalized. You might wonder whether allowing higher-order polynomials yields a much different model. The following statements fit a third spline model by penalizing polynomials that have degrees higher than 3:

```
proc gammod data=mycas.Lidar seed=12345;
  model LogRatio = spline(Range/m=3);
  output out=mycas.LidarOut3 pred=p3;
run;
```
The fit summary statistics shown in Output 6.1.7 are close to the ones from the previous two models, albeit slightly smaller.

Output 6.1.7  Fit Statistics

Scatter Plot Smoothing

The GAMMOD Procedure

<table>
<thead>
<tr>
<th>Fit Statistics</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Penalized Log Likelihood</td>
<td>249.79779</td>
</tr>
<tr>
<td>Roughness Penalty</td>
<td>9.440383E-9</td>
</tr>
<tr>
<td>Effective Degrees of Freedom</td>
<td>10.00000</td>
</tr>
<tr>
<td>Effective Degrees of Freedom for Error</td>
<td>211.00000</td>
</tr>
<tr>
<td>AIC (smaller is better)</td>
<td>-479.59559</td>
</tr>
<tr>
<td>AICC (smaller is better)</td>
<td>-478.54797</td>
</tr>
<tr>
<td>BIC (smaller is better)</td>
<td>-445.61397</td>
</tr>
<tr>
<td>GCV (smaller is better)</td>
<td>0.00664</td>
</tr>
</tbody>
</table>

As shown in Output 6.1.8, the effective degrees of freedom for the spline term where polynomials with degrees less than 4 are allowed without penalization is 8. The roughness penalty is quite small compared to the previous two fits. This also suggests that there are important contributions from the penalized part of the thin-plate regression splines even after the nonpenalized polynomials are raised to order 3.

Output 6.1.8  Estimates for Smoothing Components

<table>
<thead>
<tr>
<th>Component</th>
<th>Effective DF</th>
<th>Smoothing Parameter</th>
<th>Roughness Penalty</th>
<th>Number of Parameters</th>
<th>Rank of Penalty Matrix</th>
<th>Number of Knots</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spline(Range)</td>
<td>8.00000</td>
<td>1.0000</td>
<td>9.44E-9</td>
<td>9</td>
<td>10</td>
<td>221</td>
</tr>
</tbody>
</table>

The following statements use the DATA step to merge the predictions from the three scatter plot smoothing fits by PROC GAMMOD and use the SGPLOT procedure to visualize them:

```plaintext
data LidarPred;
    merge mycas.Lidar mycas.LidarOut mycas.LidarOut2 mycas.LidarOut3;
run;
proc sort data=LidarPred; by Range;run;
proc sgplot data=LidarPred;
    scatter x=Range y=LogRatio / markerattrs=GraphData1(size=7);
    series x=Range y=p / lineattrs =GraphData2(thickness=2)
                   legendlabel="Spline 1";
    series x=Range y=p2 / lineattrs =GraphData3(thickness=2)
                    legendlabel="Spline 2";
    series x=Range y=p3 / lineattrs =GraphData4(thickness=2)
                    legendlabel="Spline 3";
run;
```

Output 6.1.9 displays the scatter plot smoothing fits by PROC GAMMOD under three different spline specifications.
Example 6.2: Nonparametric Logistic Regression

This example shows how you can use PROC GAMMOD to build a nonparametric logistic regression model for a data set that contains a binary response and then use that model to classify observations.

The example uses the Pima Indians Diabetes data set, which can be obtained from the UCI Machine Learning Repository (Lichman 2013). It is extracted from a larger database that was originally owned by the National Institute of Diabetes and Digestive and Kidney Diseases. Data are for female patients who are at least 21 years old, are of Akimel O’otham (Pima Indian) heritage, and live near Phoenix, Arizona. The objective of this study is to investigate the relationship between a diabetes diagnosis and variables that represent physiological measurements and medical attributes. Some missing or invalid observations are removed from the analysis. The reduced data set contains 532 records. The following DATA step creates the data set DiabetesStudy:

```plaintext
title 'Diabetes Study';
data DiabetesStudy;
  input NPreg Glucose Pressure Triceps BMI Pedigree Age Diabetes Test@@;
datalines;
  6  148  72  35  33.6  0.627  50  1  1  1  85  66  29  26.6  0.351  31  0  1
  1  89  66  23  28.1  0.167  21  0  0  3  78  50  32  31  0.248  26  1  0
```

![Output 6.1.9 Scatter Plot Smoothing](image)
Example 6.2: Nonparametric Logistic Regression

The data set contains nine variables, including the binary response variable Diabetes. Table 6.16 describes the variables.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NPreg</td>
<td>Number of pregnancies</td>
</tr>
<tr>
<td>Glucose</td>
<td>Two-hour plasma glucose concentration in an oral glucose tolerance test</td>
</tr>
<tr>
<td>Pressure</td>
<td>Diastolic blood pressure (mm Hg)</td>
</tr>
<tr>
<td>Triceps</td>
<td>Triceps skin fold thickness (mm)</td>
</tr>
<tr>
<td>BMI</td>
<td>Body mass index (weight in kg/(height in m)^2)</td>
</tr>
<tr>
<td>Pedigree</td>
<td>Diabetes pedigree function</td>
</tr>
<tr>
<td>Age</td>
<td>Age (years)</td>
</tr>
<tr>
<td>Diabetes</td>
<td>0 if test negative for diabetes, 1 if test positive</td>
</tr>
<tr>
<td>Test</td>
<td>0 for training role, 1 for test</td>
</tr>
</tbody>
</table>

The Test variable splits the data set into training and test subsets. The training observations (whose Test value is 0) hold approximately 59.4% of the data. To build a model that is based on the training data and evaluate its performance by predicting the test data, you use the following statements to create a new variable, Result, whose value is the same as that of the Diabetes variable for a training observation and is missing for a test observation:

```sas
data mycas.DiabetesStudy;
  set DiabetesStudy;
  Result = Diabetes;
  if Test=1 then Result=.;
run;
```

A typical starting point of your analysis for modeling the binary response might be a parametric logistic regression model. But it is restricted in the sense that all variables affect the response in strictly linear fashion. If you are uncertain that a variable is an important factor and its contribution is linear in predicting the response, you might want to choose a nonparametric logistic regression model to fit the data. You can use PROC GAMMOD to form a nonparametric model by including the spline transformation of each explanatory variable, as shown in the following statements:

```sas
proc gammod data=mycas.DiabetesStudy seed=12345;
  model Result(event='1') = spline(NPreg) spline(Glucose) spline(Pressure) spline(Triceps) spline(BMI) spline(Pedigree) spline(Age) / dist=binary;
run;
```
Because the response variable Result is binary, the DIST=BINARY option in the MODEL statement specifies a binary distribution for the response variable. By default, PROC GAMMOD models the probability of the first ordered response category, which is a negative diabetes testing result in this case. The EVENT= option specifically requests that PROC GAMMOD model the probability of positive diabetes testing results. The “Response Profile” table in Output 6.2.1 shows the frequencies of the response in both categories.

Output 6.2.1  Response Profile

Diabetes Study

The GAMMOD Procedure

<table>
<thead>
<tr>
<th>Ordered Value</th>
<th>Result</th>
<th>Total Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>208</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>122</td>
</tr>
</tbody>
</table>

Probability modeled is Result = 1.

Output 6.2.2 lists the summary statistics from the nonparametric logistic regression model, which include spline transformations of all variables.

Output 6.2.2  Fit Statistics

<table>
<thead>
<tr>
<th>Fit Statistics</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Penalized Log Likelihood</td>
<td>-138.57851</td>
</tr>
<tr>
<td>Roughness Penalty</td>
<td>1.208632E-8</td>
</tr>
<tr>
<td>Effective Degrees of Freedom</td>
<td>16.00000</td>
</tr>
<tr>
<td>Effective Degrees of Freedom for Error</td>
<td>313.68660</td>
</tr>
<tr>
<td>AIC (smaller is better)</td>
<td>309.15702</td>
</tr>
<tr>
<td>AICC (smaller is better)</td>
<td>310.89504</td>
</tr>
<tr>
<td>BIC (smaller is better)</td>
<td>369.94250</td>
</tr>
<tr>
<td>UBRE (smaller is better)</td>
<td>0.02357</td>
</tr>
</tbody>
</table>

The “Tests for Smoothing Components” table in Output 6.2.3 shows approximate tests results. Although some spline terms are significant, others are not. The null testing hypothesis is whether the total contribution from a variable is 0. So you can form a reduced model by removing those nonsignificant spline terms from the model. In this case, spline transformations for NPreg, Pressure, BMI, and Triceps are dropped from the model because their $p$-values are larger than the 0.1 nominal level.
Output 6.2.3 Tests for Smoothing Components

<table>
<thead>
<tr>
<th>Component</th>
<th>Effective DF</th>
<th>Effective DF for Test</th>
<th>Chi-Square</th>
<th>Pr &gt; ChiSq</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spline(NPreg)</td>
<td>1.00000</td>
<td>1</td>
<td>0.0186</td>
<td>0.8916</td>
</tr>
<tr>
<td>Spline(Glucose)</td>
<td>1.00000</td>
<td>1</td>
<td>44.5949</td>
<td>&lt;.0001</td>
</tr>
<tr>
<td>Spline(Pressure)</td>
<td>1.00000</td>
<td>1</td>
<td>2.8310</td>
<td>0.0925</td>
</tr>
<tr>
<td>Spline(Triceps)</td>
<td>1.00000</td>
<td>1</td>
<td>1.2128</td>
<td>0.2708</td>
</tr>
<tr>
<td>Spline(BMI)</td>
<td>1.00000</td>
<td>1</td>
<td>5.8112</td>
<td>0.0159</td>
</tr>
<tr>
<td>Spline(Pedigree)</td>
<td>1.00000</td>
<td>1</td>
<td>6.1108</td>
<td>0.0134</td>
</tr>
<tr>
<td>Spline(Age)</td>
<td>9</td>
<td>9</td>
<td>19.0721</td>
<td>0.0246</td>
</tr>
</tbody>
</table>

The following statements use PROC GAMMOD to fit a reduced nonparametric logistic regression model. The OUTPUT statement requests predicted probabilities for both training and test data. The COPYVARS option in the OUTPUT statement requests that the Diabetes and Test variables also be included in the output data table so that you can use them to identify test observations and compute misclassification errors.

```plaintext
ods graphics on;
proc gammod data=mycas.DiabetesStudy plots seed=12345;
  model Result(event='1') = spline(Glucose)
    spline(Pedigree) spline(Age) / dist=binary;
  output out=mycas.DiabetesStudyOut copyvars=(Diabetes Test);
run;
```

Output 6.2.4 shows the summary statistics from the reduced nonparametric logistic regression model. The values of the information criteria are better than those of the parametric logistic regression model.

Output 6.2.4 Fit Statistics

<table>
<thead>
<tr>
<th>Diabetes Study</th>
</tr>
</thead>
<tbody>
<tr>
<td>The GAMMOD Procedure</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Fit Statistics</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Penalized Log Likelihood</td>
<td>-149.85766</td>
</tr>
<tr>
<td>Roughness Penalty</td>
<td>2.85613</td>
</tr>
<tr>
<td>Effective Degrees of Freedom</td>
<td>8.05241</td>
</tr>
<tr>
<td>Effective Degrees of Freedom for Error</td>
<td>320.61183</td>
</tr>
<tr>
<td>AIC (smaller is better)</td>
<td>312.96402</td>
</tr>
<tr>
<td>AICC (smaller is better)</td>
<td>313.41826</td>
</tr>
<tr>
<td>BIC (smaller is better)</td>
<td>343.55588</td>
</tr>
<tr>
<td>UBRE (smaller is better)</td>
<td>-0.00230</td>
</tr>
</tbody>
</table>

In the “Estimates for Smoothing Components” table in Output 6.2.5, PROC GAMMOD reports that the effective degrees of freedom value for spline transformations of Glucose is quite close to 1. This suggests a strictly linear form for Glucose. For Pedigree, the degrees of freedom value demonstrates a moderate amount of nonlinearity. For Age, the degrees of freedom value is much larger than 1. The measure suggests a nonlinear pattern in the dependency of the response on Age.
The “Tests for Smoothing Components” table in Output 6.2.6 shows that all spline transformations are significant in predicting diabetes testing results.

The smoothing component panel (which is produced by the PLOTS option and is shown in Output 6.2.7) visualizes the spline transformations for the four variables in addition to 95% Bayesian curvewise confidence bands. For Glucose, the spline transformation is almost a straight line. For Pedigree, the spline transformation shows a slightly nonlinear trend. For Age, the dependency is obviously nonlinear.
The following statements compute the misclassification error on the test data from the reduced nonparametric logistic regression model that PROC GAMMOD produces:

```r
data test;
  set mycas.DiabetesStudyOut(where=(Test=1));
  if ((Pred>0.5 & Diabetes=1) | (Pred<0.5 & Diabetes=0))
    then Error=0;
  else Error=1;
run;

proc freq data=test;
  tables Diabetes*Error/nocol norow;
run;
```

Output 6.2.8 shows the misclassification errors for observations in the test set and observations of each response category.
### Output 6.2.8  Crosstabulation Table for Test Set Prediction

#### Diabetes Study

The FREQ Procedure

<table>
<thead>
<tr>
<th>Error</th>
<th>Table of Diabetes by Error</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Diabetes</td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>130</td>
</tr>
<tr>
<td></td>
<td>64.36</td>
</tr>
<tr>
<td>1</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>17.33</td>
</tr>
<tr>
<td>Total</td>
<td>165</td>
</tr>
<tr>
<td></td>
<td>81.68</td>
</tr>
</tbody>
</table>

---

### Example 6.3: Nonparametric Tweedie Regression

This example illustrates how you can use the GAMMOD procedure to fit a nonparametric regression model with spline effects for a response variable that has a Tweedie distribution.

You can view this model as a generalized linear model because the Tweedie distribution belongs to the exponential family. The Tweedie distribution is particularly useful for modeling response variables that are continuous for positive values and take the value 0 with a positive probability. For example, in the insurance industry the Tweedie distribution is often assumed for claims, which are positive for customers who have filed claims and are 0 for other customers. You can use the Tweedie distribution to model such response variables without special transformations.

The GENSELECT procedure can fit Tweedie regression models that incorporate linear effects for the covariates. However, as demonstrated in this example, the assumption of linearity can be too restrictive, and you might want to use the GAMMOD procedure to explore the nonlinear dependency structures in the data.

The following DATA step simulates a response variable $Y$ by using a Tweedie distribution and four continuous predictors. Each continuous predictor is sampled independently from the uniform distribution $U(0, 1)$. The linear predictor $\eta$ for the true model is formed by applying additive nonlinear transformations to $x_1$, $x_2$, and $x_3$:

$$
\eta = \frac{2 \sin(\pi x_1) + 0.8 \exp(2x_2) + 0.2x_3^{11}(10(1 - x_3))^6 + 10(10x_3)^3(1 - x_3)^{10}}{20}
$$

The predictor $x_4$ is a nuisance parameter that does not enter the model. The parameters for the Tweedie distribution are $\phi = 0.4$ and $p = 1.5$, and the link function is $\log \mu = \eta$, where the expected value is $E(Y) = \mu$ and the variance is $\text{Var}(Y) = \phi \mu^p$. The power parameter $p$ controls the variance of the distribution. When $p$ is between 1 and 2, as in this example, a Tweedie random variable can be generated from a compound Poisson distribution (Smyth 1996).

```sas
title 'Nonparametric Tweedie Model';
%let phi=0.4;
%let power=1.5;

data mycas.one;
  do i=1 to 1000;
    %let phi=0.4;
    %let power=1.5;
    data mycas.one;
    do i=1 to 1000;
      
```
Example 6.3: Nonparametric Tweedie Regression

/* Sample the predictors */
x1=ranuni(1);
x2=ranuni(1);
x3=ranuni(1);
x4=ranuni(1);

/* Apply nonlinear transformations to predictors */
f1=2*sin(3.14159265*x1);
f2=exp(2*x2)*0.8;
f3=0.2*x3**11*(10*(1-x3))**6+10*(10*x3)**3*(1-x3)**10;
xb=f1+f2+f3;
xb=xb/20;
mu=exp(xb);

/* Compute parameters of compound Poisson distribution */
lambda=mu**(2-\phi)/(\phi*(2-\power));
alp=(2-\power)/((\power-1));
gam=\phi*\power-1)*\phi\power-1)*mu\power-1);

/* Simulate the response */
rpoi=ranpoi(1,lambda);
if rpoi=0 then y=0;
else do;
y=0;
do j=1 to rpoi;
   y=y+rangam(1,alp);
end;
y=y*gam;
end;
output;
end;
run;

You can use PROC GENSELECT to fit a parametric model for $Y$ by using the following statements:

    proc genselect data=mycas.one;
    model y=x1 x2 x3 x4/dist=tweedie;
    run;

Equivalently, you can also use PROC GAMMOD to fit a parametric linear model by using the PARAM option in the MODEL statement, as shown in the following statements.

    proc gammod data=mycas.one seed=1234;
    model y=param(x1 x2 x3 x4)/dist=tweedie;
    run;

The “Fit Statistics” table in Output 6.3.1 shows the summary statistics for the fitted parametric Tweedie model.
Output 6.3.1 Fit Statistics

Nonparametric Tweedie Model

The GAMMOD Procedure

<table>
<thead>
<tr>
<th>Fit Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Penalized Log Likelihood</td>
</tr>
<tr>
<td>Roughness Penalty</td>
</tr>
<tr>
<td>Effective Degrees of Freedom</td>
</tr>
<tr>
<td>Effective Degrees of Freedom for Error</td>
</tr>
<tr>
<td>AIC (smaller is better)</td>
</tr>
<tr>
<td>AICC (smaller is better)</td>
</tr>
<tr>
<td>BIC (smaller is better)</td>
</tr>
</tbody>
</table>

The “Parameter Estimates” table in Output 6.3.2 shows the estimates for regression parameters in addition to dispersion and power parameters ($\phi$ and $\rho$).

Output 6.3.2 Parameter Estimates

<table>
<thead>
<tr>
<th>Parameter Estimates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter   DF  Estimate  Standard Error  Chi-Square  Pr &gt; ChiSq</td>
</tr>
<tr>
<td>Intercept   1  0.441832  0.067334  43.0565  &lt;.0001</td>
</tr>
<tr>
<td>x1          1  -0.043725  0.064965  0.4530  0.5009</td>
</tr>
<tr>
<td>x2          1  0.202242  0.064216  9.9188  0.0016</td>
</tr>
<tr>
<td>x3          1  -0.253920  0.066625 14.5250  0.0001</td>
</tr>
<tr>
<td>x4          1  -0.061098  0.066040  0.8559  0.3549</td>
</tr>
<tr>
<td>Dispersion  1  0.416001  0.016987</td>
</tr>
<tr>
<td>Power       1  1.524581  0.048675</td>
</tr>
</tbody>
</table>

If you are uncertain about the relationship between the linked mean for the fitted Tweedie model and the covariates, you can add spline terms to the model to explore possible nonlinear dependence. The following statements fit a nonparametric Tweedie model and produce a plot of fitted smoothing components.

```
proc gammod data=mycas.one seed=1234 plots;
   model y=spline(x1) spline(x2) spline(x3) spline(x4)/dist=tweedie;
run;
```

The “Fit Statistics” table in Output 6.3.3 shows the summary statistics for the fitted nonparametric Tweedie model. Compared to the parametric model, the nonparametric model has more effective degrees of freedom, but smaller AIC and AICC values. The penalized likelihood value is larger for the nonparametric model. These statistics indicate that the nonparametric model provides a better fit.
The “Regression Parameter Estimates” table in Output 6.3.4 lists the estimate for the intercept, in addition to the estimates for the dispersion and power parameters for the Tweedie model. Both variance parameters are close to the true values.

The “Estimates for Smoothing Components” table in Output 6.3.5 shows the fitted information for the spline terms in the model. The degrees of freedom value and the roughness penalty value suggest some moderate nonlinear relationship between the linked mean and the three covariates \(x_1\), \(x_2\), and \(x_3\). The spline term for the nuisance parameter \(x_4\) has a linear form.

The “Tests for Smoothing Components” table in Output 6.3.6 shows the approximate Wald test for the four spline terms. The spline term for the nuisance predictor \(x_4\) is not significant.
Output 6.3.6 Tests for Smoothing Components

<table>
<thead>
<tr>
<th>Component</th>
<th>Effective DF</th>
<th>Effective DF for Test</th>
<th>F Value</th>
<th>Pr &gt; F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spline(x1)</td>
<td>2.10969</td>
<td>3</td>
<td>4.76</td>
<td>0.0027</td>
</tr>
<tr>
<td>Spline(x2)</td>
<td>2.38340</td>
<td>3</td>
<td>13.64</td>
<td>&lt;.0001</td>
</tr>
<tr>
<td>Spline(x3)</td>
<td>5.62608</td>
<td>7</td>
<td>49.02</td>
<td>&lt;.0001</td>
</tr>
<tr>
<td>Spline(x4)</td>
<td>1.00002</td>
<td>1</td>
<td>0.88</td>
<td>0.3484</td>
</tr>
</tbody>
</table>

Output 6.3.7 displays the “Smoothing Component Panel” for all the spline terms used in the model. For \( x_1, x_2, \) and \( x_3 \), the fitted curves are reasonably smooth and close to the true functions. For \( x_4 \), the spline plot shows a strictly linear fit, with 95% Bayesian confidence bands covering the horizontal line at 0. This reinforces the conclusion that \( x_4 \) does not contribute to the model and can be removed from further analysis.

Output 6.3.7 Smoothing Component Panel
References


# Chapter 7
The GENSELECT Procedure

## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview: GENSELECT Procedure</td>
<td>364</td>
</tr>
<tr>
<td>PROC GENSELECT Features</td>
<td>364</td>
</tr>
<tr>
<td>PROC GENSELECT Compared with Other SAS Procedures</td>
<td>365</td>
</tr>
<tr>
<td>Using CAS Sessions and CAS Engine Librefs</td>
<td>366</td>
</tr>
<tr>
<td>Getting Started: GENSELECT Procedure</td>
<td>367</td>
</tr>
<tr>
<td>Poisson Regression for Count Data</td>
<td>367</td>
</tr>
<tr>
<td>Syntax: GENSELECT Procedure</td>
<td>372</td>
</tr>
<tr>
<td>PROC GENSELECT Statement</td>
<td>372</td>
</tr>
<tr>
<td>BY Statement</td>
<td>375</td>
</tr>
<tr>
<td>CLASS Statement</td>
<td>375</td>
</tr>
<tr>
<td>CODE Statement</td>
<td>376</td>
</tr>
<tr>
<td>DISPLAY Statement</td>
<td>376</td>
</tr>
<tr>
<td>DISPLAYOUT Statement</td>
<td>377</td>
</tr>
<tr>
<td>EFFECT Statement</td>
<td>378</td>
</tr>
<tr>
<td>FREQ Statement</td>
<td>379</td>
</tr>
<tr>
<td>MODEL Statement</td>
<td>380</td>
</tr>
<tr>
<td>OUTPUT Statement</td>
<td>386</td>
</tr>
<tr>
<td>PARTITION Statement</td>
<td>391</td>
</tr>
<tr>
<td>SELECTION Statement</td>
<td>392</td>
</tr>
<tr>
<td>WEIGHT Statement</td>
<td>393</td>
</tr>
<tr>
<td>Details: GENSELECT Procedure</td>
<td>394</td>
</tr>
<tr>
<td>Missing Values</td>
<td>394</td>
</tr>
<tr>
<td>Exponential Family Distributions</td>
<td>394</td>
</tr>
<tr>
<td>Response Distributions</td>
<td>395</td>
</tr>
<tr>
<td>Response Probability Distribution Functions</td>
<td>396</td>
</tr>
<tr>
<td>Log-Likelihood Functions</td>
<td>401</td>
</tr>
<tr>
<td>Existence of Maximum Likelihood Estimates</td>
<td>405</td>
</tr>
<tr>
<td>The LASSO Method of Model Selection</td>
<td>406</td>
</tr>
<tr>
<td>Model Fit and Assessment Statistics</td>
<td>407</td>
</tr>
<tr>
<td>Predicted Values and Regression Diagnostics</td>
<td>408</td>
</tr>
<tr>
<td>Joint Tests and Type 3 Tests</td>
<td>411</td>
</tr>
<tr>
<td>Multithreading</td>
<td>412</td>
</tr>
<tr>
<td>Optimization Algorithms</td>
<td>412</td>
</tr>
<tr>
<td>Displayed Output</td>
<td>413</td>
</tr>
<tr>
<td>ODS Table Names</td>
<td>416</td>
</tr>
<tr>
<td>ODS Graphics</td>
<td>417</td>
</tr>
</tbody>
</table>
Overview: GENSELECT Procedure

The GENSELECT procedure fits and performs model selection for generalized linear models in SAS Viya. The models that PROC GENSELECT supports can contain main effects that consist of both continuous and classification variables and interaction effects of these variables. The models can also include constructed effects such as splines. The procedure offers a number of effect-selection methods, including stepwise methods and modern LASSO methods. It also offers extensive capabilities for customizing the model selection by using a wide variety of selection and stopping criteria, from computationally efficient significance-level-based criteria to modern, computationally intensive validation-based criteria. PROC GENSELECT also provides a variety of diagnostics that are conditional on the selected model.

PROC GENSELECT fits models for standard distributions in the exponential family, such as the normal, Poisson, binomial, gamma, and Tweedie distributions. In addition, the procedure fits multinomial models for ordinal and nominal response data, and it fits models for the beta, generalized Poisson, and negative binomial distributions.

PROC GENSELECT Features

The GENSELECT procedure does the following:

- estimates the parameters of a generalized linear regression model by using maximum likelihood techniques
- provides model-building syntax in the CLASS statement and the effect-based MODEL statement, which are familiar from SAS/STAT procedures (in particular, the GLM, GENMOD, LOGISTIC, GLIMMIX, and MIXED procedures)
- enables you to split classification effects into individual components by using the SPLIT option in the CLASS statement
- permits any degree of interaction effects that involve classification and continuous variables
- provides multiple link functions
- provides cumulative link models for ordinal response data and generalized logit modeling for unordered multinomial data
- provides response-variable options for binary and multinomial data as in the LOGISTIC procedure
- enables model building (variable selection) through the SELECTION statement
• provides a **WEIGHT** statement for weighted analysis
• provides a **FREQ** statement for grouped analysis
• provides a **CODE** statement to produce SAS code that can score a new data set
• provides an **OUTPUT** statement to produce a data table that contains predicted probabilities and other observationwise statistics
• uses ODS Graphics to create model selection plots as part of its output. For more information about ODS Graphics, see the section “ODS Graphics” on page 417.

Because the GENSELECT procedure runs on CAS, it also does the following:

• enables you to run on a cluster of machines that distribute the data and the computations
• enables you to run in single-machine mode on CAS
• exploits all the available cores and concurrent threads. For information about how PROC GENSELECT uses threads, see the section “Multithreading” on page 81 in Chapter 3, “Shared Concepts.”

---

**PROC GENSELECT Compared with Other SAS Procedures**

The GENSELECT procedure provides generalized linear modeling functionality that is comparable to that of the HPGENSELECT and GENMOD procedures in SAS/STAT software.

**PROC GENSELECT Compared with the HPGENSELECT Procedure**

The functionality of the GENSELECT procedure closely resembles that of the HPGENSELECT procedure, which is a high-performance procedure. The GENSELECT procedure is the next generation of the HPGENSELECT procedure, and it was developed specifically for SAS Viya. Both procedures are designed to run on a cluster of machines that distribute the data and the computations.

Both the GENSELECT and HPGENSELECT procedures fit and perform model selection for generalized linear models. The models can contain main effects that consist of both continuous and classification variables and interaction effects of these variables. The GENSELECT procedure is additionally capable of constructing complex effects, including univariate spline and polynomial expansions. The default parameterization of **CLASS** variables in both procedures is the GLM parameterization.

If you do not specify a link function with the **LINK=** option, the GENSELECT procedure uses the logarithm link function as the default for both the gamma and the inverse Gaussian distributions. The HPGENSELECT procedure uses the reciprocal link function as the default for the gamma distribution, and the reciprocal squared as the default for the inverse Gaussian distribution.

With the GENSELECT and HPGENSELECT procedures, you request model selection by using the **SELECTION** statement. Both procedures offer the same methods of effect selection, but the GENSELECT procedure also produces selection plots by using ODS Graphics.

Both procedures use a modification of the Newton-Raphson algorithm with a ridged Hessian by default. You can choose different optimization techniques, including first-order methods that do not require a crossproducts matrix or Hessian, by using the **TECHNIQUE=** option. The default method for the Tweedie distribution is a quasi-Newton method.
PROC GENSELECT Compared with the GENMOD Procedure

The GENMOD procedure fits generalized linear models. The GENSELECT procedure fits and performs model selection for generalized linear models. The models can contain main effects that consist of both continuous and classification variables and interaction effects of these variables. The default parameterization of CLASS variables in both procedures is the GLM parameterization.

If you do not specify a link function with the LINK= option, the GENSELECT procedure uses the logarithm link function as the default for both the gamma and the inverse Gaussian distributions. The GENMOD procedure uses the reciprocal link function as the default for the gamma distribution, and the reciprocal squared as the default for the inverse Gaussian distribution.

Both procedures use a modification of the Newton-Raphson algorithm with a ridged Hessian by default. You can instead choose Fisher scoring in PROC GENMOD.

The GENMOD procedure offers a wide variety of postfitting analyses, such as contrasts, estimates, tests of model effects, and least squares means. The GENSELECT procedure is limited in postfitting functionality because it is primarily designed for large-data tasks, such as predictive model building, model fitting, and scoring.

The GENSELECT procedure is specifically designed to operate in SAS Viya, and it performs computations in multiple threads. The GENMOD procedure executes in a single thread on a single machine.

Using CAS Sessions and CAS Engine Librefs

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:

```
proc options option=(CASHOST CASPORT);
run;
```

In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:

```
cas mysess;
  libname mycas cas sessref=mysess;
```

The CAS statement creates the CAS session named mysess, and the LIBNAME statement creates the mycas CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the CASHOST and CASPORT of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the mysess session, you can terminate it by using the TERMINATE option in the CAS statement as follows:
cas mysess terminate;

For more information about the CAS and LIBNAME statements, see the section “Introduction to Shared Concepts” on page 10 in Chapter 3, “Shared Concepts.”

---

### Getting Started: GENSELECT Procedure

#### Poisson Regression for Count Data

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

This example illustrates how you can use PROC GENSELECT to perform Poisson regression for count data. The input data must be a table on your CAS server, and a CAS session must be set up. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.” The following DATA step creates the data table `getStarted`, which consists of 100 observations on a count response variable (Y); a continuous variable (Total) to be used in a later analysis; and five categorical variables (C1–C5), each of which has four numerical levels, in your CAS session:

```sas
data mycas.getStarted;
  input C1-C5 Y Total;
  datalines;
  0 3 1 1 3 2 28.361
  2 3 0 3 1 2 39.831
  1 3 2 2 1 17.133
  1 2 0 0 3 2 12.769
  0 2 1 0 1 1 29.464
  0 2 1 0 2 1 4.152
  1 2 1 0 1 0 0.000
  0 2 1 1 2 1 20.199
  1 2 0 0 1 0 0.000
  0 1 1 3 3 2 53.376
  2 2 2 1 1 31.923
  0 3 2 0 3 2 37.987
  2 2 2 0 0 1 1.082
  0 2 0 2 0 1 6.323
  1 3 0 0 0 0 0.000
  1 2 1 2 3 2 4.217
  0 1 2 3 1 1 26.084
  1 1 0 0 1 0 0.000
  1 3 2 2 2 0 0.000
  2 1 3 1 1 2 52.640
  1 3 0 1 2 1 3.257
  2 0 2 3 0 5 88.066
  2 2 2 1 0 1 15.196
```

---
3 1 3 1 0 1 11.955
3 1 3 1 2 3 91.790
3 1 1 2 3 7 232.417
3 1 1 1 0 1 2.124
3 1 0 0 0 2 32.762
3 1 2 3 0 1 25.415
2 2 0 1 2 1 42.753
2 3 2 2 3 1 23.854
2 0 0 2 3 2 49.438
1 0 0 2 3 4 105.449
0 0 2 3 0 6 101.536
0 3 1 0 0 0 0.000
3 1 0 1 0 1 5.937
2 0 0 3 2 5 53.952
1 1 3 1 1 1 0.287
2 1 3 0 3 7 281.551
1 3 2 1 1 0 0.000
2 1 0 0 1 0 0.000
0 0 1 1 2 3 93.009
0 1 0 1 0 2 25.055
1 2 2 2 3 1 1.691
0 3 2 3 1 1 10.719
3 3 0 3 3 1 19.279
2 0 0 2 1 2 40.802
2 2 3 0 3 3 72.924
0 2 0 3 0 1 10.216
3 0 1 2 2 2 87.773
2 1 2 3 1 0 0.000
3 2 0 3 1 0 0.000
3 0 3 0 0 2 62.016
1 3 2 3 1 3 36.355
2 3 2 0 3 1 23.190
1 0 1 2 1 1 11.784
2 1 2 2 2 5 204.527
3 0 1 1 2 5 115.937
0 1 1 3 2 1 44.028
2 2 1 3 1 4 52.247
1 1 0 0 1 1 17.621
1 3 1 2 1 2 10.706
2 2 0 2 3 3 81.506
0 1 0 0 2 2 81.835
0 1 2 0 1 2 20.647
3 2 2 2 0 1 3.110
2 2 3 0 0 1 13.679
1 3 1 0 2 1 17.621
3 3 2 2 1 2 30.025
0 0 3 1 3 6 202.172
3 2 3 2 1 3 36.355
0 3 0 0 0 1 27.645
3 3 3 0 3 2 22.470
2 3 2 0 2 0 0.000
1 3 0 2 0 1 1.628
1 3 1 0 2 0 0.000
These statements assume that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

The following statements fit a log-linked Poisson model to these data by using classification effects for the variables C1–C5:

```
proc genselect data=mycas.getStarted;
   class C1-C5;
   model Y = C1-C5 / Distribution=Poisson Link=Log;
run;
```

The output from this analysis is presented in Figure 7.1 through Figure 7.7.

Figure 7.1 displays the “Model Information” table. The variable Y is an integer-valued variable that is modeled by using a Poisson probability distribution, and the mean of Y is modeled by using a log link function. The GENSELECT procedure uses a Newton-Raphson algorithm to fit the model. The CLASS variables C1–C5 are parameterized by using GLM parameterization, which is the default.

![Figure 7.1 Model Information](image-url)

**The GENSELECT Procedure**

<table>
<thead>
<tr>
<th>Model Information</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data Source</strong></td>
</tr>
<tr>
<td><strong>Response Variable</strong></td>
</tr>
<tr>
<td><strong>Distribution</strong></td>
</tr>
<tr>
<td><strong>Link Function</strong></td>
</tr>
<tr>
<td><strong>Optimization Technique</strong></td>
</tr>
</tbody>
</table>
Each of the CLASS variables C1–C5 has four unique formatted levels, which are displayed in the “Class Level Information” table in Figure 7.2.

**Figure 7.2** Class Level Information

<table>
<thead>
<tr>
<th>Class</th>
<th>Levels</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>4</td>
<td>0 1 2 3</td>
</tr>
<tr>
<td>C2</td>
<td>4</td>
<td>0 1 2 3</td>
</tr>
<tr>
<td>C3</td>
<td>4</td>
<td>0 1 2 3</td>
</tr>
<tr>
<td>C4</td>
<td>4</td>
<td>0 1 2 3</td>
</tr>
<tr>
<td>C5</td>
<td>4</td>
<td>0 1 2 3</td>
</tr>
</tbody>
</table>

Figure 7.3 displays the “Number of Observations” table. All 100 observations in the data set are used in the analysis.

**Figure 7.3** Number of Observations

| Number of Observations Read | 100 |
| Number of Observations Used  | 100 |

Figure 7.4 displays the “Dimensions” table for this model. This table summarizes some important sizes of various model components. For example, it shows that there are 21 columns in the design matrix X: 1 column for the intercept and 20 columns for the effects that are associated with the classification variables C1–C5. However, the rank of the crossproducts matrix is only 16. Because the classification variables C1–C5 use GLM parameterization and because the model contains an intercept, there is one singularity in the crossproducts matrix of the model for each classification variable. Consequently, only 16 parameters enter the optimization.

**Figure 7.4** Dimensions in Poisson Regression

<table>
<thead>
<tr>
<th>Dimensions</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Columns in Design</td>
<td>21</td>
</tr>
<tr>
<td>Number of Effects</td>
<td>6</td>
</tr>
<tr>
<td>Max Effect Columns</td>
<td>4</td>
</tr>
<tr>
<td>Rank of Design</td>
<td>16</td>
</tr>
<tr>
<td>Parameters in Optimization</td>
<td>16</td>
</tr>
</tbody>
</table>

Figure 7.5 displays the final convergence status of the Newton-Raphson algorithm. The FCONV= convergence criterion is satisfied.

**Figure 7.5** Convergence Status

Convergence criterion (FCONV=1E-7) satisfied.
The “Fit Statistics” table is shown in Figure 7.6. The –2 log likelihood at the converged estimates is 290.16169. You can use this value to compare the model to nested model alternatives by means of a likelihood ratio test. To compare models that are not nested, information criteria such as AIC (Akaike’s information criterion), AICC (Akaike’s bias-corrected information criterion), and BIC (Schwarz Bayesian information criterion) are used. These criteria penalize the –2 log likelihood for the number of parameters.

<table>
<thead>
<tr>
<th>Fit Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>-2 Log Likelihood 290.16169</td>
</tr>
<tr>
<td>AIC (smaller is better) 322.16169</td>
</tr>
<tr>
<td>AICC (smaller is better) 328.71590</td>
</tr>
<tr>
<td>SBC (smaller is better) 363.84441</td>
</tr>
</tbody>
</table>

The “Parameter Estimates” table in Figure 7.7 shows that many parameters have fairly large \( p \)-values, indicating that one or more of the model effects might not be necessary.

<table>
<thead>
<tr>
<th>Parameter Estimates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>Intercept</td>
</tr>
<tr>
<td>C1 0</td>
</tr>
<tr>
<td>C1 1</td>
</tr>
<tr>
<td>C1 2</td>
</tr>
<tr>
<td>C1 3</td>
</tr>
<tr>
<td>C2 0</td>
</tr>
<tr>
<td>C2 1</td>
</tr>
<tr>
<td>C2 2</td>
</tr>
<tr>
<td>C2 3</td>
</tr>
<tr>
<td>C3 0</td>
</tr>
<tr>
<td>C3 1</td>
</tr>
<tr>
<td>C3 2</td>
</tr>
<tr>
<td>C3 3</td>
</tr>
<tr>
<td>C4 0</td>
</tr>
<tr>
<td>C4 1</td>
</tr>
<tr>
<td>C4 2</td>
</tr>
<tr>
<td>C4 3</td>
</tr>
<tr>
<td>C5 0</td>
</tr>
<tr>
<td>C5 1</td>
</tr>
<tr>
<td>C5 2</td>
</tr>
<tr>
<td>C5 3</td>
</tr>
</tbody>
</table>
Syntax: GENSELECT Procedure

The following statements are available in the GENSELECT procedure:

```plaintext
PROC GENSELECT <options> ;
   BY variables ;
   CLASS variable <(options)> . . . variable <(options)> </global-options> ;
   CODE <options> ;
   DISPLAY <table-list> </options> ;
   DISPLAYOUT table-spec-list </options> ;
   EFFECT name=effect-type(variables </options>) ;
   FREQ variable ;
   MODEL response<(response-options)> = <effects> </model-options> ;
   MODEL events/trials<(response-options)> = <effects> </model-options> ;
   OUTPUT OUT=CAS-libref.data-table <keyword=<name>>. . .<keyword=<name>> <options> ;
   PARTITION <partition-options> ;
   SELECTION <METHOD=method<(method-options)>><options> ;
   WEIGHT variable ;
```

The PROC GENSELECT statement and at least one MODEL statement are required. The CLASS statement can appear multiple times. If a CLASS statement is specified, it must precede the MODEL statements.

PROC GENSELECT Statement

```plaintext
PROC GENSELECT <options> ;
```

The PROC GENSELECT statement invokes the procedure. Table 7.1 summarizes the available options in the PROC GENSELECT statement by function. They are then described fully in alphabetical order.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Basic Options</strong></td>
<td></td>
</tr>
<tr>
<td>ALPHA=</td>
<td>Specifies a global significance level</td>
</tr>
<tr>
<td>DATA=</td>
<td>Specifies the CAS input data table</td>
</tr>
<tr>
<td>PAGEOBS=</td>
<td>Specifies the maximum number of observations to be computed in each batch</td>
</tr>
<tr>
<td><strong>Options Related to Output</strong></td>
<td></td>
</tr>
<tr>
<td>CORRB</td>
<td>Displays the “Parameter Estimates Correlation Matrix” table</td>
</tr>
<tr>
<td>COVB</td>
<td>Displays the “Parameter Estimates Covariance Matrix” table</td>
</tr>
<tr>
<td>ITHIST</td>
<td>Displays the “Iteration History” table</td>
</tr>
<tr>
<td>NOCHECK</td>
<td>Disables checking for infinite parameters</td>
</tr>
<tr>
<td>NOSTDERR</td>
<td>Suppresses computation of the covariance matrix and standard errors</td>
</tr>
<tr>
<td>NOCLPRINT</td>
<td>Limits or suppresses the display of classification variable levels</td>
</tr>
<tr>
<td>PARTFIT</td>
<td>Displays the fit statistics that are produced when your data are partitioned</td>
</tr>
<tr>
<td>STB</td>
<td>Displays standardized estimates</td>
</tr>
</tbody>
</table>
The optimization options are fully described in the section “Optimization Options” on page 44 in Chapter 3, “Shared Concepts.” The following list describes the other options available in the PROC GENSELECT statement:

**ALPHA=**number
specifies a global significance level for the construction of confidence intervals. The confidence level is 1 - number. The value of number must be between 0 and 1. You can override this global significance level by specifying this option in the OUTPUT statement. By default, ALPHA=0.05.

**CORRB**
creates the “Parameter Estimates Correlation Matrix” table. The correlation matrix is computed by normalizing the covariance matrix \( \Sigma \). That is, if \( \sigma_{ij} \) is an element of \( \Sigma \), then the corresponding element of the correlation matrix is \( \sigma_{ij}/\sigma_i \sigma_j \), where \( \sigma_i = \sqrt{\sigma_{ii}} \).

**COVB**
creates the “Parameter Estimates Covariance Matrix” table. The covariance matrix is computed as the inverse of the negative of the matrix of second derivatives of the log-likelihood function with respect to the model parameters (the Hessian matrix).

**DATA=**CAS-libref.data-table
names the input data table for PROC GENSELECT to use. The default is the most recently created data table. *CAS-libref.data-table* is a two-level name, where
Chapter 7: The GENSELECT Procedure

**CAS-libref** refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to the data, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about **CAS-libref**, see the section “Using CAS Sessions and CAS Engine Librefs” on page 366.

**data-table** specifies the name of the input data table.

**ITHIST** generates the “Iteration History” table.

**LASSORHO=r**

specifies the base regularization parameter for the LASSO model selection method. The regularization parameter for step $i$ is $r^i$. By default, LASSORHO=0.8.

**LASSOSTEPS=n**

specifies the maximum number of steps for LASSO model selection. By default, LASSOSTEPS=20.

**LASSOTOL=r**

specifies the convergence tolerance for the optimization algorithm that solves for the LASSO parameter estimates at each step of LASSO model selection. By default, LASSOTOL=1E–6.

**NOCHECK**

disables the checking process that determines whether maximum likelihood estimates of the regression parameters exist. For more information, see the section “Existence of Maximum Likelihood Estimates” on page 405.

**NOCOMP**

suppresses the display of the “Class Level Information” table if you do not specify `number`. If you specify `number`, the values of the classification variables are displayed for only those variables whose number of levels is less than `number`. Specifying `number` helps to reduce the size of the “Class Level Information” table if some classification variables have a large number of levels.

**NOSTDERR**

suppresses computation of the covariance matrix and the standard errors of the regression coefficients. When the model contains many variables (thousands), the inversion of the Hessian matrix to derive the covariance matrix and the standard errors of the regression coefficients can be time-consuming. The CORRB, COVB, and TYPE3 options are not available when the NOSTDERR option is specified. This option also disables the quasi-complete separation check; for more information, see the section “Existence of Maximum Likelihood Estimates” on page 405.

**PAGEOBS=number | AUTO**

**MAXOPTBATCH=number | AUTO**

specifies the maximum number of observations to be included in a batch. During the optimization, the GENSELECT procedure reads at most `number` observations from the data table into memory; performs the appropriate log-likelihood, gradient, and Hessian computations on that batch of observations; then discards those observations and reads in the next batch of data for processing. Generally, a smaller `number` decreases memory usage but might lead to longer computation times, whereas a larger `number` might lead to shorter computation times but increases memory usage. The default PAGEOBS=AUTO option determines whether the entire data table can be held in a subset of your available memory; if it cannot, then `number` is set to 256.
PARTFIT displays fit statistics in the “Fit Statistics” table that are usually produced when your data are partitioned. This option is not required when you specify a PARTITION statement. The statistic that is added to the table is the average square error (or Brier score).

STB displays the standardized estimates of the parameters in the “Parameter Estimates” table. The standardized estimate of \( \hat{\beta}_i \) is given by \( \hat{\beta}_i / (s/s_i) \), where \( s_i \) is the total sample standard deviation for the \( i \)th explanatory variable and

\[
s = \begin{cases} 
\pi / \sqrt{3} & \text{LOGIT and GLOGIT links} \\
1 & \text{PROBIT link} \\
\pi / \sqrt{6} & \text{CLOGLOG and LOGLOG links} \\
1 & \text{all other links}
\end{cases}
\]

The sample standard deviations for parameters that are associated with CLASS variables are computed using their codings. The standardized estimates are not computed for the intercept parameters.

BY Statement

BY variables ;

You can specify a BY statement in PROC GENSELECT to obtain separate analyses of observations in groups that are defined by the values of the BY variables. If you specify more than one BY statement, only the last one specified is used. For more information, see the discussion of BY-group processing in SAS Language Reference: Concepts.

CLASS Statement

CLASS variable <(options)> . . . < variable <(options)>> < / global-options > ;

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. You can list the response variable for binary models in the CLASS statement, but this is not required. Table 7.2 summarizes the values that you can use for either an option or a global-option. The options are fully documented in the section “CLASS Statement” on page 12 in Chapter 3, “Shared Concepts.”

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESCENDING</td>
<td>Reverses the sort order</td>
</tr>
<tr>
<td>MISSING</td>
<td>Treats missing values as valid levels</td>
</tr>
<tr>
<td>ORDER=</td>
<td>Specifies the sort order for the levels</td>
</tr>
<tr>
<td>PARAM=</td>
<td>Specifies the parameterization of the variable</td>
</tr>
<tr>
<td>REF=</td>
<td>Specifies the reference level of the variable</td>
</tr>
<tr>
<td>SPLIT</td>
<td>Allows design columns for a variable to enter or leave the model independently</td>
</tr>
</tbody>
</table>
CODE Statement

CODE < options > ;

The CODE statement writes SAS DATA step code for computing predicted values of the fitted model to a file, to a catalog entry, or to a CAS table. To score new data, you can then include the file or the catalog entry in a DATA step, or you can specify the CAS table in the runCodeTable action in the dataStep action set (for more information, see SAS Viya: System Programming Guide).

Table 7.3 summarizes the options available in the CODE statement.

Table 7.3  CODE Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMMENT</td>
<td>Adds comments to the generated code</td>
</tr>
<tr>
<td>FILE=</td>
<td>Names the file in which to save the generated code</td>
</tr>
<tr>
<td>FORMATWIDTH=</td>
<td>Specifies the numeric format width for the regression coefficients</td>
</tr>
<tr>
<td>INDENTSIZE=</td>
<td>Specifies the number of spaces to indent the generated code</td>
</tr>
<tr>
<td>LABELID=</td>
<td>Specifies a number used to construct names and labels</td>
</tr>
<tr>
<td>LINESIZE=</td>
<td>Specifies the line size for the generated code</td>
</tr>
<tr>
<td>NOTRIM</td>
<td>Compares formatted values, including blank padding</td>
</tr>
<tr>
<td>OUT=</td>
<td>Names an output CAS table in which to save the generated code</td>
</tr>
</tbody>
</table>

For more information about the syntax of the CODE statement, see the section “CODE Statement” on page 16 in Chapter 3, “Shared Concepts.”

DISPLAY Statement

DISPLAY < table-list > < / options > ;

The DISPLAY statement enables you to specify a list of display tables to display or exclude. This statement is similar to the ODS SELECT, ODS EXCLUDE, and ODS TRACE statements. However, the DISPLAY statement can improve performance when a large number of tables could be generated (such as in BY-group processing). The procedure processes the DISPLAY statement on a CAS server and thus sends only a subset of ODS tables to the SAS client. Because ODS statements are processed on a SAS client, first all the generated display tables are sent to the client, and then the client creates a subset.

If you use both DISPLAY and ODS statements together, the DISPLAY statement takes precedence over the ODS statements. Note that the ODS EXCLUDE statement processes tables that are sent to the client after they have been filtered by the DISPLAY statement. In some cases, it might appear that the ODS EXCLUDE statement is taking precedence because it can further filter the tables. For more information about ODS, see SAS Output Delivery System: Procedures Guide.

You can specify the table-list as a list of table names, paths, partial pathnames, and regular expressions.

The table names that you can specify are listed in the section “ODS Table Names” on page 416. A path is a table name that is prefixed with dot-separated grouping information. For example, a
SelectionSummary table that a procedure produces during a selection routine might have the path `Bygroup1.Summary.SelectionSummary`. A partial pathname does not include all groups; for example, `SelectionSummary` and `Summary.SelectionSummary` are partial pathnames for `Bygroup1.Summary.SelectionSummary`.

When you specify a table name or partial pathname, all display tables whose paths end in the specified name are selected for display or exclusion. For example, both `SelectionSummary` and `Summary.SelectionSummary` select `Bygroup1.Summary.SelectionSummary`.

A regular expression is enclosed in forward slashes (/). For example, specifying “/tions/” selects all pathnames that contain the substring “tions”; in particular, the `Bygroup1.Summary.SelectionSummary` table is selected. Specifying “!/tions/” selects all pathnames that do not contain the substring “tions”; in particular, the `Bygroup1.Summary.SelectionSummary` table is not selected.

You can specify the following `options` after a slash (/):

- **CASESENSITIVE**
  - performs a case-sensitive comparison of table names in the `table-list` to display table names when tables are subsetted for display. To preserve case, you must enclose table names in the `table-list` in quotation marks.

- **EXCLUDE**
  - displays all display tables except those that you specify in the `table-list`.

- **EXCLUDEALL**
  - suppresses display of all tables. This option takes precedence over the other options.

- **TRACE**
  - displays the display table names, labels, and paths.

---

**DISPLAYOUT Statement**

```sas
DISPLAYOUT table-spec-list < / options > ;
```

The DISPLAYOUT statement enables you to create CAS output tables from your displayed output. This statement is similar to the ODS OUTPUT statement. For more information about ODS, see *SAS Output Delivery System: Procedures Guide*.

The `table-spec-list` specifies a list of CAS output tables to create. Each entry in the list has either a `key=value` format or a `key` format:

- `key=value` specifies `key` as the ODS table name, path, or partial pathname, and specifies `value` as the CAS output table name.
- `key` specifies `key` as the ODS table name and also as the CAS output table name.

The ODS table names that you can specify are listed in the section “ODS Table Names” on page 416. You cannot specify the ODS table named OutputCasTables in the `table-spec-list`.

Table names and partial pathnames are discussed under the DISPLAY statement. The DISPLAYOUT statement does not support regular expressions.

You can specify the following `options` after a slash (/):
INCLUDEALL
creates output CAS tables for all display tables. The name of the created output CAS table is the same as the corresponding display table name. If you specify this option, the table-spec-list specification is ignored.

NOREPLACE
does not replace any existing CAS output table of the same name.

REPEATED
replicates all CAS output tables on all nodes.

EFFECT Statement

EFFECT name=effect-type (variables < / options>) ;

The EFFECT statement enables you to construct special collections of columns for design matrices. These collections are referred to as constructed effects to distinguish them from the usual model effects that are formed from continuous or classification variables, as discussed in the section “GLM Parameterization of Classification Variables and Effects” on page 54 in Chapter 3, “Shared Concepts.”

You can specify the following effect-types:

COLLECTION specifies a collection effect that defines one or more variables as a single effect that has multiple degrees of freedom. The variables in a collection are considered as a unit for purposes of estimation and inference.

MULTIMEMBER | MM specifies a multimember classification effect whose levels are determined by one or more variables that appear in a CLASS statement.

POLYNOMIAL | POLY specifies a multivariate polynomial effect in the specified numeric variables.

SPLINE specifies a regression spline effect whose columns are univariate spline expansions of one or more variables. A spline expansion replaces the original variable with an expanded or larger set of new variables.

Table 7.4 summarizes the options available in the EFFECT statement.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Collection Effects Options</strong></td>
<td></td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays the constituents of the collection effect</td>
</tr>
<tr>
<td><strong>Multimember Effects Options</strong></td>
<td></td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays the levels of the multimember effect</td>
</tr>
<tr>
<td>NOEFFECT</td>
<td>Specifies that observations whose levels are all missing for the multimember variables should have 0 values in the corresponding design matrix columns</td>
</tr>
<tr>
<td>STDIZE</td>
<td>Standardizes the design matrix entries so that each observation has a sum of 1</td>
</tr>
</tbody>
</table>
Table 7.4  continued

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WEIGHT=</td>
<td>Specifies the weight variable for the contributions of each classification effect</td>
</tr>
</tbody>
</table>

**Polynomial Effects Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEGREE=</td>
<td>Specifies the degree of the polynomial</td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays details of the specified polynomial</td>
</tr>
<tr>
<td>MDEGREE=</td>
<td>Specifies the maximum degree of any variable in a term of the polynomial</td>
</tr>
<tr>
<td>NOSEPARATE</td>
<td>Treats the polynomial as a single effect with multiple degrees of freedom</td>
</tr>
<tr>
<td>STANDARDIZE=</td>
<td>Specifies centering and scaling suboptions for the variables that define the polynomial</td>
</tr>
</tbody>
</table>

**Spline Effects Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BASIS=</td>
<td>Specifies the type of basis (B-spline basis or truncated power function basis) for the spline effect</td>
</tr>
<tr>
<td>DATABOUNDARY</td>
<td>Uses the extremes of the data as boundary knots for a B-spline basis</td>
</tr>
<tr>
<td>DEGREE=</td>
<td>Specifies the degree of the spline effect</td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays the knots and locations for each spline basis function</td>
</tr>
<tr>
<td>KNOTMAX=</td>
<td>Requests equally spaced right-side boundary knots starting at the variables’ maximum and ending at the KNOTMAX= value</td>
</tr>
<tr>
<td>KNOTMETHOD=</td>
<td>Specifies how to construct the knots for the spline effect</td>
</tr>
<tr>
<td>KNOTMIN=</td>
<td>Requests equally spaced left-side boundary knots starting at the KNOTMIN= value and ending at the variables’ minimum value</td>
</tr>
<tr>
<td>NATURALCUBIC</td>
<td>Specifies a natural cubic spline basis for the spline effect</td>
</tr>
<tr>
<td>SEPARATE</td>
<td>Treats the spline basis for each variable as a separate effect when multiple variables are specified</td>
</tr>
<tr>
<td>SPLIT</td>
<td>Treats each design matrix column as a separate effect for selection methods</td>
</tr>
</tbody>
</table>

For more information about the syntax of these effect-types and how columns of constructed effects are computed, see the section “EFFECT Statement” on page 21 in Chapter 3, “Shared Concepts.”

**FREQ Statement**

FREQ variable ;

The variable in the FREQ statement identifies a numeric variable in the data set that contains the frequency of occurrence of each observation. PROC GENSELECT treats each observation as if it appears \( f \) times, where \( f \) is the value of the FREQ variable for the observation. If \( f \) is not an integer, it is truncated to an integer. If \( f \) is less than 1 or missing, the observation is not used in the analysis. When the FREQ statement is not specified, each observation is assigned a frequency of 1.
MODEL Statement

MODEL response < (response-options) > = < effects > < / model-options > ;

MODEL events / trials < (response-options) > = < effects > < / model-options > ;

The MODEL statement defines the statistical model in terms of a response variable (the target) or an events/trials specification, model effects that are constructed from variables in the input data table, and model-options. An intercept is included in the model by default. You can remove the intercept by specifying the NOINT option.

You can specify a single response variable that contains your response values. When you have binomial data, you can specify the events/trials form of the response, where one variable contains the number of positive responses (or events) and another variable contains the number of trials. Note that the values of both events and (trials – events) must be nonnegative and the value of trials must be positive.

For information about constructing the model effects, see the section “Specification and Parameterization of Model Effects” on page 51 in Chapter 3, “Shared Concepts.”

There are two sets of options in the MODEL statement. The response-options determine how the GENSELECT procedure models probabilities for binary and multinomial data. The model-options control other aspects of model formation and inference. Table 7.5 summarizes these options.

Table 7.5  MODEL Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Response Variable Options</strong></td>
<td></td>
</tr>
<tr>
<td>DESCENDING</td>
<td>Reverses the response categories</td>
</tr>
<tr>
<td>EVENT=</td>
<td>Specifies the event category</td>
</tr>
<tr>
<td>ORDER=</td>
<td>Specifies the sort order</td>
</tr>
<tr>
<td>REF=</td>
<td>Specifies the reference category</td>
</tr>
<tr>
<td><strong>Model Options</strong></td>
<td></td>
</tr>
<tr>
<td>CENTER</td>
<td>Centers and scales continuous main effects</td>
</tr>
<tr>
<td>CENTERLASSO</td>
<td>Centers and scales all effects for model selection by the LASSO method</td>
</tr>
<tr>
<td>CLB</td>
<td>Requests confidence limits</td>
</tr>
<tr>
<td>INCLUDE=</td>
<td>Includes effects in all models for model selection</td>
</tr>
<tr>
<td>INFORMATIVE</td>
<td>Models missing values by using extra indicator variables</td>
</tr>
<tr>
<td>LINK=</td>
<td>Specifies the link function</td>
</tr>
<tr>
<td>NOINT</td>
<td>Suppresses the intercept</td>
</tr>
<tr>
<td>OFFSET=</td>
<td>Specifies the offset variable</td>
</tr>
<tr>
<td>PHI=</td>
<td>Specifies a fixed dispersion parameter</td>
</tr>
<tr>
<td>START=</td>
<td>Includes effects in the initial model for model selection</td>
</tr>
<tr>
<td>TYPE3</td>
<td>Displays the Type 3 or joint tests of effects</td>
</tr>
</tbody>
</table>
Response Variable Options

Response variable options determine how the GENSELECT procedure models probabilities for binary and multinomial data.

You can specify the following `response-options` by enclosing them in parentheses after the `response` or `trials` variable.

**DESCENDING**

`DESC`

reverses the order of the response categories. If you specify both the DESCENDING and `ORDER=` options, PROC GENSELECT orders the response categories according to the `ORDER=` option and then reverses that order.

**EVENT=’category’ | FIRST | LAST**

specifies the event category for the binary and multinomial response model. PROC GENSELECT models the probability of the event category. The `EVENT=` option has no effect when there are more than two response categories.

You can specify one of the following:

- `'category'`
  
specifies the value (formatted, if a format is applied) of the event category in quotation marks.

- `FIRST`
  
designates the first ordered category as the event.

- `LAST`
  
designates the last ordered category as the event.

By default, EVENT=FIRST.

For example, the following statements specify that observations whose formatted value is 1 represent events in the data. The probability that PROC GENSELECT models is thus the probability that the variable `def` takes the (formatted) value 1.

```plaintext
proc genselect data=mycas.MyData;
  class A B C;
  model def(event = '1') = A B C x1 x2 x3;
run;
```

**ORDER=FORMATTED | FREQ | INTERNAL**

specifies the sort order for the levels of the `response` variable. You can specify the following values:

- `FORMATTED`
  
sorts the levels by external formatted value, except for numeric variables that have no explicit format, which are sorted by their unformatted (internal) value. For numeric variables for which you have supplied no explicit format (that is, for which there is no corresponding FORMAT statement in the current PROC GENSELECT run or in the DATA step that created the data table), the levels are ordered by their internal (numeric) value. The sort order is machine-dependent.
FREQ sorts the levels by descending frequency count (levels that have the most observations come first in the order).

INTERNAL sorts the levels by unformatted value. The sort order is machine-dependent.

By default, ORDER=FORMATTED.

For more information about sort order, see the chapter on the SORT procedure in the Base SAS Procedures Guide and the discussion of BY-group processing in SAS Language Reference: Concepts.

REF=’category’ | FIRST | LAST
specifies the reference category for the generalized logit model and the binary response model. For the generalized logit model, each logit contrasts a nonreference category with the reference category. For the binary response model, specifying one response category as the reference is the same as specifying the other response category as the event. You can specify one of the following:

’category’
specifies the value (formatted, if a format is applied) of the reference category in quotation marks.

FIRST designates the first ordered category as the reference.

LAST designates the last ordered category as the reference.

By default, REF=LAST.

Model Options
You can specify the following model-options after a slash (/):

CENTER requests that continuous main effects be centered and scaled internally. (Continuous main effects are centered and scaled to aid in computing maximum likelihood estimates.) Parameter estimates and related statistics are always reported on the original scale.

CENTERLASSO requests that all effects, including categorical effects, be centered and scaled internally. (Effects are centered and scaled to aid in model selection by the LASSO method.) Parameter estimates and related statistics are always reported on the original scale.

CLB constructs confidence limits for each parameter estimate. The confidence level is 0.95 by default; you can change it by specifying the ALPHA= option.

DISTRIBUTION=keyword specifies the response distribution for the model. The keywords and the associated distributions are shown in Table 7.6. For information about default and commonly used link functions for each distribution function, see Table 7.8.
### Table 7.6  Built-In Distribution Functions

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Distribution Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>BETA</td>
<td>Beta</td>
</tr>
<tr>
<td>BINARY</td>
<td>Binary</td>
</tr>
<tr>
<td>BINOMIAL</td>
<td>Binary or binomial</td>
</tr>
<tr>
<td>EXPONENTIAL</td>
<td>Exponential</td>
</tr>
<tr>
<td>GAMMA</td>
<td>Gamma</td>
</tr>
<tr>
<td>GENPOISSON</td>
<td>GPOISSON</td>
</tr>
<tr>
<td>GEOMETRIC</td>
<td>Geometric</td>
</tr>
<tr>
<td>IGAUSSIAN</td>
<td>IG</td>
</tr>
<tr>
<td>MULTINOMIAL</td>
<td>Multinomial. For ordinal responses, it fits a model that has a cumulative link function. For nominal responses, it fits a model that has a generalized logit link function.</td>
</tr>
<tr>
<td>NEGATIVEBINOMIAL</td>
<td>NB</td>
</tr>
<tr>
<td>NORMAL</td>
<td>GAUSSIAN</td>
</tr>
<tr>
<td>POISSON</td>
<td>Poisson</td>
</tr>
<tr>
<td>T&lt;(\nu)&gt;</td>
<td>(t) with (v) degrees of freedom. If you do not specify (v), a value of (v = 3) is used.</td>
</tr>
<tr>
<td>TWEEDIE&lt; (Tweedie-options)&gt;</td>
<td>Tweedie</td>
</tr>
<tr>
<td>WEIBULL</td>
<td>Two-parameter Weibull</td>
</tr>
</tbody>
</table>

When DISTRIBUTION=TWEEDIE, you can specify the following **Tweedie-options**:

- **EQL**
  - uses extended quasi-likelihood instead of Tweedie log likelihood in parameter estimation.

- **INITIALP=value**
  - specifies a starting value for iterative estimation of the Tweedie power parameter.

- **OPTMETHOD=Tweedie-optimization-option**
  - requests an optimization method for iterative estimation of the Tweedie model parameters. You can specify the following **Tweedie-optimization-options**:

  - **EQL**
    - uses extended quasi-likelihood for a sample of the data, followed by extended quasi-likelihood for the full data. This is equivalent to the EQL **Tweedie-option**.

  - **EQLLHOOD**
    - uses extended quasi-likelihood for a sample of the data, followed by Tweedie log likelihood for the full data. This is the default method.

  - **FINALLHOOD**
    - uses a four-stage approach to estimating the Tweedie model parameters. The four stages are as follows:
1. extended quasi-likelihood for a sample of the data
2. Tweedie log likelihood for a sample of the data
3. extended quasi-likelihood for the full data
4. Tweedie log likelihood for the full data

**LHOOD**

uses Tweedie log likelihood be for a sample of the data, followed by Tweedie log likelihood for the full data.

**P=value**

specifies a value to use as a fixed Tweedie power parameter.

**SAMPLEFRAC=value**

specifies a value to use as the fraction of the data that are used to compute starting values for the Tweedie distribution. The value must be between 0 and 1.

**INCLUDE=n**

**INCLUDE=single-effect**

**INCLUDE=(effect-list)**

forces effects to be included in all models. If you specify INCLUDE=n, then the first n effects that are listed in the MODEL statement are included in all models. If you specify INCLUDE=single-effect or if you specify a list of effects within parentheses, then the specified effects are forced into all models. The effects that you specify in the INCLUDE= option must be explanatory effects that are specified in the MODEL statement before the slash (/).

**INFORMATIVE**

models missing values by using extra model effects. These effects consist of dummy variables that take the value 1 when the value of a continuous model variable involved in the effect is missing, and take the value 0 otherwise. The missing value in the original model effect is replaced by the average value of the effect for the nonmissing values. For continuous-by-class effects, such as A*x, where A is a classification variable and x is a continuous variable, informative missingness creates multiple dummy columns and substitutes the effect mean of x that corresponds to the respective level of A. Missing values for classification variables are treated as valid levels. For more information about informative missingness, see the section “Informative Missingness” on page 78 in Chapter 3, “Shared Concepts.”

**LINK=keyword**

specifies the link function for the model. The keywords and their associated link functions are shown in Table 7.7. Default and commonly used link functions for the available distributions are shown in Table 7.8.
Table 7.7 Built-In Link Functions

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Link Function</th>
<th>( g(\mu) = \eta = )</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLOGLOG</td>
<td>Complementary log-log or cumulative complementary log-log</td>
<td>( \log(- \log(1 - \mu)) )</td>
</tr>
<tr>
<td>GLOGIT</td>
<td>Generalized logit</td>
<td>( \mu )</td>
</tr>
<tr>
<td>IDENTITY</td>
<td>Identity</td>
<td>( \frac{1}{\mu} )</td>
</tr>
<tr>
<td>INVERSE</td>
<td>Reciprocal square</td>
<td>( \frac{1}{\mu^2} )</td>
</tr>
<tr>
<td>POWERMINUS2</td>
<td>Power</td>
<td>( \mu^p )</td>
</tr>
<tr>
<td>POWER(p)</td>
<td>Logarithm</td>
<td>( \log(\mu) )</td>
</tr>
<tr>
<td>LOGIT</td>
<td>Logit or cumulative logit</td>
<td>( \log(\mu/(1 - \mu)) )</td>
</tr>
<tr>
<td>LOGLOG</td>
<td>Log-log or cumulative log-log</td>
<td>( - \log(- \log(\mu)) )</td>
</tr>
<tr>
<td>PROBIT</td>
<td>Probit or cumulative probit</td>
<td>( \Phi^{-1}(\mu) )</td>
</tr>
</tbody>
</table>

For the probit and cumulative probit links, \( \Phi^{-1}(\cdot) \) denotes the quantile function of the standard normal distribution.

If you do not specify the LINK= option, a default link function is used, as shown in Table 7.8. For binary or multinomial distributions, only the link functions shown in Table 7.8 are available. For the other distributions, you can use any link function shown in Table 7.7 by specifying the LINK= option. Other commonly used link functions for each distribution are shown in Table 7.8.

Table 7.8 Default and Commonly Used Link Functions

<table>
<thead>
<tr>
<th>Value of the DISTRIBUTION= Option</th>
<th>Default Link Function</th>
<th>Other Commonly Used Link Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>BETA</td>
<td>Logit</td>
<td>Probit, complementary log-log, log-log</td>
</tr>
<tr>
<td>BINARY</td>
<td>Logit</td>
<td>Probit, complementary log-log, log-log</td>
</tr>
<tr>
<td>BINOMIAL</td>
<td>Logit</td>
<td>Probit, complementary log-log, log-log</td>
</tr>
<tr>
<td>EXPONENTIAL</td>
<td>Log</td>
<td>Probit, complementary log-log, log-log</td>
</tr>
<tr>
<td>GAMMA</td>
<td>Log</td>
<td>Probit, complementary log-log, log-log</td>
</tr>
<tr>
<td>GENPOISSON</td>
<td>GPOISSON</td>
<td>Log</td>
</tr>
<tr>
<td>GEOMETRIC</td>
<td>Log</td>
<td></td>
</tr>
<tr>
<td>INVERSEGAUSSIAN</td>
<td>IG</td>
<td>Log</td>
</tr>
<tr>
<td>MULTINOMIAL</td>
<td>Cumulative logit</td>
<td></td>
</tr>
<tr>
<td>NEGATIVEBINOMIAL</td>
<td>NB</td>
<td>Log</td>
</tr>
<tr>
<td>NORMAL</td>
<td>GAUSSIAN</td>
<td>Identity</td>
</tr>
<tr>
<td>POISSON</td>
<td>Log</td>
<td>Log</td>
</tr>
<tr>
<td>T</td>
<td>Identity</td>
<td></td>
</tr>
<tr>
<td>TWEEDIE</td>
<td>Log</td>
<td></td>
</tr>
<tr>
<td>WEIBULL</td>
<td>Log</td>
<td></td>
</tr>
</tbody>
</table>
Chapter 7: The GENSELECT Procedure

NOINT
requests that no intercept be included in the model. An intercept is included by default. The NOINT option is not available for multinomial models.

OFFSET=\texttt{variable}
specifies a \texttt{variable} to be used as an offset to the linear predictor. An offset plays the role of an effect whose coefficient is known to be 1. The offset variable cannot appear in the \texttt{CLASS} statement or elsewhere in the \texttt{MODEL} statement. Observations that have missing values for the offset variable are excluded from the analysis.

\texttt{PHI=number}
specifies a fixed dispersion parameter for those distributions that have a dispersion parameter. The dispersion parameter that is used in all computations is fixed at \texttt{number} and not estimated.

\texttt{START=n}
\texttt{START=single-effect}
\texttt{START=(effects)}
begin the selection process from the designated initial model for the forward selection method. If you specify \texttt{START=n}, then the starting model includes the first \texttt{n} effects that are listed in the \texttt{MODEL} statement. If you specify \texttt{START=single-effect} or \texttt{START=(effects)}, then the starting model includes those specified effects. The effects that you specify in the \texttt{START=} option must be explanatory effects that are specified in the \texttt{MODEL} statement before the slash (/). This option is not available when you specify \texttt{METHOD=BACKWARD} in the \texttt{SELECTION} statement.

\texttt{TYPE3}
requests that Wald statistics for Type 3 contrasts be computed for each effect that is specified in the \texttt{MODEL} statement. For more information, see the section “Joint Tests and Type 3 Tests” on page 411.

\texttt{OUTPUT Statement}

\texttt{OUTPUT OUT=\texttt{CAS-libref.data-table} < \texttt{ALL} > < \texttt{ALPHA=}number> < \texttt{COPYVARS=}variables> > < \texttt{keyword=}name> . . . < \texttt{keyword=}name> ;}

The \texttt{OUTPUT} statement creates a data table that contains observationwise statistics that \texttt{PROC GENSELECT} computes after fitting the model. In order to avoid data duplication for large data tables, the variables in the input data table are not included in the output data table unless you specify them in the \texttt{COPYVAR=} option.

If the response variable has more than two categories, you can request the “Statistic Options” listed in Table 7.9; the other diagnostic statistics are not available. These statistics are computed for every response category, and the automatic variable \texttt{_LEVEL_} identifies the response category on which the computed values are based. That is, every observation generates several rows in the output data set. If you also specify the \texttt{OBSCAT} option, then the observationwise statistics are computed only for the observed response category, which is indicated by the value of the \texttt{_LEVEL_} variable.

The output statistics are computed based on the final parameter estimates. If the optimization does not converge, then the output data table is not created.
For observations in which only the response variable is missing, values of the linear predictor and the predicted values are computed even though these observations do not affect the model fit. This enables, for example, predicted values to be computed for new observations.

You must specify the following option:

```
OUT=CASE-libref.data-table
```

names the output data table for PROC GENSELECT to use. You must specify this option before any other options. `CASE-libref.data-table` is a two-level name, where

- `CASE-libref` refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to where the data table is to be stored, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about `CASE-libref`, see the section “Using CAS Sessions and CAS Engine Librefs” on page 366.
- `data-table` specifies the name of the output data table.

You can also specify the following syntax elements:

**ALL**

**ALLSTAT**

adds all available statistics to the output data table.

**ALPHA=number**

specifies the significance level for the construction of confidence intervals in the output data table. The confidence level is $1 - \text{number}$. The value of `number` must be between 0 and 1. By default, `number` is equal to the value of the `ALPHA=` option in the PROC GENSELECT statement, or 0.05 if that option is not specified.

**COPYVAR=variable**

**COPYVARS=(variables)**

transfers one or more variables from the input data table to the output data table.

**OBSCAT**

requests (for multinomial models) that observationwise statistics be produced only for the observed response level. If you do not specify this option and the response variable has $J$ levels, then the following outputs are created: for cumulative link models, $J - 1$ records are output for every observation in the input data that corresponds to the $J - 1$ lower-ordered response categories; for generalized logit models, $J$ records are output that correspond to all $J$ response categories.

**keyword <=name>**

specifies a statistic to include in the output data table and optionally names the variable `name`. If you do not provide a `name`, the GENSELECT procedure assigns a default name based on the type of statistic requested.

Table 7.9 summarizes the `keywords` available in the OUTPUT statement.
## Table 7.9 OUTPUT Statement Keywords

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
<th>Default Names</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Statistic Options</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INDIVIDUAL</td>
<td>Specifies the individual predicted probabilities</td>
<td><em>IPRED</em></td>
</tr>
<tr>
<td>LCL</td>
<td>Specifies the lower confidence limit for the linear predictor</td>
<td><em>LCL</em></td>
</tr>
<tr>
<td>LCLM</td>
<td>Specifies the lower confidence limit for the event probability</td>
<td><em>LCLM</em></td>
</tr>
<tr>
<td>PREDICTED</td>
<td>Specifies the predicted probabilities</td>
<td><em>PRED</em></td>
</tr>
<tr>
<td>STDXBETA</td>
<td>Specifies the standard error estimate of the linear predictor</td>
<td><em>STDXBETA</em></td>
</tr>
<tr>
<td>UCL</td>
<td>Specifies the upper confidence limit for the linear predictor</td>
<td><em>UCL</em></td>
</tr>
<tr>
<td>UCLM</td>
<td>Specifies the upper confidence limit for the event probability</td>
<td><em>UCLM</em></td>
</tr>
<tr>
<td>XBETA</td>
<td>Specifies the linear predictor</td>
<td><em>XBETA</em></td>
</tr>
<tr>
<td><strong>Diagnostic Options</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CBAR</td>
<td>Specifies the confidence interval displacement</td>
<td><em>CBAR</em></td>
</tr>
<tr>
<td>DIFCHISQ</td>
<td>Specifies the deletion chi-square goodness-of-fit change</td>
<td><em>DIFCHISQUARE</em></td>
</tr>
<tr>
<td>DIFDEV</td>
<td>Specifies the deletion deviance change</td>
<td><em>DIFDEVIANCE</em></td>
</tr>
<tr>
<td>H</td>
<td>Specifies the leverage</td>
<td><em>HATDIAG</em></td>
</tr>
<tr>
<td>RESCHI</td>
<td>Specifies the Pearson chi-square residual</td>
<td><em>RESCHI</em></td>
</tr>
<tr>
<td>RESDEV</td>
<td>Specifies the deviance residual</td>
<td><em>RESDEV</em></td>
</tr>
<tr>
<td>RESLIK</td>
<td>Specifies the likelihood residual</td>
<td><em>RESLIK</em></td>
</tr>
<tr>
<td>RESRAW</td>
<td>Specifies the raw residual</td>
<td><em>RESRAW</em></td>
</tr>
<tr>
<td>RESWORK</td>
<td>Specifies the working residual</td>
<td><em>RESWORK</em></td>
</tr>
<tr>
<td>STDRESCHI</td>
<td>Specifies the standardized Pearson chi-square residual</td>
<td><em>STDRESCHI</em></td>
</tr>
<tr>
<td>STDRESDEV</td>
<td>Specifies the standardized deviance residual</td>
<td><em>STDRESDEV</em></td>
</tr>
<tr>
<td><strong>Miscellaneous Options for Binary and Multinomial Response Data</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTO</td>
<td>Names the level into which the observation is classified</td>
<td><em>INTO</em></td>
</tr>
<tr>
<td>LEVEL</td>
<td>Names the response level for a row of the output</td>
<td><em>LEVEL</em></td>
</tr>
</tbody>
</table>

The following list describes these *keywords*. For more information, see the section “Predicted Values and Regression Diagnostics” on page 408.

**CBAR**

specifies the confidence interval displacement diagnostic that measures the overall change in the global regression estimates that results from deleting an individual observation. The default name is _CBAR_.

**DIFCHISQ**

specifies the change in the chi-square goodness-of-fit statistic that results from deleting the individual observation. The default name is _DIFCHISQUARE_.

**DIFDEV**

specifies the change in the deviance that results from deleting the individual observation. The default name is _DIFDEVIANCE_.

**H**

specifies the diagonal element of the hat matrix (leverage) for detecting extreme points in the design space. The default name is _HATDIAG_.
INDIVIDUAL
IPRED
IPROB
IP
specifies the individual predicted values for multinomial response variables. For a response variable \( Y \) with three levels, 1, 2, and 3, the individual probabilities are \( \Pr(Y = 1) \), \( \Pr(Y = 2) \), and \( \Pr(Y = 3) \). The default name is _IPRED_.

INTO<(cutpoint)> names the variable that contains the level of the response into which an observation is classified. The default name is _INTO_. Multinomial models classify observations into the level that has the largest model-predicted probability. For binary or binomial response variables, if the predicted probability of an observation equals or exceeds the cutpoint, the observation is classified as an event; otherwise it is classified as a nonevent. You can specify the cutpoint value as a number between 0 and 1. The default value is 0.5.

LCL
LOWERXBETA
names the variable that contains the lower confidence limits for the linear predictor. The default name is _LCL_. You can set the confidence level by specifying the ALPHA= option.

LCLM
LOWERMEAN
LOWER
specifies the lower confidence limits for the mean. The default name is _LCLM_. You can set the confidence level by specifying the ALPHA= option.

LEVEL
names the variable that contains the level of the response for a given row of the output. The default name is _LEVEL_.

PREDICTED
PRED
PROB
P
specifies the predicted values for the response variable. It specifies the predicted probabilities of events for binary and nominal response variables and the cumulative predicted probabilities for ordinal response variables. For a response variable \( Y \) with three levels, 1, 2, and 3, the cumulative probabilities are \( \Pr(Y \leq 1) \) and \( \Pr(Y \leq 2) \), but by default the last level, \( \Pr(Y \leq 3) = 1 \), is not output. The default name is _PRED_.

RESCHI
PEARSON
specifies the Pearson residual for identifying poorly fitted observations. The default name is _RESCHI_.


RESDEV
specifies the deviance residual for identifying poorly fitted observations. The default name is _RESDEV_.

RESLIK
specifies the likelihood residual for identifying poorly fitted observations. The default name is _RESLIK_.

RESRAW
specifies the raw residual for identifying poorly fitted observations. The default name is _RESRAW_.

RESWORK
specifies the working residual for identifying poorly fitted observations. The default name is _RESWORK_.

ROLE
specifies the numeric variable that indicates the role played by each observation in fitting the model. The default name is _ROLE_. Table 7.10 shows how this variable is interpreted for each observation.

<table>
<thead>
<tr>
<th>Value</th>
<th>Observation Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Not used</td>
</tr>
<tr>
<td>1</td>
<td>Training</td>
</tr>
<tr>
<td>2</td>
<td>Validation</td>
</tr>
<tr>
<td>3</td>
<td>Testing</td>
</tr>
</tbody>
</table>

If you do not partition the input data by specifying a PARTITION statement, then the role variable value is 1 for observations that are used in fitting the model and 0 for observations that have at least one missing or invalid value for the response, regressor, frequency, or weight variables.

STDRESCHI
specifies the standardized Pearson (chi-square) residual for identifying observations that are poorly accounted for by the model. The default name is _STDRESCHI_.

STDRESDEV
specifies the standardized deviance residual for identifying poorly fitted observations. The default name is _STDRESDEV_.

STDXBETA
specifies the standard error estimates of XBETA. The default name is _STDXBETA_.
UCL
UPPERXBETA
specifies the variable that contains the upper confidence limits for the linear predictor. The default name is _UCL_. You can set the confidence level by specifying the ALPHA= option.

UCLM
UPPERMEAN
UPPER
specifies the variable that contains the upper confidence limits for the mean. The default name is _UCLM_. You can set the confidence level by specifying the ALPHA= option.

XBETA
LINP
specifies the linear predictor. The default name is _XBETA_.

PARTITION Statement

PARTITION partition-option;

The PARTITION statement specifies how observations in the input data set are logically partitioned into disjoint subsets for model training, validation, and testing. For more information, see the section “Using Validation and Test Data” on page 80 in Chapter 3, “Shared Concepts.” Either you can designate a variable in the input data table and a set of formatted values of that variable to determine the role of each observation, or you can specify proportions to use for randomly assigning observations to each role.

You must specify exactly one of the following partition-options:

FRACTION(<TEST=fraction> <VALIDATE=fraction> <SEED=number>)
randomly assigns specified proportions of the observations in the input data table to the roles. You specify the proportions for testing and validation by using the TEST= and VALIDATE= suboptions. If you specify both the TEST= and VALIDATE= suboptions, then the sum of the specified fractions must be less than 1 and the remaining fraction of the observations are assigned to the training role. The SEED= option specifies an integer that is used to start the pseudorandom number generator for random partitioning of data for training, testing, and validation. If you do not specify SEED=number or if number is less than or equal to 0, the seed is generated by reading the time of day from the computer’s clock.

ROLE=variable (<TEST=value> <TRAIN=value> <VALIDATE=value>)
ROLEVAR=variable (<TEST=value> <TRAIN=value> <VALIDATE=value>)
names the variable in the input data table whose values are used to assign roles to each observation. This variable cannot also appear as an analysis variable in other statements or options. The TEST=, TRAIN=, and VALIDATE= suboptions specify the formatted values of this variable that are used to assign observation roles. If you do not specify the TRAIN= suboption, then all observations whose role is not determined by the TEST= or VALIDATE= suboption are assigned to the training role.

For more information, see the section “Partition Fit Statistics” on page 407.
**SELECTION Statement**

```plaintext
SELECTION <METHOD=method < (method-options) > > <options> ;
```

The SELECTION statement performs model selection by examining whether effects should be added to or removed from the model according to rules that are defined by model selection methods. The statement is fully documented in the section “SELECTION Statement” on page 36 in Chapter 3, “Shared Concepts.”

The GENSELECT procedure supports the following effect-selection methods in the SELECTION statement:

- **BACKWARD** performs backward elimination. This method starts with all effects in the model and deletes effects.
- **BACKWARD(FAST)** performs fast backward elimination when SELECT=SL. This method starts with all effects in the model and deletes effects without refitting the model.
- **FORWARD** performs forward selection. This method starts with no effects in the model and adds effects.
- **LASSO** performs model selection by the group LASSO method. This method adds and removes effects by using a sequence of LASSO steps. For more information, see the section “Group LASSO Selection” on page 69 in Chapter 3, “Shared Concepts.”
- **NONE** results in no model selection. This method fits the full model.

By default, METHOD=STEPWISE.

The SELECT=, CHOOSE=, and STOP= method-options default to SBC. These defaults differ from their corresponding defaults in the HPGENSELECT procedure in SAS/STAT software.

You can specify the following criteria in the SELECT=, CHOOSE=, and STOP= method-options:

- **AIC** uses Akaike’s information criterion (Akaike 1974).
- **AICC** uses a small-sample bias corrected version of Akaike’s information criterion, as promoted in Hurvich and Tsai (1989) and Burnham and Anderson (1998).
- **SBC | BIC** uses the Schwarz Bayesian criterion (Schwarz 1978).
- **SL** uses the significance level of the score test as the criterion (not available for the CHOOSE= option).
- **VALIDATE** uses the average square error (ASE) that is computed on the validation data as the criterion (not available for the SELECT= option).

For more information, see the section “Information Criteria” on page 407. If you specify the PARTITION statement, then the AIC, AICC, SBC, and SL statistics are computed on the training data; otherwise they are computed on the full data table.

Forward selection where you also specify the NOINT option is available only for the following models: binary, binomial, exponential, geometric, and Poisson.

If you specify METHOD=LASSO and do not specify the CHOOSE= option, then the model in the last LASSO step is chosen as the selected model.
If you specify METHOD=LASSO, then the STOPHORIZON= option has no effect.

**NOTE:** If you use the fast backward elimination method, then the –2 log-likelihood, AIC, AICC, and SBC statistics are approximated at each step where the model is not refit, and hence they do not match the values that are computed when that model is fit outside the selection routine. Similarly, if you specify SELECT=AIC, AICC, or SBC, the selection criteria are estimated (Lawless and Singhal 1978), and hence they do not match the values that are computed when that model is fit outside the selection routine.

**NOTE:** The default model hierarchy method is HIERARCHY=None for the forward and fast backward selection methods. The backward elimination method always uses the HIERARCHY=SINGLE method-option.

The LASSO method produces a summary table that displays the effects that are added or removed at each step; the LASSO regularization parameter; and the AIC, AICC, and SBC fit statistics. For the other methods, you can specify the following values for the DETAILS= option:

- **SUMMARY** produces a summary table that shows the effect that is added or removed at each step along with the SELECT=, CHOOSE=, and STOP= criteria. The summary table is produced by default if the DETAILS= option is not specified.
- **STEPS** produces the preceding summary table and displays the results from fitting each model at each step.
- **ALL** produces the preceding tables and a detailed listing of all candidates at each step along with their ranking in terms of the selection criterion for addition to or removal from the model.

If you specify the PLOTS=CRITERIA or PLOTS=ALL option, then a plot of the fit criterion by the selection step is created for the AIC, AICC, and SBC statistics. If you also specify a PARTITION statement or the PARTFIT option, then the same type of plot is created for the ASE (for each role).

If you specify the PLOTS=FITBYROLE or PLOTS=ALL option and a PARTITION statement, then a plot of the ASE by the selection step for each role is created.

The PLOTS= option is not available for the LASSO method.

---

**WEIGHT Statement**

```
WEIGHT variable;
```

The `variable` in the WEIGHT statement is used as a weight to perform a weighted analysis of the data. Observations that have nonpositive or missing weights are not included in the analysis. If a WEIGHT statement is not included, all observations that are used in the analysis are assigned a weight of 1.
Details: GENSELECT Procedure

Missing Values

Any observation that has missing values for the response, frequency, weight, offset, or explanatory variables is excluded from the analysis; however, missing values are valid for response and explanatory variables that are specified along with the MISSING option in the CLASS statement. Observations that have a nonpositive weight or a frequency less than 1 are also excluded.

The estimated linear predictor and the fitted probabilities are not computed for any observation that has missing offset or explanatory variable values. However, if only the response value is missing, the linear predictor and the fitted probabilities can be computed and output to a data set by using the OUTPUT statement.

You can also model the missing values by specifying the INFORMATIVE option in the MODEL statement. For more information about informative missingness, see the section “Informative Missingness” on page 78 in Chapter 3, “Shared Concepts.”

Exponential Family Distributions

Many of the probability distributions that the GENSELECT procedure fits are members of an exponential family of distributions, which have probability distributions that are expressed as follows for some functions \( b \) and \( c \) that determine the specific distribution:

\[
f(y) = \exp \left\{ \frac{y \theta - b(\theta)}{\phi} + c(y, \phi) \right\}
\]

For fixed \( \phi \), this is a one-parameter exponential family of distributions. The response variable can be discrete or continuous, so \( f(y) \) represents either a probability mass function or a probability density function. A more useful parameterization of generalized linear models is by the mean and variance of the distribution:

\[
E(Y) = b'(\theta) \\
Var(Y) = b''(\theta)\phi
\]

In generalized linear models, the mean \( \mu \) of the response distribution is related to linear regression parameters through a link function,

\[
g(\mu_i) = x_i^T \beta
\]

for the \( i \)th observation, where \( x_i \) is a fixed known vector of explanatory variables and \( \beta \) is a vector of regression parameters. The GENSELECT procedure parameterizes models in terms of the regression parameters \( \beta \) and either the dispersion parameter \( \phi \) or a parameter that is related to \( \phi \), depending on the model. For exponential family models, the distribution variance is \( \text{Var}(Y) = \phi V(\mu) \), where \( V(\mu) \) is a variance function that depends only on \( \mu \).

Other distributions that are not exponential family models but are sometimes useful in statistical modeling are also included in the GENSELECT procedure.
Response Distributions

The response distribution is the probability distribution of the response (target) variable. The GENSELECT procedure can fit data for the following exponential family distributions:

- binary distribution
- binomial distribution
- gamma distribution
- geometric distribution
- exponential distribution
- inverse Gaussian distribution
- multinomial for ordinal or nominal responses
- negative binomial distribution
- normal (Gaussian) distribution
- Poisson distribution
- Tweedie distribution

The following distributions are not exponential family members but are also included in the GENSELECT procedure:

- beta distribution
- generalized Poisson distribution
- $t$ distribution
- Weibull distribution

Expressions for the probability distributions (probability density functions for continuous variables or probability mass functions for discrete variables) are shown in the section “Response Probability Distribution Functions” on page 396. The expressions for the log-likelihood functions of these distributions are presented in the section “Log-Likelihood Functions” on page 401.

The binary (or Bernoulli) distribution is the elementary distribution of a discrete random variable that can take two values, which have the probabilities $p$ and $1 - p$. Suppose the random variable is denoted as $Y$ and

\[
\Pr(Y = 1) = p \\
\Pr(Y = 0) = 1 - p
\]

The value that is associated with probability $p$ is often called the event or “success”; the complementary event is called the nonevent or “failure.” A Bernoulli experiment is a random draw from a binary distribution and generates events with probability $p$. 
If $Y_1, \ldots, Y_n$ are $n$ independent Bernoulli random variables, then their sum follows a binomial distribution. In other words, if $Y_i = 1$ denotes an event (success) in the $i$th Bernoulli trial, a binomial random variable is the number of events (successes) in $n$ independent Bernoulli trials. If you use the events/trials syntax in the MODEL statement and you specify the DISTRIBUTION=BINOMIAL option, the GENSELECT procedure fits the model as if the data had arisen from a binomial distribution. For example, the following statements fit a binomial regression model that has the regressors $x_1$ and $x_2$. The variables $e$ and $t$ represent the events and trials, respectively, for the binomial distribution:

```
proc genselect;
  model e/t = x1 x2 / distribution=Binomial;
run;
```

If the events/trials syntax is used, then both variables must be numeric and the value of the events variable cannot be less than 0 or exceed the value of the trials variable. A “Response Profile” table is not produced for binomial data, because the response variable is not subject to levelization.

If the response variable is listed in a CLASS statement and a response distribution is not specified in a DISTRIBUTION= option, then a binary distribution or a multinomial distribution that uses the default or specified link function is assumed. If the response variable is not listed in a CLASS statement and a response distribution is not specified in a DISTRIBUTION= option, then a normal distribution that uses the default or specified link function is assumed.

The multinomial distribution is a generalization of the binary distribution and allows for more than two outcome categories. Because there are more than two possible outcomes for the multinomial distribution, the terminology of “successes,” “failures,” “events,” and “nonevents” no longer applies. With multinomial data, these outcomes are generically referred to as “categories” or levels.

Whenever the GENSELECT procedure determines that the response variable is listed in a CLASS statement and has more than two levels (unless the events/trials syntax is used), it fits the model as if the data had arisen from a multinomial distribution. By default, the procedure then assumes that the response categories are ordered, and it fits a cumulative link model by applying the default or specified link function.

If the response categories are unordered, then you should fit a generalized logit model by choosing LINK=GLOGIT in the MODEL statement.

### Response Probability Distribution Functions

#### Beta Distribution

- **Density Function:**
  \[
  f(y) = \frac{\Gamma(\phi)}{\Gamma(\mu \phi) \Gamma((1 - \mu) \phi)} y^{\mu \phi - 1} (1 - y)^{(1 - \mu) \phi - 1} \quad \text{for } 0 < y < 1
  \]

- **Mean:**
  \[
  E(Y) = \mu
  \]

- **Variance:**
  \[
  \text{Var}(Y) = \frac{\mu (1 - \mu)}{(1 + \phi)}, \quad \phi > 0
  \]

This parameterization of the beta distribution comes from Ferrari and Cribari-Neto (2004).
Binary Distribution

\[ f(y) = \begin{cases} 
  p & \text{for } y = 1 \\
  1 - p & \text{for } y = 0 
\end{cases} \]
\[ E(Y) = p \]
\[ \text{Var}(Y) = p(1 - p) \]

Binomial Distribution

\[ f(y) = \binom{n}{r} \mu^r (1 - \mu)^{n-r} \text{ for } y = \frac{r}{n}, \ r = 0, 1, 2, \ldots, n \]
\[ E(Y) = \mu \]
\[ \text{Var}(Y) = \frac{\mu(1 - \mu)}{n} \]

Exponential Distribution

\[ f(y) = \frac{1}{\mu} \exp\left(-\frac{y}{\mu}\right) \text{ for } 0 < y < \infty \]
\[ E(Y) = \mu \]
\[ \text{Var}(Y) = \mu^2 \]

The exponential distribution is a special case of the gamma where \( \nu = 1 \).

Gamma Distribution

\[ f(y) = \frac{1}{\Gamma(\nu)} \left(\frac{\nu}{\mu}\right)^\nu \exp\left(-\frac{y \nu}{\mu}\right) \text{ for } 0 < y < \infty \]
\[ \phi = \frac{1}{\nu} \]
\[ E(Y) = \mu \]
\[ \text{Var}(Y) = \frac{\mu^2}{\nu}, \ \nu > 0 \]

For the gamma distribution, \( \nu = \frac{1}{\phi} \) is the estimated dispersion parameter that is displayed in the output. The parameter \( \nu \) is sometimes called the gamma index parameter.
Generalized Poisson Distribution

\[ \begin{align*}
\xi &= 1 - \exp(-\phi), \quad \phi \geq 0 \\
\mu^* &= \mu - \xi(\mu - y) \\
f(y) &= (\mu^*)^{y-1} \frac{((1 - \xi)\mu)}{\Gamma(y + 1)} \exp(-\mu^*) \quad \text{for } y = 0, 1, 2, \ldots \\
E(Y) &= \mu \\
\text{Var}(Y) &= \frac{\mu}{(1 - \xi)^2}
\end{align*} \]

The generalized Poisson distribution provides an overdispersed alternative to the Poisson distribution; \( \phi = \xi = 0 \) produces the mass function of a regular Poisson random variable. For more information about the generalized Poisson distribution and a comparison with the negative binomial distribution, see Joe and Zhu (2005). For the generalized Poisson distribution, \( \phi \) is the estimated dispersion parameter that is displayed in the output.

Geometric Distribution

\[ \begin{align*}
f(y) &= \frac{\mu^y}{(1 + \mu)^{y+1}} \quad \text{for } y = 0, 1, 2, \ldots \\
E(Y) &= \mu \\
\text{Var}(Y) &= \mu + \mu^2
\end{align*} \]

The geometric distribution is a special case of the negative binomial where \( \phi = 1 \).

Inverse Gaussian Distribution

\[ \begin{align*}
f(y) &= \frac{1}{\sqrt{2\pi} y^3 \sigma} \exp \left[ -\frac{1}{2y} \left( \frac{y - \mu}{\mu \sigma} \right)^2 \right] \quad \text{for } 0 < y < \infty \\
\phi &= \sigma^2 \\
\text{Var}(Y) &= \phi \mu^3
\end{align*} \]

Multinomial Distribution

\[ f(y_1, y_2, \ldots, y_k) = \frac{m!}{y_1!y_2!\cdots y_k!} p_1^{y_1} p_2^{y_2} \cdots p_k^{y_k} \]
**Negative Binomial Distribution**

\[
f(y) = \frac{\Gamma(y + 1/k)}{\Gamma(y + 1)\Gamma(1/k)} \frac{(k\mu)^y}{(1 + k\mu)^{y+1/k}} \quad \text{for } y = 0, 1, 2, \ldots \\
\phi = k \\
E(Y) = \mu \\
\text{Var}(Y) = \mu + \phi \mu^2
\]

For the negative binomial distribution, \( k \) is the estimated dispersion parameter that is displayed in the output.

**Normal Distribution**

\[
f(y) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left[ -\frac{1}{2} \left( \frac{y - \mu}{\sigma} \right)^2 \right] \quad \text{for } -\infty < y < \infty \\
\phi = \sigma^2 \\
E(Y) = \mu \\
\text{Var}(Y) = \phi
\]

**Poisson Distribution**

\[
f(y) = \frac{\mu^y e^{-\mu}}{y!} \quad \text{for } y = 0, 1, 2, \ldots \\
E(Y) = \mu \\
\text{Var}(Y) = \mu
\]

**(Shifted) t Distribution**

\[
z = \frac{\Gamma(0.5(v + 1))}{\sqrt{\pi v\phi} \Gamma(0.5v)} \\
f(y) = z \left( 1 + \frac{(y - \mu)^2}{v\phi} \right)^{-0.5(v+1)} \quad \text{for } -\infty < y < \infty \\
E(Y) = \mu \\
\text{Var}(Y) = \phi v/(v - 2), \quad v \geq 3
\]
You can specify the degrees of freedom parameter \(\nu\) in the `DISTRIBUTION=t(\nu)` option. If you do not specify it, a value of \(\nu = 3\) is used.

Note that this form of the \(t\) distribution is not a noncentral distribution, but that of a shifted central \(t\) random variable.

**Tweedie Distribution**

The Tweedie model is a generalized linear model from the exponential family. The Tweedie distribution is characterized by three parameters: the mean \(\mu\), the dispersion \(\phi\), and the power \(p\). The variance of the distribution is \(\phi \mu^p\). For values of \(p\) in the range \(1 < p < 2\), a Tweedie random variable can be represented as a Poisson sum of gamma distributed random variables. That is,

\[
Y = \sum_{i=1}^{N} Y_i
\]

where \(N\) has a Poisson distribution that has mean \(\lambda = \frac{\mu^{2-p}}{\phi(2-p)}\) and the \(Y_i\)s have independent, identical gamma distributions, each of which has an expected value \(E(Y_i) = \phi(2-p)\mu^{p-1}\) and an index parameter \(\nu_i = \frac{2-p}{p-1}\).

In this case, \(Y\) has a discrete mass at 0, \(Pr(Y = 0) = Pr(N = 0) = \exp(-\lambda)\), and the probability density of \(Y\), \(f(y)\), is represented by an infinite series for \(y > 0\). The GENSELECT procedure restricts the power parameter to satisfy \(1.1 \leq p\) for numerical stability in model fitting. The Tweedie distribution does not have a general closed form representation for all values of \(p\). For more information about the Tweedie distribution, see Frees (2010).

The distribution mean and variance are:

\[
E(Y) = \mu \\
Var(Y) = \phi \mu^p
\]

**Weibull Distribution**

\[
f(y) = \frac{1}{\mu \phi} \left(\frac{y}{\mu}\right)^{-1} \exp\left(-\frac{y}{\mu}\right)^{\frac{1}{\phi}} \quad \text{for } 0 < y < \infty
\]

\[
E(Y) = \mu \Gamma(1 + \phi) \\
Var(Y) = \mu^2 \left[\Gamma(1 + 2\phi) - \Gamma^2(1 + \phi)\right], \quad \phi > 0
\]
Log-Likelihood Functions

The GENSELECT procedure forms the log-likelihood functions of the various models as

\[ L(\mu; y) = \sum_{i=1}^{n} f_i \cdot l(\mu_i; y_i, w_i) \]

where \( l(\mu_i; y_i, w_i) \) is the log-likelihood contribution of the \( i \)th observation that has weight \( w_i \), and \( f_i \) is the value of the frequency variable. For the determination of \( w_i \) and \( f_i \), see the WEIGHT and FREQ statements. The individual log-likelihood contributions for the various distributions are as follows.

In the following, the mean parameter \( \mu_i \) for each observation \( i \) is related to the regression parameters \( \beta_i \) through the linear predictor \( \eta_i = x_i' \beta \) by

\[ \mu_i = g^{-1}(\eta_i) \]

where \( g \) is the link function.

**Beta Distribution**

\[
\begin{align*}
l(\mu_i; y_i, w_i) &= \log \left\{ \frac{\Gamma(\phi/w_i)}{\Gamma(\mu_i \phi/w_i) \Gamma((1-\mu_i) \phi/w_i)} \right\} \\
&\quad + (\mu_i \phi/w_i - 1) \log\{y_i\} \\
&\quad + ((1-\mu_i) \phi/w_i - 1) \log\{1 - y_i\}
\end{align*}
\]

where \( \phi \) is the dispersion parameter that is displayed in the output.

**Binary Distribution**

\[
\begin{align*}
\eta_i &= x_i' \beta \\
\mu_i &= g^{-1}(\eta_i) \\
l(\mu_i; y_i) &= y_i \log\{\mu_i\} + (1 - y_i) \log\{1 - \mu_i\}
\end{align*}
\]

Here, \( \mu_i \) is the probability of an event, and the variable \( y_i \) takes the value 1 for an event and 0 for a nonevent. The inverse link function \( g^{-1}(\cdot) \) maps from the scale of the linear predictor \( \eta_i \) to the scale of the mean. For example, for the logit link (the default),

\[ \mu_i = \frac{\exp\{\eta_i\}}{1 + \exp\{\eta_i\}} \]

You can control which binary outcome in your data is modeled as the event by specifying the response-options in the MODEL statement, and you can choose the link function by specifying the LINK= option in the MODEL statement.

If you specify a WEIGHT statement and \( w_i \) denotes the weight for the current observation, the log-likelihood function is computed as

\[ l(\mu_i; y_i, w_i) = w_i \cdot l(\mu_i; y_i) \]
Binomial Distribution

\[ \eta_i = x_i^T \beta \]
\[ \mu_i = g^{-1}(\eta_i) \]
\[ l(\mu_i; y_i, w_i) = w_i \left( y_i \log \{ \mu_i \} + (n_i - y_i) \log \{ 1 - \mu_i \} \right) + w_i \left( \log \{ \Gamma(n_i + 1) \} - \log \{ \Gamma(y_i + 1) \} - \log \{ \Gamma(n_i - y_i + 1) \} \right) \]

where \( y_i \) and \( n_i \) are the values of the events and trials of the \( i \)th observation, respectively. The value \( \mu_i \) measures the probability of events (successes) in the underlying Bernoulli distribution whose aggregate follows the binomial distribution.

Exponential Distribution

\[ l(\mu_i; y_i, w_i) = w_i \log \left\{ \frac{w_i y_i}{\mu_i} \right\} - \frac{w_i y_i}{\mu_i} - \log \{ y_i \Gamma(w_i) \} \]

Gamma Distribution

\[ \eta_i = x_i^T \beta \]
\[ \mu_i = g^{-1}(\eta_i) \]
\[ l(\mu_i; y_i, w_i) = \frac{w_i}{\phi} \log \left( \frac{w_i y_i}{\phi \mu_i} \right) - \frac{w_i y_i}{\phi \mu_i} - \log \{ y_i \} - \log \left( \Gamma \left( \frac{w_i}{\phi} \right) \right) \]

For the gamma distribution, \( \nu = \frac{1}{\phi} \) is the estimated dispersion parameter that is displayed in the output.

Generalized Poisson Distribution

\[ \xi_i = (1 - \exp\{-\phi\}/w_i) \]
\[ \mu_i^* = \mu_i - \xi_i (\mu_i - y_i) \]
\[ l(\mu_i^*; y_i, w_i) = \log \{ \mu_i^* - \xi_i y_i \} + (y_i - 1) \log \{ \mu_i^* \} \]
\[ - \mu_i^* - \log \{ \Gamma(y_i + 1) \} \]

where \( \phi \) is the dispersion parameter that is displayed in the output.

Geometric Distribution

\[ l(\mu_i; y_i, w_i) = y_i \log \left\{ \frac{\mu_i}{w_i} \right\} - (y_i + w_i) \log \left\{ 1 + \frac{\mu_i}{w_i} \right\} \]
\[ + \log \left\{ \frac{\Gamma(y_i + w_i)}{\Gamma(w_i) \Gamma(y_i + 1)} \right\} \]
### Inverse Gaussian Distribution

\[
\eta_i = x'_i \beta \\
\mu_i = g^{-1}(\eta_i) \\
l(\mu_i; y_i, w_i) = -\frac{1}{2} \left[ \frac{w_i (y_i - \mu_i)^2}{y_i \mu^2 \phi} + \log \left( \frac{\phi y_i^3}{w_i} \right) + \log(2\pi) \right]
\]

where \( \phi \) is the dispersion parameter that is displayed in the output.

### Multinomial Distribution

The multinomial distribution that the GENSELECT procedure models is a generalization of the binary distribution; it is the distribution of a single draw from a discrete distribution that has \( J \) possible values. Thus, the log-likelihood function for the \( i \)th observation is

\[
l(\pi_i; y_i, w_i) = w_i \sum_{j=1}^{J} y_{ij} \log \{ \pi_{ij} \}
\]

In this expression, \( J \) denotes the number of response categories (the number of possible outcomes) and \( \pi_{ij} \) is the probability that the \( i \)th observation takes on the response value associated with category \( j \). The category probabilities must satisfy

\[
\sum_{j=1}^{J} \pi_j = 1
\]

and the constraint is satisfied by modeling \( J - 1 \) categories.

If your response \( Y \) has more than two values and they can be ordered (for example, \( Y \in \{1, \ldots, J-1, J\} \)), then the GENSELECT procedure fits ordinal response models of the form

\[
g(\pi_j \mid x) = \alpha_j + \beta'_j x, \quad j = 1, \ldots, J - 1
\]

where \( g \) is the link function and \( \pi_j = \Pr(Y \leq j) \) are cumulative probabilities of the ordered response categories.

If your response \( Y \) has more than two values and they have no natural ordering, then the GENSELECT procedure fits a generalized or baseline-category logit model, which has the form

\[
\log \left( \frac{\Pr(Y = j \mid x)}{\Pr(Y = J \mid x)} \right) = \alpha_j + \beta'_j x, \quad j = 1, \ldots, J - 1
\]

where the \( \beta_1, \ldots, \beta_{J-1} \) are \( J - 1 \) vectors of slope parameters. These models are a special case of the discrete choice or conditional logit models introduced by McFadden (1974).

In models that have ordered response categories, the probabilities are expressed in cumulative form, so the last category is redundant. In generalized logit models (multinomial models that have unordered categories), one category is chosen as the reference category, and the linear predictor in the reference category is set to 0. For more information, see the `REF=` response-option in the MODEL statement.
Negative Binomial Distribution

\[\eta_i = x_i'\beta\]
\[\mu_i = g^{-1}(\eta_i)\]
\[l(\mu_i; y_i, w_i) = y_i \log \left( \frac{k\mu}{w_i} \right) - (y_i + w_i/k) \log \left( 1 + \frac{k\mu}{w_i} \right) + \log \left( \frac{\Gamma(y_i + w_i/k)}{\Gamma(y_i + 1)\Gamma(w_i/k)} \right)\]

where \(k\) is the negative binomial dispersion parameter that is displayed in the output.

Normal Distribution

\[\eta_i = x_i'\beta\]
\[\mu_i = g^{-1}(\eta_i)\]
\[l(\mu_i; y_i, w_i) = -\frac{1}{2} \left\{ \frac{w_i(y_i - \mu_i)^2}{\phi} + \log \left( \frac{\phi}{w_i} \right) + \log(2\pi) \right\}\]

where \(\phi\) is the dispersion parameter that is displayed in the output.

Poisson Distribution

\[\eta_i = x_i'\beta\]
\[\mu_i = g^{-1}(\eta_i)\]
\[l(\mu_i; y_i, w_i) = w_i[y_i \log(\mu_i) - \mu_i - \log(y_i!)]\]

\(t\) Distribution

\[z_i = -0.5 \log\{\phi/w_i\} + \log\{\Gamma(0.5(v + 1))\}

- \log\{\Gamma(0.5v)\} - 0.5 \times \log\{\pi v\}\]
\[l(\mu_i; y_i, w_i) = -\left( \frac{v + 1}{2} \right) \log \left\{ 1 + \frac{w_i \left( y_i - \mu_i \right)^2}{v \phi} \right\} + z_i\]

where \(\phi\) is the dispersion parameter that is displayed in the output.

Tweedie Distribution

The Tweedie distribution does not in general have a closed form log-likelihood function in terms of the mean, dispersion, and power parameters. The form of the log likelihood is

\[L(\mu; y) = \sum_{i=1}^{n} f_i l(\mu_i; y_i, w_i)\]

where

\[l(\mu_i, y_i, w_i) = \log(f(y_i; \mu_i, p, \frac{\phi}{w_i}))\]

and \(f(y, \mu, p, \phi)\) is the Tweedie probability distribution, which is described in the section “Tweedie Distribution” on page 400. Evaluation of the Tweedie log likelihood for model fitting is performed numerically as described in Dunn and Smyth (2005, 2008).
Quasi-likelihood

The extended quasi-likelihood (EQL) is constructed according to the definition of McCullagh and Nelder (1989, Chapter 9) as

$$Q_p(y, \mu, \phi, p) = \sum_i q(y_i, \mu_i, \phi, p)$$

where the contribution from an observation is

$$q(y_i, \mu_i, \phi, p) = -0.5 \log \left( 2\pi \frac{\phi}{w_i} y_i^p \right) - w_i \left( \frac{y_i^{2-p} - (2-p)y_i \mu_i^{1-p} + (1-p)\mu_i^{2-p}}{(1-p)(1-p)} \right) / \phi$$

where $1 < p < 2$. This EQL is used in computing initial values for the iterative maximization of the Tweedie log likelihood, as specified using the OPTMETHOD= Tweedie optimization-option in Table 7.6. If you specify the OPTMETHOD=EQL Tweedie-optimization-option in Table 7.6, then the parameter estimates are computed by using the EQL instead of the log likelihood.

Weibull Distribution

$$l(\mu_i; y_i, w_i) = -w_i \left\{ \frac{\phi - 1}{\phi} \log \left( \frac{y_i}{\mu_i} \right) + \log(\mu_i \phi) + \exp \left[ \log \left( \frac{y_i}{\mu_i} \right) / \phi \right] \right\}$$

where $\phi$ is the dispersion parameter that is displayed in the output.

Existence of Maximum Likelihood Estimates

The likelihood equation for a logistic regression model does not always have a finite solution. Sometimes there is a nonunique maximum on the boundary of the parameter space, at infinity. The existence, finiteness, and uniqueness of maximum likelihood estimates for the logistic regression model depend on the patterns of data points in the observation space (Albert and Anderson 1984; Santner and Duffy 1986).

Consider a binary response model. Let $Y_j$ be the response of the $j$th subject, and let $x_j$ be the vector of explanatory variables (including the constant 1 that is associated with the intercept). There are three mutually exclusive and exhaustive types of data configurations:

Complete separation

There is a complete separation of data points if there exists a vector $b$ that correctly allocates all observations to their response groups; that is,

$$\begin{cases} b'x_j > 0 & Y_j = 0 \\ b'x_j < 0 & Y_j = 1 \end{cases}$$

This configuration produces nonunique infinite estimates. If the iterative process of maximizing the likelihood function is allowed to continue, then the log likelihood diminishes to 0 and the dispersion matrix becomes unbounded.

Quasi-complete separation

The data are not completely separable, but there is a vector $b$ such that

$$\begin{cases} b'x_j \geq 0 & Y_j = 0 \\ b'x_j \leq 0 & Y_j = 1 \end{cases}$$
and equality holds for at least one subject in each response group. This configuration also yields nonunique infinite estimates. If the iterative process of maximizing the likelihood function is allowed to continue, then the dispersion matrix becomes unbounded and the log likelihood diminishes to a nonzero constant.

**Overlap** If neither complete nor quasi-complete separation exists in the sample points, there is an overlap of sample points. In this configuration, the maximum likelihood estimates exist and are unique.

The GENSELECT procedure uses a simple empirical approach to recognize the data configurations that lead to infinite parameter estimates. The basis of this approach is that any convergence method of maximizing the log likelihood must yield a solution that indicates complete separation, if such a solution exists. Upon convergence, if the predicted response equals the observed response for every observation, there is a complete separation of data points.

If the data are not completely separated, if an observation is identified to have an extremely large probability (≥ 0.95) of predicting the observed response, and if there have been at least eight iterations, then there are two possible situations. First, there is overlap in the data set, the observation is an atypical observation of its own group, and the iterative process stopped when a maximum was reached. Second, there is quasi-complete separation in the data set, and the asymptotic dispersion matrix is unbounded. If any of the diagonal elements of the dispersion matrix for the standardized observation vector (all explanatory variables standardized to zero mean and unit variance) exceeds 5,000, then PROC GENSELECT declares quasi-complete separation; if any of the diagonal elements exceeds 1,000, then the procedure displays a message indicating that quasi-complete separation might be detectable by increasing the number of iterations. If either complete separation or quasi-complete separation is detected, a note is displayed in the procedure output.

Checking for quasi-complete separation is less foolproof than checking for complete separation. If neither type of separation is discovered and your parameter estimates have large standard errors, then your data might be separable. The **NOCHECK** option in the **MODEL** statement turns off the process of checking for infinite parameter estimates; the **MINITER=** option in the **PROC GENSELECT** statement increases the number of iterations.

---

**The LASSO Method of Model Selection**

**LASSO Selection**

The GENSELECT procedure implements the group LASSO method, which is described in the section “Group LASSO Selection” on page 69 in Chapter 3, “Shared Concepts.” The current section provides some background about the LASSO method that you need in order to understand the group LASSO method.

LASSO (least absolute shrinkage and selection operator) selection arises from a constrained form of ordinary least squares regression in which the sum of the absolute values of the regression coefficients is constrained to be smaller than a specified parameter. More precisely, let \( \mathbf{X} = (x_1, x_2, \ldots, x_m) \) denote the matrix of covariates, and let \( y \) denote the response. Then for a given parameter \( t \), the LASSO regression coefficients \( \hat{\beta} = (\hat{\beta}_1, \hat{\beta}_2, \ldots, \hat{\beta}_m) \) are the solution to the constrained least squares problem:

\[
\min \| y - \mathbf{X} \hat{\beta} \|^2 \quad \text{subject to} \quad \sum_{j=1}^{m} |\hat{\beta}_j| \leq t
\]
For generalized linear models, the LASSO regression coefficients \( \beta = (\beta_1, \beta_2, \ldots, \beta_m) \) are the solution to the constrained optimization problem,

\[
\min \{-L(\mu; y)\} \quad \text{subject to} \quad \sum_{j=1}^{m} |\beta_j| \leq t
\]

where \( L \) is the log-likelihood function defined in the section “Log-Likelihood Functions” on page 401.

Provided that the LASSO parameter \( t \) is small enough, some of the regression coefficients will be exactly zero. Hence, you can think of the LASSO method as selecting a subset of the regression coefficients for each LASSO parameter. By increasing the LASSO parameter in discrete steps, you obtain a sequence of regression coefficients for which the nonzero coefficients at each step correspond to selected parameters. For more information about the LASSO method, see, for example, Hastie, Tibshirani, and Friedman (2009).

Some distributions involve a dispersion parameter (the parameter \( \phi \) in the expressions for the log likelihood). These parameters are not estimated by the LASSO optimization algorithm, and they are set to either the default value or a value that you specify. You can use the MODEL statement option PHI= to set the dispersion to a fixed value.

**Partition Fit Statistics**

Specifying a PARTITION statement modifies the display of many tables by adding separate rows or columns for the training, validation, and test data tables. In addition, the “Fit Statistics” table displays the average square error, which is useful for assessing the model and which should be very similar for the different roles when the training data are representative of the other data partitions. For more information, see the section “Model Fit and Assessment Statistics” on page 407.

**Model Fit and Assessment Statistics**

The statistics that are defined in this section are useful for assessing the fit of the model to your data; they are displayed in the “Fit Statistics” table. The statistics are computed for each data role when you specify a PARTITION statement.

**Information Criteria**

The calculation of the information criteria uses the following formulas, where \( p \) denotes the number of effective parameters in the candidate model, \( F \) denotes the sum of frequencies used, and \( l \) is the log likelihood evaluated at the converged estimates:

\[
\text{AIC} = -2l + 2p
\]

\[
\text{AICC} = \begin{cases} 
-2l + 2pF/(F - p - 1) & \text{when } F > p + 2 \\
-2l + 2p(p + 2) & \text{otherwise}
\end{cases}
\]

\[
\text{SBC} = -2l + p \log(F)
\]

If you do not specify a FREQ statement, \( F \) equals \( n \), the number of observations used.
These measures are most useful for comparing competing models that are not necessarily nested—that is, models that cannot be reduced to one another by simple constraints on the parameter space. Smaller values of the measures indicate better models.

**Average Square Error**

The average square error (ASE) is the average of the squared differences between the responses and the predictions. When you have a discrete number of response levels, the ASE is modified as shown in Table 7.11 (Brier 1950; Murphy 1973); it is also called the Brier score or Brier reliability.

<table>
<thead>
<tr>
<th>Response Type</th>
<th>ASE (Brier Score)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary</td>
<td>( \frac{1}{F} \sum_i f_i (y_i - \hat{\pi}_i)^2 + (1 - y_i) \hat{\pi}_i^2 )</td>
</tr>
<tr>
<td>Binomial</td>
<td>( \frac{1}{F} \sum_i f_i (r_i / t_i - \hat{\pi}_i)^2 )</td>
</tr>
<tr>
<td>Multinomial</td>
<td>( \frac{1}{F} \sum_i f_i \sum_j (y_{ij} - \hat{\pi}_{ij})^2 )</td>
</tr>
</tbody>
</table>

In Table 7.11, \( F = \sum_i f_i \), \( r_i \) is the number of events, \( t_i \) is the number of trials in binomial response models, \( y_i = 1 \) for events and 0 for nonevents in binary response models, and \( \hat{\pi}_i \) is the predicted probability of an event. For polytomous response models, \( y_{ij} = 1 \) if the \( i \)th observation has response level \( j \), and \( \pi_{ij} \) is the model-predicted probability of response level \( j \) for observation \( i \).

**Predicted Values and Regression Diagnostics**

You can produce observationwise predicted values, confidence limits, and regression diagnostics developed by Williams (1987) and Pregibon (1981) by specifying the OUTPUT statement. For multinomial response data, you can likewise produce observationwise predicted probabilities, confidence limits, and raw residuals.

Given a vector of covariates \( x_i \) for the \( i \)th observation in your data table, and the model-predicted parameter estimates \( \hat{\beta} \), you can write the linear predictor \( \hat{\eta}_i = x_i' \hat{\beta} \). The mean of the \( i \)th observation \( \mu_i (\hat{\beta}) \), or the model-predicted event probability \( \hat{\pi}_i \), is \( \mu_i (\hat{\beta}) = \hat{\pi}_i = g^{-1}(\hat{\eta}_i) \), where the link function \( g \) is chosen by specifying the LINK= option. The variance function of the distribution is \( V(\mu) \), and \( \Sigma \) is the estimated covariance of \( \hat{\beta} \). Denote the frequency of the \( i \)th observation as \( f_i \) and the weight as \( w_i \).

For ordinal response models, the predicted cumulative probabilities are computed by using the appropriate model-predicted intercept parameters \( \hat{\alpha}_j \) and letting \( \beta \) consist of the slope parameters: \( \hat{\eta}_{ij} = g(Pr(Y \leq j | x_i) = \hat{\alpha}_j + x_i' \hat{\beta} \) and \( \hat{\pi}_{ij} = Pr(Y \leq j | x_i) = g^{-1}(\hat{\eta}_{ij}) \) for \( 1 \leq j < J \).

For nominal response models, the predicted probabilities are computed by using the appropriate model-predicted intercept parameters \( \hat{\alpha}_j \) and letting \( \beta_j \) consist of the slope parameters: \( \hat{\eta}_{ij} = g(Pr(Y = j | x_i) = \hat{\alpha}_j + x_i' \beta_j \) and \( \hat{\pi}_{ij} = Pr(Y = j | x_i) = g^{-1}(\hat{\eta}_{ij}) \) for \( 1 \leq j < J \).

**Confidence Intervals**

Approximate confidence intervals for predicted values can be computed as follows. The variance of the linear predictor is estimated by

\[ \sigma^2 = x_i' \Sigma x_i \]
For multinomial models, the variance also depends on the response function. Let \( \delta_j \) be a \((J - 1)\) column vector whose \( j \)th entry is equal to 1 and all other entries are equal to 0. Redefine \( x_i = (\delta_j', x_i')', \eta_i = \eta_{ij} \), and \( \pi_i = \pi_{ij} \). Then
\[
\hat{\sigma}^2(\eta_i) = x_i' \Sigma x_i
\]

The asymptotic 100(1 - \( \alpha \))% confidence interval for \( \eta_i \) is
\[
\hat{\eta}_i \pm z_{a/2} \hat{\sigma}(\hat{\eta}_i)
\]
where \( z_{a/2} \) is the 100(1 - \( a/2 \))th percentile point of a standard normal distribution.

The predicted value and the 100(1 - \( a \))% confidence limits for \( p_i \) are obtained by back-transforming the corresponding measures for the linear predictor. So the confidence limits are
\[
g^{-1} \left[ \hat{\eta}_i \pm z_{a/2} \hat{\sigma}(\hat{\eta}_i) \right]
\]

**Hat-Matrix Diagonals**

The diagonal elements of the hat matrix are useful in detecting extreme points in the design space, where they tend to have larger values. For the generalized linear model, the variance of the \( i \)th individual observation is
\[
v_i = \frac{\phi V(\mu_i)}{f_i w_i}
\]
For the \( i \)th observation, let
\[
w_{ei} = v_i^{-1} (g'(\mu_i))^{-2}
\]
where \( g'(\mu_i) \) is the derivative of the link function evaluated at \( \mu_i \). Let \( W \) be the diagonal matrix, with \( w_{ei} \) denoting the \( i \)th diagonal element, which is used in computing the expected information matrix. Define the leverage, or hat-matrix diagonal, \( h_i \), as the \( i \)th diagonal element of the matrix
\[
W^{1/2} X(X'WX)^{-1} X'W^{1/2}
\]
For binary or binomial models, if the estimated probability is extreme (less than 0.1 and greater than 0.9, approximately), then the hat-matrix diagonal might be greatly reduced in value. Consequently, when an observation has a very large or very small estimated probability, its leverage is not a good indicator of the observation’s distance from the design space (Hosmer and Lemeshow 2000, p. 171).

**Residuals**

Residuals are useful in identifying observations that are not explained well by the model. For the binomial distribution, the raw residual is defined as
\[
r_i = y_i/t_i - \hat{p}_i
\]
where \( y_i \) is the number of event responses out of \( t_i \) trials for the \( i \)th observation. For single-trial syntax, \( t_i = 1 \) and \( y_i = 1 \) if the ordered response is 1 and \( y_i = 0 \) otherwise. For multinomial response data, the raw residual is
\[
y_{ij} - \hat{\pi}_{ij}
\]
where \( y_{ij} = 1 \) if the \( i \)th observation has response level \( j \) and \( y_{ij} = 0 \) otherwise, and \( \hat{\pi}_{ij} \) are the model-predicted probabilities of response level \( j \) for observation \( i \).

For other generalized linear models, the raw residual is

\[
r_i = y_i - \hat{p}_i
\]

where \( y_i \) is the observed response and \( \hat{p}_i \) is the predicted value.

The Pearson residual is the square root of the \( i \)th observation’s contribution to Pearson’s chi-square:

\[
r_{Pi} = r_i \sqrt{\frac{f_i w_i}{V(\mu_i)}}
\]

The deviance residual is the square root of the contribution of the \( i \)th observation to the deviance, with the sign of the raw residual,

\[
r_{Di} = (\text{sign}(r_i)) \sqrt{d_i}
\]

For example, for the binomial distribution,

\[
d_i = 2f_i w_i t_i \left[ \frac{y_i}{t_i} \log\left( \frac{y_i/t_i}{\mu_i} \right) + (1 - y_i/t_i) \log\left( \frac{1 - y_i/t_i}{1 - \mu_i} \right) \right]
\]

The working residual is

\[
r_{Wi} = r_i \left( \frac{\partial \mu_i}{\partial \eta_i} \right)^{-1}
\]

The Pearson residuals, standardized to have unit asymptotic variance, are

\[
r_{SPi} = \frac{r_{Pi}}{\sqrt{1 - h_i}}
\]

The deviance residuals, standardized to have unit asymptotic variance, are

\[
r_{SDi} = \frac{r_{Di}}{\sqrt{1 - h_i}}
\]

The likelihood residuals, which estimate components of a likelihood ratio test of deleting an individual observation, are a weighted combination of the standardized Pearson and deviance residuals,

\[
r_{Li} = \text{sign}(r_i) \sqrt{h_i r_{SPi}^2 + (1 - h_i) r_{SDi}^2}
\]
Other Regression Diagnostics

The CBAR statistic is a confidence interval displacement diagnostic that provides a scalar measure of the influence of an individual observation on $\hat{\beta}$. This diagnostic is based on the same idea as Cook’s distance in linear regression theory (Cook and Weisberg 1982), but it uses the one-step estimate:

$$\bar{C}_i = r_{P_i}^2 h_i/(1-h_i)$$

The DIFDEV and DIFCHISQ statistics are diagnostics for detecting ill-fitted observations—observations that contribute heavily to the disagreement between the data and the predicted values of the fitted model. DIFDEV is the change in the deviance that results from deleting an individual observation, and DIFCHISQ is the change in the Pearson chi-square statistic that results from the same deletion. By using the one-step estimate, DIFDEV and DIFCHISQ for the $i$th observation are computed as follows:

$$\text{DIFDEV}_i = r_{Di}^2 + \bar{C}_i$$
$$\text{DIFCHISQ}_i = \bar{C}_i / h_i$$

Joint Tests and Type 3 Tests

Linear hypotheses for $\beta$ are expressed in matrix form as

$$H_0: L\beta = c$$

where $L$ is a matrix of coefficients for the linear hypotheses and $c$ is a vector of constants. The vector of regression coefficients $\beta$ includes slope parameters and intercept parameters. The Wald chi-square statistic for testing $H_0$ is computed as

$$\chi^2_W = (L\hat{\beta} - c)'[L\hat{V}(\hat{\beta})L']^{-1}(L\hat{\beta} - c)$$

where $\hat{V}(\hat{\beta})$ is the estimated covariance matrix. Under $H_0$, $\chi^2_W$ has an asymptotic chi-square distribution with $r$ degrees of freedom, where $r$ is the rank of $L$.

For models that use less-than-full-rank parameterization (as specified by the PARAM=GLM option in the CLASS statement), a Type 3 test of an effect of interest (main effect or interaction) is a test of the Type III estimable functions that are defined for that effect. When the model contains no missing cells, the Type 3 test of a main effect is equivalent to testing the hypothesis of equal marginal means. For more information about Type III estimable functions, see the chapter “The GLM Procedure” and the section “The Four Types of Estimable Functions” in SAS/STAT User’s Guide. Also see Littell, Freund, and Spector (1991).

For models that use full-rank parameterization, all parameters are estimable when there are no missing cells, so it is unnecessary to define estimable functions. The standard test of an effect of interest in this case is the joint test that the values of the parameters associated with that effect are 0. For a model that uses effects parameterization (as specified by the PARAM=EFFECT option in the CLASS statement), the joint test for a main effect is equivalent to testing the equality of marginal means. For a model that uses reference parameterization (as specified by the PARAM=REF option in the CLASS statement), the joint test is equivalent to testing the equality of cell means at the reference level of the other model effects. For more information about the coding scheme and the associated interpretation of results, see Muller and Fetterman (2002, Chapter 14).
If there is no interaction term, the Type 3 test of an effect for a model that uses GLM parameterization is the same as the joint test of the effect for the model that uses full-rank parameterization. In this situation, the joint test is also called the Type 3 test. For a model that contains an interaction term and no missing cells, the Type 3 test of a component main effect under GLM parameterization is the same as the joint test of the component main effect under effect parameterization. Both test the equality of cell means. But this Type 3 test differs from the joint test under reference parameterization, which tests the equality of cell means at the reference level of the other component main effect. If some cells are missing, you can obtain meaningful tests only by testing a Type III estimation function, so in this case you should use GLM parameterization.

The results of a Type 3 test or a joint test do not depend on the order in which you specify the terms in the MODEL statement.

---

**Multithreading**

The GENSELECT procedure allocates data to different threads and calculates the likelihood function, gradient, and Hessian by accumulating the values from all threads. For more information about how PROC GENSELECT uses threads, see the section “Multithreading” on page 81 in Chapter 3, “Shared Concepts.”

---

**Optimization Algorithms**

Several optimization techniques are available in PROC GENSELECT. You can choose a particular optimizer by using the TECHNIQUE= option in the PROC GENSELECT statement. Table 7.12 summarizes the optimization techniques available in PROC GENSELECT.

<table>
<thead>
<tr>
<th>TECHNIQUE=</th>
<th>Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRUREG</td>
<td>Trust region method</td>
</tr>
<tr>
<td>NEWRAP</td>
<td>Newton-Raphson method with line search</td>
</tr>
<tr>
<td>NRRIDG</td>
<td>Newton-Raphson method with ridging</td>
</tr>
<tr>
<td>QUANEW</td>
<td>Quasi-Newton methods</td>
</tr>
<tr>
<td>DBLDOG</td>
<td>Double-dogleg method</td>
</tr>
<tr>
<td>CONGRA</td>
<td>Conjugate gradient methods</td>
</tr>
<tr>
<td>NMSIMP</td>
<td>Nelder-Mead simplex method</td>
</tr>
</tbody>
</table>

There is no algorithm for optimizing general nonlinear functions that always finds the global optimum for a general nonlinear optimization problem in a reasonable amount of time. Because no single optimization technique is always superior to others, PROC GENSELECT provides a variety of optimization techniques that work well in various circumstances. However, you can devise problems for which none of the techniques in PROC GENSELECT can find the correct solution. Moreover, nonlinear optimization can be computationally expensive in terms of time and memory, so you must be careful when matching an algorithm to a problem. The section “Choosing an Optimization Algorithm” on page 82 in Chapter 3, “Shared Concepts,” is helpful in choosing a suitable optimization algorithm.
Displayed Output

The following sections describe the output that PROC GENSELECT produces. The output is organized into various tables, which are discussed in their order of appearance.

Model Information

The “Model Information” table displays basic information about the model, such as the response variable, the frequency variable, the link function, and the model category that the GENSELECT procedure selected based on your input and options. The “Model Information” table also displays the distribution of the data that is assumed by the GENSELECT procedure. For information about how the procedure determines the response distribution, see the section “Response Distributions” on page 395. If you specify the CODE statement, then the table displays the name of the variable in the scoring code that contains the classifications. If you do not also specify the PCATALL option, then the table also displays the name of the variable in the scoring code that contains the predicted probabilities.

Number of Observations

The “Number of Observations” table displays the number of observations read from the input data table and the number of observations used in the analysis. If a FREQ statement is present, the table displays the sum of the frequencies that are read and used. If the events/trials syntax is used, the number of events and trials is also displayed. If you specify a PARTITION statement, the table displays the values for each role.

Response Profile

The “Response Profile” table displays the ordered value from which the GENSELECT procedure determines the probability being modeled as an event in a binary model and displays the ordering of categories in a multinomial model. For each response category level, the frequency used in the analysis is reported. You can affect the ordering of the response values by specifying response-options in the MODEL statement. For a binary model, the note that follows the “Response Profile” table indicates which outcome is modeled as the event. For an ordinal multinomial model, the note that follows the “Response Profile” table indicates how the ordered response levels are accumulated; for a nominal multinomial model, the note indicates the reference response level. If you specify a PARTITION statement, the table displays the values for each role.

Class Level Information

The “Class Level Information” table lists the levels of every variable specified in the CLASS statement. You should check this information to make sure that the data are correct. You can adjust the order of the CLASS variable levels by specifying the ORDER= option in the CLASS statement. You can suppress the “Class Level Information” table completely or partially by specifying the NOCLPRINT= option in the PROC GENSELECT statement.

If the classification variables use a nonsingular parameterization, the “Class Level Information” table also displays the reference value for each variable.
Selection Information

When you specify the SELECTION statement, the GENSELECT procedure by default produces a series of tables that display information about the model selection. The “Selection Information” table informs you about the model selection method, selection and stop criteria, and other parameters that govern the selection. You can suppress this table by specifying DETAILS=NONE in the SELECTION statement.

Iteration History

When you specify the ITHIST option in the PROC GENSELECT statement, the “Iteration History” table displays, for each iteration of the optimization, the number of function evaluations (including gradient and Hessian evaluations), the value of the objective function, the change in the objective function from the previous iteration, and the absolute value of the largest (projected) gradient element. The objective function that is used in the optimization in the GENSELECT procedure is normalized by default to enable comparisons across data tables that have different sampling intensity. You can control normalization by specifying the NORMALIZE= option in the PROC GENSELECT statement.

Convergence Status

The convergence status table is a small ODS table in the default output. In the listing, it appears as a message that indicates whether the optimization succeeded and which convergence criterion was met. If the optimization fails, the message indicates the reason for the failure. If you save the convergence status table to an output data set, a numeric Status variable is added that enables you to assess convergence programmatically. The values of the Status variable indicate the following:

0 Convergence was achieved, or an optimization was not performed (because TECHNIQUE=NONE is specified).
1 The objective function could not be improved.
2 Convergence was not achieved because of a user interruption or because a limit was exceeded, such as the maximum number of iterations or the maximum number of function evaluations. To modify these limits, see the MAXITER=, MAXFUNC=, and MAXTIME= options in the PROC GENSELECT statement.
3 Optimization failed to converge because function or derivative evaluations failed at the starting values or during the iterations or because a feasible point that satisfies the parameter constraints could not be found in the parameter space.

Entry and Removal Candidates

When you specify DETAILS=ALL or DETAILS=STEPS in the SELECTION statement, the GENSELECT procedure produces the “Entry Candidates” or “Removal Candidates” table, which displays the effect names and values of the criterion used to select entering or departing effects at each step of the selection process. For each step, the effects are displayed in sorted order from best to worst of the selection criterion.

Selection Summary

When you specify the SELECTION statement, the GENSELECT procedure produces the “Selection Summary” table, which displays information about which effects were added to or removed from the model in the various steps of the model selection process. The statistic that led to the entry or removal decision is also
displayed. You can request further details about the model selection steps by specifying DETAILS=STEPS or DETAILS=ALL in the SELECTION statement. You can suppress the display of the “Selection Summary” table by specifying DETAILS=NONE in the SELECTION statement.

If you specify the LASSO selection method, then this table displays information about which effect was added to or removed from the model, the number of effects in the model, the lambda value, and the information criteria.

**Stop Reason**

When you specify the SELECTION statement, the GENSELECT procedure produces a simple table that tells you why model selection stopped.

**Selection Reason**

When you specify the SELECTION statement, the GENSELECT procedure produces a simple table that tells you why the final model was selected.

**Selected Effects**

When you specify the SELECTION statement, the GENSELECT procedure produces a simple table that tells you which effects were included in the final model.

**Dimensions**

The “Dimensions” table displays size measures that are derived from the model and the environment. It displays the number of columns in the current design matrix, the number of effects in the current design, the largest number of design columns associated with an effect, the rank of the matrix, and the number of parameters in the current model, including any dispersion parameters.

**Fit Statistics**

The “Fit Statistics” table displays a variety of likelihood-based measures of fit. All statistics are presented in “smaller is better” form. The values that the “Fit Statistics” table displays are not based on a normalized log-likelihood function. If you specify a PARTITION statement or the PARTFIT option, the table displays the values for each role along with statistics for comparing the training, validation, and testing results.

For more information about the statistics displayed in this table, see the section “Model Fit and Assessment Statistics” on page 407.

**Parameter Estimates**

The parameter estimates, their estimated (asymptotic) standard errors, and $p$-values for the hypothesis that the parameter is 0 are presented in the “Parameter Estimates” table. If you request confidence intervals by specifying the CLB option in the MODEL statement, confidence limits are produced for the estimates.

**Parameter Estimates Covariance Matrix**

When you specify the COVB option in the PROC GENSELECT statement, the GENSELECT procedure displays the covariance matrix of the parameter estimates. The covariance matrix is computed as the inverse
of the negative of the matrix of second derivatives of the log-likelihood function with respect to the model parameters (the Hessian matrix), evaluated at the parameter estimates.

**Parameter Estimates Correlation Matrix**

When you specify the CORRB option in the PROC GENSELECT statement, the GENSELECT procedure displays the correlation matrix of the parameter estimates.

**Score Code Variables for Predicted Values**

When you specify the PCATALL option in the CODE statement and you specify the DISTRIBUTION=BINARY option in the MODEL statement, the GENSELECT procedure produces the “Score Code Variables for Predicted Probability” table. For each level of the response variable, this table displays the corresponding variable in the scoring code that contains the predicted probability.

**Timing**

The “Timing” table displays the amount of time (in seconds) that PROC GENSELECT required to perform different tasks in the analysis.

**OutputCasTables Table**

The OutputCasTables table is a special table that has information about each CAS table that is created during a CAS action execution. The information for each CAS table consists of the CAS table name, the caslib in which the table resides, and the number of columns and rows in the CAS table. Because this table is not a typical ODS table that contains analytical results, you cannot include it in the \texttt{table-spec-list} in the \texttt{DISPLAYOUT} statement.

**ODS Table Names**

Each table that the GENSELECT procedure creates has a name associated with it. You must use this name to refer to the table when you use the \texttt{DISPLAY} statement, the \texttt{DISPLAYOUT} statement, or ODS statements. These names are listed in Table 7.13.

Table 7.13  ODS Tables Produced by PROC GENSELECT

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClassInfo</td>
<td>Level information from the \texttt{CLASS} statement</td>
<td>\texttt{CLASS}</td>
<td>Default</td>
</tr>
<tr>
<td>ConvergenceStatus</td>
<td>Status of optimization at conclusion of optimization</td>
<td>PROC GENSELECT</td>
<td>Default</td>
</tr>
<tr>
<td>CorrB</td>
<td>Correlation matrix of parameter estimates</td>
<td>PROC GENSELECT</td>
<td>CORRB</td>
</tr>
<tr>
<td>CovB</td>
<td>Covariance matrix of parameter estimates</td>
<td>PROC GENSELECT</td>
<td>COVB</td>
</tr>
<tr>
<td>Dimensions</td>
<td>Model dimensions</td>
<td>PROC GENSELECT</td>
<td>Default</td>
</tr>
</tbody>
</table>
Table 7.13  continued

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>EntryCandidates</td>
<td>Details about candidates for entry into the model</td>
<td>SELECTION</td>
<td>METHOD=FORWARD</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>DETAILS=STEP</td>
</tr>
<tr>
<td>FitStatistics</td>
<td>Fit statistics</td>
<td>PROC GENSELECT</td>
<td>Default</td>
</tr>
<tr>
<td>IterHistory</td>
<td>Iteration history</td>
<td>PROC GENSELECT</td>
<td>ITHIST</td>
</tr>
<tr>
<td>ModelInfo</td>
<td>Information about the modeling environment</td>
<td>PROC GENSELECT</td>
<td>Default</td>
</tr>
<tr>
<td>ModelAnova</td>
<td>Model analysis of variance (Type III)</td>
<td>MODEL</td>
<td>TYPE3</td>
</tr>
<tr>
<td>NObs</td>
<td>Number of observations read and used, and number of events and trials, if applicable</td>
<td>PROC GENSELECT</td>
<td>Default</td>
</tr>
<tr>
<td>OutputCasTables</td>
<td>See the section “OutputCasTables Table” on page 416</td>
<td>OUTPUT</td>
<td>OUT=</td>
</tr>
<tr>
<td>ParameterEstimates</td>
<td>Solutions for the parameter estimates associated with effects in MODEL statements</td>
<td>PROC GENSELECT</td>
<td>Default</td>
</tr>
<tr>
<td>PredProbName</td>
<td>Displays the predicted probability variable in the scoring code associated with each response level</td>
<td>CODE</td>
<td>PCATALL</td>
</tr>
<tr>
<td>RemovalCandidates</td>
<td>Details about candidates for removal from the model</td>
<td>SELECTION</td>
<td>METHOD=BACKWARD</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>DETAILS=STEP</td>
</tr>
<tr>
<td>ResponseProfile</td>
<td>Response categories for binary and multinomial data</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>SelectedEffects</td>
<td>List of effects selected for the model</td>
<td>SELECTION</td>
<td>Default</td>
</tr>
<tr>
<td>SelectionInfo</td>
<td>Information about the settings for model selection</td>
<td>SELECTION</td>
<td>Default</td>
</tr>
<tr>
<td>SelectionReason</td>
<td>Reason why the particular model was selected</td>
<td>SELECTION</td>
<td>Default</td>
</tr>
<tr>
<td>SelectionSummary</td>
<td>Summary information about model selection steps</td>
<td>SELECTION</td>
<td>Default</td>
</tr>
<tr>
<td>StopReason</td>
<td>Reason for termination of model selection</td>
<td>SELECTION</td>
<td>Default</td>
</tr>
<tr>
<td>Timing</td>
<td>Absolute and relative times for tasks performed by the procedure</td>
<td>PROC GENSELECT</td>
<td>Default</td>
</tr>
</tbody>
</table>

**ODS Graphics**

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in the “Statistical Graphics Using ODS” chapter in *SAS/STAT User’s Guide.*
Before you create graphs, ODS Graphics must be enabled (for example, with the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section “Enabling and Disabling ODS Graphics” in that chapter.

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section “A Primer on ODS Statistical Graphics” in that chapter.

When ODS Graphics is enabled, the SELECTION statement can produce plots to help evaluate the selection process. For information about these plots, see the section “Model Selection Plots” on page 71 in Chapter 3, “Shared Concepts.”

PROC GENSELECT assigns a name to each graph that it creates using ODS. You can use these names to reference the graphs when using ODS. The names are listed in Table 7.14.

<table>
<thead>
<tr>
<th>ODS Graph Name</th>
<th>Plot Description</th>
<th>PLOTS Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoefficientPanel</td>
<td>Coefficients and CHOOSE= criterion by step</td>
<td>COEFFICIENTS</td>
</tr>
<tr>
<td>ChooseCriterionPlot</td>
<td>CHOOSE= criterion by step</td>
<td>COEFFICIENTS(UNPACK)</td>
</tr>
<tr>
<td>CoefficientPlot</td>
<td>Coefficients by step</td>
<td>COEFFICIENTS(UNPACK)</td>
</tr>
<tr>
<td>CriterionPanel</td>
<td>Fit criteria by step</td>
<td>CRITERIA</td>
</tr>
<tr>
<td>AICCPlot</td>
<td>Corrected Akaike’s information criterion by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>AICPlot</td>
<td>Akaike’s information criterion by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>SBCPlot</td>
<td>Schwarz Bayesian information criterion by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>TEST_ASEPlot</td>
<td>Average square error on testing data by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>TRAIN_ASEPlot</td>
<td>Average square error on training data by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>VAL_ASEPlot</td>
<td>Average square error on validation data by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>FitByRolePlot</td>
<td>Progression of average square error by role</td>
<td>FITBYROLE</td>
</tr>
</tbody>
</table>

**Examples: GENSELECT Procedure**

**Example 7.1: Model Selection**

The following statements examine the same data set that is used in the section “Getting Started: GENSELECT Procedure” on page 367, but they request model selection via the forward selection technique. Model effects are added in the order of their significance until no more effects significantly improve the current model.
The DETAILS=ALL option in the SELECTION statement requests that all tables that are related to model selection be produced.

The data set getStarted is shown in the section “Getting Started: GENSELECT Procedure” on page 367. It contains 100 observations on a count response variable (Y), a continuous variable (Total) to be used in Example 7.2, and five categorical variables (C1–C5), each of which has four numerical levels.

A log-linked Poisson regression model is specified by using classification effects for variables C1–C5. The following statements request model selection by the forward selection method:

```r
ods graphics on;

proc genselect data=mycas.getStarted;
  class C1-C5;
  model Y = C1-C5 / Distribution=Poisson;
  selection method=forward details=all plots=all;
run;
ods graphics off;
```

The model selection tables are shown in Output 7.1.1 through Output 7.1.3. Selection graphics produced by the PLOTS= option are displayed in Output 7.1.6 and Output 7.1.7.

The “Selection Information” table in Output 7.1.1 summarizes the settings for the model selection. The effect that produces the smallest approximate SBC is added to the model. The forward selection stops three steps after the smallest SBC is obtained, or when all effects have been added to the model.

Output 7.1.1 Selection Information

<table>
<thead>
<tr>
<th>Selection Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selection Method</td>
</tr>
<tr>
<td>Select Criterion</td>
</tr>
<tr>
<td>Stop Criterion</td>
</tr>
<tr>
<td>Effect Hierarchy Enforced</td>
</tr>
<tr>
<td>Stop Horizon</td>
</tr>
</tbody>
</table>

The “Selection Summary” table in Output 7.1.2 shows the effects that were added to the model and the approximate SBC. Step 0 refers to the null model that contains only an intercept. In the next step, effect C2 resulted in the smallest approximate SBC among the candidate effects. In step 2, the smallest SBC when an effect is added to a model that contains the intercept and C2 was achieved by adding C5 to the model. Similarly, in steps 3, 4, and 5, effects were added in order of the minimum SBC among remaining candidates. Finally, the model in step 2 was chosen as the best because it had the smallest SBC among all the candidate models.
Output 7.1.2 Selection Summary Information

<table>
<thead>
<tr>
<th>Step</th>
<th>Effect</th>
<th>Entered</th>
<th>Number Effects In SBC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Intercept</td>
<td>1</td>
<td>354.7980</td>
</tr>
<tr>
<td>1</td>
<td>C2</td>
<td>2</td>
<td>342.8795</td>
</tr>
<tr>
<td>2</td>
<td>C5</td>
<td>3</td>
<td>333.8182*</td>
</tr>
<tr>
<td>3</td>
<td>C1</td>
<td>4</td>
<td>341.7993</td>
</tr>
<tr>
<td>4</td>
<td>C4</td>
<td>5</td>
<td>351.8375</td>
</tr>
<tr>
<td>5</td>
<td>C3</td>
<td>6</td>
<td>363.8674</td>
</tr>
</tbody>
</table>

* Optimal Value Of Criterion

Selection stopped at a local minimum of the STOP criterion.

The model at step 2 is selected.

Selected Effects: Intercept C2 C5

For each step of the selection process, the DETAILS=ALL option displays the candidate effects for entering the model along with their SELECT= criterion. Output 7.1.3 displays this table for the first step; the other steps are not shown here.

Output 7.1.3 Step1 Entry Candidates

<table>
<thead>
<tr>
<th>Rank</th>
<th>Effect</th>
<th>SBC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>C2</td>
<td>342.8795</td>
</tr>
<tr>
<td>2</td>
<td>C5</td>
<td>347.5376</td>
</tr>
<tr>
<td>3</td>
<td>C1</td>
<td>355.7067</td>
</tr>
<tr>
<td>4</td>
<td>C4</td>
<td>361.7379</td>
</tr>
<tr>
<td>5</td>
<td>C3</td>
<td>365.7339</td>
</tr>
</tbody>
</table>

The DETAILS=ALL option also displays the dimensions, fit statistics, and parameter estimates at each step of the selection process; these details are not shown here.

Output 7.1.4 displays information about the selected model. Notice that the –2 log-likelihood value in the “Fit Statistics” table is larger than the value for the full model in Figure 7.6. This is expected because the selected model contains only a subset of the parameters. Because the selected model is more parsimonious than the full model, the information criteria AIC, AICC, and BIC are smaller than in the full model, indicating a better fit.

Output 7.1.4 Fit Statistics

<table>
<thead>
<tr>
<th>Fit Statistics</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>-2 Log Likelihood</td>
<td>303.58036</td>
</tr>
<tr>
<td>AIC (smaller is better)</td>
<td>317.58036</td>
</tr>
<tr>
<td>AICC (smaller is better)</td>
<td>318.79775</td>
</tr>
<tr>
<td>SBC (smaller is better)</td>
<td>335.81655</td>
</tr>
</tbody>
</table>
The parameter estimates of the selected model are shown in Output 7.1.5. Notice that the effects are listed in the “Parameter Estimates” table in the order in which they were specified in the MODEL statement and not the order in which they were added to the model.

**Output 7.1.5** Parameter Estimates

<table>
<thead>
<tr>
<th>Parameter</th>
<th>DF</th>
<th>Estimate</th>
<th>Standard Error</th>
<th>Chi-Square</th>
<th>Pr &gt; ChiSq</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>1</td>
<td>0.528024</td>
<td>0.215231</td>
<td>6.0186</td>
<td>0.0142</td>
</tr>
<tr>
<td>C2 0</td>
<td>1</td>
<td>0.962636</td>
<td>0.236921</td>
<td>16.5088</td>
<td>&lt;.0001</td>
</tr>
<tr>
<td>C2 1</td>
<td>1</td>
<td>0.845132</td>
<td>0.233634</td>
<td>13.0851</td>
<td>0.0003</td>
</tr>
<tr>
<td>C2 2</td>
<td>1</td>
<td>0.213247</td>
<td>0.233634</td>
<td>13.0851</td>
<td>0.0003</td>
</tr>
<tr>
<td>C2 3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1.0000</td>
</tr>
<tr>
<td>C5 0</td>
<td>1</td>
<td>-0.763491</td>
<td>0.207082</td>
<td>13.5933</td>
<td>0.0002</td>
</tr>
<tr>
<td>C5 1</td>
<td>1</td>
<td>-0.785915</td>
<td>0.201271</td>
<td>15.2471</td>
<td>&lt;.0001</td>
</tr>
<tr>
<td>C5 2</td>
<td>1</td>
<td>-0.541212</td>
<td>0.208661</td>
<td>6.7275</td>
<td>0.0095</td>
</tr>
<tr>
<td>C5 3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

The coefficient panel in Figure 7.1.6 enables you to visualize the selection process. In this plot, standardized coefficients of all the effects that are selected at some step of the stepwise method are plotted as a function of the step number. This enables you to assess the relative importance of the effects that are selected at any step of the selection process and to know when effects entered the model. The lower plot in the panel shows how the criterion that is used to choose the selected model changes as effects enter or leave the model.
The criterion panel in Figure 7.1.7 provides a graphical view of the progression of the fit criteria as the selection process evolves.
Example 7.2: Gamma Model

The following statements examine the data set `getStarted`, which is used in the section “Getting Started: GENSELECT Procedure” on page 367, but they request that a log-linked gamma model be fit by using the continuous variable `Total` as the response instead of the count variable `Y`. The following statements fit a log-linked gamma model to these data by using classification effects for the variables `C1–C5`. The `CLB` MODEL statement option requests that 95% confidence limits be computed and displayed along with the parameter estimates. The `CODE` statement requests that a text file named “Scoring Parameters.txt” be created. This file contains a SAS program that has information from the model that allows scoring of a new data set based on the parameter estimates from the current model.

```sas
proc genselect data=mycas.getStarted;
   class C1-C5;
   model Total = C1-C5 / Distribution=Gamma Link=Log CLB;
   code File='ScoringParameters.txt';
run;
```

The “Parameter Estimates” table in Output 7.2.1 shows the resulting regression model parameter estimates and the estimated gamma dispersion parameter.
Now suppose you want to compute predicted values for some different data. If \( \mathbf{x} \) is a vector of explanatory variables that might not be in the original data and \( \hat{\mathbf{\beta}} \) is the vector of estimated regression parameters from the model, then \( \mu = g^{-1}(\mathbf{x}'\hat{\mathbf{\beta}}) \) is the predicted value of the mean, where \( g \) is the log link function in this case.

The following data contain new values of the regression variables \( C1 \text{–} C5 \), from which you can compute predicted values based on information in the SAS program that is created by the CODE statement. This is called scoring the new data set.

```sas
data ScoringData;
  input C1-C5;
datalines;
  3 3 1 0 2
  1 1 2 2 0
  3 2 2 2 0
  1 1 2 3 2
  1 1 2 3 3
  3 1 1 0 1
  0 2 1 0 0
  2 1 3 1 3
  3 2 3 2 0
  3 0 2 0 1
;
```

---

### Output 7.2.1 Parameter Estimates

<table>
<thead>
<tr>
<th>Parameter</th>
<th>DF</th>
<th>Estimate</th>
<th>Standard Error</th>
<th>Chi-Square</th>
<th>Pr &gt; ChiSq</th>
<th>95% Confidence Limits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>1</td>
<td>4.028096</td>
<td>0.454883</td>
<td>78.4153</td>
<td>&lt;.0001</td>
<td>3.13654 - 4.91965</td>
</tr>
<tr>
<td>C1 0</td>
<td>1</td>
<td>-0.064442</td>
<td>0.256719</td>
<td>0.0630</td>
<td>0.8018</td>
<td>0.43872</td>
</tr>
<tr>
<td>C1 1</td>
<td>1</td>
<td>-1.308470</td>
<td>0.318002</td>
<td>16.9305</td>
<td>&lt;.0001</td>
<td>-1.93174 - 0.68520</td>
</tr>
<tr>
<td>C1 2</td>
<td>1</td>
<td>0.295802</td>
<td>0.257834</td>
<td>1.3162</td>
<td>0.2513</td>
<td>0.80115</td>
</tr>
<tr>
<td>C1 3</td>
<td>0</td>
<td>0</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>C2 0</td>
<td>1</td>
<td>1.154556</td>
<td>0.295110</td>
<td>15.3060</td>
<td>&lt;.0001</td>
<td>0.57615 - 1.73296</td>
</tr>
<tr>
<td>C2 1</td>
<td>1</td>
<td>0.824472</td>
<td>0.295273</td>
<td>7.7966</td>
<td>0.0052</td>
<td>0.24575 - 1.40320</td>
</tr>
<tr>
<td>C2 2</td>
<td>1</td>
<td>-0.287942</td>
<td>0.288421</td>
<td>0.9967</td>
<td>0.3181</td>
<td>-0.85324 - 0.27735</td>
</tr>
<tr>
<td>C2 3</td>
<td>0</td>
<td>0</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>C3 0</td>
<td>1</td>
<td>-0.008546</td>
<td>0.283509</td>
<td>0.0009</td>
<td>0.9760</td>
<td>0.54712</td>
</tr>
<tr>
<td>C3 1</td>
<td>1</td>
<td>-0.319783</td>
<td>0.276053</td>
<td>1.3419</td>
<td>0.2467</td>
<td>-0.86084 - 0.22177</td>
</tr>
<tr>
<td>C3 2</td>
<td>1</td>
<td>-0.071452</td>
<td>0.296418</td>
<td>0.0581</td>
<td>0.8095</td>
<td>-0.65242 - 0.20952</td>
</tr>
<tr>
<td>C3 3</td>
<td>0</td>
<td>0</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>C4 0</td>
<td>1</td>
<td>-0.143018</td>
<td>0.288101</td>
<td>0.2464</td>
<td>0.6196</td>
<td>-0.70769 - 0.42165</td>
</tr>
<tr>
<td>C4 1</td>
<td>1</td>
<td>-0.219348</td>
<td>0.290441</td>
<td>0.5704</td>
<td>0.4501</td>
<td>-0.78860 - 0.34991</td>
</tr>
<tr>
<td>C4 2</td>
<td>1</td>
<td>0.091763</td>
<td>0.278635</td>
<td>0.1085</td>
<td>0.7419</td>
<td>-0.45435 - 0.62088</td>
</tr>
<tr>
<td>C4 3</td>
<td>0</td>
<td>0</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>C5 0</td>
<td>1</td>
<td>-1.227558</td>
<td>0.267605</td>
<td>21.0425</td>
<td>&lt;.0001</td>
<td>-1.75205 - 0.70306</td>
</tr>
<tr>
<td>C5 1</td>
<td>1</td>
<td>-0.560699</td>
<td>0.252238</td>
<td>4.9413</td>
<td>0.0262</td>
<td>-1.05508 - 0.06632</td>
</tr>
<tr>
<td>C5 2</td>
<td>1</td>
<td>-0.252865</td>
<td>0.259908</td>
<td>0.9473</td>
<td>0.3304</td>
<td>-0.76238 - 0.25645</td>
</tr>
<tr>
<td>C5 3</td>
<td>0</td>
<td>0</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>Dispersion</td>
<td>1</td>
<td>1.672305</td>
<td>0.238205</td>
<td>1.26494</td>
<td>2.21086</td>
<td>1.26494 - 2.21086</td>
</tr>
</tbody>
</table>

---
The following SAS DATA step creates the new data set Scores, which contains a variable P_Total that represents the predicted values of Total, along with the variables C1–C5. The resulting data are shown in Output 7.2.2.

```sas
data Scores;
   set ScoringData;
   %inc 'ScoringParameters.txt';
run;
proc print data=Scores;
run;
```

**Output 7.2.2** Predicted Values for Scoring Data

<table>
<thead>
<tr>
<th>Obs</th>
<th>C1</th>
<th>C2</th>
<th>C3</th>
<th>C4</th>
<th>C5</th>
<th>P_Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>27.449</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>10.349</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>12.590</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>25.020</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>32.222</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>46.020</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>7.282</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>138.244</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>13.523</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>82.063</td>
</tr>
</tbody>
</table>

**References**


# Chapter 8
## The ICA Procedure

## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview: ICA Procedure</td>
<td>428</td>
</tr>
<tr>
<td>PROC ICA Features</td>
<td>428</td>
</tr>
<tr>
<td>Using CAS Sessions and CAS Engine Librefs</td>
<td>429</td>
</tr>
<tr>
<td>Getting Started: ICA Procedure</td>
<td>429</td>
</tr>
<tr>
<td>Syntax: ICA Procedure</td>
<td>435</td>
</tr>
<tr>
<td>PROC ICA Statement</td>
<td>435</td>
</tr>
<tr>
<td>BY Statement</td>
<td>438</td>
</tr>
<tr>
<td>DISPLAY Statement</td>
<td>438</td>
</tr>
<tr>
<td>DISPLAYOUT Statement</td>
<td>439</td>
</tr>
<tr>
<td>OUTPUT Statement</td>
<td>439</td>
</tr>
<tr>
<td>VAR Statement</td>
<td>441</td>
</tr>
<tr>
<td>Details: ICA Procedure</td>
<td>441</td>
</tr>
<tr>
<td>Computing Independent Components</td>
<td>441</td>
</tr>
<tr>
<td>Whitening and Dimension Reduction</td>
<td>442</td>
</tr>
<tr>
<td>Missing Values</td>
<td>443</td>
</tr>
<tr>
<td>Displayed Output</td>
<td>443</td>
</tr>
<tr>
<td>Model Information</td>
<td>443</td>
</tr>
<tr>
<td>Dimensions</td>
<td>443</td>
</tr>
<tr>
<td>Number of Observations</td>
<td>443</td>
</tr>
<tr>
<td>Centering and Scaling Information</td>
<td>443</td>
</tr>
<tr>
<td>Eigenvalues</td>
<td>444</td>
</tr>
<tr>
<td>Whitening Transformation Matrix</td>
<td>444</td>
</tr>
<tr>
<td>Dewhitenning Transformation Matrix</td>
<td>444</td>
</tr>
<tr>
<td>Demixing Matrix</td>
<td>444</td>
</tr>
<tr>
<td>Mixing Matrix</td>
<td>444</td>
</tr>
<tr>
<td>ODS Table Names</td>
<td>444</td>
</tr>
<tr>
<td>Examples: ICA Procedure</td>
<td>445</td>
</tr>
<tr>
<td>Example 8.1: Extracting Independent Components with Dimension Reduction</td>
<td>445</td>
</tr>
<tr>
<td>References</td>
<td>449</td>
</tr>
</tbody>
</table>
Overview: ICA Procedure

The ICA procedure performs independent component analysis in SAS Viya.

Independent component analysis attempts to extract from the observed multivariate data independent components (also called factors or latent variables) that are as statistically independent from each other as possible. The methods that PROC ICA implements, where statistical independence is defined as the maximization of non-Gaussianity, seek linear combinations of the observed random variables that maximize the non-Gaussianity of the estimated independent components.

Independent component analysis is one of the techniques for performing blind source separation or blind signal separation, where source means an original signal or independent component, and blind means that the mixing process of the source signals is unknown and few assumptions about the source signals are made. You can use independent component analysis to reveal the hidden structure of the data in many applications, including feature extraction and signal separation.

PROC ICA Features

The ICA procedure has the following main features:

- provides a VAR statement to specify the numeric variables for the analysis
- supports whitening and dimension reduction
- produces an output data table that contains independent components and whitened variables

PROC ICA implements the following methods in parallel, based on the FastICA algorithm of Hyvärinen and Oja (2000):

- symmetric decorrelation, which calculates all the independent components simultaneously
- deflationary decorrelation, which extracts the independent components successively

Because PROC ICA runs on SAS Cloud Analytic Services (CAS), it also does the following:

- enables you to run on a cluster of machines that distribute the data and the computations
- enables you to run in single-machine mode
- exploits all the available cores and concurrent threads. For information about how PROC ICA uses threads, see the section “Multithreading” on page 81 in Chapter 3, “Shared Concepts.”
Using CAS Sessions and CAS Engine Librefs

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:

```
proc options option=(CASHOST CASPORT);
run;
```

In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:

```
cas mysess;
libname mycas cas sessref=mysess;
```

The CAS statement creates the CAS session named `mysess`, and the LIBNAME statement creates the `mycas` CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the CASHOST and CASPORT of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the `mysess` session, you can terminate it by using the TERMINATE option in the CAS statement as follows:

```
cas mysess terminate;
```

For more information about the CAS and LIBNAME statements, see the section “Introduction to Shared Concepts” on page 10 in Chapter 3, “Shared Concepts.”

---

Getting Started: ICA Procedure

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”
The following DATA step creates the mycas.sample data table, which provides simulated signal data, in your CAS session. The data for this example are the result of multiplying the source signal matrix $S$ and the mixing matrix $A$; that is, the data matrix $X = SA$. Figure 8.1 shows the original source signals. Figure 8.2 shows the observed mixtures of the source signals. The problem is to recover the original source signals ($s_1$, $s_2$, and $s_3$) shown in Figure 8.1 only from the observed signal mixtures ($x_1$, $x_2$, and $x_3$) shown in Figure 8.2.

```sas
data mycas.sample;
   keep t x:;
   array S[200,3]; /* S: source signals */
   array A[3,3]; /* A: mixing matrix */
   array x[3] x1-x3; /* X: mixed signals */
   N = 200;
   do i = 1 to 3;
      do j = 1 to 3;
         A[i,j] = 0.7*uniform(12345);
      end;
   end;
   do i = 1 to N;
      S[i,1] = cos(i/3);
      S[i,2] = 0.4*(mod(i,23)-11)/7)**5;
      S[i,3] = ((mod(i,29)-7)/11)-0.7;
   end;
   do i = 1 to N;
      t = i;
      do j = 1 to 3;
         x[j] = 0;
      do k = 1 to 3;
         x[j] = x[j] + S[i,k]*A[k,j];
      end;
      output;
   end;
run;
```
**Figure 8.1** Original Source Signals

![Figure 8.1](image1)

**Figure 8.2** Observed Signal Mixtures

![Figure 8.2](image2)
These statements assume that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

The following statements invoke the ICA procedure, which requests the independent component analysis of the data, outputs the computed independent components to an output data table, and produces the tables in Figure 8.3 through Figure 8.6:

```sas
proc ica data=mycas.sample seed=345;
  var x1-x3;
  output out=mycas.scores component=c copyvar=t;
run;
```

Figure 8.3 displays the “Model Information,” “Dimensions,” “Number of Observations,” and “Centering and Scaling Information” tables.

The “Model Information” table identifies the data source and shows that the independent component extraction method is symmetric decorrelation, which is the default. The nonquadratic function \( \log \cosh \) is used in the approximation of negentropy, and the eigenvalue proportion threshold is set to 0; these are the defaults. The random number seed is set to 345. Random number generation is used to initialize the demixing matrix. Changing the random number seed value changes the initial demixing matrix, which yields a different estimated demixing matrix.

The “Dimensions” table indicates that there are three variables to be analyzed and three independent components to be computed. If you omit the N= option in the PROC ICA statement, the default is the number of numeric variables to be analyzed. The table also shows that there are three whitened variables and that three independent components are actually extracted.

The “Number of Observations” table shows that all 200 of the sample observations in the input data are used in the analysis; all the samples are used because they all contain complete data.

The “Centering and Scaling Information” table displays the centering and scaling information of the analysis variables.

Figure 8.3 Model Information, Dimensions, Number of Observations, and Centering and Scaling Information

<table>
<thead>
<tr>
<th>Model Information</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source</td>
<td>SAMPLE</td>
</tr>
<tr>
<td>Component Extraction Method</td>
<td>Symmetric Decorrelation</td>
</tr>
<tr>
<td>Negentropy Approximation Function</td>
<td>Log-cosh</td>
</tr>
<tr>
<td>Eigenvalue Proportion Threshold</td>
<td>0</td>
</tr>
<tr>
<td>Random Number Seed</td>
<td>345</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dimensions</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Variables</td>
<td>3</td>
</tr>
<tr>
<td>Number of Whitened Variables</td>
<td>3</td>
</tr>
<tr>
<td>Number of Independent Components</td>
<td>3</td>
</tr>
<tr>
<td>Number of Independent Components Extracted</td>
<td>3</td>
</tr>
</tbody>
</table>

Number of Observations Read 200
Number of Observations Used 200
Figure 8.3 continued

<table>
<thead>
<tr>
<th>Variable</th>
<th>Subtracted off</th>
<th>Divided by</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>-0.01231</td>
<td>0.32312</td>
</tr>
<tr>
<td>x2</td>
<td>-0.04632</td>
<td>0.56436</td>
</tr>
<tr>
<td>x3</td>
<td>-0.05567</td>
<td>0.89319</td>
</tr>
</tbody>
</table>

Figure 8.4 displays the “Eigenvalues” table.

Figure 8.4 Eigenvalues Table

<table>
<thead>
<tr>
<th>Eigenvalues</th>
<th>Eigenvalue</th>
<th>Difference</th>
<th>Proportion</th>
<th>Cumulative</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.663628</td>
<td>2.352835</td>
<td>0.8879</td>
<td>0.8879</td>
</tr>
<tr>
<td>2</td>
<td>0.310793</td>
<td>0.285215</td>
<td>0.1036</td>
<td>0.9915</td>
</tr>
<tr>
<td>3</td>
<td>0.025578</td>
<td>0.0085</td>
<td>1.0000</td>
<td></td>
</tr>
</tbody>
</table>

Figure 8.5 displays the “Whitening Transformation Matrix” and “Dewhitening Transformation Matrix” tables. In the whitening transformation matrix that is shown, the whitened variables are represented as a linear combination of the original variables that are centered and scaled. The dewhitening transformation matrix is the pseudoinverse of the whitening matrix.

Figure 8.5 Whitening and Dewhitening Transformation Matrices

<table>
<thead>
<tr>
<th>Whitening Transformation Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variable</td>
</tr>
<tr>
<td>----------</td>
</tr>
<tr>
<td>x1</td>
</tr>
<tr>
<td>x2</td>
</tr>
<tr>
<td>x3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dewhitening Transformation Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>Whitened Variable</td>
</tr>
<tr>
<td>-------------------</td>
</tr>
<tr>
<td>White1</td>
</tr>
<tr>
<td>White2</td>
</tr>
<tr>
<td>White3</td>
</tr>
</tbody>
</table>

Figure 8.6 displays the “Demixing Matrix” and “Mixing Matrix” tables. In the demixing matrix that is shown, the independent components are represented as a linear combination of the original variables that are standardized. The mixing matrix is the pseudoinverse of the demixing matrix.
Chapter 8: The ICA Procedure

Figure 8.6 Demixing and Mixing Matrices

<table>
<thead>
<tr>
<th>Demixing Matrix</th>
<th>Comp1</th>
<th>Comp2</th>
<th>Comp3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variable</td>
<td>x1</td>
<td>x2</td>
<td>x3</td>
</tr>
<tr>
<td>Comp1</td>
<td>2.51490</td>
<td>0.32078</td>
<td>2.86227</td>
</tr>
<tr>
<td>Comp2</td>
<td>-0.43425</td>
<td>1.17723</td>
<td>1.66851</td>
</tr>
<tr>
<td>Comp3</td>
<td>-2.44766</td>
<td>-2.07527</td>
<td>-3.66231</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mixing Matrix</th>
<th>x1</th>
<th>x2</th>
<th>x3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Component</td>
<td>x1</td>
<td>x2</td>
<td>x3</td>
</tr>
<tr>
<td>Comp1</td>
<td>-0.12351</td>
<td>-0.69340</td>
<td>-0.41244</td>
</tr>
<tr>
<td>Comp2</td>
<td>-0.82569</td>
<td>-0.32078</td>
<td>-0.79147</td>
</tr>
<tr>
<td>Comp3</td>
<td>0.55043</td>
<td>0.64520</td>
<td>0.45108</td>
</tr>
</tbody>
</table>

Using the output data table mycas.scores and PROC SGPLOT or PROC SGRENDER (code not shown), you can plot the computed independent components. Figure 8.7 shows three independent components (c1, c2, and c3) that are the estimates of the three original source signals (s1, s2, and s3) in Figure 8.1. The original source signals are accurately estimated from the observed signal mixtures shown in Figure 8.2, up to multiplicative signed scalars. The order of the computed independent components cannot be determined in the independent component analysis model.

Figure 8.7 Estimated Source Signals
Syntax: ICA Procedure

The following statements are available in the ICA procedure:

```
PROC ICA <options>;
   BY variables;
   DISPLAY <table-list> </options>;
   DISPLAYOUT table-spec-list </options>;
   OUTPUT OUT=CAS-libref.data-table
      <COPYVARS=(variables)>
      <keyword = prefix>...<keyword = prefix>;
   VAR variables;
```

The PROC ICA statement is required. All other statements are optional. The following sections describe the PROC ICA statement and then describe the other statements in alphabetical order.

PROC ICA Statement

```
PROC ICA <options>;
```

The PROC ICA statement invokes the ICA procedure. Table 8.1 summarizes the options available in the PROC ICA statement.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Basic Options</strong></td>
<td></td>
</tr>
<tr>
<td>DATA=</td>
<td>Specifies the CAS input data table</td>
</tr>
<tr>
<td><strong>Analysis Options</strong></td>
<td></td>
</tr>
<tr>
<td>EIGTHRESH=</td>
<td>Specifies the threshold for the proportion of variance explained by eigenvalues</td>
</tr>
<tr>
<td>GFUNCTION=</td>
<td>Specifies the nonquadratic function to be used in the approximation of negentropy</td>
</tr>
<tr>
<td>METHOD=</td>
<td>Specifies the independent component extraction method to be used</td>
</tr>
<tr>
<td>N=</td>
<td>Specifies the number of independent components to be computed</td>
</tr>
<tr>
<td>NOCENTER</td>
<td>Suppresses centering of the numeric variables to be analyzed</td>
</tr>
<tr>
<td>NOSCALE</td>
<td>Suppresses scaling of the numeric variables to be analyzed</td>
</tr>
<tr>
<td>PREFIX=</td>
<td>Specifies a prefix for naming the independent components</td>
</tr>
<tr>
<td>SEED=</td>
<td>Specifies the seed value for pseudorandom number generation</td>
</tr>
<tr>
<td>WHITEPREFIX=</td>
<td>Specifies a prefix for naming the whitened variables</td>
</tr>
</tbody>
</table>

The following list provides details about these options.
Chapter 8: The ICA Procedure

**DATA=** *CAS-libref.data-table*

names the input data table for PROC ICA to use. The default is the most recently created data table. *CAS-libref.data-table* is a two-level name, where

*CAS-libref* refers to a collection of information that is defined in the LIBNAME statement and includes the *caslib*, which includes a path to the data, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about *CAS-libref*, see the section “Using CAS Sessions and CAS Engine Librefs” on page 429.

*data-table* specifies the name of the input data table.

**EIGTHRESH=** *p*

**EIGENTHRESH=** *p*

specifies the threshold for the proportion of variance explained by eigenvalues, where $0 \leq p \leq 1$. If the proportion of variance that an eigenvalue explains is less than the threshold, the eigenvalue is discarded in the analysis. You can use this option to reduce the dimensionality of the input data. For more information, see the section “Whitening and Dimension Reduction” on page 442. By default, EIGTHRESH=0.

**GFUNCTION=** *EXP | LOGCOSH*

**GFUNC=** *EXP | LOGCOSH*

specifies the nonquadratic function $G(x)$ to be used in the approximation of negentrop.. You can specify the following function types:

**EXP**

uses the exponential function, $G(u) = -e^{-u^2/2}$.

**LOGCOSH**

uses the log cosh function, $G(u) = \log \cosh(u)$.

By default, GFUNCTION=LOGCOSH.

**METHOD=** *DEFLATION< (defl-options) > | SYMMETRIC< (symm-options) >*

specifies the independent component extraction method to be used. You can specify the following values:

**DEFLATION< (defl-options) >**

requests the deflationary decorrelation method. You can specify the following *defl-options*:

**MAXITER=** *n*

specifies the maximum number of iterations to perform. By default, MAXITER=500.

**TOL=** *n*

**TOLERANCE=** *n*

specifies the convergence criterion. By default, TOL=1E–4.
SYMMEtrIC<(symm-options)>  
requests the symmetric decorrelation method. You can specify the following symm-options:

**MAXITER=n**  
specifies the maximum number of iterations to perform. By default, MAXITER=500.

**TOL=n**  
**TOLERANCE=n**  
specifies the convergence criterion. By default, TOL=1E–4.

By default, METHOD=SYMMETRIC.

**N=number**  
specifies the number of independent components to be computed. The default is the number of numeric variables to be analyzed. The number must be an integer greater than or equal to 0. If N=0, the default is used.

**NOCENTER**  
suppresses centering of the numeric variables to be analyzed. This option is useful if the analysis variables are already centered and scaled.

**NOSCALE**  
suppresses scaling of the numeric variables to be analyzed. This option is useful if the analysis variables are already centered and scaled.

**PREFIX=name**  
specifies a prefix for naming the independent components. By default, the names are Comp1, Comp2, ..., Compn. If you specify PREFIX=Abc, the components are named Abc1, Abc2, Abc3, and so on. The number of characters in the prefix plus the number of digits required to designate the variables should not exceed the current name length that is defined by the VALIDVARNANE= system option.

**SEED=number**  
specifies an integer as the seed value for pseudorandom number generation. PROC ICA uses random values to initialize the demixing matrix. Changing the seed value changes the initial demixing matrix, resulting in a different estimated demixing matrix. If you do not specify this option or if number is less than or equal to 0, the seed is generated by reading the time of day from the computer’s clock. By default, SEED=0.

**WHITEPREFIX=name**  
**WPREFIX=name**  
specifies a prefix for naming the whitened variables. By default, the names are White1, White2, ..., Whiten. If you specify WHITEPREFIX=Abc, the components are named Abc1, Abc2, Abc3, and so on. The number of characters in the prefix plus the number of digits required to designate the variables should not exceed the current name length that is defined by the VALIDVARNAME= system option.
**BY Statement**

```
BY variables;
```

You can specify a BY statement in PROC ICA to obtain separate analyses of observations in groups that are defined by the values of the BY variables. If you specify more than one BY statement, only the last one specified is used. For more information, see the discussion of BY-group processing in *SAS Language Reference: Concepts*.

---

**DISPLAY Statement**

```
DISPLAY <table-list> < / options> ;
```

The DISPLAY statement enables you to specify a list of display tables to display or exclude. This statement is similar to the ODS SELECT, ODS EXCLUDE, and ODS TRACE statements. However, the DISPLAY statement can improve performance when a large number of tables could be generated (such as in BY-group processing). The procedure processes the DISPLAY statement on a CAS server and thus sends only a subset of ODS tables to the SAS client. Because ODS statements are processed on a SAS client, first all the generated display tables are sent to the client, and then the client creates a subset.

If you use both DISPLAY and ODS statements together, the DISPLAY statement takes precedence over the ODS statements. Note that the ODS EXCLUDE statement processes tables that are sent to the client after they have been filtered by the DISPLAY statement. In some cases, it might appear that the ODS EXCLUDE statement is taking precedence because it can further filter the tables. For more information about ODS, see *SAS Output Delivery System: Procedures Guide*.

You can specify the `table-list` as a list of table names, paths, partial pathnames, and regular expressions.

The table names that you can specify are listed in the section “ODS Table Names” on page 444. A path is a table name that is prefixed with dot-separated grouping information. For example, a `SelectionSummary` table that a procedure produces during a selection routine might have the path `Bygroup1.Summary.SelectionSummary`. A partial pathname does not include all groups; for example, `SelectionSummary` and `Summary.SelectionSummary` are partial pathnames for `Bygroup1.Summary.SelectionSummary`.

When you specify a table name or partial pathname, all display tables whose paths end in the specified name are selected for display or exclusion. For example, both `SelectionSummary` and `Summary.SelectionSummary` select `Bygroup1.Summary.SelectionSummary`.

A regular expression is enclosed in forward slashes (/). For example, specifying “/tions/” selects all pathnames that contain the substring “tions”; in particular, the `Bygroup1.Summary.SelectionSummary` table is selected. Specifying “!/tions/” selects all pathnames that do not contain the substring “tions”; in particular, the `Bygroup1.Summary.SelectionSummary` table is not selected.

You can specify the following `options` after a slash (/):

- **CASESENSITIVE**
  performs a case-sensitive comparison of table names in the `table-list` to display table names when tables are subsetted for display. To preserve case, you must enclose table names in the `table-list` in quotation marks.
**DISPLAYOUT Statement**

DISPLAYOUT `table-spec-list` `<options>` ;

The DISPLAYOUT statement enables you to create CAS output tables from your displayed output. This statement is similar to the ODS OUTPUT statement. For more information about ODS, see *SAS Output Delivery System: Procedures Guide*.

The `table-spec-list` specifies a list of CAS output tables to create. Each entry in the list has either a `key=value` format or a `key` format:

- `key=value` specifies `key` as the ODS table name, path, or partial pathname, and specifies `value` as the CAS output table name.
- `key` specifies `key` as the ODS table name and also as the CAS output table name.

The ODS table names that you can specify are listed in the section “ODS Table Names” on page 444. You cannot specify the ODS table named OutputCasTables in the `table-spec-list`.

Table names and partial pathnames are discussed under the DISPLAY statement. The DISPLAYOUT statement does not support regular expressions.

You can specify the following `options` after a slash (/):

- **INCLUDEALL** creates output CAS tables for all display tables. The name of the created output CAS table is the same as the corresponding display table name. If you specify this option, the `table-spec-list` specification is ignored.
- **NOREPLACE** does not replace any existing CAS output table of the same name.
- **REPEATED** replicates all CAS output tables on all nodes.

**OUTPUT Statement**

OUTPUT OUT=`CAS-libref.data-table` `<COPYVARS=(variables)> `<keyword `=`prefix>> . . . `<keyword `=`prefix>> ;`
Chapter 8: The ICA Procedure

The OUTPUT statement creates a data table that contains observationwise statistics, which are computed after PROC ICA fits the model. If you do not specify a *keyword*, then only the independent component values are included.

The variables in the input data table are *not* included in the output data table, in order to avoid data duplication for large data tables; however, variables that you specify in the `COPYVARS=` option are included.

You must specify the following option:

```
OUT=CAS-libref.data-table
```

names the output data table for PROC ICA to use. You must specify this option before any other options. *CAS-libref.data-table* is a two-level name, where

```
CAS-libref
```

refers to a collection of information that is defined in the LIBNAME statement and includes the `caslib`, which includes a path to where the data table is to be stored, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about *CAS-libref*, see the section “Using CAS Sessions and CAS Engine Librefs” on page 429.

```
data-table
```

specifies the name of the output data table.

You can also specify the following syntax elements:

```
COPYVAR=variable
COPYVARS=(variables)
```

transfers one or more *variables* from the input data table to the output data table.

```
keyword <=prefix>
```

specifies a statistic to include in the output data table and optionally a *prefix* for naming the output variables. If you do not provide a *prefix*, PROC ICA assigns a default prefix that is based on the type of statistic requested.

You can specify the following *keywords* to add statistics to the output data table:

```
COMP
COMPONENT
WHITE
```

requests the source values for each independent component. The default prefix is the one that you specify in the `PREFIX=` option.

requests the source values for each whitened variable. The default prefix is the one that you specify in the `WHITEPREFIX=` option.

The output variables that contain the requested statistic are named as follows, according to the *keyword* that you specify:

- *The keyword* COMP defines an output variable for each independent component, so the variables that correspond to each successive component are named by appending a number to the prefix. For example, if the model has three independent components, then COMP=C produces the variables C1, C2, and C3.
VAR Statement

VAR variables;

The VAR statement lists the numeric variables to be analyzed. If you omit the VAR statement, all numeric variables that are not specified in other statements are analyzed.

Details: ICA Procedure

Computing Independent Components

Independent component analysis defines a generative model for the observed data. In the model, the observed data variables are considered to be linear combinations of the latent variables that are non-Gaussian and maximally independent; that is, \( X = SA \), where \( X \) is the data matrix, \( S \) are the latent variables or independent components, and \( A \) is the mixing matrix. By applying the central limit theorem, you can estimate the independent components by maximizing the non-Gaussianity of \( XW \), where \( W \) is called the demixing matrix, the pseudoinverse of \( A \).

Let \( X \) be the centered and scaled data matrix that is whitened by using the eigenvalue decomposition, and let \( w_i \) be an orthonormal weight vector for the \( i \)th independent component. The FastICA algorithm of Hyvärinen and Oja (2000) that the ICA procedure implements is based on a fixed-point iteration scheme for finding a maximum of the non-Gaussianity of \( Xw_i \), which is measured by the approximation of negentropy,

\[
J(u) \propto \left( E[G(u)] - E[G(v)] \right)^2
\]

where the random variable \( u \) is assumed to be of zero mean and unit variance, \( v \) is a Gaussian variable with zero mean and unit variance, and \( G \) is a nonquadratic function. PROC ICA provides the following choices of \( G \) as options: \( G(u) = -e^{-u^2/2} \) and \( G(u) = \log \cosh(u) \). The fixed-point iteration scheme in the FastICA algorithm is derived as an approximative Newton iteration,

\[
w_i \leftarrow E[X^T g(Xw_i)] - E[g'(Xw_i)]w_i
\]

where \( g \) is the derivative of \( G \).

To prevent the estimated weight vectors \( w_1, \ldots, w_n \) from converging to the same maxima, the projections \( Xw_1, \ldots, Xw_n \) must be decorrelated after every iteration. PROC ICA implements two methods to achieve
Chapter 8: The ICA Procedure

discussion: the deflationary decorrelation method and the symmetric decorrelation method. The deflationary
approach achieves decorrelation on the basis of the Gram-Schmidt orthogonalization method, which means
that the independent components are computed sequentially:

$$w_i \leftarrow w_i - \sum_{j=1}^{i-1} (w_j w_j^T) w_i$$

A problem with the deflationary decorrelation approach is the accumulation of numeric errors.
The symmetric approach estimates the independent components simultaneously by performing symmetric
orthogonalization of the matrix $W = (w_1, \ldots, w_n)$. This is accomplished by the classic method that involves
matrix square roots:

$$W \leftarrow W(W^T W)^{-\frac{1}{2}}$$

The inverse square root is obtained from the eigenvalue decomposition of $W^T W = E D E^T$ as
$(W^T W)^{-1/2} = E D^{-1/2} E^T$, where $E$ is an orthogonal matrix whose entries are the eigenvectors of $W^T W$
and $D$ is a diagonal matrix whose entries are the eigenvalues of $W^T W$.

In the independent component analysis model, the variances and the order of the independent components
cannot be determined. Each of the independent components is thus assumed to have unit variance in the
computation because they are random variables. However, the sign is still indeterminable. By default, PROC
ICA calculates the independent component scores by using the scaled input data and the estimated demixing
matrix. You can use the NOSCALE option in the PROC ICA statement to suppress scaling; PROC ICA then
calculates the independent component scores by using the original input data and the estimated demixing
matrix.

Independent component analysis is closely related to projection pursuit, a technique that usually attemps to
find the most non-Gaussian projections of multidimensional data. If the generative model holds for the data,
optimizing the non-Gaussianity measures produces independent components; if the model is absent for the
data, independent component analysis produces the projection pursuit directions.

Whitening and Dimension Reduction

Whitening can greatly simplify the complexity of the problem of independent component analysis by reducing
the number of parameters to be estimated. A zero-mean random vector is said to be white if its elements are
uncorrelated and have unit variances. A synonym for white is sphered.

By default, the input data variables are centered and scaled to have mean 0 and standard deviation 1. The
centered and scaled data matrix $X$ is then transformed linearly to become a new data matrix $\tilde{X}$, which is
white. In other words, the covariance matrix of $\tilde{X}$ equals the identity matrix: $E[\tilde{X}^T \tilde{X}] = I$. PROC ICA
performs the whitening transformation by using the eigenvalue decomposition of the covariance matrix
$E[X^T X] = E D E^T$, where $E$ is the orthogonal matrix of its eigenvectors and $D$ is the diagonal matrix of
its eigenvalues. Thus the whitening transformation matrix is given by $E D^{-1/2}$, and whitening is done by
$\tilde{X} = X E D^{-1/2}$. 
It can also be useful to reduce the dimensionality of the input data before extracting the independent components, which can usually reduce noise and prevent overlearning. PROC ICA performs dimension reduction and the whitening transformation at the same time by looking at the eigenvalues of $E[X^T X]$ and discarding those that are too small. You can use the EIGTHRESH= option in the PROC ICA statement to specify a threshold for the proportion of variance explained by the eigenvalues. An eigenvalue is discarded if the proportion of variance that it explains is less than the threshold.

When you use the EIGTHRESH= option for dimension reduction, you should suppress scaling by using the NOSCALE option in the PROC ICA statement. Scaling serves to place all dimensions of the space on an equal footing relative to their variation in the data. Performing dimension reduction with scaling poses the risk of dropping eigenvalues for the dimensions of the space that are spanned by the independent components but retaining those dimensions that are filled by noise.

### Missing Values

Observations that have missing values for any variable in the VAR statement are omitted from the analysis and are given missing values for independent component values in the OUT= data table that is specified in the OUTPUT statement.

### Displayed Output

The following sections describe the output that PROC ICA produces. The output is organized into various tables, which are discussed in their order of appearance.

#### Model Information

The “Model Information” table displays basic information about the model, including the input data table, the independent component extraction method, the nonquadratic function used in the negentropy approximation, and the eigenvalue proportion threshold.

#### Dimensions

The “Dimensions” table displays information about the number of VAR statement variables, the number of whitened variables after data whitening and dimension reduction, the number of independent components to extract, and the number of independent components that are actually extracted.

#### Number of Observations

The “Number of Observations” table displays the number of observations that are read from the input data table and the number of observations that are used in the analysis.

#### Centering and Scaling Information

The “Centering and Scaling Information” table displays the centering and scaling information for each analysis variable.
Eigenvalues

The “Eigenvalues” table displays eigenvalues of the covariance matrix that is calculated from the input data, along with the difference between successive eigenvalues, the proportion of variance explained by each eigenvalue, and the cumulative proportion of variance explained.

Whitening Transformation Matrix

The “Whitening Transformation Matrix” table displays the whitening transformation matrix, which is used to perform whitening and dimension reduction on the input data.

Dewhitenening Transformation Matrix

The “Dewhitenening Transformation Matrix” table displays the dewhitenening transformation matrix, which is the pseudoinverse of the whitening matrix.

Demixing Matrix

The “Demixing Matrix” table displays the demixing matrix, which is used to extract the independent components from the input data.

Mixing Matrix

The “Mixing Matrix” table displays the mixing matrix, which is the pseudoinverse of the demixing matrix.

ODS Table Names

Each table that the ICA procedure creates has a name associated with it. You must use this name to refer to the table when you use the DISPLAY statement, the DISPLAYOUT statement, or the Output Delivery System (ODS) statements. These names are listed in Table 8.2.

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Required Statement / Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>CenScaleInfo</td>
<td>Centering and scaling information</td>
<td>Default output</td>
</tr>
<tr>
<td>Demixing</td>
<td>Demixing matrix</td>
<td>Default output</td>
</tr>
<tr>
<td>Dewhitenening</td>
<td>Dewhitenening transformation matrix</td>
<td>Default output</td>
</tr>
<tr>
<td>Dimensions</td>
<td>Model dimensions</td>
<td>Default output</td>
</tr>
<tr>
<td>Eigenvalues</td>
<td>Eigenvalues</td>
<td>Default output</td>
</tr>
<tr>
<td>Mixing</td>
<td>Mixing matrix</td>
<td>Default output</td>
</tr>
<tr>
<td>ModelInfo</td>
<td>Model information</td>
<td>Default output</td>
</tr>
<tr>
<td>NObs</td>
<td>Number of observations read and used</td>
<td>Default output</td>
</tr>
<tr>
<td>OutputCasTables</td>
<td>A special ODS table that has information about</td>
<td>DISPLAYOUT statement or</td>
</tr>
<tr>
<td></td>
<td>all the CAS tables that are created during a CAS</td>
<td>OUTPUT statement</td>
</tr>
<tr>
<td></td>
<td>action execution</td>
<td></td>
</tr>
<tr>
<td>Whitening</td>
<td>Whitening transformation matrix</td>
<td>Default output</td>
</tr>
<tr>
<td>Timing</td>
<td>Absolute and relative times of tasks that are</td>
<td>Default output</td>
</tr>
<tr>
<td></td>
<td>performed by the procedure</td>
<td></td>
</tr>
</tbody>
</table>
Examples: ICA Procedure

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

---

Example 8.1: Extracting Independent Components with Dimension Reduction

This example demonstrates that applying dimension reduction in PROC ICA provides a better estimate of the independent components by using simulated data, compared to extracting the independent components without reducing the dimension of the data. The data that this example uses are from the Getting Started section, and an extra noise signal (x4) is added to the observed signal mixtures. The following DATA step generates the data:

```sas
data mycas.ex1data;
  keep t x:;
  array S[200,3]; /* S: source signals */
  array A[3,3];   /* A: mixing matrix */
  array x[4] x1-x4; /* X: observed signals */

  N = 200;

  do i = 1 to 3;
    do j = 1 to 3;
      A[i,j] = 0.7*uniform(12345);
    end;
  end;

  do i = 1 to N;
    S[i,1] = cos(i/3);
    S[i,2] = 0.4*((mod(i,23)-11)/7)**5;
    S[i,3] = ((mod(i,29)-7)/11)-0.7;
  end;

  do i = 1 to N;
    t = i;
    do j = 1 to 3;
      x[j] = 0;
      do k = 1 to 3;
        x[j] = x[j] + S[i,k]*A[k,j];
      end;
    end;
    x[4] = 0.1*uniform(67890);
  output;
end;
run;
```
These statements assume that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

The following statements extract the independent components by using dimension reduction and output the computed independent components to an output data table:

```plaintext
proc ica data=mycas.ex1data eigthresh=0.004 noscale seed=345;
  var x1-x4;
  output out=mycas.scores1 component=c copyvar=t;
run;
```

Output 8.1.1 displays the PROC ICA output. The “Model Information” table shows that the eigenvalue proportion threshold is set to 0.004. The “Dimensions” table indicates that there are four variables to be analyzed and four independent components to be computed. It also shows that three whitened variables are generated by using whitening and three independent components are actually extracted, thanks to the use of dimension reduction.

**Output 8.1.1** Results of Independent Component Analysis with Dimension Reduction

### The ICA Procedure

<table>
<thead>
<tr>
<th>Model Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source</td>
</tr>
<tr>
<td>Component Extraction Method</td>
</tr>
<tr>
<td>Negentropy Approximation Function</td>
</tr>
<tr>
<td>Eigenvalue Proportion Threshold</td>
</tr>
<tr>
<td>Random Number Seed</td>
</tr>
</tbody>
</table>

### Dimensions

| Number of Variables       | 4 |
| Number of Whitened Variables | 3 |
| Number of Independent Components | 4 |
| Number of Independent Components Extracted | 3 |

### Centering and Scaling Information

<table>
<thead>
<tr>
<th>Variable</th>
<th>Subtracted off</th>
<th>Divided by</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>-0.01231</td>
<td>1.00000</td>
</tr>
<tr>
<td>x2</td>
<td>-0.04632</td>
<td>1.00000</td>
</tr>
<tr>
<td>x3</td>
<td>-0.05567</td>
<td>1.00000</td>
</tr>
<tr>
<td>x4</td>
<td>0.05224</td>
<td>1.00000</td>
</tr>
</tbody>
</table>

### Eigenvalues

<table>
<thead>
<tr>
<th>Eigenvalue</th>
<th>Difference</th>
<th>Proportion</th>
<th>Cumulative</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.129934</td>
<td>1.045020</td>
<td>0.9251</td>
</tr>
<tr>
<td>2</td>
<td>0.084915</td>
<td>0.079040</td>
<td>0.0695</td>
</tr>
<tr>
<td>3</td>
<td>0.005874</td>
<td>0.005123</td>
<td>0.0048</td>
</tr>
<tr>
<td>4</td>
<td>0.000752</td>
<td>0.0006</td>
<td>1.0000</td>
</tr>
</tbody>
</table>
Example 8.1: Extracting Independent Components with Dimension Reduction

Output 8.1.1 continued

Whitening Transformation Matrix

<table>
<thead>
<tr>
<th>Variable</th>
<th>White1</th>
<th>White2</th>
<th>White3</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>-0.26624</td>
<td>1.12655</td>
<td>11.73306</td>
</tr>
<tr>
<td>x2</td>
<td>-0.44684</td>
<td>-2.96560</td>
<td>2.16685</td>
</tr>
<tr>
<td>x3</td>
<td>-0.78387</td>
<td>1.30782</td>
<td>-5.22151</td>
</tr>
<tr>
<td>x4</td>
<td>-0.00124</td>
<td>0.04711</td>
<td>0.78387</td>
</tr>
</tbody>
</table>

Dewhitenign Transformation Matrix

<table>
<thead>
<tr>
<th>Whitened Variable</th>
<th>x1</th>
<th>x2</th>
<th>x3</th>
<th>x4</th>
</tr>
</thead>
<tbody>
<tr>
<td>White1</td>
<td>-0.30083</td>
<td>-0.50490</td>
<td>-0.88572</td>
<td>-0.00140</td>
</tr>
<tr>
<td>White2</td>
<td>0.09566</td>
<td>-0.25182</td>
<td>0.11105</td>
<td>0.00400</td>
</tr>
<tr>
<td>White3</td>
<td>0.06892</td>
<td>0.01273</td>
<td>-0.03067</td>
<td>0.00460</td>
</tr>
</tbody>
</table>

Demixing Matrix

<table>
<thead>
<tr>
<th>Variable</th>
<th>Comp1</th>
<th>Comp2</th>
<th>Comp3</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>0.96285</td>
<td>7.75333</td>
<td>8.82969</td>
</tr>
<tr>
<td>x2</td>
<td>2.08133</td>
<td>-0.76730</td>
<td>2.96125</td>
</tr>
<tr>
<td>x3</td>
<td>-2.31137</td>
<td>-2.73321</td>
<td>-4.09587</td>
</tr>
<tr>
<td>x4</td>
<td>0.09540</td>
<td>0.50802</td>
<td>0.59118</td>
</tr>
</tbody>
</table>

Mixing Matrix

<table>
<thead>
<tr>
<th>Component</th>
<th>x1</th>
<th>x2</th>
<th>x3</th>
<th>x4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comp1</td>
<td>-0.26709</td>
<td>-0.18133</td>
<td>-0.70731</td>
<td>-0.00299</td>
</tr>
<tr>
<td>Comp2</td>
<td>-0.04024</td>
<td>-0.39162</td>
<td>-0.36922</td>
<td>0.00468</td>
</tr>
<tr>
<td>Comp3</td>
<td>0.17732</td>
<td>0.36367</td>
<td>0.40146</td>
<td>0.00289</td>
</tr>
</tbody>
</table>

You can produce a plot of the computed independent components by using the output data table mycas.scores1 and the SGRENDER procedure, as shown in the following statements:

```plaintext
proc template;
    define statgraph ScoresPanel;
    beginGraph;
        layout lattice / rows=3
columns=1
rowgutter=10
columnndatarange=unionall
order=packed;

columnaxes;
columnaxis / label="t";
endcolumnaxes;

layout overlay / yaxisopts=
    (linearopts=(viewmin=-3 viewmax=3 tickvaluelist=(-3 0 3)));
    seriesplot x=t y=c1;
endlayoutplot x=t y=c1;

layout overlay / yaxisopts=
    (linearopts=(viewmin=-2 viewmax=2 tickvaluelist=(-2 0 2)));
```
Chapter 8: The ICA Procedure

Output 8.1.2 displays the computed independent components. The original source signals \((s_1, s_2, \text{ and } s_3)\) in Figure 8.1 are accurately estimated by the computed independent components \((c_1, c_2, \text{ and } c_3)\) up to multiplicative signed scalars.

![Output 8.1.2 Independent Components Computed with Dimension Reduction](image)

The following statements extract the independent components by using all dimensions of the input data \((x_1-x_4)\) and output the computed independent components to an output data table:

```plaintext
proc ica data=mycas.ex1data noscale seed=345;
  var x1-x4;
  output out=mycas.scores2 component=c copyvar=t;
run;
```
Output 8.1.3 displays the computed independent components by using the output data table mycas.scores2 and PROC SGRENDER (code not shown). The original source signals (s1, s2, and s3) in Figure 8.1 are closely estimated by the three computed independent components (c2, c3, and c4) up to multiplicative signed scalars. However, you can clearly see that the estimates are affected by the addition of the noise signal (x4) compared to the independent components that are computed using dimension reduction in Output 8.1.2. Because there are fewer independent components than analysis variables in the input data, the computed component c1 is the projection pursuit direction.

References


# Chapter 9
The KCLUS Procedure

## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview: KCLUS Procedure</td>
<td>452</td>
</tr>
<tr>
<td>PROC KCLUS Features</td>
<td>452</td>
</tr>
<tr>
<td>Using CAS Sessions and CAS Engine Librefs</td>
<td>453</td>
</tr>
<tr>
<td>Getting Started: KCLUS Procedure</td>
<td>453</td>
</tr>
<tr>
<td>Syntax: KCLUS Procedure</td>
<td>457</td>
</tr>
<tr>
<td>PROC KCLUS Statement</td>
<td>457</td>
</tr>
<tr>
<td>CODE Statement</td>
<td>463</td>
</tr>
<tr>
<td>DISPLAY Statement</td>
<td>463</td>
</tr>
<tr>
<td>DISPLAYOUT Statement</td>
<td>464</td>
</tr>
<tr>
<td>FREQ Statement</td>
<td>465</td>
</tr>
<tr>
<td>INPUT Statement</td>
<td>465</td>
</tr>
<tr>
<td>SCORE Statement</td>
<td>465</td>
</tr>
<tr>
<td>Details: KCLUS Procedure</td>
<td>466</td>
</tr>
<tr>
<td>Obtaining the Statistics for Clustering</td>
<td>466</td>
</tr>
<tr>
<td>Missing Values</td>
<td>466</td>
</tr>
<tr>
<td>Initial Seed Selection</td>
<td>467</td>
</tr>
<tr>
<td>Standardization</td>
<td>467</td>
</tr>
<tr>
<td>Finding the Number of Clusters</td>
<td>467</td>
</tr>
<tr>
<td>Clustering Nominal Variables</td>
<td>468</td>
</tr>
<tr>
<td>Dissimilarity Measures</td>
<td>469</td>
</tr>
<tr>
<td>Computing Cluster Centers</td>
<td>470</td>
</tr>
<tr>
<td>Clustering Both Interval and Nominal Variables</td>
<td>470</td>
</tr>
<tr>
<td>Mixed Distance Measure</td>
<td>470</td>
</tr>
<tr>
<td>Weight of Nominal Distance</td>
<td>470</td>
</tr>
<tr>
<td>Score Output for Clustering Both Interval and Nominal Variables</td>
<td>470</td>
</tr>
<tr>
<td>Displayed Output</td>
<td>471</td>
</tr>
<tr>
<td>Number of Observations</td>
<td>471</td>
</tr>
<tr>
<td>Model Information</td>
<td>471</td>
</tr>
<tr>
<td>Cluster Summary</td>
<td>471</td>
</tr>
<tr>
<td>Iteration History</td>
<td>472</td>
</tr>
<tr>
<td>Descriptive Statistics</td>
<td>472</td>
</tr>
<tr>
<td>Within-Cluster Statistics</td>
<td>473</td>
</tr>
<tr>
<td>Cluster Summary for Nominal Variables</td>
<td>473</td>
</tr>
<tr>
<td>Frequencies for Nominal Variables</td>
<td>473</td>
</tr>
<tr>
<td>Cluster Summary for Mixed Variables</td>
<td>473</td>
</tr>
<tr>
<td>Standardization</td>
<td>474</td>
</tr>
</tbody>
</table>
Overview: KCLUS Procedure

The KCLUS procedure performs clustering (a common step in data exploration) in SAS Viya.

You can use the KCLUS procedure to read and write data in distributed form, and to perform clustering and scoring in parallel by making full use of multicore computers or distributed computing environments.

The KCLUS procedure performs a cluster analysis on the basis of distances that are computed from quantitative or qualitative variables (or both). The observations are divided into clusters such that every observation belongs to one and only one cluster.

The KCLUS procedure uses the \( k \)-means algorithm for clustering interval input variables, uses the \( k \)-modes algorithm for clustering nominal input variables, and uses \( k \)-prototypes algorithm for clustering mixed input that contains both interval and nominal variables.

The KCLUS procedure accomplishes the clustering by updating the cluster centroids and the cluster membership of the data iteratively until the convergence criterion (for example, the least squares criterion for the Euclidean distance in \( k \)-means clustering) is satisfied or until the maximum iteration number is reached.

PROC KCLUS produces brief summaries of the cluster analysis in two output data tables:

- The OUT= data table is produced by the SCORE statement. This data table contains the cluster membership and distance variables for each observation in the input data table. It can be used for more detailed examination of the clusters.
- The OUTSTAT= data table is produced by the PROC KCLUS statement. This data table can be used for more detailed examination of between-cluster statistics.

PROC KCLUS Features

PROC KCLUS enables you to use parallel execution for clustering in a distributed computing environment. The following list summarizes the basic features of PROC KCLUS:
• can execute clustering in parallel
• is highly multithreaded
• uses the $k$-means method to cluster interval input variables, uses the $k$-modes method to cluster nominal input variables, and uses $k$-prototypes method to cluster mixed input variables
• provides a new technique called the aligned box criterion (ABC) for estimating the number of clusters in the data table

Using CAS Sessions and CAS Engine Librefs

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:

    proc options option=(CASHOST CASPORT);
    run;

In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:

    cas mysess;
    libname mycas cas sessref=mysess;

The CAS statement creates the CAS session named mysess, and the LIBNAME statement creates the mycas CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the CASHOST and CASPORT of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the mysess session, you can terminate it by using the TERMINATE option in the CAS statement as follows:

    cas mysess terminate;

For more information about the CAS and LIBNAME statements, see the section “Introduction to Shared Concepts” on page 10 in Chapter 3, “Shared Concepts.”

Getting Started: KCLUS Procedure

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table
name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

This example shows how to use the KCLUS procedure to compute clusters of observations in a CAS table.

Suppose you want to group the observations in the input CAS table mycas.inpData, in which the variables are raw measures on interval scales.

The following DATA step creates the input data table, mycas.inpData, in your CAS session. This data table contains four variables: the first two variables are the input variables among which \( x \) has missing values, the third variable is the frequency variable, and the last variable is an index variable.

```sas
data mycas.inpData;
  title 'Using PROC KCLUS to Analyze Data';
  drop n;
  id=1;
  do n=1 to 1000;
    x=2*ranor(12345)+20;
    y=4*ranor(12345)+20;
    freq = 1;
    id = id + 1;
    output;
  end;
  do n=1 to 1000;
    x=3*ranor(12345)+10;
    y=5*ranor(12345)+10;
    freq=2;
    id = id + 1;
    output;
  end;
  do n=1 to 700;
    x=10*ranor(12345);
    y=10*ranor(12345);
    freq=1;
    id = id + 1;
    output;
  end;
  do n=1 to 200;
    x=.;
    y=10*ranor(12345);
    freq=1;
    id = id + 1;
    output;
  end;
run;
```

These statements assume that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

The following statements run PROC KCLUS and output the results to ODS tables:

```sas
proc kclus data=mycas.inpData maxclusters=3;
  input x y;
  freq freq;
run;
```
Figure 9.1 shows that the “Number of Observations Used” is less than the “Number of Observations Read”. By default, the KCLUS procedure ignores observations that have missing values, and it does not use them in the analysis. The two additional rows, “Sum of Frequencies Read” and “Sum of Frequencies Used” are displayed when the FREQ statement is specified. They provide information about the frequency values that are read and used.

**Figure 9.1** Number of Observations

*Using PROC KCLUS to Analyze Data*

The KCLUS Procedure

<table>
<thead>
<tr>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Observations Read</td>
<td>2900</td>
</tr>
<tr>
<td>Number of Observations Used</td>
<td>2700</td>
</tr>
<tr>
<td>Sum of Frequencies Read</td>
<td>3900</td>
</tr>
<tr>
<td>Sum of Frequencies Used</td>
<td>3700</td>
</tr>
</tbody>
</table>

Figure 9.2 shows the values of the parameters that are used in clustering. Because the number of clusters is not estimated by default and MAXCLUSTERS=3, three clusters are generated. Figure 9.2 shows the number of clusters and the default values for other options.

**Figure 9.2** Model Information

<table>
<thead>
<tr>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clustering Algorithm</td>
<td>K-means</td>
</tr>
<tr>
<td>Maximum Iterations</td>
<td>10</td>
</tr>
<tr>
<td>Stop Criterion</td>
<td>Cluster Change</td>
</tr>
<tr>
<td>Stop Criterion Value</td>
<td>0</td>
</tr>
<tr>
<td>Clusters</td>
<td>3</td>
</tr>
<tr>
<td>Initialization</td>
<td>Forgy</td>
</tr>
<tr>
<td>Seed</td>
<td>1771582921</td>
</tr>
<tr>
<td>Distance for Interval Variables</td>
<td>Euclidean</td>
</tr>
<tr>
<td>Standardization</td>
<td>None</td>
</tr>
<tr>
<td>Interval Imputation</td>
<td>None</td>
</tr>
</tbody>
</table>

For each cluster, Figure 9.3 shows the number of observations; the maximum, minimum, and average distances from that cluster’s centroid to the observations in that cluster; the sum of squares error; and the standard deviation. Figure 9.3 also displays information about the nearest cluster to that cluster and the distance between their centroids.
Figure 9.3  Cluster Summary

Cluster Summary for Interval Variables
Distance from Cluster Centroid to Observation

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Frequency</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Average</th>
<th>SSE</th>
<th>Standard Deviation</th>
<th>Nearest Cluster</th>
<th>Distance to Nearest Cluster Centroid</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>599</td>
<td>0.4639</td>
<td>32.9356</td>
<td>10.7158</td>
<td>91051.2</td>
<td>12.3290</td>
<td>2</td>
<td>17.1809</td>
</tr>
<tr>
<td>2</td>
<td>2008</td>
<td>0.0450</td>
<td>28.4411</td>
<td>5.0110</td>
<td>70355.9</td>
<td>5.9193</td>
<td>3</td>
<td>13.6580</td>
</tr>
<tr>
<td>3</td>
<td>1093</td>
<td>0.0616</td>
<td>29.1067</td>
<td>4.1611</td>
<td>26545.2</td>
<td>4.9281</td>
<td>2</td>
<td>13.6580</td>
</tr>
</tbody>
</table>

Figure 9.4 shows the sum of squared errors (SSE) for each iteration. If the variables are interval, then the “Iteration History” table displays SSE Change and Stop Criterion columns. The SSE Change column displays the change in within-cluster distances. The Stop Criterion column displays the stopping criterion for each iteration. If the input variables are nominal, then the “Iteration History” table displays Within Distance Change and Stop Criterion columns. If the input variables are both interval and nominal, then the “Iteration History” table also displays Within Distance Change and Stop Criterion columns, but with the distance in the sense of mixed distance with both the interval and nominal parts as detailed in Clustering Both Interval and Nominal Variables.

Figure 9.4  Iteration History

<table>
<thead>
<tr>
<th>Iteration Number</th>
<th>SSE</th>
<th>SSE Change</th>
<th>Stop Criterion</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1016049</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>345086</td>
<td>-670962</td>
<td>21.518519</td>
</tr>
<tr>
<td>2</td>
<td>304959</td>
<td>-40127</td>
<td>8.777778</td>
</tr>
<tr>
<td>3</td>
<td>296499</td>
<td>-8460.125426</td>
<td>6.555556</td>
</tr>
<tr>
<td>4</td>
<td>289527</td>
<td>-6972.200760</td>
<td>7.592593</td>
</tr>
<tr>
<td>5</td>
<td>277944</td>
<td>-11583</td>
<td>11.407407</td>
</tr>
<tr>
<td>6</td>
<td>256627</td>
<td>-21316</td>
<td>13.148148</td>
</tr>
<tr>
<td>7</td>
<td>231532</td>
<td>-25096</td>
<td>12.444444</td>
</tr>
<tr>
<td>8</td>
<td>209285</td>
<td>-22247</td>
<td>8.444444</td>
</tr>
<tr>
<td>9</td>
<td>194448</td>
<td>-14837</td>
<td>4.592593</td>
</tr>
<tr>
<td>10</td>
<td>187952</td>
<td>-6495.562139</td>
<td>3.000000</td>
</tr>
</tbody>
</table>

Figure 9.5 and Figure 9.6 show statistics for each variable in the INPUT statement. Figure 9.5 shows the variable statistics for all the observations in the input data table, and Figure 9.6 shows the variable statistics for the observations that belong to a specific cluster.

Figure 9.5  Descriptive Statistics

<table>
<thead>
<tr>
<th>Variable</th>
<th>Mean</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>11.020648</td>
<td>8.189686</td>
</tr>
<tr>
<td>y</td>
<td>10.246546</td>
<td>9.472050</td>
</tr>
</tbody>
</table>
Figure 9.6 Within-Cluster Statistics

<table>
<thead>
<tr>
<th>Variable</th>
<th>Cluster</th>
<th>Mean</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>1</td>
<td>-0.2661</td>
<td>9.6913</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>9.7857</td>
<td>10.7113</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>19.4749</td>
<td>4.9481</td>
</tr>
<tr>
<td>y</td>
<td>1</td>
<td>-3.7649</td>
<td>8.0808</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>10.1687</td>
<td>11.4546</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>19.7947</td>
<td>6.7337</td>
</tr>
</tbody>
</table>

Syntax: KCLUS Procedure

The following statements are available in the KCLUS procedure:

```
PROC KCLUS <options>;
  CODE <options>;
  DISPLAY <table-list> </options>;
  DISPLAYOUT table-spec-list </options>;
  FREQ variable;
  INPUT variables <LEVEL= NOMINAL | INTERVAL>;
  SCORE OUT=CAS-libref.data-table <options>;
```

The PROC KCLUS statement and an INPUT statement are required. You can specify multiple INPUT statements.

The following sections describe the PROC KCLUS statement and then describe the other statements in alphabetical order.

PROC KCLUS Statement

```
PROC KCLUS <options>;
```

The PROC KCLUS statement invokes the procedure. Table 9.1 summarizes the options available in the PROC KCLUS statement.

Table 9.1 PROC KCLUS Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input and Output Data Set Options</td>
<td></td>
</tr>
<tr>
<td>DATA=</td>
<td>Specifies the input data table</td>
</tr>
<tr>
<td>OUTSTAT(OUTITER)=</td>
<td>Specifies the output data table to contain cluster centroids</td>
</tr>
<tr>
<td>PRINTALLDISTANCES</td>
<td>Outputs the interval distance, nominal distance, and total distance (this option can be used only for mixed input variables)</td>
</tr>
</tbody>
</table>
### Table 9.1 continued

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Clustering Options</strong></td>
<td></td>
</tr>
<tr>
<td>DISTANCE=</td>
<td>Specifies the distance measure for similarity measurement (used for interval input variables)</td>
</tr>
<tr>
<td>DISTANCENOM=</td>
<td>Specifies the distance measure for similarity measurement (used for nominal input variables)</td>
</tr>
<tr>
<td>INIT=</td>
<td>Specifies the method for obtaining the initial estimate of cluster centers</td>
</tr>
<tr>
<td>MAXCLUSTERS=</td>
<td>Specifies the number of clusters</td>
</tr>
<tr>
<td>MAXITER=</td>
<td>Specifies the maximum number of iterations</td>
</tr>
<tr>
<td>SEED=</td>
<td>Specifies the seed that is used for pseudorandom number generation</td>
</tr>
<tr>
<td>STOPCRITERION=</td>
<td>Specifies the stop criterion method to use for convergence</td>
</tr>
<tr>
<td>KPROTOTYPEPARAMS=</td>
<td>Specifies the $k$-prototypes parameters for clustering interval and nominal variables at the same time (this option can be used only for mixed input variables)</td>
</tr>
<tr>
<td><strong>Data Processing Options</strong></td>
<td></td>
</tr>
<tr>
<td>IMPUTE=</td>
<td>Specifies the imputation method for interval input variables</td>
</tr>
<tr>
<td>IMPUTENOM=</td>
<td>Specifies the imputation method for nominal input variables</td>
</tr>
<tr>
<td>STANDARDIZE=</td>
<td>Specifies the method for standardizing interval input variables</td>
</tr>
<tr>
<td><strong>Number of Clusters Estimation Options</strong></td>
<td></td>
</tr>
<tr>
<td>NOC=</td>
<td>Specifies the method for estimating the number of clusters</td>
</tr>
<tr>
<td><strong>Performance Options</strong></td>
<td></td>
</tr>
<tr>
<td>NTHREADS=</td>
<td>Specifies the number of threads for the computation</td>
</tr>
</tbody>
</table>

You can specify the following options:

**DATA=CAS-libref.data-table**

names the input data table for PROC KCLUS to use. The default is the most recently created data table. *CAS-libref.data-table* is a two-level name, where

*CAS-libref* refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to the data, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about *CAS-libref*, see the section “Using CAS Sessions and CAS Engine Librefs” on page 453.

*data-table* specifies the name of the input data table.

**DISTANCE=EUCLIDEAN | MANHATTAN**

specifies the distance measure for similarity that is used for interval input variables.

You can choose from the following measures:

**EUCLIDEAN** calculates the Euclidean distance.

**MANHATTAN** calculates the Manhattan distance.

By default, DISTANCE=EUCLIDEAN.
**DISTANCE=** BINARY | GLOBALFREQ | RELATIVEFREQ
specifies the distance measure for similarity that is used for nominal input variables.

You can choose from the following measures:

- **BINARY** calculates a simple matching distance.
- **GLOBALFREQ** calculates the distance based on the frequency of levels of nominal input variables in the input data table.
- **RELATIVEFREQ** calculates the distance based on the frequency of levels of nominal input variables in each cluster.

By default, **DISTANCE=BINARY**. For more information about these distance functions, see the section “Clustering Nominal Variables” on page 468.

**INIT=** FORGY | RAND
specifies the method for obtaining the initial estimate of cluster centers.

You can choose from the following methods:

- **FORGY** selects the initial cluster centers randomly from observations.
- **RAND** assigns all observations randomly to one of the clusters.

By default, **INIT=FORGY**.

**IMPUTE=** MEAN | NONE
specifies the imputation method to be used when the *INPUT* statement specifies interval variables.

You can choose from the following methods:

- **MEAN** imputes missing values of variables in observations by using the mean value for that variable.
- **NONE** does not do any imputation and excludes observations that have missing values.

If **IMPUTE=NONE** and the number of observations that have nonmissing values is less than the value of the **MAXCLUSTERS=** option, then the number of clusters is set to the number of observations that have nonmissing values.

By default, **IMPUTE=NONE**.

**IMPUTENOM=** MODE | NONE
specifies the imputation method to be used when the *INPUT* statement specifies nominal variables.

You can choose from the following methods:

- **MODE** imputes missing values of variables in observations by using the mode value for that variable.
- **NONE** does not do any imputation and excludes observations that have any missing values.

If **IMPUTENOM=NONE** and the number of observations that have nonmissing values is less than the value of the **MAXCLUSTERS=** option, then the number of clusters is set to the number of observations that have nonmissing values.

By default, **IMPUTENOM=NONE**.
**MAXCLUSTERS=number**

specifies either the number of clusters to be used (if the NOC= option is not specified or NOC=NONE) or the maximum number of clusters to be searched (if NOC=ABC).

By default, MAXCLUSTERS=6.

**MAXITER=number**

specifies the maximum number of iterations for the algorithm to perform. In each iteration of the \( k \)-means or \( k \)-modes clustering method, each observation is assigned to the nearest cluster centroid, and the centroids are recomputed.

By default, MAXITER=10.

**NOC= NONE | ABC < suboptions >**

specifies the method for estimating the number of clusters.

You can specify the following values:

**ABC < suboptions >** estimates the number of clusters by using the aligned box criterion (ABC) method. For more information about this method, see the section “Finding the Number of Clusters” on page 467.

You can specify one or more of the following **suboptions**:

- **ALIGN=keyword** specifies the method for aligning the reference data based on the input data.
  
  You can specify the following **keywords**:

  - **NONE** generates the reference data from a uniform distribution over the range of values for each subset of the input data table.
  
  - **PCA** generates the reference data from a uniform distribution over a box that is aligned with the principal components of each subset of the input data table.

  By default, ALIGN=NONE.

- **B=number** specifies the number of reference data to be created for each cluster candidate. By default, B=1.

- **CRITERION=keyword** specifies the criterion to be used to estimate the number of clusters that use the statistics obtained by the ABC method.

  You can specify the following **keywords**:

  - **ALL** uses all the following options and selects the number of clusters based on which number of clusters is chosen the most often. If each option selects a different number of clusters, then the number selected by GLOBALPEAK is used.
**FIRSTMAXWITHSTD** uses the smallest $k$ such that the gap value for that $k$ is greater than the one-standard-error adjusted gap value for $k+1$.

**FIRSTPEAK** uses the first peak value among the peak values in gap statistics.

**GLOBALPEAK** uses the maximum peak value among all the peak values in gap statistics.

By default, CRITERION=GLOBALPEAK.

**MINCLUSTERS=number** specifies the minimum number of clusters for searching for the best number of clusters. By default, MINCLUSTERS=2.

**NONE** does not estimate the number of clusters and uses the value specified in MAXCLUSTERS= option.

By default, NOC=NONE.

**NTHREADS=number-of-threads** specifies the number of threads that are used in the computation. The default value is the number of CPUs available in the machine.

**OUTSTAT < (OUTITER) >= data-table** creates the output data table that contains the cluster centroids for each cluster.

This data table includes the iteration number as _ITERATION_, the cluster ID as _CLUSTER_ID_ and the cluster centroids, which consist of the variables that are specified in the INPUT statement.

If you specify STANDARDIZE=RANGE or STANDARDIZE=STD, the data table also contains the standardized values of the cluster centroids, which are displayed by adding the S_ prefix to the variables in the INPUT statement.

You can also specify the following suboption:

**OUTITER** also outputs the cluster centroids to the OUTSTAT= data table for each iteration.

**SEED=number** specifies an integer to be used to start the pseudorandom number generator. If you do not specify a seed or if you specify a value less than or equal to 0, the seed is generated from reading the time of day from the computer’s clock.

**STANDARDIZE=NONE | RANGE | STD** specifies the method for standardizing the interval input variables. PROC KCLUS uses the location and scale measures specific to the method. You can specify the following values:

**NONE** does not standardize and uses the actual values for the input variables.

**RANGE** standardizes the input variables by using the range method. PROC KCLUS uses the minimum as the location and the range as the scale.
STD standardizes the input variables by using the traditional standardization method. PROC KCLUS uses the mean as the location and the standard deviation as the scale.

By default, STANDARDIZE=NONE.

**STOPCRITERION=convergence_method<(VALUE=number)>**
specifies the method to use for convergence. If you do not specify this option, the algorithm stops after it reaches the maximum number of iterations (which is specified in MAXITER= option).

You can specify the following convergence methods:

**CLUSTER_CHANGE<(VALUE=number)>**
uses the percentile of observations that do not change their cluster membership for that iteration. The optional VALUE= suboption enables you to specify the percentile of observations.

By default or if VALUE=0, PROC KCLUS runs until the cluster centroids no longer change.

**WCSD_CHANGE<(VALUE=number)>**
uses the within-cluster distance change as a convergence criterion. The optional VALUE= suboption enables you to specify the change in SSE for the k-means algorithm or specify the sum of within-cluster distances for the k-modes algorithm and the k-prototypes algorithm. For the k-prototypes algorithm, the distance is a mixed distance that includes both the interval and nominal parts.

**KPROTOTYPEPARAMS=gamma_method**
specifies the method for obtaining the γ parameter in the k-prototypes clustering algorithm (Huang 1997) for clustering mixed input data that contain both interval and nominal variables.

You can specify either of the following gamma methods:

**USERGAMMA(VALUE=number)**
specifies the γ parameter value (in the VALUE= suboption) for the k-prototypes clustering algorithm. The value of γ should be an appropriate positive number that balances the difference between the scales of the nominal and interval input variables.

**AUTOGAMMA**
estimates the γ parameter value for the k-prototypes clustering algorithm. You can use this option when you do not have a clear idea of how large the γ value would be. When you specify the AUTOGAMMA option, the value of γ is inferred to be the average of the standard deviations of the interval input variables.

By default, KPROTOTYPEPARAMS=USERGAMMA(VALUE=0.5).

**PRINTALLDISTANCES**
prints the distance that is contributed by the interval variables (as _DISTANCEINT_) and the distance that is contributed by the nominal variables (as _DISTANCENom_) in the mixed distance that is used by the k-prototypes algorithm to the output table that is specified in the SCORE statement. For more information, see the section “Score Output for Clustering Both Interval and Nominal Variables” on page 470.
CODE Statement

```sas
CODE < options > ;
```

The CODE statement generates the SAS DATA step code that mimics the computations that are done by the SCORE statement.

You can specify the following `options`:

- **FILE=filename**
  - specifies the filename of the file to write the SAS score code to.

- **LABELID=number**
  - specifies a number used to construct array names and statement labels in the generated code. You can specify a value in the range 0 to 1024; by default, LABELID=0.

DISPLAY Statement

```sas
DISPLAY < table-list > < / options > ;
```

The DISPLAY statement enables you to specify a list of display tables to display or exclude. This statement is similar to the ODS SELECT, ODS EXCLUDE, and ODS TRACE statements. However, the DISPLAY statement can improve performance when a large number of tables could be generated (such as in BY-group processing). The procedure processes the DISPLAY statement on a CAS server and thus sends only a subset of ODS tables to the SAS client. Because ODS statements are processed on a SAS client, first all the generated display tables are sent to the client, and then the client creates a subset.

If you use both DISPLAY and ODS statements together, the DISPLAY statement takes precedence over the ODS statements. Note that the ODS EXCLUDE statement processes tables that are sent to the client after they have been filtered by the DISPLAY statement. In some cases, it might appear that the ODS EXCLUDE statement is taking precedence because it can further filter the tables. For more information about ODS, see SAS Output Delivery System: Procedures Guide.

You can specify the `table-list` as a list of table names, paths, partial pathnames, and regular expressions.

The table names that you can specify are listed in the section “ODS Table Names” on page 475. A path is a table name that is prefixed with dot-separated grouping information. For example, a SelectionSummary table that a procedure produces during a selection routine might have the path `Bygroup1.Summary.SelectionSummary`. A partial pathname does not include all groups; for example, `Selection.Summary.SelectionSummary` and `Summary.SelectionSummary` are partial pathnames for `Bygroup1.Summary.SelectionSummary`.

When you specify a table name or partial pathname, all display tables whose paths end in the specified name are selected for display or exclusion. For example, both `SelectionSummary` and `Summary.SelectionSummary` select `Bygroup1.Summary.SelectionSummary`.

A regular expression is enclosed in forward slashes (/). For example, specifying “/tions/” selects all pathnames that contain the substring “tions”; in particular, the `Bygroup1.Summary.SelectionSummary` table is selected. Specifying “!/tions/” selects all pathnames that do not contain the substring “tions”; in particular, the `Bygroup1.Summary.SelectionSummary` table is not selected.

You can specify the following `options` after a slash (/):
CASESENSITIVE
performs a case-sensitive comparison of table names in the table-list to display table names when tables are subsetted for display. To preserve case, you must enclose table names in the table-list in quotation marks.

EXCLUDE
displays all display tables except those that you specify in the table-list.

EXCLUDEALL
suppresses display of all tables. This option takes precedence over the other options.

TRACE
displays the display table names, labels, and paths.

DISPLAYOUT Statement

DISPLAYOUT table-spec-list < / options > ;

The DISPLAYOUT statement enables you to create CAS output tables from your displayed output. This statement is similar to the ODS OUTPUT statement. For more information about ODS, see SAS Output Delivery System: Procedures Guide.

The table-spec-list specifies a list of CAS output tables to create. Each entry in the list has either a key=value format or a key format:

key=value specifies key as the ODS table name, path, or partial pathname, and specifies value as the CAS output table name.

key specifies key as the ODS table name and also as the CAS output table name.

The ODS table names that you can specify are listed in the section “ODS Table Names” on page 475. You cannot specify the ODS table named OutputCasTables in the table-spec-list.

Table names and partial pathnames are discussed under the DISPLAY statement. The DISPLAYOUT statement does not support regular expressions.

You can specify the following options after a slash (/):

INCLUDEALL
creates output CAS tables for all display tables. The name of the created output CAS table is the same as the corresponding display table name. If you specify this option, the table-spec-list specification is ignored.

NOREPLACE
does not replace any existing CAS output table of the same name.

REPEATED
replicates all CAS output tables on all nodes.
FREQ Statement

FREQ variable ;

The variable in the FREQ statement identifies a numeric variable in the data set that contains the frequency of occurrence of each observation. PROC KCLUS treats each observation as if it appears \( f \) times, where \( f \) is the value of the FREQ variable for the observation. If \( f \) is not an integer, it is truncated to an integer. If \( f \) is less than 1 or missing, the observation is not used in the analysis. When the FREQ statement is not specified, each observation is assigned a frequency of 1.

INPUT Statement

INPUT variables < LEVEL= NOMINAL | INTERVAL > ;

The INPUT statement specifies the names of the variables to be used in clustering. It names one or more input variables that use common options. If you want to use different options for different variables, you can specify multiple INPUT statements.

To trigger the \( k \)-prototypes clustering algorithm for clustering mixed input data that contain both interval and nominal input variables, either you can use two INPUT statements with one INPUT statement that includes the LEVEL=INTERVAL option for interval input variables and another INPUT statement that includes the LEVEL=NOMINAL option for nominal input variables, or you can use a single INPUT statement without the LEVEL= option specified. In the latter case, PROC KCLUS detects the types of the input variables and uses the \( k \)-prototypes clustering algorithm if the input variables include both interval and nominal types.

You can include the following option in each INPUT statement:

LEVEL= NOMINAL | INTERVAL

specifies the level of measurement of the variables. For clustering, only interval, binary, and nominal variables are accepted.

By default, LEVEL=INTERVAL for numeric variables, and LEVEL=NOMINAL for categorical character variables.

SCORE Statement

SCORE OUT=CAS-libref.data-table < option > ;

OUTPUT OUT=CAS-libref.data-table < option > ;

The SCORE statement writes the cluster membership information of each observation to the output data table that is specified in the OUT= option. This information includes the variables that are specified in the COPYVARS= option and additional variables, as described in the section “Score Output for Clustering Both Interval and Nominal Variables” on page 470.

You must specify the following option:
names the output data table for PROC KCLUS to use. You must specify this option before any other options. \textit{CAS-libref.data-table} is a two-level name, where \textit{CAS-libref} refers to a collection of information that is defined in the LIBNAME statement and includes the \textit{caslib}, which includes a path to where the data table is to be stored, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about \textit{CAS-libref}, see the section “Using CAS Sessions and CAS Engine Librefs” on page 453. \textit{data-table} specifies the name of the output data table. The output data table contains the scored data. When you specify this option, all variables that are specified in the \texttt{COPYVARS=} option, followed by the \_CLUSTER\_ID\_ variable and the \_DISTANCE\_ variable, are added to the output data table that is specified in this option.

You can also specify the following \textit{option}:

\begin{description}
\item[\texttt{COPYVAR=}variable] \item[\texttt{COPYVARS=}(variables)]
\end{description}

lists one or more variables from the input data table that are transferred to the scored output data table, provided that the output data table produces one or more records per input observation. By default, the KCLUS procedure does not transfer any variables from the input data table to the output data table. The \texttt{COPYVARS=} option accepts numeric and character variables. You can also use \texttt{COPYVARS=}(_\texttt{ALL}_) option to include all the input variables.

\section*{Details: KCLUS Procedure}

\subsection*{Obtaining the Statistics for Clustering}

The KCLUS procedure calculates the following cluster-based statistics (in addition to the variable-based statistics) about the data in the input data table.

\begin{itemize}
\item summary statistics for each cluster (for example, the number of observations)
\item summary statistics for the variables in each cluster (for example, the mean value of an interval input variable in a cluster)
\end{itemize}

You can use these statistics in addition to the output data tables to further analyze the clustering results.

\subsection*{Missing Values}

Observations that have missing values affect the clustering results because the distance measures that are computed for these observations do not reflect the actual values. To solve this problem, you can use the
**Initial Seed Selection**

When INIT=FORGY in the PROC KCLUS statement (the default), random observations from the input data table are selected as initial cluster centroids by default. The initial aim of PROC KCLUS is to select the observations that have no missing values.

You can change the observations that are selected from the input data table by using the SEED= option. You can use this option to set the seed value for the random number generator, which is used for generating random observation indices.

Because the random number generator creates indices for the observations, the initial cluster centroids will be different depending on where and how you run the procedure and where the input data reside. When the data are distributed or the computation is done in parallel, the indices of the observations will change depending on the number of nodes and number of threads that are used. Thus, you might get different initial cluster centroids.

**Standardization**

Because variables that have large variances tend to affect the distance measure more than variables that have small variances, it is recommended that you standardize the variables before clustering the observations. The STANDARDIZE= option in the PROC KCLUS statement provides a convenient way to standardize the INPUT variables. This option standardizes all the variables by using the method that is specified in that option and adds the standardized values to the output data tables.

**Finding the Number of Clusters**

To estimate the number of clusters (NOC), you can specify NOC=ABC in the PROC KCLUS statement. This option uses the aligned box criterion (ABC) method to estimate an interim number of clusters and then runs the $k$-means clustering method to produce the final clusters. If the NOC= option is not specified, PROC KCLUS uses the MAXCLUSTERS= option in the PROC KCLUS statement to set the number of the clusters, and then runs the $k$-means clustering method to produce the final clusters.

The ABC method estimates the number of clusters for well-separated clusters. Similar to the gap statistics method that is presented in Tibshirani, Walther, and Hastie (2001), the ABC method uses within-cluster dispersion from the results of clustering as an error measure, making the ABC method independent of the
method that is used for clustering. In order to estimate the number of clusters, the ABC method compares the change in the error measure with the change that is expected under an appropriate reference null distribution.

The cubic clustering criterion (CCC), proposed in Sarle (1983), is based on the assumption that a uniform distribution on a hyperrectangle is divided into clusters that are shaped approximately like hypercubes. In recent and widely accepted work, Tibshirani, Walther, and Hastie (2001) propose a gap statistics method that uses Monte Carlo simulation (instead of a heuristic) to generate a hypercube reference distribution. The ABC method improves on the CCC and gap statistics methods by generating more restricted and aligned reference distributions that consider different features in the input data table.

Although the ABC method calculates the reference distribution differently than the other methods do, the computation of the error measure is similar to the method presented in Tibshirani, Walther, and Hastie (2001). The error measure—also called the gap and represented as $\text{Gap}(k)$—is obtained by subtracting the logarithm of the within-cluster sum of squares error from the logarithm of its expectation for clustering solutions over a range of possible $k$ values. The simulation-adjusted standard deviation of the reference distribution—represented as $s(k)$—is obtained by using the standard deviation of the logarithm of the within-cluster sum of squares errors for reference data. The one-standard-error adjusted gap is obtained by subtracting $s(k)$ from the gap. For more information, see Tibshirani, Walther, and Hastie (2001).

A notable feature of the ABC algorithm is that it also applies to clustering both interval and nominal inputs. In the $k$-prototypes clustering algorithm, the ABC algorithm is performed with a mixed distance to which both the interval and nominal inputs contribute.

The number of clusters is estimated by considering the $\text{Gap}(k)$ and $s(k)$ values for each cluster candidate $k$. You can choose one of four methods for estimating the number of clusters by specifying the CRITERION= suboption in the NOC=ABC option in the PROC KCLUS statement:

- The FIRSTPEAK option selects the first peak value among all the values in $\text{Gap}(k)$.
- The GLOBALPEAK option selects the peak value that has the maximum value among the peak values in $\text{Gap}(k)$.
- The FIRSTMAXWITHSTD option considers the standard deviation in each cluster in addition to the values in $\text{Gap}(k)$. It considers the values in the one-standard-error adjusted gap to estimate the number of clusters. It selects the smallest $k$ such that the gap value for that $k$ is greater than the one-standard-error adjusted gap value for $k+1$.
- The ALL option uses all the preceding options and selects the number of clusters based on which number of clusters is chosen the most often. If each option selects a different number, then the number selected by the GLOBALPEAK option is used.

---

**Clustering Nominal Variables**

The $k$-means algorithm works only with interval inputs. One way to apply the $k$-means algorithm to nominal data is to use data transformation methods to transform the nominal data into a new feature space. However, this approach can be very inefficient, and it does not produce good results. For clustering nominal inputs, the KCLUS procedure implements the $k$-modes clustering algorithm, which extends the $k$-means algorithm by using different dissimilarity measures and a different method for computing cluster centers (Huang 1997).
Dissimilarity Measures

In the $k$-modes clustering algorithm, distance measures depend on the level of nominal variables. Let $X$ and $Y$ be two observations, which are described by $F$ nominal variables. Each nominal variable $j$ in $F$ has a different number of distinct values, which are called levels (or sometimes called categories).

The dissimilarity measure between observations $X$ and $Y$ is

$$d(X,Y) = \sum_{j=1}^{F} \delta(x_j, y_j)$$

where $\delta(x_j, y_j)$ is the dissimilarity measure between two nominal variables $j$.

PROC KCLUS supports the following dissimilarity measures:

- The simple matching dissimilarity measure calculates the total number of mismatches of the corresponding variables of two observations. If the number of mismatches is small, then the observations are similar to each other. The dissimilarity measure between two nominal variables $j$ is

$$\delta(x_j, y_j) = \begin{cases} 
0 & \text{if } x_j = y_j \\
1 & \text{if } x_j \neq y_j 
\end{cases}$$

This algorithm is very efficient. However, it might lead to clusters that have weak intrasimilarity, depending on the number of levels and the number of observations that have these levels. For more information, see Huang (1997). Specify DISTANCENOM=BINARY in the PROC KCLUS statement to use this distance measure.

- The global frequency-based dissimilarity measure takes into account the frequencies of levels of each variable in the input data table. The dissimilarity measure between two nominal variables $j$ is

$$\delta(x_j, y_j) = \begin{cases} 
0 & \text{if } x_j = y_j \\
\frac{(n_{x_j}+n_{y_j})}{n_{x_j} \cdot n_{y_j}} & \text{if } x_j \neq y_j 
\end{cases}$$

where $n_{x_j}$ is the number of observations that have level $x_j$ of variable $j$ in the input data table and $n_{y_j}$ is the number of observations that have level $y_j$ of variable $j$ in the input data table. You can use this measure to find underrepresented clusters because it assigns more importance to rare categories than to frequent ones. For more information, see Huang (1997). Specify DISTANCENOM=GLOBALFREQ in the PROC KCLUS statement to use this distance measure.

- The cluster frequency-based dissimilarity measure takes into account the relative frequencies of levels of each variable in each cluster. The dissimilarity measure between two nominal variables $j$ is

$$\delta(x_j, y_j) = \begin{cases} 
1 - \frac{n_{x_j}^c}{n_c} & \text{if } x_j = y_j \\
1 & \text{if } x_j \neq y_j 
\end{cases}$$

where $n_c$ is the number of observations in cluster $c$ and $n_{x_j}^c$ is the number of objects that have level $x_j$ of variable $j$ in cluster $c$.

This distance measure enables the algorithm to assign less similar observations to the clusters that have weak intrasimilarity. Thus, the generated clusters have strong intrasimilarities. For more information, see Ng et al. (2007). Specify DISTANCENOM=RELATIVEFREQ in the PROC KCLUS statement to use this distance measure.
Computing Cluster Centers

The $k$-modes algorithm uses modes instead of means as cluster centers. The KCLUS procedure uses a frequency-based method to update the modes after each iteration in order to minimize the clustering cost function. Cluster centers are updated using the same approach for all the distance measures.

Clustering Both Interval and Nominal Variables

For clustering inputs that include only interval or only nominal variables, the KCLUS procedure implements the $k$-means or $k$-modes clustering algorithm, respectively. For clustering inputs that include both interval and nominal variables, the KCLUS procedure implements the $k$-prototypes clustering algorithm (Huang 1997). The $k$-prototypes clustering algorithm uses a mixed distance that combines the distance contributed by the interval inputs and the distance contributed by the nominal inputs. This mixed distance is characterized with a weight parameter, $\gamma$, to balance the scales of the interval and nominal distances.

Mixed Distance Measure

The mixed distance measure $d_{mixed}$ is defined as

$$d_{mixed} = d_{int}^2 + \gamma \times d_{nom}$$

where $d_{int}$ is the distance of the interval input variables and $d_{nom}$ is the distance of the nominal input variables. The interval distance measure, $d_{int}$, is specified in the DISTANCE= option, and the nominal distance measure, $d_{nom}$, is specified in the DISTANCENOM= option.

Weight of Nominal Distance

A weight parameter, $\gamma$, is used to balance any possible difference between the scales of the nominal variables and the interval variables. You can specify this value in the KPROTOTYPEPARAMS= option. By default, the weight is 0.5.

Score Output for Clustering Both Interval and Nominal Variables

The KCLUS procedure writes the cluster membership information to the output table that is specified in the OUT= option in the SCORE statement. This information always includes the variables that are specified in the COPYVARS= option and the following variables:

- _CLUSTER_ID_ (the ID of the closest cluster)
- _DISTANCE_ (the distance between the observation and the centroid of that cluster)

In addition, cluster membership information can include one or more of the following variables:

- _STANDARDIZED_DISTANCE_ (the distance between the standardized values of the observation and the standardized values of cluster centroid). This variable is included if you specify STANDARDIZE=RANGE or STANDARDIZE=STD in the PROC KCLUS statement.
• _DISTANCEINT_ (the interval part of the distance between the observation and the centroid of that cluster). This variable is included if the k-prototypes clustering algorithm is used and you specify the PRINTALLDISTANCES option in the PROC KCLUS statement.

• _DISTANCENOM_ (the nominal part of the distance between the observation and the centroid of that cluster). This variable is included if the k-prototypes clustering algorithm is used and you specify the PRINTALLDISTANCES option in the PROC KCLUS statement.

• _DISTANCE_, which is equal to _DISTANCEINT_ \(^2 + \gamma \times _DISTANCENOM_.

• _STANDARDIZED_DISTANCEINT_ (the interval part of the distance between the standardized values of the observation and the standardized values of cluster centroid). This variable is included if the k-prototypes clustering algorithm is used and you specify the both the PRINTALLDISTANCES and the STANDARDIZE= options in the PROC KCLUS statement.

• _STANDARDIZED_DISTANCE_, which is equal to _STANDARDIZED_DISTANCEINT_ \(^2 + \gamma \times _DISTANCENOM_.

### Displayed Output

The KCLUS procedure displays various tables for cluster analysis.

The following sections describe the output in the order of their appearance when the related options are specified.

#### Number of Observations

The “Number of Observations” table displays the number of observations that are read from the input data table and the number of observations that are used in the clustering. The Number of Observations Read column displays the count of raw data that are read from the input data table, and the Number of Observations Used column displays the count of data that are actually used in the clustering. The latter count is affected by the missing values in the raw data and the setting of the IMPUTE= and IMPUTENOM= options in the PROC KCLUS statement. When the FREQ statement is specified, then the Sum of Frequencies Read and Sum of Frequencies Used are also displayed.

#### Model Information

The “Model Information” table provides basic information about the parameters that are used in the cluster analysis. This information includes the clustering algorithm, maximum number of iterations, stop criterion method and the value for that criterion, number of clusters, initialization technique, seed value, distance method, standardization method, imputation method, and the value of \( \gamma \) when the k-prototypes clustering algorithm is used.

#### Cluster Summary

The “Cluster Summary” table displays the following for each cluster when the variables in INPUT statement are interval:

- cluster number
• frequency (the number of observations in the cluster)
• maximum, minimum, and average distances from the cluster centroid to the observations in the cluster
• SSE (sum of squared errors between the observations and the closest cluster centroids to them)
• standard deviation (the root mean square distances between the observations and the cluster centroids that are closest to them)
• nearest cluster (the ID of the cluster whose centroid is closest to the current cluster’s)
• distance between the cluster centroid of the current cluster and the nearest cluster

If the number of observations in a cluster is zero, then this cluster is not displayed in the “Cluster Summary” table.

**Iteration History**

The “Iteration History” table displays the following when the variables that are specified in the **INPUT** statement are interval:

• iteration number
• SSE (sum of squared distances between the observations and the closest cluster centroids)
• SSE change
• stop criterion

The “Iteration History” table displays the following when the variables that are specified in the **INPUT** statement are nominal:

• iteration number
• WithinDist (sum of distances between the observations and the closest cluster centroids)
• within distance change
• stop criterion

**Descriptive Statistics**

The “Descriptive Statistics” table displays the mean and standard deviation for each interval variable that is specified in the **INPUT** statement. The calculation of the mean and standard deviation for an interval variable uses all the data in its own column and discards the missing values. So the calculation of mean and standard deviation for one interval variable is independent of the calculation for other interval variables.
Within-Cluster Statistics

The “Within Cluster Statistics” table displays the following for each interval variable that is specified in the INPUT statement in each cluster:

- mean of the values for each variable in that cluster
- standard deviations of the values for each variable in that cluster

Cluster Summary for Nominal Variables

The “Cluster Summary for Nominal Variables” table displays the following for each cluster when the variables that are specified in the INPUT statement are nominal:

- cluster number
- frequency (the number of observations in the cluster)
- maximum, minimum, and average distances from the cluster centroid to the observations in the cluster
- within-cluster distance
- nearest cluster (the ID of the cluster whose centroid is closest to the current cluster’s centroid)
- distance between the cluster centroid of the current cluster and the nearest cluster

If the number of observations in a cluster is zero, then this cluster is not displayed in the “Cluster Summary for Nominal Variables” table.

Frequencies for Nominal Variables

The “Frequencies for Nominal Variables” table displays the following for each nominal variable in the INPUT statement in each cluster:

- levels of each variable in that cluster
- frequencies of the levels of each variable in the input data table.
- frequencies of the levels of each variable in that cluster

Cluster Summary for Mixed Variables

The “Cluster Summary for Mixed Variables” table displays the following for each cluster when the k-prototypes clustering algorithm is performed:

- cluster number
- frequency (the number of observations in the cluster)
• maximum, minimum, and average distances from the cluster centroid to the observations in the cluster
• within-cluster distance
• nearest cluster (the ID of the cluster whose centroid is closest to the current cluster’s centroid)
• distance between the cluster centroid of the current cluster and the nearest cluster

If the number of observations in a cluster is 0, then that cluster is not displayed in the “Cluster Summary for Mixed Variables” table.

**Standardization**

The “Standardization” table displays the following:

• variable name
• location value that is used for standardization
• scale value that is used for standardization

**Aligned Box Criterion Parameters**

The “ABC Parameters” table displays the following:

• minimum number of clusters that are searched
• maximum number of clusters that are searched
• reference distribution count
• alignment method used for determining the region for generating the reference data

**Aligned Box Criterion Statistics**

The “ABC Statistics” table displays the following:

• number of clusters \((k)\)
• logarithm of within-cluster sum of squares \((W_k^*)\) for the input data for \(k\) number of clusters
• logarithm of within-cluster sum of squares \((W_k)\) for the reference data for \(k\) number of clusters
• gap between the error measure from the reference data and the input data
• simulation-adjusted standard deviation \((s)\) that is obtained from the reference data
• one-standard-error adjusted gap
**Estimated Number of Clusters**

The “Estimated Number of Clusters” table displays the following:

- criterion used for estimating the number of clusters from the measures that are calculated in the specified method
- estimated number of clusters for the input data table

**OutputCasTables Table**

The OutputCasTables table is a special table that has information about each CAS table that is created during a CAS action execution. The information for each CAS table consists of the CAS table name, the caslib in which the table resides, and the number of columns and rows in the CAS table. Because this table is not a typical ODS table that contains analytical results, you cannot include it in the `table-spec-list` in the `DISPLAYOUT` statement.

**ODS Table Names**

Each table created by the KCLUS procedure has a name associated with it. You must use this name to refer to the table when you use ODS statements. These names are listed in Table 9.2.

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABCParameters</td>
<td>Parameters that are specified in the NOC= option</td>
<td>PROC KCLUS</td>
<td>NOC=ABC</td>
</tr>
<tr>
<td>ABCResults</td>
<td>Estimated number of clusters and the criteria that are used to calculate them</td>
<td>PROC KCLUS</td>
<td>NOC=ABC</td>
</tr>
<tr>
<td>ABCStats</td>
<td>Statistics that are computed for estimating the number of clusters</td>
<td>PROC KCLUS</td>
<td>NOC=ABC</td>
</tr>
<tr>
<td>ClusterSum</td>
<td>Cluster summary for interval input variables</td>
<td>INPUT</td>
<td>LEVEL=INTERVAL</td>
</tr>
<tr>
<td>ClusterSumIntNom</td>
<td>Cluster summary for mixed input variables</td>
<td>INPUT</td>
<td>LEVEL=INTERVAL and LEVEL=NOMINAL</td>
</tr>
<tr>
<td>ClusterSumNom</td>
<td>Cluster summary for nominal input variables</td>
<td>INPUT</td>
<td>LEVEL=NOMINAL</td>
</tr>
<tr>
<td>DescStats</td>
<td>Descriptive statistics for interval input variables</td>
<td>INPUT</td>
<td>LEVEL=INTERVAL</td>
</tr>
<tr>
<td>FreqNom</td>
<td>Frequencies for nominal variables in the input table and within each cluster</td>
<td>INPUT</td>
<td>LEVEL=NOMINAL</td>
</tr>
<tr>
<td>IterStats</td>
<td>Iteration history</td>
<td>PROC KCLUS</td>
<td>Default output</td>
</tr>
</tbody>
</table>
Table 9.2  continued

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>ModelInfo</td>
<td>Model information</td>
<td>PROC KCLUS</td>
<td>Default output</td>
</tr>
<tr>
<td>NObs</td>
<td>Number of observations</td>
<td>PROC KCLUS</td>
<td>Default output</td>
</tr>
<tr>
<td>OutputCasTables</td>
<td>See the section “OutputCasTables Table” on page 475</td>
<td>PROC KCLUS</td>
<td>Default output</td>
</tr>
<tr>
<td>Standardization</td>
<td>Information about the standardization method and its parameters</td>
<td>PROC KCLUS</td>
<td>STANDARDIZE=RANGE or STANDARDIZE=STD</td>
</tr>
<tr>
<td>WithinClusStats</td>
<td>Statistics for interval input variables within clusters</td>
<td>INPUT</td>
<td>LEVEL=INTERVAL</td>
</tr>
</tbody>
</table>

Examples: KCLUS Procedure

**NOTE**: Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

Example 9.1: Cluster Analysis

This example uses the Iris data set in the Sashelp library to demonstrate how to use PROC KCLUS to perform cluster analysis. The iris data published by Fisher (1936) have been widely used for examples in discriminant and cluster analyses. The sepal length, sepal width, petal length, and petal width are measured in millimeters on 50 iris specimens from each of three species: *Iris setosa*, *I. versicolor*, and *I. virginica*. Mezzich and Solomon (1980) discuss a variety of cluster analyses that use the Iris data.

You can load the Sashelp.Iris data set into your CAS session by naming your CAS engine libref in the first statement of the following DATA step:

```sql
data mycas.iris;
  set sashelp.iris;
run;
```

These statements assume that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

The following statements perform clustering:
PROC KCLUS generates the data table mycas.kclusOut1, which contains the cluster membership information for each observation in the input data table. For each observation, the mycas.kclusOut1 data table includes the variables that are specified in the COPYVARS= option in the SCORE statement and two new variables: _CLUSTER_ID_, which is the ID of the closest cluster, and _DISTANCE_, which is the distance between the observation and the centroid of the closest cluster. This example uses the variables in both the INPUT statement and the COPYVARS= option in order to transfer these variables to the output data table to do further analysis.

PROC KCLUS generates several ODS tables, some of which are shown in Output 9.1.1 through Output 9.1.6.
**Output 9.1.3** Cluster Summary

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Frequency</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Average</th>
<th>SSE</th>
<th>Standard Deviation</th>
<th>Nearest Cluster</th>
<th>Distance to Nearest Cluster Centroid</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>39</td>
<td>2.3945</td>
<td>15.5156</td>
<td>7.3185</td>
<td>2541.4</td>
<td>8.0724</td>
<td>2</td>
<td>17.8842</td>
</tr>
<tr>
<td>2</td>
<td>61</td>
<td>2.3571</td>
<td>16.4680</td>
<td>7.3111</td>
<td>3829.1</td>
<td>7.9229</td>
<td>1</td>
<td>17.8842</td>
</tr>
<tr>
<td>3</td>
<td>50</td>
<td>0.6618</td>
<td>12.4803</td>
<td>4.8171</td>
<td>1515.1</td>
<td>5.5047</td>
<td>2</td>
<td>33.4949</td>
</tr>
</tbody>
</table>

**Output 9.1.4** Iteration History

<table>
<thead>
<tr>
<th>Iteration Number</th>
<th>SSE</th>
<th>SSE Change</th>
<th>Stop Criterion</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>71498</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>13148</td>
<td>-58350</td>
<td>18.000000</td>
</tr>
<tr>
<td>2</td>
<td>8123.352556</td>
<td>-5024.490506</td>
<td>4.666667</td>
</tr>
<tr>
<td>3</td>
<td>7987.357983</td>
<td>-135.994573</td>
<td>2.000000</td>
</tr>
<tr>
<td>4</td>
<td>7934.436415</td>
<td>-52.921569</td>
<td>2.000000</td>
</tr>
<tr>
<td>5</td>
<td>7892.130972</td>
<td>-42.305442</td>
<td>0.666667</td>
</tr>
<tr>
<td>6</td>
<td>7885.566583</td>
<td>-6.564390</td>
<td>0</td>
</tr>
</tbody>
</table>

**Output 9.1.5** Descriptive Statistics

<table>
<thead>
<tr>
<th>Variable</th>
<th>Mean</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>SepalLength</td>
<td>58.433333</td>
<td>8.280661</td>
</tr>
<tr>
<td>SepalWidth</td>
<td>30.573333</td>
<td>4.358663</td>
</tr>
<tr>
<td>PetalLength</td>
<td>37.580000</td>
<td>17.652982</td>
</tr>
<tr>
<td>PetalWidth</td>
<td>11.993333</td>
<td>7.622377</td>
</tr>
</tbody>
</table>
**Example 9.1: Cluster Analysis**

**Output 9.1.6** Within-Cluster Statistics

<table>
<thead>
<tr>
<th>Variable</th>
<th>Cluster</th>
<th>Mean</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>SepalLength</td>
<td>1</td>
<td>68.5385</td>
<td>4.8820</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>58.8361</td>
<td>4.4803</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>50.0600</td>
<td>3.5249</td>
</tr>
<tr>
<td>SepalWidth</td>
<td>1</td>
<td>30.7692</td>
<td>2.8696</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>27.4098</td>
<td>2.9290</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>34.2800</td>
<td>3.7906</td>
</tr>
<tr>
<td>PetalLength</td>
<td>1</td>
<td>57.1538</td>
<td>5.1018</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>43.8852</td>
<td>5.1157</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>14.6200</td>
<td>1.7366</td>
</tr>
<tr>
<td>PetalWidth</td>
<td>1</td>
<td>20.5385</td>
<td>2.9633</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>14.3443</td>
<td>2.9994</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>2.4600</td>
<td>1.0539</td>
</tr>
</tbody>
</table>

The following statements extract the first 10 observations from the output data table; they are shown in Output 9.1.7.

```plaintext
proc print noobs data=mycas.kclusOut1(obs=10);
run;
```

**Output 9.1.7** First 10 Observations in the Output Data Table

| SepalLength | SepalWidth | PetalLength | PetalWidth | Species | _CLUSTER_ID_ | _DISTANCE_
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>33</td>
<td>14</td>
<td>2</td>
<td>Setosa</td>
<td>3</td>
<td>1.4959946524</td>
</tr>
<tr>
<td>51</td>
<td>33</td>
<td>17</td>
<td>5</td>
<td>Setosa</td>
<td>3</td>
<td>3.8259639308</td>
</tr>
<tr>
<td>52</td>
<td>34</td>
<td>14</td>
<td>2</td>
<td>Setosa</td>
<td>3</td>
<td>2.1066561181</td>
</tr>
<tr>
<td>50</td>
<td>35</td>
<td>16</td>
<td>6</td>
<td>Setosa</td>
<td>3</td>
<td>3.8675573687</td>
</tr>
<tr>
<td>48</td>
<td>30</td>
<td>14</td>
<td>3</td>
<td>Setosa</td>
<td>3</td>
<td>4.8205808779</td>
</tr>
<tr>
<td>50</td>
<td>30</td>
<td>16</td>
<td>2</td>
<td>Setosa</td>
<td>3</td>
<td>4.5208406298</td>
</tr>
<tr>
<td>58</td>
<td>40</td>
<td>12</td>
<td>2</td>
<td>Setosa</td>
<td>3</td>
<td>10.140907257</td>
</tr>
<tr>
<td>51</td>
<td>35</td>
<td>14</td>
<td>2</td>
<td>Setosa</td>
<td>3</td>
<td>1.4135062787</td>
</tr>
<tr>
<td>57</td>
<td>44</td>
<td>15</td>
<td>4</td>
<td>Setosa</td>
<td>3</td>
<td>12.048153385</td>
</tr>
<tr>
<td>52</td>
<td>41</td>
<td>15</td>
<td>1</td>
<td>Setosa</td>
<td>3</td>
<td>7.1552777724</td>
</tr>
</tbody>
</table>

PROC KCLUS creates the output statistics data table, which contains the cluster centroids. This data table includes the iteration number as _ITERATION_, the cluster ID as _CLUSTER_ID_, and the cluster centroids, which consist of the variables that are specified in the INPUT statement. Because the OUTITER= suboption is included in the OUTSTAT= option in the PROC KCLUS statement, cluster centroids for each iteration are added to the kclusOutstat1 data table.
The following statements extract the centroids before the first iteration and after the last iteration:

```plaintext
proc print noobs data=kclusOutstat1(firstobs=1 obs=3);
run;

proc print noobs data=kclusOutstat1(firstobs=16 obs=18);
run;
```

Output 9.1.8 and Output 9.1.9 show the results.

**Output 9.1.8** Cluster Centroids before the First Iteration

<table>
<thead>
<tr>
<th>ITERATION</th>
<th>CLUSTER_ID</th>
<th>SepalLength</th>
<th>SepalWidth</th>
<th>PetalLength</th>
<th>PetalWidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>63</td>
<td>25</td>
<td>49</td>
<td>15</td>
</tr>
<tr>
<td>0</td>
<td>2</td>
<td>61</td>
<td>28</td>
<td>47</td>
<td>12</td>
</tr>
<tr>
<td>0</td>
<td>3</td>
<td>64</td>
<td>29</td>
<td>43</td>
<td>13</td>
</tr>
</tbody>
</table>

**Output 9.1.9** Cluster Centroids after the Last Iteration

<table>
<thead>
<tr>
<th>ITERATION</th>
<th>CLUSTER_ID</th>
<th>SepalLength</th>
<th>SepalWidth</th>
<th>PetalLength</th>
<th>PetalWidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1</td>
<td>68.275</td>
<td>30.70</td>
<td>57.0000</td>
<td>20.6250</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>58.850</td>
<td>27.40</td>
<td>43.7667</td>
<td>14.1833</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>50.060</td>
<td>34.28</td>
<td>14.6200</td>
<td>2.4600</td>
</tr>
</tbody>
</table>

**Example 9.2: Finding the Number of Clusters**

This example uses the same data table that is loaded into your CAS session in Example 9.1.

You can find the number of clusters in the data table by specifying `NOC=ABC` in the PROC KCLUS statement as follows:

```plaintext
data mycas.iris;
   set sashelp.iris;
run;

proc kclus data=mycas.iris maxclusters=9 seed=1234
    NOC=ABC(B=10 minclusters=2 align=PCA criterion=FIRSTPEAK);
   input SepalLength SepalWidth PetalLength PetalWidth;
   ods output ABCStats=ABCStats1;
run;
```

PROC KCLUS generates several ODS tables, some of which are shown in Output 9.2.1 through Output 9.2.4. Output 9.2.1 shows the parameters that are used in the aligned box criterion (ABC) method.
Output 9.2.1  Aligned Box Criterion Parameters

The KCLUS Procedure

<table>
<thead>
<tr>
<th>ABC Parameters</th>
<th>Reference</th>
<th>Minimum Cluster</th>
<th>Maximum Cluster</th>
<th>Distribution Count</th>
<th>Alignment Method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>9</td>
<td>10</td>
<td>PCA</td>
</tr>
</tbody>
</table>

Output 9.2.2 shows the statistics that are obtained for each candidate number of clusters.

Output 9.2.2  Aligned Box Criterion Statistics

<table>
<thead>
<tr>
<th>ABC Statistics</th>
<th>Logarithm of Within-Cluster SSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Clusters</td>
<td>Input Reference Gap Simulation Adjusted Standard Deviation One Standard Error Adjusted Gap</td>
</tr>
<tr>
<td>2</td>
<td>9.6313 10.3271 0.6957 0.0304 0.6654</td>
</tr>
<tr>
<td>3</td>
<td>8.9727 9.4646 0.4919 0.0386 0.4533</td>
</tr>
<tr>
<td>4</td>
<td>8.6549 9.0401 0.3852 0.0037 0.3482</td>
</tr>
<tr>
<td>5</td>
<td>8.5167 7.6998 -0.8169 0.0738 -0.8907</td>
</tr>
<tr>
<td>6</td>
<td>8.5023 7.3959 -1.1064 0.0840 -1.1904</td>
</tr>
<tr>
<td>7</td>
<td>8.2512 6.9015 -1.3497 0.0635 -1.4132</td>
</tr>
<tr>
<td>8</td>
<td>8.1193 8.1228 0.00350 0.0497 -0.0462</td>
</tr>
<tr>
<td>9</td>
<td>8.0772 8.3575 0.2804 0.0508 0.2296</td>
</tr>
</tbody>
</table>

Output 9.2.3  Estimated Number of Clusters

<table>
<thead>
<tr>
<th>Estimated Number of Clusters</th>
<th>Number of Clusters</th>
<th>FirstPeak</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

Output 9.2.4  Cluster Summary Table for Each Cluster

<table>
<thead>
<tr>
<th>Cluster Summary for Interval Variables</th>
<th>Distance from Cluster Centroid to Observation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster</td>
<td>Frequency</td>
</tr>
<tr>
<td>---------</td>
<td>-----------</td>
</tr>
<tr>
<td>1</td>
<td>97</td>
</tr>
<tr>
<td>2</td>
<td>53</td>
</tr>
</tbody>
</table>
When you use the NOC= option, the KCLUS procedure first estimates the number of clusters, \( k \), and then it displays the cluster analysis results for each of the \( k \) clusters as shown in Output 9.2.4.

**Example 9.3: Clustering Nominal Variables**

In this example, PROC KCLUS clusters nominal variables in the Baseball data set. The Baseball data set includes 322 observations, and each observation has 24 variables. Among these 24 variables, the 5 nominal ones are selected as the input data to show an example of running \( k \)-modes clustering on a nominal data set. You can load the Sashelp.Baseball data set into your CAS session by naming your CAS engine libref in the first statement of the following DATA step.

```plaintext
data mycas.baseball;
  set sashelp.baseball;
  keep Team League Division Position Div;
run;
```

These statements assume that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

The following statements run the \( k \)-modes clustering algorithm with a frequency-based distance measure (DISTANCENOM=RELATIVEFREQ) and verify whether the clusters that the procedure obtains match the labels of the observations in the data table:

```plaintext
proc kclus data=mycas.baseball maxiter=10 maxc=5 DISTANCENOM=RELATIVEFREQ
  outstat(outiter)=kclusOutstat2;
  input Team League Division Position Div / level=nominal;
  score out=mycas.kclusOut2 copyvars=(Team League Division Position Div);
  ods output FreqNom=FreqNom1;
run;
```

Output 9.3.1 shows the cluster summary table that is produced for five clusters.

**Output 9.3.1** Cluster Summary Table for Five Clusters

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Frequency</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Average</th>
<th>Within Cluster Distance</th>
<th>Nearest Cluster</th>
<th>Distance to Nearest Cluster Centroid</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>65</td>
<td>1.6923</td>
<td>2.0000</td>
<td>1.9508</td>
<td>126.8</td>
<td>5</td>
<td>2.0000</td>
</tr>
<tr>
<td>2</td>
<td>72</td>
<td>1.6806</td>
<td>2.0000</td>
<td>1.9466</td>
<td>140.2</td>
<td>4</td>
<td>3.8824</td>
</tr>
<tr>
<td>3</td>
<td>75</td>
<td>1.6800</td>
<td>2.0000</td>
<td>1.9474</td>
<td>146.1</td>
<td>1</td>
<td>4.0000</td>
</tr>
<tr>
<td>4</td>
<td>85</td>
<td>1.7059</td>
<td>2.0000</td>
<td>1.9550</td>
<td>166.2</td>
<td>5</td>
<td>3.5200</td>
</tr>
<tr>
<td>5</td>
<td>25</td>
<td>0.9600</td>
<td>1.5200</td>
<td>1.4560</td>
<td>36.4000</td>
<td>1</td>
<td>2.0000</td>
</tr>
</tbody>
</table>

Output 9.3.2 shows the frequencies of levels for the nominal input variable Team and information about how
the levels of variables are distributed in each cluster; this information is important for revealing intracluster similarity. The following statement prints the observations from the frequency table, as shown in Output 9.3.2:

```
proc print noobs data=FreqNom1(obs=12);
run;
```

**Output 9.3.2** Frequencies for Nominal Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Level</th>
<th>FrequencyRead _1 _2 _3 _4 _5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Team Atlanta</td>
<td>11</td>
<td>0 0 11 0 0</td>
</tr>
<tr>
<td>Team Baltimore</td>
<td>15</td>
<td>0 0 0 15 0</td>
</tr>
<tr>
<td>Team Boston</td>
<td>10</td>
<td>0 0 0 10 0</td>
</tr>
<tr>
<td>Team California</td>
<td>13</td>
<td>12 0 0 0 1</td>
</tr>
<tr>
<td>Team Chicago</td>
<td>24</td>
<td>11 11 0 0 2</td>
</tr>
<tr>
<td>Team Cincinnati</td>
<td>12</td>
<td>0 0 12 0 0</td>
</tr>
<tr>
<td>Team Cleveland</td>
<td>12</td>
<td>0 0 0 12 0</td>
</tr>
<tr>
<td>Team Detroit</td>
<td>12</td>
<td>0 0 0 12 0</td>
</tr>
<tr>
<td>Team Houston</td>
<td>11</td>
<td>0 0 11 0 0</td>
</tr>
<tr>
<td>Team Kansas City</td>
<td>14</td>
<td>0 0 0 0 14</td>
</tr>
<tr>
<td>Team Los Angeles</td>
<td>14</td>
<td>0 0 14 0 0</td>
</tr>
<tr>
<td>Team Milwaukee</td>
<td>14</td>
<td>0 0 0 14 0</td>
</tr>
</tbody>
</table>

---

**Example 9.4: Clustering Mixed Variables**

In this example, PROC KCLUS uses the \( k \)-prototypes clustering algorithm to cluster mixed input data that contain both interval and nominal variables in the *Baseball* data set, which is the same data set that is used in Example 9.3. You can execute the following SAS code to load the input data table, *mycas.baseball*.

```
data mycas.baseball;
  Set sashelp.baseball;
  Keep CrAtBat CrHits CrRuns CrRbi CrBB Team League Division Position Div;
Run;
```

These statements assume that your CAS engine libref is named *mycas*, but you can substitute any appropriately defined CAS engine libref.

The following statements run the \( k \)-prototypes clustering algorithm on mixed input data. The first **INPUT** statement uses the **LEVEL=INTERVAL** option to specify the input variables *CrAtBat*, *CrHits*, *CrRuns*, *CrRbi*, and *CrBB* as the interval input variables; the second **INPUT** statement uses the **LEVEL=NOMINAL** option to specify the input variables *Team*, *League*, *Division*, *Position*, and *Div* as the nominal input variables.

The **KPROTOTYPEPARAMS=USERGAMMA(VALUE=10)** option specifies \( \gamma = 10 \) for the \( k \)-prototypes clustering algorithm. The **NOC=ABC** option uses the ABC algorithm to search for the best number of clusters in a range of 2 to 10. The **PRINTALLDISTANCES** option prints two additional variables, \_DISTANCEINT\_ and \_DISTANCENOM\_, in the output table *kclusout4*.

```
proc kclus data=mycas.baseball maxiter=10 maxc=10 distancenom=relativefreq
  outstat(outiter)=kclusOutstat4 printalldistances
  noc=abc(B=10 minclusters=2 align=none criterion=all)
  kprototypeparams=usergamma(value=10);
  input CrAtBat CrHits CrRuns CrRbi CrBB / level=interval;
  input Team League Division Position Div / level=nominal;
  score out=mycas.kclusOut4 copyvars=
```
Output 9.4.1 shows the “Model Information” table. The clustering algorithm in use is the $k$-prototypes, and the $\gamma$ value is 10. Since PROC KCLUS is implementing the $k$-prototypes algorithm on mixed input data, the distance measures for both the interval and nominal variables are displayed in the table, as Euclidean and RelativeFreq, respectively.

**Output 9.4.1** Model Information for $k$-Prototypes Clustering Algorithm

The KCLUS Procedure

<table>
<thead>
<tr>
<th>Model Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clustering Algorithm</td>
</tr>
<tr>
<td>Gamma Value</td>
</tr>
<tr>
<td>Maximum Iterations</td>
</tr>
<tr>
<td>Stop Criterion</td>
</tr>
<tr>
<td>Stop Criterion Value</td>
</tr>
<tr>
<td>Clusters</td>
</tr>
<tr>
<td>Initialization</td>
</tr>
<tr>
<td>Seed</td>
</tr>
<tr>
<td>Distance for Interval Variables</td>
</tr>
<tr>
<td>Distance for Nominal Variables</td>
</tr>
<tr>
<td>Number of Clusters Estimation</td>
</tr>
<tr>
<td>Standardization</td>
</tr>
<tr>
<td>Interval Imputation</td>
</tr>
<tr>
<td>Nominal Imputation</td>
</tr>
</tbody>
</table>

Output 9.4.2 shows the cluster summary table that is produced for the $k$-prototypes algorithm.

**Output 9.4.2** Cluster Summary Table for $k$-Prototypes Clustering Algorithm

<table>
<thead>
<tr>
<th>Cluster Summary for Mixed Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distance from Cluster Centroid to Observation</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Frequency</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Average</th>
<th>Within Cluster Distance</th>
<th>Nearest Cluster</th>
<th>Distance to Nearest Cluster Centroid</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>37</td>
<td>325.6</td>
<td>24418.9</td>
<td>8426.7</td>
<td>311786</td>
<td>7</td>
<td>125993</td>
</tr>
<tr>
<td>2</td>
<td>33</td>
<td>16854.6</td>
<td>1233915</td>
<td>401677</td>
<td>13255332</td>
<td>9</td>
<td>3110383</td>
</tr>
<tr>
<td>3</td>
<td>35</td>
<td>1734.9</td>
<td>170086</td>
<td>55027.1</td>
<td>1925950</td>
<td>8</td>
<td>505417</td>
</tr>
<tr>
<td>4</td>
<td>24</td>
<td>47889.6</td>
<td>41995031</td>
<td>2652949</td>
<td>63670772</td>
<td>2</td>
<td>4956558</td>
</tr>
<tr>
<td>5</td>
<td>44</td>
<td>7563.1</td>
<td>419330</td>
<td>115914</td>
<td>5100208</td>
<td>3</td>
<td>918178</td>
</tr>
<tr>
<td>6</td>
<td>35</td>
<td>526.8</td>
<td>53906.1</td>
<td>13513.4</td>
<td>472969</td>
<td>7</td>
<td>122221</td>
</tr>
<tr>
<td>7</td>
<td>34</td>
<td>506.1</td>
<td>31035.0</td>
<td>9465.6</td>
<td>321832</td>
<td>6</td>
<td>122221</td>
</tr>
<tr>
<td>8</td>
<td>43</td>
<td>3104.8</td>
<td>105746</td>
<td>35130.7</td>
<td>1510619</td>
<td>6</td>
<td>304674</td>
</tr>
<tr>
<td>9</td>
<td>37</td>
<td>14338.2</td>
<td>800648</td>
<td>220891</td>
<td>8172968</td>
<td>5</td>
<td>1717291</td>
</tr>
</tbody>
</table>
References


# Chapter 10
## The L MIXED Procedure

## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview: L MIXED Procedure</td>
<td>488</td>
</tr>
<tr>
<td>PROC L MIXED Features</td>
<td>489</td>
</tr>
<tr>
<td>Notation for the Mixed Model</td>
<td>490</td>
</tr>
<tr>
<td>PROC L MIXED Compared to Other SAS Procedures</td>
<td>491</td>
</tr>
<tr>
<td>Using CAS Sessions and CAS Engine Librefs</td>
<td>491</td>
</tr>
<tr>
<td>Getting Started: L MIXED Procedure</td>
<td>492</td>
</tr>
<tr>
<td>Syntax: L MIXED Procedure</td>
<td>495</td>
</tr>
<tr>
<td>PROC L MIXED Statement</td>
<td>496</td>
</tr>
<tr>
<td>BLUP Statement</td>
<td>499</td>
</tr>
<tr>
<td>BY Statement</td>
<td>500</td>
</tr>
<tr>
<td>CLASS Statement</td>
<td>501</td>
</tr>
<tr>
<td>DISPLAY Statement</td>
<td>501</td>
</tr>
<tr>
<td>DISPLAYOUT Statement</td>
<td>502</td>
</tr>
<tr>
<td>EFFECT Statement</td>
<td>503</td>
</tr>
<tr>
<td>MODEL Statement</td>
<td>504</td>
</tr>
<tr>
<td>OPTIMIZATION Statement</td>
<td>506</td>
</tr>
<tr>
<td>OUTPUT Statement</td>
<td>509</td>
</tr>
<tr>
<td>PARMS Statement</td>
<td>511</td>
</tr>
<tr>
<td>RANDOM Statement</td>
<td>514</td>
</tr>
<tr>
<td>REPEATED Statement</td>
<td>523</td>
</tr>
<tr>
<td>WEIGHT Statement</td>
<td>525</td>
</tr>
<tr>
<td>Details: L MIXED Procedure</td>
<td>525</td>
</tr>
<tr>
<td>Linear Mixed Models Theory</td>
<td>525</td>
</tr>
<tr>
<td>Matrix Notation for General Linear Model</td>
<td>525</td>
</tr>
<tr>
<td>Formulation of the Mixed Model</td>
<td>526</td>
</tr>
<tr>
<td>Estimating Covariance Parameters in the Mixed Model</td>
<td>529</td>
</tr>
<tr>
<td>Computing Methodologies for REML in the Mixed Model</td>
<td>530</td>
</tr>
<tr>
<td>Estimating Fixed and Random Effects in the Mixed Model</td>
<td>531</td>
</tr>
<tr>
<td>Statistical Properties</td>
<td>532</td>
</tr>
<tr>
<td>Common Subject Effect</td>
<td>532</td>
</tr>
<tr>
<td>Computational Method</td>
<td>534</td>
</tr>
<tr>
<td>Distributed Computing</td>
<td>534</td>
</tr>
<tr>
<td>Multithreading</td>
<td>534</td>
</tr>
<tr>
<td>Displayed Output</td>
<td>534</td>
</tr>
<tr>
<td>Model Information</td>
<td>534</td>
</tr>
<tr>
<td>Class Level Information</td>
<td>534</td>
</tr>
</tbody>
</table>
Overview: LMIXED Procedure

The LMIXED procedure fits a variety of linear mixed models to data and enables you to use these fitted models to make statistical inferences about the data. A linear mixed model is a generalization of the standard linear model that is used in the GLM procedure in SAS/STAT software; the generalization is that the data are permitted to exhibit correlation and nonconstant variability. Therefore, the linear mixed model provides you with the flexibility of modeling not only the means of your data (as in the standard linear model) but also their variances and covariances.

The primary assumptions underlying the analyses that the LMIXED procedure performs are as follows:

- The data are normally distributed (Gaussian).
- The means (expected values) of the data are linear in terms of a certain set of parameters.
- The variances and covariances of the data are conditioned on a different set of parameters, with a structure that the LMIXED procedure supports.

Because Gaussian data can be modeled entirely in terms of their means, variances and covariances, the two sets of parameters in a linear mixed model specify the complete probability distribution of the data. The parameters of the mean model are called fixed-effects parameters, and the parameters of the variance-covariance model are called covariance parameters.

The fixed-effects parameters are associated with known explanatory variables, as in the standard linear model. These variables can be either qualitative (as in the traditional analysis of variance) or quantitative (as in standard linear regression).

The covariance parameters are what distinguishes the linear mixed model from the standard linear model. The need for covariance parameters arises quite frequently in applications. A typical application follows a scenario in which the experimental units that the data are measured on can be grouped into clusters and the data from a common cluster are correlated. This scenario can be generalized to include one set of clusters...
nested within another. For example, if students are the experimental unit, they can be clustered into classes, which in turn can be clustered into schools. Each level of this hierarchy can introduce an additional source of variability and correlation.

The LMGIXED procedure provides a variety of covariance structures. The most common covariance structures arise from the use of random effects, which are additional unknown random variables that are assumed to affect the variability of the data. The variances of the random effects, commonly known as variance components, become the covariance parameters for this particular structure. Traditional linear mixed models contain both fixed and random effects; in fact, it is the combination of these two types of effects that led to the name mixed model. The LMGIXED procedure fits not only these traditional variance component models but also numerous other covariance structures.

The LMGIXED procedure fits the structure you select by using the method of restricted maximum likelihood (REML), also known as residual maximum likelihood. The method of maximum likelihood (ML), is also available.

**PROC LMGIXED Features**

The LMGIXED procedure provides easy accessibility to numerous linear mixed models that are useful in many common statistical analyses.

Here are the main features of the LMGIXED procedure:

- The RANDOM statement supports many covariance structures, including variance components, compound symmetry, unstructured, AR(1), Toeplitz, factor analytic, and so on.
- Both the MODEL statement and the RANDOM statement are supported for model specification, as in the MIXED procedure.
- The REPEATED statement supports many residual covariance structures, including variance components, compound symmetry, unstructured, AR(1), and so on.
- Inference features include standard errors and $t$ tests for fixed and random effects.
- A subject effect for blocking is supported.
- Both REML and ML estimation methods are supported; they are implemented with a variety of optimization algorithms.
- It handles unbalanced data.
- Specialized dense and sparse matrix algorithms are provided.
- The OUTPUT statement produces output data tables that contain predicted values, residuals, studentized residuals, confidence limits, and influence statistics.
- The PARMs statement enables you to fit a linear mixed model that has known covariance values or to set boundary values for the parameters.

Because the LMGIXED procedure runs on SAS Cloud Analytic Services (CAS, see the section “Using CAS Sessions and CAS Engine Librefs” on page 491), it also does the following:
enables you to run on a cluster of machines that distribute the data and the computations

enables you to run in single-machine mode on CAS

exploits all the available cores and concurrent threads. For information about how the LMIXED procedure uses threads, see the section “Multithreading” on page 81 in Chapter 3, “Shared Concepts.”

Notation for the Mixed Model

This section introduces the mathematical notation that is used throughout this chapter to describe the linear mixed model. It assumes familiarity with basic matrix algebra (for an overview, see Searle 1982). A more detailed description of the mixed model is contained in the section “Linear Mixed Models Theory” on page 525.

A statistical model is a mathematical description of how data are generated. The standard linear model, as used by the GLM procedure, is one of the most common statistical models:

\[ y = X \beta + \epsilon \]

In this expression, \( y \) represents a vector of observed data, \( \beta \) is an unknown vector of fixed-effects parameters with a known design matrix \( X \), and \( \epsilon \) is an unknown random error vector that models the statistical noise around \( X \beta \). The focus of the standard linear model is to model the mean of \( y \) by using the fixed-effects parameters \( \beta \). The residual errors \( \epsilon \) are assumed to be independent and identically distributed Gaussian random variables with mean 0 and variance \( \sigma^2 \).

The mixed model generalizes the standard linear model as follows:

\[ y = X \beta + Z \gamma + \epsilon \]

Here, \( \gamma \) is an unknown vector of random-effects parameters with a known design matrix \( Z \), and \( \epsilon \) is an unknown random-error vector whose elements are no longer required to be independent and homogeneous.

The number of fixed-effects parameters is called the number of levels for fixed effects throughout this chapter. The number of random-effects parameters is called the number of levels for random effects throughout this chapter.

To further develop this notion of variance modeling, assume that \( \gamma \) and \( \epsilon \) are Gaussian random variables that are uncorrelated, have expectations 0, and have variances \( G \) and \( R \), respectively. The variance of \( y \) is thus

\[ V = ZGZ' + R \]

Note that when \( Z = 0 \), the mixed model reduces to the standard linear model.

You can model the variance of the data \( y \) by specifying the structures of \( Z, G, \) and \( R \). The model matrix \( Z \) is set up in the same fashion as \( X \), the model matrix for the fixed-effects parameters. For \( G \) and \( R \), you must select some covariance structures. Possible covariance structures include the following:

- variance components
- compound symmetry (common covariance plus diagonal)
unstructured (general covariance)

autoregressive

By appropriately defining the model matrices $X$ and $Z$ in addition to the covariance structure matrixes $G$ and $R$, you can perform numerous mixed model analyses.

---

The LMIXED procedure provides linear mixed modeling functionality that is most comparable to that of the MIXED, HPMIXED, and HPLMIXED procedures in SAS/STAT software. The GLIMMIX procedure and the NLMIXED procedure can also be used to fit linear mixed models, but these procedures are primarily designed to fit generalized linear mixed models (GLMMs) and nonlinear mixed models, respectively.

The LMIXED procedure, like PROC HPMIXED, can fit linear mixed models by using sparse matrix storage and sparse matrix computations. These sparse matrix techniques are particularly suitable for problems in which the $|XZ'|XZ$ crossproducts matrix is large and sparse. Unlike PROC HPMIXED, the LMIXED procedure does not yet support the CONTRAST, ESTIMATE, LSMEANS and TEST statements.

The LMIXED procedure, like PROC HPLMIXED, can fit linear mixed models by using specialized efficient algorithms. This is particularly suitable for models that have thousands of levels in a common subject effect.

The LMIXED procedure is designed to fit the following large linear mixed models:

- models that have thousands of levels in the common subject effect. For information about the common subject effect, see “Common Subject Effect” on page 532.
- models that have a large and sparse design matrix.

---

Using CAS Sessions and CAS Engine Librefs

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:

```sas
proc options option=(CASHOST CASPORT);
run;
```

In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:
cas mysess;
libname mycas cas sessref=mysess;

The CAS statement creates the CAS session named mysess, and the LIBNAME statement creates the mycas CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the CASHOST and CASPORT of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the mysess session, you can terminate it by using the TERMINATE option in the CAS statement as follows:

cas mysess terminate;

For more information about the CAS and LIBNAME statements, see the section “Introduction to Shared Concepts” on page 10 in Chapter 3, “Shared Concepts.”

---

Getting Started: LMIXED Procedure

This example demonstrates how you can perform a mixed model analysis of covariance when you have many groups. Suppose you are an educational researcher who studies how student scores on math tests change over time. Students are tested four times, and you want to estimate the overall rise or fall, accounting for correlation between test response behaviors of students in the same neighborhood and school. One way to model this correlation is by using a random-effects analysis of covariance, where in addition to having a student-specific quadratic mean response function, the scores for students from the same neighborhood and school are all assumed to share the same quadratic mean test response function, whose parameters are random. The following statements simulate data that have this structure; these statements assume that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

NOTE: Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

data mycas.SchoolSample;
  do SchoolID = 1 to 1000;
    do nID = 1 to 50;
      Neighborhood = (SchoolID-1)*15 + nID;
      bInt = 15*ranuni(1);
      bTime = 15*ranuni(1);
      bTime2 = ranuni(1);
      do sID = 1 to 2;
        do Time = 1 to 4;
          Math = bInt + bTime*Time + bTime2*Time*Time + rannor(2);
          output;
        end;
      end;
    end;
  end;
run;
These data come from 1,000 schools and about 15,035 neighborhoods; neighborhoods are associated with more than one school and vice versa. There are 400,000 observations and 50,000 levels of the subject effect.

The following statements use the LMIXED procedure to fit a linear mixed model to these data and produce Figure 10.1 through Figure 10.4:

```sas
proc lmxixed data=mycas.SchoolSample;
  class Neighborhood SchoolID;
  model Math = Time Time*Time / solution;
  random int Time Time*Time / sub=Neighborhood(SchoolID) type=un;
run;
```

This model fits a quadratic mean response model that has an unstructured covariance matrix to model the covariance between the random parameters of the response model. With 15,035 neighborhood/school combinations, this model can be computationally daunting to fit, but the LMIXED procedure finishes quickly by distributing the data and the likelihood computation for different subjects in a CAS computing environment.

Figure 10.1 displays the “Model Information,” “Optimization Information,” “Class Level Information,” “Number of Observations Information,” and “Dimensions” tables.

## Figure 10.1 Mixed Model Analysis of Covariance

### The LMIXED Procedure

<table>
<thead>
<tr>
<th>Model Information</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data Source</strong></td>
<td>SCHOOLSAMPLE</td>
</tr>
<tr>
<td><strong>Response Variable</strong></td>
<td>Math</td>
</tr>
<tr>
<td><strong>Estimation Method</strong></td>
<td>Restricted Maximum Likelihood (REML)</td>
</tr>
<tr>
<td><strong>Degrees of Freedom Method</strong></td>
<td>Residual</td>
</tr>
<tr>
<td><strong>Design Matrix Method</strong></td>
<td>Dense</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Optimization Information</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Optimization Technique</strong></td>
<td>Newton-Raphson with Ridging</td>
</tr>
<tr>
<td><strong>Hessian in Optimization</strong></td>
<td>Exact</td>
</tr>
<tr>
<td><strong>Parameters in Optimization</strong></td>
<td>6</td>
</tr>
<tr>
<td><strong>Lower Boundaries</strong></td>
<td>3</td>
</tr>
<tr>
<td><strong>Upper Boundaries</strong></td>
<td>0</td>
</tr>
<tr>
<td><strong>Residual Variance</strong></td>
<td>Profiled</td>
</tr>
<tr>
<td><strong>Starting Values From</strong></td>
<td>Data</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Class Level Information</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Class</strong></td>
<td><strong>Levels</strong></td>
</tr>
<tr>
<td>Neighborhood</td>
<td>15035</td>
</tr>
<tr>
<td>SchoolID</td>
<td>1000</td>
</tr>
</tbody>
</table>

| **Number of Observations Read** | 400000 |
| **Number of Observations Used** | 400000 |
Figure 10.1  continued

<table>
<thead>
<tr>
<th>Dimensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>G-side Covariance Parameters</td>
</tr>
<tr>
<td>R-side Covariance Parameters</td>
</tr>
<tr>
<td>Columns in X</td>
</tr>
<tr>
<td>Columns in Z per Subject</td>
</tr>
<tr>
<td>Subjects (Blocks in V)</td>
</tr>
</tbody>
</table>

Figure 10.2 displays the convergence status and the “Covariance Parameter Estimates” table.

Figure 10.2  Mixed Model Analysis of Covariance

Convergence criterion (GCONV=1E-8) satisfied.

Covariance Parameter Estimates

<table>
<thead>
<tr>
<th>Cov Parm</th>
<th>Subject</th>
<th>Estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>UN(1,1)</td>
<td>Neighborhood(SchoolID)</td>
<td>18.7964</td>
</tr>
<tr>
<td>UN(2,1)</td>
<td>Neighborhood(SchoolID)</td>
<td>-0.1215</td>
</tr>
<tr>
<td>UN(2,2)</td>
<td>Neighborhood(SchoolID)</td>
<td>18.9874</td>
</tr>
<tr>
<td>UN(3,1)</td>
<td>Neighborhood(SchoolID)</td>
<td>-0.00044</td>
</tr>
<tr>
<td>UN(3,2)</td>
<td>Neighborhood(SchoolID)</td>
<td>-0.00805</td>
</tr>
<tr>
<td>UN(3,3)</td>
<td>Neighborhood(SchoolID)</td>
<td>0.08394</td>
</tr>
<tr>
<td>Residual</td>
<td></td>
<td>0.9998</td>
</tr>
</tbody>
</table>

Figure 10.3 displays the “Fit Statistics” table.

Figure 10.3  Mixed Model Analysis of Covariance

Fit Statistics

| -2 Res Log Likelihood | 1681870 |
| AIC (smaller is better)| 1681884 |
| AICC (smaller is better)| 1681884 |
| BIC (smaller is better)| 1681945 |
| CAIC (smaller is better)| 1681952 |
| HQIC (smaller is better)| 1681903 |

Figure 10.4 displays the “Solution for Fixed Effects” table.

Figure 10.4  Mixed Model Analysis of Covariance

Solution for Fixed Effects

| Effect  | Estimate | Standard Error | DF | t Value | Pr > |t| |
|---------|----------|----------------|----|---------|------|---|
| Intercept| 7.4961   | 0.02129        | 4E5| 352.04  | <.0001 |
| Time    | 7.4939   | 0.02108        | 4E5| 355.55  | <.0001 |
| Time*Time| 0.5009   | 0.002044       | 4E5| 245.03  | <.0001 |

These estimates are very close to the quadratic coefficients that are used in the DATA step multiplied by 1/2, which is the expected value of a uniform random variable on [0,1].
Syntax: LMIXED Procedure

The following statements are available in the LMIXED procedure:

```
PROC LMIXED < options > ;
   BLUP OUT=CAS-libref.data-table < options > ;
   BY variables ;
   CLASS variable < (options ) > . . . < variable < (options ) > > < / global-options > ;
   DISPLAY < table-list > < / options > ;
   DISPLAYOUT table-spec-list < / options > ;
   EFFECT name=effect-type (variables < / options > ) ;
   MODEL dependent = < fixed-effects > < / options > ;
   OPTIMIZATION < options > ;
   OUTPUT OUT=CAS-libref.data-table
      < COPYVARS=(variables)>
      < keyword =name > . . . < keyword =name > > ;
   PARMS < (value-list ) . . . > < / options > ;
   RANDOM random-effects < /options > ;
   REPEATED repeated-effect < /options > ;
   WEIGHT variable ;
```

Items within angle brackets ( < > ) are optional.

The `PROC LMIXED` and `MODEL` statements are required, and the `MODEL` statement must appear after the `CLASS` statement if a `CLASS` statement is included. The `RANDOM` statement must follow the `MODEL` statement. The `RANDOM` statement can appear multiple times. Other statements can appear only once.

The `BLUP`, `BY`, `MODEL`, `OPTIMIZATION`, `OUTPUT`, `PARMS`, `RANDOM`, `REPEATED`, and `WEIGHT` statements are described in full in alphabetical order after the `PROC LMIXED` statement. The `CLASS` and `EFFECT` statements are described by summarizing their functionality and syntax; for more information about these statements, see Chapter 3, “Shared Concepts.”

Table 10.1 summarizes the basic functions and important options and statements that are supported by the LMIXED procedure.

<table>
<thead>
<tr>
<th>Statement</th>
<th>Description</th>
<th>Important Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROC LMIXED</td>
<td>Invokes the procedure</td>
<td>The <code>DATA=</code> option specifies the input data table; the <code>METHOD=</code> option specifies the estimation method.</td>
</tr>
<tr>
<td>BLUP</td>
<td>Creates a data table that contains the estimation of fixed and random effects</td>
<td>The <code>OUT=</code> option specifies the output data table; the <code>SOLVER=</code> option specifies the mixed model equations solver.</td>
</tr>
<tr>
<td>CLASS</td>
<td>Declares qualitative variables that create indicator variables in X and Z matrices</td>
<td>None</td>
</tr>
</tbody>
</table>

Table 10.1 Summary of PROC LMIXED Statements
**Table 10.1 continued**

<table>
<thead>
<tr>
<th>Statement</th>
<th>Description</th>
<th>Important Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>MODEL</td>
<td>Specifies the dependent variable and the fixed effects, setting up X</td>
<td>The S option requests a solution for fixed-effects parameters.</td>
</tr>
<tr>
<td>OPTIMIZATION</td>
<td>Controls aspects of the nonlinear optimizations</td>
<td>The TECHNIQUE= option specifies the optimization technique.</td>
</tr>
<tr>
<td>OUTPUT</td>
<td>Creates a data table that contains observationwise statistics</td>
<td>The ALLSTATS option requests that all statistics be computed.</td>
</tr>
<tr>
<td>PARMS</td>
<td>Specifies a grid of initial values for the covariance parameters</td>
<td>The HOLD= and NOITER options hold the covariance parameters or their ratios constant; the PARMSDATA= option reads the initial values from a data table.</td>
</tr>
<tr>
<td>RANDOM</td>
<td>Specifies random effects, setting up Z and G</td>
<td>The SUBJECT= option creates block-diagonality; the TYPE= option specifies the covariance structure; the S option requests a solution for the random effects.</td>
</tr>
<tr>
<td>REPEATED</td>
<td>Specifies R</td>
<td>The SUBJECT= option creates block-diagonality; the TYPE= option specifies covariance structure; the R= option displays estimated blocks of R; the GROUP= option enables between-subject heterogeneity.</td>
</tr>
</tbody>
</table>

**PROC L MIXED Statement**

```
PROC L MIXED < options > ;
```

The PROC L MIXED statement invokes the procedure. Table 10.2 summarizes important options in the PROC L MIXED statement by function. These and other options in the PROC L MIXED statement are then described fully in alphabetical order.

**Table 10.2 PROC L MIXED Statement Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATA=</td>
<td>Specifies the input data table</td>
</tr>
<tr>
<td>METHOD=</td>
<td>Specifies the estimation method</td>
</tr>
<tr>
<td>NAMELEN=</td>
<td>Limits the length of effect names</td>
</tr>
<tr>
<td>TIMING</td>
<td>Displays the “Timing” table</td>
</tr>
</tbody>
</table>
Table 10.2  continued

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Options Related to Output</strong></td>
<td></td>
</tr>
<tr>
<td>ITDETAILS</td>
<td>Displays estimates and gradients added to the “Iteration History” table</td>
</tr>
<tr>
<td>MAXCLPRINT=</td>
<td>Specifies the maximum number of levels of CLASS variables to print</td>
</tr>
<tr>
<td>MMEQ</td>
<td>Displays the mixed model equations</td>
</tr>
<tr>
<td>NOCLPRINT</td>
<td>Suppresses the “Class Level Information” table completely or in part</td>
</tr>
<tr>
<td>RANKS</td>
<td>Displays the rank of the design matrix X</td>
</tr>
<tr>
<td>SIMPLE</td>
<td>Displays the “Descriptive Statistics” table</td>
</tr>
<tr>
<td><strong>Singularity Tolerances</strong></td>
<td></td>
</tr>
<tr>
<td>SINGCHOL=</td>
<td>Specifies the singularity criterion for Cholesky decomposition</td>
</tr>
<tr>
<td>SINGRES=</td>
<td>Specifies the singularity criterion for the residual variance</td>
</tr>
<tr>
<td>SINGULAR=</td>
<td>Specifies the general singularity criterion</td>
</tr>
<tr>
<td><strong>Design Matrix Method</strong></td>
<td></td>
</tr>
<tr>
<td>DMMETHOD=</td>
<td>Specifies the computing method (sparse or dense)</td>
</tr>
</tbody>
</table>

You can specify the following *options* in the PROC L MIXED statement.

**DATA=** CAS-libref.data-table

names the input data table for PROC L MIXED to use. The default is the most recently created data table. *CAS-libref.data-table* is a two-level name, where

- **CAS-libref** refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to the data, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about *CAS-libref*, see the section “Using CAS Sessions and CAS Engine Librefs” on page 491.

- **data-table** specifies the name of the input data table.

**DMMETHOD=DENSE | SPARSE**

specifies the method for computing the likelihood, gradient, and Hessian. For more information, see the section “Common Subject Effect” on page 532. Also see Table 10.9 for information about the covariance types that are supported by these two computing methods.

This method can significantly impact both the run-time and the memory efficiency of the L MIXED procedure.

**DENSE** is more suitable for models that have a large number of levels for a common subject effect and a relatively small number of levels for fixed-effects.

**SPARSE** is more memory-efficient and faster for models that have a large and sparse design matrix.
For more information about these two different methods, see the section “Computing Methodologies for REML in the Mixed Model” on page 530.

If you omit this option, the default is set according to the following rules:

- In fixed-effects models without CLASS variables, DMMETHOD=DENSE by default.
- In fixed-effects models with CLASS variables, DMMETHOD=SPARSE by default.
- In random-effects models without a common subject effect, DMMETHOD=SPARSE by default.
- In random-effects models with a common subject effect, DMMETHOD=DENSE by default.
- In models with a REPEATED statement, DMMETHOD=SPARSE by default.
- In all other situations, DMMETHOD=SPARSE.

ITDETAILS
  displays the parameter values at each iteration and enables the writing of notes to the SAS log that pertain to infinite likelihood and singularities during optimization iterations.

MAXCLPRINT=number
  specifies the maximum number of levels of CLASS variables to print in the ODS table “ClassLevels.” MAXCLPRINT=0 enables you to print all levels of each CLASS variable. However, the option NOCLPRINT takes precedence over MAXCLPRINT. By default, MAXCLPRINT=20.

METHOD=ML | REML
  specifies the estimation method for the covariance parameters.
  
  ML performs maximum likelihood estimation.
  
  REML performs residual (restricted) maximum likelihood estimation.
  
  By default, METHOD=REML.

MMEQ
  displays the coefficients of the mixed model equations, which are

  \[
  \begin{bmatrix}
  X\hat{R}^{-1}X & X\hat{R}^{-1}Z \\
  Z\hat{R}^{-1}X & Z\hat{R}^{-1}Z + \hat{G}^{-1}
  \end{bmatrix}
  \begin{bmatrix}
  X\hat{R}^{-1}y \\
  Z\hat{R}^{-1}y
  \end{bmatrix}
  \]

  where \( \hat{G} \) and \( \hat{R} \) are estimates of \( G \) and \( R \).

  For more information about these equations, see the section “Linear Mixed Models Theory” on page 525.

NAMELEN=number
  specifies the length to which long effect names are shortened. The minimum value is 20. Be default, NAMELEN=20.

NOCCLPRINT<=number>
  suppresses the display of the “Class Level Information” table if you do not specify \( \text{number} \). If you specify \( \text{number} \), the values of the classification variables are displayed only for variables whose number of levels is less than \( \text{number} \). Specifying a \( \text{number} \) helps reduce the size of the “Class Level Information” table if some classification variables have a large number of levels.
NOINFO
suppresses the display of the “Model Information,” “Number of Observations,” and “Dimensions” tables.

NOITPRINT
suppresses the display of the “Iteration History” table.

NOPRINT
suppresses the normal display of results. This option is useful when you want only to create one or more output data tables by using the OUTPUT statement.

NOPROFILE
includes the residual variance as one of the covariance parameters in the optimization iterations. (By default, this parameter is profiled out of the optimization iterations, except when you have specified the HOLD= option in the PARMS statement.)

The NOPROFILE option is supported only when DMMETHOD=SPARSE.

RANKS
displays the rank of the design matrix X.

This option is supported only when DMMETHOD=SPARSE.

SIMPLE
displays the mean, standard deviation, coefficient of variation, minimum, and maximum for each variable that is not a classification variable in the input DATA= table.

SINGCHOL=number
tunes the singularity criterion in Cholesky decompositions. The default is 1E4 times the machine epsilon, which is approximately 1E–12 on most computers.

SINGRES=number
sets the tolerance for which the residual variance is considered to be zero. The default is 1E4 times the machine epsilon, which is approximately 1E–12 on most computers.

SINGULAR=number
tunes the general singularity criterion that the LMIXED procedure applies in sweeps and inversions. The default is 1E4 times the machine epsilon, which is approximately 1E–12 on most computers.

TIMING
displays the amount of time (in seconds) that PROC LMIXED took to perform each different task in the analysis and its percentage of the total time.

---

**BLUP Statement**

```
BLUP OUT=CAS-libref.data-table <options> ;
```

The BLUP statement creates a CAS data table that contains the best linear unbiased estimation (BLUE) of fixed effects and the best linear unbiased prediction (BLUP) of random effects. To use the BLUP statement, you must also use the PARMS statement to specify fixed values for all covariance parameters. When you use
the BLUP statement, the ODS IterHistory table displays solver iterations that are used for BLUP solutions rather than displaying optimization iterations.

You must specify the following option:

**OUT=** `CAS-libref.data-table`

names the output data table for PROC LMIXED to use. You must specify this option before any other options. `CAS-libref.data-table` is a two-level name, where

- `CAS-libref` refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to where the data table is to be stored, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about `CAS-libref`, see the section “Using CAS Sessions and CAS Engine Librefs” on page 491.

- `data-table` specifies the name of the output data table.

You can also specify the following options:

- **ITPRINT=** `number`
  displays that the iteration history after every `number` of iterations. By default, ITPRINT=10, which means the procedure displays the iteration history for every 10 iterations.

- **MAXITER=** `number`
  specifies the maximum number of iterations. The default value is the number of parameters in the OUT= data table plus 2.

- **SOLVER=** `DIRECT | IOC | IOD`
  specifies the solver to use for BLUP solutions.

  - **DIRECT** requires storing mixed model equations (MMEQ) in memory and computing the Cholesky decomposition of MMEQ. This solver is the most accurate, but it is the most inefficient in terms of speed and memory.
  
  - **IOC** requires storing mixed model equations in memory and iterates on MMEQ to solve for the solutions. This solver is the most efficient in terms of speed.
  
  - **IOD** does not build mixed model equations; instead it iterates on data to solve for the solutions. This solver is most efficient in terms of memory.

  By default, SOLVER=IOC.

- **TOL=** `number`
  specifies the tolerance value. The default value is the square root of machine precision.

### BY Statement

**BY** `variables`;

You can specify a BY statement in PROC LMIXED to obtain separate analyses of observations in groups that are defined by the values of the BY variables. If you specify more than one BY statement, only the last one specified is used. For more information, see the discussion of BY-group processing in *SAS Language Reference: Concepts*.
CLASS Statement

```latex
CLASS variable <(options)> . . . < variable <(options)> > < / global-options > ;
```

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. You can list the response variable for binary models in the CLASS statement, but this is not required. Table 10.3 summarizes the values that you can use as either an option or a global-option. The options are fully documented in the section “CLASS Statement” on page 12 in Chapter 3, “Shared Concepts.”

**Table 10.3** CLASS Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESCENDING</td>
<td>Reverses the sort order</td>
</tr>
<tr>
<td>MISSING</td>
<td>Treats missing values as valid levels</td>
</tr>
<tr>
<td>ORDER=</td>
<td>Specifies the sort order for the levels</td>
</tr>
<tr>
<td>PARAM=</td>
<td>Specifies the parameterization of the variable</td>
</tr>
<tr>
<td>REF=</td>
<td>Specifies the reference level of the variable and puts this level at the end of the list of levels. This interpretation of the specified REF= level does not apply when DMMETHOD=SPARSE because in this case the L MIXED procedure does not necessarily solve mixed model equations in the original order.</td>
</tr>
</tbody>
</table>

DISPLAY Statement

```latex
DISPLAY < table-list > < / options > ;
```

The DISPLAY statement enables you to specify a list of display tables to display or exclude. This statement is similar to the ODS SELECT, ODS EXCLUDE, and ODS TRACE statements. However, the DISPLAY statement can improve performance when a large number of tables could be generated (such as in BY-group processing). The procedure processes the DISPLAY statement on a CAS server and thus sends only a subset of ODS tables to the SAS client. Because ODS statements are processed on a SAS client, first all the generated display tables are sent to the client, and then the client creates a subset.

If you use both DISPLAY and ODS statements together, the DISPLAY statement takes precedence over the ODS statements. Note that the ODS EXCLUDE statement processes tables that are sent to the client after they have been filtered by the DISPLAY statement. In some cases, it might appear that the ODS EXCLUDE statement is taking precedence because it can further filter the tables. For more information about ODS, see SAS Output Delivery System: Procedures Guide.

You can specify the `table-list` as a list of table names, paths, partial pathnames, and regular expressions.

The table names that you can specify are listed in the section “ODS Table Names” on page 537. A path is a table name that is prefixed with dot-separated grouping information. For example, a SelectionSummary table that a procedure produces during a selection routine might have the path `Bygroup1.Summary:SelectionSummary`. A partial pathname does not include all groups; for example, `SelectionSummary` and `Summary:SelectionSummary` are partial pathnames for `Bygroup1.Summary:SelectionSummary`. 
When you specify a table name or partial pathname, all display tables whose paths end in the specified name are selected for display or exclusion. For example, both SelectionSummary and Summary:SelectionSummary select Bygroup1.Summary:SelectionSummary.

A regular expression is enclosed in forward slashes (/). For example, specifying “/tions/” selects all pathnames that contain the substring “tions”; in particular, the Bygroup1.Summary:SelectionSummary table is selected. Specifying “!/tions/” selects all pathnames that do not contain the substring “tions”; in particular, the Bygroup1.Summary:SelectionSummary table is not selected.

You can specify the following options after a slash (/):

- **CASESENSITIVE**: performs a case-sensitive comparison of table names in the table-list to display table names when tables are subsetted for display. To preserve case, you must enclose table names in the table-list in quotation marks.

- **EXCLUDE**: displays all display tables except those that you specify in the table-list.

- **EXCLUDEALL**: suppresses display of all tables. This option takes precedence over the other options.

- **TRACE**: displays the display table names, labels, and paths.

---

**DISPLAYOUT Statement**

```plaintext
DISPLAYOUT table-spec-list < / options > ;
```

The DISPLAYOUT statement enables you to create CAS output tables from your displayed output. This statement is similar to the ODS OUTPUT statement. For more information about ODS, see SAS Output Delivery System: Procedures Guide.

The `table-spec-list` specifies a list of CAS output tables to create. Each entry in the list has either a `key=value` format or a `key` format:

- `key=value` specifies `key` as the ODS table name, path, or partial pathname, and specifies `value` as the CAS output table name.

- `key` specifies `key` as the ODS table name and also as the CAS output table name.

The ODS table names that you can specify are listed in the section “ODS Table Names” on page 537. You cannot specify the ODS table named OutputCasTables in the `table-spec-list`.

Table names and partial pathnames are discussed under the DISPLAY statement. The DISPLAYOUT statement does not support regular expressions.

You can specify the following options after a slash (/):
**INCLUDEALL**
creates output CAS tables for all display tables. The name of the created output CAS table is the same as the corresponding display table name. If you specify this option, the *table-spec-list* specification is ignored.

**NOREPLACE**
does not replace any existing CAS output table of the same name.

**REPEATED**
replicates all CAS output tables on all nodes.

---

**EFFECT Statement**

```
EFFECT name=effect-type (variables < / options>) ;
```

The EFFECT statement enables you to construct special collections of columns for design matrices. These collections are called *constructed effects* in order to distinguish them from the usual model effects that are formed from continuous or classification variables, as discussed in the section “GLM Parameterization of Classification Variables and Effects” on page 54 in Chapter 3, “Shared Concepts.”

You can specify the following *effect-types*:

- **COLLECTION** specifies a collection effect that defines one or more variables as a single effect that has multiple degrees of freedom. The variables in a collection are considered as a unit for purposes of estimation and inference.
- **MULTIMEMBER | MM** specifies a multimember classification effect whose levels are determined by one or more variables that appear in a CLASS statement.
- **POLYNOMIAL | POLY** specifies a multivariate polynomial effect in the specified numeric variables.
- **SPLINE** specifies a regression spline effect whose columns are univariate spline expansions of one or more variables. A spline expansion replaces the original variable with an expanded or larger set of new variables.

For more information about the syntax of these *effect-types* and how columns of constructed effects are computed, see the section “EFFECT Statement” on page 21 in Chapter 3, “Shared Concepts.”

Table 10.4 summarizes the *options* available in the EFFECT statement.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Collection Effects Options</strong></td>
<td></td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays the constituents of the collection effect</td>
</tr>
<tr>
<td><strong>Multimember Effects Options</strong></td>
<td></td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays the levels of the multimember effect</td>
</tr>
<tr>
<td>NOEFFECT</td>
<td>Requests that observations whose levels are all missing for the multimember variables have 0 values in the corresponding design matrix columns</td>
</tr>
</tbody>
</table>
Chapter 10: The LMXED Procedure

Table 10.4 continued

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>STDIZE</td>
<td>Standardizes the design matrix entries so that each observation has a sum of 1</td>
</tr>
<tr>
<td>WEIGHT=</td>
<td>Specifies the weight variable for the contributions of each classification effect</td>
</tr>
</tbody>
</table>

**Polynomial Effects Options**

- **DEGREE=** Specifies the degree of the polynomial
- **DETAILS** Displays details of the specified polynomial
- **MDEGREE=** Specifies the maximum degree of any variable in a term of the polynomial
- **NOSEPARATE** Treats the polynomial as a single effect with multiple degrees of freedom
- **STANDARDIZE=** Specifies centering and scaling suboptions for the variables that define the polynomial

**Spline Effects Options**

- **BASIS=** Specifies the type of basis (B-spline basis or truncated power function basis) for the spline effect
- **DATABOUNDARY** Uses the extremes of the data as boundary knots for a B-spline basis
- **DEGREE=** Specifies the degree of the spline effect
- **DETAILS** Displays the knots and locations for each spline basis function
- **KNOTMAX=** Requests equally spaced right-side boundary knots starting at the variables’ maximum and ending at the KNOTMAX= value
- **KNOTMETHOD=** Specifies how to construct the knots for the spline effect
- **KNOTMIN=** Requests equally spaced left-side boundary knots starting at the KNOTMIN= value and ending at the variables’ minimum value
- **NATURALCUBIC** Specifies a natural cubic spline basis for the spline effect

**MODEL Statement**

```
MODEL dependent = < fixed-effects > / options ;
```

The MODEL statement names a single dependent variable and the fixed effects, which determine the X matrix of the mixed model. The MODEL statement is required.

An intercept is included in the fixed-effects model by default. If no fixed effects are specified, only this intercept term is fit. The intercept can be removed by using the NOINT option.

Table 10.5 summarizes options in the MODEL statement. These are subsequently discussed in detail in alphabetical order.
Table 10.5  Summary of Important MODEL Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Model Building</strong></td>
<td></td>
</tr>
<tr>
<td>NOINT</td>
<td>Excludes the fixed-effect intercept from the model</td>
</tr>
<tr>
<td><strong>Statistical Computations</strong></td>
<td></td>
</tr>
<tr>
<td>ALPHA=α</td>
<td>Determines the confidence level ((1 - \alpha)) for fixed effects</td>
</tr>
<tr>
<td>DDFM=</td>
<td>Specifies the method for computing the denominator degrees of freedom</td>
</tr>
<tr>
<td><strong>Statistical Output</strong></td>
<td></td>
</tr>
<tr>
<td>CL</td>
<td>Displays confidence limits for fixed-effects parameter estimates</td>
</tr>
<tr>
<td>SOLUTION</td>
<td>Displays fixed-effects parameter estimates</td>
</tr>
</tbody>
</table>

You can specify the following **options** in the MODEL statement after a slash (/).

**ALPHA=α**
sets the confidence level to be \(1 - \alpha\) for each confidence interval of the fixed-effects parameters, where \(\alpha\) must be a number between 0 and 1. By default, ALPHA=0.05.

**CL**
constructs that \(t\)-type confidence limits for each of the fixed-effects parameter estimates. The confidence level is 0.95 by default; this can be changed by specifying the ALPHA= option.

**DDFM=NONE | RESIDUAL**
specifies the method for computing the denominator degrees of freedom for the tests of fixed effects.

**NONE**
does not apply any denominator degrees of freedom. PROC LIMIXED then essentially assumes that infinite degrees of freedom are available in the calculation of \(p\)-values. The \(p\)-values for \(t\) tests are then identical to \(p\)-values that are derived from the standard normal distribution.

**RESIDUAL**
performs all tests by using the residual degrees of freedom, \(n - \text{rank}(X)\), where \(n\) is the number of observations used.

By default, DDFM=RESIDUAL.

**NOINT**
requests that no intercept be included in the model. (An intercept is included by default.)

**SOLUTION**
produces a solution for the fixed-effects parameters. Using notation from the section “Linear Mixed Models Theory” on page 525, the fixed-effects parameter estimates are \(\hat{\beta}\) and their approximate standard errors are the square roots of the diagonal elements of \((X'\hat{\Sigma}^{-1}X)^{-1}\).

Along with the estimates and their approximate standard errors, a \(t\) statistic is computed as the estimate divided by its standard error. The Pr > |t| column contains the two-tailed \(p\)-value that corresponds to the \(t\) statistic and associated degrees of freedom. You can use the **CL** option to request confidence
intervals for all of the parameters; they are constructed around the estimate by using a radius that is the product of the standard error times a percentage point from the $t$ distribution.

### OPTIMIZATION Statement

```
OPTIMIZATION <options> ;
```

The OPTIMIZATION statement specifies the technique and relevant specifications that are used in nonlinear optimization of REML and ML functions.

You can specify the following `options`. The effect of some `options` depends on the technique that is specified in the TECHNIQUE= option.

- **ABSCONV=**
  Specifies an absolute function convergence criterion. For minimization, termination requires $f(\psi^{(k)}) \leq r$, where $\psi$ is the vector of parameters in the optimization and $f(\cdot)$ is the objective function. The default value of $r$ is the negative square root of the largest double-precision value, which serves only as a protection against overflows.

- **ABSFCONV=**
  Specifies an absolute function difference convergence criterion. For all techniques except Nelder-Mead simplex (TECHNIQUE=NMSIMP), termination requires a small change of the function value in successive iterations:
  $$|f(\psi^{(k-1)}) - f(\psi^{(k)})| \leq r$$
  Here, $\psi$ denotes the vector of parameters that participate in the optimization and $f(\cdot)$ is the objective function. The same formula is used for the Nelder-Mead simplex technique, but $\psi^{(k)}$ is defined as the vertex that has the lowest function value and $\psi^{(k-1)}$ is defined as the vertex that has the highest function value in the simplex. By default, ABSFCONV=0.

- **ABSGCONV=**
  Specifies an absolute gradient convergence criterion. Termination requires the maximum absolute gradient element to be small:
  $$\max_j |g_j(\psi^{(k)})| \leq r$$
  Here, $\psi$ denotes the vector of parameters that participate in the optimization and $g_j(\cdot)$ is the gradient of the objective function with respect to the $j$ parameter. This criterion is not used by the Nelder-Mead simplex technique. By default, ABSGCONV=1E–5.

- **FCONV=**
  Specifies a relative function convergence criterion. For all techniques except the Nelder-Mead simplex technique, termination requires a small relative change of the function value in successive iterations:
  $$\frac{|f(\psi^{(k)}) - f(\psi^{(k-1)})|}{|f(\psi^{(k-1)})|} \leq r$$
  Here, $\psi$ denotes the vector of parameters that participate in the optimization and $f(\cdot)$ is the objective function. The same formula is used for the Nelder-Mead simplex technique, but $\psi^{(k)}$ is defined as
the vertex that has the lowest function value and \( \theta^{(k-1)} \) is defined as the vertex that has the highest function value in the simplex.

The default is \( r = 10^{-\text{FDIGITS}} \), where \( \text{FDIGITS} = -\log_{10}(\epsilon) \) and \( \epsilon \) is the machine precision.

**FCONV2** =\( r \)

specifies a second function convergence criterion. For all techniques except the Nelder-Mead simplex technique, termination requires a small predicted reduction of the objective function:

\[
df^{(k)} \approx f(\beta^{(k)}) - f(\beta^{(k)} + s^{(k)})
\]

The predicted reduction

\[
df^{(k)} = -g^{(k)} s^{(k)} - \frac{1}{2} s^{(k)} H^{(k)} s^{(k)}
\]

\[
= -\frac{1}{2} s^{(k)} g^{(k)} \leq r
\]

is computed by approximating the objective function \( f \) by the first two terms of the Taylor series and substituting the Newton step,

\[
s^{(k)} = -[H^{(k)}]^{-1} g^{(k)}
\]

For the Nelder-Mead simplex technique, termination requires a small standard deviation of the function values of the \( p+1 \) simplex vertices \( \beta_l^{(k)} \), \( l = 0, \ldots, p \),

\[
\sqrt{\frac{1}{n+1} \sum_l \left[ f(\beta_l^{(k)}) - \overline{f}(\beta^{(k)}) \right]^2} \leq r
\]

where \( \overline{f}(\beta^{(k)}) = \frac{1}{p+1} \sum_l f(\beta_l^{(k)}) \). If there are \( p_{\text{act}} \) boundary constraints active at \( \beta^{(k)} \), the mean and standard deviation are computed only for the \( n+1-p_{\text{act}} \) unconstrained vertices.

The default value is \( r = 1\text{E}–6 \) for the Nelder-Mead simplex technique and \( r = 0 \) otherwise.

**GCONV** =\( r \)

specifies a relative gradient convergence criterion. For all techniques except the conjugate-gradient technique and the Nelder-Mead simplex technique, termination requires that the normalized predicted function reduction be small,

\[
\frac{g(\psi^{(k)}) [H^{(k)}]^{-1} g(\psi^{(k)})}{|f(\psi^{(k)})|} \leq r
\]

Here, \( \psi \) denotes the vector of parameters that participate in the optimization, \( f(\cdot) \) is the objective function, and \( g(\cdot) \) is the gradient. For the conjugate-gradient technique (where a reliable Hessian estimate \( H \) is not available), the following criterion is used:

\[
\frac{\| g(\psi^{(k)}) \|_2^2}{\| g(\psi^{(k)}) - g(\psi^{(k-1)}) \|_2} \leq r
\]

This criterion is not used by the Nelder-Mead simplex technique. The default value is \( r = 1\text{E}–8 \). 

GCONV2=r

specifies another relative gradient convergence criterion. For Newton-Raphson with ridging and
Newton-Raphson techniques, the following criterion of Browne (1982) is used:

$$\max_j \frac{|g_j(\beta^{(k)})|}{\sqrt{f(\beta^{(k)})H^{(k)}_{j,j}}} \leq r$$

This criterion is not used by the other techniques. By default, GCONV2=0.

MAXFUNC=n

specifies the maximum number n of function calls in the optimization process. The optimization
can terminate only after completing a full iteration. Therefore, the number of function calls that are
performed can exceed n. The default values are as follows, depending on the optimization technique:

- When TECHNIQUE=TRUREG, NRRIDG, or NEWRAP, MAXFUNC=123 by default.
- When TECHNIQUE=QUANEW or DBLDOG, MAXFUNC=500 by default.
- When TECHNIQUE=CONGRA, MAXFUNC=1,000 by default.
- When TECHNIQUE=NMSIMP, MAXFUNC=3,000 by default.

MAXITER=n

specifies the maximum number n of iterations in the optimization process. These default values also
apply when n is specified as a missing value. The default values are as follows, depending on the
optimization technique:

- When TECHNIQUE=TRUREG, NRRIDG, or NEWRAP, MAXITER=50 by default.
- When TECHNIQUE=QUANEW or DBLDOG, MAXITER=200 by default.
- When TECHNIQUE=CONGRA, MAXITER=400 by default.
- When TECHNIQUE=NMSIMP, MAXITER=1,000 by default.

MAXTIME=r

specifies an upper limit of r seconds of CPU time for the optimization process. The default value is the
largest floating-point double representation of your computer. The time specified by the MAXTIME=
option is checked only once at the end of each iteration. Therefore, the actual run time can be longer
than r.

MINITER=n

specifies the minimum number of iterations. The default value is 0. If you request more iterations than
are needed for convergence to a stationary point, the optimization algorithms can behave strangely.
For example, the effect of rounding errors can prevent the algorithm from continuing for the required
number of iterations.

TECHNIQUE=keyword

TECH=keyword

specifies the optimization technique for obtaining restricted maximum likelihood estimation (REML)
or maximum likelihood estimation (ML). There is no algorithm for optimizing general nonlinear
functions that always finds the global optimum for a general nonlinear optimization problem in a
reasonable amount of time. Because no single optimization technique is always superior to others,
PROC LIMIXED provides a variety of optimization techniques that work well in various circumstances.
You can specify any of the following keywords:
CONGRA performs a conjugate-gradient optimization.
DBLDOG performs a version of double-dogleg optimization.
NEWRAP performs a Newton-Raphson optimization that combines a line-search algorithm with ridging.
NMSIMP performs a Nelder-Mead simplex optimization.
NONE performs no optimization.
NRRIDG performs a Newton-Raphson optimization with ridging.
QUANEW performs a dual quasi-Newton optimization.
TRUREG performs a trust-region optimization.

By default, TECHNIQUE=NRRIDG when DMMETHOD=DENSE. Otherwise, TECHNIQUE=QUANEW by default.

XCONV=r
specifies the relative parameter convergence criterion, which depends on the technique that is specified in the TECHNIQUE= option as follows:

- For all techniques except the Nelder-Mead simplex technique, termination requires a small relative parameter change in subsequent iterations:

  \[
  \frac{\max_j |\psi_j^{(k)} - \psi_j^{(k-1)}|}{\max(|\psi_j^{(k)}|, |\psi_j^{(k-1)}|)} \leq r
  \]

- For the Nelder-Mead simplex technique, the same formula is used, but \( \psi_j^{(k)} \) is defined as the vertex that has the lowest function value and \( \psi_j^{(k-1)} \) is defined as the vertex that has the highest function value in the simplex.

The default value is \( r = 1E^{-8} \) for the NMSIMP technique and \( r = 0 \) otherwise.

OUTPUT Statement

```plaintext
OUTPUT OUT=CAS-libref.data-table
COPYVARS=(variables)>
< keyword =name >=...< keyword =name >= ;
```

The OUTPUT statement creates a data table that contains observationwise statistics, which are computed after the model is fitted. The variables in the input data table are not included in the output data table in order to avoid data duplication for large data tables; however, variables that you specify in the COPYVARS= option are included.

The output statistics are computed on the basis of the parameter estimates. By default, the predicted values are computed for observations that have missing response values. You can use the OUTPUT statement to obtain the predicted values for new observations.

You must specify the following option:
OUT=CAS-libref.data-table
names the output data table for PROC LMIXED to use. You must specify this option before any other options. CAS-libref.data-table is a two-level name, where
CAS-libref refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to where the data table is to be stored, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about CAS-libref, see the section “Using CAS Sessions and CAS Engine Librefs” on page 491.
data-table specifies the name of the output data table.

You can also specify the following syntax elements:

**ALLSTATS**

**ALLSTAT** adds all available statistics to the output data table.

**ALPHA=number** specifies the significance level for the construction of confidence intervals in the output data table. The confidence level is 1 – number, where number must be between 0 and 1. By default, number is equal to the value of the ALPHA= option in the MODEL statement, or 0.05 if that option is not specified.

**COPYVAR=variable**

**COPYVARS=(variables)** copies one or more variables from the input data table to the output data table.

**keyword <= name** specifies the statistics to include in the output data table and optionally names the new variables that contain the statistics. Specify a keyword for each desired statistic (see the following list of keywords), followed optionally by an equal sign and a variable to contain the statistic.

If you specify **keyword=name**, the new variable that contains the requested statistic has the specified name. If you omit the optional =name after a keyword, then a default name is used.

You can specify the following values for **keyword** to request statistics:

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
<th>Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRED</td>
<td>Linear predictor</td>
<td>( \hat{\eta} = \mathbf{x}'\hat{\beta} + \mathbf{z}'\hat{\gamma} )</td>
</tr>
<tr>
<td>PREDPA</td>
<td>Marginal linear predictor</td>
<td>( \hat{\eta}_m = \mathbf{x}'\hat{\beta} )</td>
</tr>
<tr>
<td>STDERR</td>
<td>Standard error of linear predictor</td>
<td>( \sqrt{\text{Var}[\hat{\eta} - \mathbf{z}'\gamma]} )</td>
</tr>
<tr>
<td>STDERRPA</td>
<td>Standard error of marginal linear predictor</td>
<td>( \sqrt{\text{Var}[\hat{\eta}_m]} )</td>
</tr>
<tr>
<td>RESIDUAL</td>
<td>Residual</td>
<td>( r = y - \hat{\eta} )</td>
</tr>
<tr>
<td>RESIDUALPA</td>
<td>Marginal residual</td>
<td>( r_m = y - \hat{\eta}_m )</td>
</tr>
<tr>
<td>PEARSON</td>
<td>Pearson-type residual</td>
<td>( r / \sqrt{\text{Var}[y</td>
</tr>
<tr>
<td>PEARSONPA</td>
<td>Marginal Pearson-type residual</td>
<td>( r_m / \sqrt{\text{Var}[y]} )</td>
</tr>
</tbody>
</table>
Table 10.6 (continued)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
<th>Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>STUDENT</td>
<td>Studentized residual</td>
<td>( r / \sqrt{\text{Var}[r]} )</td>
</tr>
<tr>
<td>STUDENTPA</td>
<td>Studentized marginal residua</td>
<td>( r_m / \sqrt{\text{Var}[r_m]} )</td>
</tr>
<tr>
<td>LCL</td>
<td>Lower prediction limit for linear predictor</td>
<td></td>
</tr>
<tr>
<td>LCLPA</td>
<td>Lower confidence limit for marginal linear predictor</td>
<td></td>
</tr>
<tr>
<td>UCL</td>
<td>Upper prediction limit for linear predictor</td>
<td></td>
</tr>
<tr>
<td>UCLPA</td>
<td>Upper confidence limit for marginal linear predictor</td>
<td></td>
</tr>
<tr>
<td>VARIANCE</td>
<td>Conditional variance of response variable</td>
<td>( \text{Var}[y</td>
</tr>
<tr>
<td>VARIANCEPA</td>
<td>Marginal variance of response variable</td>
<td>( \text{Var}[y] )</td>
</tr>
</tbody>
</table>

PARMS Statement

**PARMS** < (value-list) . . . < / options > ;

The PARMS statement specifies initial values for the covariance parameters, or it requests a grid search over several values of these parameters. You must specify the values in the order in which they appear in the “Covariance Parameter Estimates” table.

The *value-list* specification can take any of several forms:

- \( m \) a single value
- \( m_1, m_2, \ldots, m_n \) several values
- \( m \) to \( n \) a sequence in which \( m \) equals the starting value, \( n \) equals the ending value, and the increment equals 1
- \( m \) to \( n \) by \( i \) a sequence in which \( m \) equals the starting value, \( n \) equals the ending value, and the increment equals \( i \)
- \( m_1, m_2 \) to \( m_3 \) mixed values and sequences

Suppose that a model has three covariance parameters that have known values of 2, 1, and 3. You can fix the variance components at these values by using the following statement:

```
parms (2) (1) (3) /hold;
```

The **NOPROFILE** option in the **PROC LMXED** statement suppresses profiling of the residual variance parameter during its calculations, thereby enabling its value to be held at 3 as specified in the PARMS statement.

You can use the PARMS statement to input known parameters.

If you specify more than one set of initial values, the LMXED procedure performs a grid search of the likelihood surface and uses the best point on the grid for subsequent analysis. Specifying a large number of grid points can result in long computing times.
The results from the PARMS statement are the values of the parameters on the specified grid (denoted by CovP1 through CovPn), the residual variance (possibly estimated) for models that have a residual variance parameter, and various functions of the likelihood.

If you specify multiple PARMS statements, the LMIXED procedure uses the first one and ignores the rest.

You can specify the following options in the PARMS statement after a slash (/):

- **HOLD< =order-list >**
- **EQCONS< =order-list >**
- **HOLD< =ALL >**
- **EQCONS< =ALL >**

  holds the values of the covariance parameters whose order is specified in the order-list to the initial values that are specified in the value-list in the RANDOM statement.

  For example, the following statement constrains the first and third covariance parameters to equal 5 and 2, respectively:

  ```
  parms (5) (3) (2) (3) / hold=1,3;
  ```

  If you specify ALL or you do not specify order-list, then all covariance parameters are held to their initial values. When not all parameters are held to their initial values, this is referred to as partial holding.

  Specifying the HOLD= option implies the specification of NOPROFILE option in the PROC LMIXED statement.

  The partial holding cannot retain the values exactly for the TYPE=UN covariance structure. This is because the underlining structure is TYPE=CHOL in the optimization for covariance parameter estimation.

- **LOWERB=value-list**

  specifies the lower boundary constraints on the covariance parameters, where value-list is a list of numbers or missing values (.) separated by commas. You must list the numbers in the order that the LMIXED procedure uses for the covariance parameters, and each number corresponds to a lower boundary constraint. A missing value instructs the LMIXED procedure to use its default constraint. If you do not specify numbers for all the covariance parameters, the LMIXED procedure assumes that the remaining ones are missing.

  This option is useful when you want to constrain the G matrix to be positive definite in order to avoid the more computationally intensive algorithms that would be required when G becomes singular. The corresponding statements for a random coefficients model are as follows:

  ```
  proc lmixed;
  class person;
  model y = time;
  random int time / type=fa0(2) sub=person;
  parms / lowerb=1e-4,.,1e-4;
  run;
  ```
The TYPE=FA0(2) structure specifies a Cholesky root parameterization for the $2 \times 2$ unstructured blocks in $G$. This parameterization ensures that the $G$ matrix is nonnegative definite, and the PARMS statement then ensures that it is positive definite by constraining the two diagonal terms to be greater than or equal to 1E–4.

**NOITER**

requests that no optimization iterations be performed and that the LMIxED procedure use the best value from the grid search to perform inferences.

**PARMSDATA=**`CAS-libref.data-table`

**PDATA=**`CAS-libref.data-table`

reads in covariance parameter values from a data table. *CAS-libref.data-table* is a two-level name, where *CAS-libref* refers to the caslib and session identifier, and *data-table* specifies the name of the input data table. For more information about this two-level name, see the **DATA=** option and the section “Using CAS Sessions and CAS Engine Librefs” on page 491.

The data table should contain either the numerical variables Estimate and RowId or the numerical variables Covp1–Covpq, where $q$ denotes the number of covariance parameters.

If the data table contains the numerical variables Covp1–Covpq and it contains multiple sets of values for covariance parameters, then the LMIxED procedure evaluates the initial objective function for each set of values and commences the optimization step by using the set that has the lowest functional value as the starting values. For example, the following statements request that the objective function be evaluated for three sets of initial values:

```sas
data mycas.data_covp;
  input covp1-covp4;
datalines;
  180 200 170 1000
  170 190 160 900
  160 180 150 800
;proc lmixed;
  class A B C rep;
  model yield = A;
  random rep B C;
  parms / pdata=mycas.data_covp;
run;
```

Another way of specifying the initial values for the covariance parameters is to use the numerical variables Estimate and RowId in the data table. The values in RowId variable indicate the order of the covariance parameters in the “Covariance Parameter Estimates” table.

```sas
data mycas.covparm;
  input rowid estimate;
datalines;
  1 180
  2 200
  3 170
  4 1000
;proc lmixed;
  class A B C;
  model yield = A;
  random rep B C;
  parms / pdata=mycas.covparm;
run;
```
A BY variable is not supported in the PARMSDATA= data table. The data table is processed in its entirety for every BY group and a message is written to the log. The same set of starting values is used by all BY groups.

```plaintext
data mycas.data_covp;
  input covp1-covp4;
datalines;
  180 200 170 1000
;
proc lmixed;
  class A B C rep;
  model yield = A;
  random rep B C;
  parms / pdata=mycas.data_covp;
  by year;
run;
```

**UPPERB=value-list**

specifies upper boundary constraints on the covariance parameters, where the `value-list` specification is a list of numbers or missing values (.) separated by commas. You must list the numbers in the order that the LMIEXED procedure uses for the covariance parameters, and each number corresponds to the upper boundary constraint. A missing value instructs the LMIEXED procedure to use its default constraint. If you do not specify numbers for all of the covariance parameters, the LMIEXED procedure assumes that the remaining ones are missing.

---

**RANDOM Statement**

```
RANDOM random-effects </options> ;
```

The RANDOM statement defines the random effects that constitute the $\mathbf{y}$ vector in the mixed model. You can use this statement to specify traditional variance component models and to specify random coefficients. The random effects can be classification or continuous, and multiple RANDOM statements are allowed.

Using notation from the section “Linear Mixed Models Theory” on page 525, the purpose of the RANDOM statement is to define the $\mathbf{Z}$ matrix of the mixed model, the random effects in the $\mathbf{y}$ vector, and the structure of $\mathbf{G}$. The $\mathbf{Z}$ matrix is constructed exactly as the $\mathbf{X}$ matrix for the fixed effects is constructed, and the $\mathbf{G}$ matrix is constructed to correspond with the effects that constitute $\mathbf{Z}$. The structure of $\mathbf{G}$ is defined by using the **TYPE**= option.

You can specify INTERCEPT (or INT) as a random effect to indicate the intercept. The LMIEXED procedure does not include the intercept in the RANDOM statement by default as it does in the MODEL statement.

**Table 10.7** summarizes the important options in the RANDOM statement. All options are subsequently discussed in alphabetical order.
Table 10.7  Summary of Important RANDOM Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Construction of Covariance Structure</strong></td>
<td></td>
</tr>
<tr>
<td>SUBJECT=</td>
<td>Identifies the subjects in the model</td>
</tr>
<tr>
<td>TYPE=</td>
<td>Specifies the covariance structure</td>
</tr>
<tr>
<td><strong>Statistical Output</strong></td>
<td></td>
</tr>
<tr>
<td>ALPHA=(\alpha)</td>
<td>Determines the confidence level ((1 - \alpha))</td>
</tr>
<tr>
<td>CL</td>
<td>Requests confidence limits for predictors of random effects</td>
</tr>
<tr>
<td>SOLUTION</td>
<td>Displays solutions (\hat{\beta}) of the random effects</td>
</tr>
</tbody>
</table>

You can specify the following *options* in the RANDOM statement after a slash (/).

**ALPHA=\(\alpha\)**

sets the confidence level to be \(1 - \alpha\) for each confidence interval of the random-effects estimates, where \(\alpha\) must be a number between 0 and 1. By default, ALPHA=0.05.

**CL**

constructs \(t\)-type confidence limits for each of the random-effect estimates. The confidence level is 0.95 by default; this can be changed with the ALPHA= option.

**SOLUTION**

produces the solution for the random-effects parameters. Using notation from the section “Linear Mixed Models Theory” on page 525, these estimates are the empirical best linear unbiased predictors (EBLUPs), \(\hat{y}_i = \tilde{G}Z\tilde{V}^{-1}(y - X\hat{\beta})\). They can be useful for comparing the random effects from different experimental units and can also be treated as residuals in performing diagnostics for your mixed model.

The numbers displayed in the SE Pred column of the “Solution for Random Effects” table are not the standard errors of the \(\hat{\beta}\) that are displayed in the Estimate column; rather, they are the standard errors of predictions \(\hat{y}_i - y_i\), where \(\hat{y}_i\) is the \(i\)th EBLUP and \(y_i\) is the \(i\)th random-effect parameter.

**SUBJECT=effect**

identifies the subjects in your mixed model. Complete independence is assumed across subjects; thus, the SUBJECT= option produces a block-diagonal structure in \(G\) that has identical blocks. In fact, specifying an effect is equivalent to nesting all other effects in the RANDOM statement within the effect.

**TYPE= covariance-structure**

specifies the covariance structure of \(G\). Valid values for *covariance-structure* and their descriptions are listed in Table 10.8. Although a variety of structures are available, most applications call for either TYPE=VC or TYPE=UN.

By default, TYPE=VC (a structure that contains variance components), which models a different variance component for each random effect. The TYPE=UN (unstructured) option is useful for correlated random coefficient models. For example, the following statement specifies a random
intercept-slope model that has different variances for the intercept and slope and a covariance between them:

\[
\text{random intercept age / type=un subject=student;}
\]

You can also use TYPE=FA0(2) to request a G estimate that is constrained to be nonnegative definite.

If you are constructing your own columns of Z by using continuous variables, you can use the TYPE=TOEP(1) structure to group them together to have a common variance component. If you want to have different covariance structures in different parts of G, you must use multiple RANDOM statements with different TYPE= options.

In Table 10.8, the Parameters column represents the number of covariance parameters in the structure, t is the overall dimension of the covariance matrix, and 1(A) equals 1 when A is true and 0 otherwise. For example, 1(i = j) equals 1 when i = j and 0 otherwise, and 1(|i - j| < q) equals 1 when |i - j| < q and 0 otherwise. For the TYPE=TOEPH structures, \(\rho_0 = 1\); for the TYPE=UNR structures, \(\rho_{ij} = 1\) for all i.

<table>
<thead>
<tr>
<th>Structure</th>
<th>Description</th>
<th>Parameters</th>
<th>(i, j) element</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANTE(1)</td>
<td>Antedependence</td>
<td>2t - 1</td>
<td>(\sigma_i \sigma_j \prod_{k=1}^{t-1} \rho_k)</td>
</tr>
<tr>
<td>AR(1)</td>
<td>Autoregressive(1)</td>
<td>2</td>
<td>(\sigma^2 \rho^{[i-j]})</td>
</tr>
<tr>
<td>ARH(1)</td>
<td>Heterogeneous AR(1)</td>
<td>t + 1</td>
<td>(\sigma_i \sigma_j \rho^{[i-j]})</td>
</tr>
<tr>
<td>ARMA(1,1)</td>
<td>Autoregressive moving average(1,1)</td>
<td>3</td>
<td>(\sigma^2 [\gamma \rho^{[i-j]} - 11(i \neq j) + 1(i = j)])</td>
</tr>
<tr>
<td>CHOL</td>
<td>Cholesky root</td>
<td>t(t + 1)/2</td>
<td>(l_{ij})</td>
</tr>
<tr>
<td>CS</td>
<td>Compound symmetry</td>
<td>2</td>
<td>(\sigma_1 + \sigma^2 1(i = j))</td>
</tr>
<tr>
<td>CSH</td>
<td>Heterogeneous compound symmetry</td>
<td>i + 1</td>
<td>(\sigma_i \sigma_j [1(i \neq j) + 1(i = j)])</td>
</tr>
<tr>
<td>FA(q)</td>
<td>Factor analytic (FA)</td>
<td>(\frac{q}{2}(2t - q + 1) + t)</td>
<td>(\Sigma_{k=1}^{\min(i,j,q)} \lambda_{i k} \lambda_{j k} + \sigma_i^2 1(i = j))</td>
</tr>
<tr>
<td>FA0(q)</td>
<td>No diagonal FA</td>
<td>(\frac{q}{2}(2t - q + 1))</td>
<td>(\Sigma_{k=1}^{\min(i,j,q)} \lambda_{i k} \lambda_{j k})</td>
</tr>
<tr>
<td>FA1(q)</td>
<td>Equal diagonal FA</td>
<td>(\frac{q}{2}(2t - q + 1) + 1)</td>
<td>(\Sigma_{k=1}^{\min(i,j,q)} \lambda_{i k} \lambda_{j k} + \sigma^2 1(i = j))</td>
</tr>
<tr>
<td>HF</td>
<td>Huynh-Feldt</td>
<td>i + 1</td>
<td>((\sigma_i^2 + \sigma_j^2)/2 + \lambda 1(i \neq j))</td>
</tr>
<tr>
<td>SIMPLE</td>
<td>An alias for VC</td>
<td>q</td>
<td>(\sigma^2 1(i = j)) for the kth effect</td>
</tr>
<tr>
<td>TOEP</td>
<td>Toeplitz</td>
<td>t</td>
<td>(\sigma_{</td>
</tr>
<tr>
<td>TOEP(q)</td>
<td>Banded Toeplitz (TOEP)</td>
<td>q</td>
<td>(\sigma_{i-j+1} 1(</td>
</tr>
<tr>
<td>TOEPH</td>
<td>Heterogeneous TOEP</td>
<td>2t - 1</td>
<td>(\sigma_i \sigma_j \rho_{</td>
</tr>
<tr>
<td>TOEPH(q)</td>
<td>Banded heterogeneous TOEP</td>
<td>i + q - 1</td>
<td>(\sigma_i \sigma_j \rho_{</td>
</tr>
<tr>
<td>UC</td>
<td>Uniform correlation (UC)</td>
<td>2</td>
<td>(\sigma^2 [\rho 1(i \neq j) + 1(i = j)])</td>
</tr>
<tr>
<td>UCH</td>
<td>Heterogeneous UC</td>
<td>t + 1</td>
<td>(\sigma_i \sigma_j [1(i \neq j) + 1(i = j)])</td>
</tr>
<tr>
<td>UN</td>
<td>Unstructured</td>
<td>t(t + 1)/2</td>
<td>(\sigma_{ij})</td>
</tr>
<tr>
<td>UN(q)</td>
<td>Banded</td>
<td>(\frac{q}{2}(2t - q + 1))</td>
<td>(\sigma_{ij} 1(</td>
</tr>
<tr>
<td>UNR</td>
<td>Unstructured correlation</td>
<td>t(t + 1)/2</td>
<td>(\sigma_i \sigma_j \rho_{\max(i,j) \min(i,j)})</td>
</tr>
</tbody>
</table>
Table 10.8  continued

<table>
<thead>
<tr>
<th>Structure</th>
<th>Description</th>
<th>Parameters</th>
<th>(i, j) element</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNR(q)</td>
<td>Banded correlations</td>
<td>$\frac{q}{2}(2t - q + 1)$</td>
<td>$\sigma_i \sigma_j \rho_{\max(i,j)} \min(i,j)$</td>
</tr>
<tr>
<td>VC</td>
<td>Variance components</td>
<td>$q$</td>
<td>$\sigma_k^2(i = j)$ for the kth effect</td>
</tr>
</tbody>
</table>

Table 10.9 lists the covariance types and applicable computing methods (sparse or dense) for each covariance type.

Table 10.9  Covariance Structures

<table>
<thead>
<tr>
<th>Structure</th>
<th>Description</th>
<th>DMMETHOD=</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANTE(1)</td>
<td>Antedependence</td>
<td>DENSE</td>
</tr>
<tr>
<td>AR(1)</td>
<td>Autoregressive(1)</td>
<td>DENSE or SPARSE</td>
</tr>
<tr>
<td>ARH(1)</td>
<td>Heterogeneous AR(1)</td>
<td>DENSE</td>
</tr>
<tr>
<td>ARMA(1,1)</td>
<td>Autoregressive moving average(1,1)</td>
<td>DENSE</td>
</tr>
<tr>
<td>CHOL</td>
<td>Cholesky root</td>
<td>SPARSE</td>
</tr>
<tr>
<td>CS</td>
<td>Compound symmetry</td>
<td>DENSE or SPARSE</td>
</tr>
<tr>
<td>CSH</td>
<td>Heterogeneous compound symmetry</td>
<td>DENSE or SPARSE</td>
</tr>
<tr>
<td>FA(q)</td>
<td>Factor analytic (FA)</td>
<td>DENSE</td>
</tr>
<tr>
<td>FA0(q)</td>
<td>No diagonal FA</td>
<td>DENSE</td>
</tr>
<tr>
<td>FA1(q)</td>
<td>Equal diagonal FA</td>
<td>DENSE</td>
</tr>
<tr>
<td>HF</td>
<td>Huynh-Feldt</td>
<td>DENSE</td>
</tr>
<tr>
<td>SIMPLE</td>
<td>An alias for VC</td>
<td>DENSE or SPARSE</td>
</tr>
<tr>
<td>TOEP(1)</td>
<td>Toeplitz</td>
<td>DENSE or SPARSE</td>
</tr>
<tr>
<td>TOEP(q)</td>
<td>Banded Toeplitz (TOEP)</td>
<td>DENSE</td>
</tr>
<tr>
<td>TOEPH</td>
<td>Heterogeneous TOEP</td>
<td>DENSE</td>
</tr>
<tr>
<td>TOEPH(q)</td>
<td>Banded heterogeneous TOEP</td>
<td>DENSE</td>
</tr>
<tr>
<td>UC</td>
<td>Uniform correlation (UC)</td>
<td>SPARSE</td>
</tr>
<tr>
<td>UCH</td>
<td>Heterogeneous UC</td>
<td>SPARSE</td>
</tr>
<tr>
<td>UN</td>
<td>Unstructured</td>
<td>DENSE or SPARSE</td>
</tr>
<tr>
<td>UN(q)</td>
<td>Banded</td>
<td>DENSE</td>
</tr>
<tr>
<td>UNR</td>
<td>Unstructured correlation</td>
<td>DENSE</td>
</tr>
<tr>
<td>UNR(q)</td>
<td>Banded correlations</td>
<td>DENSE</td>
</tr>
<tr>
<td>VC</td>
<td>Variance components</td>
<td>DENSE or SPARSE</td>
</tr>
</tbody>
</table>

Table 10.10 lists some examples of the structures in Table 10.8.
Table 10.10 Covariance Structure Examples

<table>
<thead>
<tr>
<th>Description</th>
<th>Structure</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>First-order antedependence</td>
<td>ANTE(1)</td>
<td>$\begin{bmatrix} \sigma_1^2 &amp; \sigma_1 \sigma_2 &amp; \sigma_2 \sigma_3 \rho_1 \ \sigma_2 \sigma_1 &amp; \sigma_2^2 &amp; \sigma_2 \sigma_3 \rho_2 \ \sigma_3 \sigma_1 &amp; \sigma_3 \sigma_2 &amp; \sigma_3^2 \end{bmatrix}$</td>
</tr>
<tr>
<td>First-order autoregressive</td>
<td>AR(1)</td>
<td>$\sigma^2 \begin{bmatrix} 1 &amp; \rho &amp; \rho^2 &amp; \rho^3 \ \rho &amp; 1 &amp; \rho &amp; \rho^2 \ \rho^2 &amp; \rho &amp; 1 &amp; \rho \ \rho^3 &amp; \rho^2 &amp; \rho &amp; 1 \end{bmatrix}$</td>
</tr>
<tr>
<td>Heterogeneous autoregressive(1)</td>
<td>ARH(1)</td>
<td>$\begin{bmatrix} \sigma_1^2 &amp; \sigma_1 \sigma_2 &amp; \sigma_2 \sigma_3 &amp; \sigma_3 \sigma_4 \rho_1^3 \ \sigma_2 \sigma_1 &amp; \sigma_2^2 &amp; \sigma_2 \sigma_3 &amp; \sigma_3 \sigma_4 \rho_2^2 \ \sigma_3 \sigma_1 &amp; \sigma_3 \sigma_2 &amp; \sigma_3^2 &amp; \sigma_3 \sigma_4 \rho_3 \ \sigma_4 \sigma_1 &amp; \sigma_4 \sigma_2 &amp; \sigma_4 \sigma_3 &amp; \sigma_4^2 \end{bmatrix}$</td>
</tr>
<tr>
<td>First-order autoregressive moving average</td>
<td>ARMA(1,1)</td>
<td>$\sigma^2 \begin{bmatrix} 1 &amp; \gamma &amp; \gamma \rho &amp; \gamma \rho^2 \ \gamma &amp; 1 &amp; \gamma &amp; \gamma \rho \ \gamma \rho &amp; \gamma &amp; 1 &amp; \gamma \ \gamma \rho^2 &amp; \gamma \rho &amp; \gamma &amp; 1 \end{bmatrix}$</td>
</tr>
<tr>
<td>Cholesky root</td>
<td>CHOL</td>
<td>$\begin{bmatrix} l_{11} &amp; 0 &amp; 0 &amp; 0 \ l_{21} &amp; l_{22} &amp; 0 &amp; 0 \ l_{31} &amp; l_{32} &amp; l_{33} &amp; 0 \ l_{41} &amp; l_{42} &amp; l_{43} &amp; l_{44} \end{bmatrix}$ $\begin{bmatrix} 0 &amp; l_{21} &amp; l_{31} &amp; l_{41} \ 0 &amp; l_{22} &amp; l_{32} &amp; l_{42} \ 0 &amp; 0 &amp; l_{33} &amp; l_{43} \ 0 &amp; 0 &amp; 0 &amp; l_{44} \end{bmatrix}$</td>
</tr>
<tr>
<td>Compound symmetry</td>
<td>CS</td>
<td>$\begin{bmatrix} \sigma^2 + \sigma_1 &amp; \sigma_1 &amp; \sigma_1 &amp; \sigma_1 \ \sigma_1 &amp; \sigma^2 + \sigma_1 &amp; \sigma_1 &amp; \sigma_1 \ \sigma_1 &amp; \sigma_1 &amp; \sigma^2 + \sigma_1 &amp; \sigma_1 \ \sigma_1 &amp; \sigma_1 &amp; \sigma_1 &amp; \sigma^2 + \sigma_1 \end{bmatrix}$</td>
</tr>
<tr>
<td>Heterogeneous compound symmetry</td>
<td>CSH</td>
<td>$\begin{bmatrix} \sigma_1^2 &amp; \sigma_1 \sigma_2 &amp; \sigma_1 \sigma_3 &amp; \sigma_1 \sigma_4 \rho_1^3 \ \sigma_2 \sigma_1 &amp; \sigma_2^2 &amp; \sigma_2 \sigma_3 &amp; \sigma_2 \sigma_4 \rho_2^2 \ \sigma_3 \sigma_1 &amp; \sigma_3 \sigma_2 &amp; \sigma_3^2 &amp; \sigma_3 \sigma_4 \rho_3 \ \sigma_4 \sigma_1 &amp; \sigma_4 \sigma_2 &amp; \sigma_4 \sigma_3 &amp; \sigma_4^2 \end{bmatrix}$</td>
</tr>
<tr>
<td>First-order factor analytic</td>
<td>FA(1)</td>
<td>$\begin{bmatrix} \lambda_1^2 + d_1 &amp; \lambda_1 \lambda_2 &amp; \lambda_1 \lambda_3 &amp; \lambda_1 \lambda_4 \ \lambda_2 \lambda_1 &amp; \lambda_2^2 + d_2 &amp; \lambda_2 \lambda_3 &amp; \lambda_2 \lambda_4 \ \lambda_3 \lambda_1 &amp; \lambda_3 \lambda_2 &amp; \lambda_3^2 + d_3 &amp; \lambda_3 \lambda_4 \ \lambda_4 \lambda_1 &amp; \lambda_4 \lambda_2 &amp; \lambda_4 \lambda_3 &amp; \lambda_4^2 + d_4 \end{bmatrix}$</td>
</tr>
<tr>
<td>Huynh-Feldt</td>
<td>HF</td>
<td>$\begin{bmatrix} \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2} - \lambda &amp; \frac{\sigma_1^2 + \sigma_2^2}{\sigma_1^2 + \sigma_2^2} - \lambda \ \frac{\sigma_1^2 + \sigma_2^2}{\sigma_1^2 + \sigma_2^2} - \lambda &amp; \frac{\sigma_1^2 + \sigma_2^2}{\sigma_1^2 + \sigma_2^2} - \lambda \end{bmatrix}$</td>
</tr>
<tr>
<td>Toeplitz</td>
<td>TOEP</td>
<td>$\begin{bmatrix} \sigma_1^2 &amp; \sigma_1 \sigma_2 &amp; \sigma_1 \sigma_3 \ \sigma_1 \sigma_2 &amp; \sigma_2^2 &amp; \sigma_2 \sigma_3 \ \sigma_1 \sigma_3 &amp; \sigma_2 \sigma_3 &amp; \sigma_3^2 \end{bmatrix}$</td>
</tr>
<tr>
<td>Description</td>
<td>Structure</td>
<td>Example</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-----------</td>
<td>----------------------------------------------</td>
</tr>
</tbody>
</table>
| Toeplitz with two bands         | TOEP(2)   | \[
\begin{bmatrix}
\sigma^2 & \sigma_1 & 0 & 0 \\
\sigma_1 & \sigma^2 & \sigma_1 & 0 \\
0 & \sigma_1 & \sigma^2 & \sigma_1 \\
0 & 0 & \sigma_1 & \sigma^2
\end{bmatrix}
\] |
| Heterogeneous Toeplitz          | TOEPH     | \[
\begin{bmatrix}
\sigma_1^2 & \sigma_1 \sigma_2 \rho_1 & \sigma_1 \sigma_3 \rho_2 & \sigma_1 \sigma_4 \rho_3 \\
\sigma_2 \sigma_1 \rho_1 & \sigma_2^2 & \sigma_2 \sigma_3 \rho_1 & \sigma_2 \sigma_4 \rho_2 \\
\sigma_3 \sigma_1 \rho_2 & \sigma_3 \sigma_2 \rho_1 & \sigma_3^2 & \sigma_3 \sigma_4 \rho_1 \\
\sigma_4 \sigma_1 \rho_3 & \sigma_4 \sigma_2 \rho_2 & \sigma_4 \sigma_3 \rho_1 & \sigma_4^2
\end{bmatrix}
\] |
| Uniform correlation             | UC        | \[
\begin{bmatrix}
1 & \rho & \rho & \rho \\
\rho & 1 & \rho & \rho \\
\rho & \rho & 1 & \rho \\
\rho & \rho & \rho & 1
\end{bmatrix}
\] |
| Heterogeneous UC                | UCH       | \[
\begin{bmatrix}
\sigma_1^2 & \sigma_1 \sigma_2 \rho & \sigma_1 \sigma_3 \rho & \sigma_1 \sigma_4 \rho \\
\sigma_2 \sigma_1 \rho & \sigma_2^2 & \sigma_2 \sigma_3 \rho & \sigma_2 \sigma_4 \rho \\
\sigma_3 \sigma_1 \rho & \sigma_3 \sigma_2 \rho & \sigma_3^2 & \sigma_3 \sigma_4 \rho \\
\sigma_4 \sigma_1 \rho & \sigma_4 \sigma_2 \rho & \sigma_4 \sigma_3 \rho & \sigma_4^2
\end{bmatrix}
\] |
| Unstructured                    | UN        | \[
\begin{bmatrix}
\sigma_1^2 & 0 & 0 & 0 \\
0 & \sigma_2^2 & 0 & 0 \\
0 & 0 & \sigma_3^2 & 0 \\
0 & 0 & 0 & \sigma_4^2
\end{bmatrix}
\] |
| Banded main diagonal            | UN(1)     | \[
\begin{bmatrix}
\sigma_1^2 & \sigma_1 \sigma_2 \rho_1 & \sigma_1 \sigma_3 \rho_31 & \sigma_1 \sigma_4 \rho_41 \\
\sigma_2 \sigma_1 \rho_1 & \sigma_2^2 & \sigma_2 \sigma_3 \rho_32 & \sigma_2 \sigma_4 \rho_42 \\
\sigma_3 \sigma_1 \rho_31 & \sigma_3 \sigma_2 \rho_32 & \sigma_3^2 & \sigma_3 \sigma_4 \rho_43 \\
\sigma_4 \sigma_1 \rho_41 & \sigma_4 \sigma_2 \rho_42 & \sigma_4 \sigma_3 \rho_43 & \sigma_4^2
\end{bmatrix}
\] |
| Unstructured correlations       | UNR       | \[
\begin{bmatrix}
\sigma_B^2 & 0 & 0 & 0 \\
0 & \sigma_B^2 & 0 & 0 \\
0 & 0 & \sigma_{AB}^2 & 0 \\
0 & 0 & 0 & \sigma_{AB}^2
\end{bmatrix}
\] |

The following list provides some further information about these covariance structures:

- **TYPE=ANTE(1)** specifies the first-order antedependence structure (Kenward 1987; Patel 1991; Macchiavelli and Arnold 1994). In Table 10.8, \( \sigma_i^2 \) is the \( i \) variance parameter, and \( \rho_k \) is the \( k \)th autocorrelation parameter that satisfies \( |\rho_k| < 1 \).
  This covariance type is available only when DMMETHOD=DENSE.

- **TYPE=AR(1)** specifies a first-order autoregressive structure. The LMIXED procedure imposes the constraint \( |\rho| < 1 \) for stationarity.
TYPE=ARH(1) specifies a heterogeneous first-order autoregressive structure. As with TYPE=AR(1), the LMIXED procedure imposes the constraint $|\rho| < 1$ for stationarity.

This covariance type is available only when DMMETHOD=DENSE.

TYPE=ARMA(1,1) specifies the first-order autoregressive moving average structure. In Table 10.8, $\rho$ is the autoregressive parameter, $\gamma$ models a moving average component, and $\sigma^2$ is the residual variance. In the notation of Fuller (1976, p. 68), $\rho = \theta_1$ and

$$\gamma = \frac{(1 + b_1 \theta_1)(\theta_1 + b_1)}{1 + b_1^2 + 2b_1 \theta_1}$$

The example in Table 10.10 and $|b_1| < 1$ imply that

$$b_1 = \frac{\beta - \sqrt{\beta^2 - 4\alpha^2}}{2\alpha}$$

where $\alpha = \gamma - \rho$ and $\beta = 1 + \rho^2 - 2\gamma\rho$. The LMIXED procedure imposes the constraints $|\rho| < 1$ and $|\gamma| < 1$ for stationarity, although the resulting covariance matrix is not positive definite for some values of $\rho$ and $\gamma$ in this region. When the estimated value of $\rho$ becomes negative, the computed covariance is multiplied by $\cos(\pi d_{ij})$ to account for the negativity.

This covariance type is available only when DMMETHOD=DENSE.

TYPE=CHOL specifies an unstructured variance-covariance matrix that is parameterized through its Cholesky root. All diagonal values are constrained to be positive. This parameterization guarantees a positive definite covariance matrix. For example, a $2 \times 2$ unstructured covariance matrix can be written as

$$\text{Var}[\xi] = \begin{bmatrix} \sigma_1^2 & \sigma_{21} \\ \sigma_{21} & \sigma_2^2 \end{bmatrix}$$

Without imposing constraints on the three parameters, there is no guarantee that the estimated variance matrix is positive definite. Even if $\sigma_2^2$ and $\sigma_{21}^2$ are nonzero, a large value for $\sigma_{21}$ can lead to a negative eigenvalue of $\text{Var}[\xi]$. The Cholesky root of a positive definite matrix $A$ is a lower triangular matrix $L$ such that $LL' = A$.

The Cholesky root of the preceding $2 \times 2$ matrix can be written as

$$L = \begin{bmatrix} l_{11} & 0 \\ l_{21} & l_{22} \end{bmatrix}$$

The elements of the unstructured variance matrix are then simply $\sigma_1^2 = l_{11}^2$, $\sigma_{21} = l_{21}/l_{11}$, and $\sigma_2^2 = l_{21}^2 + l_{22}^2$. Similar operations yield the generalization to covariance matrices of higher orders.

For example, the following statements model the covariance matrix of each subject as an unstructured matrix:

```r
proc lmmixed;  
   class sub;  
   model y = x;  
   random time / sub=patient type=chol;  
run;
```
The **LMIXED** procedure constrains the diagonal elements of the Cholesky root to be positive. This guarantees that the structure is positive definite.

This covariance type is available only when DMMETHOD=SPARSE.

**TYPE=CS** specifies the compound-symmetry structure, which has constant variance and constant covariance.

**TYPE=CSH** specifies the heterogeneous compound-symmetry structure. This structure has a different variance parameter for each diagonal element, and it uses the square roots of these parameters in the off-diagonal entries. In Table 10.8, $\sigma_i^2$ is the $i$ variance parameter, and $\rho$ is the correlation parameter that satisfies $|\rho| < 1$.

**TYPE=FA($q$)** specifies the factor-analytic structure with $q$ factors (Jennrich and Schluchter 1986). This structure is of the form $\Lambda \Lambda' + D$, where $\Lambda$ is a $t \times q$ rectangular matrix and $D$ is a $t \times t$ diagonal matrix with $t$ different parameters. When $q > 1$, the elements of $\Lambda$ in its upper right corner (that is, the elements in the $i$ row and $j$ column for $j > i$) are set to 0 to fix the rotation of the structure.

This covariance type is available only when DMMETHOD=DENSE.

**TYPE=FA0($q$)** is similar to the FA($q$) structure except that no diagonal matrix $D$ is included. When $q < t$ (that is, when the number of factors is less than the dimension of the matrix), this structure is nonnegative definite but not of full rank. In this situation, you can use this structure for approximating an unstructured $G$ matrix in the RANDOM statement. When $q = t$, you can use this structure to constrain $G$ to be nonnegative definite in the RANDOM statement.

This covariance type is available only when DMMETHOD=DENSE.

**TYPE=FA1($q$)** is similar to the TYPE=FA($q$) structure except that all the elements in $D$ are constrained to be equal. This offers a useful and more parsimonious alternative to the full factor-analytic structure.

This covariance type is available only when DMMETHOD=DENSE.

**TYPE=HF** specifies the Huynh-Feldt covariance structure (Huynh and Feldt 1970). This structure is similar to the **TYPE=CSH** structure in that it has the same number of parameters and heterogeneity along the main diagonal. However, it constructs the off-diagonal elements by taking arithmetic means rather than geometric means.

You can perform a likelihood ratio test of the Huynh-Feldt conditions by running the **LMIXED** procedure twice, once with TYPE=HF and once with TYPE=UN, and then subtracting their respective values of $-2$ times the maximized likelihood.

If the **LMIXED** procedure does not converge under your Huynh-Feldt model, you can specify your own starting values in the PARMs statement. The default MIVQUE(0) starting values can sometimes be poor for this structure. A good choice for starting values is often the parameter estimates that correspond to an initial fit that uses TYPE=CS.

This covariance type is available only when DMMETHOD=DENSE.

**TYPE=SIMPLE** is an alias for **TYPE=VC**.

**TYPE=TOEP<($q$)>** specifies a banded Toeplitz structure. This structure can be viewed as a moving average structure with order equal to $q - 1$. The **TYPE=TOEP** option is a full Toeplitz matrix, which can be viewed as an autoregressive structure with order
equal to the dimension of the matrix. The specification TYPE=TOEP(1) is the same as \( \sigma^2 I \), where \( I \) is an identity matrix, and it can be useful for specifying the same variance component for several effects.

**TYPE=TOEPH<(<q>)>** specifies a heterogeneous banded Toeplitz structure. In Table 10.8, \( \sigma^2_i \) is the \( i \) variance parameter and \( \rho_j \) is the \( j \) correlation parameter that satisfies \( |\rho_j| < 1 \). If you specify the order parameter \( q \), then the LMIXED procedure estimates only the first \( q \) bands of the matrix, setting all higher bands equal to 0. **TYPE=TOEPH(1)** is equivalent to both **TYPE=UN(1)** and **TYPE=UNR(1)**.

This covariance type is available only when **DMMETHOD=DENSE**.

**TYPE=UC** specifies the uniform correlation structure, which has constant variance and constant correlation,

\[
\text{Cov} \left[ \xi_i, \xi_j \right] = \begin{cases} 
\sigma^2 & i = j \\
\sigma^2 \rho & i \neq j 
\end{cases}
\]

Under uniform correlation, the \( \mathbf{G} \) matrix has the form \( \sigma^2 [(1 - \rho) \mathbf{I} + \rho \mathbf{J}] \). The variance \( \sigma^2 \) is constrained to be positive, and the correlation \( \rho \) is constrained to be greater than \(-1/(t-1)\), where \( t \) is the dimension of the structure. These constraints guarantee that the structure is positive definite. This structure is equivalent to the compound-symmetry structure with a better numerical property in terms of optimization.

The uniform correlation structure arises frequently in agriculture and animal sciences.

This covariance type is available only when **DMMETHOD=SPARSE**.

**TYPE=UCH** specifies the heterogeneous uniform correlation structure. This structure has a different variance parameter for each diagonal element, and it uses the square roots of these parameters in the off-diagonal entries. In Table 10.8, \( \sigma^2_i \) is the \( i \)th variance parameter that satisfies \( \sigma^2_i > 0 \), and \( \rho \) is the correlation parameter that satisfies \( \rho > -1/(t-1) \), where \( t \) is the dimension of the structure. This guarantees that the structure is positive definite.

This covariance type is available only when **DMMETHOD=SPARSE**.

**TYPE=UN** specifies a completely general (unstructured) covariance matrix that is parameterized directly in terms of variances and covariances. The variances are constrained to be nonnegative, and the covariances are unconstrained. When **DMMETHOD=DENSE**, this structure is not constrained to be nonnegative definite in order to avoid nonlinear constraints; however, you can use the **TYPE=FA0** structure if you want this constraint to be imposed by a Cholesky factorization. When **DMMETHOD=SPARSE**, this structure is constrained to be nonnegative definite by a Cholesky factorization.

**TYPE=UN(\(q\))** specifies a banded unstructured covariance matrix that is parameterized directly in terms of variances and covariances. The LMIXED procedure estimates only the first \( q \) bands of the matrix, setting all higher bands equal to 0.

This covariance type is available only when **DMMETHOD=DENSE**.

**TYPE=UNR<(<q>)>** specifies a completely general (unstructured) covariance matrix that is parameterized in terms of variances and correlations. This structure fits the same model.
as the TYPE=UN\((q)\) option but with a different parameterization. The \(i\) variance parameter is \(\sigma_i^2\). The parameter \(\rho_{jk}\) is the correlation between the \(j\) and \(k\) measurements; it satisfies \(|\rho_{jk}| < 1\). If you specify the order parameter \(r\), then the LIMIXED procedure estimates only the first \(q\) bands of the matrix, setting all higher bands equal to zero.

This covariance type is available only when DMMETHOD=DENSE.

\textbf{TYPE=VC} specifies standard variance components. This is the default structure for the RANDOM statements. In the RANDOM statement, a distinct variance component is assigned to each effect.

Jennrich and Schluchter (1986) provide general information about the use of covariance structures, and Wolfinger (1996) presents details about many of the heterogeneous structures.

### REPEATED Statement

\textbf{REPEATED} \textit{repeated-effect} < / \textit{options} > ;

The REPEATED statement defines the repeated effect and the residual covariance structure in the mixed model. The residual variance-covariance matrix is denoted as \(\mathbf{R}\). The \textit{repeated-effect} is required and consists entirely of classification variables. The levels of the \textit{repeated-effect} must be different for each observation within a subject in order to avoid the singular \(\mathbf{R}\) matrix.

Table 10.11 summarizes the \textit{options} available in the REPEATED statement.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textbf{Construction of Covariance Structure}</td>
<td></td>
</tr>
<tr>
<td>\texttt{GROUP=}</td>
<td>Defines an effect that specifies heterogeneity in the residual covariance structure</td>
</tr>
<tr>
<td>\texttt{SUBJECT=}</td>
<td>Identifies the subjects in the residual covariance structure</td>
</tr>
<tr>
<td>\texttt{TYPE=}</td>
<td>Specifies the residual covariance structure (the default is VC)</td>
</tr>
<tr>
<td>\textbf{Statistical Output}</td>
<td></td>
</tr>
<tr>
<td>\texttt{R=}</td>
<td>Displays blocks of the estimated (\mathbf{R}) matrix</td>
</tr>
<tr>
<td>\texttt{RC=}</td>
<td>Display the Cholesky root (lower) of blocks of the estimated (\mathbf{R}) matrix</td>
</tr>
<tr>
<td>\texttt{RCI=}</td>
<td>Displays the inverse Cholesky root (lower) of blocks of the estimated (\mathbf{R}) matrix</td>
</tr>
<tr>
<td>\texttt{RCORR=}</td>
<td>Displays the correlation matrix that corresponds to blocks of the estimated (\mathbf{R}) matrix</td>
</tr>
<tr>
<td>\texttt{RI=}</td>
<td>Displays the inverse of blocks of the estimated (\mathbf{R}) matrix</td>
</tr>
</tbody>
</table>

In the current release, the REPEATED statement is supported only when DMMETHOD=SPARSE.

You must specify one or both of the following \textit{options} after a slash (/).
GROUP=effect

defines an effect that specifies heterogeneity in the residual covariance structure. All observations that have the same level of the effect have the same covariance parameters. Each new level of the effect produces a new set of covariance parameters that have the same structure as the original group. You should exercise caution in defining the effect, because strange covariance patterns can result when it is misused. Also, the effect can greatly increase the number of estimated covariance parameters, which can adversely affect the optimization process.

SUBJECT=effect

identifies the subjects in your mixed model. Complete independence is assumed across subjects; therefore, this option produces a block-diagonal structure in $R$ that has identical blocks. The data set is not required to be grouped by the effect. When the effect consists entirely of classification variables, the blocks of $R$ correspond to observations that share the same level of that effect. These blocks are sorted according to this effect as well.

You can also specify the following options after a slash (/).

\( R\langle value-list\rangle \)

displays blocks of the estimated $R$ matrix. The first block that is determined by the SUBJECT= effect is the default displayed block.

The value-list indicates the subjects for which blocks of $R$ are displayed. For example, the following statement displays block matrices for the first, third, and fifth persons:

\[
\text{repeated time / type=un subject=person r=1,3,5;}
\]

See the PARMS statement for the possible forms of value-list.

\( RC\langle value-list\rangle \)

displays the Cholesky root of blocks of the estimated $R$ matrix. The value-list specification is the same as for the R= option.

\( RCi\langle value-list\rangle \)

displays the inverse Cholesky root of blocks of the estimated $R$ matrix. The value-list specification is the same as for the R= option.

\( RCORR\langle value-list\rangle \)

displays the correlation matrix that corresponds to blocks of the estimated $R$ matrix. The value-list specification is the same as for the R= option.

\( RI\langle value-list\rangle \)

produces the inverse of blocks of the estimated $R$ matrix. The value-list specification is the same as for the R= option.

\( TYPE=\text{covariance-structure} \)

specifies the structure of the residual variance-covariance matrix $R$. The SUBJECT= option defines the blocks of $R$, and the TYPE= option specifies the structure of these blocks. In the current release, the valid values of covariance-structure are AR(1), CHOL, CS, CSH, UN, and VC.
For more information about these covariance structure types, see the description in the section “RAN-
DOM Statement” on page 514.

**WEIGHT Statement**

**WEIGHT** variable ;

The **variable** in the WEIGHT statement is used as a weight to perform a weighted analysis of the data. Observations that have nonpositive or missing weights are not included in the analysis. If a WEIGHT statement is not included, all observations that are used in the analysis are assigned a weight of 1.

The WEIGHT statement is supported only when DMMETHOD=SPARSE.

**Details: L MIXED Procedure**

**Linear Mixed Models Theory**

This section provides an overview of a likelihood-based approach to linear mixed models. This approach simplifies and unifies many common statistical analyses, including those that involve repeated measures, random effects, and random coefficients. The basic assumption is that the data are linearly related to unobserved multivariate normal random variables. For extensions to nonlinear and nonnormal situations, see the documentation of the GLIMMIX and NLMIXED procedures in the SAS/STAT User’s Guide. Additional theory and examples are provided in Littell et al. (2006); Verbeke and Molenberghs (1997, 2000); and Burdick and Graybill (1992).

**Matrix Notation for General Linear Model**

Suppose that you observe \( n \) data points \( y_1, \ldots, y_n \) and you want to explain them by using \( n \) values for each of \( p \) explanatory variables \( x_{11}, \ldots, x_{1p}, x_{21}, \ldots, x_{2p}, \ldots, x_{n1}, \ldots, x_{np} \). The \( x_{ij} \) values can be either regression-type continuous variables or dummy variables that indicate class membership. The general linear model for this setup is

\[
y_i = \sum_{j=1}^{p} x_{ij} \beta_j + \epsilon_i \quad i = 1, \ldots, n
\]

where \( \beta_1, \ldots, \beta_p \) are unknown fixed-effects parameters to be estimated and \( \epsilon_1, \ldots, \epsilon_n \) are unknown independent and identically distributed normal (Gaussian) random variables with mean 0 and variance \( \sigma^2 \).

The preceding equations can be written simultaneously by using vectors and a matrix, as follows:

\[
\begin{bmatrix}
  y_1 \\
  y_2 \\
  \vdots \\
  y_n
\end{bmatrix}
= 
\begin{bmatrix}
  x_{11} & x_{12} & \cdots & x_{1p} \\
  x_{21} & x_{22} & \cdots & x_{2p} \\
  \vdots & \vdots & \ddots & \vdots \\
  x_{n1} & x_{n2} & \cdots & x_{np}
\end{bmatrix}
\begin{bmatrix}
  \beta_1 \\
  \beta_2 \\
  \vdots \\
  \beta_p
\end{bmatrix}
+ 
\begin{bmatrix}
  \epsilon_1 \\
  \epsilon_2 \\
  \vdots \\
  \epsilon_n
\end{bmatrix}
\]
For convenience, simplicity, and extendability, this entire system is written as

$$y = X\beta + \epsilon$$

where $y$ denotes the vector of observed $y_i$'s, $X$ is the known matrix of $x_{ij}$'s, $\beta$ is the unknown fixed-effects parameter vector, and $\epsilon$ is the unobserved vector of independent and identically distributed Gaussian random errors.

In addition to denoting data, random variables, and explanatory variables in the preceding fashion, the subsequent sections make use of basic matrix operators such as transpose ($'$), inverse ($^{-1}$), generalized inverse ($\sim$), determinant ($| \cdot |$), and matrix multiplication. See Searle (1982) for details about these and other matrix techniques.

**Formulation of the Mixed Model**

The previous general linear model is certainly a useful one (Searle 1971), and it is the one fitted by the GLM procedure. However, the distributional assumption about $\epsilon$ is often too restrictive. The mixed model extends the general linear model by allowing a more flexible specification of the covariance matrix of $\epsilon$. In other words, it allows for both correlation and heterogeneous variances, although normality is still assumed.

The mixed model is written as

$$y = X\beta + Z\gamma + \epsilon$$

where all components are the same as in the general linear model except for the addition of the known design matrix, $Z$, and the vector of unknown random-effects parameters, $\gamma$. The matrix $Z$ can contain either continuous or dummy variables, just like $X$. The name *mixed model* comes from the fact that the model contains both fixed-effects parameters, $\beta$, and random-effects parameters, $\gamma$. See Henderson (1990) and Searle, Casella, and McCulloch (1992) for history of the development of the mixed model.

A key assumption in the preceding analysis is that $\gamma$ and $\epsilon$ are normally distributed with

$$E\begin{bmatrix} y \\ \epsilon \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\text{Var}\begin{bmatrix} y \\ \epsilon \end{bmatrix} = \begin{bmatrix} G & 0 \\ 0 & R \end{bmatrix}$$

Therefore, the variance of $y$ is $V = ZG' + R$. You can model $V$ by setting up the random-effects design matrix $Z$ and specifying covariance structures for $G$ and $R$.

Note that this is a general specification of the mixed model, in contrast to many texts and articles that discuss only simple random effects. Simple random effects are a special case of the general specification in which $Z$ contains dummy variables, $G$ contains variance components in a diagonal structure, and $I$ denotes the identity matrix. The general linear model is a further special case with $Z = 0$.

The following example illustrates the most common formulation of the general linear mixed model.

**Example: Split-Plot Design**

The split-plot design involves two experimental treatment factors, $A$ and $B$, and two different sizes of experimental units to which they are applied (Winer 1971; Snedecor and Cochran 1980; Milliken and Johnson 1992; Steel, Torrie, and Dickey 1997). The levels of $A$ are randomly assigned to the larger-sized experimental units, called *whole plots*, whereas the levels of $B$ are assigned to the smaller-sized experimental units, the
subplots. The subplots are assumed to be nested within the whole plots, so that a whole plot consists of a cluster of subplots and a level of A is applied to the entire cluster.

Such an arrangement is often necessary by nature of the experiment; the classic example is the application of fertilizer to large plots of land and different crop varieties that are planted in subdivisions of the large plots. For this example, fertilizer is the whole-plot factor A and variety is the subplot factor B.

This example is a split-plot design for which the whole plots are arranged in a randomized block design. The appropriate PROC LMMIXED statements are as follows:

```plaintext
proc lmmixed;
  class a b block;
  model y = a b a*b;
  random block a*block;
run;
```

Here

\[ R = \sigma^2 I_{24} \]

and X, Z, and G have the following forms:

\[
X = \begin{bmatrix}
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
\vdots & \vdots & \vdots & \vdots \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
\end{bmatrix}
\]
where $\sigma_B^2$ is the variance component for Block and $\sigma_{AB}^2$ is the variance component for $A*Block$. Changing the RANDOM statement as follows fits the same model, but with $Z$ and $G$ sorted differently:

```
random int a / subject=block;
```
Estimating Covariance Parameters in the Mixed Model

Estimation is more difficult in the mixed model than in the general linear model. Not only do you have $\beta$ as in the general linear model, but you also have unknown parameters in $\gamma$, $G$, and $R$. Least squares is no longer the best method. Generalized least squares (GLS) is more appropriate; it minimizes

$$(y - X\beta)'V^{-1}(y - X\beta)$$

However, GLS requires knowledge of $V$ and therefore knowledge of $G$ and $R$. When such information is lacking, one approach is to use an estimated GLS, in which you insert some reasonable estimate for $V$ into the minimization problem. The goal thus becomes to find a reasonable estimate of $G$ and $R$. 
In many situations, the best approach is to use likelihood-based methods, which exploit the assumption that \( y \) and \( \epsilon \) are normally distributed (Hartley and Rao 1967; Patterson and Thompson 1971; Harville 1977; Laird and Ware 1982; Jennrich and Schluchter 1986). The LMIXED procedure implements two likelihood-based methods: maximum likelihood (ML) and restricted (residual) maximum likelihood (REML). A favorable theoretical property of ML and REML is that they accommodate data that are missing at random (Rubin 1976; Little 1995).

The LMIXED procedure constructs an objective function that is associated with ML or REML and maximizes it over all unknown parameters. Using calculus, it is possible to reduce this maximization problem to one over only the parameters in \( G \) and \( R \). The corresponding log-likelihood functions are as follows:

**ML**:

\[
l(G, R) = -\frac{1}{2} \log |V| - \frac{1}{2} r'V^{-1}r - \frac{n}{2} \log(2\pi)
\]

**REML**:

\[
l_R(G, R) = -\frac{1}{2} \log |V| - \frac{1}{2} \log |X'V^{-1}X| - \frac{1}{2} r'V^{-1}r - \frac{n - p}{2} \log(2\pi)
\]

where \( r = y - X(X'V^{-1}X)^{-1}X'V^{-1}y \), \( p \) is the rank of \( X \), and \( n \) is the total number of observations. By default, the LMIXED procedure minimizes a normalized form of \( -2 \) times these functions by using a ridge-stabilized Newton-Raphson algorithm. Lindstrom and Bates (1988) provide reasons for preferring Newton-Raphson to the expectation-maximum (EM) algorithm that is described in Dempster, Laird, and Rubin (1977) and Laird, Lange, and Stram (1987); in addition, they provide analytical details for implementing a QR-decomposition approach to the problem. Wolfinger, Tobias, and Sall (1994) present the sweep-based algorithms that are implemented in the LMIXED procedure. You can change the optimization technique by specifying the `TECHNIQUE=` option in the OPTIMIZATION statement.

One advantage of using the Newton-Raphson algorithm is that the second derivative matrix of the objective function that is evaluated at the optima is available upon completion. Denote this matrix as \( H \); then the asymptotic theory of maximum likelihood (Serfling 1980) shows that \( 2H^{-1} \) is an asymptotic variance-covariance matrix of the estimated parameters of \( G \) and \( R \).

The residual variance \( \sigma^2 \) is profiled out of the likelihood by default. This means solving analytically for the optimal \( \sigma^2 \) and plugging this expression back into the likelihood formula (Wolfinger, Tobias, and Sall 1994). This reduces the number of optimization parameters by 1 and can improve convergence properties.

### Computing Methodologies for REML in the Mixed Model

As a high-performance procedure, PROC LMIXED can use the distributed and multithreading capabilities of your system (for more information, see the section “Computational Method” on page 534) to speed up the fitting process in some linear mixed models that use the restricted maximum likelihood (REML) and the maximum likelihood (ML) approaches. This section illustrates how to use parallel computing or the sparse matrix technique to compute the objective function in REML.

The restricted log likelihood is defined as

\[
l_R(G, R) = -\frac{1}{2} \log |V| - \frac{1}{2} \log |X'V^{-1}X| - \frac{1}{2} r'V^{-1}r - \frac{n - p}{2} \log(2\pi)
\]

where \( r = y - X(X'V^{-1}X)^{-1}X'V^{-1}y \), \( p \) is the rank of \( X \), and \( n \) is the total number of observations. The computational complexity resides mostly in calculating these components: \( \log |V| \), \( \log |X'V^{-1}X| \), and \( r'V^{-1}r \). In directly handling a matrix \( V \) of dimension \( n \times n \), a large value of \( n \) poses difficulties in both storage and computation.
In some situations, the computation of these $V$-related quantities can be greatly simplified, thus enabling parallel computing. When you have a common subject effect, the data can be subsetted and worked with in a smaller size. This leads to simplification of the REML function components,

$$\log |V| = \sum_{i=1}^{s} \log |V_i|$$

$$\log |X'V^{-1}X| = \log |\sum_{i=1}^{s} (X_i'V_i^{-1}X_i)|$$

where $s$ is the number of levels in the common subject effect and $V_i$ is the covariance matrix of response variables that belong to subject level $i$. Because summations enable parallel accumulation of the individual pieces, they are ideal for multithreaded and distributed computing. The $r'V^{-1}r$ term is calculated as a by-product of the determinant of $\sum_{i=1}^{s} (X_i'V_i^{-1}X_i)$. This method is called the dense method and can be specified in the DMMETHOD=DENSE option in the PROC LMIXED statement.

If the model does not have a common subject effect, the restricted log likelihood is computed using the mixed model equations (Henderson 1984). The three components can be rewritten as:

$$\log |V| + \log |X'V^{-1}X| = \log |M| + \log |G| + \log |R|$$

$$r'V^{-1}r = y'R^{-1}y - \hat{\beta}'X'R^{-1}y - \hat{\gamma}'Z'R^{-1}y$$

where $M$ is

$$\begin{bmatrix}
X'R^{-1}X & X'R^{-1}Z \\
Z'R^{-1}X & Z'R^{-1}Z + G^{-1}
\end{bmatrix}$$

Generally speaking, matrix $M$ is large and computing it requires substantial memory. Fortunately, $M$ is often very sparse in many real-world problems if you work with categorical variables that have many levels (sparsity in $X$ and $Z$). The LMIXED procedure uses sparse matrix techniques to compute the objective function; these techniques can be substantially more efficient than the dense approach. This method is called the sparse method and can be specified in the DMMETHOD=SPARSE option in the PROC LMIXED statement.

### Estimating Fixed and Random Effects in the Mixed Model

The maximum likelihood (ML) and the restricted maximum likelihood (REML) methods provide estimates of $G$ and $R$, which are denoted $\hat{G}$ and $\hat{R}$, respectively. To obtain estimates of $\beta$ and predicted values of $y$, the standard method is to solve the mixed model equations (Henderson 1984):

$$\begin{bmatrix}
X'\hat{R}^{-1}X & X'\hat{R}^{-1}Z \\
Z'\hat{R}^{-1}X & Z'\hat{R}^{-1}Z + \hat{G}^{-1}
\end{bmatrix}\begin{bmatrix}
\hat{\beta} \\
\hat{\gamma}
\end{bmatrix} = \begin{bmatrix}
X'\hat{R}^{-1}y \\
Z'\hat{R}^{-1}y
\end{bmatrix}$$

Assume that the covariance matrix of response variables is $\hat{V}$. Then the solutions can be written as

$$\hat{\beta} = (X'\hat{V}^{-1}X)^{-1}X'\hat{V}^{-1}y$$

$$\hat{\gamma} = \hat{G}Z'\hat{V}^{-1}(y - X\hat{\beta})$$

and have connections with empirical Bayes estimators (Laird and Ware 1982; Carlin and Louis 1996). Note that the $\gamma$ are random variables and not parameters (unknown constants) in the model. Technically, determining values for $\gamma$ from the data is thus a prediction task, whereas determining values for $\beta$ is an estimation task.
Statistical Properties

If $G$ and $R$ are known, $\hat{b}$ is the best linear unbiased estimator (BLUE) of $\beta$, and $\hat{y}$ is the best linear unbiased predictor (BLUP) of $y$ (Searle 1971; Harville 1988, 1990; Robinson 1991; McLean, Sanders, and Stroup 1991). Here, “best” means it has minimum mean square error. The covariance matrix of $(\hat{\beta} - \beta, \hat{y} - y)$ is

$$C = \begin{bmatrix} X'R^{-1}X & X'R^{-1}Z \\ Z'R^{-1}X & Z'R^{-1}Z + G^{-1} \end{bmatrix}$$

where $-$ denotes a generalized inverse (Searle 1971).

However, $G$ and $R$ are usually unknown and are estimated by using one of the aforementioned methods. These estimates, $\hat{G}$ and $\hat{R}$, are therefore simply substituted into the preceding expression to obtain

$$\hat{C} = \begin{bmatrix} X'\hat{R}^{-1}X & X'\hat{R}^{-1}Z \\ Z'\hat{R}^{-1}X & Z'\hat{R}^{-1}Z + \hat{G}^{-1} \end{bmatrix}$$

as the approximate variance-covariance matrix of $(\hat{\beta} - \beta, \hat{y} - y)$. In this case, the BLUE and BLUP acronyms no longer apply, but the word empirical is often added to indicate such an approximation. The appropriate acronyms thus become EBLUE and EBLUP.

McLean and Sanders (1988) show that $\hat{C}$ can also be written as

$$\hat{C} = \begin{bmatrix} \hat{C}_{11} & \hat{C}_{21} \\ \hat{C}_{21} & \hat{C}_{22} \end{bmatrix}$$

where

$$\hat{C}_{11} = (X'\hat{V}^{-1}X)^{-1}$$
$$\hat{C}_{21} = -\hat{G}Z'\hat{V}^{-1}X\hat{C}_{11}$$
$$\hat{C}_{22} = (Z'\hat{R}^{-1}Z + \hat{G}^{-1})^{-1} - \hat{C}_{21}X'\hat{V}^{-1}Z\hat{G}$$

Note that $\hat{C}_{11}$ is the familiar estimated generalized least squares formula for the variance-covariance matrix of $\hat{\beta}$.

Common Subject Effect

PROC LMIXED is designed to fit a subclass of large (in number of observations and number of levels in a subject) linear mixed models. The procedure takes advantage of distributed and multithreading capabilities of your system by working on subsets of the data in parallel, and it aggregates results in the estimation process, which can lead to substantial speed improvements. This subclass of models requires that the $G$ matrix have a block-diagonal structure, meaning that the data can be grouped according to a subject effect. This subject effect is referred to as a “common subject effect.”

The common subject effect should be the only subject effect that is used in a program, and it must be specified as the same SUBJECT= effect in all RANDOM statements.

For example, a model that has a single RANDOM statement with a SUBJECT= specification has a common subject effect by definition:
random intercept age / subject=student;

You can have multiple RANDOM statements. The same SUBJECT= effect is the common subject:

random time / subject=student*school;
random intercept age / subject=student*school;

In contrast, a single RANDOM statement without SUBJECT= specification, as shown in the following statements, implies that the model does not have a common subject effect:

random time;

More than one subject effect appearing in a program, as shown in the following statements, implies that the model does not have a common subject effect:

random time / subject=student*school;
random intercept age / subject=student;

If you have a variance component model (TYPE=VC), in some situations, you can use a random effect as a subject effect, which leads to the same model specification but with a common subject effect. For example, the following two RANDOM statements are equivalent (the second specification has a common subject effect in time):

random time;

random int / subject=time;

Another equivalency is that

random time / subject=student*school;
random intercept age / subject=student;

can be rewritten as

random time*school / subject=student;
random intercept age / subject=student;

where the student becomes the common subject effect.
Computational Method

Distributed Computing

Distributed computing refers to the use of multiple autonomous computers that communicate through a secure network. Distributed computing solves computational problems by dividing them into many tasks, each of which is solved by one or more computers. Each computer in this distributed environment is called a node.

Multithreading

The LMIXED procedure allocates data to different threads and calculates design matrices by accumulating the contributions from all threads. The LMIXED procedure also computes the log likelihood by accumulating the contributions from all threads. In addition, operations on matrices such as sweeps might be multithreaded if the matrices are of sufficient size to realize performance benefits from managing multiple threads for the particular matrix operation.

For more information about how the LMIXED procedure uses threads, see the section “Multithreading” on page 81 in Chapter 3, “Shared Concepts.”

Displayed Output

The following sections describe the output that the LMIXED procedure produces. The output is organized into various tables, which are discussed in the order of their appearance.

Model Information

The “Model Information” table describes the model, some of the variables it involves, and the method used in fitting it. The “Model Information” table also has a row labeled Fixed Effects SE Method. This row describes the method that was used to compute the approximate standard errors for the fixed-effects parameter estimates and related functions of them.

Class Level Information

The “Class Level Information” table lists the levels of every variable that is specified in the CLASS statement.

Dimensions

The “Dimensions” table lists the sizes of relevant matrices. This table can be useful in determining the requirements for CPU time and memory.

Number of Observations

The “Number of Observations” table shows the number of observations that were read from the data table and the number of observations that were used in fitting the model.
Optimization Information

The “Optimization Information” table displays important details about the optimization process.

The number of parameters that are updated in the optimization equals the number of parameters in this table minus the number of equality constraints. The number of constraints is displayed if you fix covariance parameters by specifying the HOLD= option in the PARMS statement. The LMMIXED procedure also lists the number of upper and lower boundary constraints. The LMMIXED procedure might impose boundary constraints for certain parameters, such as variance components and correlation parameters. If you specify the HOLD= option in the PARMS statement, covariance parameters have an upper and lower boundary equal to the parameter value.

Descriptive Statistics

The “Descriptive Statistics” table lists simple statistics such as means and standard deviations for the dependent variable, for each covariate in the MODEL statement, and for the weight variable in the WEIGHT statement.

Iteration History

The “Iteration History” table describes the optimization of the restricted log likelihood or log likelihood. The function to be minimized (the objective function) is $-2l$ for ML and $-2l_R$ for REML; the column name of the objective function in the “Iteration History” table is “-2 Log Like” for ML and “-2 Res Log Like” for REML. The minimization is performed by using a ridge-stabilized Newton-Raphson algorithm, and the rows of this table describe the iterations that this algorithm takes in order to minimize the objective function.

The Evaluations column of the “Iteration History” table tells how many times the objective function is evaluated during each iteration.

The Criterion column of the “Iteration History” table is, by default, a relative Hessian convergence quantity,

$$\frac{g_k^T H_k^{-1} g_k}{|f_k|}$$

where $f_k$ is the value of the objective function at iteration $k$, $g_k$ is the gradient (first derivative) of $f_k$, and $H_k$ is the Hessian (second derivative) of $f_k$. If $H_k$ is singular, then the LMMIXED procedure uses the following relative quantity:

$$\frac{g_k^T g_k}{|f_k|}$$

To prevent division by $|f_k|$, specify the ABSGCONV= option in the OPTIMIZATION statement. To use a relative function or gradient criterion, specify the FCONV= or GCONV= option, respectively.

The Hessian criterion is considered superior to function and gradient criteria because it measures orthogonality rather than lack of progress (Bates et al. 1987). Provided that the initial estimate is feasible and the maximum number of iterations is not exceeded, the Newton-Raphson algorithm is considered to have converged when the criterion is less than the tolerance specified in the FCONV= or GCONV= option in the OPTIMIZATION statement (or less than the default tolerance value of 1E–8). If convergence is not achieved, the LMMIXED procedure displays the estimates of the parameters at the last iteration.

A convergence criterion that is missing indicates that a boundary constraint has been dropped; it is usually not a cause for concern.
Convergence Status

The “Convergence Status” table displays the status of the iterative estimation process at the end of the optimization. The status appears as a message in the listing, and this message is repeated in the log. The ODS object “ConvergenceStatus” also contains several nonprinting columns that can be helpful in checking the success of the iterative process, in particular during batch processing. The Status variable takes on the value 0 for a successful convergence (even if the Hessian matrix is not positive definite). The values 1 and 2 of the Status variable indicate lack of convergence and infeasible initial parameter values, respectively. The variable pdG can be used to check whether the G matrix is positive definite.

The “Convergence Status” table is not produced for models that are not fit iteratively, such as for models that have no random effects or when the NOITER option is in effect.

Covariance Parameter Estimates

The “Covariance Parameter Estimates” table contains the estimates of the parameters in G and R. (See the section “Estimating Covariance Parameters in the Mixed Model” on page 529.) Their values are labeled in the table along with Subject information if applicable. The estimates are displayed in the Estimate column and are the results of either the REML or the ML estimation method.

Fit Statistics

The “Fit Statistics” table provides some statistics about the estimated mixed model. Expressions for \(-2\) times the log likelihood are provided in the section “Estimating Covariance Parameters in the Mixed Model” on page 529. If the log likelihood is an extremely large number, then the LMIXED procedure has deemed the estimated V matrix to be singular. In this case, all subsequent results should be viewed with caution.

In addition, the “Fit Statistics” table lists three information criteria: AIC, AICC, and BIC. All these criteria are in smaller-is-better form and are described in Table 10.12.

<table>
<thead>
<tr>
<th>Table 10.12 Information Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Criterion</strong></td>
</tr>
<tr>
<td>AIC</td>
</tr>
<tr>
<td>AICC</td>
</tr>
<tr>
<td>BIC</td>
</tr>
</tbody>
</table>

Here \( \ell \) denotes the maximum value of the (possibly restricted) log likelihood; \( d \) is the dimension of the model; and \( n \) equals the number of effective subjects as displayed in the “Dimensions” table unless this value equals 1, in which case \( n \) equals the number of levels of the first random effect that is specified in the first RANDOM statement or the number of levels of the interaction of the first random effect with the noncommon subject effect that is specified in the first RANDOM statement. If the number of effective subjects equals 1 and you have no RANDOM statements, then \( n \) equals the number of valid observations for maximum likelihood estimation or equals \( n - p \) for restricted maximum likelihood estimation, where \( p \) equals the rank of \( X \). For AICC (a finite-sample corrected version of AIC), \( n^* \) equals the number of valid observations for maximum likelihood estimation and \( n - p \) equals the number of valid observations for restricted maximum likelihood estimation, unless this number is less than \( d + 2 \), in which case it equals \( d + 2 \). When \( n = 0 \), the value of the BIC is \(-2\ell\). For restricted likelihood estimation, \( d \) equals \( q \), the effective number of estimated covariance parameters. For maximum likelihood estimation, \( d \) equals \( q + p \).
Timing Information

If you specify the TIMING option in the PROC LMMIXED statement, the procedure also produces a “Timing” table in which the elapsed time for each main task of the procedure is displayed.

OutputCasTables Table

The OutputCasTables table is a special table that has information about each CAS table that is created during a CAS action execution. The information for each CAS table consists of the CAS table name, the caslib in which the table resides, and the number of columns and rows in the CAS table. Because this table is not a typical ODS table that contains analytical results, you cannot include it in the table-spec-list in the DISPLAYOUT statement.

ODS Table Names

Each table that the LMMIXED procedure creates has a name associated with it, and you must use this name to refer to the table when you use ODS statements. These names are listed in Table 10.13.

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Required Statement</th>
<th>Required Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>CholR</td>
<td>Cholesky root of blocks of the estimated R matrix</td>
<td>REPEATED</td>
<td>RC</td>
</tr>
<tr>
<td>ClassInfo</td>
<td>Level information from the CLASS</td>
<td>CLASS</td>
<td></td>
</tr>
<tr>
<td>CollectionLevelInfo</td>
<td>Level information for collection effects</td>
<td>EFFECT with COLLECTION effect-type</td>
<td>DETAILS</td>
</tr>
<tr>
<td>ConvergenceStatus</td>
<td>Convergence status</td>
<td>Default output</td>
<td></td>
</tr>
<tr>
<td>CovParms</td>
<td>Estimated covariance parameters</td>
<td>Default output</td>
<td></td>
</tr>
<tr>
<td>Dimensions</td>
<td>Dimensions of the model</td>
<td>Default output</td>
<td></td>
</tr>
<tr>
<td>FitStatistics</td>
<td>Fit statistics</td>
<td>Default output</td>
<td></td>
</tr>
<tr>
<td>InvCholR</td>
<td>Inverse Cholesky root of blocks of the estimated R matrix</td>
<td>REPEATED</td>
<td>RCI</td>
</tr>
<tr>
<td>InvR</td>
<td>Inverse of blocks of the estimated R matrix</td>
<td>REPEATED</td>
<td>RI</td>
</tr>
<tr>
<td>IterHistory</td>
<td>Iteration history</td>
<td>Default output</td>
<td></td>
</tr>
<tr>
<td>MMLevelInfo</td>
<td>Level information for multimember effects</td>
<td>EFFECT with MULTIMEMBER effect-type</td>
<td>DETAILS</td>
</tr>
<tr>
<td>MMEq</td>
<td>Mixed model equations</td>
<td>PROC LMMIXED</td>
<td>MMEQ</td>
</tr>
<tr>
<td>ModelInfo</td>
<td>Model information</td>
<td>Default output</td>
<td></td>
</tr>
<tr>
<td>Table Name</td>
<td>Description</td>
<td>Required Statement</td>
<td>Required Option</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------------------------------------------------</td>
<td>--------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>NObs</td>
<td>Number of observations read and used</td>
<td>Default output</td>
<td></td>
</tr>
<tr>
<td>OptInfo</td>
<td>Optimization information</td>
<td>Default output</td>
<td></td>
</tr>
<tr>
<td>OutputCasTables</td>
<td>See the section “OutputCasTables Table” on page 537</td>
<td>OUTPUT</td>
<td>DISPLAYOUT</td>
</tr>
<tr>
<td>OverallANOVA</td>
<td>ANOVA table for model without random effects</td>
<td>Default output for fixed models</td>
<td></td>
</tr>
<tr>
<td>ParameterEstimates</td>
<td>Fixed-effects solution vector</td>
<td>MODEL</td>
<td>SOLUTION</td>
</tr>
<tr>
<td>ParmSearch</td>
<td>Parameter search values</td>
<td>PARMS</td>
<td></td>
</tr>
<tr>
<td>PolyDetails</td>
<td>Number of variables and columns, polynomial degree, and standardization method</td>
<td>EFFECT with POLYNOMIAL effect-type</td>
<td>DETAILS</td>
</tr>
<tr>
<td>Ranks</td>
<td>Rank of designed matrix X</td>
<td>PROC L MIXED</td>
<td>RANKS</td>
</tr>
<tr>
<td>R</td>
<td>Blocks of the estimated R matrix</td>
<td>REPEATED</td>
<td>R=</td>
</tr>
<tr>
<td>RCorr</td>
<td>Correlation matrix from blocks of the estimated R matrix</td>
<td>REPEATED</td>
<td>R Corr=</td>
</tr>
<tr>
<td>SimpleStatistics</td>
<td>Descriptive statistics for dependent variable and covariate variables</td>
<td>PROC L MIXED</td>
<td>SIMPLE</td>
</tr>
<tr>
<td>SolutionR</td>
<td>Random-effects solution vector</td>
<td>RANDOM</td>
<td>SOLUTION</td>
</tr>
<tr>
<td>SplineKnots</td>
<td>Knot and boundary knot values</td>
<td>EFFECT with SPLINE effect-type</td>
<td>BASIS=BSPLINE and DETAILS</td>
</tr>
<tr>
<td>Timing</td>
<td>Timing breakdown by task</td>
<td>PROC L MIXED</td>
<td>TIMING</td>
</tr>
<tr>
<td>TPFSplineDetails</td>
<td>Truncated power function spline basis details</td>
<td>EFFECT with SPLINE effect-type</td>
<td>BASIS=TPF or BA-SIS=NATURALCUBIC and DETAILS</td>
</tr>
</tbody>
</table>

### Examples: LMIXED Procedure

#### Example 10.1: Mixed Model Analysis of Microarray Data

Microarray experiments are an advanced genomic technique that is used in the discovery of new treatments for diseases. Microarray analysis enables tens of thousands of genes to be detected in a single DNA sample. A microarray is a glass slide or membrane that has been spotted or “arrayed” with DNA fragments
Example 10.1: Mixed Model Analysis of Microarray Data

Example 10.1: Mixed Model Analysis of Microarray Data

(oligonucleotides) that represent specific genes. The response of the gene that is detected by a spot is proportional to the intensity of fluorescence associated with that spot. These gene responses can indicate associations with disease conditions, but they can also be affected by systematic biases and different treatments such as sex and genotypes. Statistical models for microarray data attempt to assess the significance and magnitude of gene effects across treatments while adjusting for these systematic biases and to evaluate the significance of differences between treatments.

There are two statistical approaches frequently used in mixed model analysis for microarray data. The first approach is to fit multiple gene-specific models to data that are normalized for systematic biases (Wolfinger et al. 2001; Gibson and Wolfinger 2004). This approach is based on assuming that the biases are independent from the gene effects. If this assumption is untenable, then a second approach fits a single model that combines both the systematic biases and the gene effects (Kerr, Martin, and Churchill 2000; Churchill 2002; Littell et al. 2006). When the number of genes is very large (several hundreds to tens of thousands), this is an analysis for which the sparse matrix approach that the LMIXED procedure implements is well suited.

The following SAS statements simulate a microarray experiment with a so-called loop design structure, which is commonly used in such studies. There are 500 genes, each gene occurs in six arrays, and each array has two dyes. You are interested in the best five and worst five genes based on the best linear unbiased predicted values (BLUP) of these 500 genes.

```sas
%let narray = 6;
%let ndye = 2;
%let nrow = 4;
%let ngene = 500;
%let ntrt = 6;
%let npin = 4;
%let ndip = 4;
%let no = %eval(&ndye*&nrow*&ngene);
%let tno = %eval(&narray*&no);

data mycas.microarray;
  keep Gene MArray Dye Trt Pin Dip log2i;
  array PinDist{&tno};
  array DipDist{&tno};
  array GeneDist{&tno};
  array ArrayEffect{&narray};
  array ArrayGeneEffect{%eval(&narray*&ngene)};
  array ArrayDipEffect{%eval(&narray*&ndip)};
  array ArrayPinEffect{%eval(&narray*&npin)};

  do i = 1 to &tno;
    PinDist{i} = 1 + int(&npin*ranuni(12345));
    DipDist{i} = 1 + int(&ndip*ranuni(12345));
    GeneDist{i} = 1 + int(&ngene*ranuni(12345));
  end;

  igene = 0;
  idip = 0;
  ipin = 0;
  do i = 1 to &narray;
    ArrayEffect{i} = sqrt(0.014)*rannor(12345);
    ArrayGeneEffect{i} = %eval(&ngene*ranuni(12345));
    ArrayDipEffect{i} = %eval(&ndip*ranuni(12345));
    ArrayPinEffect{i} = %eval(&npin*ranuni(12345));
  end;
```

There are two statistical approaches frequently used in mixed model analysis for microarray data. The first approach is to fit multiple gene-specific models to data that are normalized for systematic biases (Wolfinger et al. 2001; Gibson and Wolfinger 2004). This approach is based on assuming that the biases are independent from the gene effects. If this assumption is untenable, then a second approach fits a single model that combines both the systematic biases and the gene effects (Kerr, Martin, and Churchill 2000; Churchill 2002; Littell et al. 2006). When the number of genes is very large (several hundreds to tens of thousands), this is an analysis for which the sparse matrix approach that the LMIXED procedure implements is well suited.

The following SAS statements simulate a microarray experiment with a so-called loop design structure, which is commonly used in such studies. There are 500 genes, each gene occurs in six arrays, and each array has two dyes. You are interested in the best five and worst five genes based on the best linear unbiased predicted values (BLUP) of these 500 genes.

```sas
%let narray = 6;
%let ndye = 2;
%let nrow = 4;
%let ngene = 500;
%let ntrt = 6;
%let npin = 4;
%let ndip = 4;
%let no = %eval(&ndye*&nrow*&ngene);
%let tno = %eval(&narray*&no);

data mycas.microarray;
  keep Gene MArray Dye Trt Pin Dip log2i;
  array PinDist{&tno};
  array DipDist{&tno};
  array GeneDist{&tno};
  array ArrayEffect{&narray};
  array ArrayGeneEffect{%eval(&narray*&ngene)};
  array ArrayDipEffect{%eval(&narray*&ndip)};
  array ArrayPinEffect{%eval(&narray*&npin)};

  do i = 1 to &tno;
    PinDist{i} = 1 + int(&npin*ranuni(12345));
    DipDist{i} = 1 + int(&ndip*ranuni(12345));
    GeneDist{i} = 1 + int(&ngene*ranuni(12345));
  end;

  igene = 0;
  idip = 0;
  ipin = 0;
  do i = 1 to &narray;
    ArrayEffect{i} = sqrt(0.014)*rannor(12345);
    ArrayGeneEffect{i} = %eval(&ngene*ranuni(12345));
    ArrayDipEffect{i} = %eval(&ndip*ranuni(12345));
    ArrayPinEffect{i} = %eval(&npin*ranuni(12345));
  end;
```
do j = 1 to &ngene;
   igene = igene+1;
   ArrayGeneEffect{igene} = sqrt(0.0017)*rannor(12345);
end;

do j = 1 to &ndip;
   idip = idip + 1;
   ArrayDipEffect{idip} = sqrt(0.0033)*rannor(12345);
end;

do j = 1 to &npin;
   ipin = ipin + 1;
   ArrayPinEffect{ipin} = sqrt(0.037)*rannor(12345);
end;

i = 0;
do MArray = 1 to &narray;
do Dye = 1 to &ndye;
do Row = 1 to &nrow;
do k = 1 to &ngene;
   if MArray=1 and Dye = 1 then do;
      Trt = 0;
      trtc = 0;
   end;
   else do;
      if trtc >= &no then trtc = 0;
      if trtc = 0 then do;
         Trt = Trt + 1;
         if Trt > &ntrt then do;
            Trt = 0;
            trtc = 0;
         end;
      end;
      trtc = trtc + 1;
   end;
i = i + 1;
Pin = PinDist{i};
Dip = DipDist{i};
Gene = GeneDist{i};
a = ArrayEffect(MArray);
ag = ArrayGeneEffect((MArray-1)*&ngene+Gene);
ad = ArrayDipEffect((MArray-1)*&ndip+Dip);
ap = ArrayPinEffect((MArray-1)*&npin+Pin);
log2i = 1 +
   + Dye
   + Trt
   + Gene/1000.0
   + Dye*Gene/1000.0
   + Trt*Gene/1000.0
   + Pin
   + a
   + ag
   + ad
   + ap
   + sqrt(0.02)*rannor(12345);
Example 10.1: Mixed Model Analysis of Microarray Data

A linear mixed model for fitting the log intensity data $Y_{ijkmn}$ from such a design is described by Littell et al. (2006) as follows:

$$Y_{ijkmn} = \mu + \lambda_i + \tau_j + \delta_k + (\tau \lambda)_{ij} + (\delta \lambda)_{ik} + p_r + a_m + (a \lambda)_{im} + d(a)_{mn} + (ap)_{mr} + e_{ijklmnop}$$

You can use the LMIXED procedure with the following statements to fit this model:

```sas
ods select Dimensions covParms SolutionR;
proc lmixed data=mycas.microarray dmmethod=sparse;
class marray dye trt gene pin dip;
model log2i = dye trt gene dye*gene trt*gene pin;
random int gene dip pin/subject=marray s;
ods output solutionR=BLUPs;
run;
```

By default, because the program has a RANDOM statement with a SUBJECT= effect, PROC LMIXED uses the dense method (see the section “Common Subject Effect” on page 532) in the restricted maximum likelihood (REML) estimation. However, because the design matrices are excessively large with $X$ having 4,513 columns and $Z$ having 3,054 columns (see table Output 10.1.1), the program can potentially exceed your computer’s memory capacity. Here the sparse method (DMMETHOD=SPARSE) is used to improve the handling of these large and sparse design matrices.

**Output 10.1.1 Mixed Model Dimensions**

The LMIXED Procedure

<table>
<thead>
<tr>
<th>Dimensions</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>G-side Covariance Parameters</td>
<td>4</td>
</tr>
<tr>
<td>R-side Covariance Parameters</td>
<td>1</td>
</tr>
<tr>
<td>Columns in $X$</td>
<td>4513</td>
</tr>
<tr>
<td>Columns in $Z$ per Subject</td>
<td>509</td>
</tr>
<tr>
<td>Subjects (Blocks in V)</td>
<td>6</td>
</tr>
</tbody>
</table>
Output 10.1.2 shows the “Covariance Parameter Estimation” table, which lists the estimations of covariances across each microarray.

Output 10.1.2  Covariance Parameter Estimation

<table>
<thead>
<tr>
<th>Cov Parm</th>
<th>Subject</th>
<th>Estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>MArray</td>
<td>0.003093</td>
</tr>
<tr>
<td>Gene</td>
<td>MArray</td>
<td>0.001901</td>
</tr>
<tr>
<td>Dip</td>
<td>MArray</td>
<td>0.001924</td>
</tr>
<tr>
<td>Pin</td>
<td>MArray</td>
<td>0.03371</td>
</tr>
<tr>
<td>Residual</td>
<td></td>
<td>0.02008</td>
</tr>
</tbody>
</table>

The following program ranks genes on the basis of their BLUP estimates.

```sas
ods output off;
proc sort data=BLUPs(firstobs=2 obs=501);
   by descending Estimate;
run;

data BLUPs; set BLUPs;
   Rank = _N_; run;
ods output on;
proc print data=BLUPs;
   where ((Rank <= 5) | (Rank >= 496));
   var Gene Estimate;
run;
```

The best five genes and worst five genes with their BLUP estimates are shown in Output 10.1.3.

Output 10.1.3  Highest and Lowest Gene BLUPs

<table>
<thead>
<tr>
<th>Obs</th>
<th>Gene</th>
<th>Estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>198</td>
<td>0.06753</td>
</tr>
<tr>
<td>2</td>
<td>268</td>
<td>0.06070</td>
</tr>
<tr>
<td>3</td>
<td>354</td>
<td>0.05124</td>
</tr>
<tr>
<td>4</td>
<td>489</td>
<td>0.04883</td>
</tr>
<tr>
<td>5</td>
<td>478</td>
<td>0.04558</td>
</tr>
<tr>
<td>496</td>
<td>383</td>
<td>-0.04642</td>
</tr>
<tr>
<td>497</td>
<td>333</td>
<td>-0.04684</td>
</tr>
<tr>
<td>498</td>
<td>351</td>
<td>-0.04846</td>
</tr>
<tr>
<td>499</td>
<td>54</td>
<td>-0.05353</td>
</tr>
<tr>
<td>500</td>
<td>144</td>
<td>-0.05541</td>
</tr>
</tbody>
</table>
References


# Chapter 11
The LOGSELECT Procedure

## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview: LOGSELECT Procedure</td>
<td>548</td>
</tr>
<tr>
<td>PROC LOGSELECT Features</td>
<td>549</td>
</tr>
<tr>
<td>PROC LOGSELECT Compared with Other SAS Procedures</td>
<td>549</td>
</tr>
<tr>
<td>Using CAS Sessions and CAS Engine Librefs</td>
<td>551</td>
</tr>
<tr>
<td>Getting Started: LOGSELECT Procedure</td>
<td>551</td>
</tr>
<tr>
<td>Binary Logistic Regression</td>
<td>551</td>
</tr>
<tr>
<td>Syntax: LOGSELECT Procedure</td>
<td>558</td>
</tr>
<tr>
<td>PROC LOGSELECT Statement</td>
<td>558</td>
</tr>
<tr>
<td>BY Statement</td>
<td>563</td>
</tr>
<tr>
<td>CLASS Statement</td>
<td>563</td>
</tr>
<tr>
<td>CODE Statement</td>
<td>563</td>
</tr>
<tr>
<td>DISPLAY Statement</td>
<td>564</td>
</tr>
<tr>
<td>DISPLAYOUT Statement</td>
<td>565</td>
</tr>
<tr>
<td>EFFECT Statement</td>
<td>566</td>
</tr>
<tr>
<td>FREQ Statement</td>
<td>567</td>
</tr>
<tr>
<td>MODEL Statement</td>
<td>567</td>
</tr>
<tr>
<td>OUTPUT Statement</td>
<td>572</td>
</tr>
<tr>
<td>PARTITION Statement</td>
<td>577</td>
</tr>
<tr>
<td>SELECTION Statement</td>
<td>578</td>
</tr>
<tr>
<td>WEIGHT Statement</td>
<td>579</td>
</tr>
<tr>
<td>Details: LOGSELECT Procedure</td>
<td>580</td>
</tr>
<tr>
<td>Missing Values</td>
<td>580</td>
</tr>
<tr>
<td>Response Distributions</td>
<td>580</td>
</tr>
<tr>
<td>Log-Likelihood Functions</td>
<td>581</td>
</tr>
<tr>
<td>Existence of Maximum Likelihood Estimates</td>
<td>583</td>
</tr>
<tr>
<td>The LASSO Method of Model Selection</td>
<td>584</td>
</tr>
<tr>
<td>Partition Fit Statistics</td>
<td>584</td>
</tr>
<tr>
<td>Model Fit and Assessment Statistics</td>
<td>585</td>
</tr>
<tr>
<td>Predicted Probabilities and Regression Diagnostics</td>
<td>589</td>
</tr>
<tr>
<td>Joint Tests and Type 3 Tests</td>
<td>592</td>
</tr>
<tr>
<td>Multithreading</td>
<td>592</td>
</tr>
<tr>
<td>Optimization Algorithms</td>
<td>593</td>
</tr>
<tr>
<td>Displayed Output</td>
<td>594</td>
</tr>
<tr>
<td>ODS Table Names</td>
<td>598</td>
</tr>
<tr>
<td>ODS Graphics</td>
<td>600</td>
</tr>
<tr>
<td>Examples: LOGSELECT Procedure</td>
<td>601</td>
</tr>
</tbody>
</table>
Overview: LOGSELECT Procedure

The LOGSELECT procedure fits and performs model selection for logistic regression models in SAS Viya, including binary, binomial, and multinomial response models.

The models that PROC LOGSELECT supports can contain main effects that consist of both continuous and classification variables and interaction effects of these variables. The models can also include constructed effects such as splines. The procedure offers a number of effect-selection methods, including stepwise methods and modern LASSO methods. It also offers extensive capabilities for customizing the model selection by using a wide variety of selection and stopping criteria, from computationally efficient significance-level-based criteria to modern, computationally intensive validation-based criteria. PROC LOGSELECT also provides a variety of logistic regression diagnostics that are conditional on the selected model.

Logistic regression analysis is often used to investigate the relationship between discrete responses and a set of explanatory variables. The LOGSELECT procedure fits logistic regression models in the broader sense; the procedure permits several link functions and can handle ordinal and nominal response data that have more than two response categories (multinomial data).

The logistic model shares a common feature with a more general class of linear models: a function \( g = g(\mu) \) of the mean of the response variable is assumed to be linearly related to the explanatory variables. Because the mean \( \mu \) implicitly depends on the stochastic behavior of the response and the explanatory variables are assumed to be fixed, the function \( g \) provides the link between the random (stochastic) component and the systematic (deterministic) component of the response variable \( Y \). For this reason, Nelder and Wedderburn (1972) refer to \( g(\mu) \) as a link function. The LOGSELECT procedure fits a broad class of binary response models of the form

\[
g(\pi \mid x) = \alpha + \beta' x
\]

where the link functions that are available in PROC LOGSELECT and that are widely used in practice are the logit, probit, log-log, and complementary log-log functions, and the predicted probability of an event, \( \pi \), is the mean \( \mu \) of the response variable. One advantage of the logit function over other link functions is that differences on the logistic scale are interpretable regardless of whether the data are sampled prospectively or retrospectively (McCullagh and Nelder 1989, Chapter 4).

If your response \( Y \) has more than two values and they can be ordered (for example, \( Y \in \{1, \ldots, J - 1, J\} \)), then the LOGSELECT procedure uses the preceding link functions and fits ordinal response models of the form

\[
g(\pi_j \mid x) = \alpha_j + \beta' x, \quad j = 1, \ldots, J - 1
\]

where \( \pi_j = \Pr(Y \leq j) \) are cumulative probabilities of the ordered response categories.
If your response \( Y \) has more than two values and they have no natural ordering, then the LOGSELECT procedure fits a generalized or baseline-category logit model, which has the form

\[
\log \left( \frac{\Pr(Y = j \mid x)}{\Pr(Y = J \mid x)} \right) = \alpha_j + \beta_j^T x, \quad j = 1, \ldots, J - 1
\]

where the \( \beta_1, \ldots, \beta_{J-1} \) are \( J-1 \) vectors of slope parameters. These models are a special case of the discrete choice or conditional logit models introduced by McFadden (1974).

Texts that discuss logistic regression include Agresti (2013); Allison (2012); Collett (2003); Cox and Snell (1989); Hosmer and Lemeshow (2013); Stokes, Davis, and Koch (2012).

**PROC LOGSELECT Features**

The LOGSELECT procedure estimates the parameters of a logistic regression model by using maximum likelihood techniques. It also does the following:

- provides model-building syntax with the `CLASS`, `EFFECT`, and effect-based `MODEL` statements, which are familiar from SAS/STAT analytic procedures (in particular, the GLM, LOGISTIC, GLIMMIX, and MIXED procedures)
- provides response-variable options as in the LOGISTIC procedure
- performs maximum likelihood estimation
- provides the logit, probit, log-log, and complementary log-log link functions
- provides cumulative link models for ordinal response data and generalized logit modeling for unordered multinomial data
- enables model building (variable selection) through the `SELECTION` statement
- provides a `WEIGHT` statement for weighted analysis
- provides a `FREQ` statement for grouped analysis
- provides a `CODE` statement to produce SAS code that can score a new data set
- provides an `OUTPUT` statement to produce a data table that contains predicted probabilities and other observationwise statistics
- uses ODS Graphics to create model selection plots as part of its output. For more information about ODS Graphics, see the section “ODS Graphics” on page 600.

Because the LOGSELECT procedure runs on CAS, it also does the following:

- enables you to run on a cluster of machines that distribute the data and the computations
- enables you to run in single-machine mode on CAS
- exploits all the available cores and concurrent threads. For information about how PROC LOGSELECT uses threads, see the section “Multithreading” on page 81 in Chapter 3, “Shared Concepts.”

**PROC LOGSELECT Compared with Other SAS Procedures**

The LOGSELECT procedure provides logistic regression modeling functionality that is comparable to that of the HPLOGISTIC and LOGISTIC procedures in SAS/STAT software.
PROC LOGSELECT Compared with the HPLOGISTIC Procedure

The functionality of the LOGSELECT procedure closely resembles that of the HPLOGISTIC procedure, which is a high-performance procedure. The LOGSELECT procedure is the next generation of the HPLOGISTIC procedure, and it was developed specifically for SAS Viya. Both procedures are designed to run on a cluster of machines that distribute the data and the computations. Both procedures perform computations in multiple threads.

Both the LOGSELECT and HPLOGISTIC procedures fit and perform model selection for logistic regression models. The models can contain main effects that consist of both continuous and classification variables and interaction effects of these variables. The HPLOGISTIC procedure provides GLM and reference parameterizations of classification variables; the LOGSELECT procedure provides the full set of parameterizations that are available in PROC LOGISTIC and other modeling procedures. The LOGSELECT procedure is additionally capable of creating complex constructed effects, including univariate spline and polynomial expansions.

With the LOGSELECT and HPL0GISTIC procedures, you request model selection by using the SELECTION statement. Both procedures offer the same methods of effect selection, but the LOGSELECT procedure also provides LASSO selection and produces selection plots by using ODS Graphics.

The LOGSELECT procedure provides more regression diagnostics than the HPLOGISTIC procedure. This release of the LOGSELECT procedure is more limited in postfitting functionality than the HPLOGISTIC procedure.

PROC LOGSELECT Compared with the LOGISTIC Procedure

The LOGSELECT procedure provides the full set of parameterizations that are available in PROC LOGISTIC. The LOGSELECT procedure uses the GLM parameterization for the CLASS variables by default. The LOGISTIC procedure uses the EFFECT parameterization for the CLASS variables by default. In either procedure, you can use the PARAM= option in the CLASS statement to change the parameterization.

The LOGISTIC procedure uses Fisher scoring by default. The LOGSELECT procedure uses a modification of the Newton-Raphson algorithm with a ridged Hessian by default. You can choose different optimization techniques, including first-order methods that do not require a crossproducts matrix or Hessian, by using the TECHNIQUE= option in the PROC LOGSELECT statement.

The LOGSELECT procedure provides a richer set of model selection options than the LOGISTIC procedure, including LASSO selection, information-criterion-based selection and stopping criteria, and validation-based criteria.

The LOGSELECT procedure uses ODS Graphics to display plots that help interpret the selection process; the LOGISTIC procedure displays many diagnostic and fit plots.

The LOGISTIC procedure offers a wide variety of postfitting analyses, such as contrasts, estimates, tests of model effects, least squares means, and odds ratios. This release of the LOGSELECT procedure is limited in postfitting functionality, because for large data sets the focus is primarily on model fitting and scoring.

The LOGSELECT procedure is specifically designed to operate in SAS Viya and performs computations in multiple threads. The LOGISTIC procedure executes in a single thread on a single machine.
Using CAS Sessions and CAS Engine Librefs

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:

```
proc options option=(CASHOST CASPORT);
run;
```

In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:

```
cas mysess;
  libname mycas cas sessref=mysess;
```

The CAS statement creates the CAS session named `mysess`, and the LIBNAME statement creates the `mycas` CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the CASHOST and CASPORT of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the `mysess` session, you can terminate it by using the TERMINATE option in the CAS statement as follows:

```
cas mysess terminate;
```

For more information about the CAS and LIBNAME statements, see the section “Introduction to Shared Concepts” on page 10 in Chapter 3, “Shared Concepts.”

Getting Started: LOGSELECT Procedure

Binary Logistic Regression

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

The following DATA step creates the data table `getStarted`, which consists of 100 observations on a dichotomous response variable (`y`), a character variable (`C`), and 10 continuous variables (`x1`–`x10`), in your CAS session:
data mycas.getStarted;
  input C$ y x1-x10;
datalines;
  D  0 10.2  6  1.6  38  15  2.4  20  0.8  8.5  3.9
  F  1 12.2  6  2.6  42  61  1.5  10  0.6  8.5  0.7
  D  1  7.7  1  2.1  38  61  1  90  0.6  7.5  5.2
  J  1 10.9  7  3.5  46  42  0.3  0  0.2  6  3.6
  E  0 17.3  6  3.8  26  47  0.9  10  0.4  1.5  4.7
  A  0 18.7  4  1.8  2  34  1.7  80  1  9.5  2.2
  B  0  7.2  1  0.3  48  61  1.1  10  0.8  3.5  4
  D  0  1  3  2.4  65  1.6  70  0.8  3.5  0.7
  H  1  2.4  4  0.7  38  22  0.2  20  0  3  4.2
  J  0 15.6  7  1.4  0  98  0.3  0  1  5  5.2
  J  0 11.1  3  2.4  42  55  2.2  60  0.6  4.5  0.7
  F  0  4  6  0.9  4  36  2.1  30  0.8  9  4.6
  A  0  6.2  2  1.8 14 79  1.1  70  0.2  0  5.1
  H  0  3.7  3  0.8 12 66  1.3  40  0.4  0.5  3.3
  A  1  9.2  3  2.3  48  51  2.3  50  0  6  5.4
  G  0 14  3  2  18 12  2.2  0  0  3  3.4
  E  1 19.5  6  3.7  26  81  0.1  30  0.6  5  4.8
  C  0 11  3  2.8  38  9  1.7  50  0.8  6.5  0.9
  I  0 15.3  7  2.2 20 98  2.7  100  0.4  7  0.8
  H  1  7.4  4  0.5  28  65  1.3  60  0.2  9.5  5.4
  F  0 11.4  2  1.4  42  12  2.4  10  0.4  1  4.5
  C  1 19.4  1  0.4  42  4  2.4  10  0  6.5  0.1
  G  0  5.9  4  2.6  12  57  0.8  50  0.4  2  5.8
  G  1 15.8  6  3.7  34  8  1.3  90  0.6  2.5  5.7
  I  0 10  3  1.9  16  80  3  90  0.4  9.5  1.9
  E  0 15.7  1  2.7  32  25  1.7  20  0.2  8.5  6
  G  0 11  5  2.9  48  53  0.1  50  1  3.5  1.2
  J  1 16.8  0  0.9 14 86  1.4  40  0.8  9  5
  D  1 11  4  3.2  48  63  2.8  90  0.6  0  2.2
  J  1  4.8  7  3.6  24  1  2.2  20  1  8.5  0.5
  J  1 10.4  5  2  42  56  1  20  0  3.5  4.2
  G  0 12.7  7  3.6  8  56  2.1  70  1  4.5  1.5
  G  0  6.8  1  3.2  30  27  0.6  0  0.8  2  5.6
  E  0  8.8  0  3.2  2  67  0.7  10  0.4  1  5
  I  1  0.2  0  2.9 10 41  2.3  60  0.2  9  0.3
  J  1  4.6  7  3.9  50  61  2.1  50  0.4  3  4.9
  J  1  2.3  2  3.2  36  98  0.1  40  0.6  4.5  4.3
  I  0 10.8  3  2.7  28  58  0.8  80  0.8  3  6
  B  0  9.3  2  3.3  44  44  0.3  50  0.8  5.5  0.4
  F  0  9.2  6  0.6  4  64  0.1  0  0.6  4.5  3.9
  D  0  7.4  0  2.9 14  0  0.2  30  0.8  7.5  4.5
  G  0 18.3  3  3.1  8  60  0.3  60  0.2  7  1.9
  F  0  5.3  4  0.2  48  63  2.3  80  0.2  8  5.2
  C  0  2.6  5  2.2  24  4  1.3  20  0  2  1.4
  F  0 13.8  4  3.6  4  7  1.1  10  0.4  3.5  1.9
  B  1 12.4  6  1.7  30  44  1.1  60  0.2  6  1.5
  I  0  1.3  1  1.3  8  53  1.1  70  0.6  7  0.8
  F  0 18.2  7  1.7  26  92  2.2  30  1  8.5  4.8
  J  0  5.2  2  2.2  18  12  1.4  90  0.8  4  4.9
  G  1  9.4  2  0.8  22  86  0.4  30  0.4  1  5.9
| J  | 1  | 10.4 | 2  | 1.7 | 26  | 31  | 2.4 | 10  | 0.2 | 7   | 1.6 |
| J  | 0  | 13   | 1  | 1.8 | 14  | 11  | 2.3 | 50  | 0.6 | 5.5 | 2.6 |
| A  | 0  | 17.9 | 4  | 3.1 | 46  | 58  | 2.6 | 90  | 0.6 | 1.5 | 3.2 |
| D  | 1  | 19.4 | 6  | 3   | 20  | 50  | 2.8 | 100 | 0.2 | 9   | 1.2 |
| I  | 0  | 19.6 | 3  | 3.6 | 22  | 19  | 1.2 | 0   | 0.6 | 5   | 4.1 |
| I  | 1  | 6.2  | 2  | 1.5 | 30  | 30  | 2.2 | 20  | 0.4 | 8.5 | 5.3 |
| G  | 0  | 13.8 | 1  | 2.7 | 0   | 52  | 2.4 | 20  | 0.8 | 6   | 2  |
| B  | 0  | 14.3 | 4  | 2.9 | 30  | 11  | 0.6 | 90  | 0.6 | 0.5 | 4.9 |
| E  | 0  | 15.6 | 0  | 0.4 | 38  | 79  | 0.4 | 80  | 0.4 | 1   | 3.3 |
| D  | 0  | 14   | 2  | 1   | 22  | 61  | 3.9 | 0   | 0.6 | 2   | 0.1 |
| C  | 1  | 9.4  | 5  | 0.4 | 12  | 53  | 1.7 | 40  | 0   | 3   | 1.1 |
| H  | 0  | 13.2 | 1  | 1.6 | 40  | 15  | 0.7 | 40  | 0.2 | 9   | 5.5 |
| A  | 0  | 13.5 | 5  | 2.4 | 18  | 89  | 1.6 | 20  | 0.4 | 9.5 | 4.7 |
| E  | 0  | 2.6  | 4  | 2.3 | 38  | 6   | 0.8 | 20  | 0.4 | 5   | 5.3 |
| E  | 0  | 12.4 | 3  | 1.3 | 26  | 8   | 2.8 | 10  | 0.8 | 6   | 5.8 |
| D  | 0  | 7.6  | 2  | 0.9 | 44  | 89  | 1.3 | 50  | 0.8 | 6   | 0.4 |
| I  | 0  | 12.7 | 1  | 2.3 | 42  | 6   | 2.4 | 10  | 0.4 | 1   | 3  |
| C  | 1  | 10.7 | 4  | 3.2 | 28  | 23  | 2.2 | 90  | 0.8 | 5.5 | 2.8 |
| H  | 0  | 10.1 | 2  | 2.3 | 10  | 62  | 0.9 | 50  | 0.4 | 2.5 | 3.7 |
| C  | 1  | 16.6 | 1  | 0.5 | 12  | 88  | 0.1 | 20  | 0.6 | 5.5 | 1.8 |
| I  | 1  | 0.2  | 3  | 2.2 | 8   | 71  | 1.7 | 80  | 0.4 | 0.5 | 5.5 |
| C  | 0  | 10.8 | 4  | 3.5 | 30  | 70  | 2.3 | 60  | 0.4 | 4.5 | 5.9 |
| F  | 0  | 7.1  | 4  | 3   | 14  | 63  | 2.4 | 70  | 0   | 7   | 3.1 |
| D  | 0  | 16.5 | 1  | 3.3 | 30  | 80  | 1.6 | 40  | 0   | 3.5 | 2.7 |
| H  | 0  | 17.1 | 7  | 2.1 | 30  | 45  | 1.5 | 60  | 0.6 | 0.5 | 2.8 |
| D  | 0  | 4.3  | 1  | 1.5 | 24  | 44  | 0   | 70  | 0   | 5   | 0.5 |
| H  | 0  | 15   | 2  | 0.2 | 14  | 87  | 1.8 | 50  | 0   | 4.5 | 4.7 |
| G  | 0  | 19.7 | 3  | 1.9 | 36  | 99  | 1.5 | 10  | 0.6 | 3   | 1.7 |
| H  | 1  | 2.8  | 6  | 0.6 | 34  | 21  | 2.6 | 60  | 1   | 9   | 4.7 |
| G  | 0  | 16.6 | 3  | 3.3 | 46  | 1   | 1.4 | 70  | 0.6 | 1.5 | 5.3 |
| E  | 0  | 11.7 | 5  | 2.7 | 48  | 4   | 0.9 | 60  | 0.8 | 4.5 | 1.6 |
| F  | 0  | 15.6 | 3  | 0.2 | 4   | 79  | 0.5 | 0   | 0.8 | 1.5 | 2.9 |
| C  | 1  | 5.3  | 6  | 1.4 | 8   | 64  | 2.8 | 80  | 0.4 | 9   | 4.2 |
| B  | 1  | 8.1  | 7  | 1.7 | 40  | 36  | 1.4 | 60  | 0.6 | 6   | 3.9 |
| I  | 0  | 14.8 | 2  | 3.2 | 8   | 37  | 0.4 | 10  | 0   | 4.5 | 3  |
| D  | 0  | 7.4  | 4  | 3   | 12  | 3   | 0.6 | 60  | 0.6 | 7   | 0.7 |
| D  | 0  | 4.8  | 3  | 2.3 | 44  | 41  | 1.9 | 60  | 0.2 | 3   | 3.1 |
| A  | 0  | 4.5  | 0  | 0.2 | 4   | 48  | 1.7 | 80  | 0.8 | 9   | 4.2 |
| D  | 0  | 6.9  | 6  | 3.3 | 14  | 92  | 0.5 | 40  | 0.4 | 7.5 | 5  |
| B  | 0  | 4.7  | 4  | 0.9 | 14  | 99  | 2.4 | 80  | 1   | 0.5 | 0.7 |
| I  | 1  | 7.5  | 4  | 2.1 | 20  | 79  | 0.4 | 40  | 0.4 | 2.5 | 0.7 |
| C  | 0  | 6.1  | 0  | 1.4 | 38  | 18  | 2.3 | 60  | 0.8 | 4.5 | 0.7 |
| C  | 0  | 18.3 | 1  | 1   | 26  | 98  | 2.7 | 20  | 1   | 8.5 | 0.5 |
| F  | 0  | 16.4 | 7  | 1.2 | 32  | 94  | 2.9 | 40  | 0.4 | 5.5 | 2.1 |
| I  | 0  | 9.4  | 2  | 2.3 | 32  | 42  | 0.2 | 70  | 0.4 | 8.5 | 0.3 |
| F  | 1  | 17.9 | 4  | 1.3 | 32  | 42  | 2   | 40  | 0.2 | 1   | 5.4 |
| H  | 0  | 14.9 | 3  | 1.6 | 36  | 74  | 2.6 | 60  | 0.2 | 1   | 2.3 |
| C  | 0  | 12.7 | 0  | 2.6 | 0   | 88  | 1.1 | 80  | 0.8 | 0.5 | 2.1 |
| F  | 0  | 5.4  | 4  | 1.5 | 2   | 1   | 1.8 | 70  | 0.4 | 5.5 | 3.6 |
| J  | 1  | 12.1 | 4  | 1.8 | 20  | 59  | 1.3 | 60  | 0.4 | 3   | 3.8 |

These statements assume that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.
The following statements fit a logistic model to these data by using a classification effect for variable \( C \) and 10 regressor effects for \( x1-x10 \). The \textit{ITHIST} option displays a table that summarizes the steps of the optimization.

```plaintext
proc logselect data=mycas.getStarted ithist;
  class C;
  model y = C x1-x10;
run;
```

The output from this analysis is presented in Figure 11.1 through Figure 11.10.

**Figure 11.1** displays the “Model Information” table. The \textit{LOGSELECT} procedure uses a Newton-Raphson algorithm to model a binary distribution with a logit link function for the variable \( y \). The \textit{CLASS} variable \( C \) is parameterized using the GLM parameterization, which is the default.

**Figure 11.2** displays the “Number of Observations” table. All 100 observations in the data table are used in the analysis.

**Figure 11.3** is produced by default. It shows the breakdown of the response variable levels by frequency. By default for binary data, the \textit{LOGSELECT} procedure models the probability of the event with the lower OrderedValue value in the “Response Profile” table, as indicated by the note that follows the table. In this example, the values that are represented by \( y = 0 \) are modeled as the “successes” in the Bernoulli experiments.

You can use the response-variable options in the \textit{MODEL} statement to choose which value of the response variable to model.
The CLASS variable C has 10 unique formatted levels that are displayed in the “Class Level Information” table in Figure 11.4.

**Figure 11.4** Class Level Information

<table>
<thead>
<tr>
<th>Class Level Information</th>
<th>Class Levels Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>10 A B C D E F G H I J</td>
</tr>
</tbody>
</table>

The “Iteration History” table is shown in Figure 11.5. The Newton-Raphson algorithm with ridging converged after four iterations, not counting the initial setup iteration.

**Figure 11.5** Iteration History

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Evaluations</th>
<th>Objective Function</th>
<th>Change</th>
<th>Maximum Gradient</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>4</td>
<td>0.4493546916</td>
<td>0.410972</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0.4436453992</td>
<td>0.00570929</td>
<td>0.081339</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0.4435038109</td>
<td>0.00014159</td>
<td>0.003302</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>0.4435035933</td>
<td>0.00000022</td>
<td>5.623E-6</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>0.4435035933</td>
<td>0.00000000</td>
<td>1.59E-11</td>
</tr>
</tbody>
</table>

Figure 11.6 displays the final convergence status of the Newton-Raphson algorithm. The GCONV= relative convergence criterion is satisfied.

**Figure 11.6** Convergence Status

Convergence criterion (GCONV=1E-8) satisfied.

Figure 11.7 displays the “Dimensions” table for this model. This table summarizes some important sizes of various model components. For example, it shows that the design matrix \( X \) has 21 columns, which correspond to 1 column for the intercept, 10 columns for the effect associated with the classification variable C, and 1 column each for the continuous variables \( x_1 \)–\( x_{10} \). However, the rank of the crossproducts matrix is only 20. Because the classification variable C uses GLM parameterization and because the model contains an intercept, there is one singularity in the crossproducts matrix of the model. Consequently, only 20 parameters enter the optimization.

**Figure 11.7** Dimensions in Binomial Logistic Regression

<table>
<thead>
<tr>
<th>Dimensions</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Columns in Design</td>
<td>21</td>
</tr>
<tr>
<td>Number of Effects</td>
<td>12</td>
</tr>
<tr>
<td>Max Effect Columns</td>
<td>10</td>
</tr>
<tr>
<td>Rank of Design</td>
<td>20</td>
</tr>
<tr>
<td>Parameters in Optimization</td>
<td>20</td>
</tr>
</tbody>
</table>
Figure 11.8 shows the global test for the null hypothesis that all model effects jointly do not affect the probability of success of the binary response. The test is significant \((p = 0.0135)\). One or more of the model effects thus significantly affect the probability of observing an event.

![Null Test Table](image)

The “Fit Statistics” table is shown in Figure 11.9. The \(-2\) log likelihood at the converged estimates is 88.7007. You can use this value to compare the model to nested model alternatives by means of a likelihood ratio test. To compare models that are not nested, you can use information criteria such as AIC (Akaike’s information criterion), AICC (Akaike’s bias-corrected information criterion), and SBC (Schwarz Bayesian information criterion). These criteria penalize the \(-2\) log likelihood for the number of parameters. Because of the large number of parameters relative to the number of observations, the discrepancy between the \(-2\) log likelihood and, for example, AIC is substantial in this case.

![Fit Statistics Table](image)

However, the “Parameter Estimates” table in Figure 11.10 shows that many parameters have fairly large \(p\)-values, indicating that one or more of the model effects might not be necessary.
Finally, the procedure displays the table in Figure 11.11, which shows the amount of time (in seconds) that PROC LOGSELECT required to perform different tasks in the analysis.

**Figure 11.11** Procedure Timing

<table>
<thead>
<tr>
<th>Task</th>
<th>Seconds</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setup and Parsing</td>
<td>0.02</td>
<td>16.49%</td>
</tr>
<tr>
<td>Levelization</td>
<td>0.03</td>
<td>23.29%</td>
</tr>
<tr>
<td>Model Initialization</td>
<td>0.01</td>
<td>6.98%</td>
</tr>
<tr>
<td>SSCP Computation</td>
<td>0.00</td>
<td>2.71%</td>
</tr>
<tr>
<td>Model Fitting</td>
<td>0.07</td>
<td>50.39%</td>
</tr>
<tr>
<td>Cleanup</td>
<td>0.00</td>
<td>0.00%</td>
</tr>
<tr>
<td>Total</td>
<td>0.15</td>
<td>100.00%</td>
</tr>
</tbody>
</table>
Syntax: LOGSELECT Procedure

The following statements are available in the LOGSELECT procedure:

```plaintext
PROC LOGSELECT <options> ;
  BY variables ;
  CLASS variable <(options)> . . . <variable <(options)>> < /global-options> ;
  CODE <options> ;
  DISPLAY <table-list> < /options> ;
  DISPLAYOUT table-spec-list < /options> ;
  EFFECT name=effect-type(variables < /options>) ;
  FREQ variable ;
  MODEL response< (response-options) > = < effects > < /model-options> ;
  MODEL events/trials < (response-options) > = < effects > < /model-options> ;
  OUTPUT OUT= CAS-libref.data-table <options> <keyword < = name>> . . . <keyword < = name>> ;
  PARTITION partition-options ;
  SELECTION < METHOD =method< (method-options)> > <options> ;
  WEIGHT variable ;
```

The PROC LOGSELECT statement and at least one MODEL statement are required. The CLASS statement can appear multiple times. If a CLASS statement is specified, it must precede the MODEL statements.

PROC LOGSELECT Statement

```plaintext
PROC LOGSELECT <options> ;
```

The PROC LOGSELECT statement invokes the procedure. Table 11.1 summarizes the available options in the PROC LOGSELECT statement by function.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALPHA=</td>
<td>Specifies a global significance level</td>
</tr>
<tr>
<td>BINEPS=</td>
<td>Specifies the precision for computing association and classification statistics</td>
</tr>
<tr>
<td>DATA=</td>
<td>Specifies the CAS input data table</td>
</tr>
<tr>
<td>PAGEOBS=</td>
<td>Specifies the maximum number of observations to be computed in each batch</td>
</tr>
</tbody>
</table>

**Output Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASSOCIATION</td>
<td>Displays association statistics</td>
</tr>
<tr>
<td>CORRB</td>
<td>Displays the “Parameter Estimates Correlation Matrix” table</td>
</tr>
<tr>
<td>COVB</td>
<td>Displays the “Parameter Estimates Covariance Matrix” table</td>
</tr>
<tr>
<td>CTABLE</td>
<td>Requests classification (ROC) statistics</td>
</tr>
<tr>
<td>ITHIST</td>
<td>Displays the “Iteration History” table</td>
</tr>
<tr>
<td>NOCHECK</td>
<td>Disables checking for infinite parameters</td>
</tr>
<tr>
<td>NOCLPRINT</td>
<td>Limits or suppresses the display of class levels</td>
</tr>
<tr>
<td>NOSTDERR</td>
<td>Suppresses computation of the covariance matrix and standard errors</td>
</tr>
<tr>
<td>PARTFIT</td>
<td>Displays the fit statistics that are produced when your data are partitioned</td>
</tr>
</tbody>
</table>
The optimization options are fully described in the section “Optimization Options” on page 44 in Chapter 3, “Shared Concepts.” The following list describes the other options available in the PROC LOGSELECT statement:

**ALPHA=** *number*

specifies a global significance level for the construction of confidence intervals. The confidence level is \(1 - \text{number}\). The value of *number* must be between 0 and 1. You can override the global specification by specifying the ALPHA= option in the OUTPUT statement. By default, ALPHA=0.05.

**ASSOCIATION**

displays measures of association between predicted probabilities and observed responses for binary, binomial, and ordinal response models. These measures assess the predictive ability of the model. The displayed statistics are the concordance index \(c\) (the area under the ROC curve, AUC), Somers’ \(D\) statistic (Gini’s coefficient), the Goodman-Kruskal gamma statistic, and Kendall’s tau-\(a\) statistic. These statistics are based on the number of pairs of observations that have different response values, the number of concordant pairs, and the number of discordant pairs. For more information, see the section “Association Statistics” on page 588.

**BINEPS=** *number*

specifies the precision to use for the association and classification computations. The predicted probabilities or scores are rounded to the nearest multiple of *number*, which effectively bins and sorts the observations. You can specify a *number* between 0 and 1; if *number* does not evenly divide the
unit interval, then it is reduced to a valid value. By default, BINEPS=1e–5. For more information, see the section “Classification Table and ROC Curves” on page 586.

**CORRB**
creates the “Parameter Estimates Correlation Matrix” table. The correlation matrix is computed by normalizing the covariance matrix \( \Sigma \). That is, if \( \sigma_{ij} \) is an element of \( \Sigma \), then the corresponding element of the correlation matrix is \( \rho_{ij} = \frac{\sigma_{ij}}{\sqrt{\sigma_{ii}\sigma_{jj}}} \).

**COVB**
creates the “Parameter Estimates Covariance Matrix” table. The covariance matrix is computed as the inverse of the negative of the matrix of second derivatives of the log-likelihood function with respect to the model parameters (the Hessian matrix).

**CTABLE< (options) >**
displays a table for binary or binomial response models that, for a set of probabilities or cutpoints, contains the frequencies of observations that are correctly and incorrectly classified as events and nonevents, and optionally the sensitivity, the 1–specificity, the positive and negative predictive values, the correct classification rate, the error rate, and the lift.

Because the number of cutpoints that are generated from your data can be very large, you can write the classification table to an output data table, or you can specify your own list of cutpoints.

If you specify the PRIOR= option, then the reported predicted probabilities are unchanged, but the statistics that are specified by the keywords PPV, NPV, PC, ACCURACY, MISCLASS, and LIFT are modified.

If you specify a PARTITION statement, then a table is created for each role. If you do not specify the PRIOR= option, then the training proportions are used as the prevalences for computing the statistics that are specified by the keywords PPV, NPV, PC, ACCURACY, MISCLASS, and LIFT for the validation and test data.

You can specify the following options. For more information, see the section “Classification Table and ROC Curves” on page 586.

**ALL**
requests all the statistics that are listed in Table 11.2.

**CUTPT=number-list**
specifies cutpoints to include in the classification table.

**NOCOUNTS**
suppresses the classification table columns that contain the number of true positives, the number of true negatives, the number of false positives, and the number of false negatives.

**OUT=CAS-libref.data-table**
suppresses the display of the classification table and names the output data table in which to store the classification table. *CAS-libref.data-table* is a two-level name, where *CAS-libref* refers to the caslib and session identifier, and *data-table* specifies the name of the output data table. For more information about this two-level name, see the DATA= option and the section “Using CAS Sessions and CAS Engine Librefs” on page 551.
keyword <= name >
specifies statistics to include in the classification table and optionally names the statistics. If you
do not provide a name, the LOGSELECT procedure uses the keyword as the name. Table 11.2
lists the available keywords.

Table 11.2    CTABLE Option Keywords

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCURACY</td>
<td>Requests and names the accuracy</td>
</tr>
<tr>
<td>FNF</td>
<td>Requests and names the false negative fraction</td>
</tr>
<tr>
<td>FPF</td>
<td>Requests and names the false positive fraction.</td>
</tr>
<tr>
<td></td>
<td>Also called 1–specificity.</td>
</tr>
<tr>
<td>LIFT</td>
<td>Requests and names the lift</td>
</tr>
<tr>
<td>MISCLASS</td>
<td>Requests and names the misclassification rate.</td>
</tr>
<tr>
<td></td>
<td>Also called the error rate.</td>
</tr>
<tr>
<td>NPV</td>
<td>Requests and names the negative predictive value</td>
</tr>
<tr>
<td>PC</td>
<td>Requests and names the proportion classified correctly</td>
</tr>
<tr>
<td>PPV</td>
<td>Requests and names the positive predictive value.</td>
</tr>
<tr>
<td></td>
<td>Also called the precision.</td>
</tr>
<tr>
<td>TNF</td>
<td>Requests and names the true negative fraction.</td>
</tr>
<tr>
<td></td>
<td>Also called the specificity.</td>
</tr>
<tr>
<td>TPF</td>
<td>Requests and names the true positive fraction.</td>
</tr>
<tr>
<td></td>
<td>Also called the sensitivity or recall.</td>
</tr>
</tbody>
</table>

DATA=CAS-libref.data-table
names the input data table for PROC LOGSELECT to use. The default is the most recently created
data table. CAS-libref.data-table is a two-level name, where

CAS-libref refers to a collection of information that is defined in the LIBNAME statement and
includes the caslib, which includes a path to the data, and a session identifier, which
defaults to the active session but which can be explicitly defined in the LIBNAME
statement. For more information about CAS-libref, see the section “Using CAS
Sessions and CAS Engine Librefs” on page 551.

data-table specifies the name of the input data table.

ITHIST
generates the “Iteration History” table.

LASSORHO=r
specifies the base regularization parameter for the LASSO model selection method. The regularization
parameter for step \( i \) is \( r^i \). By default, LASSORHO=0.8.

LASSOSTEPS=n
specifies the maximum number of steps for LASSO model selection. By default, LASSOSTEPS=20.
LASSOTOL=r
specifies the convergence tolerance for the optimization algorithm that solves for the LASSO parameter estimates at each step of LASSO model selection. By default, LASSOTOL=1E–6.

NOCHECK
disables the checking process that determines whether maximum likelihood estimates of the regression parameters exist. For more information, see the section “Existence of Maximum Likelihood Estimates” on page 583.

NOCLPRINT<=number>
suppresses the display of the “Class Level Information” table if you do not specify number. If you specify number, the values of the classification variables are displayed for only those variables whose number of levels is less than number. Specifying number helps to reduce the size of the “Class Level Information” table if some classification variables have a large number of levels.

NOSTDERR
suppresses computation of the covariance matrix and the standard errors of the regression coefficients. When the model contains many variables (thousands), the inversion of the Hessian matrix to derive the covariance matrix and the standard errors of the regression coefficients can be time-consuming. The CORRB, COVB, and TYPE3 options are not available when the NOSTDERR option is specified. This option also disables the quasi-complete separation check; for more information, see the section “Existence of Maximum Likelihood Estimates” on page 583.

PAGEOBS=number | AUTO
MAXOPTBATCH=number | AUTO
specifies the maximum number of observations to be included in a batch. During the optimization, the LOGSELECT procedure reads at most number observations from the data table into memory, performs the appropriate log-likelihood, gradient, and Hessian computations on that batch of observations, then discards those observations and reads in the next batch of data for processing. Generally, a smaller number decreases memory usage but might lead to longer computation times, whereas a larger number might lead to shorter computation times but increases memory usage. The default PAGEOBS=AUTO option determines whether the entire data table can be conveniently held in memory; if it cannot, then number is set to 256. For more information, see the section “Memory Usage” on page 593.

PARTFIT
displays fit statistics in the “Fit Statistics” table that are usually produced when your data are partitioned. This option is not required when you specify a PARTITION statement.

The additional statistics include the R-square, the adjusted R-square, McFadden’s R-square, the average square error (Brier score), the misclassification rate, the null-model log likelihood, and the difference in means.

STB
displays the standardized estimates of the parameters in the “Parameter Estimates” table. The standardized estimate of $\hat{\beta}_i$ is given by $\hat{\beta}_i/(s_i/s_i)$, where $s_i$ is the total sample standard deviation for the $i$th explanatory variable and

$$s = \begin{cases} \pi/\sqrt{3} & \text{LOGIT and GLOGIT links} \\ 1 & \text{PROBIT link} \\ \pi/\sqrt{6} & \text{CLOGLOG and LOGLOG links} \end{cases}$$
The sample standard deviations for parameters that are associated with CLASS variables are computed using their codings. The standardized estimates are not computed for the intercept parameters.

BY Statement

BY variables;

You can specify a BY statement in PROC LOGSELECT to obtain separate analyses of observations in groups that are defined by the values of the BY variables. If you specify more than one BY statement, only the last one specified is used. For more information, see the discussion of BY-group processing in SAS Language Reference: Concepts.

CLASS Statement

CLASS variable<(options)>…<variable<(options)>></global-options>;

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. You can list the response variable for binary models in the CLASS statement, but this is not required. Table 11.3 summarizes the values that you can use for either an option or a global-option. The options are fully documented in the section “CLASS Statement” on page 12 in Chapter 3, “Shared Concepts.”

Table 11.3  CLASS Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESCENDING</td>
<td>Reverses the sort order</td>
</tr>
<tr>
<td>MISSING</td>
<td>Treats missing values as valid levels</td>
</tr>
<tr>
<td>ORDER=</td>
<td>Specifies the sort order for the levels</td>
</tr>
<tr>
<td>PARAM=</td>
<td>Specifies the parameterization of the variable</td>
</tr>
<tr>
<td>REF=</td>
<td>Specifies the reference level of the variable</td>
</tr>
<tr>
<td>SPLIT</td>
<td>Allows design columns for a variable to enter or leave the model independently</td>
</tr>
</tbody>
</table>

CODE Statement

CODE <options>;

The CODE statement writes SAS DATA step code for computing predicted values of the fitted model to a file, to a catalog entry, or to a CAS table. To score new data, you can then include the file or the catalog entry in a DATA step, or you can specify the CAS table in the runCodeTable action in the dataStep action set (for more information, see SAS Viya: System Programming Guide).

Table 11.4 summarizes the options available in the CODE statement.
Table 11.4  CODE Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMMENT</td>
<td>Adds comments to the generated code</td>
</tr>
<tr>
<td>FILE=</td>
<td>Names the file in which to save the generated code</td>
</tr>
<tr>
<td>FORMATWIDTH=</td>
<td>Specifies the numeric format width for the regression coefficients</td>
</tr>
<tr>
<td>INDENTSIZEl=</td>
<td>Specifies the number of spaces to indent the generated code</td>
</tr>
<tr>
<td>IPROB</td>
<td>Computes individual predicted probabilities for ordinal response models</td>
</tr>
<tr>
<td>LABELID=</td>
<td>Specifies a number used to construct names and labels</td>
</tr>
<tr>
<td>LINESIZE=</td>
<td>Specifies the line size for the generated code</td>
</tr>
<tr>
<td>NOTRIM</td>
<td>Compares formatted values, including blank padding</td>
</tr>
<tr>
<td>OUT=</td>
<td>Names an output CAS table in which to save the generated code</td>
</tr>
<tr>
<td>PCATALL</td>
<td>Generates probabilities for all levels of categorical response variables</td>
</tr>
</tbody>
</table>

For more information about the syntax of the CODE statement, see the section “CODE Statement” on page 16 in Chapter 3, “Shared Concepts.”

DISPLAY Statement

```
DISPLAY <table-list> < / options> ;
```

The DISPLAY statement enables you to specify a list of display tables to display or exclude. This statement is similar to the ODS SELECT, ODS EXCLUDE, and ODS TRACE statements. However, the DISPLAY statement can improve performance when a large number of tables could be generated (such as in BY-group processing). The procedure processes the DISPLAY statement on a CAS server and thus sends only a subset of ODS tables to the SAS client. Because ODS statements are processed on a SAS client, first all the generated display tables are sent to the client, and then the client creates a subset.

If you use both DISPLAY and ODS statements together, the DISPLAY statement takes precedence over the ODS statements. Note that the ODS EXCLUDE statement processes tables that are sent to the client after they have been filtered by the DISPLAY statement. In some cases, it might appear that the ODS EXCLUDE statement is taking precedence because it can further filter the tables. For more information about ODS, see SAS Output Delivery System: Procedures Guide.

You can specify the `table-list` as a list of table names, paths, partial pathnames, and regular expressions.

The table names that you can specify are listed in the section “ODS Table Names” on page 598. A path is a table name that is prefixed with dot-separated grouping information. For example, a SelectionSummary table that a procedure produces during a selection routine might have the path `Bygroup1.Summary.SelectionSummary`. A partial pathname does not include all groups; for example, `SelectionSummary` and `Summary.SelectionSummary` are partial pathnames for `Bygroup1.Summary.SelectionSummary`.

When you specify a table name or partial pathname, all display tables whose paths end in the specified name are selected for display or exclusion. For example, both `SelectionSummary` and `Summary.SelectionSummary` select `Bygroup1.Summary.SelectionSummary`.

A regular expression is enclosed in forward slashes (/). For example, specifying “/tions/” selects all pathnames that contain the substring “tions”; in particular, the `Bygroup1.Summary.SelectionSummary` table is selected.
Specifying “!/tions!” selects all pathnames that do not contain the substring “tions”; in particular, the `Bygroup1.Summary.SelectionSummary` table is not selected.

You can specify the following options after a slash (/):

**CASESENSITIVE**
performs a case-sensitive comparison of table names in the `table-list` to display table names when tables are subsetted for display. To preserve case, you must enclose table names in the `table-list` in quotation marks.

**EXCLUDE**
displays all display tables except those that you specify in the `table-list`.

**EXCLUDEALL**
suppresses display of all tables. This option takes precedence over the other options.

**TRACE**
displays the display table names, labels, and paths.

---

**DISPLAYOUT Statement**

```plaintext
DISPLAYOUT table-spec-list </options> ;
```

The DISPLAYOUT statement enables you to create CAS output tables from your displayed output. This statement is similar to the ODS OUTPUT statement. For more information about ODS, see *SAS Output Delivery System: Procedures Guide*.

The `table-spec-list` specifies a list of CAS output tables to create. Each entry in the list has either a `key=value` format or a `key` format:

- `key=value` specifies `key` as the ODS table name, path, or partial pathname, and specifies `value` as the CAS output table name.
- `key` specifies `key` as the ODS table name and also as the CAS output table name.

The ODS table names that you can specify are listed in the section “ODS Table Names” on page 598. You cannot specify the ODS table named `OutputCasTables` in the `table-spec-list`.

Table names and partial pathnames are discussed under the DISPLAY statement. The DISPLAYOUT statement does not support regular expressions.

You can specify the following options after a slash (/):

**INCLUDEALL**
creates output CAS tables for all display tables. The name of the created output CAS table is the same as the corresponding display table name. If you specify this option, the `table-spec-list` specification is ignored.

**NOREPLACE**
does not replace any existing CAS output table of the same name.
REPEATED
replicates all CAS output tables on all nodes.

---

**Chapter 11: The LOGSELECT Procedure**

**EFFECT Statement**

EFFECT name=effect-type (variables < / options>);

The EFFECT statement enables you to construct special collections of columns for design matrices. These collections are referred to as *constructed effects* to distinguish them from the usual model effects that are formed from continuous or classification variables, as discussed in the section “GLM Parameterization of Classification Variables and Effects” on page 54 in Chapter 3, “Shared Concepts.”

You can specify the following *effect-types*:

- **COLLECTION** specifies a collection effect that defines one or more variables as a single effect that has multiple degrees of freedom. The variables in a collection are considered as a unit for purposes of estimation and inference.

- **MULTIMEMBER | MM** specifies a multimember classification effect whose levels are determined by one or more variables that appear in a CLASS statement.

- **POLYNOMIAL | POLY** specifies a multivariate polynomial effect in the specified numeric variables.

- **SPLINE** specifies a regression spline effect whose columns are univariate spline expansions of one or more variables. A spline expansion replaces the original variable with an expanded or larger set of new variables.

Table 11.5 summarizes the *options* available in the EFFECT statement.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Collection Effects Options</strong></td>
<td></td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays the constituents of the collection effect</td>
</tr>
<tr>
<td><strong>Multimember Effects Options</strong></td>
<td></td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays the levels of the multimember effect</td>
</tr>
<tr>
<td>NOEFFECT</td>
<td>Specifies that observations whose levels are all missing for the multimember variables should have 0 values in the corresponding design matrix columns</td>
</tr>
<tr>
<td>STDIZE</td>
<td>Standardizes the design matrix entries so that each observation has a sum of 1</td>
</tr>
<tr>
<td>WEIGHT=</td>
<td>Specifies the weight variable for the contributions of each classification effect</td>
</tr>
<tr>
<td><strong>Polynomial Effects Options</strong></td>
<td></td>
</tr>
<tr>
<td>DEGREE=</td>
<td>Specifies the degree of the polynomial</td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays details of the specified polynomial</td>
</tr>
<tr>
<td>MDEGREE=</td>
<td>Specifies the maximum degree of any variable in a term of the polynomial</td>
</tr>
</tbody>
</table>
Table 11.5  continued

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOSEPARATE</td>
<td>Treats the polynomial as a single effect with multiple degrees of freedom</td>
</tr>
<tr>
<td>STANDARDIZE=</td>
<td>Specifies centering and scaling suboptions for the variables that define the polynomial</td>
</tr>
</tbody>
</table>

**Spline Effects Options**

- BASIS= Specifies the type of basis (B-spline basis or truncated power function basis) for the spline effect
- DATABOUNDARY Uses the extremes of the data as boundary knots for a B-spline basis
- DEGREE= Specifies the degree of the spline effect
- DETAILS Displays the knots and locations for each spline basis function
- KNOTMAX= Requests equally spaced right-side boundary knots starting at the variables’ maximum and ending at the KNOTMAX= value
- KNOTMETHOD= Specifies how to construct the knots for the spline effect
- KNOTMIN= Requests equally spaced left-side boundary knots starting at the KNOTMIN= value and ending at the variables’ minimum value
- NATURALCUBIC Specifies a natural cubic spline basis for the spline effect
- SEPARATE Treats the spline basis for each variable as a separate effect when multiple variables are specified
- SPLIT Treats each design matrix column as a separate effect for selection methods

For more information about the syntax of these effect-types and how columns of constructed effects are computed, see the section “EFFECT Statement” on page 21 in Chapter 3, “Shared Concepts.”

---

**FREQ Statement**

```
FREQ variable;
```

The `variable` in the FREQ statement identifies a numeric variable in the data set that contains the frequency of occurrence of each observation. PROC LOGSELECT treats each observation as if it appears \( f \) times, where \( f \) is the value of the FREQ `variable` for the observation. If \( f \) is not an integer, it is truncated to an integer. If \( f \) is less than 1 or missing, the observation is not used in the analysis. When the FREQ statement is not specified, each observation is assigned a frequency of 1.

---

**MODEL Statement**

```
MODEL response < (response-options) >= < effects > < / model-options > ;
MODEL events / trials < (response-options) >= < effects > < / model-options > ;
```

The MODEL statement defines the statistical model in terms of a `response` variable (the target) or an `events/trials` specification, model `effects` that are constructed from variables in the input data table, and
model-options. An intercept is included in the model by default. You can remove the intercept by specifying the NOINT option.

You can specify a single response variable that contains your response values. When you have binomial data, you can specify the events/trials form of the response, where one variable contains the number of positive responses (or events) and another variable contains the number of trials. Note that the values of both events and (trials – events) must be nonnegative and the value of trials must be positive.

For information about constructing the model effects, see the section “Specification and Parameterization of Model Effects” on page 51 in Chapter 3, “Shared Concepts.”

There are two sets of options in the MODEL statement. The response-options determine how the LOGSELECT procedure models probabilities for binary and multinomial data. The model-options control other aspects of model formation and inference. Table 11.6 summarizes these options.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESCENDING</td>
<td>Reverses the response categories</td>
</tr>
<tr>
<td>EVENT=</td>
<td>Specifies the event category</td>
</tr>
<tr>
<td>ORDER=</td>
<td>Specifies the sort order</td>
</tr>
<tr>
<td>REF=</td>
<td>Specifies the reference category</td>
</tr>
<tr>
<td>CENTER</td>
<td>Centers and scales continuous main effects</td>
</tr>
<tr>
<td>CENTERLASSO</td>
<td>Centers and scales all effects for model selection by the LASSO method</td>
</tr>
<tr>
<td>CLB</td>
<td>Requests confidence limits</td>
</tr>
<tr>
<td>INCLUDE=</td>
<td>Includes effects in all models for model selection</td>
</tr>
<tr>
<td>INFORMATIVE</td>
<td>Models missing values by using extra indicator variables</td>
</tr>
<tr>
<td>LINK=</td>
<td>Specifies the link function</td>
</tr>
<tr>
<td>NOINT</td>
<td>Suppresses the intercept</td>
</tr>
<tr>
<td>OFFSET=</td>
<td>Specifies the offset variable</td>
</tr>
<tr>
<td>PRIOR=</td>
<td>Specifies prior probabilities</td>
</tr>
<tr>
<td>START=</td>
<td>Includes effects in the initial model for model selection</td>
</tr>
<tr>
<td>TYPE3</td>
<td>Displays the Type 3 or joint tests of effects</td>
</tr>
</tbody>
</table>

### Response Variable Options

Response variable options determine how the LOGSELECT procedure models probabilities for binary and multinomial response data. You can specify the following response-options by enclosing them in parentheses after the response or trials variable.

**DESCENDING**

DESC

reverses the order of the response categories. If you specify both the DESCENDING and ORDER= options, PROC LOGSELECT orders the response categories according to the ORDER= option and then reverses that order.
EVENT='category' | FIRST | LAST

specifies the event category for the binary response model. PROC LOGSELECT models the probability of the event category. The EVENT= option has no effect when there are more than two response categories.

You can specify one of the following:

'category'

specifies the value (formatted, if a format is applied) of the event category in quotation marks.

FIRST
designates the first ordered category as the event.

LAST
designates the last ordered category as the event.

By default, EVENT=FIRST.

For example, the following statements specify that observations with the formatted value ‘1’ represent events in the data. The probability that the LOGSELECT procedure models is thus the probability that the variable def takes on the (formatted) value ‘1’.

```sas
proc logselect data=mycas.MyData;
  class A B C;
  model def(event = '1') = A B C x1 x2 x3;
run;
```

ORDER=FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the response variable. When ORDER=FORMATTED (the default) for numeric variables for which you have supplied no explicit format (that is, for which there is no corresponding FORMAT statement in the current PROC LOGSELECT run or in the DATA step that created the data table), the levels are ordered by their internal (numeric) value. The following table shows how the ORDER= option is interpreted:

<table>
<thead>
<tr>
<th>ORDER=</th>
<th>Levels Sorted By</th>
</tr>
</thead>
<tbody>
<tr>
<td>FORMATTED</td>
<td>External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value; the sort order is machine-dependent.</td>
</tr>
<tr>
<td>FREQ</td>
<td>Descending frequency count (levels with the most observations come first in the order)</td>
</tr>
<tr>
<td>INTERNAL</td>
<td>Unformatted value; the sort order is machine-dependent.</td>
</tr>
</tbody>
</table>

By default, ORDER=FORMATTED.

For more information about sort order, see the chapter on the SORT procedure in the Base SAS Procedures Guide and the discussion of BY-group processing in SAS Language Reference: Concepts.

REF='category' | FIRST | LAST

specifies the reference category for the generalized logit model and the binary response model. For the generalized logit model, each logit contrasts a nonreference category with the reference category. For the binary response model, specifying one response category as the reference is the same as specifying the other response category as the event. You can specify one of the following:
Chapter 11: The LOGSELECT Procedure

'category'
specifies the value (formatted, if a format is applied) of the reference category in quotation marks.

FIRST
designates the first ordered category as the reference.

LAST
designates the last ordered category as the reference.

By default, REF=LAST.

Model Options

CENTER
requests that continuous main effects be centered and scaled internally. (Continuous main effects are centered and scaled to aid in computing maximum likelihood estimates.) Parameter estimates and related statistics are always reported on the original scale.

CENTERLASSO
requests that all effects, including categorical effects, be centered and scaled internally. (Effects are centered and scaled to aid in model selection by the LASSO method.) Parameter estimates and related statistics are always reported on the original scale.

CLB
constructs confidence limits for each of the parameter estimates. The confidence level is 0.95 by default; you can change it by specifying the ALPHA= option.

INCLUDE=n
INCLUDE=single-effect
INCLUDE=effect-list
forces effects to be included in all models. If you specify INCLUDE=n, then the first n effects that are listed in the MODEL statement are included in all models. If you specify INCLUDE=single-effect or if you specify INCLUDE=effect-list, then the specified effects are forced into all models. The effects that you specify in this option must be explanatory effects that are specified in the MODEL statement before the slash (/).

INFORMATIVE
models missing values by using extra model effects. These effects consist of dummy variables that take the value 1 when the value of a continuous model variable involved in the effect is missing, and take the value 0 otherwise. The missing value in the original model effect is replaced by the average value of the effect for the nonmissing values. For continuous-by-class effects, such as A*x, where A is a classification variable and x is a continuous variable, informative missingness creates multiple dummy columns and substitutes the effect mean of x that corresponds to the respective level of A. Missing values for classification variables are treated as valid levels. For more information about informative missingness, see the section “Informative Missingness” on page 78 in Chapter 3, “Shared Concepts.”

LINK=keyword
specifies the link function for the model. The default link is the logit. The keywords and the associated link functions are shown in Table 11.7.
Table 11.7  Built-In Link Functions of the LOGSELECT Procedure

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Link Function</th>
<th>( g(\pi) = \eta = )</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLOGLOG</td>
<td>Complementary log-log</td>
<td>( \log(-\log(1 - \pi)) )</td>
</tr>
<tr>
<td>GLOGIT</td>
<td>Generalized logit</td>
<td>( \log(\pi_j / \pi_j) )</td>
</tr>
<tr>
<td>LOGIT</td>
<td>Logit</td>
<td>( \log(\pi / (1 - \pi)) )</td>
</tr>
<tr>
<td>LOGLOG</td>
<td>Log-log</td>
<td>( -\log(-\log(\pi)) )</td>
</tr>
<tr>
<td>PROBIT</td>
<td>Probit</td>
<td>( \Phi^{-1}(\pi) )</td>
</tr>
</tbody>
</table>

For the probit and cumulative probit links, \( \Phi^{-1}(\cdot) \) denotes the quantile function of the standard normal distribution.

If the response variable has more than two categories, the LOGSELECT procedure fits a model that has a cumulative link function that is based on the specified link. However, if you specify LINK=GLOGIT, the procedure assumes a generalized logit model for nominal (unordered) data.

**NOINT**

requests that no intercept be included in the model. An intercept is included by default. The NOINT option is not available for multinomial models.

**OFFSET=variable**

specifies a variable to be used as an offset to the linear predictor. An offset plays the role of an effect whose coefficient is known to be 1. The offset variable cannot appear in the CLASS statement or elsewhere in the MODEL statement. Observations that have missing values for the offset variable are excluded from the analysis.

**PRIOR=number | number-list**

specifies prior probabilities (prevalences) that are used for computing posterior predicted probabilities. When you know what percentage of the population has a rare event and you oversample that rare event, specifying the prior probabilities as the prevalence of events in your population enables you to produce posterior probabilities that reflect the population, not the data. The priors do not affect the model-fitting process.

If you specify a number-list that has the same number of entries as there are response levels, then these values are scaled to sum to 1. If you specify a number-list that has one entry less than there are response levels, then the specified values should sum to less than 1, and the last response level is assigned the remaining value. For binary and binomial response models, you can thus specify the probability of an event as a single number. The scaled priors are displayed in the ResponseProfile table.

If your response \( Y \) takes values \( i = 1, \ldots, k \) that have observed empirical training probabilities \( \text{OldPrior}_i = n_i / n \), you specify priors \( \text{Prior}_i \), and your model predicted probabilities are \( \hat{p}_i \), then the posterior predicted probabilities \( \text{Post}_i \) are computed as

\[
\text{Post}_i = \frac{\hat{p}_i \text{Prior}_i}{\sum_{j=1}^{k} \hat{p}_j \text{Prior}_j / \text{OldPrior}_j}
\]

The POST= option in the OUTPUT statement writes the posterior to the output data set. If your priors are identical to the empirical probabilities, then the posteriors are identical to the model-predicted probabilities.
The priors adjust the “Classification” table statistics PPV, NPV, adequacy, percentage correct, misclassification rate, and lift as discussed in the section “Classification Table and ROC Curves” on page 586. If you specify a PARTITION statement, then the posterior probabilities and the preceding statistics for the validation and test data also use the specified priors.

**START=n**
**START=single-effect**
**START=(effects)**

begins the selection process from the designated initial model for the forward and stepwise selection methods. If you specify START=n, then the starting model includes the first n effects that are listed in the MODEL statement. If you specify START=single-effect or if you specify START=(effects), then the starting model includes those specified effects. The effects that you specify in the START= option must be explanatory effects that are specified in the MODEL statement before the slash (/). This option is not available when you specify METHOD=BACKWARD in the SELECTION statement.

**TYPE3**

requests that Wald statistics for Type 3 contrasts be computed for each effect that is specified in the MODEL statement. For more information, see the section “Joint Tests and Type 3 Tests” on page 592.

**OUTPUT Statement**

```
OUTPUT OUT=CAS-libref.data-table < options >
     < keyword =name >=...< keyword =name >>
```

The OUTPUT statement creates a data table that contains observationwise statistics that PROC LOGSELECT computes after fitting the model. In order to avoid data duplication for large data tables, the variables in the input data table are not included in the output data table unless you specify them in the COPYVAR= option.

If the response variable has more than two categories, you can request the “Statistic Options” listed in Table 11.8; the other diagnostic statistics are not available. These statistics are computed for every response category, and the automatic variable _LEVEL_ identifies the response category on which the computed values are based. That is, every observation generates several rows in the output data set. If you also specify the OBSCAT option, then the observationwise statistics are computed only for the observed response category, which is indicated by the value of the _LEVEL_ variable. If you specify the PREDPROBS option, then only the PRED and IPRED variables are output to the data table, and a variable is created to contain the values that correspond to each response category. PROC LOGSELECT names these variables by concatenating the PRED= or IPRED= name with the response category.

The output statistics are computed based on the final parameter estimates. If the optimization does not converge, then the output data table is not created.

For observations in which only the response variable is missing, values of the linear predictor and the predicted probabilities are computed even though these observations do not affect the model fit. This enables, for example, predicted probabilities to be computed for new observations.

You must specify the following option:
OUT=\textit{CAS-libref.data-table} names the output data table for PROC LOGSELECT to use. You must specify this option before any other options. \textit{CAS-libref.data-table} is a two-level name, where

\textit{CAS-libref} refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to where the data table is to be stored, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about \textit{CAS-libref}, see the section “Using CAS Sessions and CAS Engine Librefs” on page 551.

\textit{data-table} specifies the name of the output data table.

You can also specify the following options:

\textbf{ALL}

\textbf{ALLSTAT} adds all available statistics to the output data table.

\textbf{ALPHA=number} specifies the significance level for the construction of confidence intervals in the output data table. The confidence level is $1 - \text{number}$. The value of \text{number} must be between 0 and 1. By default, \text{number} is equal to the value of the \textbf{ALPHA=} option in the \textit{PROC LOGSELECT} statement, or 0.05 if that option is not specified.

\textbf{COPYVAR=variable}

\textbf{COPYVARS=(variables)} transfers one or more \textit{variables} from the input data table to the output data table.

\textbf{FIXEDOFFSET=number} specifies a value to use as an offset to the linear predictor instead of obtaining the value from the offset variable. This option is ignored unless you also specify the \textbf{OFFSET=} variable in the \textit{MODEL} statement.

\textbf{OBSCAT} requests (for multinominal models) that observationwise statistics be produced only for the observed response level. If you do not specify this option and the response variable has $J$ levels, then the following outputs are created: for cumulative link models, $J-1$ records are output for every observation in the input data that corresponds to the $J-1$ lower-ordered response categories; for generalized logit models, $J$ records are output that correspond to all $J$ response categories.

\textbf{PREDPROBS} creates variables (for multinominal models) for each response category that corresponds to the requested \textbf{IPRED} and \textbf{PREDICTED} keywords.

\textit{keyword <= name>} specifies a statistic to include in the output data table and optionally names the variable \textit{name}. If you do not provide a \textit{name}, the LOGSELECT procedure assigns a default name based on the type of statistic requested.

Table 11.8 summarizes the \textit{keywords} that are available in the \textit{OUTPUT} statement.
### Table 11.8 OUTPUT Statement Keywords

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
<th>Default Names</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Statistic Options</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INDIVIDUAL</td>
<td>Specifies the individual predicted probabilities</td>
<td><em>IPRED</em></td>
</tr>
<tr>
<td>LCL</td>
<td>Specifies the lower confidence limit for the linear predictor</td>
<td><em>LCL</em></td>
</tr>
<tr>
<td>LCLM</td>
<td>Specifies the lower confidence limit for the event probability</td>
<td><em>LCLM</em></td>
</tr>
<tr>
<td>POST</td>
<td>Specifies the posterior predicted probabilities</td>
<td><em>POST</em></td>
</tr>
<tr>
<td>PREDICTED</td>
<td>Specifies the predicted probabilities</td>
<td><em>PRED</em></td>
</tr>
<tr>
<td>RESRAW</td>
<td>Specifies the raw residual</td>
<td><em>RESRAW</em></td>
</tr>
<tr>
<td>STDXBETA</td>
<td>Specifies the standard error estimate of the linear predictor</td>
<td><em>STDXBETA</em></td>
</tr>
<tr>
<td>UCL</td>
<td>Specifies the upper confidence limit for the linear predictor</td>
<td><em>UCL</em></td>
</tr>
<tr>
<td>UCLM</td>
<td>Specifies the upper confidence limit for the event probability</td>
<td><em>UCLM</em></td>
</tr>
<tr>
<td>XBETA</td>
<td>Specifies the linear predictor</td>
<td><em>XBETA</em></td>
</tr>
<tr>
<td><strong>Diagnostic Options for Binary and Binomial Response Data</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CBAR</td>
<td>Specifies the confidence interval displacement</td>
<td><em>CBAR</em></td>
</tr>
<tr>
<td>DIFCHISQ</td>
<td>Specifies the deletion chi-square goodness-of-fit change</td>
<td><em>DIFCHISQUARE</em></td>
</tr>
<tr>
<td>DIFDEV</td>
<td>Specifies the deletion deviance change</td>
<td><em>DIFDEVIANCE</em></td>
</tr>
<tr>
<td>H</td>
<td>Specifies the leverage</td>
<td><em>HATDIAG</em></td>
</tr>
<tr>
<td>RESCHI</td>
<td>Specifies the Pearson chi-square residual</td>
<td><em>RESCHI</em></td>
</tr>
<tr>
<td>RESDEV</td>
<td>Specifies the deviance residual</td>
<td><em>RESDEV</em></td>
</tr>
<tr>
<td>RESLIK</td>
<td>Specifies the likelihood residual</td>
<td><em>RESLIK</em></td>
</tr>
<tr>
<td>RESWORK</td>
<td>Specifies the working residual</td>
<td><em>RESWORK</em></td>
</tr>
<tr>
<td>STDRESCHI</td>
<td>Specifies the standardized Pearson chi-square residual</td>
<td><em>STDRESCHI</em></td>
</tr>
<tr>
<td>STDRESDEV</td>
<td>Specifies the standardized deviance residual</td>
<td><em>STDRESDEV</em></td>
</tr>
<tr>
<td><strong>Miscellaneous Options</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTO</td>
<td>Names the level into which the observation is classified</td>
<td><em>INTO</em></td>
</tr>
<tr>
<td>LEVEL</td>
<td>Names the response level for a row of the output</td>
<td><em>LEVEL</em></td>
</tr>
<tr>
<td>ROLE</td>
<td>Names the role played by the observation in fitting the model</td>
<td><em>ROLE</em></td>
</tr>
</tbody>
</table>

The following list describes these *keywords*. For more information, see the section “Predicted Probabilities and Regression Diagnostics” on page 589.

**CBAR**

specifies the confidence interval displacement diagnostic that measures the overall change in the global regression estimates that results from deleting an individual observation. The default name is _CBAR_.

**DIFCHISQ**

specifies the change in the chi-square goodness-of-fit statistic that results from deleting the individual observation. The default name is _DIFCHISQUARE_.

**DIFDEV**

specifies the change in the deviance that results from deleting the individual observation. The default name is _DIFDEVIANCE_.
H
specifies the diagonal element of the hat matrix (leverage) for detecting extreme points in the
design space. The default name is _HATDIAG_.

INDIVIDUAL
IPRED
IPROB
IP
specifies the individual predicted values for multinomial response variables. For a response
variable Y with three levels, 1, 2, and 3, the individual probabilities are Pr(Y = 1), Pr(Y = 2),
and Pr(Y = 3). The default name is _IPRED_.

INTO<(cutpoint)>
names the variable that contains the level of the response into which an observation is classified.
The default name is _INTO_. Multinomial models classify observations into the level that has the
largest model-predicted probability. For binary or binomial response variables, if the predicted
probability of an observation equals or exceeds the cutpoint, the observation is classified as an
event; otherwise it is classified as a nonevent. You can specify the cutpoint value as a number
between 0 and 1. The default value is 0.5.

LCL
LOWERXBETA
names the variable that contains the lower confidence limits for the linear predictor. You can set
the confidence level by specifying the ALPHA= option. The default name is _LCL_.

LCLM
LOWERMEAN
LOWER
specifies the lower confidence limits for the probability of the event. You can set the confidence
level by specifying the ALPHA= option. The default name is _LCLM_.

LEVEL
names the variable that contains the level of the response for a given row of the output. The
default name is _LEVEL_.

POSTERIOR
POST
specifies the posterior predicted probability of each observation. If you do not specify the PRIOR=
option in the MODEL statement, then the observed proportions from the training data are used as
priors, in which case the posteriors from the training data are the same as the individual predicted
probabilities. The default name is _POST_.

PREDICTED
PRED
PROB
P
specifies the predicted values (predicted probabilities of events) for binary and nominal response
variables and the cumulative predicted probabilities for ordinal response variables. For a response
variable Y with three levels, 1, 2, and 3, the cumulative probabilities are Pr(Y ≤ 1) and Pr(Y ≤ 2),
but by default the last level, Pr(Y ≤ 3) = 1, is not output. The default name is _PRED_.
RESCHI

PEARSON specifies the Pearson residual for identifying poorly fitted observations. The default name is _RESCHI_.

RESDEV

specifies the deviance residual for identifying poorly fitted observations. The default name is _RESDEV_.

RESLIK

specifies the likelihood residual for identifying poorly fitted observations. The default name is _RESLIK_.

RESRAW

RESIDUAL R

specifies the raw residual for identifying poorly fitted observations. The default name is _RESRAW_.

RESWORK

specifies the working residual for identifying poorly fitted observations. The default name is _RESWORK_.

ROLE

specifies the numeric variable that indicates the role played by each observation in fitting the model. The default name is _ROLE_. Table 11.9 shows how this variable is interpreted for each observation.

Table 11.9 Role Interpretation

<table>
<thead>
<tr>
<th>Value</th>
<th>Observation Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Not used</td>
</tr>
<tr>
<td>1</td>
<td>Training</td>
</tr>
<tr>
<td>2</td>
<td>Validation</td>
</tr>
<tr>
<td>3</td>
<td>Testing</td>
</tr>
</tbody>
</table>

If you do not partition the input data by specifying a PARTITION statement, then the role variable value is 1 for observations that are used in fitting the model and 0 for observations that have at least one missing or invalid value for the response, regressor, frequency, or weight variables.

STDRESCHI

specifies the standardized Pearson (chi-square) residual for identifying observations that are poorly accounted for by the model. The default name is _STDRESCHI_.

STDRESDEV

specifies the standardized deviance residual for identifying poorly fitted observations. The default name is _STDRESDEV_.

STDXBETA
specifies the standard error estimates of XBETA. The default name is _STDXBETA_.

UCL
UPPERXBETA
specifies the variable that contains the upper confidence limits for the linear predictor. The default name is _UCL_. You can set the confidence level by specifying the ALPHA= option.

UCLM
UPPERMEAN
UPPER
specifies the variable that contains the upper confidence limits for the probability of the event response. The default name is _UCLM_. You can set the confidence level by specifying the ALPHA= option.

XBETA
LINP
specifies the linear predictor. The default name is _XBETA_.

PARTITION Statement
PARTITION partition-option ;
The PARTITION statement specifies how observations in the input data set are logically partitioned into disjoint subsets for model training, validation, and testing. For more information, see the section “Using Validation and Test Data” on page 80 in Chapter 3, “Shared Concepts.” Either you can designate a variable in the input data table and a set of formatted values of that variable to determine the role of each observation, or you can specify proportions to use for randomly assigning observations to each role.

You must specify exactly one of the following partition-options:

FRACTION(< TEST=fraction > < VALIDATE=fraction > < SEED=number >)
randomly assigns specified proportions of the observations in the input data table to the roles. You specify the proportions for testing and validation by using the TEST= and VALIDATE= suboptions. If you specify both the TEST= and VALIDATE= suboptions, then the sum of the specified fractions must be less than 1 and the remaining fraction of the observations are assigned to the training role. The SEED= option specifies an integer that is used to start the pseudorandom number generator for random partitioning of data for training, testing, and validation. If you do not specify SEED=number or if number is less than or equal to 0, the seed is generated by reading the time of day from the computer’s clock.

ROLE=variable (< TEST='value' > < TRAIN='value' > < VALIDATE='value' >)
ROLEVAR=variable (< TEST='value' > < TRAIN='value' > < VALIDATE='value' >)
names the variable in the input data table whose values are used to assign roles to each observation. This variable cannot also appear as an analysis variable in other statements or options. The TEST=, TRAIN=, and VALIDATE= suboptions specify the formatted values of this variable that are used to assign observation roles. If you do not specify the TRAIN= suboption, then all observations whose role is not determined by the TEST= or VALIDATE= suboption are assigned to the training role.
For more information, see the section “Partition Fit Statistics” on page 584. For an illustration, see Example 11.3.

---

**SELECTION Statement**

```
SELECTION < METHOD=method< (method-options)> > < options > ;
```

The SELECTION statement performs model selection by examining whether effects should be added to or removed from the model according to rules that are defined by model selection methods. The statement is fully documented in the section “SELECTION Statement” on page 36 in Chapter 3, “Shared Concepts.”

The LOGSELECT procedure supports the following effect-selection methods in the SELECTION statement:

- **BACKWARD** performs backward elimination. This method starts with all effects in the model and deletes effects.
- **BACKWARD(FAST)** performs fast backward elimination. This method starts with all effects in the model and deletes effects without refitting the model.
- **FORWARD** performs forward selection. This method starts with no effects in the model and adds effects.
- **LASSO** performs model selection by the group LASSO method. This method adds and removes effects by using a sequence of LASSO steps. For more information, see the section “Group LASSO Selection” on page 69 in Chapter 3, “Shared Concepts.”
- **NONE** results in no model selection. This method fits the full model.
- **STEPWISE** performs stepwise selection. This method is similar to the FORWARD method except that effects already in the model do not necessarily stay there.

By default, METHOD=STEPWISE.

The SELECT=, CHOOSE=, and STOP= method-options default to SBC. These defaults differ from their corresponding defaults in the HPLLOGISTIC procedure in SAS/STAT software.

You can specify the following criteria in the SELECT=, CHOOSE=, and STOP= method-options:

- **AIC** uses Akaike’s information criterion (Akaike 1974) computed on the training data.
- **AICC** uses a small-sample bias-corrected version of Akaike’s information criterion, as promoted in Hurvich and Tsai (1989) and Burnham and Anderson (1998), computed on the training data.
- **SBC | BIC** uses the Schwarz Bayesian criterion (Schwarz 1978) computed on the training data.
- **SL** uses the significance level of the score test computed on the training data as the criterion (not available for the CHOOSE= option).
- **VALIDATE** uses the average square error (ASE) computed on the validation data as the criterion (not available for the SELECT= option).

For more information, see the section “Information Criteria” on page 585.
If you specify METHOD=LASSO and you do not specify either the CHOOSE= or STOP= option, then the model in the last LASSO step is chosen as the selected model.

If you specify METHOD=LASSO, then the STOPHORIZON= option has no effect.

**NOTE:** If you use the fast backward elimination method, then the –2 log likelihood, AIC, AICC, and SBC statistics are approximated at each step where the model is not refit, and hence they do not match the values that are computed when that model is fit outside the selection routine. Similarly, if you specify SELECT=AIC, AICC, or SBC, the selection criteria are estimated (Lawless and Singhal 1978), and hence they do not match the values that are computed when that model is fit outside the selection routine.

**NOTE:** The default model hierarchy method is HIERARCHY=NONE for the stepwise, forward, and fast backward selection methods. The backward elimination method always uses the HIERARCHY=SINGLE method option.

The LASSO method produces a summary table that displays the effects that are added or removed at each step; the LASSO regularization parameter; and the AIC, AICC, and SBC fit statistics. For the other methods, you can specify the following values for the DETAILS= option:

- **SUMMARY** produces a summary table that shows the effect that is added or removed at each step along with the SELECT=, CHOOSE=, and STOP= criteria. The summary table is produced by default if the DETAILS= option is not specified.
- **STEPS** produces the preceding summary table and displays the results from fitting each model at each step.
- **ALL** produces the preceding tables and a detailed listing of all candidates at each step along with their ranking in terms of the selection criterion for addition to or removal from the model.

If you specify the PLOTS=CRITERIA or PLOTS=ALL option, then a plot of the fit criterion by the selection step is created for the AIC, AICC, and SBC statistics. If you also specify a PARTITION statement or the PARTFIT option, then the same type of plot is created for the ASE (for each role), max-rescaled R-square, McFadden’s R-square, misclassification rate, and difference of means statistics.

If you specify the PLOTS=FITBYROLE or PLOTS=ALL option and a PARTITION statement, then a plot of the ASE by the selection step for each role is created.

The PLOTS= option is not available for the LASSO method.

---

### WEIGHT Statement

**WEIGHT variable ;**

The *variable* in the WEIGHT statement is used as a weight to perform a weighted analysis of the data. Observations that have nonpositive or missing weights are not included in the analysis. If a WEIGHT statement is not included, all observations that are used in the analysis are assigned a weight of 1.
Details: LOGSELECT Procedure

Missing Values

Any observation that has missing values for the response, frequency, weight, offset, or explanatory variables is excluded from the analysis; however, missing values are valid for response and explanatory variables that are specified along with the MISSING option in the CLASS statement. Observations that have a nonpositive weight or a frequency less than 1 are also excluded.

The estimated linear predictor and the fitted probabilities are not computed for any observation that has missing offset or explanatory variable values. However, if only the response value is missing, the linear predictor and the fitted probabilities can be computed and output to a data table by using the OUTPUT statement.

You can also model the missing values by specifying the INFORMATIVE option in the MODEL statement. For more information about informative missingness, see the section “Informative Missingness” on page 78 in Chapter 3, “Shared Concepts.”

Response Distributions

The response distribution is the probability distribution of the response (target) variable. The LOGSELECT procedure can fit data for the following distributions:

- binary distribution
- binomial distribution
- multinomial distribution

The expressions for the log-likelihood functions of these distributions are given in the next section.

The binary (or Bernoulli) distribution is the elementary distribution of a discrete random variable that can take two values, which have the probabilities $p$ and $1 - p$. Suppose the random variable is denoted as $Y$ and

\[
\Pr(Y = 0) = p \\
\Pr(Y = 1) = 1 - p
\]

The value that is associated with probability $p$ is often called the event or “success”; the complementary event is called the nonevent or “failure.” A Bernoulli experiment is a random draw from a binary distribution and generates events with probability $p$.

If $Y_1, \ldots, Y_n$ are $n$ independent Bernoulli random variables, then their sum follows a binomial distribution. In other words, if $Y_i = 1$ denotes an event (success) in the $i$th Bernoulli trial, a binomial random variable is the number of events (successes) in $n$ independent Bernoulli trials. If you use the events/trials syntax in the MODEL statement, the LOGSELECT procedure fits the model as if the data had arisen from a binomial distribution. For example, the following statements fit a binomial regression model with regressors $x_1$ and $x_2$. The variables $e$ and $t$ represent the events and trials for the binomial distribution.
If the events/trials syntax is used, both variables must be numeric and the value of the events variable cannot be less than 0 or exceed the value of the trials variable.

The multinomial distribution is a generalization of the binary distribution and allows for more than two outcome categories. Because there are more than two possible outcomes for the multinomial distribution, the terminology of “successes,” “failures,” “events,” and “nonevents” no longer applies. With multinomial data, these outcomes are generically referred to as “categories” or levels.

Whenever the LOGSELECT procedure determines that the response variable has more than two levels (unless the events/trials syntax is used), it fits the model as if the data had arisen from a multinomial distribution. By default, the procedure then assumes that the response categories are ordered, and it fits a cumulative link model by applying the default or specified link function. If the response categories are unordered, then you should fit a generalized logit model by specifying LINK=GLOGIT in the MODEL statement.

Log-Likelihood Functions

The LOGSELECT procedure forms the log-likelihood functions of the various models as

\[ L(\mu; y) = \sum_{i=1}^{n} f_i l(\mu_i; y_i, w_i) \]

where \( \mu_i \) is the probability that the \( i \)th observation is an event, \( l(\mu_i; y_i, w_i) \) is the log-likelihood contribution of the \( i \)th observation, \( w_i \) is the weight, and \( f_i \) is its frequency. For the determination of \( w_i \) and \( f_i \), see the WEIGHT and FREQ statements. The individual log-likelihood contributions for the various distributions are described in the following sections.

Binary Distribution

The LOGSELECT procedure computes the log-likelihood function \( l(\mu_i(\beta); y_i) \) for the \( i \)th binary observation as

\[ \eta_i = x_i^{\top} \beta \]
\[ \mu_i(\beta) = g^{-1}(\eta_i) \]
\[ l(\mu_i(\beta); y_i) = y_i \log\{\mu_i\} + (1 - y_i) \log\{1 - \mu_i\} \]

where the variable \( y_i \) takes the value 1 for an event and 0 for a nonevent. The inverse link function \( g^{-1}(\cdot) \) maps from the scale of the linear predictor \( \eta_i \) to the scale of the mean. For example, for the logit link (the default),

\[ \mu_i(\beta) = \frac{\exp\{\eta_i\}}{1 + \exp\{\eta_i\}} \]
You can control which binary outcome in your data is modeled as the event by using the `response-options` in the `MODEL` statement, and you can choose the link function by using the `LINK=` option in the `MODEL` statement.

If a `WEIGHT` statement is specified and \( w_i \) denotes the weight for the current observation, the log-likelihood function is computed as

\[
l(\mu_i(\beta); y_i, w_i) = w_i l(\mu_i(\beta); y_i)
\]

### Binomial Distribution

The LOGSELECT procedure computes the log-likelihood function \( l(\mu_i(\beta); y_i) \) for the \( i \)th binomial observation as

\[
\eta_i = x_i^T \beta \\
\mu_i(\beta) = g^{-1}(\eta_i) \\
l(\mu_i(\beta); y_i, w_i) = w_i \left( y_i \log(\mu_i) + (n_i - y_i) \log(1 - \mu_i) \right) \\
+ w_i \log \left( \frac{n_i}{y_i} \right)
\]

where \( y_i \) and \( n_i \) are the values of the events and trials of the \( i \)th observation, respectively. The value \( \mu_i \) measures the probability of events (successes) in the underlying Bernoulli distribution whose aggregate follows the binomial distribution.

### Multinomial Distribution

The multinomial distribution that the LOGSELECT procedure models is a generalization of the binary distribution; it is the distribution of a single draw from a discrete distribution that has \( J \) possible values. Thus, the log-likelihood function for the \( i \)th observation is

\[
l(\mu_i, y_i, w_i) = w_i \sum_{j=1}^{J} y_{ij} \log(\mu_{ij})
\]

In this expression, \( J \) denotes the number of response categories (the number of possible outcomes) and \( \mu_{ij} \) is the probability that the \( i \)th observation takes on the response value associated with category \( j \). The category probabilities must satisfy

\[
\sum_{j=1}^{J} \mu_j = 1
\]

and the constraint is satisfied by modeling \( J - 1 \) categories. In models that have ordered response categories, the probabilities are expressed in cumulative form, so the last category is redundant. In generalized logit models (multinomial models that have unordered categories), one category is chosen as the reference category, and the linear predictor in the reference category is set to 0. For more information, see the `REF=` `response-option` in the `MODEL` statement.
Existence of Maximum Likelihood Estimates

The likelihood equation for a logistic regression model does not always have a finite solution. Sometimes there is a nonunique maximum on the boundary of the parameter space, at infinity. The existence, finiteness, and uniqueness of maximum likelihood estimates for the logistic regression model depend on the patterns of data points in the observation space (Albert and Anderson 1984; Santner and Duffy 1986).

Consider a binary response model. Let \( Y_j \) be the response of the \( j \)th subject, and let \( x_j \) be the vector of explanatory variables (including the constant 1 that is associated with the intercept). There are three mutually exclusive and exhaustive types of data configurations:

**Complete separation** There is a complete separation of data points if there exists a vector \( \mathbf{b} \) that correctly allocates all observations to their response groups; that is,

\[
\begin{align*}
  b'x_j &> 0 & Y_j &= 0 \\
  b'x_j &< 0 & Y_j &= 1
\end{align*}
\]

This configuration produces nonunique infinite estimates. If the iterative process of maximizing the likelihood function is allowed to continue, then the log likelihood diminishes to 0 and the dispersion matrix becomes unbounded.

**Quasi-complete separation** The data are not completely separable, but there is a vector \( \mathbf{b} \) such that

\[
\begin{align*}
  b'x_j &\geq 0 & Y_j &= 0 \\
  b'x_j &\leq 0 & Y_j &= 1
\end{align*}
\]

and equality holds for at least one subject in each response group. This configuration also yields nonunique infinite estimates. If the iterative process of maximizing the likelihood function is allowed to continue, then the dispersion matrix becomes unbounded and the log likelihood diminishes to a nonzero constant.

**Overlap** If neither complete nor quasi-complete separation exists in the sample points, there is an overlap of sample points. In this configuration, the maximum likelihood estimates exist and are unique.

The LOGSELECT procedure uses a simple empirical approach to recognize the data configurations that lead to infinite parameter estimates. The basis of this approach is that any convergence method of maximizing the log likelihood must yield a solution that indicates complete separation, if such a solution exists. Upon convergence, if the predicted response equals the observed response for every observation, there is a complete separation of data points.

If the data are not completely separated, an observation is identified to have an extremely large probability (\( \geq 0.95 \)) of predicting the observed response, and if there have been at least eight iterations, then there are two possible situations. First, there is overlap in the data set, the observation is an atypical observation of its own group, and the iterative process stopped when a maximum was reached. Second, there is quasi-complete separation in the data set, and the asymptotic dispersion matrix is unbounded. If any of the diagonal elements of the dispersion matrix for the standardized observation vector (all explanatory variables standardized to zero mean and unit variance) exceeds 5,000, then PROC LOGSELECT declares quasi-complete separation; if any of the diagonal elements exceeds 1,000, then the procedure displays a message indicating that quasi-complete separation might be detectable by increasing the number of iterations. If either complete separation or quasi-complete separation is detected, a note is displayed in the procedure output.

Checking for quasi-complete separation is less foolproof than checking for complete separation. If neither type of separation is discovered and your parameter estimates have large standard errors, then your data might
be separable. The NOCHECK option in the MODEL statement turns off the process of checking for infinite parameter estimates; the MINITER= option in the PROC LOGSELECT statement increases the number of iterations.

The LASSO Method of Model Selection

LASSO Selection

The LOGSELECT procedure implements the group LASSO method, which is described in the section “Group LASSO Selection” on page 69 in Chapter 3, “Shared Concepts.” The current section provides some background about the LASSO method that you need in order to understand the group LASSO method.

LASSO (least absolute shrinkage and selection operator) selection arises from a constrained form of ordinary least squares regression in which the sum of the absolute values of the regression coefficients is constrained to be smaller than a specified parameter. More precisely, let $X = (x_1, x_2, \ldots, x_m)$ denote the matrix of covariates, and let $y$ denote the response. Then for a given parameter $t$, the LASSO regression coefficients $\beta = (\beta_1, \beta_2, \ldots, \beta_m)$ are the solution to the following constrained least squares problem:

$$\min ||y - X\beta||^2 \quad \text{subject to} \quad \sum_{j=1}^{m} |\beta_j| \leq t$$

For generalized linear models, the LASSO regression coefficients $\beta = (\beta_1, \beta_2, \ldots, \beta_m)$ are the solution to the following constrained optimization problem,

$$\min \{ -L(\mu; y) \} \quad \text{subject to} \quad \sum_{j=1}^{m} |\beta_j| \leq t$$

where $L$ is the log-likelihood function defined in the section “Log-Likelihood Functions” on page 581.

Provided that the LASSO parameter $t$ is small enough, some of the regression coefficients will be exactly zero. Hence, you can think of the LASSO method as selecting a subset of the regression coefficients for each LASSO parameter. By increasing the LASSO parameter in discrete steps, you obtain a sequence of regression coefficients for which the nonzero coefficients at each step correspond to selected parameters. For more information about the LASSO method, see, for example, Hastie, Tibshirani, and Friedman (2009).

Partition Fit Statistics

Specifying a PARTITION statement modifies the display of many tables by adding separate rows or columns for the training, validation, and test data tables, and adds statistics to the “Fit Statistics” table. These additional statistics are useful for assessing the model and should be very similar for the different roles when the training data are representative of the other data partitions: average square error, misclassification rate, and the difference of means. The $R^2$, maximum-rescaled $R^2$, and McFadden’s $R^2$ are also computed. For more information, see the section “Model Fit and Assessment Statistics” on page 585.
Model Fit and Assessment Statistics

The statistics that are defined in this section are useful for assessing the fit of the model to your data; they are displayed in the “Fit Statistics” table. The statistics are computed for each data role when you specify a PARTITION statement.

Information Criteria

The calculation of the information criteria uses the following formulas, where \( p \) denotes the number of effective parameters in the candidate model, \( F \) denotes the sum of frequencies used, and \( l \) is the log likelihood evaluated at the converged estimates:

\[
\begin{align*}
\text{AIC} & = -2l + 2p \\
\text{AICC} & = \begin{cases} 
   -2l + 2pF/(F - p - 1) & \text{when } F > p + 2 \\
   -2l + 2p(p + 2) & \text{otherwise}
\end{cases} \\
\text{SBC} & = -2l + p \log(F)
\end{align*}
\]

If you do not specify a FREQ statement, \( F \) equals \( n \), the number of observations used.

Generalized Coefficient of Determination

The goal of a coefficient of determination, also known as an R-square measure, is to express the agreement between a stipulated model and the data in terms of variation in the data that is explained by the model. In linear models, the R-square measure is based on residual sums of squares; because these are additive, a measure bounded between 0 and 1 is easily derived.

In more general models where parameters are estimated by the maximum likelihood principle, Cox and Snell (1989, pp. 208–209) and Magee (1990) proposed the following generalization of the coefficient of determination:

\[
R^2 = 1 - \left\{ \frac{L(\mathbf{0})}{L(\hat{\theta})} \right\}^{\frac{2}{n}}
\]

Here, \( L(\mathbf{0}) \) is the likelihood of the intercept-only model, \( L(\hat{\theta}) \) is the likelihood of the specified model, and \( n \) denotes the number of observations used in the analysis. This number is adjusted for frequencies if a FREQ statement is present, and it is based on the trials variable for binomial models.

As discussed in Nagelkerke (1991), this generalized R-square measure has properties similar to those of the coefficient of determination in linear models. If the model effects do not contribute to the analysis, \( L(\hat{\theta}) \) approaches \( L(\mathbf{0}) \) and \( R^2 \) approaches zero.

However, \( R^2 \) does not have an upper limit of 1. Nagelkerke suggested a rescaled generalized coefficient of determination, \( R^2_N \), which achieves an upper limit of 1 by dividing \( R^2 \) by its maximum value:

\[
\begin{align*}
R^2_{\text{max}} & = 1 - \left\{ L(\mathbf{0}) \right\}^{\frac{2}{n}} \\
R^2_N & = \frac{R^2}{R^2_{\text{max}}}
\end{align*}
\]
Another measure, from McFadden (1974), is also bounded by 0 and 1:

\[ R^2_M = 1 - \left( \frac{\log L(\hat{\beta})}{\log L(\theta)} \right) \]

These measures are most useful for comparing competing models that are not necessarily nested—that is, models that cannot be reduced to one another by simple constraints on the parameter space. Larger values of the measures indicate better models.

**Average Square Error**

The average square error (ASE) is the average of the squared differences between the responses and the predictions. When you have a discrete number of response levels, the ASE is modified as shown in Table 11.10 (Brier 1950; Murphy 1973); it is also called the Brier score or Brier reliability.

<table>
<thead>
<tr>
<th>Response Type</th>
<th>ASE (Brier Score)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary</td>
<td>[ \frac{1}{n} \sum_i f_i (y_i (1 - \hat{\pi}_i)^2 + (1 - y_i)\hat{\pi}_i^2) ]</td>
</tr>
<tr>
<td>Binomial</td>
<td>[ \frac{1}{F} \sum_i f_i (r_i / t_i - \hat{\pi}_i)^2 ]</td>
</tr>
<tr>
<td>Multinomial</td>
<td>[ \frac{1}{F} \sum_i f_i \sum_j (y_{ij} - \hat{\pi}_{ij})^2 ]</td>
</tr>
</tbody>
</table>

In Table 11.10, \( F = \sum_j f_j \), \( r_i \) is the number of trials in binomial response models, \( y_i = 1 \) for events and 0 for nonevents in binary response models, and \( \hat{\pi}_i \) is the predicted probability of an event. For polytomous response models, \( y_{ij} = 1 \) if the \( i \)th observation has response level \( j \), and \( \pi_{ij} \) is the model-predicted probability of response level \( j \) for observation \( i \).

**Difference of Means**

For a binary response model, write the mean of the model-predicted probabilities of event (\( Y=0 \)) observations as \( \bar{X}_1 = \sum_{i=1}^{n_1} f_i \hat{\pi}_i \) and of nonevent (\( Y=1 \)) observations as \( \bar{X}_2 = \sum_{i=1}^{n_2} f_i \hat{\pi}_i \), where \( \hat{\pi}_i \) is the predicted probability of an event. The difference of means is \( \bar{X}_1 - \bar{X}_2 \), which Tjur (2009) relates to other R-square measures and calls the coefficient of discrimination, because it is a measure of the model’s ability to distinguish between the event and nonevent distributions. The difference of means is also the \( d' \) or \( \Delta m \) statistic (with unit standard error) that is discussed in the signal detection literature (McNicol 2005).

**Classification Table and ROC Curves**

For binary response data, the response \( Y \) is either an event or a nonevent; let the response \( Y \) take the value 1 for an event and 2 for a nonevent. From the fitted model, a predicted event probability \( \hat{\pi}_i \) can be computed for each observation \( i \). Define your decision rule as follows: if the predicted event probability equals or exceeds a cutpoint value \( z \in [0, 1] \), the observation is classified as an event; otherwise, it is classified as a nonevent. Suppose \( n_1 \) of \( n \) individuals experience an event, such as a disease, and the remaining \( n_2 = n - n_1 \) individuals do not experience that event (are nonevents). The \( 2 \times 2 \) classification (confusion, decision, error) matrix in Table 11.11 is obtained by cross-classifying the observed and predicted responses, where \( n_{ij} \) is the total number of observations that are observed to have \( Y = i \) and are classified into \( j \). In this table, let \( Y = 1 \) denote an observed event and \( Y = 2 \) denote a nonevent, and let \( D = 1 \) indicate that the observation is classified as an event and \( D = 2 \) denote that the observation is classified as a nonevent.
Table 11.11  Classification Matrix

<table>
<thead>
<tr>
<th></th>
<th>(D = 1 (\hat{\pi} \geq z))</th>
<th>(D = 2 (\hat{\pi} &lt; z))</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Y = 1) (event)</td>
<td>(n_{11})</td>
<td>(n_{12})</td>
<td>(n_1)</td>
</tr>
<tr>
<td>(Y = 2) (nonevent)</td>
<td>(n_{21})</td>
<td>(n_{22})</td>
<td>(n_2)</td>
</tr>
</tbody>
</table>

The cells of the classification matrix of Table 11.11 are as follows:

\[
\begin{align*}
n_{11} &= \text{the number of true positives, which is the number of event observations that are correctly classified as events} \\
n_{21} &= \text{the number of false positives, which is the number of nonevent observations that are incorrectly classified as events} \\
n_{12} &= \text{the number of false negatives, which is the number of event observations that are incorrectly classified as nonevents} \\
n_{22} &= \text{the number of true negatives, which is the number of nonevent observations that are correctly classified as nonevents} \\
n_1 &= \text{the total number of actual events} \\
n_2 &= \text{the total number of actual nonevents}
\end{align*}
\]

The accuracy of the classification is measured by its ability to predict events and nonevents correctly. Sensitivity (true positive fraction, TPF, recall) is the proportion of event responses that are predicted to be events. Specificity (true negative fraction, 1–FPF) is the proportion of nonevent responses that are predicted to be nonevents.

You can also measure accuracy by how well the classification predicts the response. The positive predictive value (precision, PPV) is the proportion of observations classified as events that are correctly classified. The negative predictive value (NPV) is the proportion of observations classified as nonevents that are correctly classified. The correct classification rate (accuracy, PC) is the proportion of observations that are correctly classified, whereas the misclassification rate (error rate) is the proportion of observations that are incorrectly classified. The lift is the ratio of the proportion of correctly classified events to the proportion of observations classified as events.

The prevalence, \(\Pr(Y = 1)\), is the prior probability of an event. If the prevalence is different from the observed empirical event probability in the training data, \(n_1/n\), then applying Bayes’ theorem shows that the PPV, NPV, accuracy, PC, misclassification rate, and lift equations depend on the prevalence (Fleiss, Levin, and Paik 2003). For a stratified sampling situation in which \(n_1\) and \(n_2\) are chosen a priori, \(n_1/n\) is not a desirable estimate of \(\Pr(Y = 1)\). You can specify the prevalence by using the PRIOR= option. If you specify a PARTITION statement without specifying the PRIOR= option, then the observed empirical probabilities for the training data are used as prevalences for computations of the validation and test statistics.

PROC LOGSELECT constructs the data for a receiver operating characteristic (ROC) curve by initially rounding the predicted probabilities to the nearest multiple of the value of the BINEPS= option. This effectively sorts the observations in increasing order of their estimated event probability. A classification matrix is created for each of these bins by using the rounded probability as the cutpoint. As the cutpoint moves from 0 to 1, those cutpoints for which the classification matrix changes are selected.

Alternatively, if you do not want to generate the entire ROC curve, you can use the CUTPT= option to specify your own list of cutpoints. In this case, the specified cutpoints are used to generate the classification matrices without rounding the predicted probabilities.
You can specify the CTABLE option to produce a classification table that includes these cutpoints and, for each cutpoint, any of the statistics in Table 11.12 that you request.

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Equation</th>
<th>Column Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cutpoint</td>
<td>$z$</td>
<td>ProbLevel</td>
</tr>
<tr>
<td>Number of true positives</td>
<td>$n_{11}$</td>
<td>TruePos</td>
</tr>
<tr>
<td>Number of true negatives</td>
<td>$n_{22}$</td>
<td>TrueNeg</td>
</tr>
<tr>
<td>Number of false positives</td>
<td>$n_{21}$</td>
<td>FalsePos</td>
</tr>
<tr>
<td>Number of false negatives</td>
<td>$n_{12}$</td>
<td>FalseNeg</td>
</tr>
<tr>
<td>True positive fraction (sensitivity)</td>
<td>$n_{11}/n_1$</td>
<td>TPF</td>
</tr>
<tr>
<td>False positive fraction (1–specificity)</td>
<td>$n_{21}/n_2$</td>
<td>FPF</td>
</tr>
<tr>
<td>True negative fraction</td>
<td>$n_{22}/n_2$</td>
<td>TNF</td>
</tr>
<tr>
<td>False negative fraction</td>
<td>$n_{12}/n_1$</td>
<td>FNF</td>
</tr>
<tr>
<td>Correct classification rate</td>
<td>$(n_{11} + n_{22})/n$</td>
<td>Accuracy</td>
</tr>
<tr>
<td>Percentage correct (PC)</td>
<td>100×Accuracy</td>
<td>PC</td>
</tr>
<tr>
<td>Misclassification rate</td>
<td>1 – Accuracy</td>
<td>Misclass</td>
</tr>
<tr>
<td>Positive predictive value</td>
<td>$n_{11}/(n_{11} + n_{21})$</td>
<td>PPV</td>
</tr>
<tr>
<td>Negative predictive value</td>
<td>$n_{22}/(n_{12} + n_{22})$</td>
<td>NPV</td>
</tr>
<tr>
<td>Lift</td>
<td>$TPF/((n_{11} + n_{21})/n)$</td>
<td>Lift</td>
</tr>
</tbody>
</table>

You can output this classification table by specifying the OUT= option, and you can display the ROC curve by using the SG PLOT procedure as shown in Example 11.2.

The area under the ROC curve (AUC), as determined by the trapezoidal rule, is given by the concordance index $c$, which is described in the section “Association Statistics” on page 588.

For more information about the topics in this section, see Pepe (2003).

**Association Statistics**

If you specify the ASSOCIATION option in the PROC LOGSELECT statement, PROC LOGSELECT displays measures of association between predicted probabilities and observed responses for binary, binomial, and ordinal response models. These measures assess the predictive ability of a model.

For ordinal response data, let the predicted mean score of an observation be the sum of the OrderedValue values (shown in the “Response Profile” table) minus one, weighted by the corresponding predicted probabilities for that observation; that is, the predicted means score $= \sum_{i=1}^{k+1}(i-1)\hat{\pi}_i$, where $k + 1$ is the number of response levels and $\hat{\pi}_i$ is the predicted probability of the $i$th (ordered) response.

For binary and binomial responses, let the predicted mean score be the predicted event probability.

The predicted mean score is rounded to the nearest multiple of the value that you specify in the BINEPS= option. This effectively sorts the observations in increasing order of their predicted mean score. Of the $n$ pairs of observations in the data that have different responses, let $n_c$ be the number of pairs where the observation that has the lower-ordered response value has a lower predicted mean score, let $n_d$ be the number of pairs where the observation that has the lower-ordered response value has a higher predicted mean score, and let $n_t = n - n_c - n_d$ be the rest of the observations. Let $N$ be the sum of observation frequencies in the data.
Then the following statistics are reported:

- Concordance index \( c \) (AUC) = \( (n_c + 0.5n_t)/n \)
- Somers’ \( D \) (Gini coefficient) = \( (n_c - n_d)/n \)
- Goodman-Kruskal gamma = \( (n_c - n_d)/(n_c + n_d) \)
- Kendall’s tau-\( a \) = \( (n_c - n_d)/(0.5N(N - 1)) \)

If there are no ties, then Somers’ \( D \) (Gini’s coefficient) = 2\( c - 1 \). For binary responses, the concordance index, \( c \), is an estimate of the AUC, which is the area under the ROC curve.

If you specify a PARTITION statement, then PROC LOGSELECT displays a column for each of the roles.

---

**Predicted Probabilities and Regression Diagnostics**

For binary response data, you can produce observationwise predicted probabilities, confidence limits, and regression diagnostics developed by Pregibon (1981) by specifying the OUTPUT statement. For multinomial response data, you can likewise produce observationwise predicted probabilities, confidence limits, and raw residuals.

For a binary response model, given a vector of covariates \( x_i \) for the \( i \)th observation in your data table and the model-predicted parameter estimates \( \hat{\beta} \), you can write the linear predictor \( \hat{\eta}_i = x'_i \hat{\beta} \). The mean of the \( i \)th observation \( \mu_i(\hat{\beta}) \), or the model-predicted event probability \( \hat{\pi}_i \), is \( \mu_i(\hat{\beta}) = \hat{\pi}_i = g^{-1}(\eta_i) \), where the link function \( g \) is chosen by specifying the LINK= option. The variance of the binary distribution is \( V(\mu) = \mu(1 - \mu) = \hat{\pi}_i(1 - \hat{\pi}_i) = V(\pi) \), and \( \Sigma \) is the estimated covariance of \( \hat{\beta} \). Denote the frequency of the \( i \)th observation as \( f_i \) and the weight as \( w_i \).

For ordinal response models, the predicted cumulative probabilities are computed in the same fashion by using the appropriate model-predicted intercept parameters \( \hat{\alpha}_j \) and letting \( \hat{\beta} \) consist of the slope parameters: 
\[
\hat{\eta}_{ij} = g(\Pr(Y \leq j | x_i)) = \hat{\alpha}_j + x'_i \hat{\beta} \text{ and } \hat{\pi}_{ij} = \Pr(Y \leq j | x_i) = g^{-1}(\eta_{ij}) \text{ for } 1 \leq j < J.
\]

For nominal response models, the predicted probabilities are computed by using the appropriate model-predicted intercept parameters \( \hat{\alpha}_j \) and letting \( \hat{\beta}_j \) consist of the slope parameters: 
\[
\hat{\eta}_{ij} = g(\Pr(Y = j | x_i)) = \hat{\alpha}_j + x'_i \hat{\beta}_j \text{ and } \hat{\pi}_{ij} = \Pr(Y = j | x_i) = g^{-1}(\eta_{ij}) \text{ for } 1 \leq j < J.
\]

**Confidence Intervals**

Approximate confidence intervals for predicted probabilities can be computed as follows. The variance of the linear predictor is estimated by
\[
\hat{\sigma}^2(\eta_i) = x'_i \Sigma x_i
\]

For multinomial models, the variance also depends on the response function. Let \( \delta_j \) be a \((J-1)\) column vector whose \( j \)th entry is equal to 1 and all other entries are equal to 0. Redefine \( x_i = (\delta'_j, x'_i)' \). \( \eta_i = \eta_{ij} \), and \( \pi_i = \pi_{ij} \). Then
\[
\hat{\sigma}^2(\eta_i) = x'_i \Sigma x_i
\]

The asymptotic 100(1 - \( \alpha \))% confidence interval for \( \eta_i \) is
\[
\hat{\eta}_i \pm z_{\alpha/2} \hat{\sigma}(\hat{\eta}_i)
\]
where $z_{\alpha/2}$ is the $100(1 - \alpha/2)$th percentile point of a standard normal distribution.

The predicted probability and the $100(1 - \alpha)\%$ confidence limits for $\pi_i$ are obtained by back-transforming the corresponding measures for the linear predictor. So the confidence limits are

$$g^{-1}\left[\hat{\pi}_i \pm z_{\alpha/2} \hat{\sigma}(\hat{\pi}_i)\right]$$

**Hat-Matrix Diagonals**

The diagonal elements of the hat matrix are useful in detecting extreme points in the design space, where they tend to have larger values. For the generalized linear model, the variance of the $i$th individual observation is

$$v_i = \frac{V(\pi_i)}{f_i w_i}$$

For the $i$th observation, let

$$w_{ei} = v_i^{-1} (g'(\pi_i))^{-2}$$

where $g'(\pi_i)$ is the derivative of the link function evaluated at $\pi_i$. The weight matrix $W_e$ is a diagonal matrix, with $w_{ei}$ denoting the $i$th diagonal element, which is used in computing the expected information matrix.

Define the leverage, or hat-matrix diagonal, $h_i$, as the $i$th diagonal element of the matrix

$$W_e^{1/2} X (X' W_e X)^{-1} X' W_e^{1/2}$$

If the estimated probability is extreme (less than 0.1 and greater than 0.9, approximately), then the hat-matrix diagonal might be greatly reduced in value. Consequently, when an observation has a very large or very small estimated probability, its leverage is not a good indicator of the observation’s distance from the design space (Hosmer and Lemeshow 2000, p. 171).

**Residuals**

Residuals are useful in identifying observations that are not explained well by the model. For binary and binomial response data, the raw residual is

$$r_i = y_i / t_i - \hat{\pi}_i$$

where $y_i$ is the number of event responses out of $t_i$ trials for the $i$th observation. For single-trial syntax, $t_i = 1$ and $y_i = 1$ if the ordered response is 1 and $y_i = 0$ otherwise. For multinomial response data, the raw residual is

$$y_{ij} - \hat{\pi}_{ij}$$

where $y_{ij} = 1$ if the $i$th observation has response level $j$ and $y_{ij} = 0$ otherwise, and $\hat{\pi}_{ij}$ are the model-predicted probabilities of response level $j$ for observation $i$.

The Pearson residual is the square root of the $i$th observation’s contribution to Pearson’s chi-square:

$$r_{Pi} = r_i \sqrt{\frac{f_i w_i}{V(\pi_i)}}$$
The deviance residual is the square root of the contribution of the $i$th observation to the deviance, with the sign of the raw residual,

$$r_{Di} = (\text{sign}(r_i)) \sqrt{d_i}$$

where

$$d_i = 2f_i w_i t_i \left[ \frac{y_i / t_i}{\pi_i} \log \left( \frac{y_i / t_i}{\pi_i} \right) + \left( 1 - \frac{y_i / t_i}{1 - \pi_i} \right) \right]$$

The working residual is

$$r_{Wi} = r_i \left( \frac{\partial \pi_i}{\partial \eta_i} \right)^{-1}$$

The Pearson residuals, standardized to have unit asymptotic variance, are

$$r_{SPi} = \frac{r_{Pi}}{\sqrt{1 - h_i}}$$

The deviance residuals, standardized to have unit asymptotic variance, are

$$r_{SDi} = \frac{r_{Di}}{\sqrt{1 - h_i}}$$

The likelihood residuals, which estimate components of a likelihood ratio test of deleting an individual observation, are a weighted combination of the standardized Pearson and deviance residuals:

$$r_{Li} = \text{sign}(r_i) \sqrt{h_i r_{SPi}^2 + (1 - h_i) r_{SDi}^2}$$

**Other Regression Diagnostics**

The CBAR statistic is a confidence interval displacement diagnostic that provides a scalar measure of the influence of an individual observation on $\hat{\beta}$. This diagnostic is based on the same idea as the Cook distance in linear regression theory (Cook and Weisberg 1982), but it uses the one-step estimate:

$$\overline{C}_i = r_{Pi}^2 h_i / (1 - h_i)$$

The DIFDEV and DIFCHISQ statistics are diagnostics for detecting ill-fitted observations—observations that contribute heavily to the disagreement between the data and the predicted values of the fitted model. DIFDEV is the change in the deviance that results from deleting an individual observation, and DIFCHISQ is the change in the Pearson chi-square statistic that results from the same deletion. By using the one-step estimate, DIFDEV and DIFCHISQ for the $i$th observation are computed as follows:

$$\text{DIFDEV}_i = r_{Di}^2 + \overline{C}_i$$
$$\text{DIFCHISQ}_i = \overline{C}_i / h_i$$
Joint Tests and Type 3 Tests

Linear hypotheses for \( \beta \) are expressed in matrix form as

\[
H_0: L\beta = c
\]

where \( L \) is a matrix of coefficients for the linear hypotheses and \( c \) is a vector of constants. The vector of regression coefficients \( \beta \) includes slope parameters and intercept parameters. The Wald chi-square statistic for testing \( H_0 \) is computed as

\[
\chi^2_W = (L\hat{\beta} - c)'[L\hat{V}(\hat{\beta})L']^{-1}(L\hat{\beta} - c)
\]

where \( \hat{V}(\hat{\beta}) \) is the estimated covariance matrix. Under \( H_0 \), \( \chi^2_W \) has an asymptotic chi-square distribution with \( r \) degrees of freedom, where \( r \) is the rank of \( L \).

For models that use less-than-full-rank parameterization (as specified by the PARAM=GLM option in the CLASS statement), a Type 3 test of an effect of interest (main effect or interaction) is a test of the Type III estimable functions that are defined for that effect. When the model contains no missing cells, the Type 3 test of a main effect is equivalent to testing the hypothesis of equal marginal means. For more information about Type III estimable functions, see the chapter “The GLM Procedure” and the section “The Four Types of Estimable Functions” in SAS/STAT User’s Guide. Also see Littell, Freund, and Spector (1991).

For models that use full-rank parameterization, all parameters are estimable when there are no missing cells, so it is unnecessary to define estimable functions. The standard test of an effect of interest in this case is the joint test that the values of the parameters associated with that effect are 0. For a model that uses effects parameterization (as specified by the PARAM=EFFECT option in the CLASS statement), the joint test for a main effect is equivalent to testing the equality of marginal means. For a model that uses reference parameterization (as specified by the PARAM=REF option in the CLASS statement), the joint test is equivalent to testing the equality of cell means at the reference level of the other model effects. For more information about the coding scheme and the associated interpretation of results, see Muller and Fetterman (2002, Chapter 14).

If there is no interaction term, the Type 3 test of an effect for a model that uses GLM parameterization is the same as the joint test of the effect for the model that uses full-rank parameterization. In this situation, the joint test is also called the Type 3 test. For a model that contains an interaction term and no missing cells, the Type 3 test of a component main effect under GLM parameterization is the same as the joint test of the component main effect under effect parameterization. Both test the equality of cell means. But this Type 3 test differs from the joint test under reference parameterization, which tests the equality of cell means at the reference level of the other component main effect. If some cells are missing, you can obtain meaningful tests only by testing a Type III estimation function, so in this case you should use GLM parameterization.

The results of a Type 3 test or a joint test do not depend on the order in which you specify the terms in the MODEL statement.

Multithreading

The LOGSELECT procedure allocates data to different threads and calculates the likelihood function, gradient, and Hessian by accumulating the values from all threads. For more information about how PROC LOGSELECT uses threads, see the section “Multithreading” on page 81 in Chapter 3, “Shared Concepts.”
Optimization Algorithms

Several optimization techniques are available in PROC LOGSELECT. You can choose a particular optimizer by using the TECHNIQUE= option in the PROC LOGSELECT statement. Table 11.13 summarizes the optimization techniques available in PROC LOGSELECT.

<table>
<thead>
<tr>
<th>TECHNIQUE=</th>
<th>Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRUREG</td>
<td>Trust region method</td>
</tr>
<tr>
<td>NEWRAP</td>
<td>Newton-Raphson method with line search</td>
</tr>
<tr>
<td>NRRIDG</td>
<td>Newton-Raphson method with ridging</td>
</tr>
<tr>
<td>QUANEW</td>
<td>Quasi-Newton methods</td>
</tr>
<tr>
<td>DBLDOG</td>
<td>Double-dogleg method</td>
</tr>
<tr>
<td>CONGRA</td>
<td>Conjugate gradient methods</td>
</tr>
<tr>
<td>NMSIMP</td>
<td>Nelder-Mead simplex method</td>
</tr>
</tbody>
</table>

There is no algorithm for optimizing general nonlinear functions that always finds the global optimum for a general nonlinear optimization problem in a reasonable amount of time. Because no single optimization technique is always superior to others, PROC LOGSELECT provides a variety of optimization techniques that work well in various circumstances. However, you can devise problems for which none of the techniques in PROC LOGSELECT can find the correct solution. Moreover, nonlinear optimization can be computationally expensive in terms of time and memory, so you must be careful when matching an algorithm to a problem. The section “Choosing an Optimization Algorithm” on page 82 in Chapter 3, “Shared Concepts,” is helpful in choosing a suitable optimization algorithm.

Memory Usage

In order to use memory efficiently in computing the components for optimization, PROC LOGSELECT reads your data table in multiple batches, or pages; each page contains at most a certain number of observations. During the optimization, the procedure reads the first page of observations from the data table into memory, creates the design rows, performs the appropriate log-likelihood, gradient, and Hessian computations on that page of observations, and then discards those observations and reads in the next page of data for processing.

Generally, smaller pages use less memory but can lead to longer computation times, whereas larger pages can run faster but use more memory. In particular, with sufficient memory, the optimization is typically fastest if you can fit all your data on a single page, because then you do not have to repeatedly reaccess the data table and recompute the design rows. The default PAGEOBS=AUTO option determines that you have enough memory to use one page for the optimization if the number of observations in your data table for each thread on each machine node is less than

$$\min(b, 1\text{GB})/(8\text{prt})$$

where
$b$ = the approximate number of bytes available to a machine node, or 1GB
$t$ = the number of threads available to a machine node
$p$ = the number of parameters in your model
$r = 2n + 10$ for continuous response models
$r = 2n + y + 9$ for categorical response models
$n$ = the number of observations in the data table
$y$ = the number of response levels for categorical response models

If PROC LOGSELECT determines that you do not have enough memory, then PAGEOBS=256 is chosen to optimize the matrix computations, and a note is displayed in the SAS log that gives the value of the PAGEOBS= option that is required if you want to store all the data for a machine node on a single page.

---

**Displayed Output**

The following sections describe the output that PROC LOGSELECT produces. The output is organized into various tables, which are discussed in their order of appearance.

**Model Information**

The “Model Information” table displays basic information about the model, such as the response variable, the frequency variable, the link function, and the distribution of the data that is assumed by the LOGSELECT procedure. For information about how PROC LOGSELECT determines the response distribution, see the section “Response Distributions” on page 580. If you specify the CODE statement, then the table displays the name of the variable in the scoring code that contains the classifications. If you do not also specify the PCATALL option, then the table also displays the name of the variable in the scoring code that contains the predicted probabilities.

**Number of Observations**

The “Number of Observations” table displays the number of observations read from the input data table and the number of observations used in the analysis. If a FREQ statement is present, the table displays the sum of the frequencies that are read and used. If a WEIGHT statement is specified, the table displays the sum of $f_i w_i$ that are read and used. If you specify a PARTITION statement, the table displays the values for each role.

**Response Profile**

The “Response Profile” table displays the ordered value from which the LOGSELECT procedure determines the probability being modeled as an event in a binary model and displays the ordering of categories in a multinomial model. For each response category level, the frequency used in the analysis is reported. If you also specify the PRIOR= option, the scaled prior probabilities are also displayed. You can affect the ordering of the response values by specifying response-options in the MODEL statement. For a binary model, the note that follows the “Response Profile” table indicates which outcome is modeled as the event. For an ordinal multinomial model, the note that follows the “Response Profile” table indicates how the ordered response levels are accumulated; for a nominal multinomial model, the note indicates the reference response level. If you specify a PARTITION statement, the table displays the values for each role.
Class Level Information
The “Class Level Information” table lists the levels of every variable specified in the CLASS statement. You should check this information to make sure that the data are correct. You can adjust the order of the CLASS variable levels by using the ORDER= option in the CLASS statement. You can suppress the “Class Level Information” table completely or partially by using the NOCLPRINT= option in the PROC LOGSELECT statement.

If the classification variables use a nonsingular parameterization, the “Class Level Information” table also displays the reference value for each variable.

Selection Information
When you specify the SELECTION statement, the LOGSELECT procedure by default produces a series of tables that display information about the model selection. The “Selection Information” table informs you about the model selection method, selection and stop criteria, and other parameters that govern the selection. You can suppress this table by specifying DETAILS=NONE in the SELECTION statement.

Iteration History
When you specify the ITHIST option in the PROC LOGSELECT statement, the “Iteration History” table displays, for each iteration of the optimization, the number of function evaluations (including gradient and Hessian evaluations), the value of the objective function, the change in the objective function from the previous iteration, and the absolute value of the largest (projected) gradient element. The objective function that is used in the optimization in the LOGSELECT procedure is normalized by default to enable comparisons across data tables that have different sampling intensity. You can control normalization by specifying the NORMALIZE= option in the PROC LOGSELECT statement.

Convergence Status
The convergence status table is a small ODS table that appears as a message that indicates whether the optimization succeeded and which convergence criterion was met. If the optimization fails, the message indicates the reason for the failure. If you save the convergence status table to an output data set, a numeric Status variable is added that enables you to assess convergence programatically. The values of the Status variable indicate the following:

0  Convergence was achieved, or an optimization was not performed (because TECHNIQUE=NONE is specified).
1  The objective function could not be improved.
2  Convergence was not achieved because of a user interruption or because a limit was exceeded, such as the maximum number of iterations or the maximum number of function evaluations. To modify these limits, see the MAXITER=, MAXFUNC=, and MAXTIME= options in the PROC LOGSELECT statement.
3  Optimization failed to converge because function or derivative evaluations failed at the starting values or during the iterations or because a feasible point that satisfies the parameter constraints could not be found in the parameter space.
Entry and Removal Candidates

When you specify DETAILS=ALL or DETAILS=STEPS in the SELECTION statement, the LOGSELECT procedure produces the “Entry Candidates” or “Removal Candidates” table, which displays the effect names and values of the criterion used to select entering or departing effects at each step of the selection process. For each step, the effects are displayed in sorted order from best to worst of the selection criterion.

Selection Summary

When you specify the SELECTION statement, the LOGSELECT procedure produces the “Selection Summary” table, which displays information about which effects were added to or removed from the model in the various steps of the model selection process. The statistic that led to the entry or removal decision is also displayed. You can request further details about the model selection steps by specifying DETAILS=STEPS or DETAILS=ALL in the SELECTION statement. You can suppress the display of the “Selection Summary” table by specifying DETAILS=NONE in the SELECTION statement.

If you specify the LASSO selection method, then this table displays information about which effect was added to the model, the number of effects in the model, the lambda value, and the information criteria.

Stop Reason

When you specify the SELECTION statement, the LOGSELECT procedure produces a simple table that tells you why model selection stopped.

Selection Reason

When you specify the SELECTION statement, the LOGSELECT procedure produces a simple table that tells you why the final model was selected.

Selected Effects

When you specify the SELECTION statement, the LOGSELECT procedure produces a simple table that tells you which effects are in the final model.

Dimensions

The “Dimensions” table displays size measures that are derived from the model. It displays the number of columns in the current design matrix, the number of effects in the current design, the largest number of design columns associated with an effect, the rank of the matrix, and the number of parameters in the current model, including any scale parameters.

Global Test

The “Global Test” table provides a likelihood ratio test for the hypothesis of whether the final model provides a better fit than a model without effects (an “intercept-only” model). This table is not produced when the NOSTDERR option is specified nor when LASSO selection is performed.

If you specify the NOINT option in the MODEL statement, the reference model is one where the linear predictor is 0 for all observations.
**Fit Statistics**

The “Fit Statistics” table displays a variety of likelihood-based measures of fit. The values that the “Fit Statistics” table displays are not based on a normalized log-likelihood function. If you specify a PARTITION statement or the PARTFIT option, the table displays the values for each role along with statistics for comparing the training, validation, and testing results.

For more information about the statistics displayed in this table, see the section “Model Fit and Assessment Statistics” on page 585.

**Parameter Estimates**

The parameter estimates, their estimated (asymptotic) standard errors, and p-values for the hypothesis that the parameter is 0 are presented in the “Parameter Estimates” table. If you request confidence intervals by specifying the CLB option in the MODEL statement, confidence limits are produced for the estimates.

**Parameter Estimates Covariance Matrix**

When you specify the COVB option in the PROC LOGSELECT statement, the LOGSELECT procedure displays the covariance matrix of the parameter estimates. The covariance matrix is computed as the inverse of the negative of the matrix of second derivatives of the log-likelihood function with respect to the model parameters (the Hessian matrix), evaluated at the parameter estimates.

**Parameter Estimates Correlation Matrix**

When you specify the CORRB option in the PROC LOGSELECT statement, the LOGSELECT procedure displays the correlation matrix of the parameter estimates.

**Model Analysis of Variance (Type III)**

When you specify the TYPE3 option in the MODEL statement, the LOGSELECT procedure produces the “Model Analysis of Variance (Type III)” table. This table displays tests that all parameters for a particular effect are equal to zero. For more information, see the section “Joint Tests and Type 3 Tests” on page 592.

**Association Statistics**

When you specify the ASSOCIATION option in the PROC LOGSELECT statement and have binary, binomial, or ordinal response data, the “Association Statistics” table displays the concordance index c (the area under the ROC curve, AUC), Somers’ D statistic (Gini’s coefficient), Goodman-Kruskal’s gamma statistic, and Kendall’s tau-α statistic, along with raw counts of pairs of observations. If you also specify a PARTITION statement, a row is displayed for each role. For more information, see the section “Association Statistics” on page 588.

**Classification Table**

The “Classification” table is displayed if you specify the CTABLE option without specifying an output data set. If you also specify a PARTITION statement, a table is displayed for each role. For more information, see the section “Classification Table and ROC Curves” on page 586.
Score Code Variables for Predicted Probability

When you specify the PCATALL option in the CODE statement, the LOGSELECT procedure produces the “Score Code Variables for Predicted Probability” table. For each level of the response variable, this table displays the corresponding variable in the scoring code that contains the predicted probability.

Timing

The “Timing” table displays the amount of time (in seconds) that PROC LOGSELECT required to perform different tasks in the analysis.

OutputCasTables Table

The OutputCasTables table is a special table that has information about each CAS table that is created during a CAS action execution. The information for each CAS table consists of the CAS table name, the caslib in which the table resides, and the number of columns and rows in the CAS table. Because this table is not a typical ODS table that contains analytical results, you cannot include it in the table-spec-list in the DISPLAYOUT statement.

ODS Table Names

Each table that the LOGSELECT procedure creates has a name associated with it. You must use this name to refer to the table when you use the DISPLAY statement, the DISPLAYOUT statement, or ODS statements. These names are listed in Table 11.14.

NOTE: The EFFECT statement also create tables which are not listed in this section. For information about these tables, see “ODS Table Names” on page 35 in Chapter 3, “Shared Concepts.”

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>Association</td>
<td>Association of predicted probabilities and observed responses</td>
<td>PROC LOGSELECT</td>
<td>ASSOCIATION</td>
</tr>
<tr>
<td>Classification</td>
<td>Classification table</td>
<td>PROC LOGSELECT</td>
<td>CTABLE</td>
</tr>
<tr>
<td>ClassInfo</td>
<td>Level information from the CLASS statement</td>
<td>CLASS</td>
<td>Default</td>
</tr>
<tr>
<td>ConvergenceStatus</td>
<td>Status of optimization at conclusion of optimization</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>CorrB</td>
<td>Correlation matrix of parameter estimates</td>
<td>PROC LOGSELECT</td>
<td>CORRB</td>
</tr>
<tr>
<td>CovB</td>
<td>Covariance matrix of parameter estimates</td>
<td>PROC LOGSELECT</td>
<td>COVB</td>
</tr>
<tr>
<td>Dimensions</td>
<td>Model dimensions</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>EntryCandidates</td>
<td>Details about candidates for entry into the model</td>
<td>SELECT</td>
<td>METHOD=FORWARD or STEPWISE and DETAILS=STEP</td>
</tr>
<tr>
<td>Table Name</td>
<td>Description</td>
<td>Statement</td>
<td>Option</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------------------------------------------------------</td>
<td>-------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>FitStatistics</td>
<td>Fit statistics</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>GlobalTest</td>
<td>Test of the model versus the null model</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>IterHistory</td>
<td>Iteration history</td>
<td>PROC LOGSELECT</td>
<td>ITHIST</td>
</tr>
<tr>
<td>ModelInfo</td>
<td>Information about the modeling environment</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>ModelAnova</td>
<td>Model analysis of variance (Type III)</td>
<td>MODEL TYPE3</td>
<td></td>
</tr>
<tr>
<td>NObs</td>
<td>Number of observations read and used, and number of events and trials, if applicable</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>OutputCasTables</td>
<td>See the section “OutputCasTables Table” on page 598</td>
<td>OUTPUT DISPLAYOUT</td>
<td>OUT=</td>
</tr>
<tr>
<td>ParameterEstimates</td>
<td>Solutions for the parameter estimates associated with effects in MODEL statements</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>PredProbName</td>
<td>Displays the predicted probability variable in the scoring code associated with each response level</td>
<td>CODE PCATALL</td>
<td></td>
</tr>
<tr>
<td>RemovalCandidates</td>
<td>Details about candidates for removal from the model</td>
<td>SELECTION METHOD=BACKWARD or STEPWISE and DETAILS=STEP</td>
<td></td>
</tr>
<tr>
<td>ResponseProfile</td>
<td>Response categories for binary and multinomial data</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>SelectedEffects</td>
<td>List of effects selected for the model</td>
<td>SELECTION</td>
<td>Default</td>
</tr>
<tr>
<td>SelectionInfo</td>
<td>Information about the settings for model selection</td>
<td>SELECTION</td>
<td>Default</td>
</tr>
<tr>
<td>SelectionReason</td>
<td>Reason why the particular model was selected</td>
<td>SELECTION</td>
<td>Default</td>
</tr>
<tr>
<td>SelectionSummary</td>
<td>Summary information about model selection steps</td>
<td>SELECTION</td>
<td>Default</td>
</tr>
<tr>
<td>StopReason</td>
<td>Reason for termination of model selection</td>
<td>SELECTION</td>
<td>Default</td>
</tr>
<tr>
<td>Timing</td>
<td>Absolute and relative times for tasks performed by the procedure</td>
<td>Default</td>
<td></td>
</tr>
</tbody>
</table>
ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in the “Statistical Graphics Using ODS” chapter in SAS/STAT User’s Guide.

Before you create graphs, ODS Graphics must be enabled (for example, with the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section “Enabling and Disabling ODS Graphics” in that chapter.

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section “A Primer on ODS Statistical Graphics” in that chapter.

When ODS Graphics is enabled, the SELECTION statement can produce plots to help evaluate the selection process. For information about these plots, see the section “Model Selection Plots” on page 71 in Chapter 3, “Shared Concepts.”

PROC LOGSELECT assigns a name to each graph that it creates using ODS. You can use these names to reference the graphs when using ODS. The names are listed in Table 11.15.

Table 11.15  Graphs Produced by PROC LOGSELECT

<table>
<thead>
<tr>
<th>ODS Graph Name</th>
<th>Plot Description</th>
<th>PLOTS Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoefficientPanel</td>
<td>Coefficients and CHOOSE= criterion by step</td>
<td>COEFFICIENTS</td>
</tr>
<tr>
<td>ChooseCriterionPlot</td>
<td>CHOOSE= criterion by step</td>
<td>COEFFICIENTS(UNPACK)</td>
</tr>
<tr>
<td>CoefficientPlot</td>
<td>Coefficients by step</td>
<td>COEFFICIENTS(UNPACK)</td>
</tr>
<tr>
<td>CriterionPanel</td>
<td>Fit criteria by step</td>
<td>CRITERIA</td>
</tr>
<tr>
<td>AdjRSqPlot</td>
<td>Max-rescaled (adjusted) R-square by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>AICCPlot</td>
<td>Corrected Akaike’s information criterion by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>AICPlot</td>
<td>Akaike’s information criterion by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>DiffMeanPlot</td>
<td>Difference of means by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>McFaddenPlot</td>
<td>McFadden’s R-square by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>MisclassPlot</td>
<td>Misclassification rate by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>SBCPlot</td>
<td>Schwarz Bayesian information criterion by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>TEST_ASEPlot</td>
<td>Average square error on testing data by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>TRAIN_ASEPlot</td>
<td>Average square error on training data by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>VAL_ASEPlot</td>
<td>Average square error on validation data by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>FitByRolePlot</td>
<td>Progression of average square error by role</td>
<td>FITBYROLE</td>
</tr>
</tbody>
</table>
Examples: LOGSELECT Procedure

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

Example 11.1: Model Selection

The following statements examine the same data as in the section “Getting Started: LOGSELECT Procedure” on page 551, but they request model selection via the forward selection technique. Effects that provide the best improvement to the selection criterion, SBC, are added until no more effects can improve the selection criterion. The DETAILS=ALL option in the SELECTION statement produces all tables that are related to model selection. The PLOTS=ALL option produces graphics to help you interpret the selection process. ODS Graphics must be enabled before you can request plots. For more information about ODS Graphics, see the section “ODS Graphics” on page 600.

```sas
ods graphics on;
proc logselect data=mycas.getStarted;
  class C;
  model y = C x1-x10;
  selection method=forward details=all plots=all;
run;
ods graphics off;
```

The model selection tables are shown in Output 11.1.1 through Output 11.1.3. Results from the selected model are shown in Output 11.1.4 and Output 11.1.5. Selection graphics that the PLOTS= option produces are displayed in Output 11.1.6 and Output 11.1.7.

The “Selection Information” table in Output 11.1.1 summarizes the settings for the model selection. Effects are added to the model only if they produce a significant improvement, which is determined by comparing the values of their SBCs. The forward selection stops three steps after the smallest SBC is obtained or when all effects have been added to the model.

**Output 11.1.1 Selection Information**

The LOGSELECT Procedure

<table>
<thead>
<tr>
<th>Selection Information</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Selection Method</strong></td>
</tr>
<tr>
<td><strong>Select Criterion</strong></td>
</tr>
<tr>
<td><strong>Stop Criterion</strong></td>
</tr>
<tr>
<td><strong>Effect Hierarchy Enforced</strong></td>
</tr>
<tr>
<td><strong>Stop Horizon</strong></td>
</tr>
</tbody>
</table>
For each step of the selection process, the DETAILS=ALL option displays the candidate effects for entering the model along with their SELECT= criterion. Output 11.1.2 displays this table for the first step; the other steps are not shown here.

**Output 11.1.2**  Step1 Entry Candidates

<table>
<thead>
<tr>
<th>Rank</th>
<th>Effect</th>
<th>SBC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>x8</td>
<td>128.7318</td>
</tr>
<tr>
<td>2</td>
<td>x2</td>
<td>129.0423</td>
</tr>
<tr>
<td>3</td>
<td>x4</td>
<td>129.3708</td>
</tr>
<tr>
<td>4</td>
<td>x9</td>
<td>131.0854</td>
</tr>
<tr>
<td>5</td>
<td>x1</td>
<td>131.8215</td>
</tr>
<tr>
<td>6</td>
<td>x10</td>
<td>132.4430</td>
</tr>
<tr>
<td>7</td>
<td>x5</td>
<td>132.9424</td>
</tr>
<tr>
<td>8</td>
<td>x3</td>
<td>132.9667</td>
</tr>
<tr>
<td>9</td>
<td>x7</td>
<td>133.0021</td>
</tr>
<tr>
<td>10</td>
<td>x6</td>
<td>133.0283</td>
</tr>
</tbody>
</table>

The DETAILS=ALL option also displays the dimensions, global test, fit statistics, and parameter estimates at each step of the selection process; these details are not shown here.

When the selection procedure is complete, the “Selection Summary” table in Output 11.1.3 shows the effects that were added to the model and the value of their selection criterion (and the choose and stop criteria, if they are specified). Step 0 refers to the null model that contains only an intercept. In the next step, effect x8 made the most significant contribution to the model among the candidate effects, according to the SBC statistic. In step 2, x2 made the most significant contribution when an effect was added to a model that contains the intercept and x8. In the three subsequent steps, no effect could be added to the model that would reduce the SBC, so variable selection stopped because the stop horizon (see Output 11.1.1) indicates that at most three steps beyond the minimum SBC value are taken.

In Output 11.1.3, the “Selection Summary” table is followed by three small tables that summarize why the process stopped and which model is selected.

**Output 11.1.3**  Selection Summary Information

<table>
<thead>
<tr>
<th>Step</th>
<th>Effect</th>
<th>Number Entered</th>
<th>Number Effects In</th>
<th>SBC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Intercept</td>
<td>1</td>
<td>128.4253</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>x8</td>
<td>2</td>
<td>128.7318</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>x2</td>
<td>3</td>
<td>128.2892*</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>x4</td>
<td>4</td>
<td>130.0901</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>x9</td>
<td>5</td>
<td>131.9534</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>x1</td>
<td>6</td>
<td>134.7945</td>
<td></td>
</tr>
</tbody>
</table>

* Optimal Value Of Criterion

Selection stopped at a local minimum of the STOP criterion.

The model at step 2 is selected.

**Selected Effects:** Intercept x2 x8
Output 11.1.4 displays information about the selected model. Notice that the \(-2\) log-likelihood value in the “Fit Statistics” table is larger than the value for the full model in Figure 11.9. This is expected because the selected model contains only a subset of the parameters. Because the selected model is more parsimonious than the full model, the discrepancy between the \(-2\) log likelihood and the information criteria is less severe than previously noted.

**Output 11.1.4**  Fit Statistics and Null Test

<table>
<thead>
<tr>
<th>Dimensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Columns in Design</td>
</tr>
<tr>
<td>Number of Effects</td>
</tr>
<tr>
<td>Max Effect Columns</td>
</tr>
<tr>
<td>Rank of Design</td>
</tr>
<tr>
<td>Parameters in Optimization</td>
</tr>
</tbody>
</table>

### Testing Global Null Hypothesis: BETA=0

<table>
<thead>
<tr>
<th>Test</th>
<th>DF</th>
<th>Chi-Square</th>
<th>Pr &gt; ChiSq</th>
</tr>
</thead>
<tbody>
<tr>
<td>Likelihood Ratio</td>
<td>2</td>
<td>9.4237</td>
<td>0.0090</td>
</tr>
</tbody>
</table>

### Fit Statistics

- \(-2\) Log Likelihood: 114.39646
- AIC (smaller is better): 120.39646
- AICC (smaller is better): 120.64646
- SBC (smaller is better): 128.21197

The parameter estimates of the selected model are shown in Output 11.1.5. Notice that the effects are listed in the “Parameter Estimates” table in the order in which they were specified in the MODEL statement and not in the order in which they were added to the model.

**Output 11.1.5**  Parameter Estimates

<table>
<thead>
<tr>
<th>Parameter Estimates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>Intercept</td>
</tr>
<tr>
<td>(x2)</td>
</tr>
<tr>
<td>(x8)</td>
</tr>
</tbody>
</table>

You can construct the prediction equation for this model from the parameter estimates as follows. The estimated linear predictor for an observation is

\[
\hat{\eta} = 0.8584 - 0.2502 \times x_2 + 1.7840 \times x_8
\]

The predicted probability that variable \(y\) takes the value 0 is

\[
\hat{Pr}(Y = 0) = \frac{1}{1 + \exp(-\hat{\eta})}
\]
The coefficient panel in Figure 11.1.6 enables you to visualize the selection process. In this plot, standardized coefficients of all the effects that are selected at some step of the stepwise method are plotted as a function of the step number. This enables you to assess the relative importance of the effects that are selected at any step of the selection process and to know when effects entered the model. The lower plot in the panel shows how the criterion that is used to choose the selected model changes as effects enter or leave the model.

**Output 11.1.6  Coefficient Progression**

The criterion panel in Figure 11.1.7 provides a graphical view of the progression of the fit criteria as the selection process evolves. Notice at Step 0 that the SBC criterion seems to be at its minimum. Because the stop horizon value is 3 (see Output 11.1.1), three more steps are taken in case Step 0 is a local optimum. In this example, the global optimum is found two steps later.
Example 11.2: Modeling Binomial Data

If \( Y_1, \ldots, Y_n \) are independent binary (Bernoulli) random variables with common success probability \( \pi \), then their sum is a binomial random variable. In other words, a binomial random variable with parameters \( n \) and \( \pi \) can be generated as the sum of \( n \) Bernoulli(\( \pi \)) random experiments. The LOGSELECT procedure uses a special syntax, the events/trials syntax, to express data in binomial form.

Consider the following data, taken from Cox and Snell (1989, pp. 10–11), of the number of ingots not ready for rolling (\( r \)) out of \( n \) tested for a number of combinations of heating time and soaking time. If each test is carried out independently and if for a particular combination of heating and soaking time there is a constant probability that the tested ingot is not ready for rolling, then the random variable \( r \) follows a binomial(\( n, \pi \)) distribution, where the success probability \( \pi \) is a function of heating and soaking time.

```latex
\begin{verbatim}
data Ingots;
  input Heat Soak r n @@;
  Obsnum= _n_;
  datalines;
  7 1.0 0 10 14 1.0 0 31 27 1.0 1 56 51 1.0 3 13
  7 1.7 0 17 14 1.7 0 43 27 1.7 4 44 51 1.7 0 1
\end{verbatim}
```

Output 11.1.7  Criterion Panel

**Fit Criteria for \( y \)**

- AIC
- AICC
- SBC
data mycas.Ingots;
  set Ingots;
run;

The following statements show the use of the events/trials syntax to model the binomial response. The events variable in this situation is \( r \) (the number of ingots not ready for rolling), and the trials variable is \( n \) (the number of ingots that are tested). The dependency of the probability of not being ready for rolling is modeled as a function of heating time, soaking time, and their interaction. The ASSOCIATION option displays ordinal measures of association between the observed responses and the predicted probabilities. The CTABLE option saves statistics to the mycas.Roc data table that are used to evaluate the predictive power of the model. The OUTPUT statement stores the linear predictors and the predicted probabilities in the mycas.Out data table along with the other variables from the input data table.

```
proc logselect data=mycas.Ingots association ctable(out=mycas.Roc nocounts tpf fpf);
  model r/n = Heat Soak Heat*Soak;
  output out=mycas.Out xbeta predicted=Pred copyvars=(Heat Soak);
run;
```

The “Model Information” table shows that the data are modeled as binomially distributed with a logit link function (Output 11.2.1). This is the default link function in the LOGSELECT procedure for binary and binomial data. The procedure estimates the parameters of the model by a Newton-Raphson algorithm.

**Output 11.2.1** Model Information and Number of Observations

The LOGSELECT Procedure

<table>
<thead>
<tr>
<th>Model Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source</td>
</tr>
<tr>
<td>Response Variable (Events)</td>
</tr>
<tr>
<td>Response Variable (Trials)</td>
</tr>
<tr>
<td>Distribution</td>
</tr>
<tr>
<td>Link Function</td>
</tr>
<tr>
<td>Optimization Technique</td>
</tr>
</tbody>
</table>

| Number of Observations Read | 19 |
| Number of Observations Used | 19 |

<table>
<thead>
<tr>
<th>Response Profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ordered Value</td>
</tr>
<tr>
<td>1 Event</td>
</tr>
<tr>
<td>2 Nonevent</td>
</tr>
</tbody>
</table>

The second table in Output 11.2.1 shows that all 19 observations in the input data table were used in the analysis and the “Response Profile” table shows that the total number of events and nonevents equals 12 and 387, respectively.

**Output 11.2.2** displays the convergence status table for this run. The LOGSELECT procedure satisfies the GCONV= convergence criterion.
Output 11.2.2 Convergence Status

Convergence criterion (GCONV=1E-8) satisfied.

Output 11.2.3 displays the “Dimensions” table for the model. The design matrix of the model (the $X$ matrix) has four columns, which correspond to the intercept, the Heat effect, the Soak effect, and the interaction of the Heat and Soak effects. The model is nonsingular because the rank of the crossproducts matrix equals the number of columns in $X$. All parameters are estimable, and all participate in the optimization.

Output 11.2.3 Dimensions in Binomial Logistic Regression

<table>
<thead>
<tr>
<th>Dimensions</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Columns in Design</td>
<td>4</td>
</tr>
<tr>
<td>Number of Effects</td>
<td>4</td>
</tr>
<tr>
<td>Max Effect Columns</td>
<td>1</td>
</tr>
<tr>
<td>Rank of Design</td>
<td>4</td>
</tr>
<tr>
<td>Parameters in Optimization</td>
<td>4</td>
</tr>
</tbody>
</table>

Output 11.2.4 displays the “Fit Statistics” table for this run. Evaluated at the converged estimates, $-2$ times the value of the log-likelihood function equals 27.9569. Further fit statistics are also displayed, all of them in “smaller is better” form. The AIC, AICC, and SBC criteria are used to compare non-nested models and to penalize the model fit for the number of observations and parameters. You can use the $-2$ log-likelihood value to compare nested models by way of a likelihood ratio test.

Output 11.2.4 Fit Statistics

<table>
<thead>
<tr>
<th>Fit Statistics</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>-2 Log Likelihood</td>
<td>27.9569</td>
</tr>
<tr>
<td>AIC (smaller is better)</td>
<td>35.95689</td>
</tr>
<tr>
<td>AICC (smaller is better)</td>
<td>38.81403</td>
</tr>
<tr>
<td>SBC (smaller is better)</td>
<td>39.73464</td>
</tr>
</tbody>
</table>

Output 11.2.5 shows the test of the global hypothesis that the effects jointly do not affect the probability of ingot readiness. You can obtain the chi-square test statistic by comparing the $-2$ log-likelihood value of the model with covariates to the value in the intercept-only model. The test is significant, with a $p$-value of 0.0082. One or more of the effects in the model have a significant impact on the probability of ingot readiness.

Output 11.2.5 Null Test

<table>
<thead>
<tr>
<th>Testing Global Null Hypothesis: BETA=0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test</td>
</tr>
<tr>
<td>Likelihood Ratio</td>
</tr>
</tbody>
</table>

The “Parameter Estimates” table in Output 11.2.6 displays the estimates and standard errors of the model effects.
Output 11.2.6 Parameter Estimates

<table>
<thead>
<tr>
<th>Parameter</th>
<th>DF</th>
<th>Estimate</th>
<th>Standard Error</th>
<th>Chi-Square</th>
<th>Pr &gt; ChiSq</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>1</td>
<td>-5.990191</td>
<td>1.666622</td>
<td>12.9183</td>
<td>0.0003</td>
</tr>
<tr>
<td>Heat</td>
<td>1</td>
<td>0.096339</td>
<td>0.047067</td>
<td>4.1896</td>
<td>0.0407</td>
</tr>
<tr>
<td>Soak</td>
<td>1</td>
<td>0.299574</td>
<td>0.755068</td>
<td>0.1574</td>
<td>0.6916</td>
</tr>
<tr>
<td>Heat * Soak</td>
<td>1</td>
<td>-0.008840</td>
<td>0.025319</td>
<td>0.1219</td>
<td>0.7270</td>
</tr>
</tbody>
</table>

Output 11.2.7 displays the “Association Statistics” table, which is produced when you specify the ASSOCIATION option in the PROC LOGSELECT statement. The table contains four measures of association for assessing the predictive ability of a model. For more information, see the section “Association Statistics” on page 588.

Output 11.2.7 Association of Observed Responses and Predicted Probabilities

<table>
<thead>
<tr>
<th>Association of Predicted Probabilities and Observed Responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concordance Index (AUC)</td>
</tr>
<tr>
<td>0.7706</td>
</tr>
<tr>
<td>Somers' D</td>
</tr>
<tr>
<td>0.5411</td>
</tr>
<tr>
<td>Gamma</td>
</tr>
<tr>
<td>0.5858</td>
</tr>
<tr>
<td>Tau-a</td>
</tr>
<tr>
<td>0.0326</td>
</tr>
<tr>
<td>Pairs</td>
</tr>
<tr>
<td>4500</td>
</tr>
<tr>
<td>Percent Concordant</td>
</tr>
<tr>
<td>73.2444</td>
</tr>
<tr>
<td>Percent Discordant</td>
</tr>
<tr>
<td>19.1333</td>
</tr>
<tr>
<td>Percent Tied</td>
</tr>
<tr>
<td>7.6222</td>
</tr>
</tbody>
</table>

You can construct the prediction equation of the model from the “Parameter Estimates” table. For example, an observation with Heat equal to 14 and Soak equal to 1.7 has the linear predictor

\[ \hat{\eta} = -5.9902 + 0.09634 \times 14 + 0.2996 \times 1.7 - 0.00884 \times 14 \times 1.7 = -4.3425 \]

The probability that an ingot with these characteristics is not ready for rolling is

\[ \hat{\pi} = \frac{1}{1 + \exp\{-(-4.3425)\}} = 0.01284 \]

PROC LOGSELECT computes these linear predictors and probabilities and stores them in the mycas.Out data set. The following PROC PRINT statement produces Output 11.2.8, which shows the probability that an ingot with Heat equal to 14 and Soak equal to 1.7 is not ready for rolling:

```
proc print data=mycas.Out;
  where Heat=14 & Soak=1.7;
run;
```

Output 11.2.8 Predicted Probability for Heat=14 and Soak=1.7

<table>
<thead>
<tr>
<th>Obs</th>
<th>Pred <em>XBETA</em></th>
<th>Heat</th>
<th>Soak</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.012836</td>
<td>14</td>
<td>1.7</td>
</tr>
</tbody>
</table>
The **CTABLE** option computes statistics for binary response models (on the basis of classifying observations according to whether their predicted probabilities exceed certain values) and stores the results in the `mycas.Roc` data set. For more information, see the section “Classification Table and ROC Curves” on page 586. You can use this data set to display the ROC curve by using the SGPLOT procedure as follows:

```plaintext
ods graphics on;
proc sgplot data=mycas.Roc aspect=1 noautolegend;
  title 'ROC Curve';
  xaxis values=(0 to 1 by 0.25) grid offsetmin=.05 offsetmax=.05;
  yaxis values=(0 to 1 by 0.25) grid offsetmin=.05 offsetmax=.05;
  lineparm x=0 y=0 slope=1 / lineattrs=(color=ligr);
  series x=FPF y=TPF;
  inset 'Area under the curve=0.7706' / position=bottomright;
run;
```

**Output 11.2.9** Receiver Operating Characteristics Curve

Binomial data are a form of grouped binary data in which “successes” in the underlying Bernoulli trials are totaled. You can thus unwind data for which you use the events/trials syntax and fit these data by using techniques for binary data.

The following DATA step expands the `ingots` data set, which contains 12 events in 387 trials, into a data table that has 387 observations:
data mycas.Ingots_binary;
    set Ingots;
    do i=1 to n;
        if i <= r then y=1; else y = 0;
        output;
    end;
run;

The following statements fit the model that contains the Heat effect, the Soak effect, and their interaction to the data table. The event='1' response-variable option in the MODEL statement ensures that the LOGSELECT procedure models the probability that the variable \( y \) takes the value 1.

```
proc logselect data=mycas.Ingots_binary;
    model y(event='1') = Heat Soak Heat*Soak;
run;
```

Output 11.2.10 displays the “Model Information,” “Number of Observations,” and “Response Profile” tables. The data are now modeled as binary (Bernoulli distributed) with a logit link function. The “Response Profile” table shows that the binary response breaks down into 375 observations where \( y=0 \) and 12 observations where \( y=1 \).

**Output 11.2.10** Model Information in Binary Model

<table>
<thead>
<tr>
<th>The LOGSELECT Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Model Information</strong></td>
</tr>
<tr>
<td>Data Source</td>
</tr>
<tr>
<td>Response Variable</td>
</tr>
<tr>
<td>Distribution</td>
</tr>
<tr>
<td>Link Function</td>
</tr>
<tr>
<td>Optimization Technique</td>
</tr>
<tr>
<td>Number of Observations Read</td>
</tr>
<tr>
<td>Number of Observations Used</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response Profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ordered Value</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
</tbody>
</table>

Probability modeled is \( y = 1 \).

Output 11.2.11 displays the result of the test of the global null hypothesis and the parameter estimates. These results match those in Output 11.2.5 and Output 11.2.6.

**Output 11.2.11** Null Test and Parameter Estimates

<table>
<thead>
<tr>
<th>Testing Global Null Hypothesis: BETA=0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test</td>
</tr>
<tr>
<td>Likelihood Ratio</td>
</tr>
</tbody>
</table>
Output 11.2.11 continued

<table>
<thead>
<tr>
<th>Parameter</th>
<th>DF</th>
<th>Estimate</th>
<th>Standard Error</th>
<th>Chi-Square</th>
<th>Pr &gt; ChiSq</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>1</td>
<td>-5.990191</td>
<td>1.666622</td>
<td>12.9183</td>
<td>0.0003</td>
</tr>
<tr>
<td>Heat</td>
<td>1</td>
<td>0.096339</td>
<td>0.047067</td>
<td>4.1896</td>
<td>0.0407</td>
</tr>
<tr>
<td>Soak</td>
<td>1</td>
<td>0.299574</td>
<td>0.755068</td>
<td>0.1574</td>
<td>0.6916</td>
</tr>
<tr>
<td>Heat * Soak</td>
<td>1</td>
<td>-0.008840</td>
<td>0.025319</td>
<td>0.1219</td>
<td>0.7270</td>
</tr>
</tbody>
</table>

Example 11.3: Partitioning Data

The Sashelp.JunkMail data set comes from a study that classifies whether an email is junk email (coded as 1) or not (coded as 0). The data were collected by Hewlett-Packard Labs and donated by George Forman. The data set, which is specified in the following DATA step, contains 4,601 observations, with 2 binary variables and 57 continuous explanatory variables. The response variable, Class, is a binary indicator of whether an email is considered spam or not. The partitioning variable, Test, is a binary indicator that is used to divide the data into training and testing sets. The 57 explanatory variables are continuous variables that represent frequencies of some common words and characters and lengths of uninterrupted sequences of capital letters in emails.

data mycas.JunkMail;
  set Sashelp.JunkMail;
run;

In the following program, the PARTITION statement divides the data into two parts. The training data have a Test value of 0 and contain about two-thirds of the data; the rest of the data are used to evaluate the fit. A forward selection method selects the best model based on the training observations.

proc logselect data=mycas.JunkMail;
  model Class(event='1')=Make Address All _3d Our Over Remove Internet Order Mail Receive Will People Report Addresses Free Business Email You Credit Your Font _000 Money HP HPL George _650 Lab Labs Telnet _857 Data _415 _85 Technology _1999 Parts FM Direct CS Meeting Original Project RE Edu Table Conference Semicolon Paren Bracket Exclamation Dollar Pound CapAvg CapLong CapTotal;
  partition rolevar=Test(train='0' test='1');
  selection method=forward;
run;

Selected results from the analysis are shown in Output 11.3.1 and Output 11.3.2.

The “Number of Observations” and “Response Profile” tables in Output 11.3.1 are divided into training and testing columns.
Output 11.3.1 Partitioned Counts

The LOGSELECT Procedure

<table>
<thead>
<tr>
<th>Description</th>
<th>Total</th>
<th>Training</th>
<th>Testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Observations Read</td>
<td>4601</td>
<td>3065</td>
<td>1536</td>
</tr>
<tr>
<td>Number of Observations Used</td>
<td>4601</td>
<td>3065</td>
<td>1536</td>
</tr>
</tbody>
</table>

Response Profile

<table>
<thead>
<tr>
<th>Ordered Value</th>
<th>Class</th>
<th>Total Frequency</th>
<th>Training</th>
<th>Testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>2788</td>
<td>1847</td>
<td>941</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1813</td>
<td>1218</td>
<td>595</td>
</tr>
</tbody>
</table>

Probability modeled is Class = 1.

The likelihood-based fit statistics for the selected model are displayed in the “Fit Statistics” table in Output 11.3.2, which has columns for the training and testing subsets.

Output 11.3.2 Partitioned Fit Statistics

<table>
<thead>
<tr>
<th>Description</th>
<th>Training</th>
<th>Testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>-2 Log Likelihood</td>
<td>1242.59491</td>
<td>823.68742</td>
</tr>
<tr>
<td>AIC (smaller is better)</td>
<td>1292.59491</td>
<td>873.68742</td>
</tr>
<tr>
<td>AICC (smaller is better)</td>
<td>1293.02268</td>
<td>874.54835</td>
</tr>
<tr>
<td>SBC (smaller is better)</td>
<td>1443.28998</td>
<td>1007.11085</td>
</tr>
<tr>
<td>Average Square Error</td>
<td>0.05659</td>
<td>0.06353</td>
</tr>
<tr>
<td>-2 Log L (Intercept-only)</td>
<td>4118.98701</td>
<td>2050.73514</td>
</tr>
<tr>
<td>R-Square</td>
<td>0.60877</td>
<td>0.55016</td>
</tr>
<tr>
<td>Max-rescaled R-Square</td>
<td>0.82359</td>
<td>0.74661</td>
</tr>
<tr>
<td>McFadden's R-Square</td>
<td>0.69833</td>
<td>0.59835</td>
</tr>
<tr>
<td>Misclassification Rate</td>
<td>0.07471</td>
<td>0.07813</td>
</tr>
<tr>
<td>Difference of Means</td>
<td>0.75122</td>
<td>0.73431</td>
</tr>
</tbody>
</table>

These statistics are computed for both the training and testing data. The statistics include the likelihood-based R-square statistics as well as several prediction-based statistics that are described in the section “Model Fit and Assessment Statistics” on page 585. The ASE, the misclassification rate, and the difference of means should be very similar between the two groups when the training data are representative of the testing data; for this model, the values of these statistics seem similar between the two disjoint subsets.

If you want to display all the fit statistics without partitioning your data table, you can specify the PARTFIT option as follows:

```plaintext
proc logselect data=mycas.JunkMail(where Test=0) partfit;
   model Class(event='1')= Our Over Remove Internet Order Will
             Free Business You Your Font _000 Money HP George Parts
             Meeting RE Edu Semicolon Exclamation Dollar CapAvg
             CapLong;
run;
```

The resulting “Fit Statistics” table is identical to the Training column in Output 11.3.2.
Example 11.4: Ordinal Logistic Regression

Consider a study of the effects of various cheese additives on taste. Researchers tested four cheese additives and obtained 52 response ratings for each additive. Each response was measured on a scale of nine categories ranging from strong dislike (1) to excellent taste (9). The data, given in McCullagh and Nelder (1989, p. 175) in the form of a two-way frequency table of additive by rating, are saved in the data table mycas.Cheese by using the following program. The variable \( y \) contains the response rating. The variable Additive specifies the cheese additive (1, 2, 3, or 4). The variable freq gives the frequency with which each additive received each rating.

```sas
data mycas.Cheese;
  do Additive = 1 to 4;
    do y = 1 to 9;
      input freq @@;
      output;
    end;
  end;
  label y='Taste Rating';
datalines;
0 0 1 7 8 8 19 8 1
6 9 12 11 7 6 1 0 0
1 1 6 8 23 7 5 1 0
0 0 0 1 3 7 14 16 11
;
```

The response variable \( y \) is ordinally scaled. A cumulative logit model is used to investigate the effects of the cheese additives on taste. The following statements invoke PROC LOGSELECT to fit this model with \( y \) as the response variable and three indicator variables as explanatory variables, with the fourth additive as the reference level. With this parameterization, each Additive parameter compares an additive to the fourth additive.

```sas
proc logselect data=mycas.Cheese;
  freq freq;
  class Additive(ref='4') / param=ref ;
  model y=Additive;
run;
```

Results from the logistic analysis are shown in Output 11.4.1 through Output 11.4.3.

The “Response Profile” table in Output 11.4.1 shows that the strong dislike (\( y=1 \)) end of the rating scale is associated with lower OrderedValue values in the “Response Profile” table; hence the probability of disliking the additives is modeled.
Output 11.4.1 Proportional Odds Model Regression Analysis

The LOGSELECT Procedure

<table>
<thead>
<tr>
<th>Model Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source</td>
</tr>
<tr>
<td>Response Variable</td>
</tr>
<tr>
<td>Number of Response Levels</td>
</tr>
<tr>
<td>Frequency Variable</td>
</tr>
<tr>
<td>Distribution</td>
</tr>
<tr>
<td>Link Type</td>
</tr>
<tr>
<td>Link Function</td>
</tr>
<tr>
<td>Optimization Technique</td>
</tr>
</tbody>
</table>

<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Observations Read: 36</td>
</tr>
<tr>
<td>Number of Observations Used: 28</td>
</tr>
<tr>
<td>Sum of Frequencies Read: 208</td>
</tr>
<tr>
<td>Sum of Frequencies Used: 208</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response Profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ordered Value</td>
</tr>
<tr>
<td>Total Frequency</td>
</tr>
<tr>
<td>y</td>
</tr>
<tr>
<td>1 7</td>
</tr>
<tr>
<td>2 10</td>
</tr>
<tr>
<td>3 19</td>
</tr>
<tr>
<td>4 27</td>
</tr>
<tr>
<td>5 41</td>
</tr>
<tr>
<td>6 28</td>
</tr>
<tr>
<td>7 39</td>
</tr>
<tr>
<td>8 25</td>
</tr>
<tr>
<td>9 12</td>
</tr>
</tbody>
</table>

Probabilities modeled are cumulated over the lower Ordered Values.

<table>
<thead>
<tr>
<th>Class Level Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class Levels Values</td>
</tr>
<tr>
<td>Additive 4 1 2 3 4</td>
</tr>
</tbody>
</table>

Output 11.4.2 Proportional Odds Model Regression Analysis

Convergence criterion (GCONV=1E-8) satisfied.

<table>
<thead>
<tr>
<th>Dimensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Columns in Design: 11</td>
</tr>
<tr>
<td>Number of Effects: 2</td>
</tr>
<tr>
<td>Max Effect Columns: 8</td>
</tr>
<tr>
<td>Rank of Design: 11</td>
</tr>
<tr>
<td>Parameters in Optimization: 11</td>
</tr>
</tbody>
</table>
Output 11.4.2  continued

<table>
<thead>
<tr>
<th></th>
<th>DF</th>
<th>Chi-Square</th>
<th>Pr &gt; ChiSq</th>
</tr>
</thead>
<tbody>
<tr>
<td>Likelihood Ratio</td>
<td>3</td>
<td>148.4539</td>
<td>&lt;.0001</td>
</tr>
</tbody>
</table>

Fit Statistics

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>-2 Log Likelihood</td>
<td>711.34790</td>
</tr>
<tr>
<td>AIC (smaller is better)</td>
<td>733.34790</td>
</tr>
<tr>
<td>AICC (smaller is better)</td>
<td>734.69484</td>
</tr>
<tr>
<td>SBC (smaller is better)</td>
<td>770.06082</td>
</tr>
</tbody>
</table>

The positive value (1.6128) for the parameter estimate for Additive=1 in Output 11.4.3 indicates a tendency toward the lower-numbered categories of the first cheese additive relative to the fourth. In other words, the fourth additive tastes better than the first additive. Similarly, the second and third additives are both less favorable than the fourth additive. The relative magnitudes of these slope estimates imply the preference ordering: fourth, first, third, second.

Output 11.4.3  Proportional Odds Model Regression Analysis

<table>
<thead>
<tr>
<th>Parameter</th>
<th>y</th>
<th>DF</th>
<th>Estimate</th>
<th>Standard Error</th>
<th>Chi-Square</th>
<th>Pr &gt; ChiSq</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept  1</td>
<td>1</td>
<td>-7.080166</td>
<td>0.564010</td>
<td>157.5844</td>
<td>&lt;.0001</td>
<td></td>
</tr>
<tr>
<td>Intercept  2</td>
<td>1</td>
<td>-6.024980</td>
<td>0.476431</td>
<td>159.9230</td>
<td>&lt;.0001</td>
<td></td>
</tr>
<tr>
<td>Intercept  3</td>
<td>1</td>
<td>-4.925416</td>
<td>0.425651</td>
<td>133.8992</td>
<td>&lt;.0001</td>
<td></td>
</tr>
<tr>
<td>Intercept  4</td>
<td>1</td>
<td>-3.856801</td>
<td>0.425651</td>
<td>98.7968</td>
<td>&lt;.0001</td>
<td></td>
</tr>
<tr>
<td>Intercept  5</td>
<td>1</td>
<td>-2.520552</td>
<td>0.345268</td>
<td>53.2940</td>
<td>&lt;.0001</td>
<td></td>
</tr>
<tr>
<td>Intercept  6</td>
<td>1</td>
<td>-1.568538</td>
<td>0.312208</td>
<td>25.2408</td>
<td>&lt;.0001</td>
<td></td>
</tr>
<tr>
<td>Intercept  7</td>
<td>1</td>
<td>0.066675</td>
<td>0.273819</td>
<td>0.0596</td>
<td>0.8071</td>
<td></td>
</tr>
<tr>
<td>Intercept  8</td>
<td>1</td>
<td>1.492974</td>
<td>0.335696</td>
<td>19.7794</td>
<td>&lt;.0001</td>
<td></td>
</tr>
<tr>
<td>Additive 1</td>
<td>1</td>
<td>1.612791</td>
<td>0.380544</td>
<td>17.9617</td>
<td>&lt;.0001</td>
<td></td>
</tr>
<tr>
<td>Additive 2</td>
<td>1</td>
<td>4.964640</td>
<td>0.476721</td>
<td>108.4546</td>
<td>&lt;.0001</td>
<td></td>
</tr>
<tr>
<td>Additive 3</td>
<td>1</td>
<td>3.322683</td>
<td>0.421830</td>
<td>62.0444</td>
<td>&lt;.0001</td>
<td></td>
</tr>
</tbody>
</table>

References


Overview: MBC Procedure

The MBC procedure fits mixtures of multivariate Gaussian and uniform distributions to achieve unsupervised and semisupervised clustering of data. It treats the cluster memberships as missing data and uses the expectation-maximization (EM) algorithm to maximize the likelihood. The procedure can produce output data tables that contain cluster membership weights and component log likelihoods. PROC MBC can also store the fitted model for scoring subsequent input data.

PROC MBC assumes that the data can be described by a mixture of multivariate Gaussian components and an optional noise component. The procedure also assumes that the data are continuous, and it does not support CLASS variables.

Motivation for Model-Based Clustering

The traditional goal of clustering has been to identify groups of observations that are similar by some measurement. Standard clustering methods achieve this primarily through heuristic methods, by using a measurement such as the distance to a center or a boundary. These methods appeal to intuition, but they cannot answer fundamental questions about the number of clusters, the suitability of certain cluster structures, or the handling of outliers. In addition, standard methods do not have well-defined statistical properties and so do not support formal inference.

In contrast, model-based clustering (MBC) models the observations by using a mixture of specific distributions. In this framework, the qualities of these clusters and cluster membership are parameters to be estimated by formal statistical methods.

The MBC procedure implements model-based clustering by using mixtures of multivariate Gaussian distributions. PROC MBC allows for a noise component and automatic model selection through the use of information criteria. In addition, it provides posterior scoring of new input data.

PROC MBC Compared with Other Procedures

SAS/STAT and SAS Viya offer several other clustering and mixture-modeling procedures.

PROC MBC is based on the model likelihood, whereas the clustering methods in the FASTCLUS, CLUSTER, KCLUS, MODECLUS, and FASTKNN procedures use a distance-based clustering criterion. The likelihood-based approach allows direct comparison of models through the use of information criteria.

This likelihood-based approach means that the MBC procedure can produce “soft” clusterings, which are based on posterior probabilities. These clusterings indicate relative strengths of affinity that each observation has for each cluster in the model. In contrast, the other clustering procedures produce “hard” clusterings, which associate each observation with a single cluster.

Like PROC MBC, the FMM and HPFMM procedures use a likelihood-based approach. PROC FMM and PROC HPFMM can fit several different univariate finite mixture models, but they cannot fit multivariate models. PROC MBC uses the expectation-maximization (EM) technique (Dempster, Laird, and Rubin 1977) to find parameter estimates, whereas the FMM and HPFMM procedures use traditional maximum likelihood and Bayesian techniques.
Using CAS Sessions and CAS Engine Librefs

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:

```
proc options option=(CASHOST CASPORT);
run;
```

In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:

```
cas mysess;
libname mycas cas sessref=mysess;
```

The CAS statement creates the CAS session named `mysess`, and the LIBNAME statement creates the `mycas` CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the CASHOST and CASPORT of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the `mysess` session, you can terminate it by using the TERMINATE option in the CAS statement as follows:

```
cas mysess terminate;
```

For more information about the CAS and LIBNAME statements, see the section “Introduction to Shared Concepts” on page 10 in Chapter 3, “Shared Concepts.”

---

Getting Started: MBC Procedure

**Old Faithful Data**

This example uses the Old Faithful data on geyser eruptions (Cook and Weisberg 1982) to illustrate a mixture of multivariate Gaussian components. This data set contains time before eruption and duration of eruption for Old Faithful geyser in Yellowstone National Park in the United States. The following DATA step creates the data table `mycas.getStarted`, which consists of 272 observations on the duration of each eruption and the wait time until the next eruption. Both duration and wait time are measured in minutes.

```
data mycas.getStarted;
  input Duration Wait @@;
datalines;
```
Chapter 12: The MBC Procedure

1.950 51 4.350 85 1.833 54 3.917 84 4.200 78 1.750 47 4.700 83 2.167 52

... more lines ...

1.983 43 2.250 60 4.750 75 4.117 81 2.150 46 4.417 90 1.817 46 4.467 74
;

These statements assume that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

The following statements produce a scatter plot of the duration and wait times:

    proc sgplot data=mycas.getStarted;
    scatter x=duration y=wait / markerattrs=(symbol=circlefilled size=4);
    run; quit;

It is clear from the scatter plot in Figure 12.1 that there are two clusters of eruptions. The first consists of short, frequent eruptions, and the second consists of long, infrequent eruptions. However, the shape of these clusters is not spherical. The MBC procedure can help you explore the shapes.

Figure 12.1 Scatter Plot of Old Faithful Data
The following statements fit several different mixture models to these data:

```plaintext
proc mbc data=mycas.getStarted nclusters=(1 to 5)
   noise=(YES NO)
   covstruct=(EEV EII VVV)
   seed=1418410433;
   var duration wait;
   output out=mycas.scores copyvars=(duration wait);
run;
```

The NCLUSTERS= option list fits separate models by using one to five Gaussian clusters. The NOISE= option list fits these models with and without noise. The COVSTRUCT= option list fits each of these models by using three different covariance structures. Finally, the SEED= option specifies a starting random seed to generate initial cluster weights. In total, PROC MBC fits 30 different models, corresponding to the unique combinations of each element in the NCLUSTERS=, NOISE=, and COVSTRUCT= option lists.

The VAR statement identifies the two variables, Duration and Wait, that define the points to be clustered.

The OUTPUT statement specifies a CAS data table to contain the posterior cluster weights for each observation. The COPYVARS= option includes the two analysis variables, Duration and Wait, in the output data table.

After fitting all 30 models, PROC MBC chooses the model that has the best Bayesian information criterion (BIC) value. BIC is a quantity composed of the log likelihood and a penalty term. Smaller BIC values indicate models with better fit. The “Fit Summary” table in Figure 12.2 shows model information for the 10 models that have the best (smallest) BIC values, sorted so that the best model appears first. The selected model has two nonspherical clusters, as suggested by the plot in Figure 12.1.

![Figure 12.2 Summary of Fit Statistics](image)

**The MBC Procedure**

<table>
<thead>
<tr>
<th>Covariance Structure</th>
<th>Number of Clusters</th>
<th>Noise Component</th>
<th>Number of Parameters</th>
<th>-2 Log L</th>
<th>AIC</th>
<th>AICC</th>
<th>BIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>VVV</td>
<td>2</td>
<td>N</td>
<td>11</td>
<td>2260.52818 2282.52818 2283.54356 2322.19200</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VVV</td>
<td>2</td>
<td>Y</td>
<td>13</td>
<td>2260.61415 2286.61415 2288.02500 2333.48957</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VVV</td>
<td>3</td>
<td>N</td>
<td>17</td>
<td>2238.48109 2272.48109 2274.89054 2333.77973</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EEV</td>
<td>2</td>
<td>Y</td>
<td>11</td>
<td>2278.66416 2300.66416 2301.67955 2340.32798</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VVV</td>
<td>3</td>
<td>Y</td>
<td>19</td>
<td>2238.48207 2276.48207 2279.49794 2344.99231</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VVV</td>
<td>4</td>
<td>N</td>
<td>23</td>
<td>2216.08379 2262.08379 2266.53541 2345.01724</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EEV</td>
<td>4</td>
<td>Y</td>
<td>19</td>
<td>2241.44464 2279.44464 2282.46051 2347.95488</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EEV</td>
<td>3</td>
<td>Y</td>
<td>15</td>
<td>2266.00364 2296.00364 2297.87864 2350.09067</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EEV</td>
<td>5</td>
<td>N</td>
<td>21</td>
<td>2239.56249 2281.56249 2285.25849 2357.28433</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VVV</td>
<td>5</td>
<td>N</td>
<td>29</td>
<td>2208.49803 2266.49803 2273.68811 2371.06629</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 12.3 shows the “Model Information” table. This table summarizes the characteristics of the selected model and of the modeling technique that was used.
The “Parameter Estimates” table in Figure 12.4 contains the estimates of the centers of the two clusters and the estimates of the covariance for the multivariate Gaussian that is represented by each cluster.

If you compare the mean estimates to the plot in Figure 12.1, you can see that the estimates correspond well to the data.

Figure 12.1 suggests that one cluster is slightly heavier than the other. The “Mixing Estimates” table in Figure 12.5 supports this impression.

The following statements use the output table mycas.scores to produce a plot of the maximum posterior weights for each observation:
data scores;
   set mycas.scores;
   maxwt = max(of next:);
run;

proc sgplot data=scores noautolegend;
   scatter x=duration y=wait / markerattrs=(symbol=circlefilled)
      colorresponse=maxwt
colormodel=TwoColorRamp;
   scatter x=duration y=wait / markerattrs=(symbol=circle color=black);
run;
quit;

Figure 12.6 plots the observations with shading to indicate the strength of the affinity that each point has for the cluster that it is more strongly associated with. Only one point has light shading, indicating that it is in a region where the larger posterior cluster weight is not as large as it is in other regions. This indicates a region where the posterior weights for the two clusters are closer in value.

Figure 12.6 Old Faithful Data and Selected Model
### Syntax: MBC Procedure

The following statements are available in the MBC procedure:

```
PROC MBC <options> ;
   BY variables ;
   VAR variables ;
   INIT variables ;
   OUTPUT OUT= <name> ... <keyword <name>> <options> ;
   DISPLAY <options> ;
   DISPLAYOUT <options> ;
   STORE <options> ;
```

The PROC MBC statement is required.

### PROC MBC Statement

```
PROC MBC <options> ;
```

The PROC MBC statement invokes the procedure. Table 12.1 summarizes the available options in the PROC MBC statement by function. They are then described fully in alphabetical order.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Basic Options</strong></td>
<td></td>
</tr>
<tr>
<td>DATA=</td>
<td>Specifies the CAS input data table</td>
</tr>
<tr>
<td><strong>Options Related to Model Fitting</strong></td>
<td></td>
</tr>
<tr>
<td>COVSTRUCT=</td>
<td>Specifies the covariance structure</td>
</tr>
<tr>
<td>CRITERION=</td>
<td>Specifies the fit statistic to use for model selection</td>
</tr>
<tr>
<td>EMCRITERION=</td>
<td>Tunes the expectation-maximization (EM) convergence criterion</td>
</tr>
<tr>
<td>INIT=</td>
<td>Specifies the initialization method</td>
</tr>
<tr>
<td>ITHIST=</td>
<td>Displays the iteration history</td>
</tr>
<tr>
<td>MAXITER=</td>
<td>Specifies the maximum number of EM iterations</td>
</tr>
<tr>
<td>NCLUSTERS=</td>
<td>Specifies the number of Gaussian components</td>
</tr>
<tr>
<td>NOISE=</td>
<td>Specifies the presence of noise</td>
</tr>
<tr>
<td>SEED=</td>
<td>Specifies the pseudorandom number seed</td>
</tr>
<tr>
<td>SINGPARM=</td>
<td>Tunes the parameter singularity criterion</td>
</tr>
<tr>
<td>SINGULAR=</td>
<td>Tunes the singularity criterion</td>
</tr>
<tr>
<td>TECHNIQUE=</td>
<td>Specifies the EM technique</td>
</tr>
<tr>
<td>TOPMODELS=</td>
<td>Specifies the number of models to display in summary</td>
</tr>
</tbody>
</table>

The following list describes the options available in the PROC MBC statement:
COVSTRUCT=EEE | EEI | EEV | EII | EVI | EVV | VII | VVI | VVV | (name < name ... name>)
specifies the covariance structure for the Gaussian components. By default, COVSTRUCT=VVV. For a complete discussion of the different covariance structures, see the section “Covariance Structures” on page 635. If you specify a list of covariance structures, PROC MBC fits a separate model for each value in the list and for each value of the NCLUSTERS= and NOISE= options. The procedure identifies the model that has the best fit, where the fit is computed according to the value of the CRITERION= option.

CRITERION=AIC | AICC | BIC | LOGL
specifies the model fit criterion to use in model selection. You can specify the following values:

  AIC specifies Akaike’s information criterion.
  AICC specifies Akaike’s information criterion (corrected).
  BIC specifies the Bayesian information criterion.
  LOGL specifies the log likelihood.

By default, CRITERION=BIC.

DATA=NAME | LIBREF.data-table
names the input data table for PROC MBC to use. The default is the most recently created data table. NAME | LIBREF.data-table is a two-level name, where

  NAME
refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to the data, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about NAME | LIBREF, see the section “Using CAS Sessions and CAS Engine Librefs” on page 621.

data-table specifies the name of the input data table.

EMCRITERION=number
specifies the convergence criterion for the expectation-maximization (EM) technique. The EM process terminates when \( |f^{(k+1)} - f^{(k)}| / |f^{(k)}| \leq number \), where \( f^{(k)} \) is the model log likelihood after the \( k \)th iteration. The value of number must be greater than or equal to zero. By default, EMCRITERION=1E–5.

INIT=KMEANS | RANDOM
specifies which method to use to initialize cluster membership. This option is ignored if you specify initial weights in the INIT statement. You can specify one of the following values:

  KMEANS uses \( k \)-means clustering to generate starting values. BY-group processing is not available when you specify this method.
  RANDOM specifies the random weight method.

By default, INIT=RANDOM.

If you use the INIT statement to specify initialization variables, the procedure uses those variables for initialization and ignores any INIT= option.
ITHIST <=DETAILS | NONE | SUMMARY>
produces the “Iteration History” table. You can specify one of the following levels:

DETAILS produces a table that contains expanded information about each iteration.
NONE suppresses the production of the table.
SUMMARY produces a table that contains basic information about each iteration.

If you specify the ITHIST option but do not specify one of the levels, the procedure uses the SUMMARY level.

MAXITER=number
specifies the maximum number of expectation-maximization (EM) iterations before termination. The EM process terminates after the specified number of iterations, even if the process has not converged. By default, MAXITER=500.

NCLUSTERS=number | (number < number ... number>) | (number to number < by number>)
specifies the number of Gaussian components in the model. By default, NCLUSTERS=2. If you specify a list of numbers or a range of numbers, PROC MBC fits a separate model for each value in the list or range and for all values of the NOISE= and COVSTRUCT= options. The procedure identifies the model that has the best fit, where the fit is computed according to the value of the CRITERION= option.

If you use the (smallest to largest increment) syntax, the list contains the values (smallest, smallest+increment, smallest+2*increment, ... , smallest+p*increment), where each value in the list is less than or equal to largest. If the resulting list contains any noninteger values or values less than one, the procedure terminates with a syntax error.

NOISE=YES | NO | (value < value ... value>)
specifies whether to include a noise component. By default, NOISE=NO. If you specify a list of values, PROC MBC fits a separate model for each distinct value in the list and for all values of the COVSTRUCT= and NCLUSTERS= options. The procedure identifies the model that has the best fit, where the fit is computed according to the value of the CRITERION= option.

SEED=integer
specifies the pseudorandom number seed for generating random starting weights. Random starting weights are used only when you do not specify starting weight variables in the INIT statement and when you do not specify INIT=RANDOM.

SINGPARM=number
specifies the parameter singularity criterion. If the mixing estimate for a model component is less than or equal to number, it is considered to be zero. In models that have noise components, if the computed noise volume is less than or equal to number, the procedure will not fit the model. The value of number must be greater than zero. By default, SINGPARM=1E–8.

SINGULAR=number
specifies the general singularity criterion to use in matrix operations. The value of number must be greater than or equal to zero. By default, SINGULAR=1E–8.
**BY Statement**

**BY** variables ;

You can specify a **BY** statement in PROC MBC to obtain separate analyses of observations in groups that are defined by the values of the **BY** variables. If you specify more than one **BY** statement, only the last one specified is used. For more information, see the discussion of **BY**-group processing in *SAS Language Reference: Concepts*.

**DISPLAY Statement**

**DISPLAY** <table-list> < / options> ;

The **DISPLAY** statement enables you to specify a list of display tables to display or exclude. This statement is similar to the **ODS SELECT**, **ODS EXCLUDE**, and **ODS TRACE** statements. However, the **DISPLAY** statement can improve performance when a large number of tables could be generated (such as in **BY**-group processing). The procedure processes the **DISPLAY** statement on a CAS server and thus sends only a subset of **ODS** tables to the SAS client. Because **ODS** statements are processed on a SAS client, first all the generated display tables are sent to the client, and then the client creates a subset.

If you use both **DISPLAY** and **ODS** statements together, the **DISPLAY** statement takes precedence over the **ODS** statements. Note that the **ODS EXCLUDE** statement processes tables that are sent to the client after they have been filtered by the **DISPLAY** statement. In some cases, it might appear that the **ODS EXCLUDE** statement is taking precedence because it can further filter the tables. For more information about **ODS**, see *SAS Output Delivery System: Procedures Guide*.

You can specify the **table-list** as a list of table names, paths, partial pathnames, and regular expressions.

The table names that you can specify are listed in the section “**ODS Table Names**” on page 639. A path is a table name that is prefixed with dot-separated grouping information. For example, a SelectionSummary table that a procedure produces during a selection routine might have the path **Bygroup1.Summary.SelectionSummary**. A partial pathname does not include all groups; for example, **SelectionSummary** and **Summary.SelectionSummary** are partial pathnames for **Bygroup1.Summary.SelectionSummary**.

**TECHNIQUE=EM | CEM**

specifies the expectation-maximization (EM) technique to use. You can specify the following values:

- **EM** specifies the traditional EM technique.
- **CEM** specifies the classification EM technique.

By default, **TECHNIQUE=EM**.

**TOPMODELS=number**

specifies the maximum number of models to include in the FitSummary table. The value of number must be an integer greater than or equal to 1. This option is applicable only when at least one of the **NCLUSTERS=**, **NOISE=**, or **COVSTRUCT=** options has multiple values. The FitSummary table displays summary information for number of models or for as many models as were fit, whichever is smaller. By default, **TOPMODELS=10**.
When you specify a table name or partial pathname, all display tables whose paths end in the specified name are selected for display or exclusion. For example, both SelectionSummary and Summary.SelectionSummary select Bygroup1.Summary.SelectionSummary.

A regular expression is enclosed in forward slashes (/). For example, specifying “/tions/” selects all pathnames that contain the substring “tions”; in particular, the Bygroup1.Summary.SelectionSummary table is selected. Specifying “!/tions/” selects all pathnames that do not contain the substring “tions”; in particular, the Bygroup1.Summary.SelectionSummary table is not selected.

You can specify the following options after a slash (/):

**CASESENSITIVE**
- Performs a case-sensitive comparison of table names in the `table-list` to display table names when tables are subsetted for display. To preserve case, you must enclose table names in the `table-list` in quotation marks.

**EXCLUDE**
- Displays all display tables except those that you specify in the `table-list`.

**EXCLUDEALL**
- Suppresses display of all tables. This option takes precedence over the other options.

**TRACE**
- Displays the display table names, labels, and paths.

---

**DISPLAYOUT Statement**

```
DISPLAYOUT table-spec-list </ options > ;
```

The DISPLAYOUT statement enables you to create CAS output tables from your displayed output. This statement is similar to the ODS OUTPUT statement. For more information about ODS, see SAS Output Delivery System: Procedures Guide.

The `table-spec-list` specifies a list of CAS output tables to create. Each entry in the list has either a `key=value` format or a `key` format:

- **key=value** specifies `key` as the ODS table name, path, or partial pathname, and specifies `value` as the CAS output table name.

- **key** specifies `key` as the ODS table name and also as the CAS output table name.

The ODS table names that you can specify are listed in the section “ODS Table Names” on page 639. You cannot specify the ODS table named OutputCasTables in the `table-spec-list`.

Table names and partial pathnames are discussed under the DISPLAY statement. The DISPLAYOUT statement does not support regular expressions.

You can specify the following options after a slash (/):
**INIT Statement**

INIT variables;

The INIT statement specifies the variables that contain initial weights for the mixture components. By convention, if the model includes a noise component, the first variable in the INIT statement list defines the observation’s initial weight for the noise component. If you are fitting many different models by using the NCLUSTERS=, NOISE=, or COVSTRUCT= option, you must supply enough INIT variables for the model combination that has the most components. The INIT statement is not required. Without an INIT statement, PROC MBC uses the default initialization method.

**OUTPUT Statement**

OUTPUT OUT=\texttt{CAS-libref.data-table} < options > ;

The OUTPUT statement creates a data table that contains observationwise statistics that PROC MBC computes after fitting the model. The variables in the input data table are not included in the output data table, in order to avoid data duplication for large data tables; however, variables that you specify in the COPYVAR= option are included. The output statistics are computed on the basis of the final parameter estimates. If no model converges, then the output data table is not created.

You must specify the following option:

\texttt{OUT=}\texttt{CAS-libref.data-table}

names the output data table for PROC MBC to use. You must specify this option before any other options. \texttt{CAS-libref.data-table} is a two-level name, where

\texttt{CAS-libref} refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to where the data table is to be stored, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about \texttt{CAS-libref}, see the section “Using CAS Sessions and CAS Engine Librefs” on page 621.

\texttt{data-table} specifies the name of the output data table.
You can also specify the following options:

**ALL**
- **ALLSTAT**
  adds all available statistics to the output data table.

**COPYVAR=** `variable`
**COPYVARS=(** `variables` **)**
- transfers one or more `variables` from the input data table to the output data table.

*keyword* `< = name>`
- specifies a statistic to include in the output data table and optionally names the variable `name`. If you provide the *keyword* but do not provide a `name`, PROC MBC assigns a default name that is based on the type of statistic requested. PROC MBC produces the NEXTCLUS statistics even if you do not provide the corresponding keyword.

Table 12.2 summarizes the *keywords* available in the OUTPUT statement.

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
<th>Default Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>CURRCLUS</td>
<td>Specifies the cluster weights that produce the final parameter estimates</td>
<td>CURR</td>
</tr>
<tr>
<td>LOGLIK</td>
<td>Specifies the log likelihood produced by the final parameter estimates</td>
<td>LOGLIK</td>
</tr>
<tr>
<td>MAXPOST</td>
<td>Specifies the index of the component with the maximum posterior probability</td>
<td>MAXPOST</td>
</tr>
<tr>
<td>NEXTCLUS</td>
<td>Specifies the cluster weights produced by the final parameter estimates</td>
<td>NEXT</td>
</tr>
</tbody>
</table>

The following list describes these *keywords*. For more information, see the section “Posterior Probabilities and Clustering” on page 633.

**CURRCLUS < = name >**
- specifies the cluster weights that produce the final parameter estimates. The default name is CURR\(n\), where \(n\) indicates the cluster index. You can modify the name by specifying the *name* option.

**LOGLIK < = name >**
- specifies the component and total log likelihood by using the final parameter estimates. The default name for the component log likelihoods is LOGLIK\(n\), where \(n\) indicates the cluster index. The default name for the total log likelihood is LOGLIK. You can modify this name by specifying the *name* option.

**MAXPOST < = name >**
- specifies the identification of the cluster that has the maximum posterior weight by using the final parameter estimates. The default name is MAXPOST. You can modify this name by specifying the *name* option.

**NEXTCLUS < = name >**
- specifies the cluster weights that the final parameter estimates produce. The default name is NEXT\(n\), where \(n\) indicates the cluster index. You can modify the name by specifying the *name* option.
STORE Statement

STORE <OUT=>CAS-libref.data-table ;

The STORE statement enables you to save the fitted model to a store. You can then use this store with PROC CAS and an appropriate scoring action to score other data.  \textit{CAS-libref.data-table} is a two-level name, where \textit{CAS-libref} refers to the caslib and session identifier, and \textit{data-table} specifies the name of the input data table. For more information about this two-level name, see the \texttt{DATA=} option and the section “Using CAS Sessions and CAS Engine Librefs” on page 621.

VAR Statement

VAR variables ;

The VAR statement specifies the variables that you want to cluster. This statement is required.

Details: MBC Procedure

Log-Likelihood Definitions

The MBC procedure uses two definitions of the log likelihood to assess convergence. Which definition the procedure uses depends on the value of the \texttt{TECHNIQUE=} option.

If you do not specify the \texttt{TECHNIQUE=} option or if you specify \texttt{TECHNIQUE=EM}, PROC MBC uses the mixture log likelihood \( \ell_{\text{mix}} \), defined as

\[
\ell_{\text{mix}} = \sum_{i=1}^{N} \log \left[ \sum_{k=0}^{G} \tau_k f \left( y_i | \theta_k \right) \right]
\]

where \( \tau_k \) are the mixture weights and \( \theta_k \) are the mixture component parameters.

If you specify \texttt{TECHNIQUE=CEM}, the procedure uses the classification complete data log likelihood \( \ell_{\text{cdl}} \), defined as

\[
\ell_{\text{cdl}} = \sum_{i=1}^{N} \sum_{k=0}^{G} \tilde{z}_{ik} \log \left[ \tau_k f \left( y_i | \theta_k \right) \right]
\]

In this definition, \( \tilde{z}_{ik} \) is an indicator variable.

Posterior Probabilities and Clustering

In the \texttt{OUTPUT} statement, the \texttt{CURRCLUS} and \texttt{NEXTCLUS} options produce posterior probabilities (weights) for each observation. The weights that are computed for the \texttt{CURRCLUS} option are the observation
weights that produce the final parameter estimates. In contrast, the weights that are produced for the NEXTCLUS option are the weights that the final parameter estimates produce for each observation.

The MAXPOST option outputs the index of the cluster that has the maximum posterior probability for each observation. This maximum is taken over the posterior probabilities that are produced by the final parameter estimates.

---

**Expectation-Maximization (EM) Algorithm**

**Traditional EM**

The expectation-maximization (EM) algorithm (Dempster, Laird, and Rubin 1977) is a tool that addresses problems of missing data. The EM algorithm proceeds by finding the conditional expectation of the missing data, given the observed data, treating that conditional expectation as observed data, and iterating until the model converges. In the models that PROC MBC fits, the missing data are the identity of the cluster to which each observation belongs. The steps of the EM algorithm follow:

1. Initialize the cluster membership for each observation through the specified technique (INIT= or INIT statement).
2. Update the cluster-specific parameters by using the current cluster memberships. Retain these new parameters as $\hat{\theta}^{(t+1)}$.
3. Given the current parameters $\hat{\theta}^{(t+1)}$, compute the conditional expected values of cluster membership for each observation. These are estimates $\hat{z}_{ik}$ such that
   
   $\hat{z}_{ik} = \text{estimated probability that observation } i \text{ belongs to cluster } k; 0 \leq z_{ik} \leq 1 \forall i, k; \sum_{k=1}^{G} \hat{z}_{ik} = 1$

4. Evaluate the mixture log likelihood at the current parameter and cluster assignments, $\ell^{(k+1)}$. If the quantity $|\ell_{\max}^{(k+1)} - \ell_{\max}^{(k)}| / \ell_{\max}^{(k)}$ is less than or equal to the specified value of the EMCRITERION= option, terminate the algorithm. Otherwise, go to step 2.

**Classification EM**

Classification EM (Celeux and Govaert 1995) proceeds in the same manner as traditional EM, but it uses the classification log likelihood to assess convergence in step 4 and replaces the “fuzzy” $\hat{z}_{ik}$ in step 3 with hard classifications: $\tilde{z}_{ik} = 1$ if $\hat{z}_{ik} = \max_j \{\hat{z}_{ij}\}$, $\tilde{z}_{ik} = 0$ otherwise.

**Singular Covariance Matrices**

In some circumstances, the covariance matrix for a Gaussian component can become singular during the progress of the EM algorithm. This terminates the model fitting process. This can happen if the cluster represents a single observation or a set of effectively identical observations. When this happens, PROC MBC issues a note that indicates the existence of the singularity. You might be able to use a different initialization method or random seed to alter the progress of the algorithm and so avoid the singularity, but this is not possible in all cases.
Covariance Structures

The covariance structures available in PROC MBC follow the notation of Banfield and Raftery (1993). In this treatment, the three aspects of each component’s covariance (shape, volume, and orientation) can be left arbitrary or can be forced to be equal across clusters. In addition, the shape can be forced to be spherical, and the orientation can be forced to align with the coordinate axes.

The eigenvalue decomposition of the covariance matrix for the $k$th Gaussian mixture component is written as

$$\Sigma_k = \lambda_k D_k A_k D_k'$$

where $\lambda_k$ is the volume factor for $\Sigma_k$, $A_k$ is a diagonal matrix with the normalized eigenvalues of $\Sigma_k$ on the diagonal, and $D_k$ is the matrix of eigenvectors of $\Sigma_k$. In a geometric description of the covariance matrix $\Sigma_k$, $\lambda_k$ characterizes the volume, $A_k$ characterizes the shape, and $D_k$ characterizes the orientation.

There are several ways to restrict the covariance structures across the Gaussian clusters. You can hold the volume constant by setting each $\lambda_k$ equal to $\lambda$. To leave the shape arbitrary but common across the clusters, you can require each $A_k$ to equal $A$. To require spherical covariance matrices, you can restrict each $A_k$ to equal $I$, the identity matrix. If you hold each $D_k$ equal to $D$, then all covariance matrices will have the same orientation. You can also require all covariance matrices to be oriented to the coordinate axes by specifying $D_k$ equal to $I$.

PROC MBC accommodates a subset of the possible covariance structure restrictions. Table 12.3 describes the structures that the procedure supports. The Name column indicates how you specify each structure when invoking the MBC procedure. The Profile column shows a two-dimensional visualization of the covariance structure in a model that contains two clusters.

<table>
<thead>
<tr>
<th>Name</th>
<th>Structure</th>
<th>Volume</th>
<th>Shape</th>
<th>Orientation</th>
<th>Profile</th>
</tr>
</thead>
</table>
| EEE  | $\lambda D A D'$ | Equal  | Equal | Equal       | ![Profile](image)
| EEI  | $\lambda A$ | Equal  | Equal | Coordinate axes | ![Profile](image)
| EEV  | $\lambda D_k A D_k'$ | Equal  | Equal | Variable    | ![Profile](image)
| EII  | $\lambda I$ | Equal  | Spherical | Coordinate axes | ![Profile](image)
<table>
<thead>
<tr>
<th>Name</th>
<th>Structure</th>
<th>Volume</th>
<th>Shape</th>
<th>Orientation</th>
<th>Profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>EVI</td>
<td>$\lambda A_k$</td>
<td>Equal</td>
<td>Variable</td>
<td>Coordinate axes</td>
<td><img src="image" alt="Profile" /></td>
</tr>
<tr>
<td>EVV</td>
<td>$\lambda D_k A_k D'_k$</td>
<td>Equal</td>
<td>Variable</td>
<td>Variable</td>
<td><img src="image" alt="Profile" /></td>
</tr>
<tr>
<td>VII</td>
<td>$\lambda_k I$</td>
<td>Variable</td>
<td>Spherical</td>
<td>Coordinate axes</td>
<td><img src="image" alt="Profile" /></td>
</tr>
<tr>
<td>VVI</td>
<td>$\lambda_k A_k$</td>
<td>Variable</td>
<td>Variable</td>
<td>Coordinate axes</td>
<td><img src="image" alt="Profile" /></td>
</tr>
<tr>
<td>VVV</td>
<td>$\lambda_k D_k A_k D'_k$</td>
<td>Variable</td>
<td>Variable</td>
<td>Variable</td>
<td><img src="image" alt="Profile" /></td>
</tr>
</tbody>
</table>

### Noise Component

A noise component is uniform noise. This can be useful for modeling outliers. PROC MBC estimates the density of this cluster as the inverse of the volume of the space that contains the observations. The procedure computes the volume as the product of the range of the data in each dimension.

### Missing Data

PROC MBC excludes from the analysis observations that have missing values for any variables in the VAR list.

If you use the INIT statement to specify initial weights and any of the variables in the INIT list have missing values, PROC MBC uses a small positive weight for each variable that has a missing value.

### Model Selection

You can evaluate several different models by specifying lists of numbers of components, covariance structures, or noise components in the PROC MBC statement. The CRITERION= option in the PROC MBC statement...
specifies the model fit criterion that the procedure uses to compare different models. Each criterion uses the log likelihood of the model, given the training data, with a criterion-specific penalty term that favors smaller models.

Table 12.4 summarizes the computation of the available criteria.

### Table 12.4  Model Selection Criteria

<table>
<thead>
<tr>
<th>Name</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIC</td>
<td>$-2LL + 2p$</td>
</tr>
<tr>
<td>AICC</td>
<td>$-2LL + \frac{2pn}{n-p-1}$</td>
</tr>
<tr>
<td>BIC</td>
<td>$-2LL + p \log(n)$</td>
</tr>
<tr>
<td>LOGL</td>
<td>$-2 \sum_{i=1}^{n} \log \left[ \sum_{k=1}^{K} \tau_k f_k(y_i</td>
</tr>
</tbody>
</table>

---

**Number of Parameters**

The “Model Selection” section defines each information criterion as a function of the number of observations $n$ and the number of parameters $p$. The number of parameters depends on the number and type of components in the model, as well as the covariance structure that is chosen for the Gaussian components. Table 12.5 summarizes the methods that PROC MBC uses to determine the number of parameters in a particular model. In this table, $G$ refers to the number of Gaussian clusters in the model, and $d$ refers to the number of variables in the model. In the entries for each covariance structure, the total number of parameters is presented as the sum of contributions from volume, shape, and orientation.

### Table 12.5  Number of Parameters Computation

<table>
<thead>
<tr>
<th>Source</th>
<th>Number of Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mixing probabilities $\tau_k$</td>
<td>$G - 1$ without noise; $G$ with noise</td>
</tr>
<tr>
<td>Cluster means $\mu_k$</td>
<td>$Gd$</td>
</tr>
<tr>
<td>Noise volume</td>
<td>1</td>
</tr>
<tr>
<td><strong>Covariance Matrix $\Sigma_k$</strong></td>
<td></td>
</tr>
<tr>
<td>EEE</td>
<td>$1 + (d - 1) + (d - 1)d/2$</td>
</tr>
<tr>
<td>EEI</td>
<td>$1 + (d - 1)$</td>
</tr>
<tr>
<td>EEV</td>
<td>$1 + (d - 1) + G(d - 1)d/2$</td>
</tr>
<tr>
<td>EII</td>
<td>1</td>
</tr>
<tr>
<td>EVI</td>
<td>$1 + G(d - 1)$</td>
</tr>
<tr>
<td>EVV</td>
<td>$1 + G(d - 1) + G(d - 1)d/2$</td>
</tr>
<tr>
<td>VII</td>
<td>$G$</td>
</tr>
<tr>
<td>VVI</td>
<td>$G + G(d - 1)$</td>
</tr>
<tr>
<td>VVV</td>
<td>$G + G(d - 1) + G(d - 1)d/2$</td>
</tr>
</tbody>
</table>
Displayed Output

The following sections describe the output that PROC MBC produces. The output is organized into various tables, which are discussed in their order of appearance.

Model Information

The “Model Information” table displays basic information about the model, including the number of Gaussian clusters, the covariance structure of those clusters, and whether a noise component is included in the model. The table also includes information about the method that is used to estimate the parameters in the model and the tolerances for covariance matrices and parameters. If you specify multiple models, the table displays information about the selected model and also includes the model selection criterion that is used.

Number of Observations

The “Number of Observations” table displays the number of observations that are read from the input data table and the number of observations that are used in the analysis.

Iteration History

The “Iteration History” table displays the progress of the estimation algorithm for each model that is fitted. By default, this table includes information about the model, the iteration number, the log likelihood at that iteration, the change in the log likelihood, and the computed convergence criterion value. If you specify the ITHIST=DETAILS option, the table also includes the values of the model parameters at each iteration.

Convergence Status

The “Convergence Status” table displays the convergence status of each model. The table includes information about the model and a message that indicates whether the model converged or encountered an error.

Fit Statistics

The “Fit Statistics” table displays several likelihood-based measures of fit for the specified model. If you specify more than one model, these values correspond to the selected model. The table also includes the number of parameters that are used in the computation of the fit statistics. For more information about the computation of the fit statistics, see Table 12.4.

Parameter Estimates

The “Parameter Estimates” table displays the mean and covariance estimates for the specified model. If you specify more than one model, these values correspond to the selected model.

Mixing Estimates

The “Mixing Estimates” table displays the mixing probability estimates for the specified model. These values correspond to the \( \tau_k \) that is presented in the section “Log-Likelihood Definitions” on page 633. If you specify more than one model, these values correspond to the selected model.
Fit Summary

The “Fit Summary” table displays the fit statistics for a number of the fitted models. By default, this table shows the 10 models that have the best (smallest) value of the model selection criterion that is specified in the CRITERION= option. You can modify the number of models that the table includes by using the TOPMODELS= option. This table is displayed only if you specify more than one model.

Timing

The “Timing” table displays the amount of time (in seconds) that PROC MBC required to perform different tasks in the analysis.

OutputCasTables Table

The OutputCasTables table is a special table that has information about each CAS table that is created during a CAS action execution. The information for each CAS table consists of the CAS table name, the caslib in which the table resides, and the number of columns and rows in the CAS table. Because this table is not a typical ODS table that contains analytical results, you cannot include it in the table-spec-list in the DISPLAYOUT statement.

**ODS Table Names**

Each table that the MBC procedure creates has a name associated with it. You must use this name to refer to the table when you use the DISPLAY statement, the DISPLAYOUT statement, or ODS statements. These names are listed in Table 12.6.

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConvergenceStatus</td>
<td>Convergence status of each model</td>
<td>PROC MBC</td>
<td>Default</td>
</tr>
<tr>
<td>FitStatistics</td>
<td>Fit statistics for specified or selected model</td>
<td>PROC MBC</td>
<td>Default</td>
</tr>
<tr>
<td>FitSummary</td>
<td>Comparison of fit statistics for all models</td>
<td>PROC MBC</td>
<td>Multiple values in COVSTRUCT=, NOISE=, or NCLUSTERS= ITHIST</td>
</tr>
<tr>
<td>IterHistory</td>
<td>Iteration history from the EM algorithm</td>
<td>PROC MBC</td>
<td>Default</td>
</tr>
<tr>
<td>MixingEstimates</td>
<td>Estimates for mixing proportions</td>
<td>PROC MBC</td>
<td>Default</td>
</tr>
<tr>
<td>ModelInfo</td>
<td>Information about the fitted model</td>
<td>PROC MBC</td>
<td>Default</td>
</tr>
<tr>
<td>NObs</td>
<td>Information about observations</td>
<td>PROC MBC</td>
<td>Default</td>
</tr>
<tr>
<td>OutputCasTables</td>
<td>See the section “OutputCasTables Table” on page 639</td>
<td>OUTPUT DISPLAYOUT</td>
<td>OUT=</td>
</tr>
<tr>
<td>ParameterEstimates</td>
<td>Mean and covariance estimates</td>
<td>PROC MBC</td>
<td>Default</td>
</tr>
<tr>
<td>Timing</td>
<td>Time used for different tasks</td>
<td>PROC MBC</td>
<td>Default</td>
</tr>
</tbody>
</table>
Example 12.1: Storing and Scoring

When you use PROC MBC to fit a model, you can use the STORE statement to save the model for later application to a new data table. For example, you can score new observations to find their posterior probabilities of cluster membership for each cluster in the model. This example uses PROC CAS to run the \texttt{mbcScore} action to do this subsequent scoring.

The following statements generate a data table that represents a scenario in product marketing. The different market segments are distinguished by different levels of certain financial characteristics.

Segment 1 is a generally younger group with a higher debt-to-income ratio, and it might be attracted by newer, more luxurious goods. When marketing to this group, you would like to highlight newer lines with higher price points. In contrast, segment 2, with a low debt-to-income ratio and an average age between that of segments 1 and 3, might represent a group of bargain hunters who would prefer a product that offers the best value. The customers in segment 3 have a higher average age and a moderate debt-to-income ratio, and they represent a group that is not interested in luxury items and not strongly motivated by price. Finally, segment 4 represents customers who do not fit into these groups and so might respond to a variety of offers.

These statements assume that your CAS engine libref is named \texttt{mycas}, but you can substitute any appropriately defined CAS engine libref.

```plaintext
data mycas.marketdata;
  label moninc = 'Monthly Income'
  mondebt = 'Monthly Debt'
  tenancy = 'Months at Current Residence'
  ageyrs = 'Age';
  call streaminit(869884);

  * -- Segment 1 : Young Spenders ------------ ;
  do j = 1 to 1000;
    moninc = 12 + sqrt(2) * rand('normal');
    mondebt = 6 + sqrt(1) * rand('normal');
    tenancy = 18 + sqrt(2) * rand('normal');
    ageyrs = 30 + sqrt(8) * rand('normal');
    output;
  end;

  * -- Segment 2 : Middle-Aged Savers --------- ;
  do j = 1 to 3000;
    moninc = 10 + sqrt(2.5) * rand('normal');
    mondebt = 2 + sqrt(0.25) * rand('normal');
    tenancy = 24 + sqrt(3) * rand('normal');
    ageyrs = 40 + sqrt(8) * rand('normal');
    output;
  end;
```

* -- Segment 3: Comfortably Established --- ;
  do j = 1 to 6000;
    moninc = 15 + sqrt(2) * rand('normal');
    mondebt = 3 + sqrt(0.5) * rand('normal');
    tenancy = 32 + sqrt(4) * rand('normal');
    ageyrs = 50 + sqrt(5) * rand('normal');
    output;
  end;
* -- Segment 4: Defying Classification ----- ;
  do j = 1 to 300;
    moninc = rand('uniform') * 15 + 5;
    mondebt = rand('uniform') * 9 + 0;
    tenancy = rand('uniform') * 30 + 10;
    ageyrs = rand('uniform') * 40 + 20;
    output;
  end;
  drop j;

The existence of outliers is a good motivation to include a noise component in the model, because the outliers can otherwise distort the cluster structure in the model. You can use the NOISE=YES option to specify this directly. The following statements fit a range of models to these data. Each model includes a noise component.

```
proc mbc data=mycas.marketdata nclusters=(2 3 4 5) noise=YES seed=1389035719;
  var moninc mondebt tenancy ageyrs;
run;
```

Output 12.1.1 shows the “Fit Summary” table, indicating that the model with the best fit has three Gaussian clusters and a noise component.

```
Output 12.1.1 Mean and Covariance Estimates for Selected Model

The MBC Procedure

<table>
<thead>
<tr>
<th>Covariance Structure</th>
<th>Number of Clusters</th>
<th>Noise Component</th>
<th>Number of Parameters</th>
<th>-2 Log L</th>
<th>AIC</th>
<th>AICC</th>
<th>BIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>VVV</td>
<td>3</td>
<td>Y</td>
<td>46</td>
<td>170967</td>
<td>171059</td>
<td>171059</td>
<td>171392</td>
</tr>
<tr>
<td>VVV</td>
<td>4</td>
<td>Y</td>
<td>61</td>
<td>170965</td>
<td>171087</td>
<td>171088</td>
<td>171529</td>
</tr>
<tr>
<td>VVV</td>
<td>5</td>
<td>Y</td>
<td>76</td>
<td>170942</td>
<td>171094</td>
<td>171095</td>
<td>171644</td>
</tr>
<tr>
<td>VVV</td>
<td>2</td>
<td>Y</td>
<td>31</td>
<td>182253</td>
<td>182315</td>
<td>182315</td>
<td>182539</td>
</tr>
</tbody>
</table>
```

The goal of this analysis is to identify segments within the group of customers, and you can use the OUTPUT statement to label each customer with its most strongly associated segment. The MAXPOST option in the following OUTPUT statement includes the index of the component that has the highest posterior clustering probability for each observation. The indices start at zero for a model that includes a noise component:

```
proc mbc data=mycas.marketdata nclusters=(2 3 4 5) noise=YES;
  var moninc mondebt tenancy ageyrs;
  output out=mycas.mktscore copyvars=(moninc mondebt tenancy ageyrs) maxpost;
run;
```
In Output 12.1.2, the “Mixing Estimates” table indicates a mixture component with index 0; this is the noise component.

**Output 12.1.2  Mixing Estimates for Selected Model**

<table>
<thead>
<tr>
<th>Mixing Component</th>
<th>Mixing Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.02865</td>
</tr>
<tr>
<td>1</td>
<td>0.58248</td>
</tr>
<tr>
<td>2</td>
<td>0.09776</td>
</tr>
<tr>
<td>3</td>
<td>0.29110</td>
</tr>
</tbody>
</table>

Output 12.1.3 shows the parameter estimates for the selected model.

**Output 12.1.3  Parameter Estimates for Selected Model**

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Variable</th>
<th>Mean moninc</th>
<th>mondebt</th>
<th>tenancy</th>
<th>ageyrs</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>moninc</td>
<td>15.00354</td>
<td>2.05211</td>
<td>-0.00126</td>
<td>0.02988</td>
</tr>
<tr>
<td></td>
<td>mondebt</td>
<td>3.00321</td>
<td>0.50819</td>
<td>0.05722</td>
<td>0.00485</td>
</tr>
<tr>
<td></td>
<td>tenancy</td>
<td>31.99125</td>
<td>4.01148</td>
<td>-0.03269</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ageyrs</td>
<td>50.04533</td>
<td>5.03305</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>moninc</td>
<td>11.96829</td>
<td>2.12425</td>
<td>0.02417</td>
<td>-0.09442</td>
</tr>
<tr>
<td></td>
<td>mondebt</td>
<td>6.01248</td>
<td>1.06665</td>
<td>-0.00584</td>
<td>0.05854</td>
</tr>
<tr>
<td></td>
<td>tenancy</td>
<td>17.99029</td>
<td>2.01793</td>
<td>0.03317</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ageyrs</td>
<td>29.99537</td>
<td>7.41317</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>moninc</td>
<td>9.99463</td>
<td>2.52433</td>
<td>-0.01639</td>
<td>0.05279</td>
</tr>
<tr>
<td></td>
<td>mondebt</td>
<td>1.99783</td>
<td>0.24534</td>
<td>0.00233</td>
<td>0.00226</td>
</tr>
<tr>
<td></td>
<td>tenancy</td>
<td>24.01474</td>
<td>2.93710</td>
<td>0.05776</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ageyrs</td>
<td>39.91170</td>
<td>8.17630</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The clusters that the procedure has identified align well with the segments that are generated in the example. To market effectively to these different groups, you want to be able to assign a cluster membership score to each potential customer. You can use the STORE statement as follows to save the fitted model for later scoring:

```plaintext
proc mbc data=mycas.marketdata nclusters=(2 3 4 5) noise=YES seed=1389035719;
  var moninc mondebt tenancy ageyrs;
  store mycas.mktgroups;
run;
```

The mycas.mktgroups store contains a representation of the model that you can apply to a new set of customer data. You can use the mbcScore action to score new observations according to this saved model.
The following statements show a possible set of new data:

```
data mycas.newcusts;
  input moninc mondebt tenancy ageyrs;
datalines;
  7   5   20   35
  12  6   20   30
  18  4   31   52
  10  4   25   37
;
```

The following statements show how to score the new set by using PROC CAS to invoke the `mbcScore` action. The `mbcScore` action call takes several parameters. You specify the new observations to score by using the `table` parameter. The saved model `mycas.mktgroups` is specified in the `restore` parameter. You use the `casOut` parameter to designate that the data table `mycas.newScores` will contain the scores for the new data. The `copyvars` parameter includes the listed input variables in the output data table. The `maxpost` parameter requests that the variable `Group` in the output set identify the cluster that has the highest posterior weight for each observation. Finally, you use the `nextclus` parameter to specify the root of the variable name for each cluster weight.

```
proc cas;
  action mbc.mbcScore /
    table={name='newcusts'}
    restore={name='mktgroups'}
    casOut={name='newScores', replace=true}
    copyvars={'moninc' , 'mondebt' , 'tenancy' , 'ageyrs'}
    maxpost='group'
    nextclus='cluswt';
  run;
quit;
```

When the scoring is finished, you can examine the clusterings in the data table `mycas.newScores`. Output 12.1.4 shows the input data and the resulting weights and clustering.

**Output 12.1.4 Weights and Scores for New Data**

<table>
<thead>
<tr>
<th>clusw0</th>
<th>clusw1</th>
<th>clusw2</th>
<th>clusw3</th>
<th>group</th>
<th>moninc</th>
<th>mondebt</th>
<th>tenancy</th>
<th>ageyrs</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.73350</td>
<td>0.00000</td>
<td>0.26650</td>
<td>0.00000</td>
<td>0</td>
<td>7</td>
<td>5</td>
<td>20</td>
<td>35</td>
</tr>
<tr>
<td>0.00096</td>
<td>0.00000</td>
<td>0.99904</td>
<td>0.00000</td>
<td>2</td>
<td>12</td>
<td>6</td>
<td>20</td>
<td>30</td>
</tr>
<tr>
<td>0.00176</td>
<td>0.99824</td>
<td>0.00000</td>
<td>0.00000</td>
<td>1</td>
<td>18</td>
<td>4</td>
<td>31</td>
<td>52</td>
</tr>
<tr>
<td>0.35676</td>
<td>0.00000</td>
<td>0.00001</td>
<td>0.64323</td>
<td>3</td>
<td>10</td>
<td>4</td>
<td>25</td>
<td>37</td>
</tr>
</tbody>
</table>

You can use the values in the `Group` variable to decide what action to take for each customer. You can also consider the strength of the association shown in the variables `clusw0` through `clusw3`. In this case, each observation has a strong association with its assigned cluster, but one observation has a noticeable affinity for the noise cluster as well.
Example 12.2: Adjusting Starting Values

This example shows how you can use alternative starting values to influence the model fitting process.

The data in this example consist of physical measurements of 159 freshwater fish. The data set is available from Puranen (1917).

Each measurement includes species, weight, height, width, and three length variables. The values that are recorded for height and width are percentages of the third length variable.

The following DATA step creates the data table mycas.fish. These statements assume that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

data mycas.fish (drop=HtPct WidthPct);
  input Species Weight Length1 Length2 Length3 HtPct WidthPct @@;
  *** transform variables;
  if Weight <= 0 or Weight =. then delete;
  Weight3=Weight**(1/3);
  Height=HtPct*Length3/(Weight3*100);
  Width=WidthPct*Length3/(Weight3*100);
  Length1=Length1/Weight3;
  Length2=Length2/Weight3;
  Length3=Length3/Weight3;
  logLengthRatio=log(Length3/Length1);
datalines;
  1 242.0 23.2 25.4 30.0 38.4 13.4 1 290.0 24.0 26.3 31.2 40.0 13.8
  1 340.0 23.9 26.5 31.1 39.8 15.1 1 363.0 26.3 29.0 33.5 38.0 13.3
  1 430.0 26.5 29.0 34.0 36.6 15.1 1 450.0 26.8 29.7 34.7 39.2 14.2
  1 500.0 26.8 29.7 34.5 41.1 15.3 1 390.0 27.6 30.0 35.0 36.2 13.4
  1 450.0 27.6 30.0 35.1 39.9 13.8 1 500.0 28.5 30.7 36.2 39.3 13.7
  1 475.0 28.4 31.0 36.2 39.4 14.1 1 500.0 28.7 31.0 36.2 39.7 13.3
  1 500.0 29.1 31.5 36.4 37.8 12.0 1 29.5 32.0 37.3 37.3 13.6
  1 600.0 29.4 32.0 37.2 42.2 13.9 1 600.0 29.4 32.0 37.2 41.5 15.0
  1 700.0 30.4 33.0 38.3 38.8 13.8 1 700.0 30.4 33.0 38.5 38.8 13.5
  1 610.0 30.9 33.5 38.6 40.5 13.3 1 650.0 31.0 33.5 38.7 37.4 14.8
  1 575.0 31.3 34.0 39.5 38.3 14.1 1 685.0 31.4 34.0 39.2 40.8 13.7
  1 620.0 31.5 34.5 39.7 39.1 13.3 1 680.0 31.8 35.0 40.6 38.1 15.1

  ... more lines ...

  7  19.7 13.2 14.3 15.2 18.9 13.6 7  19.9 13.8 15.0 16.2 18.1 11.6
;

This example uses the variables height, width, and weight3 for analysis. Your goal is to find the model that best represents the clustering in these three variables.

The “Getting Started” example and “Example 12.1: Storing and Scoring” on page 640 both show how to use PROC MBC to compare several models, but both examples rely on random starting values to begin the model fitting. PROC MBC provides two other ways to generate starting values: the $k$-means technique and predefined starting values.
When you specify INIT=KMEANS, the procedure uses the \( k \)-means technique to get starting values for the cluster weights, means, and covariances. This is equivalent to starting with the M-step in the EM algorithm. The \( k \)-means method might be a better method for starting values in some situations, particularly where the clusters are roughly spherical.

You use the INIT statement to specify variables that contain initial cluster assignment weights for each observation. The procedure uses these weights for the initial E-step in the EM algorithm. You might want to use the INIT statement in cases where you have prior knowledge of clustering and want to use that knowledge to influence the model fitting process.

The following statements use the default, or random start, method to generate initial values and to choose from several models that are distinguished by covariance structure:

```
proc mbc data=mycas.fish seed=9982346 covstruct=(eee eei eev eii evi evv vii vvi vvv)
nclusters=3;
var height width weight3;
run;
```

Output 12.2.1 summarizes the parameter estimates, mixing estimates, and fit statistics for the selected model, which has three clusters and uses the VVV covariance structure.

**Output 12.2.1**  Selected Model Using INIT=RANDOM Option

<table>
<thead>
<tr>
<th>The MBC Procedure</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Parameter Estimates for Selected Model</th>
<th>Covariance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster</td>
<td>Variable</td>
</tr>
<tr>
<td>1</td>
<td>Height</td>
</tr>
<tr>
<td></td>
<td>Width</td>
</tr>
<tr>
<td></td>
<td>Weight3</td>
</tr>
<tr>
<td>2</td>
<td>Height</td>
</tr>
<tr>
<td></td>
<td>Width</td>
</tr>
<tr>
<td></td>
<td>Weight3</td>
</tr>
<tr>
<td>3</td>
<td>Height</td>
</tr>
<tr>
<td></td>
<td>Width</td>
</tr>
<tr>
<td></td>
<td>Weight3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mixing Probability Estimates for Selected Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mixing Component</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
</tbody>
</table>
To see the effect of a different method of determining starting values, you can use the INIT=KMEANS option. The following statements use the k-means method of initialization:

```plaintext
proc mbc data=mycas.fish seed=9982346 init=kmeans
covstruct=(eee eei eev eii evi evv vii vvi vvv)
    nclusters=3;
    var height width weight3;
run;
```

Output 12.2.2 summarizes the model selection process for the k-means method. The selected model uses the EVV covariance structure, and the fit statistics for other models are different from the result that uses the default random start method. In particular, the log-likelihood statistic for each of the top three models is a good deal worse with the k-means method than with the default method. This might be associated with the fact that the three clusters are not really spherical, so the starting values that are determined using the k-means method might not be suitable in this situation.

```
Output 12.2.2  Model Selection Summary Using INIT=KMEANS Option
```
Output 12.2.3 shows the parameter estimates and mixing estimates for the selected model. The parameter estimates and mixing estimates are broadly similar to those for the default initialization method, but dissimilar enough to drive the difference in the selected model.

**Output 12.2.3** Parameter Estimates and Mixing Estimates Using INIT=KMEANS Option

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Variable</th>
<th>Mean</th>
<th>Height</th>
<th>Width</th>
<th>Weight3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Height</td>
<td>1.21348</td>
<td>0.00430</td>
<td>0.00014041</td>
<td>0.03749</td>
</tr>
<tr>
<td></td>
<td>Width</td>
<td>0.71109</td>
<td>0.00176</td>
<td>0.02778</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Weight3</td>
<td>6.27437</td>
<td></td>
<td>4.08255</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Height</td>
<td>0.93892</td>
<td>0.00327</td>
<td>0.00015841</td>
<td>-0.11160</td>
</tr>
<tr>
<td></td>
<td>Width</td>
<td>0.59277</td>
<td>0.00135</td>
<td>0.01159</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Weight3</td>
<td>5.88795</td>
<td></td>
<td>9.76185</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Height</td>
<td>1.77964</td>
<td>0.00757</td>
<td>0.00124</td>
<td>0.08550</td>
</tr>
<tr>
<td></td>
<td>Width</td>
<td>0.63619</td>
<td>0.00142</td>
<td>0.03318</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Weight3</td>
<td>7.65945</td>
<td></td>
<td>4.02246</td>
<td></td>
</tr>
</tbody>
</table>

The remaining option for adjusting starting values is the INIT statement. Using the INIT statement requires you to specify variables that give initial weights for each observation.

The following statements produce a scatter plot of the Height and Width variables, labeling each point with a species number:

```sas
proc sgplot data=mycas.fish;
scatter x=height y=width / group=species;
run; quit;
```

Output 12.2.4 suggests that the species encodes useful information about the clustering.
You can use these visible clusterings to generate initialization variables. The following DATA step shows one way to generate initialization variables that reflect your observations:

```plaintext
data mycas.fishlbl;
  set mycas.fish;
  if (species in (6,7)) then do; z1=1; z2=0; z3=0; end;
  else if (species in (1,4)) then do; z1=0; z2=1; z3=0; end;
  else do; z1=0; z2=0; z3=1; end;
;
PROC MBC uses the initialization values for each observation as a set of weights that drive the parameter estimates and mixing estimates for the first EM iteration. The following statements use these initialization values by including them in the desired order in the INIT statement:

```plaintext
proc mbc data=mycas.fishlbl
  covstruct=(eee eei eev eii evi evv vii vvi vvv)
  nclusters=3;
  var height width weight3;
  init z1 z2 z3;
run;
```
Example 12.2: Adjusting Starting Values

Output 12.2.5 summarizes the model selection process. The ordering of models does not match that from either of the previous results, but the selected model uses the same covariance structure (VVV) as the random start approach.

**Output 12.2.5** Model Selection Summary Using INIT Statement

<table>
<thead>
<tr>
<th>Covariance Structure</th>
<th>Number of Clusters</th>
<th>Noise Component</th>
<th>Number of Parameters</th>
<th>-2 Log L</th>
<th>AIC</th>
<th>AICC</th>
<th>BIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>VVV</td>
<td>3</td>
<td>N</td>
<td>29</td>
<td>-19.57464</td>
<td>38.42536</td>
<td>52.12615</td>
<td>127.05649</td>
</tr>
<tr>
<td>EVV</td>
<td>3</td>
<td>N</td>
<td>27</td>
<td>-9.23633</td>
<td>44.76367</td>
<td>56.48460</td>
<td>127.28230</td>
</tr>
<tr>
<td>EEE</td>
<td>3</td>
<td>N</td>
<td>23</td>
<td>12.65665</td>
<td>58.65665</td>
<td>66.95741</td>
<td>128.95031</td>
</tr>
<tr>
<td>EEI</td>
<td>3</td>
<td>N</td>
<td>17</td>
<td>48.50809</td>
<td>82.50809</td>
<td>86.91097</td>
<td>134.46427</td>
</tr>
<tr>
<td>EVI</td>
<td>3</td>
<td>N</td>
<td>14</td>
<td>64.01230</td>
<td>92.01230</td>
<td>94.97005</td>
<td>134.79974</td>
</tr>
<tr>
<td>VVI</td>
<td>3</td>
<td>N</td>
<td>18</td>
<td>12.65665</td>
<td>58.65665</td>
<td>66.95741</td>
<td>128.95031</td>
</tr>
<tr>
<td>VII</td>
<td>3</td>
<td>N</td>
<td>12</td>
<td>1026.60540</td>
<td>1054.60540</td>
<td>1057.56314</td>
<td>1097.39284</td>
</tr>
<tr>
<td>EII</td>
<td>3</td>
<td>N</td>
<td>18</td>
<td>1075.29045</td>
<td>1099.29045</td>
<td>1101.45712</td>
<td>1135.96540</td>
</tr>
</tbody>
</table>

Output 12.2.6 shows the parameter estimates and mixing estimates. The selected model matches the selected model from the random start approach, but with a different order for the clusters.

**Output 12.2.6** Parameter Estimates and Mixing Estimates Using INIT Statement

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Variable</th>
<th>Mean</th>
<th>Height</th>
<th>Width</th>
<th>Weight3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Height</td>
<td>0.93901</td>
<td>0.00394</td>
<td>0.00019642</td>
<td>-0.13391</td>
</tr>
<tr>
<td></td>
<td>Width</td>
<td>0.59276</td>
<td>0.01162</td>
<td>0.01388</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Weight3</td>
<td>5.88904</td>
<td></td>
<td>11.69267</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Height</td>
<td>1.77964</td>
<td>0.00529</td>
<td>0.00086794</td>
<td>0.05971</td>
</tr>
<tr>
<td></td>
<td>Width</td>
<td>0.63619</td>
<td>0.0098863</td>
<td>0.02317</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Weight3</td>
<td>7.65945</td>
<td></td>
<td>2.80942</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Height</td>
<td>1.21341</td>
<td>0.00472</td>
<td>0.00015780</td>
<td>0.04118</td>
</tr>
<tr>
<td></td>
<td>Width</td>
<td>0.71108</td>
<td>0.00192</td>
<td>0.03042</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Weight3</td>
<td>6.27394</td>
<td></td>
<td>4.46530</td>
<td></td>
</tr>
</tbody>
</table>

This approach results in different fit statistics for some of the other models that are considered. In this case, the use of initialization variables through the INIT statement does not select a model different from that selected using the default method. However, the difference in the fit statistics for the other models that are
considered in the selection process shows the effect of the different initialization method.

References


Chapter 13
The MODELMATRIX Procedure

## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview: MODELMATRIX Procedure</td>
<td>652</td>
</tr>
<tr>
<td>PROC MODELMATRIX Features</td>
<td>652</td>
</tr>
<tr>
<td>PROC MODELMATRIX Compared with Other SAS Procedures</td>
<td>653</td>
</tr>
<tr>
<td>Using CAS Sessions and CAS Engine Librefs</td>
<td>653</td>
</tr>
<tr>
<td>Getting Started: MODELMATRIX Procedure</td>
<td>654</td>
</tr>
<tr>
<td>Syntax: MODELMATRIX Procedure</td>
<td>657</td>
</tr>
<tr>
<td>PROC MODELMATRIX Statement</td>
<td>658</td>
</tr>
<tr>
<td>BY Statement</td>
<td>658</td>
</tr>
<tr>
<td>CLASS Statement</td>
<td>658</td>
</tr>
<tr>
<td>DISPLAY Statement</td>
<td>659</td>
</tr>
<tr>
<td>DISPLAYOUT Statement</td>
<td>660</td>
</tr>
<tr>
<td>EFFECT Statement</td>
<td>661</td>
</tr>
<tr>
<td>FREQ Statement</td>
<td>662</td>
</tr>
<tr>
<td>MODEL Statement</td>
<td>662</td>
</tr>
<tr>
<td>OUTPUT Statement</td>
<td>663</td>
</tr>
<tr>
<td>WEIGHT Statement</td>
<td>664</td>
</tr>
<tr>
<td>Details: MODELMATRIX Procedure</td>
<td>664</td>
</tr>
<tr>
<td>Computational Method</td>
<td>664</td>
</tr>
<tr>
<td>Displayed Output</td>
<td>664</td>
</tr>
<tr>
<td>ODS Table Names</td>
<td>665</td>
</tr>
<tr>
<td>Examples: MODELMATRIX Procedure</td>
<td>666</td>
</tr>
<tr>
<td>Example 13.1: Spline Effects and BY-Group Processing</td>
<td>666</td>
</tr>
<tr>
<td>References</td>
<td>671</td>
</tr>
</tbody>
</table>
Overview: MODELMATRIX Procedure

The MODELMATRIX procedure creates a design matrix (matrix of covariates) that is associated with a user-specified MODEL statement and a user-specified data table in SAS Viya.

The design matrix is the fundamental component of any regression model; it is often expressed as the X matrix.

Each observation in the user-specified data table corresponds to a row in the design matrix. The columns of the design matrix are the parameters that are included in a regression model.

The models that PROC MODELMATRIX supports can contain main effects for continuous and classification variables and interaction effects of these variables. The models can also include constructed effects such as splines.

PROC MODELMATRIX Features

The MODELMATRIX procedure has the following main features:

- **Model specification**
  - supports multiple parameterizations for classification effects
  - supports any degree of interaction (crossed effects) and nested effects
  - supports a hierarchy among effects
  - provides a FREQ statement for grouped analysis
  - provides a WEIGHT statement for weighted analysis

- **Display and output**
  - produces a CAS data table that contains the design matrix
  - produces an output data table (OutDesignInfo) that names the parameters (columns) of the design matrix
  - produces output data tables that provide information about the number of observations read (NObs), model information (ModelInfo), dimensions information (Dimensions), class levelization details (ClassInfo), and effect details
  - supports custom prefixing for the names of the columns of the design matrix along with a default value (Col) if the prefix is not specified

Because the MODELMATRIX procedure runs in the CAS environment, it also does the following:

- enables you to run on a cluster of machines that distribute the data and the computations
- enables you to run in single-machine mode on CAS
- exploits all the available cores and concurrent threads. For information about how PROC MODELMATRIX uses threads, see the section “Multithreading” on page 81 in Chapter 3, “Shared Concepts.”
PROC MODELMATRIX Compared with Other SAS Procedures

The MODELMATRIX procedure is the counterpart of the GLMMOD procedure (in SAS/STAT) for a CAS-enabled environment. It creates a design matrix from explanatory variables in a regression model. A number of other procedures, including the LOGISTIC and GLMSELECT procedures in SAS/STAT, enable you to create the design matrix as a by-product of other statistical analysis.

PROC MODELMATRIX produces a global design matrix across all BY groups. In other words, the columns of the design matrix represent the union of the parameters across all BY groups. The elements of any design row that do not belong to a BY group are replaced with missing values.

PROC MODELMATRIX Compared with the LOGISTIC Procedure

The functionality of the MODELMATRIX procedure closely resembles that of the LOGISTIC procedure when you specify the OUTDESIGNONLY option in the PROC LOGISTIC statement. The only difference between the results of the MODELMATRIX and LOGISTIC procedures is in the way invalid and missing variables in the FREQ and WEIGHT statements are treated. In PROC MODELMATRIX, rows that have invalid values (< 1 for the FREQ statement or ≤ 0 for the WEIGHT statement) are filled in with missing values. Observations that have missing covariates are treated in the same way. In short, all the columns of invalid observations are filled in with missing values. In contrast, PROC LOGISTIC does not treat invalid observations differently. If data are missing for a variable (including FREQ and WEIGHT variables), only the corresponding column (in the case of interval variables) or columns (in the case of categorical variables) are filled in with missing values. In addition, PROC LOGISTIC uses the EFFECT encoding for parameterization of the categorical variables by default, whereas PROC MODELMATRIX uses the GLM encoding.

Using CAS Sessions and CAS Engine Librefs

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:

```sas
proc options option=(CASHOST CASPORT);
run;
```

In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:

```sas
cas mysess;
libname mycas cas sessref=mysess;
```

The CAS statement creates the CAS session named mysess, and the LIBNAME statement creates the mycas CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the
CASHOST and CASPORT of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the mysess session, you can terminate it by using the TERMINATE option in the CAS statement as follows:

```cas mysess terminate;
```

For more information about the CAS and LIBNAME statements, see the section “Introduction to Shared Concepts” on page 10 in Chapter 3, “Shared Concepts.”

---

**Getting Started: MODELMATRIX Procedure**

The Sashelp.Baseball data set contains salary and performance information for Major League Baseball players, excluding pitchers, who played at least one game in both the 1986 and 1987 seasons. The salaries (*Sports Illustrated*, April 20, 1987) are from the 1987 season, and the performance measures are from 1986 (*Collier Books, The 1987 Baseball Encyclopedia Update*). The following step displays (in Figure 13.1) the variables in the data set that are used in this example:

```proc contents varnum data=sashelp.baseball
    (KEEP = League Division nHits yrMajor);
    ods select position;
run;
```

![Figure 13.1 Sashelp.Baseball Data Set](image)

The CONTENTS Procedure

<table>
<thead>
<tr>
<th>#</th>
<th>Variable</th>
<th>Type</th>
<th>Len</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>nHits</td>
<td>Num</td>
<td>8</td>
<td>Hits in 1986</td>
</tr>
<tr>
<td>9</td>
<td>YrMajor</td>
<td>Num</td>
<td>8</td>
<td>Years in the Major Leagues</td>
</tr>
<tr>
<td>16</td>
<td>League</td>
<td>Char</td>
<td>8</td>
<td>League at the End of 1986</td>
</tr>
<tr>
<td>17</td>
<td>Division</td>
<td>Char</td>
<td>8</td>
<td>Division at the End of 1986</td>
</tr>
</tbody>
</table>

Assume that you want to predict the logSalary by using the categorical variables League and Division and the interaction of the nHits and YrMajor variables. The goal is to create a design matrix that represents this regression model. This example shows how you can use PROC MODELMATRIX for this purpose. Because the variation in salaries is much greater for the higher salaries, a log transformation is first applied to the salaries.

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

You can load the Sashelp.Baseball data set into your CAS session by using your CAS engine libref with the following DATA step:
data mycas.baseball;
  set sashelp.baseball;
  index = _N_; 
run;
These statements assume that your CAS engine libref is named mycas, as in the section “Using CAS Sessions and CAS Engine Librefs” on page 653, but you can substitute any appropriately defined CAS engine libref.

The following statements define the described model, create the design matrix, and store the design matrix in the CAS table mycas.designMat:

    proc modelmatrix data=mycas.baseball;
      class League Division;
      model logSalary = League Division nHits*yrMajor;
      output out = mycas.designMat copyvar=index;
run;

The output from this analysis is presented in Figure 13.2 through Figure 13.4.

**Figure 13.2** Number of Observations, Dimensions, and Column Names of the Design Matrix

<table>
<thead>
<tr>
<th>The MODELMATRIX Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Observations Read</td>
</tr>
<tr>
<td>Number of Observations Used</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dimensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Effects</td>
</tr>
<tr>
<td>Number of Parameters</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Column Names of the Design Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name Parameter</td>
</tr>
<tr>
<td>Col1</td>
</tr>
<tr>
<td>Col2</td>
</tr>
<tr>
<td>Col3</td>
</tr>
<tr>
<td>Col4</td>
</tr>
<tr>
<td>Col5</td>
</tr>
<tr>
<td>Col6</td>
</tr>
</tbody>
</table>

**Figure 13.2** displays the “Number of Observations,” “Dimensions,” and the “Column Names of the Design Matrix” tables. The “Number of Observations” table shows that only 263 of the 322 observations in the input data are used, because some of the observations contain missing data. When you specify effects that contain classification variables, the number of parameters is usually larger than the number of effects. The “Dimensions” table shows the number of effects and the number of parameters that are considered.
The “Column Names of the Design Matrix” table lists all the parameters that are included in the design matrix and indicates the column name that is associated with each parameter. As presented in Figure 13.2, the model includes a total of six parameters (two categorical variables with two levels each, one interaction variable, and the intercept), and therefore the design matrix has six columns. Because the OUTPUT statement does not specify a PREFIX= option, the default prefix (col) is used for the column names. For more information about how to change the column names, see the PREFIX= option and Examples.

The “Model Information” table in Figure 13.3 shows the main components of the model.

**Figure 13.3 Model Information Table**

**The MODELMATRIX Procedure**

<table>
<thead>
<tr>
<th>Model Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source</td>
</tr>
<tr>
<td>Response Variable</td>
</tr>
</tbody>
</table>

To be able to inspect the created design matrix, you first need to transfer the output data table (mycas.designMat) from the computing cluster to a local data table (designMat) and then use the PRINT procedure. Note that PROC MODELMATRIX does not preserve the order of the rows in the original data table. Therefore the design matrix that is produced is sorted before printing. The following statements show how you can use a DATA step along with a PROC PRINT step to display the first 15 rows of the transferred design matrix:

```plaintext
data designMat;
  set mycas.designMat;
run;

proc sort data=designMat;
  by index;
run;

proc print data = designMat(obs=15 drop=index) label;
  label COL1='Intercept'
  COL2='League American'
  COL3='League National'
  COL4='Division East'
  COL5='Division West'
  COL6='nHits * YrMajor';
run;
```
Figure 13.4 First 15 Rows of the Design Matrix Stored in the designMat Data Table

<table>
<thead>
<tr>
<th>Obs</th>
<th>League</th>
<th>Division</th>
<th>nHits</th>
<th>YrMajor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>14</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Syntax: MODELMATRIX Procedure

The following statements are available in the MODELMATRIX procedure:

```
PROC MODELMATRIX <options>;
   BY variables;
   CLASS variable <(options)> . . . <variable <(options)>> < / global-options> ;
   DISPLAY <table-list> < / options> ;
   DISPLAYOUT table-spec-list < / options> ;
   EFFECT name = effect-type (variables < / options>) ;
   FREQ variable;
   MODEL dependent = <effects> < / model-options> ;
   OUTPUT OUT= CAS-libref.data-table <options> ;
   WEIGHT variable;
```

The PROC MODELMATRIX statement, a single MODEL statement, and a single OUTPUT statement are required. All other statements are optional.

The CLASS statement can appear multiple times. If a CLASS statement is specified, it must precede the MODEL statement.

The rest of this section provides detailed syntax information about each of the preceding statements, beginning with the PROC MODELMATRIX statement. The remaining statements are described in alphabetical order.
PROC MODELMATRIX Statement

PROC MODELMATRIX <options>;

The PROC MODELMATRIX statement invokes the procedure. Table 13.1 summarizes the options in the PROC MODELMATRIX statement by function.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Options</td>
<td></td>
</tr>
<tr>
<td>DATA=</td>
<td>Specifies the input data table</td>
</tr>
</tbody>
</table>

You can specify the following options:

DATA=CAS-libref.data-table

names the input data table for PROC MODELMATRIX to use. The default is the most recently created data table. CAS-libref.data-table is a two-level name, where

CAS-libref refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to the data, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about CAS-libref, see the section “Using CAS Sessions and CAS Engine Librefs” on page 653.

data-table specifies the name of the input data table.

BY Statement

BY variables;

You can specify a BY statement in PROC MODELMATRIX to obtain separate analyses of observations in groups that are defined by the values of the BY variables. If you specify more than one BY statement, only the last one specified is used. For more information, see the discussion of BY-group processing in SAS Language Reference: Concepts.

CLASS Statement

CLASS variable <(options)> . . . <variable <(options)>> / global-options;

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. Table 13.2 summarizes the values that you can use for either an option or a global-option. The options are fully documented in the section “CLASS Statement” on page 12 in Chapter 3, “Shared Concepts.”
Table 13.2  CLASS Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESCENDING</td>
<td>Reverses the sort order</td>
</tr>
<tr>
<td>MISSING</td>
<td>Treats missing values as valid levels</td>
</tr>
<tr>
<td>ORDER=</td>
<td>Specifies the sort order for the levels</td>
</tr>
<tr>
<td>PARAM=</td>
<td>Specifies the parameterization of the variable</td>
</tr>
<tr>
<td>REF=</td>
<td>Specifies the reference level of the variable</td>
</tr>
<tr>
<td>SPLIT</td>
<td>Treats each column of the design matrix as a separate effect</td>
</tr>
</tbody>
</table>

**DISPLAY Statement**

`DISPLAY < table-list > < / options > ;`

The DISPLAY statement enables you to specify a list of display tables to display or exclude. This statement is similar to the ODS SELECT, ODS EXCLUDE, and ODS TRACE statements. However, the DISPLAY statement can improve performance when a large number of tables could be generated (such as in BY-group processing). The procedure processes the DISPLAY statement on a CAS server and thus sends only a subset of ODS tables to the SAS client. The ODS statements are all first processed on a SAS client, then a subset of the produced tables is displayed.

If you use both DISPLAY and ODS statements together, the DISPLAY statement takes precedence over the ODS statements. Note that the ODS EXCLUDE statement processes tables that are sent to the client after they have been filtered by the DISPLAY statement. In some cases, it might appear that the ODS EXCLUDE statement is taking precedence because it can further filter the tables. For more information about ODS, see *SAS Output Delivery System: Procedures Guide*.

You can specify the `table-list` as a list of table names, paths, partial pathnames, and regular expressions.

The valid table names are listed in the section “ODS Table Names” on page 665. A path is a table name that is prefixed with dot-separated grouping information. For example, a SelectionSummary table that a procedure produces during a selection routine might have the path `Bygroup1.Summary.SelectionSummary`. A partial pathname does not include all groups; for example, `SelectionSummary` and `Summary.SelectionSummary` are partial pathnames for `Bygroup1.Summary.SelectionSummary`.

When you specify a table name or partial pathname, all display tables whose paths end in the specified name are selected for display or exclusion. For example, both `SelectionSummary` and `Summary.SelectionSummary` select `Bygroup1.Summary.SelectionSummary`.

A regular expression is enclosed in forward slashes (/). For example, specifying “/tions/” selects all pathnames that contain the substring “tions”; in particular, the `Bygroup1.Summary.SelectionSummary` table is selected. Specifying “!/tions!/” selects all pathnames that do not contain the substring “tions”; in particular, the `Bygroup1.Summary.SelectionSummary` table is not selected.

You can specify the following `options` after a slash (/):
CASESENSITIVE
performs a case-sensitive comparison of table names in the table-list to display table names when tables are subsetted for display. To preserve case, you must enclose table names in the table-list in quotation marks.

EXCLUDE
specifies that tables in the table-list are excluded.

EXCLUDEALL
suppresses the display of all tables. This option takes precedence over the other options.

TRACE
displays the display table names, labels, and paths.

DISPLAYOUT Statement

DISPLAYOUT table-spec-list </ options> ;

The DISPLAYOUT statement enables you to create CAS output tables from your displayed output. This statement is similar to the ODS OUTPUT statement. For more information about ODS, see SAS Output Delivery System: Procedures Guide.

The table-spec-list specifies a list of CAS output tables to create. Each entry in the list has either a key=value format or a key format:

key=value specifies key as the ODS table name, path, or partial pathname, and specifies value as the CAS output table name.

key specifies key as the ODS table name and also as the CAS output table name.

The ODS table names that you can specify are listed in the section “ODS Table Names” on page 665. You cannot specify the ODS table named OutputCasTables in the table-spec-list.

Table names and partial pathnames are discussed under the DISPLAY statement. The DISPLAYOUT statement does not support regular expressions.

You can specify the following options after a slash (/):

INCLUDEALL
creates output CAS tables for all display tables. The name of the created output CAS table is the same as the corresponding display table name. If you specify this option, the table-spec-list specification is ignored.

NOREPLACE
does not replace any existing CAS output table of the same name.

REPEATED
replicates all CAS output tables on all nodes.
The EFFECT statement enables you to construct special collections of columns for design matrices. These collections are referred to as constructed effects to distinguish them from the usual model effects that are formed from continuous or classification variables, as discussed in the section “GLM Parameterization of Classification Variables and Effects” on page 54 in Chapter 3, “Shared Concepts.”

You can specify the following effect-types:

- **COLLECTION** specifies a collection effect that defines one or more variables as a single effect that has multiple degrees of freedom. The variables in a collection are considered as a unit for purposes of estimation and inference.
- **MULTIMEMBER | MM** specifies a multimember classification effect whose levels are determined by one or more variables that appear in a CLASS statement.
- **POLYNOMIAL | POLY** specifies a multivariate polynomial effect in the specified numeric variables.
- **SPLINE** specifies a regression spline effect whose columns are univariate spline expansions of one or more variables. A spline expansion replaces the original variable with an expanded or larger set of new variables.

Table 13.3 summarizes the options available in the EFFECT statement.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collection Effects Option</td>
<td></td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays the constituents of the collection effect</td>
</tr>
<tr>
<td>Multimember Effects Options</td>
<td></td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays the levels of the multimember effect</td>
</tr>
<tr>
<td>NOEFFECT</td>
<td>Specifies that observations whose levels are all missing for the multimember variables should have values of 0 in the corresponding design matrix columns</td>
</tr>
<tr>
<td>STDIZE</td>
<td>Standardizes the design matrix entries so that each observation has a sum of 1</td>
</tr>
<tr>
<td>WEIGHT=</td>
<td>Specifies the weight variable for the contributions of each classification effect</td>
</tr>
<tr>
<td>Polynomial Effects Options</td>
<td></td>
</tr>
<tr>
<td>DEGREE=</td>
<td>Specifies the degree of the polynomial</td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays details of the specified polynomial</td>
</tr>
<tr>
<td>MDEGREE=</td>
<td>Specifies the maximum degree of any variable in a term of the polynomial</td>
</tr>
<tr>
<td>NOSEPARATE</td>
<td>Treats the polynomial as a single effect that has multiple degrees of freedom</td>
</tr>
<tr>
<td>STANDARDIZE=</td>
<td>Specifies centering and scaling suboptions for the variables that define the polynomial</td>
</tr>
</tbody>
</table>
### Table 13.3 continued

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Spline Effects Options</strong></td>
<td></td>
</tr>
<tr>
<td>BASIS=</td>
<td>Specifies the type of basis (B-spline basis or truncated power function basis) for the spline effect</td>
</tr>
<tr>
<td>DATABOUNDARY</td>
<td>Uses the extremes of the data as boundary knots for a B-spline basis</td>
</tr>
<tr>
<td>DEGREE=</td>
<td>Specifies the degree of the spline effect</td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays the knots and locations for each spline basis function</td>
</tr>
<tr>
<td>KNOTMAX=</td>
<td>Requests equally spaced right-side boundary knots that start at the variables’ maximum and end at the specified KNOTMAX= value</td>
</tr>
<tr>
<td>KNOTMETHOD=</td>
<td>Specifies how to construct the knots for the spline effect</td>
</tr>
<tr>
<td>KNOTMIN=</td>
<td>Requests equally spaced left-side boundary knots that start at the specified KNOTMIN= value and end at the variables’ minimum value</td>
</tr>
<tr>
<td>NATURALCUBIC</td>
<td>Specifies a natural cubic spline basis for the spline effect</td>
</tr>
<tr>
<td>SEPARATE</td>
<td>Treats the spline basis for each variable as a separate effect when multiple variables are specified</td>
</tr>
<tr>
<td>SPLIT</td>
<td>Treats each design matrix column as a separate effect</td>
</tr>
</tbody>
</table>

For more information about the syntax of these effect-types and how columns of constructed effects are computed, see the section “EFFECT Statement” on page 21 in Chapter 3, “Shared Concepts.”

---

### FREQ Statement

**FREQ** variable ;

The variable in the FREQ statement identifies a numeric variable in the data set that contains the frequency of occurrence of each observation. PROC MODELMATRIX treats each observation as if it appears \( f \) times, where \( f \) is the value of the variable for the observation. If \( f \) is not an integer, it is truncated to an integer. If \( f \) is less than 1 or missing, all the columns of the design matrix for that observation are filled in with missing values. When the FREQ statement is omitted, each observation is assigned a frequency of 1.

---

### MODEL Statement

**MODEL** dependent = < effects > / < model-options> ;

The MODEL statement names the dependent variable and the explanatory effects, including covariates, main effects, interactions, and nested effects. If you omit the explanatory effects, the procedure produces a single column of ones.

After the keyword MODEL, the dependent (response) variable is specified, followed by an equal sign. The explanatory effects follow the equal sign.

For information about constructing the model effects, see the section “Specification and Parameterization of Model Effects” on page 51 in Chapter 3, “Shared Concepts.”
The *model-options* control other aspects of model formation and inference. Table 13.4 summarizes these options.

| Table 13.4  MODEL Statement Options |
|-------------|-----------------------------------|
| **Option**  | **Description**                   |
| Model Options |                                   |
| INFORMATIVE | Models missing values by using extra indicator variables |
| NOINT       | Suppresses the intercept           |

**Model Options**

You can specify the following *model-options* in the MODEL statement after a slash (/):

**INFORMATIVE**
models missing values by using extra model effects. These effects consist of dummy variables that take the value 1 when the value of a continuous model variable involved in the effect is missing, and take the value 0 otherwise. The missing value in the original model effect is replaced by the average value of the effect for the nonmissing values. For continuous-by-class effects, such as A*x, where A is a classification variable and x is a continuous variable, informative missingness creates multiple dummy columns and substitutes the effect mean of x that corresponds to the respective level of A. Missing values for classification variables are treated as valid levels. For more information about informative missingness, see the section “Informative Missingness” on page 78 in Chapter 3, “Shared Concepts.”

**NOINT**
suppresses the intercept term that is otherwise included in the model.

**OUTPUT Statement**

```
OUTPUT OUT=CAS-libref.data-table <options> ;
```

The OUTPUT statement creates a CAS table that contains the design matrix, which is computed after the model parameters are created (and levelized when CLASS variables are included). The variables in the input data table are not included in the output data table, in order to avoid data duplication for large data tables; however, variables that you specify in the **COPYVARS=** option are included.

You must specify the following option:

**OUT=** *CAS-libref.data-table*

names the output data table for PROC MODELMATRIX to use. You must specify this option before any other options. *CAS-libref.data-table* is a two-level name, where

- **CAS-libref** refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to where the data table is to be stored, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about *CAS-libref*, see the section “Using CAS Sessions and CAS Engine Librefs” on page 653.
Chapter 13: The MODELMATRIX Procedure

\textit{data-table} specifies the name of the output data table.

You can also specify the following syntax elements:

\textbf{PREFIX=} \textit{name}

enforces the column names of the design matrix to be a user-defined prefix appended to a sequence of one-based indices in the OUTDESIGN= data set. The default prefix is \textit{Col} and the column name is formed by appending the column number to the prefix, so that by default the columns are named \textit{Col1}, \textit{Col2}, and so on.

\textbf{COPYVAR=} \textit{variable}

\textbf{COPYVARS=} (\textit{variables})

transfers one or more \textit{variables} from the input data table to the output data table and appends them to the right end of the design matrix in the specified order.

\underline{WEIGHT Statement}

\textbf{WEIGHT} \textit{variable} ;

The \textit{variable} in the WEIGHT statement is used as a weight. Observations that have nonpositive or missing weights are considered missing in the produced design matrix. If a WEIGHT statement is not included, all observations that are used are assigned a weight of 1.

\underline{Details: MODELMATRIX Procedure}

\underline{Computational Method}

\textbf{Multithreading}

The MODELMATRIX procedure uses the parallel and multithreaded capabilities of the host environment. When the input data table is distributed across multiple nodes, the resulting design matrix is written in a distributed fashion. The user can specify the number of threads. For more information about how PROC MODELMATRIX uses threads, see the section “Multithreading” on page 81 in Chapter 3, “Shared Concepts.”

\underline{Displayed Output}

The following sections describe the output that PROC MODELMATRIX produces. The output is organized into various tables, which are discussed in their order of appearance.

\textbf{Column Names of the Design Matrix}

The “OutDesignInfo” table displays the column names (valid SAS names) that are assigned to the model parameters (which can include complex parameters from interaction effects and levelization of the classification variables). Unless the user specifies a different value for the \textbf{PREFIX=} option, the default prefix (\textit{Col}) is
used for naming the columns; hence the column names are Col1, Col2, and so on. The “OutDesignInfo” table lists the names of the columns in the design matrix that correspond to each model parameter.

**Number of Observations**

The “Number of Observations” table displays the number of observations that are read from the input data table and the number of observations that are used in constructing the design matrix. If you specify a FREQ statement, this table displays the sum of frequencies that are read and used. If you specify a WEIGHT statement, the table displays the sum of \( f_i w_i \) that are read and used.

**Class Level Information**

The “Class Level Information” table lists the levels of every variable that you specify in the CLASS statement. You can adjust the order of the CLASS variable levels by specifying the ORDER= option in the CLASS statement.

If the classification variables use a nonsingular parameterization, the “Class Level Information” table also displays the reference value for each variable. This table is produced for each BY group separately.

**Dimensions**

The “Dimensions” table displays the number of effects and the number of parameters for each BY group.

**Model Information**

The “Model Information” table lists a summary of the model variables. It includes the data source name and the name of the response variables. If you specify the FREQ and WEIGHT statements, the names of the FREQ and WEIGHT variables are displayed in the “Model Information” table.

**Effect Details**

If you specify the DETAILS option in the EFFECT statement, a table is displayed that lists all constructed effects in the model.

**ODS Table Names**

Each table that the MODELMATRIX procedure creates has a name associated with it. You must use this name to refer to the table when you use ODS statements. These names are listed in Table 13.5.

**NOTE:** The EFFECT statement also create tables which are not listed in this section. For information about these tables, see “ODS Table Names” on page 35 in Chapter 3, “Shared Concepts.”

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClassInfo</td>
<td>Level information from the CLASS statement</td>
<td>CLASS</td>
<td>Default</td>
</tr>
</tbody>
</table>

Table 13.5 ODS Tables Produced by PROC MODELMATRIX
Table 13.5 continued

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dimensions</td>
<td>Model dimensions</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>ModelInfo</td>
<td>Information about the modeling environment</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>NObs</td>
<td>Number of observations read and used</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>OutDesignInfo</td>
<td>Column names of the design matrix</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>OutputCASTables</td>
<td>A special ODS table that has information about all the CAS tables that are created during a CAS action execution</td>
<td>OUTPUT</td>
<td>DISPLAYOUT</td>
</tr>
<tr>
<td>Timing</td>
<td>Absolute and relative times for tasks performed by the procedure</td>
<td>Default</td>
<td></td>
</tr>
</tbody>
</table>

Examples: MODELMATRIX Procedure

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

Example 13.1: Spline Effects and BY-Group Processing

This example showcases several features of the MODELMATRIX procedure. It demonstrates the use of classification variables in a MODEL statement along with the use of interaction and SPLINE effects in a MODEL statement. In addition, it demonstrates how the procedure handles BY-group processing for a particular model.

The following DATA step produces analysis data that contain a variable that has 20 observations. These statements assume that your CAS engine libref is named mycas, as in the section “Using CAS Sessions and CAS Engine Librefs” on page 653, but you can substitute any appropriately defined CAS engine libref.

data mycas.sample;
    call streaminit(1);
    do byVar=1 to 2;
        do i=1 to 10;
            if byVar=1 then c=rand("Integer",0,3);
            else if byVar=2 then c=rand("Integer",0,1);
            x=rand("Normal");
            y=1;
            wgt = 1;
            if i=5 and byVar=2 then frq = -1; else frq=1;
            index + 1;
        end;
    end;
end;
The preceding DATA step generates a data table (mycas.sample) that contains 20 observations. It has two continuous variables (x and y) and a single classification variable (c), which can have a maximum of four unique values for 10 observations and a maximum of two unique values for the rest of the observations. Also, a BY variable named byVar is defined that divides the data table into two equally sized BY groups, each of which has 10 observations. Also included are the frq and wgt variables that are going to be used as the FREQ and WEIGHT variables, respectively. For simplicity, all the WEIGHT variables are assigned a value of 1. The same applies to FREQ variables, with the exception of the observation where INDEX=15, which has an invalid FREQ value (that is, –1). This invalid FREQ value demonstrates how the MODELMATRIX procedure treats observations that have invalid FREQ values.

The following DATA step and PROC PRINT statement show the contents of the mycas.sample data table. The output of these statements is displayed in Output 13.1.1.

```
data sample;
  set mycas.sample;
run;

proc print data=sample;
run;
```

**Output 13.1.1** Contents of the mycas.sample Data Table

<table>
<thead>
<tr>
<th>Obs</th>
<th>byVar</th>
<th>i</th>
<th>c</th>
<th>x</th>
<th>y</th>
<th>wgt</th>
<th>frq</th>
<th>index</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>0.46319</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>0.00741</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>-0.06703</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>1.52070</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>0.51495</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>6</td>
<td>3</td>
<td>1.00574</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>7</td>
<td>3</td>
<td>0.36049</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>8</td>
<td>3</td>
<td>-1.75802</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>9</td>
<td>2</td>
<td>1.02322</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>10</td>
<td>1</td>
<td>0.33004</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>11</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1.48699</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>12</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>-0.31012</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>13</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0.60637</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>13</td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td>4</td>
<td>0</td>
<td>1.96293</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>14</td>
</tr>
<tr>
<td>15</td>
<td>2</td>
<td>5</td>
<td>0</td>
<td>-1.21157</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>15</td>
</tr>
<tr>
<td>16</td>
<td>2</td>
<td>6</td>
<td>1</td>
<td>-1.22244</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>16</td>
</tr>
<tr>
<td>17</td>
<td>2</td>
<td>7</td>
<td>1</td>
<td>0.53368</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>17</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>8</td>
<td>1</td>
<td>1.27392</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>18</td>
</tr>
<tr>
<td>19</td>
<td>2</td>
<td>9</td>
<td>0</td>
<td>-0.99679</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>19</td>
</tr>
<tr>
<td>20</td>
<td>2</td>
<td>10</td>
<td>0</td>
<td>-1.55454</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>20</td>
</tr>
</tbody>
</table>

To demonstrate the capabilities of the MODELMATRIX procedure, the following code includes BY-group processing, CLASS variables, the interaction effect, and the SPLINE effect. It also demonstrates how you can
change the name of the output data table by using the PREFIX= option. Moreover, the OUTPUT statement enables you to include the input data table variables as additional columns in the output data table by using the COPYVARS= option.

```sas
proc modelmatrix data=mycas.sample nthreads=1;
   by byVar;
   class c;
   freq frq;
   weight wgt;
   effect spl= spline(x);
   model y = x c spl x*c;
   output out=mycas.designMat prefix=param copyVars=(c frq index);
run;
```

The output tables are displayed in Output 13.1.2, Output 13.1.3, and Output 13.1.4.

**Output 13.1.2** Model Information and Column Names of the Design Matrix

---

The “Column Names of the Design Matrix” table in Output 13.1.2 shows that the defined model creates a total of 17 parameters across the two BY groups. Therefore, the design matrix has 17+3 columns (the additional columns are reserved for the variables c, frq, and index that you append to the design matrix by using the COPYVARS= option).
Apart from the intercept, the continuous variable $x$ is included as a main effect. However, because $c$ is defined as a classification variable, all of its unique values (0–3) are considered to be separate parameters (a total of four). Output 13.1.2 shows that the SPLINE effect adds seven more parameters to the model. Finally, the interaction of $x$ and $c$ ($x \times c$) adds four more parameters.

Also shown in Output 13.1.2 is the “Model Information” table, which displays a brief summary of the response variable, the FREQ variable, and the WEIGHT variable.

The rest of the output tables depend on the BY groups. Therefore, they are produced separately and are shown in Output 13.1.3 and Output 13.1.4.

**Output 13.1.3** Number of Observations, Class Level Information, and Dimensions Tables for BY Group 2

The `MODELMATRIX` Procedure

<table>
<thead>
<tr>
<th>byVar=1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Observations Read: 10</td>
</tr>
<tr>
<td>Number of Observations Used: 10</td>
</tr>
<tr>
<td>Sum of Frequencies Read: 10</td>
</tr>
<tr>
<td>Sum of Frequencies Used: 10</td>
</tr>
<tr>
<td>Sum of Weights Read: 10</td>
</tr>
<tr>
<td>Sum of Weights Used: 10</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>byVar=1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class Level Information</td>
</tr>
<tr>
<td>Class Levels Values</td>
</tr>
<tr>
<td>$c$</td>
</tr>
<tr>
<td>3 \text{,} 1 \text{,} 2 \text{,} 3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>byVar=1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dimensions</td>
</tr>
<tr>
<td>Number of Effects: 5</td>
</tr>
<tr>
<td>Number of Parameters: 15</td>
</tr>
</tbody>
</table>

**Output 13.1.4** Number of Observations, Class Level Information, and Dimensions Tables for BY Group 2

The `MODELMATRIX` Procedure

<table>
<thead>
<tr>
<th>byVar=2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Observations Read: 10</td>
</tr>
<tr>
<td>Number of Observations Used: 9</td>
</tr>
<tr>
<td>Sum of Frequencies Read: 9</td>
</tr>
<tr>
<td>Sum of Frequencies Used: 9</td>
</tr>
<tr>
<td>Sum of Weights Read: 9</td>
</tr>
<tr>
<td>Sum of Weights Used: 9</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>byVar=2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class Level Information</td>
</tr>
<tr>
<td>Class Levels Values</td>
</tr>
<tr>
<td>$c$</td>
</tr>
<tr>
<td>2 \text{,} 0 \text{,} 1</td>
</tr>
</tbody>
</table>
PROC MODELMATRIX displays the number of observations that are read and used. Because the FREQ and WEIGHT variables are defined, the procedure displays the number of FREQ and WEIGHT variables that are read and used. You can see that BY group 2 (Output 13.1.4) has one invalid observation (the observation where INDEX=15). Therefore, the number of FREQ variables that are used is 9, which is different from BY group 1, where all FREQ variables have a value of 1.

The “Dimensions” table shows the number of effects and parameters that are considered within each BY group. Notice that the number of considered effects for both BY groups is equal to five. However, neither BY group includes all the parameters listed in the OutDesignInfo table, so for all BY groups, the number of parameters is less than 17.

This fact can be seen in the “Class Level Information” tables shown in Output 13.1.3 and Output 13.1.4. These tables show the levelization of the classification variables within each BY group. You can see that none of the BY groups include all levels of c (that is, 0, 1, 2, and 3), which is why the number of parameters is less than 17 for all BY groups.

PROC MODELMATRIX constructs a global design matrix. Therefore, it constructs a table that includes the union of all parameters in each BY group. To inspect the design matrix that resides on the CAS server as a CAS table, follow the same steps that are given in the previous example. The constructed table with 20 rows is sent to the client by using the following statements. The output is displayed in Output 13.1.5.

```plaintext
data work.designMat;
  set mycas.designMat;
run;

proc print data=designMat;
  format BEST 5. param1-param17;
  var index c frq param1-param7 param17;
  where
    (index < 11) or
    (index = 15);
run;
```
Output 13.1.5  Rows with index Values of 1–10 and Row with index Value of 15 from Design Matrix

<table>
<thead>
<tr>
<th>Obs</th>
<th>index</th>
<th>c</th>
<th>frq</th>
<th>param1</th>
<th>param2</th>
<th>param3</th>
<th>param4</th>
<th>param5</th>
<th>param6</th>
<th>param7</th>
<th>param17</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>0.46319</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>.00000</td>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0.00741</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>.00000</td>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>-0.06703</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>.00000</td>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>1.52070</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>.00000</td>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>2</td>
<td>1</td>
<td>0.51495</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>.00000</td>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>3</td>
<td>1</td>
<td>1.00574</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>.00000</td>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>3</td>
<td>1</td>
<td>0.36049</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>.00000</td>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>3</td>
<td>1</td>
<td>-1.75802</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>.16667</td>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>2</td>
<td>1</td>
<td>1.02322</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>.00000</td>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>1</td>
<td>1</td>
<td>0.33004</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>.00000</td>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>15</td>
<td>0</td>
<td>-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>.</td>
</tr>
</tbody>
</table>

Output 13.1.5 displays rows of the design matrix whose index value is 1–10 and the row whose index value is 15. Only the index, c, frq, param1–param7, and param17 variables are shown. As expected, the generated data table contains 20 rows associated with each observation and 20 columns (17 parameters plus three columns for the input variables c, frq, and index). You can see that the column names are param1 through param17, based on the PREFIX= option specified in the OUTPUT statement.

The “Column Names of the Design Matrix” table lists the model parameters with their corresponding column names in the design matrix. Columns param3 through param6 contain all levels of the c variable. The rows in the design matrix whose index value is less than 11 belong to the first BY group (byVar=1), and according to the “Class Information” table, level c3 does not exist in either BY group. This observation can be confirmed by the fact that the param6 and param17 columns of the design matrix (which are associated with the c3 and x * c3 parameters, respectively) are empty.

You can also see that the row of the design matrix whose index value is 15 is all blanks. This is because the value of the defined FREQ variable for this observation is invalid and equal to −1 (see the frq column in the design matrix), and therefore this observation is not considered when the design matrix is created.

References


Chapter 14
The NLMOD Procedure

Contents

Overview: NLMOD Procedure ................................................................. 674
PROC NLMOD Features ................................................................. 674
PROC NLMOD Compared with the Other SAS Procedures ......................... 674
  PROC NLMOD Compared with the HPNLMOD Procedure ................. 675
  PROC NLMOD Compared with the NLIN Procedure .................... 675
  PROC NLMOD Compared with the NLMIXED Procedure .......... 675
Using CAS Sessions and CAS Engine Librefs .................................. 675
Getting Started: NLMOD Procedure .................................................. 676
  Least Squares Model ......................................................... 676
  Binomial Model .............................................................. 677
Syntax: NLMOD Procedure ................................................................. 679
  PROC NLMOD Statement ................................................. 680
  BOUNDS Statement ...................................................... 682
  BY Statement ............................................................... 682
  DISPLAY Statement ..................................................... 682
  DISPLAYOUT Statement .................................................. 683
  ESTIMATE Statement .................................................... 684
  ID Statement ............................................................... 685
  MODEL Statement ......................................................... 685
  PARAMETERS Statement .................................................. 685
  PREDICT Statement ....................................................... 688
  RESTRICT Statement ....................................................... 689
Programming Statements ................................................................. 690
Details: NLMOD Procedure ................................................................. 691
  Least Squares Estimation .................................................. 691
  Built-In Log-Likelihood Functions .................................... 692
  Multithreading ............................................................... 694
  Optimization Algorithms ................................................. 694
  Displayed Output .......................................................... 695
  ODS Table Names ........................................................... 697
Examples: NLMOD Procedure ............................................................. 699
  Example 14.1: Segmented Model ......................................... 699
References ..................................................................................... 702
Overview: NLMOD Procedure

The NLMOD procedure fits nonlinear regression models with standard or general distributions in SAS Viya. PROC NLMOD enables you to specify the model by using SAS programming statements, which give you greater flexibility in modeling the relationship between the response variable and independent (regressor) variables than SAS procedures that use a more structured MODEL statement. The NLMOD procedure uses either nonlinear least squares or maximum likelihood to fit nonlinear regression models.

PROC NLMOD Features

The NLMOD procedure does the following:

- is highly multithreaded during all phases of analytic execution
- computes analytical derivatives of user-provided expressions for more robust parameter estimations
- evaluates user-provided expressions and their confidence limits by using the ESTIMATE and PREDICT statements
- estimates parameters without specifying a particular distribution function by using the least squares method
- estimates parameters by using the maximum likelihood method when either a built-in distribution function is specified or a likelihood function is provided

Because the NLMOD procedure runs on CAS, it also does the following:

- enables you to run in distributed mode on a cluster of machines that distribute the data and the computations
- enables you to run in single-machine mode on CAS
- exploits all the available cores and concurrent threads, regardless of execution mode. For information about how PROC NLMOD uses threads, see the section “Multithreading” on page 81 in Chapter 3, “Shared Concepts.”

PROC NLMOD Compared with the Other SAS Procedures

The NLMOD procedure provides nonlinear regression modeling functionality that is comparable to that of the HPNLMOD, NLIN, and NLMIXED procedures in SAS/STAT software.
**PROC NLMOD Compared with the HPNLMOD Procedure**

The functionality of the NLMOD procedure is identical to the HPNLMOD procedure, which is a high-performance procedure. The NLMOD procedure is the next generation of the HPNLMOD procedure, and it was developed specifically for SAS Viya. Both procedures are designed to run on a cluster of machines that distribute the data and the computations. Both procedures perform computations in multiple threads.

Both the NLMOD and HPNLMOD procedures fit nonlinear regression models.

**PROC NLMOD Compared with the NLIN Procedure**

Like the NLIN procedure in SAS/STAT software, the NLMOD procedure estimates parameters by using least squares minimization for models that are specified by SAS programming statements. However, PROC NLMOD can also perform maximum likelihood estimation when information about the response variable’s distribution is provided. PROC NLMOD also has a RESTRICT statement for specifying restrictions on parameter estimates that are more general than those available in PROC NLIN. Because the NLMOD and NLIN procedures use different optimization techniques, the available options that control the estimation process and resulting parameter estimates can differ between these procedures when equivalent models and data are analyzed.

**PROC NLMOD Compared with the NLMIXED Procedure**

Although it does not support the specification of random effects, PROC NLMOD is similar to the NLMIXED procedure in SAS/STAT. Both procedures perform maximum likelihood estimation by using the same programming syntax and set of distributions to specify the model’s mean term. In addition, both PROC NLMOD and PROC NLMIXED use the same optimization techniques and options. However, PROC NLMIXED does not support least squares parameter estimation.

**Using CAS Sessions and CAS Engine Librefs**

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:

```sas
proc options option=(CASHOST CASPORT);
run;
```

In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:

```sas
cas mysess;
libname mycas cas sessref=mysess;
```
The CAS statement creates the CAS session named mysess, and the LIBNAME statement creates the mycas CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the CASHOST and CASPORT of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the mysess session, you can terminate it by using the TERMINATE option in the CAS statement as follows:

```cas mysess terminate;```

For more information about the CAS and LIBNAME statements, see the section “Introduction to Shared Concepts” on page 10 in Chapter 3, “Shared Concepts.”

---

**Getting Started: NLMOD Procedure**

The most common use of the NLMOD procedure is to estimate the parameters in a model in which the response variable is a nonlinear function of one or more of the parameters.

**Least Squares Model**

The Michaelis-Menten model of enzyme kinetics (Ratkowsky 1990, p. 59) relates a substrate’s concentration to its catalyzed reaction rate. You can analyze the Michaelis-Menten model by using a least squares estimation because it does not specify how the reaction rate is distributed around its predicted value. The relationship between reaction rate and substrate concentration is

\[
 f(x, \theta) = \frac{\theta_1 x_i}{\theta_2 + x_i}, \quad \text{for } i = 1, 2, \ldots, n
\]

where \( x_i \) represents the concentration for \( n \) trials and \( f(x, \theta) \) is the reaction rate. The vector \( \theta \) contains the rate parameters.

The input data must be a table on your CAS server, and a CAS session must be set up. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.” The following DATA step creates the Enzyme data table, which consists of experimental measurements of reaction rate and concentration, in your CAS session:

```data mycas.Enzyme;
  input conc rate @@;
datalines;
  0.26 124.7 0.30 126.9
  0.48 135.9 0.50 137.6
  0.54 139.6 0.68 141.1
  0.82 142.8 1.14 147.6
  1.28 149.8 1.38 149.4
  1.80 153.9 2.30 152.5
  2.44 154.5 2.48 154.7
;```
These statements assume that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

The following SAS statements estimate the parameters $\theta_1$ and $\theta_2$:

```sas
proc nlmod data=mycas.Enzyme;
  parms theta1=0 theta2=0;
  model rate ~ residual(theta1*conc / (theta2 + conc));
run;
```

The least squares estimation that PROC NLMOD performs for this enzyme kinetics problem produces the analysis of variance table in Figure 14.1. The table displays the degrees of freedom, sums of squares, and mean squares along with the model $F$ test.

**Figure 14.1** Nonlinear Least Squares Analysis of Variance

<table>
<thead>
<tr>
<th>Source</th>
<th>DF</th>
<th>Sum of Squares</th>
<th>Mean Square</th>
<th>F Value</th>
<th>Approx Pr &gt; F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>2</td>
<td>290116</td>
<td>145058</td>
<td>88537.2</td>
<td>&lt;.0001</td>
</tr>
<tr>
<td>Error</td>
<td>12</td>
<td>19.6606</td>
<td>1.6384</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Uncorrected Total</td>
<td>14</td>
<td>290135</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*An intercept was not specified for this model.*

Finally, Figure 14.2 displays the parameter estimates, standard errors, $t$ statistics, and 95% confidence intervals for $\theta_1$ and $\theta_2$.

**Figure 14.2** Parameter Estimates and Approximate 95% Confidence Intervals

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Estimate</th>
<th>Standard Error</th>
<th>DF</th>
<th>t Value</th>
<th>Approx Pr &gt;</th>
<th>t</th>
<th>Lower</th>
<th>Upper</th>
</tr>
</thead>
<tbody>
<tr>
<td>theta1</td>
<td>158.1</td>
<td>0.6737</td>
<td>12</td>
<td>234.67</td>
<td>&lt;.0001</td>
<td>156.6</td>
<td>159.6</td>
<td></td>
</tr>
<tr>
<td>theta2</td>
<td>0.0741</td>
<td>0.00313</td>
<td>12</td>
<td>23.69</td>
<td>&lt;.0001</td>
<td>0.0673</td>
<td>0.0809</td>
<td></td>
</tr>
</tbody>
</table>

In the enzyme kinetics model, no information was supplied about the distribution of the reaction rate around the model’s mean value. Therefore, the residual model distribution was specified to perform a least squares parameter fit.

---

**Binomial Model**

In the example “Probit Model with Likelihood Function” in the chapter “The NLIN Procedure” in *SAS/STAT User’s Guide*, cancer remission is modeled by expressing the maximum likelihood function for a binary distribution as a nonlinear least squares optimization. The following statements show an equivalent formulation of this model that uses PROC NLMOD and specifies the binary distribution explicitly:
data mycas.Remiss;
  input remiss cell smear infil li blast temp;
  label remiss = 'complete remission';
  like = 0;
  label like = 'dummy variable for nlin';
  datalines;
1 0.8 .83 .66 1.9 1.10 .996
1 0.9 .36 .32 1.4 0.74 .992
0 0.8 .88 .70 0.8 0.176 .982
0 1 .87 .87 0.7 1.053 .986
1 0.9 .75 .68 1.3 0.519 .980
0 1 .65 .65 0.6 0.519 .982
1 0.95 .97 .92 1 1.23 .992
0 0.95 .87 .83 1.9 1.354 1.020
0 1 .45 .45 0.8 0.322 .999
0 0.95 .36 .34 0.5 0 1.038
0 0.85 .39 .33 0.7 0.279 .988
0 0.7 .76 .53 1.2 0.146 .982
0 0.8 .46 .37 0.4 0.38 1.006
0 0.2 .39 .08 0.8 0.114 .990
0 1 .90 .90 1.1 1.037 .990
1 1 .84 .84 1.9 2.064 1.020
0 0.65 .42 .27 0.5 0.114 1.014
0 1 .75 .75 1 1.322 1.004
0 0.5 .44 .22 0.6 0.114 .990
1 1 .63 .63 1.1 1.072 .986
0 1 .33 .33 0.4 0.176 1.010
0 0.9 .93 .84 0.6 1.591 1.020
1 1 .58 .58 1 0.531 1.002
0 0.95 .32 .30 1.6 0.886 .988
1 1 .60 .60 1.7 0.964 .990
1 1 .69 .69 0.9 0.398 .986
0 1 .73 .73 0.7 0.398 .986
;
proc nlmod data=mycas.Remiss corr;
  parms int=-10 a=-2 b=-1 c=6;
  linp = int + a*cell + b*li + c*temp;
  p = probnorm(linp);
  model remiss ~ binary(1-p);
run;

This binary distribution model displays information about the quality of the estimation that is different from the information displayed in the section “Least Squares Model” on page 676. No analysis of variance table is produced for this model; fit statistics based on the value of the likelihood function are displayed in Figure 14.3.
Parameter estimates for the binary distribution model that uses the same quantities as in the section “Least Squares Model” on page 676 are displayed in Figure 14.4.

**Figure 14.4** Parameter Estimates and Approximate 95% Confidence Intervals

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Estimate</th>
<th>Standard Error</th>
<th>DF</th>
<th>t Value</th>
<th>Approx Pr &gt;</th>
<th>Lower</th>
<th>Upper</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>-36.7548</td>
<td>32.3607</td>
<td>27</td>
<td>-1.14</td>
<td>0.2660</td>
<td>-103.2</td>
<td>29.6439</td>
</tr>
<tr>
<td>a</td>
<td>-5.6298</td>
<td>4.6376</td>
<td>27</td>
<td>-1.21</td>
<td>0.2353</td>
<td>-15.1454</td>
<td>3.8858</td>
</tr>
<tr>
<td>b</td>
<td>-2.2513</td>
<td>0.9790</td>
<td>27</td>
<td>-2.30</td>
<td>0.0294</td>
<td>-4.2599</td>
<td>-0.2426</td>
</tr>
<tr>
<td>c</td>
<td>45.1815</td>
<td>34.9095</td>
<td>27</td>
<td>1.29</td>
<td>0.2065</td>
<td>-26.4469</td>
<td>116.8</td>
</tr>
</tbody>
</table>

**Syntax: NLMOD Procedure**

The following statements are available in the NLMOD procedure:

```plaintext
PROC NLMOD < options > ;
   BOUNDS constraint < , constraint . . . > ;
   BY variables ;
   DISPLAY < table-list>< / options > ;
   DISPLAYOUT table-spec-list< / options > ;
   ESTIMATE 'label' expression < options > ;
   ID variables ;
   MODEL dependent-variable ~ distribution ;
   PARAMETERS < parameter-specification > < , . . . , parameter-specification > < / options > ;
   PREDICT 'label' expression < options > ;
   RESTRICT restriction1 < , restriction2 . . . > ;
   Programming Statements ;
```

The `PROC NLMOD` statement and exactly one `MODEL` statement are required.
Chapter 14: The NLMOD Procedure

PROC NLMOD Statement

PROC NLMOD <options>;

The PROC NLMOD statement invokes the procedure. Table 14.1 summarizes important options in the PROC NLMOD statement by function. These and other options in the PROC NLMOD statement are then described fully in alphabetical order.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Basic Options</strong></td>
<td></td>
</tr>
<tr>
<td>DATA=</td>
<td>Specifies the input data table</td>
</tr>
<tr>
<td>OUT=</td>
<td>Specifies the output data table</td>
</tr>
<tr>
<td><strong>Output Options</strong></td>
<td></td>
</tr>
<tr>
<td>CORR</td>
<td>Specifies the correlation matrix</td>
</tr>
<tr>
<td>COV</td>
<td>Specifies the covariance matrix</td>
</tr>
<tr>
<td>ECORR</td>
<td>Specifies the correlation matrix of additional estimates</td>
</tr>
<tr>
<td>ECOV</td>
<td>Specifies the covariance matrix of additional estimates</td>
</tr>
<tr>
<td>DF</td>
<td>Specifies the default degrees of freedom</td>
</tr>
<tr>
<td>NOPRINT</td>
<td>Suppresses ODS output</td>
</tr>
<tr>
<td>NOITPRINT</td>
<td>Suppresses output about iterations within the optimization process</td>
</tr>
<tr>
<td><strong>Optimization Options</strong></td>
<td></td>
</tr>
<tr>
<td>ABSCONV=</td>
<td>Tunes an absolute function convergence criterion</td>
</tr>
<tr>
<td>ABSFCONV=</td>
<td>Tunes an absolute difference function convergence criterion</td>
</tr>
<tr>
<td>ABSSGCONV=</td>
<td>Tunes the absolute gradient convergence criterion</td>
</tr>
<tr>
<td>FCONV=</td>
<td>Tunes the relative function convergence criterion</td>
</tr>
<tr>
<td>GCONV=</td>
<td>Tunes the relative gradient convergence criterion</td>
</tr>
<tr>
<td>MAXITER=</td>
<td>Chooses the maximum number of iterations in any optimization</td>
</tr>
<tr>
<td>MAXFUNC=</td>
<td>Specifies the maximum number of function evaluations in any optimization</td>
</tr>
<tr>
<td>MAXTIME=</td>
<td>Specifies the upper limit (in seconds) of CPU time for any optimization</td>
</tr>
<tr>
<td>MINTITER=</td>
<td>Specifies the minimum number of iterations in any optimization</td>
</tr>
<tr>
<td>TECHNIQUE=</td>
<td>Selects the optimization technique</td>
</tr>
<tr>
<td><strong>Tolerance Options</strong></td>
<td></td>
</tr>
<tr>
<td>SINGULAR=</td>
<td>Tunes the general singularity criterion</td>
</tr>
</tbody>
</table>

The optimization options are fully described in the section “Optimization Options” on page 44 in Chapter 3, “Shared Concepts.” Other options available in the PROC NLMOD statement are described in the following sections.
You can specify the following *options* in the PROC NLMOD statement:

**ALPHA=</no_br>**

specifies the level of significance $\alpha$ that is used in constructing $100(1 - \alpha)\%$ confidence intervals. The value must be strictly between 0 and 1; the default value of $\alpha = 0.05$ results in 95% intervals. This value is used as the default confidence level for limits that are computed in the “Parameter Estimates” table and is used in the LOWER and UPPER options in the PREDICT statement.

**CORR**

requests the approximate correlation matrix for the parameter estimates.

**COV**

requests the approximate covariance matrix for the parameter estimates.

**DATA=</no_br><no_br>**

names the input data table for PROC NLMOD to use. The default is the most recently created data table. *CAS-libref.data-table* is a two-level name, where

- **CAS-libref** refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to the data, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about CAS-libref, see the section “Using CAS Sessions and CAS Engine Librefs” on page 675.

- **data-table** specifies the name of the input data table.

**DF=</no_br><no_br>**

specifies the default number of degrees of freedom to use in the calculation of $p$-values and confidence limits for additional parameter estimates.

**ECORR**

requests the approximate correlation matrix for all expressions that are specified in ESTIMATE statements.

**ECOV**

requests the approximate covariance matrix for all expressions that are specified in ESTIMATE statements.

**NOITPRINT**

suppresses the display of the “Iteration History” table.

**NOPRINT**

suppresses the generation of ODS output.

**OUT=</no_br><no_br>**

names the output data table for PROC NLMOD to use. *CAS-libref.data-table* is a two-level name, where

- **CAS-libref** refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to where the data table is to be stored, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about CAS-libref, see the section “Using CAS Sessions and CAS Engine Librefs” on page 675.

- **data-table** specifies the name of the output data table.
A single OUT= data table is created to contain all predicted values when more than one PREDICT statement is specified. An error message is generated if a PREDICT statement is specified and an OUT= data table is not specified. The variables in the input data table are not included in the output data table, in order to avoid data duplication for large data tables; however, variables that you specify in the ID statement are included.

**SINGULAR=number**

tunes the general singularity criterion that is applied in sweeps and inversions. The default is 1E4 times the machine epsilon; this product is approximately 1E-12 on most computers.

---

**BOUNDS Statement**

```
BOUNDS constraint <, constraint ...> ;
```

where `constraint` represents

```
< number operator > parameter-list < operator number >
```

Boundary constraints are specified in a BOUNDS statement. One- or two-sided boundary constraints are allowed. Elements in a list of boundary constraints are separated by commas. For example:

```
bounds 0 <= a1-a9 x <= 1, -1 <= c2-c5;
bounds b1-b10 y >= 0;
```

You can specify more than one BOUNDS statement. If you specify more than one lower (or upper) bound for the same parameter, the maximum (or minimum) of these is taken.

If the maximum of all lower bounds is larger than the minimum of all upper bounds for the same parameter, the boundary constraint is replaced by \( \theta_j := l_j := \min(u_j) \), which is defined by the minimum of all upper bounds specified for \( \theta_j \).

---

**BY Statement**

```
BY variables ;
```

You can specify a BY statement in PROC NLMOD to obtain separate analyses of observations in groups that are defined by the values of the BY variables. If you specify more than one BY statement, only the last one specified is used. For more information, see the discussion of BY-group processing in SAS Language Reference: Concepts.

---

**DISPLAY Statement**

```
DISPLAY < table-list > < / options > ;
```

The DISPLAY statement enables you to specify a list of display tables to display or exclude. This statement is similar to the ODS SELECT, ODS EXCLUDE, and ODS TRACE statements. However, the DISPLAY statement can improve performance when a large number of tables could be generated (such as in BY-group processing). The procedure processes the DISPLAY statement on a CAS server and thus sends only a
subset of ODS tables to the SAS client. Because ODS statements are processed on a SAS client, first all the generated display tables are sent to the client, and then the client creates a subset.

If you use both DISPLAY and ODS statements together, the DISPLAY statement takes precedence over the ODS statements. Note that the ODS EXCLUDE statement processes tables that are sent to the client after they have been filtered by the DISPLAY statement. In some cases, it might appear that the ODS EXCLUDE statement is taking precedence because it can further filter the tables. For more information about ODS, see SAS Output Delivery System: Procedures Guide.

You can specify the table-list as a list of table names, paths, partial pathnames, and regular expressions.

The table names that you can specify are listed in the section “ODS Table Names” on page 697. A path is a table name that is prefixed with dot-separated grouping information. For example, a SelectionSummary table that a procedure produces during a selection routine might have the path Bygroup1.Summary.SelectionSummary. A partial pathname does not include all groups; for example, SelectionSummary and Summary.SelectionSummary are partial pathnames for Bygroup1.Summary.SelectionSummary.

When you specify a table name or partial pathname, all display tables whose paths end in the specified name are selected for display or exclusion. For example, both SelectionSummary and Summary.SelectionSummary select Bygroup1.Summary.SelectionSummary.

A regular expression is enclosed in forward slashes (/). For example, specifying “/tions/” selects all pathnames that contain the substring “tions”; in particular, the Bygroup1.Summary.SelectionSummary table is selected. Specifying “!/tions/” selects all pathnames that do not contain the substring “tions”; in particular, the Bygroup1.Summary.SelectionSummary table is not selected.

You can specify the following options after a slash (/):

**CASESENSITIVE**
performs a case-sensitive comparison of table names in the table-list to display table names when tables are subsetted for display. To preserve case, you must enclose table names in the table-list in quotation marks.

**EXCLUDE**
displays all display tables except those that you specify in the table-list.

**EXCLUDEALL**
suppresses display of all tables. This option takes precedence over the other options.

**TRACE**
displays the display table names, labels, and paths.

---

**DISPLAYOUT Statement**

```plaintext
DISPLAYOUT table-spec-list < / options > ;
```

The DISPLAYOUT statement enables you to create CAS output tables from your displayed output. This statement is similar to the ODS OUTPUT statement. For more information about ODS, see SAS Output Delivery System: Procedures Guide.

The table-spec-list specifies a list of CAS output tables to create. Each entry in the list has either a key=value format or a key format:
key=value specifies key as the ODS table name, path, or partial pathname, and specifies value as the CAS output table name.

key specifies key as the ODS table name and also as the CAS output table name.

The ODS table names that you can specify are listed in the section “ODS Table Names” on page 697. You cannot specify the ODS table named OutputCasTables in the table-spec-list.

Table names and partial pathnames are discussed under the DISPLAY statement. The DISPLAYOUT statement does not support regular expressions.

You can specify the following options after a slash (/):

**INCLUDEALL**
creates output CAS tables for all display tables. The name of the created output CAS table is the same as the corresponding display table name. If you specify this option, the table-spec-list specification is ignored.

**NOREPLACE**
does not replace any existing CAS output table of the same name.

**REPEATED**
replicates all CAS output tables on all nodes.

---

**ESTIMATE Statement**

```
ESTIMATE 'label' expression < options > ;
```

The ESTIMATE statement enables you to compute an additional estimate that is a function of the parameter values. You must provide a quoted string to identify the estimate and then provide a valid SAS expression. Multiple ESTIMATE statements are permitted, and results from all ESTIMATE statements are listed in a common table. PROC NLMOD computes approximate standard errors for the estimates by using the delta method (Billingsley 1986). It uses these standard errors to compute corresponding t statistics, p-values, and confidence limits.

The **ECOV** option in the PROC NLMOD statement produces a table that contains the approximate covariance matrix of all the additional estimates that you specify. The **ECORR** option produces the corresponding correlation matrix.

You can specify the following options:

**ALPHA=** \( \alpha \)
specifies the alpha level to be used to compute confidence limits. The default value corresponds to the **ALPHA=** option in the PROC NLMOD statement.

**DF=** \( d \)
specifies the degrees of freedom to be used to compute p-values and confidence limits. The default value corresponds to the **DF=** option in the PROC NLMOD statement.
ID Statement

ID variables;

The ID statement lists one or more variables from the input data table that are transferred to output data tables that PROC NLMOD creates, provided that the output data table contains one or more records per input observation. For example, when you use an OUT= option in PROC NLMOD statement to produce prediction statistics, ID variables are added to the output data table.

By default, PROC NLMOD do not include all variables from the input data table in output data tables.

MODEL Statement

MODEL dependent-variable ~ distribution;

The MODEL statement is the mechanism for either using a distribution specification to specify the distribution of the data or using the RESIDUAL distribution to specify a predicted value. You must specify a single dependent variable from the input data table, a tilde (~), and then a distribution and its parameters. You can specify the following values for distribution:

RESIDUAL(m) or LS(m) specifies no particular distribution. Instead the sum of squares of the differences between m and the dependent variable is minimized.
NORMAL(m, v) specifies a normal (Gaussian) distribution that has mean m and variance v.
BINARY(p) specifies a binary (Bernoulli) distribution that has probability p.
BINOMIAL(n, p) specifies a binomial distribution that has count n and probability p.
GAMMA(a, b) specifies a gamma distribution that has shape a and scale b.
NEGBIN(n, p) specifies a negative binomial distribution that has count n and probability p.
POISSON(m) specifies a Poisson distribution that has mean m.
GENERAL(ll) specifies a general log-likelihood function that you construct by using SAS programming statements.

The MODEL statement must follow any SAS programming statements that you specify for computing parameters of the preceding distributions. For information about the built-in log-likelihood functions, see the section “Built-In Log-Likelihood Functions” on page 692.

PARAMETERS Statement

PARAMETERS < parameter-specification > <, . . . , parameter-specification > </ options > ;
PARMS < parameter-specification > <, . . . , parameter-specification > </ options > ;

The purpose of the PARAMETERS statement is to provide starting values for the NLMOD procedure. You can provide values that define a single point in the parameter space or that define a set of points. For more information about the parameter-specification, see the section “Assigning Starting Values by Using a Parameter Specification” on page 686.

You can specify the following options after a slash (/):
**BEST=i > 0**
specifies the maximum number of parameter grid points and the corresponding objective function values to display in the “Parameters” table. If you specify this option, the parameter grid points are listed in ascending order of objective function value. By default, all parameter grid points are displayed.

**PDATA=CAS-libref.data-table**
**DATA=CAS-libref.data-table**
specifies the data table that provides parameter starting values. *CAS-libref.data-table* is a two-level name, where *CAS-libref* refers to the caslib and session identifier, and *data-table* specifies the name of the input data table. For more information about this two-level name, see the **DATA=** option and the section “Using CAS Sessions and CAS Engine Librefs” on page 675.

**START=value**
**DEFSSTART=value**
specifies a default starting value for all parameters.

Four methods are available for providing starting values to the optimization process. In order of precedence, the methods are as follows:

1. Specify values directly in the PARAMETERS statement.
2. Specify values in the PDATA= data table option.
3. Specify a single value for all parameters by using the START= option.
4. Use the default value 1.0.

The names that you assign to parameters must be valid SAS names and must not be the same as names of variables in the input data table (see the **DATA=** option in the **PROC NLMOD** statement). Parameters that are assigned starting values through the PARAMETERS statement can be omitted from the estimation if the expression in the **MODEL** statement does not depend on them.

**Assigning Starting Values by Using a Parameter Specification**

A **parameter-specification** has the following general form, where *name* identifies the parameter and *value-list* provides the set of starting values for the parameter:

\[ \text{name} = \text{value-list} \]

Often the *value-list* contains only a single value, but you can use more general and flexible list specifications such as these:

- \(m\) a single value
- \(m_1, m_2, \ldots, m_n\) several values
- \(m \text{ TO } n\) a sequence in which \(m\) equals the starting value, \(n\) equals the ending value, and the increment is 1
- \(m \text{ TO } n \text{ BY } i\) a sequence in which \(m\) equals the starting value, \(n\) equals the ending value, and the increment is \(i\)
- \(m_1, m_2 \text{ TO } m_3\) mixed values and sequences
When you specify more than one value for a parameter, PROC NLMOD sorts the values in ascending order and removes duplicate values from the parameter list before forming the grid for the parameter search. If you specify several values for each parameter, PROC NLMOD evaluates the model at each point on the grid. The iterations then commence from the point on the grid that yields the smallest objective function value.

For example, the following PARMS statement specifies five parameters and sets their possible starting values as shown in the table:

```
parms b0 = 0
   b1 = 4 to 8
   b2 = 0 to .6 by .2
   b3 = 1, 10, 100
   b4 = 0, .5, 1 to 4;
```

<table>
<thead>
<tr>
<th>Possible Starting Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>B0</td>
</tr>
<tr>
<td>----</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
</tbody>
</table>

The objective function values are calculated for each of the $1 \times 5 \times 4 \times 3 \times 6 = 360$ combinations of possible starting values.

If you specify a starting value by using a **parameter-specification**, any starting values that are provided for this parameter through the PDATA= data table are ignored. The **parameter-specification** overrides the information in the PDATA= data table.

### Assigning Starting Values from a Data Table That Resides in a CAS Library

The PDATA= option in the PARAMETERS statement enables you to assign starting values for parameters by using a data table. You must specify the CAS library where your data table resides and the data table name. The data table must contain at least two variables: a character variable named Parameter (or Parm) that identifies the parameter, and a numeric variable named Estimate (or Est) that contains the starting values. For example, the PDATA= option enables you to use the contents of the “ParameterEstimates” table from one PROC NLMOD run to supply starting values for a subsequent run, as follows:

```
proc nlmod data=mycas.D(obs=30);
   parameters alpha=100 beta=3 gamma=4;
   Switch = 1/(1+gamma*exp(beta*log(dose)));    
   model y ~ residual(alpha*Switch);
   displayout ParameterEstimates=pest;
run;

proc nlmod data=mycas.D;
   parameters / pdata=mycas.pest;
   Switch = 1/(1+gamma*exp(beta*log(dose)));    
   model y ~ residual(alpha*Switch);
run;
```
You can specify multiple values for a parameter in the PDATA= data table, and the parameters can appear in any order. The starting values are collected by parameter and arranged in ascending order, and duplicate values are removed. The parameter names in the PDATA= data table are not case-sensitive. For example, the following DATA step defines starting values for three parameters and a starting grid that has $1 \times 3 \times 1 = 3$ points:

```sas
data mycas.Test;
  input Parameter $ Estimate;
datalines;
alpha 100
BETA 4
beta 4.1
beta 4.2
beta 4.1
gamma 30
;
```

**PREDICT Statement**

```sas
PREDICT 'label' expression < options > ;
PREDICT 'label' MEAN < options > ;
```

The PREDICT statement enables you to construct predictions of an expression across all the observations in the input data table. Multiple PREDICT statements are permitted. Results for all PREDICT statements are displayed in the output data table that you specify in the OUT= option in the PROC NLMOD statement. The variables in the input data table are not included in the output data table, in order to avoid data duplication for large data tables; however, variables that you specify in the ID statement are included.

You must specify the following arguments:

- `'label'` identifies the predicted expression.
- `expression` provides the predicted value. You can specify the predicted value either by using a SAS programming expression that contains the input data table variables and parameters or by using the keyword MEAN. If you specify the keyword MEAN, the predicted mean value for the distribution specified in the MODEL statement is used. Predicted values are computed using the final parameter estimates. Standard errors of prediction are computed using the delta method (Billingsley 1986; Cox 1998).

You can also specify the following options:

- `ALPHA=\alpha` specifies the alpha level to be used to compute confidence limits. The default value corresponds to the `ALPHA=` option in the PROC NLMOD statement.
- `DF=d` specifies the degrees of freedom to be used to compute confidence limits. The default value corresponds to the `DF=` option in the PROC NLMOD statement.
LOWER=name
specifies a variable that contains the lower confidence limit of the predicted value.

PRED=name
specifies a variable that contains the predicted value.

PROBT=name
specifies a variable that contains the p-value of the predicted value.

STDERR=name
specifies a variable that contains the standard error of the predicted value.

TVALUE=name
specifies a variable that contains the t statistic for the predicted value.

UPPER=name
specifies a variable that contains the upper confidence limit of the predicted value.

RESTRICT Statement

RESTRICT  restriction1 <, restriction2 . . . > ;

The RESTRICT statement imposes linear restrictions on the model’s parameters estimates. You can specify any number of RESTRICT statements.

Each restriction is written as an expression, optionally followed by an equality operator (=) or an inequality operator (<, >, <=, >=), followed by a second expression:

expression < operator expression>

The operator can be =, <, >, <=, or >=. The operator and second expression are optional. When they are omitted, the operator defaults to = and the second expression defaults to the value 0.

Restriction expressions can be composed of parameter names, arithmetic operators, functions, and constants. You cannot use comparison operators (such as = or <) or logical operators (such as &) in RESTRICT statement expressions. Parameters that are named in restriction expressions must be among the parameters that are estimated by the model. Restriction expressions cannot refer to other variables that are defined in the program or in the DATA= data table. The restriction expressions must be linear functions of the parameters.

The following example illustrates how to use the RESTRICT statement to impose a linear constraint on parameters:

```
proc nlmdd;
   parms alpha beta;
   f = (x/alpha + beta)**2
   model y ~ residual(f);
   restrict beta < 2*(alpha + constant('pi'));
run;
```

The preceding RESTRICT statement represents the following model constraint:

$$\beta < 2(\alpha + \pi)$$
Chapter 14: The NLMOD Procedure

Programming Statements

Programming statements define the arguments of the MODEL, ESTIMATE, and PREDICT statements in PROC NLMOD. Most of the programming statements that you can use in the SAS DATA step can also be used in the NLMOD procedure. For a description of SAS programming statements, see SAS Language Reference: Concepts. You can use the following programming statements:

- **ABORT**;
- **CALL** name [( expression [, expression ...] )];
- **DELETE**;
- **DO** [variable = expression
  [ TO expression ] [ BY expression ]
  [, expression [ TO expression ] [ BY expression ] ...]
  ]
  [ WHILE expression ] [ UNTIL expression ] ;
- **END**;
- **GOTO** statement_label;
- **IF** expression;
- **IF** expression **THEN** program_statement;
  **ELSE** program_statement;
- **variable** = expression;
- **variable** + expression;
- **LINK** statement_label;
- **PUT** [variable] [ = ] [...];
- **RETURN**;
- **SELECT**[(expression)];
- **STOP**;
- **SUBSTR** ( variable, index, length ) = expression;
- **WHEN** (expression) program_statement;
  **OTHERWISE** program_statement;

For the most part, these SAS programming statements work the same as they do in the SAS DATA step, as documented in SAS Language Reference: Concepts. However, they differ as follows:

- The ABORT statement does not allow any arguments.
- The DO statement does not allow a character index variable. Thus, PROC NLMOD supports the first of the following statements, but not the second:

  do i = 1,2,3;

  do i = 'A', 'B', 'C';

- In contrast to other procedures that share PROC NLMOD’s programming syntax, PROC NLMOD does not support the LAG function. Because observations are not processed sequentially when high-performance analytical procedures perform the parameter optimization, information for computing lagged values is not available.
The PUT statement, used mostly for program debugging in PROC NLMOD, supports only some of the features of the DATA step PUT statement, and it has some new features that the DATA step PUT statement does not have:

- The PROC NLMOD PUT statement does not support line pointers, factored lists, iteration factors, overprinting, _INFILE_, the colon (:) format modifier, or “$”.
- The PROC NLMOD PUT statement supports expressions, but the expression must be enclosed in parentheses. For example, the following statement displays the square root of \( x \):

\[
\text{put } (\sqrt{x});
\]

- The PROC NLMOD PUT statement supports the item _PDV_, which displays a formatted listing of all variables in the program. For example, the following statement displays a much more readable listing of the variables than the _ALL_ print item:

\[
\text{put _pdv_;}
\]

The WHEN and OTHERWISE statements enable you to specify more than one programming statement. That is, DO/END groups are not necessary for multiple WHEN statements. For example, the following syntax is valid:

\[
\text{select;}
\text{when (exp1) stmt1;}
\text{stmt2;}
\text{when (exp2) stmt3;}
\text{stmt4;}
\text{end;}
\]

When you write your programming statements, avoid defining variables that begin with an underscore (_), because they might conflict with internal variables that PROC NLMOD creates. The MODEL statement must follow any SAS programming statements that define or modify terms that are used to specify the model.

---

**Least Squares Estimation**

Models that are estimated by PROC NLMOD can be represented by using the equations

\[
Y = f(\beta; z_1, \ldots, z_k) + \epsilon
\]

\[
E[\epsilon] = 0
\]

\[
\text{Var}[\epsilon] = \sigma^2 I
\]

where
$Y$ is the $(n \times 1)$ vector of observed responses.

$f$ is the nonlinear prediction function of parameters and regressor variables.

$\beta$ is the vector of model parameters to be estimated.

$z_1, \ldots, z_k$ are the $(n \times 1)$ vectors of each of the $k$ regressor variables.

$\epsilon$ is the $(n \times 1)$ vector of residuals.

$\sigma^2$ is the variance of the residuals.

In these models, the distribution of the residuals is not specified and the model parameters are estimated using the least squares method. For the standard errors and confidence limits in the “ParameterEstimates” table to apply, the errors are assumed to be homoscedastic and uncorrelated and to have zero mean.

### Built-In Log-Likelihood Functions

For models in which the distribution of model errors is specified, the NLMOD procedure estimates parameters by maximizing the value of a log-likelihood function for the specified distribution. The log-likelihood functions that PROC NLMOD uses for the supported error distributions are as follows:

$Y \sim \text{normal}(m, v)$

$$l(m, v; y) = \frac{1}{2} \left( \log{2\pi} + \frac{(y - m)^2}{v} + \log{v} \right)$$

$$E[Y] = m$$

$$\text{Var}[Y] = v$$

$v > 0$

$Y \sim \text{binary}(p)$

$$l_1(p; y) = \begin{cases} y \log{p} & y > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$l_2(p; y) = \begin{cases} (1 - y) \log{1 - p} & y < 1 \\ 0 & \text{otherwise} \end{cases}$$

$$l(p; y) = l_1(p; y) + l_2(p; y)$$

$$E[Y] = p$$

$$\text{Var}[Y] = p \left(1 - p\right)$$

$0 < p < 1$
\[
Y \sim \text{binomial}(n, p)
\]
\[
l_c = \log\{\Gamma(n + 1)\} - \log\{\Gamma(y + 1)\} - \log\{\Gamma(n - y + 1)\}
\]
\[
l_1(n, p; y) = \begin{cases} y \log\{p\} & y > 0 \\ 0 & \text{otherwise} \end{cases}
\]
\[
l_2(n, p; y) = \begin{cases} (n - y) \log\{1 - p\} & n - y > 0 \\ 0 & \text{otherwise} \end{cases}
\]
\[
l(n, p; y) = l_c + l_1(n, p; y) + l_2(n, p; y)
\]
\[
E[Y] = np \\
\text{Var}[Y] = np(1 - p) \\
0 < p < 1
\]

\[
Y \sim \text{gamma}(a, b)
\]
\[
l(a, b; y) = -a \log\{b\} - \log\{\Gamma(a)\} + (a - 1) \log\{y\} - y/b
\]
\[
E[Y] = ab \\
\text{Var}[Y] = ab^2 \\
a > 0 \\
b > 0
\]

This parameterization of the gamma distribution differs from the parameterization that the GENSELECT procedure use. The scale parameter in PROC NLMOD is expressed as the inverse of the scale parameter that PROC GENSELECT uses. The PROC NLMOD parameter represents the scale of the magnitude of the residuals. You can estimate the scale parameter in PROC GENSELECT by using the following statements:

```
proc genselect;
   model y = x / dist=gamma s;
run;
```

PROC NLMOD uses the following statements to estimate the equivalent scale parameter:

```
proc nlmom;
   parms b0=1 b1=0 scale=14;
   linp = b0 + b1*x;
   mu = exp(linp);
   b = mu*scale;
   model y ~ gamma(1/scale,b);
run;
```
The NLMOD Procedure

Y ∼ negbin(n, p)

\[ l(n, p; y) = \log \{ \Gamma(n + y) \} - \log \{ \Gamma(n) \} - \log \{ \Gamma(y + 1) \} + n \log \{ p \} + y \log \{ 1 - p \} \]

\[ E[Y] = n \left( \frac{1}{p} \right) \]

\[ \text{Var}[Y] = n \left( \frac{1}{p^2} \right) \]

\[ n \geq 0 \]

\[ 0 < p < 1 \]

The value of parameter n can be a real number; it does not have to be an integer.

Y ∼ Poisson(m)

\[ l(m; y) = y \log \{ m \} - m - \log \{ \Gamma(y + 1) \} \]

\[ E[Y] = m \]

\[ \text{Var}[Y] = m \]

\[ m > 0 \]

Multithreading

The NLMOD procedure allocates data to different threads and calculates the likelihood function, gradient, and Hessian by accumulating the values from all threads. For more information about how PROC NLMOD uses threads, see the section “Multithreading” on page 81 in Chapter 3, “Shared Concepts.”

Optimization Algorithms

Several optimization techniques are available in PROC NLMOD. You can choose a particular optimizer by using the TECHNIQUE= option in the PROC NLMOD statement. Table 14.3 summarizes the optimization techniques available in PROC NLMOD.

<table>
<thead>
<tr>
<th>TECHNIQUE=</th>
<th>Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRUREG</td>
<td>Trust region method</td>
</tr>
<tr>
<td>NEWRAP</td>
<td>Newton-Raphson method with line search</td>
</tr>
<tr>
<td>NRRIDG</td>
<td>Newton-Raphson method with ridging</td>
</tr>
<tr>
<td>QUANEW</td>
<td>Quasi-Newton methods</td>
</tr>
<tr>
<td>DBLDOG</td>
<td>Double-dogleg method</td>
</tr>
<tr>
<td>CONGRA</td>
<td>Conjugate gradient methods</td>
</tr>
<tr>
<td>LEVMAR</td>
<td>Levenberg-Marquardt method</td>
</tr>
<tr>
<td>NMSIMP</td>
<td>Nelder-Mead simplex method</td>
</tr>
</tbody>
</table>
There is no algorithm for optimizing general nonlinear functions that always finds the global optimum for a general nonlinear optimization problem in a reasonable amount of time. Because no single optimization technique is always superior to others, PROC NLMOD provides a variety of optimization techniques that work well in various circumstances. However, you can devise problems for which none of the techniques in PROC NLMOD can find the correct solution. Moreover, nonlinear optimization can be computationally expensive in terms of time and memory, so you must be careful when matching an algorithm to a problem. The section “Choosing an Optimization Algorithm” on page 82 in Chapter 3, “Shared Concepts,” is helpful in choosing a suitable optimization algorithm.

Displayed Output

The following sections describe the output that PROC NLMOD produces by default. The output is organized into various tables, which are discussed in their order of appearance.

Specifications

The “Specifications” table displays basic information about the model, such as the data source, the dependent variable, the distribution being modeled, and the optimization technique.

Number of Observations

The “Number of Observations” table displays the number of observations that are read from the input data table and the number of observations that are used in the analysis.

Dimensions

The “Dimensions” table displays the number of parameters that are estimated in the model and the number of upper and lower bounds that are imposed on the parameters.

Parameters

The “Parameters” table displays the initial values of parameters that are used to start the estimation process. You can limit this information by specifying the BEST= option in the PARAMETERS statement when you specify a large number of initial parameter value combinations. The parameter combinations and their corresponding objective function values are listed in increasing order of objective function value.

Iteration History

For each iteration of the optimization, the “Iteration History” table displays the number of function evaluations (including gradient and Hessian evaluations), the value of the objective function, the change in the objective function from the previous iteration, and the absolute value of the largest (projected) gradient element.

Convergence Status

The convergence status table is a small ODS table that appears as a message that indicates whether the optimization succeeded and which convergence criterion was met. If the optimization fails, the message indicates the reason for the failure. If you save the convergence status table to an output data set, a numeric
Chapter 14: The NLMOD Procedure

Status variable is added that enables you to programmatically assess convergence. The values of the Status variable indicate the following:

0 Convergence was achieved or an optimization was not performed because TECHNIQUE=None.
1 The objective function could not be improved.
2 Convergence was not achieved because of a user interruption or because a limit (such as the maximum number of iterations or the maximum number of function evaluations) was reached. To modify these limits, see the MAXITER=, MAXFUNC=, and MAXTIME= options in the PROC NLMOD statement.
3 Optimization failed to converge because function or derivative evaluations failed at the starting values or during the iterations or because a feasible point that satisfies the parameter constraints could not be found in the parameter space.

Linear Constraints

The “Linear Constraints” table summarizes the linear constraints that are applied to the model by using the RESTRICT statements. The table lists all the constraints that are specified in the model, along with information about whether each constraint represents an inequality or equality condition and whether that constraint is active for the final parameter estimates.

Fit Statistics

The “Fit Statistics” table displays a variety of measures of fit, depending on whether the model was estimated using least squares or maximum likelihood. In both cases, smaller values of the fit statistics indicate better fit.

For least squares estimations, the table displays the sum of squares of errors and the variance of errors.

For maximum likelihood estimations, the table uses the following formulas to display information criteria, where \( p \) denotes the number of effective parameters, \( n \) denotes the number of observations used, and \( l \) is the log likelihood that is evaluated at the converged estimates:

\[
\text{AIC} = -2l + 2p \\
\text{AICC} = \begin{cases} 
-2l + 2pn/(n - p - 1) & n > p + 2 \\
-2l + 2p(p + 2) & \text{otherwise}
\end{cases} \\
\text{BIC} = -2l + p \log(n)
\]

The information criteria values that are displayed in the “Fit Statistics” table are not based on a normalized log-likelihood function.

ANOVA

The “Analysis of Variance” (ANOVA) table is displayed only for least squares estimations. It displays the number of degrees of freedom and the sum of squares that are attributed to the model, the error, and the total. The ANOVA table also reports the variance of the model and the errors, the \( F \) statistic, and its probability for the model.
Parameter Estimates

The “Parameter Estimates” table displays the parameter estimates, their estimated (asymptotic) standard errors $t$ statistics, and associated $p$-values for the hypothesis that the parameter is 0. Confidence limits, which are displayed for each parameter, are based on the value of the ALPHA= option that you specify in the PROC NLMOD statement.

Additional Estimates

The “Additional Estimates” table displays the same information as the “Parameter Estimates” table for the expressions that appear in the optional ESTIMATE statements. The table is generated when you specify one or more ESTIMATE statements. Because you can specify a separate ALPHA= option for each ESTIMATE statement, the “Additional Estimates” table also includes a column that indicates each confidence interval’s corresponding significance level.

Covariance

The “Covariance” table appears when you specify the COV option in the PROC NLMOD statement. It displays a matrix of covariances between each pair of estimated parameters.

Correlation

The “Correlation” table appears when you specify the CORR option in the PROC NLMOD statement. It displays the correlation matrix for the estimated parameters.

Additional Estimates Covariance

The “Covariance of Additional Estimates” table appears when you specify the ECOV option in the PROC NLMOD statement. It displays a matrix of covariances between each pair of expressions that are specified in ESTIMATE statements.

Additional Estimates Correlation

The “Correlation of Additional Estimates” table appears when you specify the ECORR option in the PROC NLMOD statement. It displays the correlation matrix for the expressions that are specified in ESTIMATE statements.

Procedure Task Timing

The “Timing” table displays the amount of time (in seconds) that PROC NLMOD required to perform different tasks in the analysis.

ODS Table Names

Each table that the NLMOD procedure creates has a name associated with it. You must use this name to refer to the table when you use the DISPLAY statement, the DISPLAYOUT statement, or ODS statements. These names are listed in Table 14.4.
## Table 14.4 ODS Tables Produced by PROC NLMOD

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>AdditionalEstimates</td>
<td>Functions of estimated parameters and their associated statistics</td>
<td>ESTIMATE</td>
<td>Default</td>
</tr>
<tr>
<td>ANOVA</td>
<td>Least squares analysis of variance information</td>
<td>MODEL</td>
<td>RESIDUAL</td>
</tr>
<tr>
<td>Constraints</td>
<td>Information about the model’s linear constraints</td>
<td>RESTRICT</td>
<td>Default</td>
</tr>
<tr>
<td>ConvergenceStatus</td>
<td>Optimization success and convergence information</td>
<td>PROC NLMOD</td>
<td>Default</td>
</tr>
<tr>
<td>CorrB</td>
<td>Parameter correlation matrix</td>
<td>PROC NLMOD</td>
<td>CORR</td>
</tr>
<tr>
<td>CovB</td>
<td>Parameter covariance matrix</td>
<td>PROC NLMOD</td>
<td>COV</td>
</tr>
<tr>
<td>Dimensions</td>
<td>Number of parameters and their bounds</td>
<td>PROC NLMOD</td>
<td>Default</td>
</tr>
<tr>
<td>ECorrB</td>
<td>Additional estimates’ correlation matrix</td>
<td>PROC NLMOD</td>
<td>ECORR</td>
</tr>
<tr>
<td>ECovB</td>
<td>Additional estimates’ covariance matrix</td>
<td>PROC NLMOD</td>
<td>ECOV</td>
</tr>
<tr>
<td>FitStatistics</td>
<td>Statistics about the quality of the fit</td>
<td>PROC NLMOD</td>
<td>Default</td>
</tr>
<tr>
<td>IterHistory</td>
<td>Optimizer iteration information</td>
<td>PROC NLMOD</td>
<td>Default</td>
</tr>
<tr>
<td>NObs</td>
<td>Number of observations read and used</td>
<td>PROC NLMOD</td>
<td>Default</td>
</tr>
<tr>
<td>OutputCasTables</td>
<td>A special ODS table that has information about all the CAS tables that are</td>
<td>PROC NLMOD</td>
<td>OUT=</td>
</tr>
<tr>
<td></td>
<td>created during a CAS action execution</td>
<td></td>
<td>DISPLAYOUT</td>
</tr>
<tr>
<td>ParameterEstimates</td>
<td>Parameter estimates and associated statistics</td>
<td>PROC NLMOD</td>
<td>Default</td>
</tr>
<tr>
<td>Parameters</td>
<td>Initial parameter values</td>
<td>PROC NLMOD</td>
<td>Default</td>
</tr>
<tr>
<td>Specifications</td>
<td>Basic model characteristics</td>
<td>PROC NLMOD</td>
<td>Default</td>
</tr>
<tr>
<td>Timing</td>
<td>Absolute and relative times for tasks performed by the procedure</td>
<td>PROC NLMOD</td>
<td>Default</td>
</tr>
</tbody>
</table>
Example 14.1: Segmented Model

Suppose you are interested in fitting a model that consists of two segments that connect in a smooth fashion. For example, the following model states that the mean of $Y$ is a quadratic function in $x$ for values of $x$ less than $x_0$ and that the mean of $Y$ is constant for values of $x$ greater than $x_0$:

$$E[Y|x] = \begin{cases} \alpha + \beta x + \gamma x^2 & \text{if } x < x_0 \\ c & \text{if } x \geq x_0 \end{cases}$$

In this model equation, $\alpha$, $\beta$, and $\gamma$ are the coefficients of the quadratic segment, and $c$ is the plateau of the mean function. The NLMOD procedure can fit such a segmented model even when the join point, $x_0$, is unknown.

Suppose you also want to impose conditions on the two segments of the model. First, the curve should be continuous—that is, the quadratic and the plateau section need to meet at $x_0$. Second, the curve should be smooth—that is, the first derivative of the two segments with respect to $x$ needs to coincide at $x_0$.

The continuity condition requires that

$$c = E[Y|x_0] = \alpha + \beta x_0 + \gamma x_0^2$$

The smoothness condition requires that

$$\frac{\partial E[Y|x_0]}{\partial x} = \beta + 2\gamma x_0 \equiv 0$$

If you solve for $x_0$ and substitute your result in the expression for $c$, the two conditions jointly imply that

$$x_0 = -\beta/2\gamma$$
$$c = \alpha - \beta^2/4\gamma$$

Although there are five unknowns, the model contains only three independent parameters. Together the continuity and smoothness restrictions completely determine two parameters, given the other three.

The following DATA step creates the data table for this example in your CAS session:

```r
data mycas.A;
  input y x @@;
datalines;
.46 1 .47 2 .57 3 .61 4 .62 5 .68 6 .69 7 .78 8 .70 9 .74 10 .77 11 .78 12 .74 13 .80 13 .80 15 .78 16 ;
```

The following PROC NLMOD statements fit this segmented model:
Chapter 14: The NLMOD Procedure

proc nlmod data=mycas.A out=mycas.B;
   parms alpha=.45 beta=.05 gamma=-.0025;
   
x0 = -.5*beta / gamma;
   
   if (x < x0) then
      yp = alpha + beta*x + gamma*x*x;
   else
      yp = alpha + beta*x0 + gamma*x0*x0;
   
   model y ~ residual(yp);

   estimate 'join point' -beta/2/gamma;
   estimate 'plateau value c' alpha - beta**2/(4*gamma);
   predict 'predicted' yp pred=yp;
   predict 'response' y pred=y;
   predict 'x' x pred=x;
run;

The parameters of the model are $\alpha$, $\beta$, and $\gamma$, which are represented in the PROC NLMOD statements by the variables alpha, beta, and gamma, respectively. To model the two segments, a conditional statement assigns the appropriate expression to the mean function, depending on the value of $x_0$. The ESTIMATE statements compute the values of $x_0$ and $c$. The PREDICT statement computes predicted values for plotting and saves them to data table $b$.

The results from fitting this model are shown in Output 14.1.1 through Output 14.1.3. The iterative optimization converges after six iterations (Output 14.1.1). Output 14.1.2 shows the estimated parameters. Output 14.1.3 indicates that the join point is 12.7477 and the plateau value is 0.7775.

**Output 14.1.1** Nonlinear Least Squares Iterative Phase

Quadratic Model with Plateau

The NLMOD Procedure

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Evaluations</th>
<th>Objective Function</th>
<th>Change</th>
<th>Maximum Gradient</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>5</td>
<td>0.0035144531</td>
<td>7.184063</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0.0007352716</td>
<td>0.00277918</td>
<td>2.145337</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0.0006292751</td>
<td>0.00010600</td>
<td>0.032551</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>0.0006291261</td>
<td>0.00000015</td>
<td>0.002952</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>0.0006291244</td>
<td>0.00000000</td>
<td>0.000238</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>0.0006291244</td>
<td>0.00000000</td>
<td>0.000023</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>0.0006291244</td>
<td>0.00000000</td>
<td>2.313E-6</td>
</tr>
</tbody>
</table>

Convergence criterion (GCONV=1E-8) satisfied.
Example 14.1: Segmented Model

Output 14.1.2 Least Squares Analysis of the Quadratic Model

<table>
<thead>
<tr>
<th>Source</th>
<th>DF</th>
<th>Sum of Squares</th>
<th>Mean Square</th>
<th>F Value</th>
<th>Approx Pr &gt; F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>2</td>
<td>0.1769</td>
<td>0.0884</td>
<td>114.22</td>
<td>&lt;.0001</td>
</tr>
<tr>
<td>Error</td>
<td>13</td>
<td>0.0101</td>
<td>0.000774</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Corrected Total</td>
<td>15</td>
<td>0.1869</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Parameter Estimates

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Estimate</th>
<th>Standard Error</th>
<th>DF</th>
<th>t Value</th>
<th>Approx Pr &gt;</th>
<th>Lower</th>
<th>Upper</th>
</tr>
</thead>
<tbody>
<tr>
<td>alpha</td>
<td>0.3921</td>
<td>0.0267</td>
<td>13</td>
<td>14.70</td>
<td>&lt;.0001</td>
<td>0.3345</td>
<td>0.4497</td>
</tr>
<tr>
<td>beta</td>
<td>0.0605</td>
<td>0.00842</td>
<td>13</td>
<td>7.18</td>
<td>&lt;.0001</td>
<td>0.0423</td>
<td>0.0787</td>
</tr>
<tr>
<td>gamma</td>
<td>-0.00237</td>
<td>0.000551</td>
<td>13</td>
<td>-4.30</td>
<td>&lt;.0001</td>
<td>-0.00356</td>
<td>-0.00118</td>
</tr>
</tbody>
</table>

Output 14.1.3 Additional Estimates for the Quadratic Model

<table>
<thead>
<tr>
<th>Label</th>
<th>Estimate</th>
<th>Standard Error</th>
<th>DF</th>
<th>t Value</th>
<th>Approx Pr &gt;</th>
<th>Alpha</th>
<th>Lower</th>
<th>Upper</th>
</tr>
</thead>
<tbody>
<tr>
<td>join point</td>
<td>12.7477</td>
<td>1.2781</td>
<td>13</td>
<td>9.97</td>
<td>&lt;.0001</td>
<td>0.05</td>
<td>9.9864</td>
<td>15.5089</td>
</tr>
<tr>
<td>plateau value c</td>
<td>0.7775</td>
<td>0.0123</td>
<td>13</td>
<td>63.11</td>
<td>&lt;.0001</td>
<td>0.05</td>
<td>0.7509</td>
<td>0.8041</td>
</tr>
</tbody>
</table>

The following statements produce a graph, shown in Output 14.1.4, of the observed and predicted values along with reference lines for the join point and plateau estimates:

```sas
data B;
  set mycas.B;
run;

proc sort data = B;
  by x;
run;

proc sgplot data=B noautolegend;
  yaxis label='Observed or Predicted';
  refline 0.7775 / axis=y label="Plateau" labelpos=min;
  refline 12.7477 / axis=x label="Join point" labelpos=min;
  scatter y=y x=x;
  series y=yp x=x;
run;
```
References


Chapter 15
The PCA Procedure

Contents

Overview: PCA Procedure .................................................. 704
PROC PCA Features ....................................................... 704
PROC PCA Compared with Other SAS Procedures .................... 705
   PROC PCA Compared with the HPRINCOMP Procedure ............ 705
   PROC PCA Compared with the PRINCOMP Procedure ............. 706
Using CAS Sessions and CAS Engine Librefs ......................... 707
Getting Started: PCA Procedure ........................................ 708
Syntax: PCA Procedure ...................................................... 712
   PROC PCA Statement .................................................. 712
   BY Statement ......................................................... 718
   CODE Statement ..................................................... 718
   DISPLAY Statement ................................................ 718
   DISPLAYOUT Statement ............................................ 719
   FREQ Statement ..................................................... 720
   OUTPUT Statement .................................................. 720
   PARTIAL Statement ................................................ 722
   VAR Statement ....................................................... 722
   WEIGHT Statement .................................................. 723
Details: PCA Procedure ................................................... 723
   Computing Principal Components ................................ 723
      Eigenvalue Decomposition ..................................... 724
      NIPALS ........................................................... 724
      ITERGS ......................................................... 725
      RANDOM ......................................................... 725
   Missing Values ....................................................... 725
   OUTSTAT= Data Table ............................................... 726
   Displayed Output .................................................... 728
      Model Information ............................................... 728
      Number of Variables ............................................ 728
      Number of Observations ....................................... 728
      Simple Statistics ............................................... 728
      Centering and Scaling Information ............................. 728
      Explained Variation of Variables .............................. 728
      Correlation Matrix .............................................. 728
      Regression Statistics .......................................... 728
      Regression Coefficients ...................................... 729
Overview: PCA Procedure

The PCA procedure performs principal component analysis in SAS Viya.

Principal component analysis is a multivariate technique for examining relationships among several quantitative variables. It provides an optimal way to reduce dimensionality by projecting the data onto a lower-dimensional orthogonal subspace that explains as much variation as possible in those variables. The choice between using factor analysis and using principal component analysis depends in part on your research objectives. You should use the PCA procedure if you are interested in summarizing data and detecting linear relationships. You can use principal component analysis to reduce the number of variables in regression, clustering, and so on.

PROC PCA Features

The PCA procedure does the following:

- provides a PARTIAL statement for analyzing a partial correlation or covariance matrix
- provides a FREQ statement for grouped analysis
- provides a WEIGHT statement for weighted analysis
- provides a CODE statement for scoring new data
- produces an output data table that contains principal component scores and other observationwise statistics
- produces an output data table that contains means, standard deviations, number of observations, correlations or covariances, eigenvalues, and eigenvectors
- produces a scree plot of eigenvalues and proportion variance explained and a pattern profile plot
The PCA procedure implements the following methods:

- eigenvalue decomposition, which uses the correlation or covariance of the data matrix and calculates all the principal components simultaneously
- the nonlinear iterative partial least squares (NIPALS) method, which uses the data matrix and extracts the principal components successively
- the iterative method based on the Gram-Schmidt orthogonalization (ITERGS) of Andrecut (2009), which uses the data matrix and extracts the principal components successively. The algorithm applies reorthogonalization correction to both the scores and the loadings at each iteration step.
- the iterative method based on random projection (RANDOM) of Halko, Martinsson, and Tropp (2011), which uses the data matrix and extracts all requested principal components simultaneously. Because of the nature of the algorithm, the principal components are an approximation of the actual principal components.

Because the PCA procedure runs on CAS, it also does the following:

- enables you to run on a cluster of machines that distribute the data and the computations
- enables you to run in single-machine mode on CAS
- exploits all the available cores and concurrent threads. For information about how PROC PCA uses threads, see the section “Multithreading” on page 81 in Chapter 3, “Shared Concepts.”

PROC PCA Compared with Other SAS Procedures

The PCA procedure provides functionality to perform principal component analysis that is comparable to that of the HPPRINCOMP and PRINCOMP procedures in SAS/STAT software.

PROC PCA Compared with the HPPRINCOMP Procedure

The functionality of the PCA procedure closely resembles that of the HPPRINCOMP procedure, which is a high-performance procedure. The PCA procedure is the next generation of the HPPRINCOMP procedure, and it was developed specifically for SAS Viya. Both procedures are designed to run on a cluster of machines that distribute the data and the computations.

The PCA procedure and the HPPRINCOMP procedure have the similarities and differences shown in Table 15.1.
### Table 15.1 Comparison of PROC PCA and PROC HPRINCOMP

<table>
<thead>
<tr>
<th>Feature</th>
<th>PROC PCA</th>
<th>PROC HPRINCOMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supports various methods to extract principal components by using the METHOD= option in the PROC statement</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Supports BY statement</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Supports DISPLAY and DISPLAYOUT statements</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Supports CODE and OUTPUT statements</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Supports ID statement</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Supports PLOTS= option</td>
<td>No, with limited number of options</td>
<td>No</td>
</tr>
<tr>
<td>Includes input variables in output data table</td>
<td>Not by default, in order to avoid data duplication for large data tables. To include input variables, specify them in the COPYVARS= option in the OUTPUT statement.</td>
<td>Not by default, in order to avoid data duplication for large data sets. To include input variables, specify them in the ID statement.</td>
</tr>
<tr>
<td>Threading</td>
<td>Specifically designed for CAS; executes on multiple threads</td>
<td>Primarily designed for a distributed environment; executes on multiple threads</td>
</tr>
</tbody>
</table>

### PROC PCA Compared with the PRINCOMP Procedure

The PCA procedure and the PRINCOMP procedure have the similarities and differences shown in Table 15.2.

### Table 15.2 Comparison of PROC PCA and PROC PRINCOMP

<table>
<thead>
<tr>
<th>Feature</th>
<th>PROC PCA</th>
<th>PROC PRINCOMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supports various methods to extract principal components by using the METHOD= option in the PROC statement</td>
<td>Yes</td>
<td>No, supports only eigenvalue decomposition</td>
</tr>
<tr>
<td>Supports BY statement</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Supports DISPLAY and DISPLAYOUT statements</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Supports CODE and OUTPUT statements</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Supports ID statement</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Supports PLOTS= option</td>
<td>Limited number of options</td>
<td>Full set of options</td>
</tr>
</tbody>
</table>
Using CAS Sessions and CAS Engine Librefs

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:

```sas
proc options option=(CASHOST CASPORT);
run;
```

In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:

```sas
cas mysess;
libname mycas cas sessref=mysess;
```

The CAS statement creates the CAS session named `mysess`, and the LIBNAME statement creates the `mycas` CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the CASHOST and CASPORT of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the `mysess` session, you can terminate it by using the TERMINATE option in the CAS statement as follows:

```sas
cas mysess terminate;
```

For more information about the CAS and LIBNAME statements, see the section “Introduction to Shared Concepts” on page 10 in Chapter 3, “Shared Concepts.”
NOTE: Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

The following DATA step creates the Crime data table, which provides crime rates per 100,000 people in seven categories for each of the 50 US states in 1977, in your CAS session:

```sas
data mycas.Crime;
  title 'Crime Rates per 100,000 Population by State';
  input State $1-15 Murder Rape Robbery Assault Burglary Larceny Auto_Theft;
datalines;
Alabama  14.2  25.2  96.8  278.3  1135.5  1881.9  280.7
Alaska   10.8  51.6  96.8  284.0  1331.7  3698.4  753.3
Arizona  9.5  34.2  138.2  312.8  2346.1  4467.4  439.5
Arkansas  8.8  27.6  83.2  294.0  1952.6  1862.1  183.4
California 11.5  49.4  287.0  358.0  2139.4  3499.8  663.5
Colorado  6.3  42.0  170.7  292.9  1935.2  3903.2  477.1
Connecticut 4.2  16.8  129.5  312.3  2346.1  4467.4  439.5
Delaware  6.0  24.9  157.0  194.2  1682.6  3678.4  467.0
Florida  10.2  39.6  187.9  449.1  1859.9  3840.5  351.4
Georgia  11.7  31.1  140.5  256.5  1351.1  2170.2  297.9
Hawaii   2.3  10.6  41.2  89.8  1253.1  2350.7  246.9
Idaho    2.4  13.5  38.7  170.0  1253.1  2350.7  246.9
Illinois 9.9  21.8  213.2  209.0  1085.0  2828.5  528.6
Indiana  7.4  26.5  132.2  153.5  1086.2  2498.7  377.4
Iowa     2.3  10.6  41.2  89.8  1253.1  2350.7  246.9
Kansas   6.6  22.0  100.7  180.5  1270.4  2739.3  244.3
Kentucky 10.1  19.1  81.1  123.3  872.2  1662.1  245.4
Louisiana 15.5  30.9  142.9  335.5  1165.5  2469.9  337.7
Maine    2.4  13.5  38.7  170.0  1253.1  2350.7  246.9
Maryland 8.0  34.8  292.1  358.9  1400.0  3177.7  428.5
Massachusetts 3.1  20.8  169.1  231.6  1532.2  2311.3  1140.1
Michigan  9.3  38.9  261.9  274.6  1522.7  3159.0  545.5
Minnesota 2.7  19.5  85.9  85.8  1134.7  2559.3  343.1
Mississippi 14.3  19.6  65.7  189.1  915.6  1239.9  144.4
Missouri  9.6  28.3  189.0  233.5  1318.3  2424.2  378.4
Montana   5.4  16.7  39.2  156.8  804.9  2773.2  309.2
Nebraska  3.9  18.1  64.7  112.7  760.0  2316.1  249.1
Nevada   15.8  49.1  323.1  355.0  2453.1  4212.6  559.2
New Hampshire  3.2  10.7  23.2  76.0  1041.7  2343.9  293.4
New Jersey  5.6  21.0  180.4  185.1  1435.8  2774.5  511.5
New Mexico  8.8  39.1  109.6  343.4  1418.7  3008.6  259.5
New York  10.7  29.4  472.6  319.1  1728.0  2782.0  745.8
North Carolina 10.6  17.0  61.3  318.3  1154.1  2037.8  192.1
North Dakota 0.9  9.0  13.3  43.8  446.1  1843.0  144.7
Ohio   7.8  27.3  190.5  181.1  1216.0  2696.8  400.4
Oklahoma 8.6  29.2  73.8  205.0  1288.2  2228.1  326.8
Oregon  4.9  39.9  124.1  286.9  1636.4  3506.1  388.9
```
Pennsylvania 5.6 19.0 130.3 128.0 877.5 1624.1 333.2
Rhode Island 3.6 10.5 86.5 201.0 1489.5 2844.1 791.4
South Carolina 11.9 33.0 105.9 485.3 1613.6 2342.4 245.1
South Dakota 2.0 13.5 17.9 155.7 570.5 1704.4 147.5
Tennessee 10.1 29.7 145.8 143.9 1259.7 1776.5 314.0
Texas 13.3 33.8 152.4 208.2 1603.1 2988.7 397.6
Utah 3.5 20.3 68.8 147.3 1171.6 3004.6 334.5
Vermont 1.4 15.9 30.8 101.2 1348.2 2201.0 265.2
Virginia 4.3 106.2 165.7 986.2 2521.2 226.7
Washington 6.0 39.6 42.2 597.4 1341.7 163.3
West Virginia 2.8 12.9 92.1 101.2 1259.7 1776.5 314.0
Wisconsin 2.1 39.7 173.9 811.6 2772.2 282.0

These statements assume that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

The following statements invoke the PCA procedure, which requests a principal component analysis of the data and produces Figure 15.1 through Figure 15.4:

```plaintext
proc pca data=mycas.Crime;
run;
```

Figure 15.1 displays the “Model Information,” “Number of Variables,” “Number of Observations,” and “Simple Statistics” tables.

The “Model Information” table identifies the data source and shows that the principal component extraction method is eigenvalue decomposition, which is the default.

The “Number of Variables” table indicates that there are seven variables to be analyzed and seven principal components to be computed. By default, if you omit the VAR statement, all numeric variables that are not listed in other statements are used in the analysis.

The “Number of Observations” table shows that of the 50 observations in the input data, only 48 observations are used in the analysis, because some observations have incomplete data.

The “Simple Statistics” table displays the mean and standard deviation of the analysis variables.
Figure 15.1 continued

<table>
<thead>
<tr>
<th>Variable</th>
<th>Mean</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Murder</td>
<td>7.51667</td>
<td>3.93059</td>
</tr>
<tr>
<td>Rape</td>
<td>26.07500</td>
<td>10.81304</td>
</tr>
<tr>
<td>Robbery</td>
<td>127.55625</td>
<td>88.49374</td>
</tr>
<tr>
<td>Assault</td>
<td>214.58750</td>
<td>100.64360</td>
</tr>
<tr>
<td>Burglary</td>
<td>1316.37917</td>
<td>423.31261</td>
</tr>
<tr>
<td>Larceny</td>
<td>2696.88542</td>
<td>714.75023</td>
</tr>
<tr>
<td>Auto_Theft</td>
<td>383.97917</td>
<td>194.37033</td>
</tr>
</tbody>
</table>

Figure 15.2 displays the “Correlation Matrix” table. By default, the PROC PCA statement requests that principal components be computed from the correlation matrix, so the total variance is equal to the number of variables, 7.

<table>
<thead>
<tr>
<th>Correlation Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variable</td>
</tr>
<tr>
<td>Murder</td>
</tr>
<tr>
<td>Rape</td>
</tr>
<tr>
<td>Robbery</td>
</tr>
<tr>
<td>Assault</td>
</tr>
<tr>
<td>Burglary</td>
</tr>
<tr>
<td>Larceny</td>
</tr>
<tr>
<td>Auto_Theft</td>
</tr>
</tbody>
</table>

Figure 15.3 displays the “Eigenvalues” table. The first principal component accounts for about 57.8% of the total variance, the second principal component accounts for about 18.1%, and the third principal component accounts for about 10.7%. Note that the eigenvalues sum to the total variance.

The eigenvalues indicate that two or three components provide a good summary of the data: two components account for 76% of the total variance, and three components account for 87%. Subsequent components account for less than 5% each.

<table>
<thead>
<tr>
<th>Eigenvalues of the Correlation Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eigenvalue</td>
</tr>
<tr>
<td>------------</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
</tbody>
</table>
Figure 15.4 displays the “Eigenvectors” table. From the eigenvectors matrix, you can represent the first principal component, Prin1, as a linear combination of the original variables:

\[
\text{Prin1} = -0.30289 \times (\text{Murder}) \\
- 0.43410 \times (\text{Rape}) \\
- 0.39705 \times (\text{Robbery}) \\
\vdots \\
- 0.28834 \times (\text{Auto_Theft})
\]

Similarly, the second principal component, Prin2, is

\[
\text{Prin2} = 0.61893 \times (\text{Murder}) \\
+ 0.17053 \times (\text{Rape}) \\
- 0.04713 \times (\text{Robbery}) \\
\vdots \\
- 0.50400 \times (\text{Auto_Theft})
\]

where the variables are standardized.

The first component is a measure of the overall crime rate, because the first eigenvector shows approximately equal loadings on all variables. The second eigenvector has high negative loadings on the variables Auto_Theft and Larceny and high positive loadings on the variables Murder and Assault. There is also a small negative loading on the variable Burglary and a small positive loading on the variable Rape. This component seems to measure the preponderance of property crime compared to violent crime. The interpretation of the third component is not obvious.
Syntax: PCA Procedure

The following statements are available in the PCA procedure:

```
PROC PCA < options > ;
BY variables ;
CODE < options > ;
DISPLAY < table-list > ] < / options > ;
DISPLAYOUT table-spec-list ] < / options > ;
FREQ variable ;
OUTPUT OUT=CAS-libref.data-table
   < COPYVARS=(variables)> < keyword < =prefix >> . . . < keyword < =prefix >> ;
   PARTIAL variables ;
   VAR variables ;
   WEIGHT variable ;
```

The rest of this section provides detailed syntax information about each of the preceding statements, beginning with the PROC PCA statement. The remaining statements are described in alphabetical order.

PROC PCA Statement

```
PROC PCA < options > ;
```

The PROC PCA statement invokes the PCA procedure. Optionally, it also identifies the input and output data sets, specifies the analyses to be performed, and controls displayed output. Table 15.3 summarizes the options available in the PROC PCA statement.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify Data Sets</td>
<td>data sets options</td>
</tr>
<tr>
<td>DATA=</td>
<td>Specifies the CAS input data table</td>
</tr>
<tr>
<td>OUTSTAT=</td>
<td>Specifies the CAS output data table that contains various statistics</td>
</tr>
<tr>
<td>Specify Details of Analysis</td>
<td>analysis options</td>
</tr>
<tr>
<td>COV</td>
<td>Computes the principal components from the covariance matrix</td>
</tr>
<tr>
<td>METHOD=</td>
<td>Specifies the principal component extraction method to be used</td>
</tr>
<tr>
<td>N=</td>
<td>Specifies the number of principal components to be computed</td>
</tr>
<tr>
<td>NOINT</td>
<td>Omits the intercept from the model</td>
</tr>
<tr>
<td>PREFIX=</td>
<td>Specifies a prefix for naming the principal components</td>
</tr>
<tr>
<td>PARPREFIX=</td>
<td>Specifies a prefix for naming the residual variables</td>
</tr>
<tr>
<td>SINGULAR=</td>
<td>Specifies the singularity criterion</td>
</tr>
<tr>
<td>STD</td>
<td>Standardizes the principal component scores</td>
</tr>
<tr>
<td>VARDEF=</td>
<td>Specifies the divisor used to calculate variances and standard deviations</td>
</tr>
</tbody>
</table>
Table 15.3 continued

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify ODS Graphics Details</td>
<td>Specifies options that control the details of the plots</td>
</tr>
</tbody>
</table>

The following list provides details about these options.

**COVARIANCE**

**COV**

computes the principal components from the covariance matrix. If you omit this option, the correlation matrix is analyzed. The COV option causes variables that have large variances to be more strongly associated with components that have large eigenvalues, and it causes variables that have small variances to be more strongly associated with components that have small eigenvalues. You should not specify this option unless the units in which the variables are measured are comparable or the variables are standardized in some way.

**DATA=**CAS-libref.data-table

names the input data table for PROC PCA to use. The default is the most recently created data table. **CAS-libref.data-table** is a two-level name, where

- **CAS-libref** refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to the data, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about **CAS-libref**, see the section “Using CAS Sessions and CAS Engine Librefs” on page 707.

- **data-table** specifies the name of the input data table.

**METHOD=EIG | ITERGS< (iter-options) > | NIPALS< (nipals-options) > | RANDOM< (rand-options) >**

specifies the principal component extraction method to be used. You can specify the following values:

**EIG**

requests eigenvalue decomposition.

**ITERGS< (iter-options) >**

requests the iterative method based on Gram-Schmidt orthogonalization (ITERGS) of Andrecut (2009). You can specify the following **iter-options**.

- **EPSILON=n**
  
specifies the convergence criterion for the iterative method. By default, EPSILON=1E–12.

- **MAXITER=n**
  
specifies the maximum number of iterations for the iterative method. By default, MAX-ITER=500.
NOCENTER
suppresses centering of the numeric variables to be analyzed. This option is useful if the
analysis variables are already centered and scaled.

NOSCALE
suppresses scaling of the numeric variables to be analyzed. This option is useful if the
analysis variables are already centered and scaled.

NIPALS< (nipals-options)> requests the nonlinear iterative partial least squares (NIPALS) method. You can specify the following nipals-options.

EPSILON=n specifies the convergence criterion for the iterative method. By default, EPSILON=1E–12.

MAXITER=n specifies the maximum number of iterations for the iterative method. By default, MAX-ITER=500.

NOCENTER suppresses centering of the numeric variables to be analyzed. This option is useful if the
analysis variables are already centered and scaled.

NOSCALE suppresses scaling of the numeric variables to be analyzed. This option is useful if the
analysis variables are already centered and scaled.

RANDOM< (rand-options)> requests the iterative method based on random projection (RANDOM) of Halko, Martinsson, and Tropp (2011). You can specify the following rand-options.

NITER=n specifies the number of iterations. By default, NITER=1.

NOCENTER suppresses centering of the numeric variables to be analyzed. This option is useful if the
analysis variables are already centered and scaled.

NOSCALE suppresses scaling of the numeric variables to be analyzed. This option is useful if the
analysis variables are already centered and scaled.

SEED=n specifies the random number generator seed. By default, SEED=1.

By default, METHOD=EIG. If you specify METHOD=ITERGS, METHOD=NIPALS, or METHOD=RANDOM, the following options in the PROC PCA statement are not supported: COV, NOINT, OUTSTAT=, PARPREFIX=, SINGULAR=, and STD.
N=number
specifies the number of principal components to be computed. If you specify METHOD=ITERGS, METHOD=NIPALS, or METHOD=EIG, the default is the number of variables. If you specify METHOD=RANDOM, the default is 1. The number must be an integer greater than or equal to 0. If N=0, the default is used.

NOINT
omits the intercept from the model. In other words, this option requests that the covariance or correlation matrix not be corrected for the mean. When you specify this option, the covariance matrix and, hence, the standard deviations are not corrected for the mean. If you want to obtain the standard deviations corrected for the mean, you can obtain them by using a procedure such as PROC MEANS in Base SAS.

If you specify this option and also create an OUTSTAT= data table, the data table’s type is UCORR or UCOV rather than CORR or COV. For more information, see the section “OUTSTAT= Data Table” on page 726.

OUTSTAT=CAS-libref.data-table
creates an output data table to contain means, standard deviations, number of observations, correlations or covariances, eigenvalues, and eigenvectors. CAS-libref.data-table is a two-level name, where CAS-libref refers to the caslib and session identifier, and data-table specifies the name of the output data table. For more information about this two-level name, see the DATA= option and the section “Using CAS Sessions and CAS Engine Librefs” on page 707.

If you specify the COV option, the data table’s type is COV or UCOV, depending on the NOINT option, and it contains covariances; otherwise, the data table’s type is CORR or UCORR, depending on the NOINT option, and it contains correlations. If you specify the PARTIAL statement, the OUTSTAT= data table also contains R-squares.

For more information about the OUTSTAT= data table, see the section “OUTSTAT= Data Table” on page 726.

PLOTS < (global-plot-options) > <= plot-request < (options) >>
PLOTS < (global-plot-options) > <= (plot-request < (options) > <...plot-request < (options) >>)>
controls the plots that are produced through ODS Graphics. When you specify only one plot-request, you can omit the parentheses around it. Some examples follow:

plots
plots=none
plots=(eigen profile)
plots(unpack)=scree
plots=pattern(ncomp=3 circles=25 50 75 1)

ODS Graphics must be enabled before you can request plots. For example:

ods graphics on;
proc pca plots=all;
   var x1--x10;
run;
ods graphics off;
You must specify the PLOTS option in order for the PCA procedure to produce plots. If you do not specify a plot-request, PROC PCA produces the scree plot by default.

You can specify the following global-plot-options:

**UNPACKPANEL**

**UNPACK**

suppresses paneling in the scree plot. By default, multiple plots can appear in an output panel. Specify UNPACKPANEL if you want each plot to appear in a separate panel. You can specify PLOTS(UNPACKPANEL) to unpack the default plots. You can also specify UNPACKPANEL as a suboption of the SCREE option (such as PLOTS=SCREE(UNPACKPANEL)).

You can specify the following plot-requests:

**ALL**

produces all appropriate plots. You can specify other options along with ALL; for example, to request all plots and unpack only the scree plot, specify PLOTS=(ALL SCREE(UNPACKPANEL)).

**EIGEN < (UNPACKPANEL)>**

**EIGENVALUE < (UNPACKPANEL)>**

**SCREE < (UNPACKPANEL)>**

produces the scree plot of eigenvalues and proportion variance explained. By default, both plots appear in the same panel. Specify PLOTS=SCREE(UNPACKPANEL) if you want each plot to appear in a separate panel.

**NONE**

suppresses the display of all plots.

**PATTERN < (pattern-options)>**

produces the pairwise component pattern plots. Each variable is plotted as an observation whose coordinates are correlations between the variable and the two corresponding components in the plot. Use the NCOMP= option (for example, PLOTS=PATTERN(NCOMP=3)) as described in the following list to control the number of plots to display.

You can specify the following pattern-options:

**CIRCLES < number-list >**

plots the variance percentage circles. For each number \( c (0 < c \leq 1) \) that you specify, a \( (c \times 100\%) \) variance circle is displayed. For each number \( c (c > 1) \) that you specify, a \( c\% \) variance circle is displayed. You can specify either CIRCLES=0.05 1 or CIRCLES=5 100 to display 5% and 100% variance circles. PLOTS=PATTERN(CIRCLES) and PLOTS=PATTERN(VECTOR) both display a unit circle (100% variance). By default, no circle is displayed when you specify PLOTS=PATTERN.

**FLIP**

interchanges (flips) the X-axis and Y-axis dimensions of the component pattern plots.

**NCOMP=n**

specifies the number of components \( n (\geq 2) \) to be plotted. The default is 5 or the total number of components \( m (\geq 2) \), whichever is smaller. If \( n > m \), NCOMP=m is used. Be aware that the number of plots \( n \times (n - 1)/2 \) that are produced grows quadratically when \( n \) increases.
**VECTOR**
plots the pattern in a vector form.

**PATTERNPROFILE**
**PROFILE**
produces the pattern profile plot. Each component has its own profile. The Y-axis value represents the correlation between the variable (corresponding to the X-axis value) and the profiled principal component.

**PREFIX=** *name*
specifies a prefix for naming the principal components. By default, the names are *Prin1*, *Prin2*, ..., *Prin*<sub>*n*</sub>. If you specify PREFIX=Abc, the components are named *Abc1*, *Abc2*, *Abc3*, and so on. The number of characters in the prefix plus the number of digits required to designate the variables should not exceed the current name length that is defined by the VALIDVARNAMES system option.

**PARPREFIX=** *name*
**PPREFIX=** *name*
**RPREFIX=** *name*
specifies a prefix for naming the residual variables in the OUTSTAT= data table. By default, the prefix is *R_*<sub>_</sub>. The number of characters in the prefix plus the maximum length of the variable names should not exceed the current name length that is defined by the VALIDVARNAMES system option.

**SINGULAR=** *p*
**SING=** *p*
specifies the singularity criterion, where 0 < *p* < 1. If a variable in a PARTIAL statement has an R-square as large as 1 − *p* when predicted from the variables listed before it in the statement, the variable is assigned a standardized coefficient of 0. By default, SINGULAR=1E−8.

**STANDARD**
**STD**
standardizes the principal component scores in the OUT= data table in the OUTPUT statement to unit variance. If you omit this option, the scores have a variance equal to the corresponding eigenvalue. Note that this option has no effect on the eigenvalues themselves.

**VARDEF=** *DF | N | WDF | WEIGHT | WGT*
specifies the divisor to be used to calculate variances and standard deviations. By default, VARDEF=DF. The following table displays the values and associated divisors:

<table>
<thead>
<tr>
<th>Value</th>
<th>Divisor</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>DF</td>
<td>Error degrees of freedom</td>
<td>( n - i ) (before partialing) \n( n - p - i ) (after partialing)</td>
</tr>
<tr>
<td>N</td>
<td>Number of observations</td>
<td>( n )</td>
</tr>
<tr>
<td>WEIGHT</td>
<td>Sum of weights</td>
<td>( \sum_{j=1}^{n} w_j )</td>
</tr>
<tr>
<td>WDF</td>
<td>Sum of weights minus one</td>
<td>( \left( \sum_{j=1}^{n} w_j \right) - i ) (before partialing) \n( \left( \sum_{j=1}^{n} w_j \right) - p - i ) (after partialing)</td>
</tr>
</tbody>
</table>
In the formulas for VARDEF=DF and VARDEF=WDF, $p$ is the number of degrees of freedom of the variables in the PARTIAL statement, and $i$ is 0 if you specify the NOINT option and 1 otherwise.

**BY Statement**

```sas
BY variables ;
```

You can specify a BY statement in PROC PCA to obtain separate analyses of observations in groups that are defined by the values of the BY variables. If you specify more than one BY statement, only the last one specified is used. For more information, see the discussion of BY-group processing in SAS Language Reference: Concepts.

**CODE Statement**

```sas
CODE <options> ;
```

The CODE statement writes SAS DATA step code for computing the principal component scores either to a file or to a catalog entry. This code can then be included in a DATA step to score new data.

The CODE statement is not supported when you specify the PARTIAL statement. If you specify more than one CODE statement, only the first one specified is used.

Table 15.4 summarizes the options available in the CODE statement.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMMENT</td>
<td>Adds comments to the generated code</td>
</tr>
<tr>
<td>FILE=</td>
<td>Names the file where the generated code is saved</td>
</tr>
<tr>
<td>FORMATWIDTH=</td>
<td>Specifies the numeric format width for the regression coefficients</td>
</tr>
<tr>
<td>INDENTSIZE=</td>
<td>Specifies the number of spaces to indent the generated code</td>
</tr>
<tr>
<td>LABELID=</td>
<td>Specifies a number used to construct names and labels</td>
</tr>
<tr>
<td>LINESIZE=</td>
<td>Specifies the line size for the generated code</td>
</tr>
<tr>
<td>NOTRIM</td>
<td>Compares formatted values, including blank padding</td>
</tr>
</tbody>
</table>

For more information about the syntax of the CODE statement, see the section “CODE Statement” on page 16 in Chapter 3, “Shared Concepts.”

**DISPLAY Statement**

```sas
DISPLAY <table-list> </options> ;
```

The DISPLAY statement enables you to specify a list of display tables to display or exclude. This statement is similar to the ODS SELECT, ODS EXCLUDE, and ODS TRACE statements. However, the DISPLAY
statement can improve performance when a large number of tables could be generated (such as in BY-group processing). The procedure processes the DISPLAY statement on a CAS server and thus sends only a subset of ODS tables to the SAS client. Because ODS statements are processed on a SAS client, first all the generated display tables are sent to the client, and then the client creates a subset.

If you use both DISPLAY and ODS statements together, the DISPLAY statement takes precedence over the ODS statements. Note that the ODS EXCLUDE statement processes tables that are sent to the client after they have been filtered by the DISPLAY statement. In some cases, it might appear that the ODS EXCLUDE statement is taking precedence because it can further filter the tables. For more information about ODS, see *SAS Output Delivery System: Procedures Guide*.

You can specify the `table-list` as a list of table names, paths, partial pathnames, and regular expressions.

The table names that you can specify are listed in the section “ODS Table Names” on page 729. A path is a table name that is prefixed with dot-separated grouping information. For example, a `SelectionSummary` table that a procedure produces during a selection routine might have the path `Bygroup1.Summary:SelectionSummary`. A partial pathname does not include all groups; for example, `SelectionSummary` and `Summary:SelectionSummary` are partial pathnames for `Bygroup1.Summary:SelectionSummary`.

When you specify a table name or partial pathname, all display tables whose paths end in the specified name are selected for display or exclusion. For example, both `SelectionSummary` and `Summary:SelectionSummary` select `Bygroup1.Summary:SelectionSummary`.

A regular expression is enclosed in forward slashes (/). For example, specifying “/tions/” selects all pathnames that contain the substring “tions”; in particular, the `Bygroup1.Summary:SelectionSummary` table is selected. Specifying “!/tions/” selects all pathnames that do not contain the substring “tions”; in particular, the `Bygroup1.Summary:SelectionSummary` table is not selected.

You can specify the following `options` after a slash (/):

**CASESENSITIVE**
performs a case-sensitive comparison of table names in the `table-list` to display table names when tables are subsetted for display. To preserve case, you must enclose table names in the `table-list` in quotation marks.

**EXCLUDE**
displays all display tables except those that you specify in the `table-list`.

**EXCLUDEALL**
suppresses display of all tables. This option takes precedence over the other options.

**TRACE**
displays the display table names, labels, and paths.

---

**DISPLAYOUT Statement**

```
DISPLAYOUT table-spec-list < /options >;
```

The DISPLAYOUT statement enables you to create CAS output tables from your displayed output. This statement is similar to the ODS OUTPUT statement. For more information about ODS, see *SAS Output Delivery System: Procedures Guide*. 
The **table-spec-list** specifies a list of CAS output tables to create. Each entry in the list has either a `key=value` format or a `key` format:

- `key=value` specifies `key` as the ODS table name, path, or partial pathname, and specifies `value` as the CAS output table name.
- `key` specifies `key` as the ODS table name and also as the CAS output table name.

The ODS table names that you can specify are listed in the section “ODS Table Names” on page 729. You cannot specify the ODS table named `OutputCasTables` in the `table-spec-list`.

Table names and partial pathnames are discussed under the `DISPLAY` statement. The DISPLAYOUT statement does not support regular expressions.

You can specify the following `options` after a slash (/):

- **INCLUDEALL**
  - creates output CAS tables for all display tables. The name of the created output CAS table is the same as the corresponding display table name. If you specify this option, the `table-spec-list` specification is ignored.

- **NOREPLACE**
  - does not replace any existing CAS output table of the same name.

- **REPEATED**
  - replicates all CAS output tables on all nodes.

---

**FREQ Statement**

```
FREQ variable ;
```

The `variable` in the FREQ statement identifies a numeric variable in the data set that contains the frequency of occurrence of each observation. PROC PCA treats each observation as if it appears \( f \) times, where \( f \) is the value of the FREQ `variable` for the observation. If \( f \) is not an integer, it is truncated to an integer. If \( f \) is less than 1 or missing, the observation is not used in the analysis. When the FREQ statement is not specified, each observation is assigned a frequency of 1.

The FREQ statement is not supported if you specify `METHOD=ITERGS`, `METHOD=NIPALS`, or `METHOD=RANDOM` in the PROC PCA statement.

---

**OUTPUT Statement**

```
OUTPUT OUT=CAS-libref.data-table
    < COPYVARS=(variables) >
    < keyword <=prefix>=prefix > . . . < keyword <=prefix>=prefix > ;
```

The OUTPUT statement creates a data table that contains observationwise statistics, which are computed after PROC PCA fits the model. If you do not specify a `keyword`, then only the principal component scores are included.
The variables in the input data table are *not* included in the output data table, in order to avoid data duplication for large data tables; however, variables that you specify in the COPYVARS= option are included.

You must specify the following option:

```
OUT=CAS-libref.data-table
```

names the output data table for PROC PCA to use. You must specify this option before any other options. *CAS-libref.data-table* is a two-level name, where

- **CAS-libref** refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to where the data table is to be stored, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about *CAS-libref*, see the section “Using CAS Sessions and CAS Engine Librefs” on page 707.

- **data-table** specifies the name of the output data table.

You can also specify the following syntax elements:

```
COPYVAR=variable
COPYVARS=(variables)
```

transfers one or more *variables* from the input data table to the output data table.

```
keyword < =prefix >
```

specifies a statistic to include in the output data table and optionally a *prefix* for naming the output variables. If you do not provide a *prefix*, the PCA procedure assigns a default prefix based on the type of statistic requested. For example, for the VAR statement variables x1 and x2, RESIDUAL produces two residual value variables, R_x1 and R_x2.

You can specify the following *keywords* to add statistics to the OUTPUT data table:

- **H**
  requests the approximate leverage. The default prefix is H.

- **STD**
  requests standardized (centered and scaled) VAR statement variable values for each VAR statement variable. The default prefix is Std.

- **STDSSE**
  requests the sum of squares of residuals for standardized VAR statement variables. The default prefix is StdSSE.

- **TSQUARE**
- **T2**
  requests scaled sum of squares of score values. The default prefix is TSquare.

- **RESIDUAL**
- **RESID**
- **R**
  requests residuals for each VAR statement variable. The default prefix is R.
Chapter 15: The PCA Procedure

SCORE requests principal component scores for each principal component. The default prefix is Score.

If you specify METHOD=EIG, the only valid keywords are RESIDUAL (if you also specify the PARTIAL statement; PROC PCA computes the residuals by predicting the VAR statement variables from the PARTIAL statement variables) and SCORE. Other keywords are ignored. The output variables that contain the principal component scores have mean 0 and a variance equal to the corresponding eigenvalue, unless you specify the STANDARD option to standardize the scores to unit variance. Also, if you specify the COV option, PROC PCA computes the principal component scores from the corrected variables or (if the NOINT option is specified) uncorrected variables rather than from the standardized variables.

The output variables that contain the requested statistic are named as follows, according to the keyword that you specify:

- The keywords RESIDUAL and STD define an output variable for each VAR statement variable, so the variables that correspond to each VAR statement variable are named by appending the name of the VAR statement variable to the prefix. For example, if the model has the VAR statement variables x1 and x2, then RESIDUAL=R produces the variables R_x1 and R_x2.
- The keyword SCORE defines an output variable for each principal component, so the variables that correspond to each successive component are named by appending the component number to the prefix. For example, if the model has three principal components, then SCORE=T produces the variables T1, T2, and T3.
- The keywords H, STDSSE, and TSQUARE each define a single output variable, so the variable name matches the prefix.

PARTIAL Statement

PARTIAL variables ;

If you want to analyze a partial correlation or covariance matrix, use the PARTIAL statement to specify the names of the numeric variables to be partialed out. The PCA procedure computes the principal components of the residuals by predicting the VAR statement variables from the PARTIAL statement variables. If you request an OUTSTAT= data table, the residual variables are named by prefixing either the characters R_ (by default) or the string specified in the PARPREFIX= option to the VAR statement variable names.

The PARTIAL statement is not supported if you specify METHOD=ITERGS, METHOD=NIPALS, or METHOD=RANDOM in the PROC PCA statement.

VAR Statement

VAR variables ;

The VAR statement lists the numeric variables to be analyzed. If you omit the VAR statement, all numeric variables that are not specified in other statements are analyzed.
WEIGHT Statement

`WEIGHT variable ;`

The `variable` in the WEIGHT statement is used as a weight to perform a weighted analysis of the data. Observations that have nonpositive or missing weights are not included in the analysis. If a WEIGHT statement is not included, all observations that are used in the analysis are assigned a weight of 1.

The WEIGHT statement is not supported if you specify METHOD=ITERGS, METHOD=NIPALS, or METHOD=RANDOM in the PROC PCA statement.

Details: PCA Procedure

Computing Principal Components

The PCA procedure implements several methods of calculating principal components: eigenvalue decomposition (EIG), the nonlinear iterative partial least squares (NIPALS) method, the iterative method based on the Gram-Schmidt orthogonalization (ITERGS) of Andrecut (2009), and the iterative method based on random projection (RANDOM) of Halko, Martinsson, and Tropp (2011). The EIG, NIPALS, and ITERGS methods provide exact principal component solutions, whereas the RANDOM method gives approximate principal components.

Each method has computational strengths; Table 15.5 provides guidelines for selecting an efficient (fast) method. The guidelines include the concept of a “small to moderate” and “moderate to large” number of variables. This concept depends on the number of data set observations, the number of principal components required, the PCA method parameter settings, and the grid computing environment. As a simple illustration of the efficiency relationship between the EIG and RANDOM methods, consider a scenario in which one principal component is extracted from a data set that consists of 1,000,000 observations that are distributed across a grid of 16 servers. Figure 15.5 depicts execution time for RANDOM divided by the execution time of EIG (labeled “Execution Time Ratio”) as a function of the number of data set variables. Where execution time ratio is greater than 1, EIG is more efficient than RANDOM. For this scenario, “small to moderate” means approximately 700 variables or less and “moderate to large” means anything more than 700 variables.

<table>
<thead>
<tr>
<th>Method</th>
<th>Efficient in These Scenarios</th>
</tr>
</thead>
<tbody>
<tr>
<td>EIG</td>
<td>Small to moderate number of variables, and/or computing all principal components</td>
</tr>
<tr>
<td>NIPALS</td>
<td>Computing only the first few principal components</td>
</tr>
<tr>
<td>ITERGS</td>
<td>Computing only the first few principal components</td>
</tr>
<tr>
<td>RANDOM</td>
<td>Moderate to large number of variables</td>
</tr>
</tbody>
</table>
Eigenvalue Decomposition

Let $X$ be a centered and scaled data matrix that has $k$ numeric variables. The eigenvalue decomposition method bases the component extraction on the eigenvalue decomposition of the covariance matrix $X'X$, which extracts all the $k$ principal components simultaneously. Each principal component is a linear combination of the original variables, and each component is orthogonal, with coefficients equal to the eigenvectors of the covariance matrix $X'X$. The eigenvectors are usually normalized to have unit length. The principal components are sorted by descending order of the eigenvalues, which are equal to the variances of the components.

NIPALS

The nonlinear iterative partial least squares (NIPALS) method extracts the principal components successively based on the data matrix $X$. The NIPALS method starts by calculating the loadings, $p$, as $p = (t't)^{-1}t'X$, where $t$ is the score vector. It then calculates an improved score vector, $t = Xp$. The method iteratively computes the improved $p$ and $t$ until convergence is reached.

This process accounts for how the first principal component is extracted. The second component is extracted in the same way, by replacing $X$ with the residual from the first component: $E = X - tp'$.

For large data matrices or matrices that have a high degree of column collinearity, the NIPALS method suffers from loss of orthogonality because of the machine-precision errors that accumulate at each iteration step. In practice, the NIPALS method is used to extract only the first few principal components.
**ITERGS**

The iterative method based on Gram-Schmidt orthogonalization (ITERGS) of Andrecut (2009) overcomes the issue of loss of orthogonality in the NIPALS method by applying Gram-Schmidt reorthogonalization correction to both the loadings and the scores at each iteration step:

\[
\begin{align*}
  p_c &= p - P_k P_k^t p \\
  t_c &= t - T_k T_k^t t
\end{align*}
\]

Here, \( p_c \) and \( t_c \) are the corrected loading vector and score vector, respectively. \( P_k \) is the matrix that is formed by using the first \( k \) loadings. \( T_k \) is the matrix that is formed by using the first \( k \) scores.

The ITERGS method stabilizes the iterative process at the cost of increased computational effort.

**RANDOM**

The iterative method based on random projection (RANDOM) of Halko, Martinsson, and Tropp (2011) computes approximate scores and loadings by using a two-step process. The first step calculates an approximate basis for the range of \( X \) (that is, \( X \approx QQ^T X \)), where \( Q \) has orthonormal columns. The second step computes a sketch of \( X \) from \( B = Q^T X \). The singular value decomposition of \( B \) gives an equivalent representation \( USV^T \), and simple manipulations of these matrices provide the approximate loadings \( P = V \) and the scores \( T = QS \).

The iterative part of this method is the construction of the orthonormal basis matrix \( Q \). Conceptually, you can obtain \( Q \) by QR factorization of \( \tilde{Y} = (XX^T)^i X \Omega \), where \( \Omega \) is a random Gaussian projection matrix and \( i \) is the number of iterations. Here the notion of iteration refers to multiplication of a sample matrix by \( X^T \) and then \( X \). This conceptual approach is susceptible to rounding error, so in practice you orthonormalize the columns of the sample matrix for numerical stability. Here, the initial sample matrix is the subspace defined by the QR factorization of a random projection of the data matrix \( X \Omega = Q_0 R_0 \). Each iteration \( (j = 1, 2, \ldots, i) \) then consists of alternating projection and orthonormalization steps of

\[
\begin{align*}
  \tilde{Y}_j &= X^T Q_{j-1} \\
  \tilde{Q}_j \tilde{R}_j &= \tilde{Y}_j
\end{align*}
\]

and

\[
\begin{align*}
  Y_j &= X \tilde{Q}_j \\
  Q_j R_j &= Y_j
\end{align*}
\]

When \( j = i \), \( Q_j \) is the desired orthonormal basis matrix.

**Missing Values**

Observations that have missing values for any variable in the VAR, PARTIAL, FREQ, or WEIGHT statement are omitted from the analysis and are given missing values for principal component scores in the OUT= data table that is specified in the OUTPUT statement.
The following table relates the type of the OUTSTAT= data table to the options that are specified in the PROC PCA statement:

<table>
<thead>
<tr>
<th>Options</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>CORR</td>
</tr>
<tr>
<td>COV</td>
<td>COV</td>
</tr>
<tr>
<td>NOINT</td>
<td>UCORR</td>
</tr>
<tr>
<td>COV NOINT</td>
<td>UCOV</td>
</tr>
</tbody>
</table>

Note that the default (neither the COV nor NOINT option) produces a data table of type CORR.

The new data table contains the following variables:

- the BY variables, if any
- two new variables, `_TYPE_` and `_NAME_`, both character variables
- the variables that are analyzed (that is, those in the VAR statement); or, if there is no VAR statement, all numeric variables not listed in any other statement; or, if there is a PARTIAL statement, the residual variables that contain the residuals computed by predicting the VAR statement variables from the PARTIAL statement variables.

Each observation in the new data table contains some type of statistic, as indicated by the `_TYPE_` variable. The values of the `_TYPE_` variable are as follows:

<table>
<thead>
<tr>
<th><code>_TYPE_</code></th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>MEAN</td>
<td>mean of each variable. If you specify the PARTIAL statement, this observation is omitted.</td>
</tr>
<tr>
<td>STD</td>
<td>standard deviations. If you specify the COV option, this observation is omitted. If you specify the PARTIAL statement, the standard deviation of a variable is computed as its root mean squared error as predicted from the PARTIAL statement variables.</td>
</tr>
<tr>
<td>USTD</td>
<td>uncorrected standard deviations. When you specify the NOINT option in the PROC PCA statement, the OUTSTAT= data table contains standard deviations not corrected for the mean. However, if you also specify the COV option in the PROC PCA statement, this observation is omitted.</td>
</tr>
<tr>
<td>N</td>
<td>number of observations on which the analysis is based. This value is the same for each variable. If you specify the PARTIAL statement and the value of the VARDEF= option is DF or unspecified, then the number of observations is decremented by the degrees of freedom for the PARTIAL statement variables.</td>
</tr>
</tbody>
</table>
| SUMWGT   | the sum of the weights of the observations. This value is the same for each variable. If you specify the PARTIAL statement and VARDEF=WDF, then the sum of the weights is decremented by the degrees of freedom for the PARTIAL statement variables. This observation is output only if the value is different from that in the observation for which `_TYPE_` = ‘N’.
OUTSTAT= Data Table ♦ 727

**CORR** correlations between each variable and the variable specified by the _NAME_ variable. The number of observations for which _TYPE_ = 'CORR' is equal to the number of variables being analyzed. If you specify the COV option, no _TYPE_ = 'CORR' observations are produced. If you use the PARTIAL statement, then the partial correlations, not the raw correlations, are output.

**UCORR** uncorrected correlation matrix. When you specify the NOINT option without the COV option in the PROC PCA statement, the OUTSTAT= data table contains a matrix of correlations not corrected for the means. However, if you also specify the COV option in the PROC PCA statement, this observation is omitted.

**COV** covariances between each variable and the variable specified by the _NAME_ variable. _TYPE_ = 'COV' observations are produced only if you specify the COV option. If you specify the PARTIAL statement, the partial covariances, not the raw covariances, are output.

**UCOV** uncorrected covariance matrix. When you specify the NOINT and COV options in the PROC PCA statement, the OUTSTAT= data table contains a matrix of covariances not corrected for the means.

**EIGENVAL** eigenvalues. If the N= option requests less than the maximum number of principal components, only the specified number of eigenvalues are produced, and missing values fill out the observation.

**SCORE** eigenvectors. The _NAME_ variable contains the name of the corresponding principal component as constructed from the PREFIX= option. The number of observations for which _TYPE_ = 'SCORE' equals the number of principal components computed. The eigenvectors have unit length unless you specify the STD option, in which case the unit-length eigenvectors are divided by the square roots of the eigenvalues to produce scores that have unit standard deviations.

To obtain the principal component scores, if the COV option is not specified, these coefficients should be multiplied by the standardized data. For the COV option, these coefficients should be multiplied by the centered data. To center and standardize the data, you should use means that are obtained from the observation for which _TYPE_ = 'MEAN' and standard deviations that are obtained from the observation for which _TYPE_ = 'STD'.

**USCORE** scoring coefficients to be applied without subtracting the mean from the raw variables. Observations for which _TYPE_ = 'USCORE' are produced when you specify the NOINT option in the PROC PCA statement.

To obtain the principal component scores, these coefficients should be multiplied by the data that are standardized by the uncorrected standard deviations obtained from the observation for which _TYPE_ = 'USTD'.

**RSQUARED** R-squares for each VAR statement variable as predicted by the PARTIAL statement variables.

**B** regression coefficients for each VAR statement variable as predicted by the PARTIAL statement variables. This observation is produced only if you specify the COV option.

**STB** standardized regression coefficients for each VAR statement variable as predicted by the PARTIAL statement variables. If you specify the COV option, this observation is omitted.
Display Output

The following sections describe the output that PROC PCA produces. The output is organized into various tables, which are discussed in their order of appearance.

Model Information

The “Model Information” table displays basic information about the model, including the input data table and the principal component extraction method that is used in the analysis.

Number of Variables

The “Number of Variables” table displays the number of VAR statement variables, the number of PARTIAL statement variables, and the number of principal components to be extracted.

Number of Observations

The “Number of Observations” table displays the number of observations read from the input data table and the number of observations used in the analysis. If you specify a FREQ statement, the table also displays the sum of frequencies read and used.

Simple Statistics

If you specify METHOD=EIG, the PCA procedure produces a “Simple Statistics” table that displays the mean and standard deviation (std) for each variable. If you specify the NOINT option, the uncorrected standard deviation (ustd) is displayed.

Centering and Scaling Information

If you specify METHOD=ITERGS, METHOD=NIPALS, or METHOD=RANDOM, the PCA procedure produces a “Centering and Scaling Information” table that displays the centering and scaling information for each variable.

Explained Variation of Variables

If you specify METHOD=ITERGS, METHOD=NIPALS, or METHOD=RANDOM, the PCA procedure produces an “Explained Variation of Variables” table that displays the fraction of variation that is accounted for in each variable by each successive principal component.

Correlation Matrix

If you specify METHOD=EIG, the PCA procedure produces a “Correlation Matrix” table that displays the correlation matrix or, if you specify the COV option, the covariance matrix.

Regression Statistics

When you specify the PARTIAL statement, the PCA procedure produces a “Regression Statistics” table that displays the R-square and root mean squared error (RMSE) for each VAR statement variable as predicted by the PARTIAL statement variables.
Regression Coefficients
When you specify the PARTIAL statement, the PCA procedure produces a “Regression Coefficients” table that displays standardized regression coefficients or, if you specify the COV option, regression coefficients for predicting the VAR statement variables from the PARTIAL statement variables.

Partial Correlation Matrix
When you specify the PARTIAL statement, the PCA procedure produces a “Partial Correlation Matrix” table that displays the partial correlation matrix or, if you specify the COV option, the partial covariance matrix.

Total Variance
If you specify METHOD=EIG and the COV option, the PCA procedure produces a simple table that displays the total variance.

Eigenvalues
The “Eigenvalues” table displays eigenvalues of the correlation or covariance matrix (if you specify METHOD=EIG) or eigenvalues of the data matrix (if you specify METHOD=ITERGS, METHOD=NIPALS, or METHOD=RANDOM), along with the difference between successive eigenvalues, the proportion of variance explained by each eigenvalue, and the cumulative proportion of variance explained.

Eigenvectors
If you specify METHOD=EIG, the PCA procedure produces an “Eigenvectors” table that displays the eigenvectors.

Loadings
If you specify METHOD=ITERGS, METHOD=NIPALS, or METHOD=RANDOM, the PCA procedure produces a “Loadings” table that displays the loadings.

Timing Information
The “Timing” table displays the elapsed time of each main task of the procedure.

ODS Table Names
Each table that the PCA procedure creates has a name associated with it. You must use this name to refer to the table when you use the DISPLAY statement, the DISPLAYOUT statement, or the Output Delivery System (ODS) statements. These names are listed in Table 15.6.

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Required Statement / Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>CenScaleInfo</td>
<td>Centering and scaling information</td>
<td>METHOD=ITERGS</td>
</tr>
</tbody>
</table>
### Table 15.6 continued

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Required Statement / Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>Corr</td>
<td>Correlation matrix</td>
<td>METHOD=EIG</td>
</tr>
<tr>
<td>Cov</td>
<td>Covariance matrix</td>
<td>METHOD=EIG and COV</td>
</tr>
<tr>
<td>Eigenvalues</td>
<td>Eigenvalues</td>
<td>Default output</td>
</tr>
<tr>
<td>Eigenvectors</td>
<td>Eigenvectors</td>
<td>METHOD=EIG</td>
</tr>
<tr>
<td>Loadings</td>
<td>Loadings</td>
<td>METHOD=ITERGS</td>
</tr>
<tr>
<td>ModelInfo</td>
<td>Model information</td>
<td>Default output</td>
</tr>
<tr>
<td>NObs</td>
<td>Number of observations read and used</td>
<td>Default output</td>
</tr>
<tr>
<td>NVars</td>
<td>Number of variables, partial variables, and principal components</td>
<td>Default output</td>
</tr>
<tr>
<td>OutputCasTables</td>
<td>A special ODS table that has information about all the CAS tables that are created during a CAS action execution</td>
<td>DISPLAYOUT statement / OUTPUT statement / OUTSTAT=</td>
</tr>
<tr>
<td>ParCorr</td>
<td>Partial correlation matrix</td>
<td>PARTIAL statement</td>
</tr>
<tr>
<td>ParCov</td>
<td>Uncorrected partial covariance matrix</td>
<td>PARTIAL statement and COV</td>
</tr>
<tr>
<td>RegCoef</td>
<td>Regression coefficients</td>
<td>PARTIAL statement and COV</td>
</tr>
<tr>
<td>RSquareRMSE</td>
<td>Regression statistics: R-squares and RMSEs</td>
<td>PARTIAL statement</td>
</tr>
<tr>
<td>SimpleStatistics</td>
<td>Simple statistics</td>
<td>METHOD=EIG</td>
</tr>
<tr>
<td>StdRegCoef</td>
<td>Standardized regression coefficients</td>
<td>PARTIAL statement</td>
</tr>
<tr>
<td>Timing</td>
<td>Absolute and relative times of tasks that are performed by the procedure</td>
<td>Default output</td>
</tr>
<tr>
<td>TotalVariance</td>
<td>Total variance</td>
<td>METHOD=EIG and COV</td>
</tr>
<tr>
<td>Variation</td>
<td>Explained variation of variables</td>
<td>METHOD=ITERGS</td>
</tr>
</tbody>
</table>

### ODS Graphics

You can refer to every graph produced through ODS Graphics by name. The names of the graphs that PROC PCA generates are listed in Table 15.7, along with a description of each graph and the required statements and options.

### Table 15.7 Graphics Produced by PROC PCA

<table>
<thead>
<tr>
<th>ODS Graph Name</th>
<th>Plot Description</th>
<th>Required Statement / Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>PatternPlot</td>
<td>Component pattern plot</td>
<td>PLOTS=PATTERN</td>
</tr>
<tr>
<td>PatternProfilePlot</td>
<td>Component pattern profile plot</td>
<td>PLOTS=PATTERNPROFILE</td>
</tr>
<tr>
<td>ScreePlot</td>
<td>Scree and variance plots</td>
<td>Default output</td>
</tr>
<tr>
<td>VariancePlot</td>
<td>Variance proportion-explained plot</td>
<td>PLOTS=SCREE(UNPACKPANEL)</td>
</tr>
</tbody>
</table>
Examples: PCA Procedure

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

Example 15.1: Analyzing Mean Temperatures of US Cities

This example analyzes mean daily temperatures of selected US cities in January and July. The following DATA step creates the data:

```sas
data mycas.Temperature;
  length Cityid $ 2;
  title 'Mean Temperature of Selected Cities in January and July';
  input City $1-15 January July;
  Cityid = substr(City,1,2);
  datalines;
  Mobile  51.2  81.6
  Phoenix 51.2  91.2
  Little Rock 39.5  81.4
  Sacramento 45.1  75.2
  Denver  29.9  73.0
  ... more lines ...
  Cheyenne 26.6  69.1
;
```

The following statements invoke the PCA procedure, which requests a principal component analysis of these data and outputs the scores to an output data table. The Cityid variable is also included in the output data table (COPYVARS= Cityid).

```sas
  title 'Mean Temperature of Selected Cities in January and July';
  proc pca data=mycas.Temperature cov;
    var July January;
    output out=mycas.Scores copyvars=Cityid;
  run;
```

Output 15.1.1 displays the PROC PCA output. The standard deviation of January (11.712) is higher than the standard deviation of July (5.128). The COV option in the PROC PCA statement requests that the principal components be computed from the covariance matrix. The total variance is 163.474. The first principal component accounts for about 94% of the total variance, and the second principal component accounts for only about 6%. The eigenvalues sum to the total variance.

Note that January receives a higher loading on Prin1 because it has a higher standard deviation than July. Also note that the PCA procedure calculates the scores by using the centered variables rather than the standardized variables.
Output 15.1.1 Results of Principal Component Analysis

Mean Temperature of Selected Cities in January and July

The PCA Procedure

<table>
<thead>
<tr>
<th>Model Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source</td>
</tr>
<tr>
<td>Component Extraction Method</td>
</tr>
</tbody>
</table>

| Number of Variables | 2 |
| Number of Principal Components | 2 |

| Number of Observations Read | 64 |
| Number of Observations Used | 64 |

<table>
<thead>
<tr>
<th>Simple Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variable</td>
</tr>
<tr>
<td>July</td>
</tr>
<tr>
<td>January</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Covariance Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variable</td>
</tr>
<tr>
<td>July</td>
</tr>
<tr>
<td>January</td>
</tr>
</tbody>
</table>

| Total Variance | 163.47356647 |

<table>
<thead>
<tr>
<th>Eigenvalues of the Covariance Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eigenvalue</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Eigenvectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variable</td>
</tr>
<tr>
<td>July</td>
</tr>
<tr>
<td>January</td>
</tr>
</tbody>
</table>

Example 15.2: Extracting Principal Components with NIPALS

This example demonstrates the NIPALS method in PROC PCA, which extracts principal components successively. The data that this example uses are from the Getting Started section; they provide crime rates per 100,000 people in seven categories for each of the 50 US states in 1977. The following DATA step generates the data:

data mycas.Crime;
    title 'Crime Rates per 100,000 Population by State';
    input State $1-15 Murder Rape Robbery Assault Burglary Larceny Auto_Theft;
    datalines;
Example 15.2: Extracting Principal Components with NIPALS

... more lines ...

The following statements use PROC PCA to extract principal components by using the NIPALS method, and produce default plots:

```sas
ods graphics on;
proc pca data=mycas.Crime method=nipals plots;
run;
```

Output 15.2.1 displays the PROC PCA output. The “Model Information” table shows that the NIPALS method is used to extract principal components. The “Explained Variation of Variables” table lists the fraction of variation that is accounted for in each variable by each of the seven principal components. All the variation in each variable is accounted for by seven principal components because there are only seven variables. The eigenvalues indicate that two or three components provide a good summary of the data: two components account for 76% of the total variance, and three components account for 87%. Subsequent components account for less than 5% each.

Note that in the Getting Started section, the principal components are extracted from the same data by using the eigenvalue decomposition method; the “Eigenvalues” table that is generated there matches the one generated by the NIPALS method. Also, the eigenvectors in the “Eigenvectors” table match the loading factors in the “Loadings” table.

Output 15.2.1 Results of Principal Component Analysis Using NIPALS

<table>
<thead>
<tr>
<th>Crime Rates per 100,000 Population by State</th>
</tr>
</thead>
<tbody>
<tr>
<td>The PCA Procedure</td>
</tr>
<tr>
<td>Model Information</td>
</tr>
<tr>
<td>Data Source CRIME</td>
</tr>
<tr>
<td>Component Extraction Method NIPALS</td>
</tr>
<tr>
<td>Number of Variables 7</td>
</tr>
<tr>
<td>Number of Principal Components 7</td>
</tr>
<tr>
<td>Number of Observations Read 50</td>
</tr>
<tr>
<td>Number of Observations Used 48</td>
</tr>
</tbody>
</table>

Alabama 14.2 25.2 96.8 278.3 1135.5 1881.9 280.7
Alaska 10.8 51.6 96.8 284.0 1331.7 3369.8 753.3
Arizona 9.5 34.2 138.2 312.3 2346.1 4467.4 439.5
Arkansas 8.8 27.6 83.2 203.4 972.6 1862.1 183.4
California 11.5 49.4 287.0 358.0 2139.4 3499.8 663.5

... more lines ...

Wisconsin 2.8 12.9 52.2 63.7 846.9 2614.2 220.7
Wyoming . 21.9 39.7 173.9 811.6 2772.2 282.0

;
### Centering and Scaling Information

<table>
<thead>
<tr>
<th>Variable</th>
<th>Subtracted off</th>
<th>Divided by</th>
</tr>
</thead>
<tbody>
<tr>
<td>Murder</td>
<td>7.51667</td>
<td>3.93059</td>
</tr>
<tr>
<td>Rape</td>
<td>26.07500</td>
<td>10.81304</td>
</tr>
<tr>
<td>Robbery</td>
<td>127.55625</td>
<td>88.49374</td>
</tr>
<tr>
<td>Assault</td>
<td>214.58750</td>
<td>100.64360</td>
</tr>
<tr>
<td>Burglary</td>
<td>1316.37917</td>
<td>423.31261</td>
</tr>
<tr>
<td>Larceny</td>
<td>2696.88542</td>
<td>714.75023</td>
</tr>
<tr>
<td>Auto_Theft</td>
<td>383.97917</td>
<td>194.37033</td>
</tr>
</tbody>
</table>

### Explained Variation of Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Prin1</th>
<th>Prin2</th>
<th>Prin3</th>
<th>Prin4</th>
<th>Prin5</th>
<th>Prin6</th>
<th>Prin7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Murder</td>
<td>0.37117</td>
<td>0.85539</td>
<td>0.87790</td>
<td>0.89562</td>
<td>0.97555</td>
<td>0.99143</td>
<td>1.00000</td>
</tr>
<tr>
<td>Rape</td>
<td>0.76242</td>
<td>0.79917</td>
<td>0.84059</td>
<td>0.84199</td>
<td>0.85065</td>
<td>0.99041</td>
<td>1.00000</td>
</tr>
<tr>
<td>Robbery</td>
<td>0.63783</td>
<td>0.64064</td>
<td>0.82164</td>
<td>0.92942</td>
<td>0.99788</td>
<td>0.99992</td>
<td>1.00000</td>
</tr>
<tr>
<td>Assault</td>
<td>0.63517</td>
<td>0.79127</td>
<td>0.79341</td>
<td>0.91781</td>
<td>0.98822</td>
<td>0.99513</td>
<td>1.00000</td>
</tr>
<tr>
<td>Burglary</td>
<td>0.78913</td>
<td>0.84414</td>
<td>0.88183</td>
<td>0.88207</td>
<td>0.88544</td>
<td>0.94800</td>
<td>1.00000</td>
</tr>
<tr>
<td>Larceny</td>
<td>0.51373</td>
<td>0.72178</td>
<td>0.93718</td>
<td>0.95479</td>
<td>0.95492</td>
<td>0.95530</td>
<td>1.00000</td>
</tr>
<tr>
<td>Auto_Theft</td>
<td>0.33638</td>
<td>0.65746</td>
<td>0.90481</td>
<td>0.96197</td>
<td>0.99623</td>
<td>0.99706</td>
<td>1.00000</td>
</tr>
</tbody>
</table>

### Eigenvalues

<table>
<thead>
<tr>
<th>Eigenvalue</th>
<th>Difference</th>
<th>Proportion</th>
<th>Cumulative</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4.045824</td>
<td>2.781795</td>
<td>0.5780</td>
</tr>
<tr>
<td>2</td>
<td>1.264030</td>
<td>0.516529</td>
<td>0.1806</td>
</tr>
<tr>
<td>3</td>
<td>0.747500</td>
<td>0.421175</td>
<td>0.1068</td>
</tr>
<tr>
<td>4</td>
<td>0.326325</td>
<td>0.061119</td>
<td>0.0466</td>
</tr>
<tr>
<td>5</td>
<td>0.265207</td>
<td>0.036843</td>
<td>0.0379</td>
</tr>
<tr>
<td>6</td>
<td>0.228364</td>
<td>0.105613</td>
<td>0.0326</td>
</tr>
<tr>
<td>7</td>
<td>0.122750</td>
<td>0.0175</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

### Loadings

<table>
<thead>
<tr>
<th>Variable</th>
<th>Prin1</th>
<th>Prin2</th>
<th>Prin3</th>
<th>Prin4</th>
<th>Prin5</th>
<th>Prin6</th>
<th>Prin7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Murder</td>
<td>0.30289</td>
<td>-0.61893</td>
<td>0.17353</td>
<td>-0.23308</td>
<td>0.54896</td>
<td>-0.26371</td>
<td>-0.26428</td>
</tr>
<tr>
<td>Rape</td>
<td>0.43410</td>
<td>-0.17053</td>
<td>-0.23539</td>
<td>0.06540</td>
<td>0.18075</td>
<td>0.78232</td>
<td>0.27946</td>
</tr>
<tr>
<td>Robbery</td>
<td>0.39705</td>
<td>0.04713</td>
<td>0.49208</td>
<td>-0.57470</td>
<td>-0.50808</td>
<td>0.09452</td>
<td>0.02497</td>
</tr>
<tr>
<td>Assault</td>
<td>0.39622</td>
<td>-0.35142</td>
<td>-0.05343</td>
<td>0.61744</td>
<td>-0.51525</td>
<td>-0.17395</td>
<td>-0.19921</td>
</tr>
<tr>
<td>Burglary</td>
<td>0.44164</td>
<td>0.20861</td>
<td>-0.22454</td>
<td>-0.02750</td>
<td>0.11273</td>
<td>-0.52340</td>
<td>0.65085</td>
</tr>
<tr>
<td>Larceny</td>
<td>0.35634</td>
<td>0.40570</td>
<td>-0.53681</td>
<td>-0.23231</td>
<td>0.02172</td>
<td>-0.04085</td>
<td>-0.60346</td>
</tr>
<tr>
<td>Auto_Theft</td>
<td>0.28834</td>
<td>0.50400</td>
<td>0.57524</td>
<td>0.41853</td>
<td>0.35939</td>
<td>0.06024</td>
<td>-0.15487</td>
</tr>
</tbody>
</table>
Example 15.2: Extracting Principal Components with NIPALS

PROC PCA produces the scree plot as shown in Output 15.2.2 by default when ODS Graphics is enabled and no plot request is specified in the PLOTS option. You can obtain more plots by specifying the plot requests in the PLOTS option in the PROC PCA statement.

The scree plot on the left shows that the eigenvalue of the first component is approximately 4 and the eigenvalue of the second component is largely decreased to under 1.5. The variance-explained plot on the right shows that the first two principal components account for nearly 80% of the total variance.

Output 15.2.2 Scree Plot from the PCA Procedure

In addition to the scree plot, PROC PCA also produces the component pattern plot and the component pattern profile plot. The following statements request these plots:

```plaintext
proc pca data=mycas.Crime method=nipals
   plots=(pattern(ncomp=3) patternprofile);
run;
```

The pairwise component pattern plots are shown in Output 15.2.3 through Output 15.2.5. The pattern plots show the following:

- All variables positively and evenly correlate with the first principal component (Output 15.2.3 and Output 15.2.4).
- The variable Auto_Theft correlates highly with the second component, and the variable Murder correlates highly but negatively with the second component (Output 15.2.3).
- All the variables have low correlations (either positively or negatively) with the third component (Output 15.2.4).
- The variable Auto_Theft correlates highly with the second component, but it has low correlation with the third component; the variable Murder correlates highly but negatively with the second component, but it has very low correlation with the third component (Output 15.2.5).
Chapter 15: The PCA Procedure

Output 15.2.3  Pattern Plot of Component 2 by Component 1

Output 15.2.4  Pattern Plot of Component 3 by Component 1
Output 15.2.5 Pattern Plot of Component 3 by Component 2

Output 15.2.6 shows a component pattern profile. As shown in the pattern plots, the nearly horizontal profile of the first component indicates that the first component is mostly correlated evenly across all variables.
Example 15.3: Approximating Principal Components with the RANDOM Method

This example illustrates that the iterative method based on random projection (RANDOM) of Halko, Martinsson, and Tropp (2011) provides a good approximation to the true principal components by using simulated data. The data for this example are the result of multiplying the low-rank matrices $A$ and $B$ and adding random noise. Matrix $A$ is 1,000,000 rows by 50 columns, and matrix $B$ is 50 rows by 5,000 columns, so the final data table consists of 1,000,000 observations and 5,000 variables.

The following DATA step generates the data:

```plaintext
data mycas.testdata / sessref=mysess single=no;
  title;
  keep x:;
  drop rank number_of_obs number_of_var sigma ii idum;
  drop rv1 rv2 rsq fac row col;
  drop nobs_per_thread nextras start_obs obs j k;
  array B[50,5000]; /* dimensions: rank, number_of_var */
  array A[50]; /* dimension: rank */
  array x[5000] x1-x5000; /* dimension: number_of_var */

  target_nthreads = min(_nthreads_,96);
  if (_threadid_=1) then
    put "Number of threads = " target_nthreads;
  rank=50;
  number_of_obs=1000000;
  number_of_var=5000;
```

Example 15.3: Approximating Principal Components with the RANDOM Method

sigma=0.1;

call streaminit(1);

if (_threadid_ <= target_nthreads) then do;
   ii = 0;
   idum = 0;
   do while (ii < rank * number_of_var);
      idum = mod(mod(1664525*idum,4294967296)+1013904223,4294967296);
      rv1 = 2.0*(idum/4294967296)-1.0;
      idum = mod(mod(1664525*idum,4294967296)+1013904223,4294967296);
      rv2 = 2.0*(idum/4294967296)-1.0;
      rsq = rv1*rv1+rv2*rv2;
      if ((rsq < 1.0) and (rsq ^= 0.0)) then do;
         fac = sqrt(-2.0*log(rsq)/rsq);
         row = int(ii/number_of_var)+1;
         col = mod(ii,number_of_var)+1;
         B[row,col] = rv1*fac;
         ii = ii + 1;
      end;
   end;
   nobs_per_thread = int(number_of_obs /target_nthreads);
   nextras = number_of_obs - nobs_per_thread*target_nthreads;
   if (_threadid_ <= nextras) then do;
      nobs_per_thread = nobs_per_thread+1;
      start_obs = (_threadid_-1)*nobs_per_thread+1;
   end;
   else
      start_obs = nextras+(_threadid_-1)*nobs_per_thread+1;
   end;
   do obs = start_obs to (start_obs+nobs_per_thread-1);
      do j = 1 to rank;
         A[j] = rand('Normal');
      end;
      do k = 1 to number_of_var;
         x[k] = sigma*rand('Normal');
         do j = 1 to rank;
            x[k] = x[k] + A[j]*B[j,k];
         end;
      end;
   end;
output;
end;
run;
The following statements use PROC PCA to extract principal components by using the RANDOM and EIG methods. The suboptions for the RANDOM method specify 1, 5, or 10 iterations. Both the EIG and RANDOM methods extract 25 principal components. The DISPLAY statement suppresses the default output except for eigenvalues, and the DISPLAYOUT statement saves the computed eigenvalues to specific CAS output tables.

```sas
proc pca data=mycas.testdata n=25 method=random(niter=1);
  var x:;
  display Eigenvalues;
  displayout Eigenvalues=oneiter;
run;

proc pca data=mycas.testdata n=25 method=random(niter=5);
  var x:;
  display Eigenvalues;
  displayout Eigenvalues=fiveiter;
run;

proc pca data=mycas.testdata n=25 method=random(niter=10);
  var x:;
  display Eigenvalues;
  displayout Eigenvalues=teniter;
run;

proc pca data=mycas.testdata n=25 method=eig;
  var x:;
  display Eigenvalues;
  displayout Eigenvalues=trueeig;
run;
```

To assess the accuracy of the RANDOM method, compare the estimated eigenvalues to those computed by the EIG method. Using the DATA step and PROC SGPLOT (code not shown), you can combine the estimated eigenvalue data sets with the true eigenvalue data set and plot the results. Output 15.3.1 shows this composite plot and demonstrates the following heuristic rules for the RANDOM method:

- One iteration usually provides a reasonable approximation of the true eigenvalues.
- Three to five iterations usually provide the best trade-off between approximation accuracy and computational cost.
- Beyond five iterations, the improvement in approximation accuracy is usually small compared to the computational cost.

For additional examples that show the impact of the number of iterations for both simulated and real data, see Halko, Martinsson, and Tropp (2011).
Output 15.3.1 RANDOM Method Performance with Varying Number of Iterations

**NOTE:** The heuristic rules provide guidance for selecting the number of iterations for data tables that are (approximately) low-rank. You might need to increase the number of iterations beyond what is recommended for data tables that are not low-rank.

References


Chapter 16
The PHSELECT Procedure

Contents

Overview: PHSELECT Procedure .................................................. 744
PROC PHSELECT Features ......................................................... 744
Using CAS Sessions and CAS Engine Librefs ............................... 745
Getting Started: PHSELECT Procedure ....................................... 746
  Cox Regression ................................................................. 746
Syntax: PHSELECT Procedure .................................................. 751
  PROC PHSELECT Statement .................................................. 751
  BY Statement ................................................................. 754
  CLASS Statement ............................................................ 754
  CODE Statement ............................................................ 754
  DISPLAY Statement ......................................................... 756
  DISPLAYOUT Statement ..................................................... 757
  EFFECT Statement ........................................................... 758
  FREQ Statement ............................................................ 760
  MODEL Statement ............................................................ 760
  OUTPUT Statement ........................................................... 761
  PARTITION Statement ....................................................... 764
  SELECTION Statement ....................................................... 765
  STRATA Statement .......................................................... 767
  WEIGHT Statement .......................................................... 767
Details: PHSELECT Procedure .................................................. 767
  Missing Values ............................................................... 767
  Cox Proportional Hazards Regression Model ............................. 767
  Partial Likelihood for the Cox Model ..................................... 768
  The LASSO Method of Model Selection ................................... 768
  Model Fit and Assessment Statistics ...................................... 769
  Predicted Values and Regression Diagnostics ............................ 770
  Joint Tests and Type 3 Tests ............................................... 772
  Multithreading ............................................................... 773
  Optimization Algorithms .................................................... 773
  Displayed Output ............................................................ 773
  ODS Table Names ........................................................... 777
  ODS Graphics ................................................................... 778
Examples: PHSELECT Procedure ................................................. 779
  Example 16.1: Model Selection ............................................. 779
  Example 16.2: Stratified Analysis .......................................... 783
References ............................................................................. 785
Overview: PHSELECT Procedure

The PHSELECT procedure fits the Cox proportional hazards regression models for survival data and performs variable selection in SAS Viya.

The models that PROC PHSELECT supports can contain main effects that consist of both continuous and classification variables and interaction effects of these variables. The models can also include constructed effects such as splines. The procedure offers a number of effect-selection methods, including stepwise methods and modern LASSO methods. It also offers extensive capabilities for customizing the model selection by using a wide variety of selection and stopping criteria, from computationally efficient significance-level-based criteria to modern, computationally intensive validation-based criteria. PROC PHSELECT also provides a variety of Cox regression diagnostics that are conditional on the selected model.

PROC PHSELECT Features

The PHSELECT procedure estimates the parameters of a Cox regression model by maximizing the partial likelihood. It also does the following:

- provides model-building syntax through the CLASS, EFFECT, and effect-based MODEL statements, which are familiar from SAS/STAT analytic procedures (in particular, the GLM, LOGISTIC, GLIMMIX, and PHREG procedures)
- enables model building (variable selection) through the SELECTION statement
- enables you to split classification effects into individual components by using the SPLIT option in the CLASS statement
- permits any degree of interaction effects that involve classification and continuous variables
- provides a WEIGHT statement for weighted analysis
- provides a FREQ statement for grouped analysis
- provides a STRATA statement for stratified analysis
- provides a CODE statement to produce SAS code that can score a new data set
- provides an OUTPUT statement to produce a data table that contains predicted values and other observationwise statistics
- uses ODS Graphics to create model selection plots as part of its output. For more information about ODS Graphics, see the section “ODS Graphics” on page 778.

Because the PHSELECT procedure runs on CAS, it also does the following:

- enables you to run on a cluster of machines that distribute the data and the computations
- enables you to run in single-machine mode
- exploits all the available cores and concurrent threads. For information about how PROC PHSELECT uses threads, see the section “Multithreading” on page 81 in Chapter 3, “Shared Concepts.”
**PROC PHSELECT Compared with the PHREG Procedure**

The PHSELECT procedure provides the full set of parameterizations that are available in the PHREG procedure. The PHSELECT procedure uses the GLM parameterization for the CLASS variables by default. The PHREG procedure uses the REF parameterization for the CLASS variables by default. In either procedure, you can use the PARAM= option in the CLASS statement to change the parameterization.

The PHSELECT procedure uses a quasi-Newton algorithm by default. You can choose different optimization techniques, including second-order methods that require a crossproducts matrix, by using the TECHNIQUE= option in the PROC PHSELECT statement. The PHREG procedure uses the Newton-Raphson optimization algorithm exclusively.

The PHREG procedure can fit a variety of models, but it has very limited model selection capability. The PHSELECT procedure provides a richer set of model selection options, including LASSO selection, information-criterion-based selection and stopping criteria, and validation-based criteria.

The PHSELECT procedure uses ODS Graphics to display plots that help interpret the selection process; the PHREG procedure use ODS graphics to display predicted survival curves.

The PHSELECT procedure is specifically designed to operate in SAS Viya and performs computations in multiple threads. The PHREG procedure executes in a single thread on a single machine.

---

**Using CAS Sessions and CAS Engine Librefs**

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:

```plaintext
proc options option=(CASHOST CASPORT);
run;
```

In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:

```plaintext
cas mysess;
  libname mycas cas sessref=mysess;
```

The CAS statement creates the CAS session named mysess, and the LIBNAME statement creates the mycas CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the CASHOST and CASPORT of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the mysess session, you can terminate it by using the TERMINATE option in the CAS statement as follows:
Chapter 16: The PHSELECT Procedure

CAS MYSESS TERMINATE;

For more information about the CAS and LIBNAME statements, see the section “Introduction to Shared Concepts” on page 10 in Chapter 3, “Shared Concepts.”

Getting Started: PHSELECT Procedure

Cox Regression

NOTE: Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

The following DATA step creates the data table MYCAS.GETSTARTED in your CAS session. This data table consists of 100 observations on a failure time variable (Time), an indicator variable (Status) that has two values (0 for censored observations and 1 for event observations), three classification variables (C1–C3), and four continuous variables (X1–X4). This DATA step assumes that your CAS engine libref is named MYCAS, but you can substitute any appropriately defined CAS engine libref.

data mycas.getstarted;
  input Time Status C1$ C2 C3$ X1-X4;
datalines;
  53 0 Low 1 M 1.11 2.000 3.6128 12.0
  12 0 High 1 M 1.40 1.362 3.8388 8.8
  11 1 Low 1 F 1.57 1.672 3.8865 7.5
  7 1 Medium 0 M 1.04 2.000 3.7324 5.1
  2 1 Low 1 M 1.52 2.000 3.8751 9.8
  41 0 High 1 M 1.76 1.447 3.7243 12.8
  6 1 Critical 1 F 1.36 1.462 3.5441 9.0
  6 1 Critical 1 M 1.42 1.690 3.9294 10.4
  16 1 Medium 1 M 1.32 0.699 3.6990 8.8
  41 1 Medium 1 M 1.00 1.477 3.4771 10.2
  2 1 High 0 M 1.30 2.000 3.7243 5.1
  58 1 Medium 1 M 1.20 1.580 3.6990 12.1
  11 1 High 1 M 1.08 1.903 3.5051 9.6
  12 0 Critical 1 F 1.15 1.146 3.6435 11.6
  16 0 High 1 F 1.15 0.903 3.8573 13.0
  54 1 Medium 1 M 1.26 1.699 3.7243 9.0
  51 1 Low 0 M 1.57 1.041 3.4150 7.7
  67 1 Medium 1 M 1.32 1.041 3.6435 12.8
  1 1 Low 1 M 1.94 1.954 3.9868 12.0
  19 0 Medium 1 M 1.32 2.000 3.7709 13.0
  1 1 Medium 1 M 2.22 1.954 3.6628 9.4
  35 1 Medium 0 M 1.11 1.176 3.6532 7.0
  41 1 High 1 M 1.15 1.342 3.5185 5.0
  58 1 Medium 1 M 1.20 1.580 3.6990 12.1
  11 1 Medium 1 M 1.11 1.279 3.8808 14.0
<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>Medium</th>
<th>1</th>
<th>M</th>
<th>1.00</th>
<th>1.477</th>
<th>3.4771</th>
<th>10.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>19</td>
<td>1</td>
<td>High</td>
<td>0</td>
<td>M</td>
<td>1.26</td>
<td>1.929</td>
<td>3.7924</td>
<td>7.5</td>
</tr>
<tr>
<td>89</td>
<td>1</td>
<td>High</td>
<td>1</td>
<td>M</td>
<td>1.32</td>
<td>1.623</td>
<td>3.6532</td>
<td>14.0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>Medium</td>
<td>1</td>
<td>F</td>
<td>1.95</td>
<td>0.778</td>
<td>4.0453</td>
<td>10.2</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>Critical</td>
<td>1</td>
<td>F</td>
<td>1.36</td>
<td>1.462</td>
<td>3.5441</td>
<td>9.0</td>
</tr>
<tr>
<td>57</td>
<td>0</td>
<td>Low</td>
<td>1</td>
<td>F</td>
<td>1.26</td>
<td>1.954</td>
<td>3.9685</td>
<td>12.5</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>High</td>
<td>1</td>
<td>M</td>
<td>2.24</td>
<td>1.663</td>
<td>4.9542</td>
<td>10.1</td>
</tr>
<tr>
<td>17</td>
<td>1</td>
<td>Medium</td>
<td>1</td>
<td>F</td>
<td>1.59</td>
<td>1.613</td>
<td>3.4314</td>
<td>11.2</td>
</tr>
<tr>
<td>77</td>
<td>0</td>
<td>Low</td>
<td>1</td>
<td>F</td>
<td>1.08</td>
<td>0.954</td>
<td>3.6812</td>
<td>14.0</td>
</tr>
<tr>
<td>66</td>
<td>1</td>
<td>High</td>
<td>1</td>
<td>M</td>
<td>1.45</td>
<td>1.820</td>
<td>3.7853</td>
<td>6.6</td>
</tr>
<tr>
<td>16</td>
<td>1</td>
<td>Medium</td>
<td>1</td>
<td>M</td>
<td>1.32</td>
<td>0.699</td>
<td>3.6990</td>
<td>8.8</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>Critical</td>
<td>1</td>
<td>M</td>
<td>1.08</td>
<td>1.653</td>
<td>3.8325</td>
<td>9.9</td>
</tr>
<tr>
<td>19</td>
<td>1</td>
<td>High</td>
<td>0</td>
<td>M</td>
<td>1.26</td>
<td>1.929</td>
<td>3.7924</td>
<td>7.5</td>
</tr>
<tr>
<td>37</td>
<td>1</td>
<td>High</td>
<td>1</td>
<td>F</td>
<td>1.60</td>
<td>1.204</td>
<td>3.9542</td>
<td>11.0</td>
</tr>
<tr>
<td>52</td>
<td>1</td>
<td>Medium</td>
<td>1</td>
<td>M</td>
<td>1.00</td>
<td>1.653</td>
<td>3.8573</td>
<td>10.1</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>High</td>
<td>0</td>
<td>F</td>
<td>1.66</td>
<td>1.792</td>
<td>3.6435</td>
<td>4.9</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>Medium</td>
<td>1</td>
<td>F</td>
<td>1.54</td>
<td>1.935</td>
<td>4.4757</td>
<td>6.7</td>
</tr>
<tr>
<td>51</td>
<td>1</td>
<td>Low</td>
<td>0</td>
<td>M</td>
<td>1.57</td>
<td>1.041</td>
<td>3.4150</td>
<td>7.7</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>High</td>
<td>0</td>
<td>M</td>
<td>1.30</td>
<td>2.000</td>
<td>3.7243</td>
<td>5.1</td>
</tr>
<tr>
<td>25</td>
<td>1</td>
<td>Medium</td>
<td>1</td>
<td>M</td>
<td>1.00</td>
<td>1.644</td>
<td>3.8195</td>
<td>12.4</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>Low</td>
<td>1</td>
<td>F</td>
<td>1.57</td>
<td>1.672</td>
<td>3.8865</td>
<td>7.5</td>
</tr>
<tr>
<td>19</td>
<td>1</td>
<td>Low</td>
<td>1</td>
<td>M</td>
<td>1.08</td>
<td>2.000</td>
<td>3.9191</td>
<td>14.4</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>Low</td>
<td>1</td>
<td>M</td>
<td>1.72</td>
<td>1.740</td>
<td>3.7993</td>
<td>8.2</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>Medium</td>
<td>1</td>
<td>M</td>
<td>1.11</td>
<td>1.398</td>
<td>3.5185</td>
<td>9.7</td>
</tr>
<tr>
<td>16</td>
<td>1</td>
<td>Medium</td>
<td>1</td>
<td>M</td>
<td>1.34</td>
<td>2.000</td>
<td>3.9345</td>
<td>9.0</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>High</td>
<td>1</td>
<td>M</td>
<td>1.40</td>
<td>1.362</td>
<td>3.8388</td>
<td>8.8</td>
</tr>
<tr>
<td>17</td>
<td>1</td>
<td>High</td>
<td>1</td>
<td>M</td>
<td>1.23</td>
<td>1.447</td>
<td>3.8808</td>
<td>10.0</td>
</tr>
<tr>
<td>17</td>
<td>1</td>
<td>High</td>
<td>1</td>
<td>M</td>
<td>1.23</td>
<td>1.447</td>
<td>3.8808</td>
<td>10.0</td>
</tr>
<tr>
<td>41</td>
<td>0</td>
<td>High</td>
<td>1</td>
<td>M</td>
<td>1.76</td>
<td>1.447</td>
<td>3.7243</td>
<td>12.8</td>
</tr>
<tr>
<td>88</td>
<td>1</td>
<td>High</td>
<td>1</td>
<td>F</td>
<td>1.18</td>
<td>1.756</td>
<td>3.5563</td>
<td>10.6</td>
</tr>
<tr>
<td>16</td>
<td>0</td>
<td>High</td>
<td>1</td>
<td>F</td>
<td>1.15</td>
<td>0.903</td>
<td>3.8573</td>
<td>13.0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>Low</td>
<td>1</td>
<td>F</td>
<td>1.92</td>
<td>1.623</td>
<td>3.9590</td>
<td>10.0</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>Low</td>
<td>1</td>
<td>M</td>
<td>1.18</td>
<td>1.519</td>
<td>3.7243</td>
<td>11.4</td>
</tr>
<tr>
<td>19</td>
<td>0</td>
<td>Medium</td>
<td>1</td>
<td>M</td>
<td>1.32</td>
<td>1.519</td>
<td>3.8808</td>
<td>10.8</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>Critical</td>
<td>1</td>
<td>M</td>
<td>1.08</td>
<td>1.653</td>
<td>3.8325</td>
<td>9.9</td>
</tr>
<tr>
<td>57</td>
<td>0</td>
<td>Low</td>
<td>1</td>
<td>F</td>
<td>1.26</td>
<td>1.954</td>
<td>3.9685</td>
<td>12.5</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>Medium</td>
<td>1</td>
<td>M</td>
<td>1.98</td>
<td>1.568</td>
<td>3.3617</td>
<td>9.5</td>
</tr>
<tr>
<td>19</td>
<td>1</td>
<td>Low</td>
<td>1</td>
<td>M</td>
<td>1.08</td>
<td>2.000</td>
<td>3.9191</td>
<td>14.4</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>Low</td>
<td>1</td>
<td>F</td>
<td>1.53</td>
<td>1.881</td>
<td>3.5911</td>
<td>10.2</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>High</td>
<td>1</td>
<td>F</td>
<td>1.68</td>
<td>1.732</td>
<td>3.7324</td>
<td>6.5</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>Low</td>
<td>0</td>
<td>M</td>
<td>1.75</td>
<td>1.255</td>
<td>3.8062</td>
<td>11.3</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Low</td>
<td>1</td>
<td>M</td>
<td>1.94</td>
<td>1.954</td>
<td>3.9868</td>
<td>12.0</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>Low</td>
<td>1</td>
<td>M</td>
<td>1.60</td>
<td>1.431</td>
<td>3.6902</td>
<td>10.6</td>
</tr>
<tr>
<td>26</td>
<td>1</td>
<td>Low</td>
<td>1</td>
<td>M</td>
<td>1.23</td>
<td>2.000</td>
<td>3.6021</td>
<td>11.2</td>
</tr>
<tr>
<td>92</td>
<td>1</td>
<td>Low</td>
<td>1</td>
<td>M</td>
<td>1.43</td>
<td>1.415</td>
<td>4.0755</td>
<td>11.0</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>Medium</td>
<td>1</td>
<td>M</td>
<td>1.30</td>
<td>1.820</td>
<td>3.7993</td>
<td>13.2</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>Medium</td>
<td>1</td>
<td>F</td>
<td>1.54</td>
<td>1.935</td>
<td>4.4757</td>
<td>6.7</td>
</tr>
<tr>
<td>66</td>
<td>1</td>
<td>High</td>
<td>1</td>
<td>M</td>
<td>1.45</td>
<td>1.820</td>
<td>3.7853</td>
<td>6.6</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Medium</td>
<td>1</td>
<td>M</td>
<td>2.22</td>
<td>1.954</td>
<td>3.6628</td>
<td>9.4</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>Medium</td>
<td>1</td>
<td>M</td>
<td>1.30</td>
<td>1.820</td>
<td>3.7993</td>
<td>13.2</td>
</tr>
<tr>
<td>14</td>
<td>1</td>
<td>Medium</td>
<td>1</td>
<td>M</td>
<td>1.40</td>
<td>1.255</td>
<td>3.7243</td>
<td>14.6</td>
</tr>
<tr>
<td>32</td>
<td>1</td>
<td>High</td>
<td>1</td>
<td>M</td>
<td>1.32</td>
<td>1.634</td>
<td>3.6990</td>
<td>10.6</td>
</tr>
<tr>
<td>24</td>
<td>1</td>
<td>High</td>
<td>1</td>
<td>M</td>
<td>1.30</td>
<td>0.477</td>
<td>4.0899</td>
<td>14.6</td>
</tr>
<tr>
<td>18</td>
<td>1</td>
<td>Critical</td>
<td>1</td>
<td>F</td>
<td>1.45</td>
<td>0.903</td>
<td>3.5682</td>
<td>7.5</td>
</tr>
</tbody>
</table>
The following statements fit a Cox proportional hazards model to these data by using three classification effects for the variables C1–C3 and four regressor effects for the variables X1–X4. The ITHIST option displays a table that summarizes the steps of the optimization.

```plaintext
proc phselect data=mycas.getStarted ithist;
  class C1-C3;
  model Time*Status(0) = C1-C3 X1-X4;
run;
```

The output from this analysis is presented in Figure 16.1 through Figure 16.8.

**Figure 16.1** Model Information

The PHSELECT Procedure

<table>
<thead>
<tr>
<th>Model Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source</td>
</tr>
<tr>
<td>Response Variable</td>
</tr>
<tr>
<td>Censoring Variable</td>
</tr>
<tr>
<td>Censoring Values</td>
</tr>
<tr>
<td>Optimization technique</td>
</tr>
</tbody>
</table>

**Figure 16.2** displays the “Number of Observations” table. All 100 observations in the data table are used in the analysis; of them, 26 are censored and 74 are uncensored.
The classification variables C1–C3 are parameterized using the GLM parameterization, which is the default. The variable C1 has four unique formatted levels; each of the two variables C2 and C3 has two levels. The classification levels are displayed in the “Class Level Information” table in Figure 16.3.

The “Iteration History” table is shown in Figure 16.4. The quasi-Newton algorithm converged after 12 iterations, not counting the initial setup iteration.

Figure 16.5 displays the final convergence status of the quasi-Newton algorithm. The GCONV=1E-8 convergence criterion is satisfied.
Figure 16.6 displays the “Dimensions” table for this model. This table summarizes some important sizes of various model components. For example, it shows that the design matrix $X$ has 12 columns: 4 columns for the effects that are associated with the classification variable $C_1$, 2 columns for each of the classification variables $C_2$ and $C_3$, and 1 column for each of the continuous variables $X_1$–$X_4$. However, the rank of the crossproducts matrix is only 9. Because the classification variables $C_1$–$C_3$ use GLM parameterization, there is one singularity in the crossproducts matrix of the model for each classification variable. Consequently, only nine parameters enter the optimization.

**Figure 16.6 Dimensions in Cox Regression**

<table>
<thead>
<tr>
<th>Dimensions</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Effects</td>
<td>7</td>
</tr>
<tr>
<td>Max Effect Columns</td>
<td>4</td>
</tr>
<tr>
<td>Columns in Design</td>
<td>12</td>
</tr>
<tr>
<td>Rank of Design</td>
<td>9</td>
</tr>
</tbody>
</table>

The “Fit Statistics” table is shown in Figure 16.7. The $-2 \log$ likelihood at the converged estimates is 510.78672. You can use this value to compare the model to nested model alternatives by means of a likelihood ratio test. To compare models that are not nested, you can use information criteria such as Akaike’s information criterion (AIC), Akaike’s bias-corrected information criterion (AICC), and the Schwarz Bayesian information criterion (BIC). These criteria penalize the $-2 \log$ partial likelihood for the number of parameters.

**Figure 16.7 Fit Statistics**

<table>
<thead>
<tr>
<th>Fit Statistics</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$-2 \log$ Likelihood</td>
<td>510.78672</td>
</tr>
<tr>
<td>AIC (smaller is better)</td>
<td>528.78672</td>
</tr>
<tr>
<td>AICC (smaller is better)</td>
<td>531.59922</td>
</tr>
<tr>
<td>SBC (smaller is better)</td>
<td>549.52331</td>
</tr>
</tbody>
</table>

The “Parameter Estimates” table in Figure 16.8 shows that many parameters have fairly large $p$-values, indicating that one or more of the model effects might not be necessary.

**Figure 16.8 Parameter Estimates**

<table>
<thead>
<tr>
<th>Parameter Estimates</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
<td>DF</td>
</tr>
<tr>
<td>C1 Critical</td>
<td>1</td>
</tr>
<tr>
<td>C1 High</td>
<td>1</td>
</tr>
<tr>
<td>C1 Low</td>
<td>1</td>
</tr>
<tr>
<td>C1 Medium</td>
<td>0</td>
</tr>
<tr>
<td>C2 0</td>
<td>1</td>
</tr>
<tr>
<td>C2 1</td>
<td>0</td>
</tr>
<tr>
<td>C3 F</td>
<td>1</td>
</tr>
<tr>
<td>C3 M</td>
<td>0</td>
</tr>
<tr>
<td>X1</td>
<td>1</td>
</tr>
<tr>
<td>X2</td>
<td>1</td>
</tr>
<tr>
<td>X3</td>
<td>1</td>
</tr>
<tr>
<td>X4</td>
<td>1</td>
</tr>
</tbody>
</table>
Finally, the procedure displays the table in Figure 16.9, which shows the amount of time (in seconds) that PROC PHSELECT required to perform different tasks in the analysis.

**Figure 16.9** Procedure Timing

<table>
<thead>
<tr>
<th>Task</th>
<th>Seconds</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setup and Parsing</td>
<td>0.03</td>
<td>10.77%</td>
</tr>
<tr>
<td>Levelization</td>
<td>0.01</td>
<td>5.88%</td>
</tr>
<tr>
<td>Model Initialization</td>
<td>0.01</td>
<td>4.31%</td>
</tr>
<tr>
<td>SSCP Computation</td>
<td>0.00</td>
<td>0.67%</td>
</tr>
<tr>
<td>Model Fitting</td>
<td>0.18</td>
<td>73.60%</td>
</tr>
<tr>
<td>Cleanup</td>
<td>0.01</td>
<td>2.21%</td>
</tr>
<tr>
<td>Total</td>
<td>0.24</td>
<td>100.00%</td>
</tr>
</tbody>
</table>

**Syntax: PHSELECT Procedure**

The following statements are available in the PHSELECT procedure:

```
PROC PHSELECT <options> ;
   BY variables ;
   CLASS variable <(options)> . . . <variable <(options)> > <global-options> ;
   CODE <options> ;
   DISPLAY <table-list> <options> ;
   DISPLAYOUT table-spec-list <options> ;
   EFFECT name=effect-type(variables <options>) ;
   FREQ variable ;
   MODEL response < * censor(list) > = <effects> <model-options> ;
   OUTPUT OUT=CAS-libref.data-table <options> <keyword <name>> . . . <keyword <name>> ;
   PARTITION partition-options ;
   SELECTION <METHOD=method< (method-options) >> <options> ;
   STRATA variable <MISSING> ;
   WEIGHT variable ;
```

The PROC PHSELECT statement and the MODEL statement are required. The CLASS statement can appear multiple times. If you use a CLASS statement, it must precede the MODEL statements.

**PROC PHSELECT Statement**

```
PROC PHSELECT <options> ;
```

The PROC PHSELECT statement invokes the procedure. Table 16.1 summarizes the available options in the PROC PHSELECT statement by function. They are then described fully in alphabetical order.
## Table 16.1 PROC PHSELECT Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALPHA=</td>
<td>Specifies a global significance level</td>
</tr>
<tr>
<td>DATA=</td>
<td>Specifies the CAS input data table</td>
</tr>
<tr>
<td><strong>Output Options</strong></td>
<td></td>
</tr>
<tr>
<td>CORRB</td>
<td>Displays the “Parameter Estimates Correlation Matrix” table</td>
</tr>
<tr>
<td>COVB</td>
<td>Displays the “Parameter Estimates Covariance Matrix” table</td>
</tr>
<tr>
<td>ITHIST</td>
<td>Displays the “Iteration History” table</td>
</tr>
<tr>
<td>LOGLIKNULL</td>
<td>Displays the “–2 Log Likelihood for Null Model” table</td>
</tr>
<tr>
<td>NOCLPRINT</td>
<td>Limits or suppresses the display of class levels</td>
</tr>
<tr>
<td>NOSTDERR</td>
<td>Suppresses computation of the covariance matrix and standard errors</td>
</tr>
<tr>
<td><strong>Optimization Options</strong></td>
<td></td>
</tr>
<tr>
<td>ABSCONV=</td>
<td>Tunes the absolute function convergence criterion</td>
</tr>
<tr>
<td>ABSFCONV=</td>
<td>Tunes the absolute function difference convergence criterion</td>
</tr>
<tr>
<td>ABSGCONV=</td>
<td>Tunes the absolute gradient convergence criterion</td>
</tr>
<tr>
<td>ABSXCONV=</td>
<td>Tunes the absolute parameter convergence criterion</td>
</tr>
<tr>
<td>FCONV=</td>
<td>Tunes the relative function difference convergence criterion</td>
</tr>
<tr>
<td>FCONV2=</td>
<td>Tunes the second relative function difference convergence criterion</td>
</tr>
<tr>
<td>GCONV=</td>
<td>Tunes the relative gradient convergence criterion</td>
</tr>
<tr>
<td>GCONV2=</td>
<td>Tunes the second relative gradient convergence criterion</td>
</tr>
<tr>
<td>XCONV=</td>
<td>Tunes the relative gradient convergence criterion</td>
</tr>
<tr>
<td>HESSIAN</td>
<td>Uses analytic Hessian instead of finite-difference Hessian</td>
</tr>
<tr>
<td>MAXFUNC=</td>
<td>Specifies the maximum number of function evaluations in any optimization</td>
</tr>
<tr>
<td>MAXITER=</td>
<td>Specifies the maximum number of iterations in any optimization</td>
</tr>
<tr>
<td>MAXTIME=</td>
<td>Specifies the upper limit of CPU time (in seconds) for any optimization</td>
</tr>
<tr>
<td>MINITER=</td>
<td>Specifies the minimum number of iterations in any optimization</td>
</tr>
<tr>
<td>TECHNIQUE=</td>
<td>Selects the optimization technique</td>
</tr>
<tr>
<td><strong>LASSO Options</strong></td>
<td></td>
</tr>
<tr>
<td>LASSORHO=</td>
<td>Specifies the base regularization parameter for the LASSO method</td>
</tr>
<tr>
<td>LASSOSTEPS=</td>
<td>Specifies the maximum number of steps for the LASSO method</td>
</tr>
<tr>
<td>LASSOTOL=</td>
<td>Specifies the convergence criterion for the LASSO method</td>
</tr>
</tbody>
</table>

The optimization options, with the exception of the HESSIAN option, are fully described in the section “Optimization Options” on page 44 in Chapter 3, “Shared Concepts.” The following list describes the other options available in the PROC PHSELECT statement:

**ALPHA=number**

specifies a global significance level for the construction of confidence intervals. The confidence level is 1–number. The value of number must be between 0 and 1. By default, ALPHA=0.05.

**CORRB**

creates the “Parameter Estimates Correlation Matrix” table. The correlation matrix is computed by normalizing the covariance matrix $\Sigma$. That is, if $\sigma_{ij}$ is an element of $\Sigma$, then the corresponding element of the correlation matrix is $\frac{\sigma_{ij}}{\sqrt{\sigma_{ii} \sigma_{jj}}}$, where $\sigma_i = \sqrt{\sigma_{ii}}$. 
**COVB**

creates the “Parameter Estimates Covariance Matrix” table. The covariance matrix is computed as the inverse of the negative Hessian matrix, which is the matrix of second derivatives of the log-likelihood function with respect to the model parameters.

**DATA=** *CAS-libref.data-table*

names the input data table for PROC PHSELECT to use. The default is the most recently created data table. *CAS-libref.data-table* is a two-level name, where

- **CAS-libref** refers to a collection of information that is defined in the LIBNAME statement and includes the *caslib*, which includes a path to the data, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about *CAS-libref*, see the section “Using CAS Sessions and CAS Engine Librefs” on page 745.

- **data-table** specifies the name of the input data table.

**HESSIAN**

computes the Hessian matrix by using the analytic expression for the second-order derivatives of the log partial likelihood function instead of using the finite-difference method. When you specify this option, the optimization technique defaults to the Newton-Raphson method with ridging (TECH=NRRIDG), but you can use the TECH= option to specify the technique of your choice. The HESSIAN option requires a large amount of memory for each machine in the cluster.

**ITHIST**

generates the “Iteration History” table.

**LASSORHO=** *r*

specifies the base regularization parameter for the LASSO model selection method. The regularization parameter for step *i* is *r^i*. By default, LASSORHO=0.8.

**LASSOSTEPS=** *n*

specifies the maximum number of steps for LASSO model selection. By default, LASSOSTEPS=20.

**LASSOTOL=** *r*

specifies the convergence tolerance for the optimization algorithm that solves for the LASSO parameter estimates at each step of LASSO model selection. By default, LASSOTOL=1E–6.

**LOGLIKENULL**

creates the “–2 Log Likelihood for the Null Model” table. If you also specify the PARTITION statement, the table displays the –2 log likelihood for the null model for each data partition.

**NOCLPRINT<=** *number>*

suppresses the display of the “Class Level Information” table if you do not specify *number*. If you specify *number*, the values of the classification variables are displayed for only those variables whose number of levels is less than *number*. Specifying *number* helps reduce the size of the “Class Level Information” table if some classification variables have a large number of levels.
NOSTDERR suppresses computation of the covariance matrix and the standard errors of the regression coefficients. When the model contains many variables (such as thousands), inverting the Hessian matrix to derive the covariance matrix and the standard errors of the regression coefficients can be time-consuming. The CORRB, COVB, and TYPE3 options are not available when you specify this option.

### BY Statement

**BY** variables;

You can specify a BY statement in PROC PHSELECT to obtain separate analyses of observations in groups that are defined by the values of the BY variables. If you specify more than one BY statement, only the last one specified is used. For more information, see the discussion of BY-group processing in *SAS Language Reference: Concepts*.

### CLASS Statement

**CLASS** variable <(options)> ... variable <(options)> < / global-options>;

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. You can list the response variable for binary models in the CLASS statement, but this is not required. Table 16.2 summarizes the values that you can use for either an option or a global-option. The options are fully documented in the section “CLASS Statement” on page 12 in Chapter 3, “Shared Concepts.”

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESCENDING</td>
<td>Reverses the sort order</td>
</tr>
<tr>
<td>MISSING</td>
<td>Treats missing values as valid levels</td>
</tr>
<tr>
<td>ORDER=</td>
<td>Specifies the sort order for the levels</td>
</tr>
<tr>
<td>PARAM=</td>
<td>Specifies the parameterization of the variable</td>
</tr>
<tr>
<td>REF=</td>
<td>Specifies the reference level of the variable</td>
</tr>
<tr>
<td>SPLIT</td>
<td>Allows design columns for a variable to enter or leave the model independently</td>
</tr>
</tbody>
</table>

### CODE Statement

**CODE** <options>;

The CODE statement writes SAS DATA step code for computing predicted values of the fitted model to a file, to a catalog entry, or to a CAS table. The predicted values include survival probabilities and cumulative hazards at specific time points. To score new data, you can then include the file or the catalog entry in a DATA step, or you can specify the CAS table in the runCodeTable action in the dataStep action set (for more information, see *SAS Viya: System Programming Guide*).
Table 16.3 summarizes the options available in the CODE statement.

### Table 16.3  CODE Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMMENT</td>
<td>Adds comments to the generated code</td>
</tr>
<tr>
<td>CUMHAZ</td>
<td>Generates cumulative hazard values at specified time points</td>
</tr>
<tr>
<td>FILE=</td>
<td>Names the file in which to save the generated code</td>
</tr>
<tr>
<td>FORMATWIDTH=</td>
<td>Specifies the numeric format width for the regression coefficients</td>
</tr>
<tr>
<td>INDENTSIZE=</td>
<td>Specifies the number of spaces to indent the generated code</td>
</tr>
<tr>
<td>LABELID=</td>
<td>Specifies a number used to construct names and labels</td>
</tr>
<tr>
<td>LINESIZE=</td>
<td>Specifies the line size for the generated code</td>
</tr>
<tr>
<td>NOSURVIVAL</td>
<td>Suppresses the generation of survival probabilities</td>
</tr>
<tr>
<td>NOTRIM</td>
<td>Compares formatted values, including blank padding</td>
</tr>
<tr>
<td>OUT=</td>
<td>Names an output CAS table in which to save the generated code</td>
</tr>
<tr>
<td>SHOWTIME</td>
<td>Creates variables that contain the time points at which predictions are made</td>
</tr>
<tr>
<td>TIMEPOINT=</td>
<td>Specifies the time points at which survival probabilities or cumulative hazards are predicted</td>
</tr>
</tbody>
</table>

The following CODE statement options are particularly important:

**CUMHAZ**

generates SAS code to predict the cumulative hazard function at the time points that you specify in the **TIMEPOINT=** option.

**FILE=filename**

names the external file that saves the generated code. When enclosed in a quoted string (for example, **FILE="c:\mydir\scorecode.sas"**), this option specifies the path and filename for writing the code to an external file. If you do not specify a path but your SAS client has a default path, then the code is written to an external file named **filename** at that location. You can also specify an unquoted **filename** of no more than eight characters. If the **filename** is assigned as a fileref in a Base SAS FILENAME statement, the file specified in the FILENAME statement is opened; otherwise, if your SAS client has a default path, an external file named **filename** is created. This option cannot be specified with the **OUT=** option.

**NOSURVIVAL**

suppresses SAS code for survival probabilities prediction. If you do not specify this option, PROC PHSELECT generates SAS code to predict survival probabilities at the time points that you specify in the **TIMEPOINT=** option.

**SHOWTIME**

creates variables that contain the time points at which predictions are made. If T is the name of the failure time variable, these time point variables are named T_1, T_2, and so on.

**TIMEPOINT=list | QUANTILE(probability-list)**

specifies the time points at which survival probabilities or cumulative hazards are predicted. You can specify a list of numbers that represent exact time points. For example:
code cumhaz timepoint = 40 to 60 by 10;

If T is the name of the failure time variable, the preceding specification requests the predicted cumulative hazard and survival probability at the time points T=40, T=50, and T=60. If you also specify the SHOWTIME option, these time points are saved in the variables T_1, T_2, and T_3, respectively. The predicted variables are named and labeled as shown in Table 16.4.

<table>
<thead>
<tr>
<th>T</th>
<th>Name</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>C_T_1</td>
<td>Cumulative Hazard at T_1</td>
</tr>
<tr>
<td>50</td>
<td>C_T_2</td>
<td>Cumulative Hazard at T_2</td>
</tr>
<tr>
<td>60</td>
<td>C_T_3</td>
<td>Cumulative Hazard at T_3</td>
</tr>
<tr>
<td>40</td>
<td>S_T_1</td>
<td>Survival Probability at T_1</td>
</tr>
<tr>
<td>50</td>
<td>S_T_2</td>
<td>Survival Probability at T_2</td>
</tr>
<tr>
<td>60</td>
<td>S_T_3</td>
<td>Survival Probability at T_3</td>
</tr>
</tbody>
</table>

Alternatively, you can use the keyword QUANTILE to specify a list of quantile probabilities. For example:

code timepoint = quantile(.2, .5, .8);

The PHSELECT procedure computes the 20th, 50th, and 80th percentiles from the Kaplan-Meier curve (or the Breslow curve if you also specify the ENTRY= option in the MODEL statement) and use them as the time points for the prediction.

By default, TIMEPOINT=QUANTILE(0.25, 0.50, 0.75).

For more information about the syntax of the CODE statement, see the section “CODE Statement” on page 16 in Chapter 3, “Shared Concepts.”

**DISPLAY Statement**

**DISPLAY <table-list> </options> ;**

The DISPLAY statement enables you to specify a list of display tables to display or exclude. This statement is similar to the ODS SELECT, ODS EXCLUDE, and ODS TRACE statements. However, the DISPLAY statement can improve performance when a large number of tables could be generated (such as in BY-group processing). The procedure processes the DISPLAY statement on a CAS server and thus sends only a subset of ODS tables to the SAS client. Because ODS statements are processed on a SAS client, first all the generated display tables are sent to the client, and then the client creates a subset.

If you use both DISPLAY and ODS statements together, the DISPLAY statement takes precedence over the ODS statements. Note that the ODS EXCLUDE statement processes tables that are sent to the client after they have been filtered by the DISPLAY statement. In some cases, it might appear that the ODS EXCLUDE
statement is taking precedence because it can further filter the tables. For more information about ODS, see *SAS Output Delivery System: Procedures Guide*.

You can specify the *table-list* as a list of table names, paths, partial pathnames, and regular expressions.

The table names that you can specify are listed in the section “ODS Table Names” on page 777. A path is a table name that is prefixed with dot-separated grouping information. For example, a SelectionSummary table that a procedure produces during a selection routine might have the path `Bygroup1.Summary.SelectionSummary`. A partial pathname does not include all groups; for example, `SelectionSummary` and `Summary.SelectionSummary` are partial pathnames for `Bygroup1.Summary.SelectionSummary`.

When you specify a table name or partial pathname, all display tables whose paths end in the specified name are selected for display or exclusion. For example, both `SelectionSummary` and `Summary.SelectionSummary` select `Bygroup1.Summary.SelectionSummary`.

A regular expression is enclosed in forward slashes (/). For example, specifying “/tions/” selects all pathnames that contain the substring “tions”; in particular, the `Bygroup1.Summary.SelectionSummary` table is selected. Specifying “^tions/” selects all pathnames that do not contain the substring “tions”; in particular, the `Bygroup1.Summary.SelectionSummary` table is not selected.

You can specify the following *options* after a slash (/):

- **CASESENSITIVE**
  performs a case-sensitive comparison of table names in the *table-list* to display table names when tables are subsetted for display. To preserve case, you must enclose table names in the *table-list* in quotation marks.

- **EXCLUDE**
  displays all display tables except those that you specify in the *table-list*.

- **EXCLUDEALL**
  suppresses display of all tables. This option takes precedence over the other options.

- **TRACE**
  displays the display table names, labels, and paths.

---

**DISPLAYOUT Statement**

**DISPLAYOUT** *table-spec-list* [/options] ;

The DISPLAYOUT statement enables you to create CAS output tables from your displayed output. This statement is similar to the ODS OUTPUT statement. For more information about ODS, see *SAS Output Delivery System: Procedures Guide*.

The *table-spec-list* specifies a list of CAS output tables to create. Each entry in the list has either a *key=value* format or a *key* format:

- **key=value** specifies *key* as the ODS table name, path, or partial pathname, and specifies *value* as the CAS output table name.

- **key** specifies *key* as the ODS table name and also as the CAS output table name.
The ODS table names that you can specify are listed in the section “ODS Table Names” on page 777. You cannot specify the ODS table named OutputCasTables in the table-spec-list.

Table names and partial pathnames are discussed under the DISPLAY statement. The DISPLAYOUT statement does not support regular expressions.

You can specify the following options after a slash (/):

- **INCLUDEALL**: creates output CAS tables for all display tables. The name of the created output CAS table is the same as the corresponding display table name. If you specify this option, the table-spec-list specification is ignored.
- **NOREPLACE**: does not replace any existing CAS output table of the same name.
- **REPEATED**: replicates all CAS output tables on all nodes.

## EFFECT Statement

```
EFFECT name=effect-type (variables </options>) ;
```

The EFFECT statement enables you to construct special collections of columns for design matrices. These collections are referred to as constructed effects to distinguish them from the usual model effects that are formed from continuous or classification variables, as discussed in the section “GLM Parameterization of Classification Variables and Effects” on page 54 in Chapter 3, “Shared Concepts.”

You can specify the following effect-types:

- **COLLECTION**: specifies a collection effect that defines one or more variables as a single effect that has multiple degrees of freedom. The variables in a collection are considered as a unit for purposes of estimation and inference.
- **MULTIMEMBER | MM**: specifies a multimember classification effect whose levels are determined by one or more variables that appear in a CLASS statement.
- **POLYNOMIAL | POLY**: specifies a multivariate polynomial effect in the specified numeric variables.
- **SPLINE**: specifies a regression spline effect whose columns are univariate spline expansions of one or more variables. A spline expansion replaces the original variable with an expanded or larger set of new variables.

Table 16.5 summarizes the options available in the EFFECT statement.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Collection Effects Options</strong></td>
<td>Displays the constituents of the collection effect</td>
</tr>
</tbody>
</table>
### Table 16.5  continued

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Multimember Effects Options</strong></td>
<td></td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays the levels of the multimember effect</td>
</tr>
<tr>
<td>NOEFFECT</td>
<td>Specifies that observations whose levels are all missing for the multimember variables should have 0 values in the corresponding design matrix columns</td>
</tr>
<tr>
<td>STDIZE</td>
<td>Standardizes the design matrix entries so that each observation has a sum of 1</td>
</tr>
<tr>
<td>WEIGHT=</td>
<td>Specifies the weight variable for the contributions of each classification effect</td>
</tr>
<tr>
<td><strong>Polynomial Effects Options</strong></td>
<td></td>
</tr>
<tr>
<td>DEGREE=</td>
<td>Specifies the degree of the polynomial</td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays details of the specified polynomial</td>
</tr>
<tr>
<td>MDEGREE=</td>
<td>Specifies the maximum degree of any variable in a term of the polynomial</td>
</tr>
<tr>
<td>NOSEPARATE</td>
<td>Treats the polynomial as a single effect with multiple degrees of freedom</td>
</tr>
<tr>
<td>STANDARDIZE=</td>
<td>Specifies centering and scaling suboptions for the variables that define the polynomial</td>
</tr>
<tr>
<td><strong>Spline Effects Options</strong></td>
<td></td>
</tr>
<tr>
<td>BASIS=</td>
<td>Specifies the type of basis (B-spline basis or truncated power function basis) for the spline effect</td>
</tr>
<tr>
<td>DATABOUNDARY</td>
<td>Uses the extremes of the data as boundary knots for a B-spline basis</td>
</tr>
<tr>
<td>DEGREE=</td>
<td>Specifies the degree of the spline effect</td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays the knots and locations for each spline basis function</td>
</tr>
<tr>
<td>KNOTMAX=</td>
<td>Requests equally spaced right-side boundary knots starting at the variables’ maximum and ending at the KNOTMAX= value</td>
</tr>
<tr>
<td>KNOTMETHOD=</td>
<td>Specifies how to construct the knots for the spline effect</td>
</tr>
<tr>
<td>KNOTMIN=</td>
<td>Requests equally spaced left-side boundary knots starting at the KNOTMIN= value and ending at the variables’ minimum value</td>
</tr>
<tr>
<td>NATURALCUBIC</td>
<td>Specifies a natural cubic spline basis for the spline effect</td>
</tr>
<tr>
<td>SEPARATE</td>
<td>Treats the spline basis for each variable as a separate effect when multiple variables are specified</td>
</tr>
<tr>
<td>SPLIT</td>
<td>Treats each design matrix column as a separate effect for selection methods</td>
</tr>
</tbody>
</table>

For more information about the syntax of these `effect-types` and how columns of constructed effects are computed, see the section “EFFECT Statement” on page 21 in Chapter 3, “Shared Concepts.”
Chapter 16: The PHSELECT Procedure

FREQ Statement

FREQ variable;

The variable in the FREQ statement identifies a numeric variable in the data set that contains the frequency of occurrence of each observation. PROC PHSELECT treats each observation as if it appears \( f \) times, where \( f \) is the value of the FREQ variable for the observation. If \( f \) is not an integer, it is truncated to an integer. If \( f \) is less than 1 or missing, the observation is not used in the analysis. When the FREQ statement is not specified, each observation is assigned a frequency of 1.

MODEL Statement

MODEL response < * censor (list)> = effects / options;

The MODEL statement identifies the variables to be used as the failure time variable, the optional censoring variable, and the explanatory effects, including covariates, main effects, interactions, and nested effects.

The name of the failure time variable precedes the equal sign. This name can optionally be followed by an asterisk, the name of the censoring variable, and a list of censoring values (separated by blanks or commas) enclosed in parentheses.

The censoring variable must be numeric, and the failure time variables must contain nonnegative values. Any observation that has a negative failure time is excluded from the analysis, as is any observation that has a missing value for any of the variables listed in the MODEL statement. Failure time variables in SAS date format are not recommended, because the dates might be translated into negative numbers and consequently the corresponding observations would be discarded.

If the censoring variable takes one of these values, the corresponding failure time is considered to be censored. Following the equal sign are the explanatory effects (sometimes called independent variables or covariates) for the model.

For information about constructing the model effects, see the section “Specification and Parameterization of Model Effects” on page 51 in Chapter 3, “Shared Concepts.”

Table 16.6 summarizes the options available in the MODEL statement. You can specify these options after a slash (/).

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model Options</td>
<td></td>
</tr>
<tr>
<td>CLB</td>
<td>Requests confidence limits</td>
</tr>
<tr>
<td>ENTRY=</td>
<td>Specifies the left-truncation time variable</td>
</tr>
<tr>
<td>INCLUDE=</td>
<td>Includes effects in all models for model selection</td>
</tr>
<tr>
<td>INFORMATIVE</td>
<td>Models missing values by using extra indicator variables</td>
</tr>
<tr>
<td>OFFSET=</td>
<td>Specifies the offset variable</td>
</tr>
<tr>
<td>START=</td>
<td>Includes effects in the initial model for model selection</td>
</tr>
<tr>
<td>TYPE3</td>
<td>Displays the Type 3 or joint tests of effects</td>
</tr>
</tbody>
</table>
The following list describes these options:

**CLB**
constructs confidence limits for each of the parameter estimates. The confidence level is 0.95 by default; you can change it by specifying the **ALPHA=** option in the **PROC PHSELECT** statement.

**ENTRY=** *variable*
specifies the left-truncation time variable. The left-truncation time is the time at which an individual starts to be observed.

**INCLUDE=** *n | single-effect | (effects)*
forces effects to be included in all models. If you specify **INCLUDE=** *n*, then the first *n* effects that you list in the **MODEL** statement are included in all models. If you specify **INCLUDE=** *single-effect* or **INCLUDE=** *(effects)*, then the specified *effects* are forced into all models. The *effects* that you specify in this option must be explanatory effects that you specify in the **MODEL** statement before the slash (/).

**INFORMATIVE**
models missing values by using extra model effects. These effects consist of dummy variables that take the value 1 when the value of a continuous model variable involved in the effect is missing, and take the value 0 otherwise. The missing value in the original model effect is replaced by the average value of the effect for the nonmissing values. For continuous-by-class effects, such as A*x, where A is a classification variable and x is a continuous variable, informative missingness creates multiple dummy columns and substitutes the effect mean of x that corresponds to the respective level of A. Missing values for classification variables are treated as valid levels. For more information about informative missingness, see the section “Informative Missingness” on page 78 in Chapter 3, “Shared Concepts.”

**OFFSET=** *variable*
specifies a *variable* to be used as an offset to the linear predictor. An offset plays the role of an effect whose coefficient is known to be 1. The offset variable cannot appear in the **CLASS** statement or elsewhere in the **MODEL** statement. Observations that have missing values for the offset variable are excluded from the analysis.

**START=** *n | single-effect | (effects)*
begins the selection process from the designated initial model for the forward and stepwise selection methods. If you specify **START=** *n*, then the starting model includes the first *n* effects that you list in the **MODEL** statement. If you specify **START=** *single-effect* or **START=** *(effects)*, then the starting model includes those specified *effects*. The *effects* that you specify in the **START=** option must be explanatory effects that you specify in the **MODEL** statement before the slash (/). This option is not available when you specify **METHOD=BACKWARD** in the **SELECTION** statement.

**TYPE3**
requests that Wald statistics for Type 3 contrasts be computed for each effect that you specify in the **MODEL** statement. For more information, see the section “Joint Tests and Type 3 Tests” on page 772.
The OUTPUT statement creates a data table that contains observationwise statistics that PROC PHSELECT computes after fitting the model. In order to avoid data duplication for large data tables, the variables in the input data table are not included in the output data table unless you specify them in the COPYVAR= option.

The output statistics are computed based on the final parameter estimates. If the optimization does not converge, then the output data table is not created.

For observations in which only the censoring variable is missing, values of the linear predictor and the predicted survival probabilities are computed even though these observations do not affect the model fit. This enables, for example, predicted survival probabilities to be computed for new observations.

You must specify the following option:

\textbf{OUT=} \textit{CAS-libref.data-table}

names the output data table for PROC PHSELECT to use. You must specify this option before any other options. \textit{CAS-libref.data-table} is a two-level name, where

- \textit{CAS-libref} refers to a collection of information that is defined in the LIBNAME statement and includes the \textit{caslib}, which includes a path to where the data table is to be stored, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about \textit{CAS-libref}, see the section “Using CAS Sessions and CAS Engine Librefs” on page 745.

- \textit{data-table} specifies the name of the output data table.

You can also specify the following syntax elements:

\textbf{COPYVAR=} \textit{variable}

\textbf{COPYVARS=} (\textit{variables})

transfers one or more \textit{variables} from the input data table to the output data table.

\textit{keyword} < = \textit{name} >

specifies a statistic to include in the output data table. Four statistics (DFBETA, RESSCH, RESSCO, and WTRESSCH) are multidimensional; each has a dimension equal to the number of parameters in the model. For the one-dimensional statistic, you can name the variable \textit{name}. For the multidimensional statistic, \textit{name} is used as the prefix; for example, if X1–X10 are the explanatory variables in the model, specifying RESSCH=sch creates the variables schX1–schX10 as the 10-dimensional Schoenfeld residual variables. If you do not provide a \textit{name}, the PHSELECT procedure assigns a default name based on the type of statistic that is requested.

Table 16.7 summarizes the \textit{keywords} available in the OUTPUT statement.

\begin{table}[h]
\centering
\begin{tabular}{|l|l|l|}
\hline
\textbf{Keyword} & \textbf{Description} & \textbf{Default Names} \\
\hline
\textbf{Statistic Options} & & \\
\textbf{CUMHAZ} & Specifies the predicted cumulative hazard & \_CUMHAZ\_ \\
\textbf{STDXBETA} & Specifies the standard error estimate of the linear predictor & \_STDXBETA\_ \\
\textbf{SURVIVAL} & Specifies the predicted survival probability & \_SURVIVAL\_ \\
\textbf{XBETA} & Specifies the linear predictor & \_XBETA\_ \\
\hline
\end{tabular}
\caption{OUTPUT Statement Keywords}
\end{table}
The following list describes these *keywords*. For more information, see the section “Predicted Values and Regression Diagnostics” on page 770.

**CUMHAZ**

specifies the predicted cumulative hazard function at the observed time. The default *name* is _CUMHAZ_.

**DFBETA**

requests the approximate changes in the parameter estimates \((\hat{\beta} - \hat{\beta}_{(j)})\) when the *jth* observation is omitted. These variables are a weighted transform of the score residual variables and are useful in assessing local influence. The default *name* that is used as a prefix is _DFBETA_.

**LD**

specifies the approximate likelihood displacement when the observation is omitted. This diagnostic can be used to assess the impact of each observation on the overall fit of the model. The default *name* is _LD_.

**RESDEV**

specifies the deviance residual. This variable is a transform of RESMART to achieve a more symmetric distribution. The default *name* is _RESDEV_.

**RESSMART**

specifies the martingale residual, which, at the observed time *t*, can be interpreted as the difference over \([0, t]\) in the observed number of events minus the expected number of events. The default *name* is _RESSMART_.

**RESSCH**

requests the Schoenfeld residuals, which are useful in assessing the proportional hazards assumption. Schoenfeld residuals are computed only at uncensored times and are missing for censored times. If an effect in the MODEL statement is not included in the final model, the corresponding Schoenfeld residuals are set to missing. The default *name* that is used as a prefix is _RESSCH_.

**RESSCO**

specifies the score residuals. The default *name* is _RESSCO_.

**WTRESSCH**

specifies the weighted Schoenfeld residuals. The default *name* that is used as a prefix is _WTRESSCH_.

---

**Table 16.7 continued**

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
<th>Default Names</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Diagnostic Options</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DFBETA</td>
<td>Specifies the standardize deletion parameter differences <em>DFBETA</em></td>
<td></td>
</tr>
<tr>
<td>LD</td>
<td>Specifies the likelihood displacement <em>LD</em></td>
<td></td>
</tr>
<tr>
<td>RESDEV</td>
<td>Specifies the deviance residual <em>RESDEV</em></td>
<td></td>
</tr>
<tr>
<td>RESMART</td>
<td>Specifies the martingale residual <em>RESSMART</em></td>
<td></td>
</tr>
<tr>
<td>RESSCH</td>
<td>Specifies the Schoenfeld residuals <em>RESSCH</em></td>
<td></td>
</tr>
<tr>
<td>RESSCO</td>
<td>Specifies the score residuals <em>RESSCO</em></td>
<td></td>
</tr>
<tr>
<td>WTRESSCH</td>
<td>Specifies the weighted Schoenfeld residuals <em>WTRESSCH</em></td>
<td></td>
</tr>
<tr>
<td><strong>Other Option</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ROLE</td>
<td>Specifies the observation role <em>ROLE</em></td>
<td></td>
</tr>
</tbody>
</table>
Chapter 16: The PHSELECT Procedure

RESSCO
requests the score residuals, which are a decomposition of the first partial derivative of the log likelihood. They can be used to assess the leverage that each subject exerts in the parameter estimation. The default name that is used as a prefix is _RESSCO_.

ROLE
specifies the numeric variable that indicates the role that each observation plays in fitting the model. The default name is _ROLE_. Table 16.8 shows how this variable is interpreted for each observation.

<table>
<thead>
<tr>
<th>Value</th>
<th>Observation Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Not used</td>
</tr>
<tr>
<td>1</td>
<td>Training</td>
</tr>
<tr>
<td>2</td>
<td>Validation</td>
</tr>
<tr>
<td>3</td>
<td>Testing</td>
</tr>
</tbody>
</table>

If you do not partition the input data by specifying a PARTITION statement, then the role variable value is 1 for observations that are used in fitting the model and 0 for observations that have at least one missing or invalid value for the response, regressor, frequency, or weight variables.

STDXBETA
specifies the standard error estimates ofXBETA. The default name is _STDXBETA_.

SURVIVAL
requests the predicted survival probabilities at the observed times. The default name is _SURVIVAL_.

WTRESSCH
requests the weighted Schoenfeld residuals, which are useful in investigating the nature of nonproportionality if the proportional hazard assumption does not hold. If an effect in the MODEL statement is not included in the final model, the corresponding weighted Schoenfeld residuals are set to missing. The default name that is used as a prefix is _WTRESSCH_.

XBETA
specifies the linear predictor. The default name is _XBETA_.

PARTITION Statement

PARTITION partition-option;

The PARTITION statement specifies how observations in the input data set are logically partitioned into disjoint subsets for model training, validation, and testing. For more information, see the section “Using Validation and Test Data” on page 80 in Chapter 3, “Shared Concepts.” Either you can designate a variable in the input data table and a set of formatted values of that variable to determine the role of each observation, or you can specify proportions to use for randomly assigning observations to each role.

You must specify exactly one of the following partition-options:
FRACTION(<TEST=fraction> <VALIDATE=fraction> <SEED=number>) randomly assigns specified proportions of the observations in the input data table to the roles. You specify the proportions for testing and validation by using the TEST= and VALIDATE= suboptions. If you specify both the TEST= and VALIDATE= suboptions, then the sum of the specified fractions must be less than 1 and the remaining fraction of the observations are assigned to the training role. The SEED= option specifies an integer that is used to start the pseudorandom number generator for random partitioning of data for training, testing, and validation. If you do not specify SEED=number or if number is less than or equal to 0, the seed is generated by reading the time of day from the computer’s clock.

ROLE=variable (<TEST=value> <TRAIN=value> <VALIDATE=value>) ROLEVAR=variable (<TEST=value> <TRAIN=value> <VALIDATE=value>) names the variable in the input data table whose values are used to assign roles to each observation. This variable cannot also appear as an analysis variable in other statements or options. The TEST=, TRAIN=, and VALIDATE= suboptions specify the formatted values of this variable that are used to assign observation roles. If you do not specify the TRAIN= suboption, then all observations whose role is not determined by the TEST= or VALIDATE= suboption are assigned to the training role.

The SELECTION statement performs model selection by examining whether effects should be added to or removed from the model according to rules that are defined by model selection methods. You can use the PLOT= option to produce graphical summaries of the selection process or the HIERARCHY= option to specify whether and how a requirement of model hierarchy is applied. The statement is fully documented in the section “SELECTION Statement” on page 36 in Chapter 3, “Shared Concepts.”

The PHSELECT procedure supports the following effect-selection methods in the SELECTION statement:

- **BACKWARD** performs backward elimination. This method starts with all effects in the model and deletes effects.
- **BACKWARD(FAST)** performs fast backward elimination. This method starts with all effects in the model and deletes effects without refitting the model.
- **FORWARD** performs forward selection. This method starts with no effects in the model and adds effects.
- **LASSO** performs model selection by the group LASSO method. This method adds and removes effects by using a sequence of LASSO steps. For more information, see the section “Group LASSO Selection” on page 69 in Chapter 3, “Shared Concepts.”
- **NONE** results in no model selection. This method fits the full model.
- **STEPWISE** performs stepwise selection. This method is similar to the FORWARD method, except that effects already in the model do not necessarily stay there.

By default, METHOD=STEPWISE.
The SELECT=, CHOOSE=, and STOP= *method-options* default to the Schwartz Bayesian criterion (SBC).

You can specify the following criteria in the SELECT=, CHOOSE=, and STOP= *method-options*:

- **AIC** uses Akaike’s information criterion (Akaike 1974) on the training data.
- **AICC** uses a small-sample bias-corrected version of Akaike’s information criterion, as promoted by Hurvich and Tsai (1989) and Burnham and Anderson (1998), on the training data.
- **SBC** uses the Schwarz Bayesian criterion (Schwarz 1978) on the training data.
- **SL** uses the significance level of the score test as the criterion on the training data (not available for the CHOOSE= option).
- **VALIDATE** uses the –2 log-likelihood (M2LL) value computed on the validation data as the criterion (not available for the SELECT= option).

If you specify METHOD=LASSO and do not specify either the CHOOSE= or STOP= option, then the model in the last LASSO step is chosen as the selected model.

**NOTE:** If you use the fast backward elimination method, then the –2 log-likelihood, AIC, AICC, and SBC statistics are approximated at each step where the model is not refit, and hence they do not match the values that are computed when that model is fit outside the selection routine. Similarly, if you specify SELECT=AIC, AICC, or SBC, the selection criteria are estimated (Lawless and Singhal 1978), and hence they do not match the values that are computed when that model is fit outside the selection routine.

**NOTE:** The default model hierarchy method is **HIERARCHY=NONE** for the stepwise, forward, and fast backward selection methods. The backward elimination method always uses the **HIERARCHY=SINGLE** *method-option*.

If you specify DETAILS=ALL, the LASSO method produces a table that displays the effects that are added or removed at each step and the corresponding fit statistics. For the other methods, you can specify the following values for the DETAILS= option:

- **SUMMARY** produces a summary table that shows the effect that is added or removed at each step along with the criteria specified in the SELECT=, CHOOSE=, and STOP= *method-options*. The summary table is produced by default if you do not specify the DETAILS= option.
- **STEPS** produces the preceding summary table and displays the results from fitting each model at each step.
- **ALL** produces the preceding tables and a detailed listing of all candidates at each step along with their ranking in terms of the selection criterion for addition to or removal from the model.

If you specify the PLOTS=CRITERIA or PLOTS=ALL option, then a plot of the fit criterion by the selection step is created for the AIC, AICC, and SBC statistics. If you also specify a PARTITION statement, then the same type of plot is created for the M2LL statistics.

If you specify the PLOTS=FITBYROLE or PLOTS=ALL option and a PARTITION statement, then a plot of the M2LL by the selection step for each role is created.

The PLOTS= option is not available for the LASSO method.
STRATA Statement

`STRATA variable < / MISSING > ;`

The STRATA statement specifies a `variable` whose values determine the stratification of the data. Strata are formed according to the nonmissing values of the STRATA `variable` unless you specify the `MISSING` option.

WEIGHT Statement

`WEIGHT variable ;`

The `variable` in the WEIGHT statement is used as a weight to perform a weighted analysis of the data. Observations that have nonpositive or missing weights are not included in the analysis. If a WEIGHT statement is not included, all observations that are used in the analysis are assigned a weight of 1.

Details: PHSELECT Procedure

Missing Values

Any observation that has missing values for the response, frequency, weight, offset, or explanatory variables is excluded from the analysis; however, missing values are valid for response and explanatory variables that are specified along with the `MISSING` option in the CLASS statement. Observations that have a nonpositive weight or a frequency less than 1 are also excluded.

The estimated linear predictor and the fitted probabilities are not computed for any observation that has missing offset or explanatory variable values. However, if only the response value is missing, the linear predictor and the fitted probabilities can be computed and output to a data table by using the OUTPUT statement.

You can also model the missing values by specifying the INFORMATIVE option in the MODEL statement. For more information about informative missingness, see the section “Informative Missingness” on page 78 in Chapter 3, “Shared Concepts.”

Cox Proportional Hazards Regression Model

The Cox proportional hazards regression model is a semiparametric model that assumes a parametric form for the effects of the explanatory variables, but it allows an unspecified form for the underlying survivor function. The survival time of each member of a population is assumed to follow its own hazard function, $\lambda_j(t)$, expressed as

$$\lambda_j(t) = \lambda(t; Z_j) = \lambda_0(t) \exp(Z_j' \beta)$$

where $\lambda_0(t)$ is an arbitrary and unspecified baseline hazard function, $Z_j$ is the vector of explanatory variables for the $j$th individual, and $\beta$ is the vector of unknown regression parameters that is associated with the explanatory variables.
Chapter 16: The PHSELECT Procedure

The population under study can consist of a number of subpopulations, each of which has its own baseline hazard function. A stratified analysis is needed to adjust for such subpopulation differences. Under the stratified model, the hazard function for the $j$th individual in the $i$th stratum is expressed as

$$\lambda_{ij}(t) = \lambda_{i0}(t) \exp(Z_{ij}' \beta)$$

where $\lambda_{i0}(t)$ is the baseline hazard function for the $i$th stratum and $Z_{ij}$ is the vector of explanatory variables for the individual. The regression coefficients are assumed to be the same for all individuals across all strata.

To estimate $\beta$, Cox (1972, 1975) introduced the partial likelihood function, which eliminates the unknown baseline hazard functions and accounts for censored survival times.

---

**Partial Likelihood for the Cox Model**

Let $Z_l$ denote the vector explanatory variables for the $l$th individual, and let $w_l$ be the associated weight. Let $t_1 < t_2 < \cdots < t_k$ denote the $k$ distinct, ordered event times. Let $d_i$ denote the multiplicity of failures at $t_i$; that is, $d_i$ is the size of the set $D_i$ of individuals that fail at $t_i$. Let $R_i$ be the risk set at $t_i$. Using this notation, the following equations describe the partial likelihood functions that PROC PHSELECT uses to estimate $\beta$.

**Breslow Likelihood**

$$L(\beta) = \prod_{i=1}^{k} \frac{\exp(\beta' \sum_{j \in D_i} Z_j)}{\sum_{l \in R_i} \exp(\beta' Z_l) d_l}$$

If you incorporate weights, the Breslow likelihood becomes

$$L(\beta) = \prod_{i=1}^{k} \frac{\exp(\beta' \sum_{j \in D_i} w_j Z_j)}{\sum_{l \in R_i} \sum_{j \in D_i} w_j \exp(\beta' Z_l)}$$

In a stratified analysis, the partial likelihood is the product of the partial likelihood functions for the individual strata.

The regression coefficients are estimated by maximizing the log of the partial likelihood function with respect to $\beta$.

---

**The LASSO Method of Model Selection**

**LASSO Selection**

The PHSELECT procedure implements the group LASSO method, which is described in the section “Group LASSO Selection” on page 69 in Chapter 3, “Shared Concepts.” The current section provides some background about the LASSO method that you need in order to understand the group LASSO method.
LASSO (least absolute shrinkage and selection operator) selection arises from a constrained form of ordinary least squares regression in which the sum of the absolute values of the regression coefficients is constrained to be smaller than a specified parameter. More precisely, let $Z = (z_1, z_2, \ldots, z_m)$ denote the matrix of covariates, and let $y$ denote the response. Then for a given parameter $t$, the LASSO regression coefficients $\beta = (\beta_1, \beta_2, \ldots, \beta_m)$ are the solution to the following constrained least squares problem:

$$\min ||y - Z\beta||^2 \quad \text{subject to} \quad \sum_{j=1}^{m} |\beta_j| \leq t$$

For the Cox proportional hazards regression model, the LASSO regression coefficients $\beta = (\beta_1, \beta_2, \ldots, \beta_m)$ are the solution to the following constrained optimization problem,

$$\min \{-l(\beta)\} \quad \text{subject to} \quad \sum_{j=1}^{m} |\beta_j| \leq t$$

where $l(\beta)$ is the log of the partial likelihood function $L(\beta)$ defined in the section “Partial Likelihood for the Cox Model” on page 768.

Provided that the LASSO parameter $t$ is small enough, some of the regression coefficients will be exactly 0. Hence, you can think of the LASSO method as selecting a subset of the regression coefficients for each LASSO parameter. By increasing the LASSO parameter in discrete steps, you obtain a sequence of regression coefficients for which the nonzero coefficients at each step correspond to selected parameters. For more information about the LASSO method, see, for example, Hastie, Tibshirani, and Friedman (2009).

---

**Model Fit and Assessment Statistics**

The statistics that are defined in this section are useful for assessing the fit of the model to your data; they are displayed in the “Fit Statistics” table. The statistics are computed for each data role when you specify a PARTITION statement.

**Information Criteria**

The calculation of the information criteria uses the following formulas, where $p$ denotes the number of effective parameters in the candidate model, $F$ denotes the sum of the frequencies of the uncensored observations that are used, and $l$ is the log partial likelihood that is evaluated at the converged estimates:

$$\text{AIC} = -2l + 2p$$

$$\text{AICC} = \begin{cases} 
-2l + 2pF/(F - p - 1) & \text{when } F > p + 2 \\
-2l + 2p(p + 2) & \text{otherwise}
\end{cases}$$

$$\text{SBC} = -2l + p \log(F)$$

If you do not specify a FREQ statement, $F$ equals $n$, the number of uncensored observations that are used.
Predicted Values and Regression Diagnostics

For the $i$th observation, let $(X_i, \Delta_i, Z_i)$ represent the failure time, the event indicator, and the vector of covariate values, respectively. Let $\beta$ be the vector of regression coefficients. For $t \geq 0$, let

$$
Y_i(t) = I(X_i \geq t),
$$
$$
N_i(t) = I(X_i \leq t),
$$
$$
S^{(0)}(\beta, t) = \sum_i Y_i(t)e^{\beta'Z_i},
$$
$$
S^{(1)}(\beta, t) = \sum_i Y_i(t)e^{\beta'Z_i}Z_i,
$$
$$
\tilde{Z}(\beta, t) = \frac{S^{(1)}(\beta, t)}{S^{(0)}(\beta, t)},
$$
$$
d\Lambda_0(\beta, t) = \sum_i \frac{dN_i(t)}{S^{(0)}(\beta, t)},
$$
$$
dM_i(\beta, t) = dN_i(t) - Y_i(t)e^{\beta'Z_i(t)}d\Lambda_0(\beta, t).
$$

Let $\hat{\beta}$ be the estimator of $\beta$, and let $I(\hat{\beta})$ be the observed information matrix.

Predicted Values

Let $z$ be a vector of covariates. The linear predictor and its standard error estimate are, respectively,

$$
z'\hat{\beta} \quad \text{and} \quad \sqrt{z'I^{-1}(\hat{\beta})z}
$$

The predicted cumulative hazard at a time point $t$ is

$$
\hat{\Lambda}(t, z) = e^{\hat{\beta}'z} \int_0^td\Lambda_0(\hat{\beta}, s)
$$

and the predicted survival probability is

$$
\hat{S}(t, z) = e^{-\hat{\Lambda}(t, z)}
$$

Residuals

The martingale residual for the $i$th observation is

$$
\hat{M}_i = \Delta_i - \hat{\Lambda}(X_i, Z_i)
$$

The deviance residual is a transform of the martingale residual,

$$
\hat{D}_i = \text{sign}(\hat{M}_i)\sqrt{2[-\hat{M}_i - \Delta_i \log(\Delta_i - \hat{M}_i)]}
$$

The square root shrinks large negative martingale residuals, and the logarithmic transformation expands martingale residuals that are close to unity.
The Schoenfeld (1982) residual vector is calculated per event time. If the \( i \)th observation is uncensored, the Schoenfeld residual vector is

\[
\hat{U}_i = Z_i - \tilde{Z}(\hat{\beta}, X_i)
\]

Otherwise, the elements of the Schoenfeld residual vector are set to missing values. Under the proportional hazards assumption, the Schoenfeld residuals have the sample path of a random walk; therefore, they are useful in assessing time trend or lack of proportionality. Harrell (1986) proposed a \( z \)-transform of the Pearson correlation between these residuals and the rank order of the failure time as a test statistic for nonproportional hazards.

The score residual vector for the \( i \)th observation is

\[
\hat{L}_i = \int_0^\infty [Z_i(s) - \tilde{Z}(\hat{\beta}, s)]dM_i(\hat{\beta}, s)
\]

The score residuals are a decomposition of the first partial derivative of the log likelihood. They are useful in assessing the influence of each observation on individual parameter estimates. Therneau, Grambsch, and Fleming (1990) have considered a Kolmogorov-type test based on the cumulative sum of the residuals for detecting nonproportional hazards.

**Other Regression Diagnostics**

The vector of weighted Schoenfeld residuals, \( W_i \), is computed as

\[
W_i = n_e \mathcal{I}^{-1}(\hat{\beta}) \hat{U}_i
\]

where \( n_e \) is the total number of events and \( \hat{U}_i \) is the vector of Schoenfeld residuals for the \( i \) observation. The weighted Schoenfeld residuals are useful in assessing the proportional hazards assumption. The idea is that most of the common alternatives to the proportional hazards can be cast in terms of a time-varying coefficient model,

\[
\lambda(t, Z) = \lambda_0(t) \exp(\beta_1(t)Z_1 + \beta_2(t)Z_2 + \cdots)
\]

where \( \lambda(t, Z) \) and \( \lambda_0(t) \) are hazard rates. Let \( \hat{\beta}_j \) and \( r_{ij} \) be the \( j \)th component of \( \hat{\beta} \) and \( r_i \), respectively. Grambsch and Therneau (1994) suggest using a smoothed plot of \( (\hat{\beta}_j + r_{ij}) \) versus \( t_i \) to discover the functional form of the time-varying coefficient \( \beta_j(t) \). A zero slope indicates that the coefficient does not vary with time.

The DFBETA diagnostics approximate the changes in the parameter estimates, \( \delta_i \hat{\beta}_j = \hat{\beta} - \hat{\beta}_{(-i)} \), when the \( i \)th observation is left out. The \( j \)th component of \( \delta_i \hat{\beta} \) can be used to assess any unusual effect of the \( i \)th observation on \( \hat{\beta}_j \). The exact computation of \( \delta_i \hat{\beta} \) involves refitting the model each time a subject is omitted. Cain and Lange (1984) derived the following approximation of \( \delta_i \hat{\beta} \) as weighted score residuals,

\[
\text{DFBETA}_i = \mathcal{I}^{-1}(\hat{\beta}) \hat{L}_i
\]

where \( \hat{L}_i \) is the score residual vector of the \( i \)th observation.

The LD statistic approximates the likelihood displacement, which is the amount by which minus twice the log likelihood \((-2 \log L(\hat{\beta}))\), under a fitted model, changes when each observation in turn is left out. When the \( i \)th subject is omitted, the likelihood displacement is

\[
2 \log L(\hat{\beta}) - 2 \log L(\hat{\beta}_{(-i)})
\]
where $\hat{\beta}_{(-i)}$ is the vector of parameter estimates that you obtain by fitting the model without the $i$th observation. Instead of refitting the model without the $i$th observation, Pettitt and Bin Daud (1989) propose that the likelihood displacement for the $i$th observation be approximated by

$$LD_i = \hat{L}_i' I^{-1}(\hat{\beta}) \hat{L}_i$$

where $\hat{L}_i$ is the score residual vector of the $i$th observation.

## Joint Tests and Type 3 Tests

Linear hypotheses for $\beta$ are expressed in matrix form as

$$H_0: L\hat{\beta} = c$$

where $L$ is a matrix of coefficients for the linear hypotheses and $c$ is a vector of constants. The vector of regression coefficients $\beta$ includes slope parameters and intercept parameters. The Wald chi-square statistic for testing $H_0$ is computed as

$$\chi^2_W = (L\hat{\beta} - c)' [L\hat{V}(\hat{\beta})L']^{-1} (L\hat{\beta} - c)$$

where $\hat{V}(\hat{\beta})$ is the estimated covariance matrix. Under $H_0$, $\chi^2_W$ has an asymptotic chi-square distribution with $r$ degrees of freedom, where $r$ is the rank of $L$.

For models that use less-than-full-rank parameterization (as specified by the PARAM=GLM option in the CLASS statement), a Type 3 test of an effect of interest (main effect or interaction) is a test of the Type III estimable functions that are defined for that effect. When the model contains no missing cells, the Type 3 test of a main effect is equivalent to testing the hypothesis of equal marginal means. For more information about Type III estimable functions, see the chapter “The GLM Procedure” and the section “The Four Types of Estimable Functions” in SAS/STAT User’s Guide. Also see Littell, Freund, and Spector (1991).

For models that use full-rank parameterization, all parameters are estimable when there are no missing cells, so it is unnecessary to define estimable functions. The standard test of an effect of interest in this case is the joint test that the values of the parameters associated with that effect are 0. For a model that uses effects parameterization (as specified by the PARAM=EFFECT option in the CLASS statement), the joint test for a main effect is equivalent to testing the equality of marginal means. For a model that uses reference parameterization (as specified by the PARAM=REF option in the CLASS statement), the joint test is equivalent to testing the equality of cell means at the reference level of the other model effects. For more information about the coding scheme and the associated interpretation of results, see Muller and Fetterman (2002, Chapter 14).

If there is no interaction term, the Type 3 test of an effect for a model that uses GLM parameterization is the same as the joint test of the effect for the model that uses full-rank parameterization. In this situation, the joint test is also called the Type 3 test. For a model that contains an interaction term and no missing cells, the Type 3 test of a component main effect under GLM parameterization is the same as the joint test of the component main effect under effect parameterization. Both test the equality of cell means. But this Type 3 test differs from the joint test under reference parameterization, which tests the equality of cell means at the reference level of the other component main effect. If some cells are missing, you can obtain meaningful tests only by testing a Type III estimation function, so in this case you should use GLM parameterization.

The results of a Type 3 test or a joint test do not depend on the order in which you specify the terms in the MODEL statement.
Multithreading

The PHSELECT procedure allocates data to different threads and calculates the likelihood function, gradient, and Hessian by accumulating the values from all threads. For more information about how PROC PHSELECT uses threads, see the section “Multithreading” on page 81 in Chapter 3, “Shared Concepts.”

Optimization Algorithms

Several optimization techniques are available in PROC PHSELECT. You can choose a particular optimizer by using the TECHNIQUE= option in the PROC PHSELECT statement. Table 16.9 summarizes the optimization techniques available in PROC PHSELECT.

<table>
<thead>
<tr>
<th>TECHNIQUE=</th>
<th>Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRUREG</td>
<td>Trust region method</td>
</tr>
<tr>
<td>NEWRAP</td>
<td>Newton-Raphson method with line search</td>
</tr>
<tr>
<td>NRRIDG</td>
<td>Newton-Raphson method with ridging</td>
</tr>
<tr>
<td>QUANEW</td>
<td>Quasi-Newton methods</td>
</tr>
<tr>
<td>DBLDOG</td>
<td>Double-dogleg method</td>
</tr>
<tr>
<td>CONGRA</td>
<td>Conjugate gradient methods</td>
</tr>
<tr>
<td>NMSIMP</td>
<td>Nelder-Mead simplex method</td>
</tr>
</tbody>
</table>

There is no algorithm for optimizing general nonlinear functions that always finds the global optimum for a general nonlinear optimization problem in a reasonable amount of time. Because no single optimization technique is always superior to others, PROC PHSELECT provides a variety of optimization techniques that work well in various circumstances. However, you can devise problems for which none of the techniques in PROC PHSELECT can find the correct solution. Moreover, nonlinear optimization can be computationally expensive in terms of time and memory, so you must be careful when matching an algorithm to a problem. The section “Choosing an Optimization Algorithm” on page 82 in Chapter 3, “Shared Concepts,” is helpful in choosing a suitable optimization algorithm.

In addition, PROC PHSELECT uses the finite-difference method to compute second-order derivatives, unless you specify the HESSIAN option. The HESSIAN option requires extra memory, the size of which also depends on the number of distinct uncensored failure times in the data.

Displayed Output

The following sections describe the output that PROC PHSELECT produces. The output is organized into various tables, which are discussed in their order of appearance.
Model Information

The “Model Information” table displays basic information about the model, such as the response variable, the frequency variable, the link function, and the distribution of the data that is assumed by the PHSELECT procedure.

Number of Observations

The “Number of Observations” table displays the number of observations that are read from the input data table and the number of observations that are used in the analysis. If you specify a FREQ statement, the table displays the sum of the frequencies that are read and used. If you specify a WEIGHT statement, the table displays the sum of $f_i w_i$ that are read and used. If you specify a PARTITION statement, the table displays the values for each role.

$-2$ Log Likelihood for Null Model

When you specify the LOGLIKENULL option in the PROC PHSELECT statement, the PHSELECT procedure displays the $-2$ log-likelihood value of the null model for the training data. If you also specify a PARTITION statement, PROC PHSELECT displays the corresponding statistic for the validation data and the test data.

Class Level Information

The “Class Level Information” table lists the levels of every variable that you specify in the CLASS statement. You should check this information to make sure that the data are correct. You can adjust the order of the CLASS variable levels by using the ORDER= option in the CLASS statement. You can suppress the “Class Level Information” table completely or partially by using the NOCLPRINT= option in the PROC PHSELECT statement.

If the classification variables use a nonsingular parameterization, the “Class Level Information” table also displays the reference value for each variable.

Selection Information

When you specify the SELECTION statement, the PHSELECT procedure by default produces a series of tables that display information about the model selection. The “Selection Information” table informs you about the model selection method, selection and stop criteria, and other parameters that govern the selection. You can suppress this table by specifying DETAILS=NONE in the SELECTION statement.

Iteration History

When you specify the ITHIST option in the PROC PHSELECT statement, the “Iteration History” table displays, for each iteration of the optimization, the number of function evaluations (including gradient and Hessian evaluations), the value of the objective function, the change in the objective function from the previous iteration, and the absolute value of the largest (projected) gradient element.

Convergence Status

The convergence status table is a small ODS table that appears as a message that indicates whether the optimization succeeded and which convergence criterion was met. If the optimization fails, the message
indicates the reason for the failure. If you save the convergence status table to an output data set, a numeric Status variable is added that enables you to assess convergence programmatically. The values of the Status variable indicate the following:

0 Convergence was achieved, or an optimization was not performed (because TECHNIQUE=NONE is specified).

1 The objective function could not be improved.

2 Convergence was not achieved because of a user interruption or because a limit was exceeded, such as the maximum number of iterations or the maximum number of function evaluations. To modify these limits, see the MAXITER=, MAXFUNC=, and MAXTIME= options in the PROC PHSELECT statement.

3 Optimization failed to converge because function or derivative evaluations failed at the starting values or during the iterations or because a feasible point that satisfies the parameter constraints could not be found in the parameter space.

**Entry and Removal Candidates**

When you specify DETAILS=ALL or DETAILS=STEPS in the SELECTION statement, the PHSELECT procedure produces the “Entry Candidates” or “Removal Candidates” table, which displays the effect names and values of the criterion used to select entering or departing effects at each step of the selection process. For each step, the effects are displayed in sorted order from best to worst of the selection criterion.

**Selection Summary**

When you specify the SELECTION statement, the PHSELECT procedure produces the “Selection Summary” table, which displays information about which effects were added to or removed from the model in the various steps of the model selection process. The statistic that led to the entry or removal decision is also displayed. You can request further details about the model selection steps by specifying DETAILS=STEPS or DETAILS=ALL in the SELECTION statement. You can suppress the display of this table by specifying DETAILS=NONE in the SELECTION statement.

If you specify the LASSO selection method, then this table displays information about which effect was added to the model, the number of effects in the model, the lambda value, and the information criteria.

**Stop Reason**

When you specify the SELECTION statement, the PHSELECT procedure produces a simple table that tells you why model selection stopped.

**Selection Reason**

When you specify the SELECTION statement, the PHSELECT procedure produces a simple table that tells you why the final model was selected.

**Selected Effects**

When you specify the SELECTION statement, the PHSELECT procedure produces a simple table that tells you which effects are in the final model.
Dimensions

The “Dimensions” table displays size measures that are derived from the model. It displays the number of columns in the current design matrix, the number of effects in the current design, the largest number of design columns associated with an effect, the rank of the matrix, and the number of parameters in the current model, including any scale parameters.

Fit Statistics

The “Fit Statistics” table displays a variety of likelihood-based measures of fit. The values that the “Fit Statistics” table displays are not based on a normalized log-likelihood function. If you specify a PARTITION statement, the table displays the values for each role along with statistics for comparing the training, validation, and testing results. For more information about the statistics that this table displays, see the section “Model Fit and Assessment Statistics” on page 769.

Parameter Estimates

The parameter estimates, their estimated (asymptotic) standard errors, and $p$-values for the hypothesis that the parameter is 0 are displayed in the “Parameter Estimates” table. If you request confidence intervals by specifying the CLB option in the MODEL statement, confidence limits are displayed for the parameters.

Parameter Estimates Covariance Matrix

When you specify the COVB option in the PROC PHSELECT statement, the PHSELECT procedure displays the covariance matrix of the parameter estimates. The covariance matrix is computed as the inverse of the negative of the matrix of second derivatives of the log-likelihood function with respect to the model parameters (the Hessian matrix), evaluated at the parameter estimates.

Parameter Estimates Correlation Matrix

When you specify the CORRB option in the PROC PHSELECT statement, the PHSELECT procedure displays the correlation matrix of the parameter estimates.

Model Analysis of Variance (Type III)

When you specify the TYPE3 option in the MODEL statement, the PHSELECT procedure produces the “Model Analysis of Variance (Type III)” table. This table displays tests that all parameters for a particular effect are equal to zero. For more information, see the section “Joint Tests and Type 3 Tests” on page 772.

Timing

The “Timing” table displays the amount of time (in seconds) that PROC PHSELECT required to perform different tasks in the analysis.

OutputCasTables Table

The OutputCasTables table is a special table that has information about each CAS table that is created during a CAS action execution. The information for each CAS table consists of the CAS table name, the caslib in which the table resides, and the number of columns and rows in the CAS table. Because this table is
not a typical ODS table that contains analytical results, you cannot include it in the `table-spec-list` in the DISPLAYOUT statement.

**ODS Table Names**

Each table that the PHSELECT procedure creates has a name associated with it. You must use this name to refer to the table when you use the DISPLAY statement, the DISPLAYOUT statement, or ODS statements. These names are listed in Table 16.10.

NOTE: The EFFECT statement also creates tables that are not listed in this section. For information about these tables, see the section “ODS Table Names” on page 35 in Chapter 3, “Shared Concepts.”

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClassInfo</td>
<td>Level information from the CLASS statement</td>
<td>CLASS</td>
<td>Default</td>
</tr>
<tr>
<td>ConvergenceStatus</td>
<td>Status of optimization at conclusion of optimization</td>
<td></td>
<td>Default</td>
</tr>
<tr>
<td>CorrB</td>
<td>Correlation matrix of parameter estimates</td>
<td>PROC PHSELECT</td>
<td>CORRB</td>
</tr>
<tr>
<td>CovB</td>
<td>Covariance matrix of parameter estimates</td>
<td>PROC PHSELECT</td>
<td>COVB</td>
</tr>
<tr>
<td>Dimensions</td>
<td>Model dimensions</td>
<td></td>
<td>Default</td>
</tr>
<tr>
<td>EntryCandidates</td>
<td>Details about candidates for entry into the model</td>
<td>SELECTION</td>
<td>METHOD=FORWARD or STEPWISE and DETAILS=STEP</td>
</tr>
<tr>
<td>FitStatistics</td>
<td>Fit statistics</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>IterHistory</td>
<td>Iteration history</td>
<td>PROC PHSELECT</td>
<td>ITHIST</td>
</tr>
<tr>
<td>LogLikeNull</td>
<td>–2 log likelihood for null model</td>
<td>PROC PHSELECT</td>
<td>LOGLIKENEULL TYPE3</td>
</tr>
<tr>
<td>ModelAnova</td>
<td>Model analysis of variance (Type III)</td>
<td>MODEL</td>
<td></td>
</tr>
<tr>
<td>ModelInfo</td>
<td>Information about the modeling environment</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>NObs</td>
<td>Number of observations read and used, and breakdown of censored and uncensored values</td>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>OutputCasTable</td>
<td>See the section “OutputCasTables Table” on page 776</td>
<td>OUTPUT DISPLAYOUT</td>
<td>OUT=</td>
</tr>
<tr>
<td>ParameterEstimates</td>
<td>Solutions for the parameter estimates associated with effects in MODEL statements</td>
<td>Default</td>
<td></td>
</tr>
</tbody>
</table>
Table 16.10  continued

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>RemovalCandidates</td>
<td>Details about candidates for removal from the model</td>
<td>SELECTION</td>
<td>METHOD=BACKWARD or STEPWISE and DETAILS=STEP</td>
</tr>
<tr>
<td>SelectedEffects</td>
<td>List of effects selected for the model</td>
<td>SELECTION</td>
<td>Default</td>
</tr>
<tr>
<td>SelectionInfo</td>
<td>Information about the settings for model selection</td>
<td>SELECTION</td>
<td>Default</td>
</tr>
<tr>
<td>SelectionReason</td>
<td>Reason why the particular model was selected</td>
<td>SELECTION</td>
<td>Default</td>
</tr>
<tr>
<td>SelectionSummary</td>
<td>Summary information about model selection steps</td>
<td>SELECTION</td>
<td>Default</td>
</tr>
<tr>
<td>StopReason</td>
<td>Reason for termination of model selection</td>
<td>SELECTION</td>
<td>Default</td>
</tr>
<tr>
<td>Timing</td>
<td>Absolute and relative times for tasks performed by the procedure</td>
<td>Default</td>
<td></td>
</tr>
</tbody>
</table>

**ODS Graphics**

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in the “Statistical Graphics Using ODS” chapter in *SAS/STAT User’s Guide*.

Before you create graphs, ODS Graphics must be enabled (for example, with the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section “Enabling and Disabling ODS Graphics” in that chapter.

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section “A Primer on ODS Statistical Graphics” in that chapter.

When ODS Graphics is enabled, the SELECTION statement can produce plots to help evaluate the selection process. For information about these plots, see the section “Model Selection Plots” on page 71 in Chapter 3, “Shared Concepts.”

PROC PHSELECT assigns a name to each graph that it creates using ODS. You can use these names to refer to the graphs when using ODS. The names are listed in Table 16.11.

**Table 16.11  Graphs Produced by PROC PHSELECT**

<table>
<thead>
<tr>
<th>ODS Graph Name</th>
<th>Plot Description</th>
<th>PLOTS= Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoefficientPanel</td>
<td>Coefficients and CHOOSE= criterion by step</td>
<td>COEFFICIENTS</td>
</tr>
<tr>
<td>ChooseCriterionPlot</td>
<td>CHOOSE= criterion by step</td>
<td>COEFFICIENTS(UNPACK)</td>
</tr>
<tr>
<td>CoefficientPlot</td>
<td>Coefficients by step</td>
<td>COEFFICIENTS(UNPACK)</td>
</tr>
</tbody>
</table>
Table 16.11 continued

<table>
<thead>
<tr>
<th>ODS Graph Name</th>
<th>Plot Description</th>
<th>PLOTS= Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>CriterionPanel</td>
<td>Fit criteria by step</td>
<td>CRITERIA</td>
</tr>
<tr>
<td>TEST_M2LLPlot</td>
<td>–2 log likelihood on testing data by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>TRAIN_M2LLPlot</td>
<td>–2 log likelihood on training data by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>VAL_M2LLPlot</td>
<td>–2 log likelihood on validation data by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>FitByRolePlot</td>
<td>Progression of average square error by role</td>
<td>FITBYROLE</td>
</tr>
</tbody>
</table>

Examples: PHSELECT Procedure

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

The following examples assume that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

Example 16.1: Model Selection

The following statements examine the same data as in the section “Getting Started: PHSELECT Procedure” on page 746, but they request model selection via the forward selection technique. Effects that provide the best improvement to the selection criterion, the Schwartz Bayesian criterion (SBC), are added until no more effects can improve the selection criterion. The DETAILS=ALL option in the SELECTION statement produces all tables that are related to model selection. The PLOTS=ALL option produces graphics to help you interpret the selection process. ODS Graphics must be enabled before you can request plots. (For more information about ODS Graphics, see the section “ODS Graphics” on page 778.)

```sas
ods graphics on;
proc phselect data=mycas.getStarted;
   class C1-C3;
   model Time*Status(0) = C1-C3 X1-X4;
   selection method=forward(stop=sbc select=sbc) details=all plots=all;
run;
```

The model selection tables are shown in Output 16.1.1 through Output 16.1.3. Results from the selected model are shown in Output 16.1.4 and Output 16.1.5. Selection graphics that the PLOTS= option produces are displayed in Output 16.1.6 and Output 16.1.7.
The “Selection Information” table in Output 16.1.1 summarizes the settings for the model selection. Effects are added to the model only if they produce a significant improvement, which is determined by comparing their SBC values. The forward selection stops three steps after the smallest SBC is obtained or when all effects have been added to the model.

**Output 16.1.1** Selection Information

<table>
<thead>
<tr>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selection Method</td>
<td>Forward</td>
</tr>
<tr>
<td>Select Criterion</td>
<td>SBC</td>
</tr>
<tr>
<td>Stop Criterion</td>
<td>SBC</td>
</tr>
<tr>
<td>Effect Hierarchy Enforced</td>
<td>None</td>
</tr>
<tr>
<td>Stop Horizon</td>
<td>3</td>
</tr>
</tbody>
</table>

For each step of the selection process, the DETAILS=ALL option displays the candidate effects for entering the model along with their SELECT= criterion. Output 16.1.2 displays this table for the first step; the other steps are not shown here.

**Output 16.1.2** Step1 Entry Candidates

<table>
<thead>
<tr>
<th>Rank</th>
<th>Effect</th>
<th>SBC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>X1</td>
<td>536.7462</td>
</tr>
<tr>
<td>2</td>
<td>X4</td>
<td>539.8497</td>
</tr>
<tr>
<td>3</td>
<td>C2</td>
<td>544.9948</td>
</tr>
<tr>
<td>4</td>
<td>X3</td>
<td>548.2815</td>
</tr>
<tr>
<td>5</td>
<td>X2</td>
<td>548.9490</td>
</tr>
<tr>
<td>6</td>
<td>C3</td>
<td>549.1132</td>
</tr>
<tr>
<td>7</td>
<td>C1</td>
<td>551.8649</td>
</tr>
</tbody>
</table>

The DETAILS=ALL option also displays the dimensions, fit statistics, and parameter estimates at each step of the selection process; these details are not shown here.

When the selection procedure is complete, the “Selection Summary” table in Output 16.1.3 shows the effects that were added to the model and the value of their selection criterion (and the choose and stop criteria, if they are specified). In step 1, effect X1 made the most significant contribution to the model among the candidate effects, according to the SBC statistic. In step 2, X4 made the most significant contribution when an effect was added to a model that contains X1. In the three subsequent steps, no effect could be added to the model that would reduce the SBC value, so variable selection stopped because the stop horizon (see Output 16.1.1) indicates that at most three steps beyond the minimum SBC value are taken.

In Output 16.1.3, the “Selection Summary” table is followed by three small tables that summarize why the process stopped and which model is selected.
**Output 16.1.3** Selection Summary Information

The PHSELECT Procedure

Selection Details

<table>
<thead>
<tr>
<th>Step</th>
<th>Effect</th>
<th>Entered</th>
<th>Number Effects In</th>
<th>SBC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>X1</td>
<td>1</td>
<td>1</td>
<td>536.7462</td>
</tr>
<tr>
<td>2</td>
<td>X4</td>
<td>2</td>
<td>2</td>
<td>535.1652*</td>
</tr>
<tr>
<td>3</td>
<td>C1</td>
<td>3</td>
<td>3</td>
<td>538.2917</td>
</tr>
<tr>
<td>4</td>
<td>X2</td>
<td>4</td>
<td>4</td>
<td>539.2661</td>
</tr>
<tr>
<td>5</td>
<td>X3</td>
<td>5</td>
<td>5</td>
<td>542.4842</td>
</tr>
</tbody>
</table>

* Optimal Value Of Criterion

Selection stopped at a local minimum of the SBC criterion.

The model at step 2 is selected.

Selected Effects: X1 X4

**Output 16.1.4** displays information about the selected model. Notice that the –2 log-likelihood value in the “Fit Statistics” table is larger than the value for the full model in Figure 16.7. This is expected because the selected model contains only a subset of the parameters. Because the selected model is more parsimonious than the full model, the discrepancy between the –2 log likelihood and the information criteria is less severe than that shown in Figure 16.7 in the “Getting Started” example.

**Output 16.1.4** Dimensions and Fit Statistics

The PHSELECT Procedure

Selected Model

<table>
<thead>
<tr>
<th>Dimensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Effects</td>
</tr>
<tr>
<td>Max Effect Columns</td>
</tr>
<tr>
<td>Columns in Design</td>
</tr>
<tr>
<td>Rank of Design</td>
</tr>
</tbody>
</table>

Fit Statistics

<table>
<thead>
<tr>
<th>-2 Log Likelihood</th>
<th>526.74445</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIC (smaller is better)</td>
<td>530.74445</td>
</tr>
<tr>
<td>AICC (smaller is better)</td>
<td>530.91346</td>
</tr>
<tr>
<td>SBC (smaller is better)</td>
<td>535.35258</td>
</tr>
</tbody>
</table>

The parameter estimates of the selected model are shown in **Output 16.1.5**. Notice that the effects are listed in the “Parameter Estimates” table in the order in which they are specified in the MODEL statement and not necessarily in the order in which they are added to the model.
Output 16.1.5 Parameter Estimates

<table>
<thead>
<tr>
<th>Parameter</th>
<th>DF</th>
<th>Estimate</th>
<th>Standard Error</th>
<th>Chi-Square</th>
<th>Pr &gt; ChiSq</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1</td>
<td>1</td>
<td>1.605306</td>
<td>0.513811</td>
<td>9.7613</td>
<td>0.0018</td>
</tr>
<tr>
<td>X4</td>
<td>1</td>
<td>-0.123304</td>
<td>0.046697</td>
<td>6.9723</td>
<td>0.0083</td>
</tr>
</tbody>
</table>

The coefficient panel in Figure 16.1.6 enables you to visualize the selection process. In this plot, standardized coefficients of all the effects that are selected at some step of the stepwise method are plotted as a function of the step number. This enables you to assess the relative importance of the effects that are selected at any step of the selection process and to know when effects entered the model. The lower plot in the panel shows how the criterion that is used to choose the selected model changes as effects enter or leave the model.

Output 16.1.6 Coefficient Progression

The criterion panel in Figure 16.1.7 provides a graphical view of the progression of the fit criteria as the selection process evolves. Notice at Step 2 that the SBC value seems to be at its minimum. Because the stop horizon value is 3 (see Output 16.1.1), three more steps are taken to determine if Step 2 is a global optimum. None of these three subsequent steps have a smaller SBC value than in Step 2, so the global optimum is at step 2.
Example 16.2: Stratified Analysis

In the `getStarted` data set that is presented in the section “Getting Started: PHSELECT Procedure” on page 746, suppose the variable `C3` represents the gender of the subjects. It is conceivable that males and females belong to different subpopulations and might have different baseline hazard functions. A stratified analysis is conducted by specifying `C3` as the stratifying variable in the `STRATA` statement. The `CODE` statement requests that a text file named `ScoreCode.txt` be created. This file contains SAS DATA step code to predict survival probabilities at time points that are specified by the `TIMEPOINT=` option. By default, PROC PHSELECT uses the estimated quartiles of the KM curve (no covariates) of the training data. The `SHOWTIME` option produces variables that contain these estimated time points.

```sas
proc phselect data=mycas.getStarted;
  class C1 C2;
  model Time*Status(0) = C1 C2 X1-X4;
  strata C3;
  code file='ScoreCode.txt' showtime;
run;
```
The “Parameter Estimates” table in Output 16.2.1 shows the resulting regression parameter estimates of the stratified analysis.

**Output 16.2.1** Parameter Estimates

<table>
<thead>
<tr>
<th>Parameter</th>
<th>DF</th>
<th>Estimate</th>
<th>Standard Error</th>
<th>Chi-Square</th>
<th>Pr &gt; ChiSq</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1 Critical</td>
<td>1</td>
<td>0.301314</td>
<td>0.512375</td>
<td>0.3458</td>
<td>0.5565</td>
</tr>
<tr>
<td>C1 High</td>
<td>1</td>
<td>-1.030679</td>
<td>0.334457</td>
<td>9.4966</td>
<td>0.0021</td>
</tr>
<tr>
<td>C1 Low</td>
<td>1</td>
<td>-0.649179</td>
<td>0.342083</td>
<td>3.6014</td>
<td>0.0577</td>
</tr>
<tr>
<td>C1 Medium</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C2 0</td>
<td>1</td>
<td>0.519843</td>
<td>0.405972</td>
<td>1.6396</td>
<td>0.2004</td>
</tr>
<tr>
<td>C2 1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X1</td>
<td>1</td>
<td>1.970290</td>
<td>0.534947</td>
<td>13.5656</td>
<td>0.0002</td>
</tr>
<tr>
<td>X2</td>
<td>1</td>
<td>0.752247</td>
<td>0.419589</td>
<td>3.2142</td>
<td>0.0730</td>
</tr>
<tr>
<td>X3</td>
<td>1</td>
<td>0.959931</td>
<td>0.600959</td>
<td>2.5515</td>
<td>0.1102</td>
</tr>
<tr>
<td>X4</td>
<td>1</td>
<td>-0.117182</td>
<td>0.059406</td>
<td>3.8910</td>
<td>0.0485</td>
</tr>
</tbody>
</table>

You can use the DATA step code from the CODE statement to compute direct adjusted survival probabilities by taking the average of individual predicted survival probabilities. The method of direct adjustment controls for possible confounders that result from an imbalance of subject characteristics between groups. This adjustment is especially useful in nonrandomized studies. In the following statements, the DATA step code is applied to the getStarted data set to obtain the predicted survival probabilities for individuals in that data set and the predicted probabilities at each time point are averaged by gender:

```r
data Scores;
  set mycas.getStarted;
  %inc 'ScoreCode.txt';
run;
```

```r
proc print data=Scores (obs=1);
  var Time_:;
run;
```

```r
proc means data=Scores mean;
  class C3;
  var S_Time_:;
run;
```

Output 16.2.2 displays the time points at which the survival probabilities were predicted. These time points, Time_1=7, Time_2=19, and Time_3=54, are the quartiles of the failure times based on the Kaplan-Meier curve. Output 16.2.3 displays the direct adjusted survival probabilities at these quartiles. The predicted survival probability at Time_1 for females is slightly lower than that for males, but at Time_3 the direction is reversed. Also revealed is that the getStarted data set has 76 males but only 24 females.

**Output 16.2.2** Quartiles of the Observed Failure Times

<table>
<thead>
<tr>
<th>Obs</th>
<th>Time_1</th>
<th>Time_2</th>
<th>Time_3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7</td>
<td>19</td>
<td>54</td>
</tr>
</tbody>
</table>
Output 16.2.3 Direct Adjusted Survival Probabilities

The MEANS Procedure

<table>
<thead>
<tr>
<th>N</th>
<th>C3</th>
<th>Obs</th>
<th>Variable</th>
<th>Label</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>24</td>
<td>S_Time_1</td>
<td>Survival Probability at Time_1</td>
<td>0.7016195</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>S_Time_2</td>
<td>Survival Probability at Time_2</td>
<td>0.3693234</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>S_Time_3</td>
<td>Survival Probability at Time_3</td>
<td>0.2765601</td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>76</td>
<td>S_Time_1</td>
<td>Survival Probability at Time_1</td>
<td>0.7485799</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>S_Time_2</td>
<td>Survival Probability at Time_2</td>
<td>0.4568778</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>S_Time_3</td>
<td>Survival Probability at Time_3</td>
<td>0.2354066</td>
<td></td>
</tr>
</tbody>
</table>

References


Chapter 17

The PLSMOD Procedure

Contents

Overview: PLSMOD Procedure ............................................................... 788
    PROC PLSMOD Features ............................................................... 788
    PROC PLSMOD Compared with Other SAS Procedures ....................... 789
        PROC PLSMOD Compared with the HPPLS Procedure ..................... 789
        PROC PLSMOD Compared with the PLS Procedure ....................... 790
    Using CAS Sessions and CAS Engine Librefs ................................ 791
Getting Started: PLSMOD Procedure ..................................................... 792
    Fitting a PLS Model .................................................................... 793
    Selecting the Number of Factors by Test Set Validation ................. 795
    Predicting New Observations ....................................................... 799
Syntax: PLSMOD Procedure ................................................................. 800
    PROC PLSMOD Statement ............................................................. 800
    BY Statement ............................................................................. 803
    CLASS Statement ....................................................................... 803
    DISPLAY Statement .................................................................... 804
    DISPLAYOUT Statement ............................................................... 805
    EFFECT Statement ...................................................................... 805
    MODEL Statement ....................................................................... 807
    OUTPUT Statement ..................................................................... 807
    PARTITION Statement .................................................................. 810
Details: PLSMOD Procedure ................................................................. 811
    Regression Methods ..................................................................... 811
        Partial Least Squares ................................................................ 811
        SIMPLS .................................................................................. 812
        Principal Component Regression ............................................. 812
        Reduced Rank Regression ....................................................... 812
        Relationships between Methods .............................................. 813
    Test Set Validation ...................................................................... 815
    Centering and Scaling .................................................................. 817
    Missing Values ............................................................................ 817
    Displayed Output ......................................................................... 818
        Model Information ................................................................... 818
        Dimensions ............................................................................ 818
        Number of Observations ......................................................... 818
        Class Level Information ........................................................... 818
        Centering and Scaling Information ........................................... 818
Overview: PLSMOD Procedure

The PLSMOD procedure fits reduced-rank linear models in SAS Viya by using any one of a number of linear predictive methods, including partial least squares (PLS).

Ordinary least squares regression, as implemented in SAS/STAT procedures such as the GLM and REG procedures, has the single goal of minimizing sample response prediction error, and it seeks linear functions of the predictors that explain as much variation in each response as possible. The PLSMOD procedure implements techniques that have the additional goal of accounting for variation in the predictors, under the assumption that directions in the predictor space that are well sampled should provide better prediction for new observations when the predictors are highly correlated. All the techniques that the PLSMOD procedure implements work by extracting successive linear combinations of the predictors, called factors (also called components, latent vectors, or latent variables), that optimally address one or both of these two goals: explaining response variation and explaining predictor variation. In particular, the method of partial least squares balances the two objectives by seeking factors that explain both response and predictor variation.

The name “partial least squares” also applies to a more general statistical method that is not implemented in this procedure. The partial least squares method was originally developed in the 1960s by the econometrician Herman Wold (1966) for modeling “paths” of causal relation between any number of “blocks” of variables. However, the PLSMOD procedure fits only predictive partial least squares models that have one “block” of predictors and one “block” of responses. If you are interested in fitting more general path models, consider using the CALIS procedure in SAS/STAT.

PROC PLSMOD Features

The main features of the PLSMOD procedure are as follows:

- provides model-building syntax with classification variables, continuous variables, interactions, and nestings
- provides effect-construction syntax for polynomial and spline effects
- supports partitioning of data into training and testing roles
・ provides test set validation to choose the number of extracted factors, where the model is fit to only part of the available data (the training set) and the fit is evaluated over the other part of the data (the test set)

・ produces an output data table that contains predicted values and other observationwise statistics

The PLSMOD procedure implements the following methods:

・ principal component regression, which extracts factors to explain as much predictor sample variation as possible

・ reduced rank regression, which extracts factors to explain as much response variation as possible. This technique, also known as (maximum) redundancy analysis, differs from multivariate linear regression only when there are multiple responses.

・ partial least squares regression, which balances the two objectives of explaining response variation and explaining predictor variation. Two different formulations for partial least squares are available: the original predictive method of Wold (1966) and the straightforward implementation of a statistically inspired modification of the partial least squares (SIMPLS) method of De Jong (1993).

Because the PLSMOD procedure runs on SAS Cloud Analytic Services (CAS), it also does the following:

・ enables you to run on a cluster of machines that distribute the data and the computations

・ enables you to run in single-machine mode

・ exploits all the available cores and concurrent threads. For information about how PROC PLSMOD uses threads, see the section “Multithreading” on page 81 in Chapter 3, “Shared Concepts.”

**PROC PLSMOD Compared with Other SAS Procedures**

The PLSMOD procedure provides functionality to fit reduced-rank linear models that is comparable to that of the HPPLS and PLS procedures in SAS/STAT software.

**PROC PLSMOD Compared with the HPPLS Procedure**

The functionality of the PLSMOD procedure closely resembles that of the HPPLS procedure, which is a high-performance procedure. The PLSMOD procedure is the next generation of the HPPLS procedure, and it was developed specifically for SAS Viya. Both procedures are designed to run on a cluster of machines that distribute the data and the computations.

The PLSMOD procedure and the HPPLS procedure have the similarities and differences shown in Table 17.1.
### Table 17.1 Comparison of PROC PLSMOD and PROC HPPLS

<table>
<thead>
<tr>
<th>Feature</th>
<th>PROC PLSMOD</th>
<th>PROC HPPLS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supports general factor extraction methods</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Supports <code>EFFECT</code> statement</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Supports <code>DISPLAY</code> and <code>DISPLAYOUT</code> statements</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Supports <code>BY</code> statement</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Supports <code>ID</code> statement</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Supports <code>MISSING=</code>, <code>VARSCALE</code>, and <code>PLOTS</code> options</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Supports cross validation methods</td>
<td>Only test set validation by using the <code>PARTITION</code> statement</td>
<td>Only test set validation by using the <code>PARTITION</code> statement</td>
</tr>
<tr>
<td>Includes input variables in output data table</td>
<td>Not by default, in order to avoid data duplication for large data tables. To include input variables, specify them in the <code>COPYVARS=</code> option in the <code>OUTPUT</code> statement.</td>
<td>Not by default, in order to avoid data duplication for large data sets. To include input variables, specify them in the <code>ID</code> statement.</td>
</tr>
<tr>
<td>Supports parameterization of classification variables</td>
<td>Full set of parameterizations (supports all values of the <code>PARAM=</code> option in the <code>CLASS</code> statement)</td>
<td>Supports only <code>PARAM=GLM</code> or <code>PARAM=REFERENCE</code></td>
</tr>
<tr>
<td>Threading</td>
<td>Specifically designed for CAS; executes on multiple threads</td>
<td>Primarily designed for a distributed environment; executes on multiple threads</td>
</tr>
</tbody>
</table>

### PROC PLSMOD Compared with the PLS Procedure

The PLSMOD procedure and the PLS procedure have the similarities and differences shown in Table 17.2.

<table>
<thead>
<tr>
<th>Feature</th>
<th>PROC PLSMOD</th>
<th>PROC PLS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supports general factor extraction methods</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Supports <code>EFFECT</code> statement</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Supports <code>DISPLAY</code> and <code>DISPLAYOUT</code> statements</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Supports <code>BY</code> statement</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Supports <code>ID</code> statement</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Supports <code>MISSING=</code>, <code>VARSCALE</code>, and <code>PLOTS</code> options</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Supports cross validation methods</td>
<td>Only test set validation by using the <code>PARTITION</code> statement</td>
<td>Various methods</td>
</tr>
</tbody>
</table>

---

**Table 17.2** Comparison of PROC PLSMOD and PROC PLS
Table 17.2  continued

<table>
<thead>
<tr>
<th>Feature</th>
<th>PROC PLSMOD</th>
<th>PROC PLS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Syntax of OUTPUT statement</td>
<td>No prefix needed; default prefix used if none is provided. If you do not</td>
<td>You must provide keyword and prefix</td>
</tr>
<tr>
<td></td>
<td>specify any keywords, the output data table includes predicted values for</td>
<td></td>
</tr>
<tr>
<td></td>
<td>response variables.</td>
<td></td>
</tr>
<tr>
<td>Includes input variables in output data</td>
<td>Not by default, in order to avoid data duplication for large data tables.</td>
<td>Yes</td>
</tr>
<tr>
<td>table</td>
<td>To include input variables, specify them in the COPYVARS= option in the</td>
<td></td>
</tr>
<tr>
<td></td>
<td>OUTPUT statement.</td>
<td></td>
</tr>
<tr>
<td>Supports parameterization of classification</td>
<td>Full set of parameterizations (supports all values of the PARAM= option in</td>
<td>Only GLM parameterization</td>
</tr>
<tr>
<td>variables</td>
<td>the CLASS statement)</td>
<td></td>
</tr>
<tr>
<td>Threading</td>
<td>Specifically designed for CAS; executes on multiple threads</td>
<td>Executes on a single thread</td>
</tr>
</tbody>
</table>

Using CAS Sessions and CAS Engine Librefs

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:

```
proc options option=(CASHOST CASPORT);
run;
```

In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:

```
cas mysess;
libname mycas cas sessref=mysess;
```

The CAS statement creates the CAS session named mysess, and the LIBNAME statement creates the mycas CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the CASHOST and CASPORT of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the mysess session, you can terminate it by using the TERMINATE option in the CAS statement as follows:
cas mysess terminate;

For more information about the CAS and LIBNAME statements, see the section “Introduction to Shared Concepts” on page 10 in Chapter 3, “Shared Concepts.”

Getting Started: PLSMOD Procedure

The example in this section illustrates basic features of the PLSMOD procedure. The data are reported in Umetrics (1995); the original source is Lindberg, Persson, and Wold (1983). Suppose you are researching pollution in the Baltic Sea and you want to use the fluorescence spectra of seawater samples to determine the amounts of three compounds present: lignin sulfonate ($ls$: pulp industry pollution), humic acids ($ha$: natural forest products), and optical whitener from detergent ($dt$). Spectrometric calibration is a type of problem in which partial least squares can be very effective. The predictors are the spectra emission intensities at different frequencies in a sample spectrum, and the responses are the amounts of various chemicals in the sample.

For the purpose of calibrating the model, samples that have known compositions are used. The calibration data consist of 16 samples of known concentrations of $ls$, $ha$, and $dt$, with spectra based on 27 frequencies (or, equivalently, wavelengths). In order to demonstrate the use of test set validation, the data contain a variable $Role$, which is used to assign observations to the training and testing roles. In this case, the training role has nine samples and the testing role has seven samples.

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

The following DATA step creates the `mycas.Sample` data table, which provides the calibration data, in your CAS session. These statements assume that your CAS engine libref is named `mycas`, but you can substitute any appropriately defined CAS engine libref.

```sas
data mycas.Sample;
  input obsnam $ v1-v27 ls ha dt Role $5. @@;
  datalines;
EM1 2766 2610 3306 3630 3600 3438 3213 3051 2907 2844 2796
  2787 2760 2754 2670 2520 2310 2100 1917 1755 1602 1467
  1353 1260 1167 1101 1017 3.0110 0.0000 0.00 TRAIN
EM2 1492 1419 1369 1158 958 887 905 929 920 887 800
  710 617 535 451 368 296 241 190 157 128 106
  89 70 65 56 50 0.0000 0.4005 0.00 TEST
EM3 2450 2379 2400 2055 1689 1355 1109 908 750 673 644
  640 630 618 571 512 440 368 305 247 196 156
  120 98 80 61 50 0.0000 0.0000 90.63 TRAIN
EM4 2751 2883 3492 3570 3282 2937 2634 2370 2187 2070 2007
  1974 1950 1890 1824 1680 1527 1350 1206 1080 984 888
  810 732 669 630 582 1.4820 0.1580 40.00 TEST
EM5 2652 2691 3225 3285 3033 2784 2520 2340 2235 2148 2094
  2049 2007 1917 1800 1650 1464 1299 1140 1020 909 810
  726 657 594 549 507 1.1160 0.4104 30.45 TEST
```

```
To isolate a few underlying spectral factors that provide a good predictive model, you can fit a PLS model to the 16 samples by using the following SAS statements:

```
proc plsmmod data=mycas.sample;
   model ls ha dt = v1-v27;
run;
```

By default, the PLSMOD procedure extracts at most 15 factors. The default output from this analysis is presented in Figure 17.1 and Figure 17.2.

**Figure 17.1** displays the “Model Information,” “Dimensions,” and “Number of Observations” tables.

The “Model Information” table identifies the data source and shows that the factor extraction method is partial least squares regression (which is the default) and that the nonlinear iterative partial least squares (NIPALS) algorithm (which is also the default) is used to compute extracted PLS factors.
The “Dimensions” table shows the number of response variables, the number of effects, the number of predictor parameters, and the number of factors to extract.

The “Number of Observations” table shows that all 16 of the sample observations in the input data are used in the analysis; all the samples are used because they all contain complete data.

**Figure 17.1** Model Information, Dimensions, and Number of Observations

<table>
<thead>
<tr>
<th>Model Information</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data Source</strong></td>
</tr>
<tr>
<td><strong>Factor Extraction Method</strong></td>
</tr>
<tr>
<td><strong>PLS Algorithm</strong></td>
</tr>
<tr>
<td><strong>Validation Method</strong></td>
</tr>
</tbody>
</table>

**Dimensions**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Number of Response Variables</strong></td>
<td>3</td>
</tr>
<tr>
<td><strong>Number of Effects</strong></td>
<td>27</td>
</tr>
<tr>
<td><strong>Number of Predictor Parameters</strong></td>
<td>27</td>
</tr>
<tr>
<td><strong>Number of Factors</strong></td>
<td>15</td>
</tr>
</tbody>
</table>

| Number of Observations Read | 16     |
| Number of Observations Used  | 16     |

Figure 17.2 lists the amount of variation, both individual and cumulative, that is accounted for by each of the 15 factors. All the variation in both the predictors and the responses is accounted for by only 15 factors because there are only 16 sample observations. More important, almost all the variation is accounted for by even fewer factors—one or two for the predictors and three to eight for the responses.
### Figure 17.2 PLS Variation Summary

<table>
<thead>
<tr>
<th>Number of Extracted Factors</th>
<th>Model Effects</th>
<th>Response Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Current</td>
<td>Total</td>
</tr>
<tr>
<td>1</td>
<td>97.46068</td>
<td>97.46068</td>
</tr>
<tr>
<td>2</td>
<td>2.18296</td>
<td>99.64365</td>
</tr>
<tr>
<td>3</td>
<td>0.17806</td>
<td>99.82170</td>
</tr>
<tr>
<td>4</td>
<td>0.11973</td>
<td>99.94143</td>
</tr>
<tr>
<td>5</td>
<td>0.04146</td>
<td>99.98289</td>
</tr>
<tr>
<td>6</td>
<td>0.01058</td>
<td>99.99347</td>
</tr>
<tr>
<td>7</td>
<td>0.00168</td>
<td>99.99515</td>
</tr>
<tr>
<td>8</td>
<td>0.00097586</td>
<td>99.99613</td>
</tr>
<tr>
<td>9</td>
<td>0.00142</td>
<td>99.99755</td>
</tr>
<tr>
<td>10</td>
<td>0.00097037</td>
<td>99.99852</td>
</tr>
<tr>
<td>11</td>
<td>0.00032725</td>
<td>99.99884</td>
</tr>
<tr>
<td>12</td>
<td>0.00029338</td>
<td>99.99914</td>
</tr>
<tr>
<td>13</td>
<td>0.00024792</td>
<td>99.99939</td>
</tr>
<tr>
<td>14</td>
<td>0.00042742</td>
<td>99.99981</td>
</tr>
<tr>
<td>15</td>
<td>0.00018639</td>
<td>100.00000</td>
</tr>
</tbody>
</table>

## Selecting the Number of Factors by Test Set Validation

A PLS model is not complete until you choose the number of factors. You can choose the number of factors by using test set validation, in which the data table is divided into two groups called the training data and test data. You fit the model to the training data, and then you check the capability of the model to predict responses for the test data. The predicted residual sum of squares (PRESS) statistic is based on the residuals that are generated by this process.

To select the number of extracted factors by test set validation, you use the `PARTITION` statement to specify how to logically divide observations in the input data table into two subsets for model training and testing. For example, you can designate a variable in the input data table and a set of formatted values of that variable to determine the role of each observation, as in the following SAS statements:

```sas
proc plsmod data=mycas.sample;
    model ls ha dt = v1-v27;
    partition roleVar = Role(train='TRAIN' test='TEST');
run;
```

The resulting output is shown in Figure 17.3 through Figure 17.5.
Figure 17.3  Model Information, Dimensions, and Number of Observations with Test Set Validation

The PLSMOD Procedure

<table>
<thead>
<tr>
<th>Model Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source</td>
</tr>
<tr>
<td>Factor Extraction Method</td>
</tr>
<tr>
<td>PLS Algorithm</td>
</tr>
<tr>
<td>Validation Method</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dimensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Response Variables</td>
</tr>
<tr>
<td>Number of Effects</td>
</tr>
<tr>
<td>Number of Predictor Parameters</td>
</tr>
<tr>
<td>Maximum Number of Factors</td>
</tr>
</tbody>
</table>

| Number of Observations Read  | 16              |
| Number of Observations Used  | 16              |
| Number of Observations Used for Training | 9       |
| Number of Observations Used for Testing  | 7               |

Figure 17.4  Test-Set-Validated PRESS Statistics for Number of Factors

The PLSMOD Procedure

<table>
<thead>
<tr>
<th>Test Set Validation for the Number of Extracted Factors</th>
<th>Root Mean PRESS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Extracted Factors</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1.426362</td>
</tr>
<tr>
<td>1</td>
<td>1.276694</td>
</tr>
<tr>
<td>2</td>
<td>1.181752</td>
</tr>
<tr>
<td>3</td>
<td>0.656999</td>
</tr>
<tr>
<td>4</td>
<td>0.43457</td>
</tr>
<tr>
<td>5</td>
<td>0.420916</td>
</tr>
<tr>
<td>6</td>
<td>0.585031</td>
</tr>
<tr>
<td>7</td>
<td>0.576586</td>
</tr>
<tr>
<td>8</td>
<td>0.563935</td>
</tr>
<tr>
<td>9</td>
<td>0.563935</td>
</tr>
</tbody>
</table>

Minimum Root Mean PRESS 0.420916
Minimizing Number of Factors 5
In Figure 17.3, the “Model Information” table indicates that test set validation is used. The “Dimensions” table shows that the maximum number of factors to extract is nine. The “Number of Observations” table shows that nine sample observations are assigned for training roles and seven are assigned for testing roles.

Figure 17.4 provides details about the results from test set validation. These results show that the absolute minimum PRESS is achieved with five extracted factors. Notice, however, that this is not much smaller than the PRESS for three factors. By using the CVTEST option, you can perform a statistical model comparison that is suggested by Van der Voet (1994) to test whether this difference is significant, as shown in the following SAS statements:

```
proc plsmod data=mycas.sample cvtest(pval=0.15 seed=12345);
  model ls ha dt = v1-v27;
  partition roleVar = Role(train='TRAIN' test='TEST');
run;
```

The model comparison test is based on a rerandomization of the data. By default, the seed for this randomization is based on the system clock, but it is specified here. The resulting output is presented in Figure 17.6 through Figure 17.8.

---

**Selecting the Number of Factors by Test Set Validation**

**Figure 17.5** PLS Variation Summary for Test-Set-Validated Model

<table>
<thead>
<tr>
<th>Number of Extracted Factors</th>
<th>Current</th>
<th>Total</th>
<th>Current</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>95.92495</td>
<td>95.92495</td>
<td>37.27071</td>
<td>37.27071</td>
</tr>
<tr>
<td>2</td>
<td>3.86407</td>
<td>99.78903</td>
<td>32.38167</td>
<td>69.65238</td>
</tr>
<tr>
<td>3</td>
<td>0.10170</td>
<td>99.89073</td>
<td>20.76882</td>
<td>90.42120</td>
</tr>
<tr>
<td>4</td>
<td>0.08979</td>
<td>99.98052</td>
<td>4.66666</td>
<td>95.08787</td>
</tr>
<tr>
<td>5</td>
<td>0.01142</td>
<td>99.99194</td>
<td>3.88184</td>
<td>98.96971</td>
</tr>
</tbody>
</table>

---

**Figure 17.6** Model Information with Model Comparison Test

The PLSMOD Procedure

<table>
<thead>
<tr>
<th>Model Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source</td>
</tr>
<tr>
<td>Factor Extraction Method</td>
</tr>
<tr>
<td>PLS Algorithm</td>
</tr>
<tr>
<td>Validation Method</td>
</tr>
<tr>
<td>Validation Testing Criterion</td>
</tr>
<tr>
<td>Number of Random Permutations</td>
</tr>
<tr>
<td>Random Number Seed for Permutation</td>
</tr>
</tbody>
</table>
The "Model Information" table in Figure 17.6 displays information about the options that are used in the model comparison test. In Figure 17.7, the $p$-value in comparing the test-set-validated residuals from models that have five and three factors indicates that the difference between the two models is insignificant; therefore, the model with fewer factors is preferred. The variation summary in Figure 17.8 shows that more than 99% of the predictor variation and more than 90% of the response variation are accounted for by the three factors.
Predicting New Observations

Now that you have chosen a three-factor PLS model for predicting pollutant concentrations that are based on sample spectra, suppose that you have two new samples. The following SAS statements create a data table that contains the spectra for the new samples:

```sas
data newobs;
  input obsnam $ v1-v27 @@;
datalines;
EM17 3933 4518 5637 6006 5721 5187 4641 4149 3789 3579 3447 3381 3327 3234 3078 2832 2571 2274 2040 1818 1629 1470 1350 1245 1134 1050 987
EM25 2904 2997 3255 3150 2922 2778 2700 2646 2571 2487 2370 2250 2127 2052 1713 1419 1200 984 795 648 525 426 351 291 240 204 162
;
```

You can apply the PLS model to these samples to estimate pollutant concentration by appending the new samples to the original 16 and specifying that the predicted values for all 18 samples be output to a data table, as shown in the following statements:

```sas
data mycas.all;
  set mycas.sample newobs;
run;

proc plsmod data=mycas.all nfac=2;
  model ls ha dt = v1-v27;
  partition roleVar = Role(train='TRAIN' test='TEST');
  output out=mycas.result pred=p copyvars=(obsnam);
run;

proc print data=mycas.result;
  where (obsnam in ('EM17','EM25'));
  var obsnam p_ls p_ha p_dt;
run;
```

The COPYVARS= option in the OUTPUT statement requests that the variable obsnam from the input data table be copied to the output data table. The new observations are not used in calculating the PLS model because they have no response values. Their predicted concentrations are shown in Figure 17.9.

![Figure 17.9](image-url)
Syntax: PLSMOD Procedure

The following statements are available in the PLSMOD procedure:

```
PROC PLSMOD < options > ;
   BY variables ;
   CLASS variable < (options) > . . . < variable < (options) > > < / global-options > ;
   DISPLAY < table-list > < / options > ;
   DISPLAYOUT table-spec-list < / options > ;
   EFFECT name = effect-type (variables < / options > ) ;
   MODEL response-variables = predictor-effects < / options > ;
   OUTPUT OUT = CAS-libref.data-table
      < COPYVARS = (variables) >
      < keyword = prefix > . . . < keyword = prefix > ;
   PARTITION partition-options ;
```

The PROC PLSMOD statement and a single MODEL statement are required. All other statements are optional. The CLASS statement can appear multiple times. If a CLASS statement is specified, it must precede the MODEL statement. The following sections describe the PROC PLSMOD statement and then describe the other statements in alphabetical order.

PROC PLSMOD Statement

```
PROC PLSMOD < options > ;
```

The PROC PLSMOD statement invokes the PLSMOD procedure. Table 17.3 summarizes the options available in the PROC PLSMOD statement.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Options</td>
<td></td>
</tr>
<tr>
<td>DATA=</td>
<td>Specifies the CAS input data table</td>
</tr>
<tr>
<td>Model Fitting Options</td>
<td>Requests that van der Voet’s (1994) randomization-based model comparison test be performed</td>
</tr>
<tr>
<td>CVTEST</td>
<td></td>
</tr>
<tr>
<td>METHOD=</td>
<td>Specifies the general factor extraction method to be used</td>
</tr>
<tr>
<td>NFAC=</td>
<td>Specifies the number of factors to extract</td>
</tr>
<tr>
<td>NOCENTER</td>
<td>Suppresses centering of the responses and predictors before fitting</td>
</tr>
<tr>
<td>NOCVSTDIZE</td>
<td>Suppresses re-centering and rescaling of the responses and predictors when cross validating</td>
</tr>
<tr>
<td>NOSCALE</td>
<td>Suppresses scaling of the responses and predictors before fitting</td>
</tr>
<tr>
<td>Output Options</td>
<td></td>
</tr>
<tr>
<td>CENSCALE</td>
<td>Displays the centering and scaling information</td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays the details of the fitted model</td>
</tr>
<tr>
<td>Option</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>NOCLPRINT</td>
<td>Limits or suppresses the display of class levels</td>
</tr>
<tr>
<td>VARSS</td>
<td>Displays the amount of variation accounted for in each response and predictor</td>
</tr>
</tbody>
</table>

The following list provides details about these options.

**CENSCALE**

lists the centering and scaling information for each response and predictor.

**CVTEST < (cvtest-options) >**

requests that van der Voet’s (1994) randomization-based model comparison test be performed to test models that have different numbers of extracted factors against the model that minimizes the predicted residual sum of squares. For more information, see the section “Test Set Validation” on page 815. You can also specify the following cvtest-options in parentheses:

- **NSAMP=number**
  - specifies the number of randomizations to perform. By default, NSAMP=1000.

- **PVAL=number**
  - specifies the cutoff probability for declaring an insignificant difference. By default, PVAL=0.10.

- **SEED=number**
  - specifies the seed value for the random number stream. If you do not specify this option or if number is less than or equal to 0, the seed is generated by reading the time of day from the computer’s clock.

  Analyses that use the same (nonzero) seed are not completely reproducible if they are executed on a different number of compute nodes, because the random number streams in separate compute nodes are independent.

- **STAT=PRESS | T2**
  - specifies the test statistic for the model comparison. You can specify the following values:

    - **PRESS**
      - uses the predicted residual sum of squares.

    - **T2**
      - uses Hotelling’s $T^2$ statistic.

  By default, STAT=T2.

**DATA=CAS-libref.data-table**

names the input data table for PROC PLSMOD to use. The default is the most recently created data table. **CAS-libref.data-table** is a two-level name, where

- **CAS-libref**
  - refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to the data, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about **CAS-libref**, see the section “Using CAS Sessions and CAS Engine Librefs” on page 791.
data-table specifies the name of the input data table.

DETAILS
lists the details of the fitted model for each successive factor. The listed details are different for different extraction methods. For more information, see the section “Displayed Output” on page 818.

METHOD=PLS<(PLS-options)> | SIMPLS | PCR | RRR
specifies the general factor extraction method to be used. You can specify the following values:

PCR
uses principal component regression.

PLS<(PLS-options)>
uses partial least squares. You can also specify the following optional PLS-options in parentheses:

ALGORITHM=NIPALS | SVD | EIG
names the specific algorithm to use to compute extracted PLS factors. You can specify the following values:

NIPALS requests the usual iterative NIPALS algorithm.

SVD bases the extraction on the singular value decomposition of $X'Y$. This algorithm is the most accurate but least efficient approach.

EIG bases the extraction on the eigenvalue decomposition of $Y'XX'Y$.

By default, ALGORITHM=NIPALS.

EPSILON=number
specifies the convergence criterion for the NIPALS algorithm. By default, EPSILON=$10^{-12}$.

MAXITER=number
specifies the maximum number of iterations for the NIPALS algorithm. By default, MAXITER=200.

RRR
uses reduced rank regression.

SIMPLS
uses the straightforward implementation of a statistically inspired modification of the partial least squares (SIMPLS) method of De Jong (1993).

By default, METHOD=PLS(NIPALS).

NFAC=number
specifies the number of factors to extract. The default is $\min\{15, p, N\}$, where $p$ is the number of predictors (or the number of response variables when METHOD=RRR) and $N$ is the number of runs (observations). You probably do not need to extract this many factors for most applications. Extracting too many factors can lead to an overfitted model (one that matches the training data too well), sacrificing predictive ability. Thus, if you use the default, you should also either specify the PARTITION statement to select the appropriate number of factors for the final model or consider the analysis to be preliminary and examine the results to determine the appropriate number of factors for a subsequent analysis.
NOCENTER
suppresses centering of the responses and predictors before fitting. This option is useful if the analysis variables are already centered and scaled. For more information, see the section “Centering and Scaling” on page 817.

NOCLPRINT<=number>
suppresses the display of the “Class Level Information” table if you do not specify number. If you specify number, the values of the classification variables are displayed only for variables whose number of levels is less than number. Specifying a number helps to reduce the size of the “Class Level Information” table if some classification variables have a large number of levels.

NOCVSTDIZE
suppresses re-centering and rescaling of the responses and predictors before each model is fit in the cross validation. For more information, see the section “Centering and Scaling” on page 817.

NOSCALE
suppresses scaling of the responses and predictors before fitting. This option is useful if the analysis variables are already centered and scaled. For more information, see the section “Centering and Scaling” on page 817.

VARSS
lists, in addition to the average response and predictor sum of squares accounted for by each successive factor, the amount of variation accounted for in each response and predictor.

BY Statement
BY variables;

You can specify a BY statement in PROC PLSDM to obtain separate analyses of observations in groups that are defined by the values of the BY variables. If you specify more than one BY statement, only the last one specified is used. For more information, see the discussion of BY-group processing in SAS Language Reference: Concepts.

CLASS Statement
CLASS variable <(options)> . . . < variable <(options) > > < / global-options > ;

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. You can list the response variable for binary models in the CLASS statement, but this is not required. Table 17.4 summarizes the values that you can use for either an option or a global-option. The options are fully documented in the section “CLASS Statement” on page 12 in Chapter 3, “Shared Concepts.”

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESCENDING</td>
<td>Reverses the sort order</td>
</tr>
<tr>
<td>MISSING</td>
<td>Treats missing values as valid levels</td>
</tr>
</tbody>
</table>
Table 17.4  continued

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORDER=</td>
<td>Specifies the sort order for the levels</td>
</tr>
<tr>
<td>PARAM=</td>
<td>Specifies the parameterization of the variable</td>
</tr>
<tr>
<td>REF=</td>
<td>Specifies the reference level of the variable</td>
</tr>
<tr>
<td>SPLIT</td>
<td>Allows design columns for a variable to enter or leave the model independently</td>
</tr>
</tbody>
</table>

**DISPLAY Statement**

DISPLAY <table-list> < / options > ;

The DISPLAY statement enables you to specify a list of display tables to display or exclude. This statement is similar to the ODS SELECT, ODS EXCLUDE, and ODS TRACE statements. However, the DISPLAY statement can improve performance when a large number of tables could be generated (such as in BY-group processing). The procedure processes the DISPLAY statement on a CAS server and thus sends only a subset of ODS tables to the SAS client. Because ODS statements are processed on a SAS client, first all the generated display tables are sent to the client, and then the client creates a subset.

If you use both DISPLAY and ODS statements together, the DISPLAY statement takes precedence over the ODS statements. Note that the ODS EXCLUDE statement processes tables that are sent to the client after they have been filtered by the DISPLAY statement. In some cases, it might appear that the ODS EXCLUDE statement is taking precedence because it can further filter the tables. For more information about ODS, see *SAS Output Delivery System: Procedures Guide*.

You can specify the table-list as a list of table names, paths, partial pathnames, and regular expressions.

The table names that you can specify are listed in the section “ODS Table Names” on page 819. A path is a table name that is prefixed with dot-separated grouping information. For example, a SelectionSummary table that a procedure produces during a selection routine might have the path Bygroup1.Summary.SelectionSummary. A partial pathname does not include all groups; for example, SelectionSummary and Summary.SelectionSummary are partial pathnames for Bygroup1.Summary.SelectionSummary.

When you specify a table name or partial pathname, all display tables whose paths end in the specified name are selected for display or exclusion. For example, both SelectionSummary and Summary.SelectionSummary select Bygroup1.Summary.SelectionSummary.

A regular expression is enclosed in forward slashes (/). For example, specifying “/tions/” selects all pathnames that contain the substring “tions”; in particular, the Bygroup1.Summary.SelectionSummary table is selected. Specifying “!/tions/” selects all pathnames that do not contain the substring “tions”; in particular, the Bygroup1.Summary.SelectionSummary table is not selected.

You can specify the following options after a slash (/):

**CASESENSITIVE**

performs a case-sensitive comparison of table names in the table-list to display table names when tables are subsetted for display. To preserve case, you must enclose table names in the table-list in quotation marks.
EXCLUDE displays all display tables except those that you specify in the table-list.

EXCLUDEALL suppresses display of all tables. This option takes precedence over the other options.

TRACE displays the display table names, labels, and paths.

### DISPLAYOUT Statement

**DISPLAYOUT** table-spec-list </ options> ;

The DISPLAYOUT statement enables you to create CAS output tables from your displayed output. This statement is similar to the ODS OUTPUT statement. For more information about ODS, see *SAS Output Delivery System: Procedures Guide*.

The table-spec-list specifies a list of CAS output tables to create. Each entry in the list has either a key=value format or a key format:

- **key=value** specifies key as the ODS table name, path, or partial pathname, and specifies value as the CAS output table name.
- **key** specifies key as the ODS table name and also as the CAS output table name.

The ODS table names that you can specify are listed in the section “ODS Table Names” on page 819. You cannot specify the ODS table named OutputCasTables in the table-spec-list.

Table names and partial pathnames are discussed under the DISPLAY statement. The DISPLAYOUT statement does not support regular expressions.

You can specify the following options after a slash (/):

- **INCLUDEALL** creates output CAS tables for all display tables. The name of the created output CAS table is the same as the corresponding display table name. If you specify this option, the table-spec-list specification is ignored.

- **NOREPLACE** does not replace any existing CAS output table of the same name.

- **REPEATED** replicates all CAS output tables on all nodes.

### EFFECT Statement

**EFFECT** name=effect-type (variables </ options>) ;

The EFFECT statement enables you to construct special collections of columns for design matrices. These collections are referred to as *constructed effects* to distinguish them from the usual model effects that are
formed from continuous or classification variables, as discussed in the section “GLM Parameterization of Classification Variables and Effects” on page 54 in Chapter 3, “Shared Concepts.”

You can specify the following `effect-types`:

- **COLLECTION** specifies a collection effect that defines one or more variables as a single effect that has multiple degrees of freedom. The variables in a collection are considered as a unit for purposes of estimation and inference.

- **MULTIMEMBER | MM** specifies a multimember classification effect whose levels are determined by one or more variables that appear in a CLASS statement.

- **POLYNOMIAL | POLY** specifies a multivariate polynomial effect in the specified numeric variables.

- **SPLINE** specifies a regression spline effect whose columns are univariate spline expansions of one or more variables. A spline expansion replaces the original variable with an expanded or larger set of new variables.

Table 17.5 summarizes the `options` available in the EFFECT statement.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Collection Effects Options</strong></td>
<td></td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays the constituents of the collection effect</td>
</tr>
<tr>
<td><strong>Multimember Effects Options</strong></td>
<td></td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays the levels of the multimember effect</td>
</tr>
<tr>
<td>NOEFFECT</td>
<td>Specifies that observations whose levels are all missing for the multimember variables should have 0 values in the corresponding design matrix columns</td>
</tr>
<tr>
<td>STDIZE</td>
<td>Standardizes the design matrix entries so that each observation has a sum of 1</td>
</tr>
<tr>
<td>WEIGHT=</td>
<td>Specifies the weight variable for the contributions of each classification effect</td>
</tr>
<tr>
<td><strong>Polynomial Effects Options</strong></td>
<td></td>
</tr>
<tr>
<td>DEGREE=</td>
<td>Specifies the degree of the polynomial</td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays details of the specified polynomial</td>
</tr>
<tr>
<td>MDEGREE=</td>
<td>Specifies the maximum degree of any variable in a term of the polynomial</td>
</tr>
<tr>
<td>NOSEPARATE</td>
<td>Treats the polynomial as a single effect with multiple degrees of freedom</td>
</tr>
<tr>
<td>STANDARDIZE=</td>
<td>Specifies centering and scaling suboptions for the variables that define the polynomial</td>
</tr>
<tr>
<td><strong>Spline Effects Options</strong></td>
<td></td>
</tr>
<tr>
<td>BASIS=</td>
<td>Specifies the type of basis (B-spline basis or truncated power function basis) for the spline effect</td>
</tr>
<tr>
<td>DATABOUNDARY</td>
<td>Uses the extremes of the data as boundary knots for a B-spline basis</td>
</tr>
</tbody>
</table>
Table 17.5  continued

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEGREE=</td>
<td>Specifies the degree of the spline effect</td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays the knots and locations for each spline basis function</td>
</tr>
<tr>
<td>KNOTMAX=</td>
<td>Requests equally spaced right-side boundary knots starting at the variables’ maximum and ending at the KNOTMAX= value</td>
</tr>
<tr>
<td>KNOTMETHOD=</td>
<td>Specifies how to construct the knots for the spline effect</td>
</tr>
<tr>
<td>KNOTMIN=</td>
<td>Requests equally spaced left-side boundary knots starting at the KNOTMIN= value and ending at the variables’ minimum value</td>
</tr>
<tr>
<td>NATURALCUBIC</td>
<td>Specifies a natural cubic spline basis for the spline effect</td>
</tr>
<tr>
<td>SEPARATE</td>
<td>Treats the spline basis for each variable as a separate effect when multiple variables are specified</td>
</tr>
<tr>
<td>SPLIT</td>
<td>Treats each design matrix column as a separate effect for selection methods</td>
</tr>
</tbody>
</table>

For more information about the syntax of these effect-types and how columns of constructed effects are computed, see the section “EFFECT Statement” on page 21 in Chapter 3, “Shared Concepts.”

MODEL Statement

MODEL response-variables = predictor-effects < / options > ;

The MODEL statement names the responses and the predictors, which determine, respectively, the Y and X matrices of the model. You can simply list the names of the predictor variables as the model effects, but you can also specify other types of effects, including polynomial effects and interactions. For information about constructing the model effects, see the section “Specification and Parameterization of Model Effects” on page 51 in Chapter 3, “Shared Concepts.”

The MODEL statement is required. You can specify only one MODEL statement.

You can specify the following options after a slash (/).

INTERCEPT
overides the default, in which the responses and predictors are centered. When responses and predictors are centered, no intercept is required in the model.

SOLUTION
lists the coefficients of the final predictive model for the responses. The coefficients for predicting the centered and scaled responses that are based on the centered and scaled predictors are displayed, in addition to the coefficients for predicting the raw responses that are based on the raw predictors.

OUTPUT Statement

OUTPUT OUT=CAS-libref.data-table
  < COPYVARS=(variables) >
  < keyword =prefix > . . . < keyword =prefix > ;
The OUTPUT statement creates a data table that contains observationwise statistics, which are computed after fitting the model. If you do not specify any *keywords*, then only the predicted values for responses are included.

The variables in the input data table are *not* included in the output data table, in order to avoid data duplication for large data tables; however, variables that you specify in the COPYVARS= option are included.

You must specify the following option:

```
OUT=CAS-libref.data-table
```

names the output data table for PROC PLSMOD to use. You must specify this option before any other options. *CAS-libref.data-table* is a two-level name, where

- **CAS-libref** refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to where the data table is to be stored, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about *CAS-libref*, see the section “Using CAS Sessions and CAS Engine Librefs” on page 791.

- **data-table** specifies the name of the output data table.

You can also specify the following syntax elements:

```
COPYVAR=variable
COPYVARS=(variables)
```

copies one or more *variables* from the input data table to the output data table.

```
keyword <=prefix>
```

specifies a statistic to include in the output data table and optionally a *prefix* for naming the output variables. If you do not provide a *prefix*, the PLSMOD procedure assigns a default prefix based on the type of statistic requested. For example, for response variables y1 and y2, a specification of PREDICTED produces two predicted value variables, Pred_y1 and Pred_y2.

You can specify the following *keywords* to add statistics to the OUTPUT data table:

```
H
```

requests the approximate leverage. The default prefix is H.

```
PREDICTED
PRED
P
```

requests predicted values for each response. The default prefix is Pred.

```
PRESS
```

requests approximate predicted residuals for each response. The default prefix is PRESS.

```
ROLE
```

requests numeric values that indicate the role played by each observation in fitting the model. The default prefix is _ROLE_. Table 17.6 shows the interpretation of this variable for each observation.
Table 17.6  Role Interpretation

<table>
<thead>
<tr>
<th>Value</th>
<th>Observation Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Not used</td>
</tr>
<tr>
<td>1</td>
<td>Training</td>
</tr>
<tr>
<td>2</td>
<td>Testing</td>
</tr>
</tbody>
</table>

If you do not partition the input data by using a PARTITION statement, then the role variable value is 1 for observations that are used in fitting the model, and 0 for observations that have at least one missing or invalid value for the responses or predictors.

**STDX**
requests standardized (centered and scaled) predictor values for each predictor. The default prefix is StdX.

**STDXSSE**
requests the sum of squares of residuals for standardized predictors. The default prefix is StdXSSE.

**STDY**
requests standardized (centered and scaled) response values for each response. The default prefix is StdY.

**STDYSSE**
requests the sum of squares of residuals for standardized responses. The default prefix is StdYSSE.

**TSQUARE**
**T2**
requests a scaled sum of squares of score values. The default prefix is TSquare.

**XRESIDUAL**
**XRESID**
**XR**
requests residuals for each predictor. The default prefix is XResid.

**XSCORE**
requests extracted factors (X-scores, latent vectors, latent variables, and T) for each selected model factor. The default prefix is XScore.

**YRESIDUAL**
**YRESID**
**YR**
requests residuals for each response. The default prefix is YResid.
Chapter 17: The PLSMOD Procedure

**YSOURCE**

requests extracted responses (Y-scores and $U$) for each selected model factor. The default prefix is **YScore**.

The output variables that contain the requested statistic are named as follows, according to the *keyword* that you specify:

- The *keywords* **XRESIDUAL** and **STDX** define an output variable for each predictor, so the variables that correspond to each predictor are named by appending a number (which starts from 1) to the prefix. For each defined variable, a label is also generated automatically; the label contains the prefix of the variable and the name of the predictor. For example, if the model has three predictors, then a specification of **XRESIDUAL=XR** produces the variables XR1, XR2, and XR3.

- The *keywords* **PREDICTED**, **YRESIDUAL**, **STDY**, and **PRESS** define an output variable for each response, so the variables that correspond to each response are named by appending an underscore followed by the name of the response variable to the prefix. For example, if the model has response variables $y_1$ and $y_2$, then a specification of **PREDICTED=P** produces the variables $P_{y1}$ and $P_{y2}$.

- The *keywords* **XSCORE** and **YSCORE** define an output variable for each selected model factor, so the variables that correspond to each successive factor are named by appending the factor number to the prefix. For example, if the model has three selected factors, then a specification of **XSCORE=T** produces the variables T1, T2, and T3.

- The *keywords* **H**, **TSQUARE**, **STDXSSE**, **STDYSSE**, and **ROLE** each define a single output variable, so the variable name matches the prefix.

---

**PARTITION Statement**

**PARTITION** partition-options ;

The **PARTITION** statement specifies how to logically partition observations in the input data table into disjoint subsets for model training and testing. Either you can designate a variable in the input data table and a set of formatted values of that variable to determine the role of each observation, or you can specify proportions to use for random assignment of observations to each role.

You must specify exactly one of the following *partition-options*:

**FRACTION( < TEST=fraction > < SEED=number > )**

requests that specified proportions of the observations in the input data table be randomly assigned training and testing roles. You specify the proportions for testing by using the **TEST=** suboption; the specified fraction must be less than 1 and the remaining fraction of the observations are assigned to the training role. If you do not specify the **TEST=** suboption, all observations are assigned to the training role. The **SEED=** suboption specifies an integer that is used to start the pseudorandom number generator for random partitioning of data for training and testing. If you do not specify **SEED=** *number* or if *number* is less than or equal to 0, the seed is generated by reading the time of day from the computer’s clock.

Because *fraction* is a per-observation probability (which means that any particular observation has a probability of *fraction* of being assigned the testing role), using the **FRACTION** option can cause
different numbers of observations to be assigned training and testing roles. You can specify the SEED= suboption to create the same partition data tables for a particular number of compute nodes. However, changing the number of compute nodes changes the initial distribution of data, resulting in different partition data tables.

**ROLEVAR | ROLE=variable (<TEST='value'> <TRAIN='value'>)**

names the variable in the input data table whose values are used to assign roles to each observation. This variable cannot also appear as an analysis variable in other statements or options. The TEST= and TRAIN= suboptions specify the formatted values of this variable that are used to assign observation roles. If you specify only the TEST= suboption, then all observations whose role is not determined by the TEST= suboption are assigned to training.

For more information, see the section “Test Set Validation” on page 815. For an illustration, see Example 17.1.

---

**Details: PLDMOD Procedure**

**Regression Methods**

All the predictive methods that PROC PLDMOD implements work essentially by finding linear combinations of the predictors (factors) to use to predict the responses linearly. The methods differ only in how the factors are derived, as explained in the following sections.

**Partial Least Squares**

Partial least squares (PLS) works by extracting one factor at a time. Let $X = X_0$ be the centered and scaled matrix of predictors, and let $Y = Y_0$ be the centered and scaled matrix of response values. The PLS method starts with a linear combination $t = X_0 w$ of the predictors, where $t$ is called a **score vector** and $w$ is its associated **weight vector**. The PLS method predicts both $X_0$ and $Y_0$ by regression on $t$:

$$
\hat{X}_0 = tp', \text{ where } p' = (t't)^{-1}t'X_0 \\
\hat{Y}_0 = tc', \text{ where } c' = (t't)^{-1}t'Y_0
$$

The vectors $p$ and $c$ are called the X- and Y-**loadings**, respectively.

The specific linear combination $t = X_0 w$ is the combination that has maximum covariance $t'u$ with some response linear combination $u = Y_0 q$. Another characterization is that the X-weight, $w$, and the Y-weight, $q$, are proportional to the first left- and right-singular vectors, respectively, of the covariance matrix $X_0'Y_0$ or, equivalently, the first eigenvectors of $X_0'Y_0Y_0'X_0$ and $Y_0'X_0X_0'Y_0$, respectively.

This accounts for how the first PLS factor is extracted. The second factor is extracted in the same way by replacing $X_0$ and $Y_0$ with the X- and Y-residuals from the first factor:

$$
X_1 = X_0 - \hat{X}_0 \\
Y_1 = Y_0 - \hat{Y}_0
$$
These residuals are also called the deflated X and Y blocks. The process of extracting a score vector and deflating the data matrices is repeated for as many extracted factors as you want.

**SIMPLS**

The fact that each extracted PLS factor is defined in terms of different X-variables leads to difficulties in comparing different scores, weights, and so on. The SIMPLS method of De Jong (1993) overcomes these difficulties by computing each score $t_i = Xr_i$ in terms of the original (centered and scaled) predictors $X$. The SIMPLS X-weight vectors $r_i$ are similar to the eigenvectors of $SS' = X'YY'X$, but they satisfy a different orthogonality condition. The $r_1$ vector is just the first eigenvector $e_1$ (so that the first SIMPLS score is the same as the first PLS score). However, the second eigenvector maximizes

$$e_1' SS' e_2 \quad \text{subject to} \quad e_1' e_2 = 0$$

whereas the second SIMPLS weight $r_2$ maximizes

$$r_1' S S' r_2 \quad \text{subject to} \quad r_1' X X r_2 = t_1' t_2 = 0$$

The SIMPLS scores are identical to the PLS scores for one response but slightly different for more than one response; see De Jong (1993) for details. The X- and Y-loadings are defined as in PLS, but it is easy to compute the overall model coefficients $B$ because the scores are all defined in terms of $X$:

$$\hat{Y} = \sum_i t_i c_i'$$

$$= \sum_i Xr_i c_i'$$

$$= XB, \text{ where } B = RC'$$

**Principal Component Regression**

Like the SIMPLS method, principal component regression (PCR) defines all the scores in terms of the original (centered and scaled) predictors $X$. However, unlike both the PLS and SIMPLS methods, the PCR method chooses the X-weights and X-scores without regard to the response data. The X-scores are chosen to explain as much variation in $X$ as possible; equivalently, the X-weights for the PCR method are the eigenvectors of the predictor covariance matrix $X'X$. Again, the X- and Y-loadings are defined as in PLS; but, as in SIMPLS, it is easy to compute overall model coefficients for the original (centered and scaled) responses $Y$ in terms of the original predictors $X$.

**Reduced Rank Regression**

As discussed in the preceding sections, partial least squares depends on selecting the factors $t = Xw$ of the predictors and $u = Yq$ of the responses that have maximum covariance, whereas principal component regression effectively ignores $u$ and selects $t$ to have maximum variance, subject to orthogonality constraints. In contrast, reduced rank regression selects $u$ to account for as much variation in the predicted responses as possible, effectively ignoring the predictors for the purpose of factor extraction. In reduced rank regression, the Y-weights, $q_i$, are the eigenvectors of the covariance matrix $\hat{Y}_{LS}' \hat{Y}_{LS}$ of the responses that are predicted by ordinary least squares regression, and the X-scores are the projections of the Y-scores, $Yq_i$, onto the X space.
Relationships between Methods

When you develop a predictive model, it is important to consider not only the explanatory power of the model for current responses, but also how well the predictive functions are sampled, because the sampling affects how well the model can extrapolate to future observations. All the techniques that the PLSMOD procedure implements work by extracting successive factors (linear combinations of the predictors) that optimally address one or both of these two goals: explaining response variation and explaining predictor variation. In particular, principal component regression selects factors that explain as much predictor variation as possible, reduced rank regression selects factors that explain as much response variation as possible, and partial least squares balances the two objectives, seeking factors that explain both response and predictor variation.

To see the relationships between these methods, consider how each one extracts a single factor from the following artificial data table, artData, which consists of two predictors and one response:

```plaintext
data artData;
  input x1 x2 y;
datalines;
  3.37651 2.30716 0.75615
  0.74193 -0.88845 1.15285
  4.18747 2.17373 1.42392
  0.96097 0.57301 0.27433
 -1.11161 -0.75225 -0.25410
 -1.38029 -1.31343 -0.04728
  1.28153 -0.13751 1.00341
 -1.39242 -2.03615 0.45518
  0.63741 0.06183 0.40699
 -2.52533 -1.23726 -0.91080
  2.44277 3.61077 -0.82590
;

data mycas.artData;
  set artData;
run;

proc plsmod data=mycas.artData nfac=1 method=rrr;
  model y = x1 x2;
run;

proc plsmod data=mycas.artData nfac=1 method=pcr;
  model y = x1 x2;
run;

proc plsmod data=mycas.artData nfac=1 method=pls;
  model y = x1 x2;
run;
```

The amount of model and response variation that are explained by the first factor for each method is shown in Figure 17.10 through Figure 17.12.
Figure 17.10 Variation Explained by the First Reduced Rank Regression Factor

The PLSMOD Procedure

<table>
<thead>
<tr>
<th>Number of Extracted Factors</th>
<th>Current</th>
<th>Total</th>
<th>Current</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>15.06605</td>
<td>15.06605</td>
<td>100.00000</td>
<td>100.00000</td>
</tr>
</tbody>
</table>

Figure 17.11 Variation Explained by the First Principal Component Regression Factor

The PLSMOD Procedure

<table>
<thead>
<tr>
<th>Number of Extracted Factors</th>
<th>Current</th>
<th>Total</th>
<th>Current</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>92.99959</td>
<td>92.99959</td>
<td>9.37874</td>
<td>9.37874</td>
</tr>
</tbody>
</table>

Figure 17.12 Variation Explained by the First Partial Least Squares Regression Factor

The PLSMOD Procedure

<table>
<thead>
<tr>
<th>Number of Extracted Factors</th>
<th>Current</th>
<th>Total</th>
<th>Current</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>88.53567</td>
<td>88.53567</td>
<td>26.53038</td>
<td>26.53038</td>
</tr>
</tbody>
</table>

Notice that although the first reduced rank regression factor explains all of the response variation, it accounts for only about 15% of the predictor variation. In contrast, the first principal component regression factor accounts for most of the predictor variation (93%) but only 9% of the response variation. The first partial least squares factor accounts for only slightly less predictor variation than principal components but about three times as much response variation.

Figure 17.13 illustrates how partial least squares balances the goals of explaining response and predictor variation in this case.
The ellipse shows the general shape of the 11 observations in the predictor space, with the contours of increasing $y$ overlaid. Also shown are the directions of the first factor for each of the three methods. Notice that although the predictors vary most in the $x_1 = x_2$ direction, the response changes most in the orthogonal $x_1 = -x_2$ direction. This explains why the first principal component accounts for little variation in the response and why the first reduced rank regression factor accounts for little variation in the predictors. The direction of the first partial least squares factor represents a compromise between the other two directions.

**Test Set Validation**

None of the regression methods that the PLSMOD procedure implements fit the observed data any better than ordinary least squares (OLS) regression; in fact, all the methods approach OLS as more factors are extracted. Basing the model on more extracted factors improves the model fit to the observed data, but extracting too many factors can cause overfitting—that is, tailoring the model too much to the current data to the detriment of future predictions. So the crucial point is that when there are many predictors, OLS can overfit the observed data; biased regression methods that use fewer extracted factors can provide better predictability of future observations. However, as the preceding observations imply, the quality of the observed data fit cannot be
used to choose the number of factors to extract; the number of extracted factors must be chosen on the basis of how well the model fits observations that are not involved in the modeling procedure itself.

PROC PLSMOD implements the test set validation method of choosing the number of extracted factors. When you have sufficient data, you can subdivide your data into two parts: training data and test data. During the validation process, the model is fit on the training data, and the predicted residual sum of squares (PRESS) for models that have different numbers of extracted factors is found by using the test data. The number of factors chosen is usually the one that minimizes PRESS.

You use the PARTITION statement to logically subdivide the DATA= data table into separate roles. You can name the fractions of the data that you want to reserve as training data and test data. For example, the following statements randomly subdivide the inData data table, reserving 50% each for training and testing:

```plaintext
proc plsmod data=mycas.inData;
  partition fraction(test=0.5);
  ... run;
```

In some cases you might need to exercise more control over the partitioning of the input data table. You can do this by naming both a variable in the input data table and a formatted value of that variable for each role. For example, the following statements assign roles to the observations in the inData data table based on the value of the variable Group in that data table. Observations whose value of Group is 'group 1' are assigned for training, and those whose value is 'group 2' are assigned to testing. All other observations are ignored.

```plaintext
proc plsmod data=mycas.inData;
  partition roleVar=Group(train='group 1' test='group 2')
  ... run;
```

By default, the number of extracted factors is chosen to be the one that minimizes PRESS. However, models that have fewer factors often have PRESS statistics that are only marginally larger than the absolute minimum. To address this, Van der Voet (1994) proposed a statistical test for comparing the predicted residuals from different models; when you apply van der Voet’s test, the number of factors chosen is the fewest while still producing residuals that are insignificantly larger than the residuals of the model that has a minimum PRESS.

To see how van der Voet’s test works, let \( R_{i,jk} \) be the \( j \)th predicted residual for response \( k \) for the model that has \( i \) extracted factors. Then, the PRESS statistic is \( \sum_{jk} R_{i,jk}^2 \). Also, let \( i_{\text{min}} \) be the number of factors for which PRESS is minimized. The critical value for van der Voet’s test is based on the differences between squared predicted residuals:

\[
D_{i,jk} = R_{i,jk}^2 - R_{i_{\text{min}},jk}^2
\]

One alternative for the critical value is \( C_i = \sum_{jk} D_{i,jk} \), which is simply the difference between the PRESS statistics for \( i \) and \( i_{\text{min}} \) factors; alternatively, van der Voet suggests Hotelling’s \( T^2 \) statistic \( C_i = d_i' S_i^{-1} d_i \), where \( d_i \) is the sum of the vectors \( d_{i,j} = \{D_{i,j1}, \ldots, D_{i,jN_y}\}' \) and \( S_i \) is the sum of squares and crossproducts matrix,
Centering and Scaling

Theoretically, the significance level for van der Voet’s test is obtained by comparing $C_i$ with the distribution of values that result from randomly exchanging $R^2_{i,j,k}$ and $R^2_{i_{\text{min}},j,k}$. In practice, a Monte Carlo sample of such values is simulated and the significance level is approximated as the proportion of simulated critical values that are greater than $C_i$. If you apply van der Voet’s test by specifying the CVTEST option, then, by default, the number of extracted factors that are chosen is the smallest number of factors that have an approximate significance level that is greater than 0.1.

Centering and Scaling

By default, the predictors and the responses are centered and scaled to have mean 0 and standard deviation 1. Centering the predictors and the responses ensures that the criterion for choosing successive factors is based on how much variation they explain in either the predictors or the responses or in both. (For more information about how different methods explain variation, see the section “Regression Methods” on page 811.) Without centering, both the value of the mean variable and the variation around that mean are involved in selecting factors. Scaling serves to place all predictors and responses on an equal footing relative to their variation in the data. For example, if Time and Temp are two of the predictors, then scaling says that a change of $\text{std}(\text{Time})$ in Time is approximately equivalent to a change of $\text{std}(\text{Temp})$ in Temp.

Usually, both the predictors and responses should be centered and scaled. However, if their values already represent variation around a nominal or target value, then you can use the NOCENTER option in the PROC PLSMOD statement to suppress centering. Likewise, if the predictors or responses are already all on comparable scales, then you can use the NOSCALE option to suppress scaling.

If the predictors involve crossproduct terms, PROC PLSMOD does not standardize the variables before it standardizes the crossproduct. That is, if the $i$th values of two predictors are denoted $x^1_i$ and $x^2_i$, then the default standardized $i$th value of the crossproduct is

$$\frac{x^1_i x^2_i - \text{mean}_j(x^1_j x^2_j)}{\text{std}_j(x^1_j x^2_j)}$$

When test set validation is performed for the number of effects, some practitioners disagree as to whether the training data should be retransformed. By default, PROC PLSMOD does retransform the training data, but you can suppress this behavior by specifying the NOCVSTDIZE option in the PROC PLSMOD statement.

Missing Values

Observations that have any missing independent variables (including all classification variables) are excluded from the analysis, and no predictions are computed for such observations. However, if you specify the MISSING option in the CLASS statement, missing values are treated as valid values for the classification variable. Observations that have no missing independent variables but do have missing response variables are
also excluded from the analysis, but predictions are computed. If you use the \texttt{PARTITION} statement and specify the \texttt{ROLEVAR=} option, observations that contain missing values for the \texttt{ROLEVAR=} variable are excluded from the analysis, but predictions are computed for them.

\section*{Displayed Output}

The following sections describe the output that PROC PLSMOD produces. The output is organized into various tables, which are discussed in the order of their appearance.

\section*{Model Information}

The “Model Information” table displays basic information about the model, such as the input data table, the factor extraction method, the validation method, and the type of parameterization used for classification variables that are named in the \texttt{CLASS} statement. If you use the \texttt{PARTITION} statement, the table also displays the random number seed for partitioning, the validation testing criterion, the number of random permutations, and the random number seed for permutation, depending on whether you specify the \texttt{FRACTION} option in the \texttt{PARTITION} statement and the \texttt{CVTEST} option in the \texttt{PROC PLSMOD} statement.

\section*{Dimensions}

The “Dimensions” table displays information about the number of response variables, the number of effects, and the number of predictor parameters. It also displays the number of factors to extract.

\section*{Number of Observations}

The “Number of Observations” table displays the number of observations that are read from the input data table and the number of observations that are used in the analysis. If you use a \texttt{PARTITION} statement, the table also displays the number of observations that are used for each data role.

\section*{Class Level Information}

The “Class Level Information” table lists the levels of every variable that is specified in the \texttt{CLASS} statement. You should check this information to make sure that the data are correct. You can adjust the order of the \texttt{CLASS} variable levels by specifying the \texttt{ORDER=} option in the \texttt{CLASS} statement. You can suppress the “Class Level Information” table completely or partially by specifying the \texttt{NOCLPRINT=} option in the \texttt{PROC PLSMOD} statement.

If the classification variables use a nonsingular parameterization, the “Class Level Information” table also displays the reference value for each variable.

\section*{Centering and Scaling Information}

If you specify the \texttt{CENSCALE} option in the \texttt{PROC PLSMOD} statement, the PLSMOD procedure produces “Model Effect Centering and Scaling” and “Response Variable Centering and Scaling” tables, which display the centering and scaling information for each predictor and response.
Test Set Validation

If you use the PARTITION statement to perform a test set validation for choosing the number of extracted factors, the PLSMOD procedure produces a “Test Set Validation Residual Summary” table to display a residual summary of the validation for each number of factors. It also produces a “Test Set Validation Results” table to display information about the optimal number of factors.

Percentage Variation Accounted for by Extracted Factors

By default, the PLSMOD procedure produces the “Percentage Variation Accounted for by Extracted Factors” table to display the amount of predictor variation and response variation that are accounted for by each factor. If you specify the VARSS option in the PROC PLSMOD statement, the PLSMOD procedure also produces the “Model Effect Percentage Variation Accounted for by Extracted Factors” table and the “Response Variable Percentage Variation Accounted for by Extracted Factors” table to display the amount of variation that is accounted for in each predictor and response, in addition to the average predictor and response sum of squares that are accounted for by each successive factor.

Model Details

If you specify the DETAILS option in the PROC PLSMOD statement, the PLSMOD procedure produces tables to display details about the fitted model for each successive factor. These tables include the following:

- “Model Effect Loadings” table, which displays the predictor loadings
- “Model Effect Weights” table, which displays predictor weights
- “Response Variable Weights” table, which displays the response weights
- “Coded Regression Coefficients” tables, which display the coded regression coefficients, if you specify METHOD=SIMPLS, METHOD=PCR, or METHOD=RRR in the PROC PLSMOD statement.

Parameter Estimates

If you specify the SOLUTION option in the MODEL statement, the PLSMOD procedure produces a “Parameter Estimates” table to display the coefficients of the final predictive model for the responses. The coefficients for predicting the centered and scaled responses that are based on the centered and scaled predictors are displayed, in addition to the coefficients for predicting the raw responses based on the raw predictors.

Timing Information

The “Timing” table displays the elapsed time of each main task of the procedure.

ODS Table Names

Each table that the PLSMOD procedure creates has a name associated with it. You must use this name to refer to the table when you use the DISPLAY statement, the DISPLAYOUT statement, or the Output Delivery System (ODS) statements. These names are listed in Table 17.7.
Table 17.7  ODS Tables Produced by PROC PLSMOD

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Required Statement and Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>CVResults</td>
<td>Results of test set validation</td>
<td>PARTITION statement</td>
</tr>
<tr>
<td>CenScaleParms</td>
<td>Parameter estimates for centered and scaled data</td>
<td>SOLUTION option in MODEL</td>
</tr>
<tr>
<td>ClassInfo</td>
<td>Level information from the CLASS statement</td>
<td>CLASS statement</td>
</tr>
<tr>
<td>CodedCoef</td>
<td>Coded regression coefficients</td>
<td>DETAILS option in PROC PLSMOD</td>
</tr>
<tr>
<td>Dimensions</td>
<td>Model dimensions</td>
<td>Default output</td>
</tr>
<tr>
<td>ModelInfo</td>
<td>Model information</td>
<td>Default output</td>
</tr>
<tr>
<td>NObs</td>
<td>Number of observations read and used</td>
<td>Default output</td>
</tr>
<tr>
<td>OutputCasTables</td>
<td>A special ODS table that has information about all the CAS tables that are created during a CAS action execution</td>
<td>DISPLAYOUT statement or OUTPUT</td>
</tr>
<tr>
<td>ParameterEstimates</td>
<td>Parameter estimates for raw data</td>
<td>SOLUTION option in MODEL</td>
</tr>
<tr>
<td>PercentVariation</td>
<td>Predictor and response variation that are accounted for by each factor</td>
<td>Default output</td>
</tr>
<tr>
<td>ResidualSummary</td>
<td>Residual summary from test set validation</td>
<td>PARTITION statement</td>
</tr>
<tr>
<td>Timing</td>
<td>Absolute and relative times spent by tasks that are performed by the procedure</td>
<td>Default output</td>
</tr>
<tr>
<td>XEffectCenScale</td>
<td>Centering and scaling information for predictor effects</td>
<td>CENSCALE option in PROC PLSMOD</td>
</tr>
<tr>
<td>XLoadings</td>
<td>Loadings for predictor effects</td>
<td>DETAILS option in PROC PLSMOD</td>
</tr>
<tr>
<td>XPercentVariation</td>
<td>Variation that is accounted for by each factor for predictor effects</td>
<td>VARSS option in PROC PLSMOD</td>
</tr>
<tr>
<td>XWeights</td>
<td>Weights for predictor effects</td>
<td>DETAILS option in PROC PLSMOD</td>
</tr>
<tr>
<td>YPercentVariation</td>
<td>Variation that is accounted for by each factor for responses</td>
<td>VARSS option in PROC PLSMOD</td>
</tr>
<tr>
<td>YVariableCenScale</td>
<td>Centering and scaling information for responses</td>
<td>CENSCALE option in PROC PLSMOD</td>
</tr>
<tr>
<td>YWeights</td>
<td>Weights for responses</td>
<td>DETAILS option in PROC PLSMOD</td>
</tr>
</tbody>
</table>
Example 17.1: Choosing a PLS Model by Test Set Validation

This example demonstrates issues in spectrometric calibration. The data (Umetrics 1995) consist of spectrographic readings on 33 samples that contain known concentrations of two amino acids, tyrosine and tryptophan. The spectra are measured at 30 frequencies across the overall range of frequencies. For example, Output 17.1.1 shows the observed spectra for three samples: one with only tryptophan, one with only tyrosine, and one with a mixture of the two, all at a total concentration of $10^{-6}$.

Output 17.1.1 Spectra for Three Samples of Tyrosine and Tryptophan
Of the 33 samples, 18 are used as a training set and 15 as a test set. The data originally appear in McAvoy et al. (1989).

These data were created in a lab, where the concentrations are fixed in order to provide a wide range of applicability for the model. This example uses a linear function of the logarithms of the spectra to predict the logarithms of tyrosine and tryptophan concentration and the logarithm of the total concentration. Actually, because zeros are possible in both the responses and the predictors, slightly different transformations are used. The following statements create a data table named `ex1Data` for these data. The data table also contains a variable `Role` that is used to assign samples to the training and testing roles.

```plaintext
data ex1Data;
input obsnam $ Role : $5. tot tyr f1-f30 @@;
try = tot - tyr;
if (tyr) then tyr_log = log10(tyr); else tyr_log = -8;
if (try) then try_log = log10(try); else try_log = -8;
tot_log = log10(tot);
datalines;
17mix35 TRAIN 0.00003 0
  -6.215 -5.809 -5.114 -3.963 -2.897 -2.269 -1.675 -1.235
  -0.900 -0.659 -0.497 -0.395 -0.335 -0.315 -0.333 -0.377
  -0.453 -0.549 -0.658 -0.797 -0.878 -0.954 -1.060 -1.266
  -1.520 -1.804 -2.044 -2.269 -2.496 -2.714
19mix35 TRAIN 0.00003 3E-7
  -5.516 -5.294 -4.823 -3.858 -2.827 -2.249 -1.683 -1.218
  -0.907 -0.658 -0.501 -0.400 -0.345 -0.323 -0.342 -0.387
  -0.461 -0.554 -0.665 -0.803 -0.887 -0.960 -1.072 -1.272
  -1.541 -1.814 -2.058 -2.289 -2.496 -2.712
21mix35 TRAIN 0.00003 7.5E-7
  -5.519 -5.294 -4.501 -3.863 -2.827 -2.280 -1.716 -1.262
  -0.939 -0.694 -0.536 -0.444 -0.384 -0.369 -0.377 -0.421
  -0.495 -0.596 -0.706 -0.824 -0.917 -0.988 -1.103 -1.294
  -1.565 -1.841 -2.084 -2.320 -2.521 -2.729
23mix35 TRAIN 0.00003 1.5E-6
  ... more lines ...
  -5.138 -5.463 -5.461 -5.461 -5.461 -5.461 -5.461
  tyro2 TEST 0.0001 0.0001
  -1.081 -0.710 -0.470 -0.337 -0.327 -0.433 -0.602 -0.841
  -4.668 -4.668 -4.865 -4.865 -5.109 -5.111
;
data mycas.ex1Data;
set ex1Data;
run;
```

The following statements fit a PLS model that has 10 factors:

```plaintext
proc plsmod data=mycas.ex1Data nfac=10;
model tot_log tyr_log try_log = f1-f30;
run;
```
Example 17.1: Choosing a PLS Model by Test Set Validation

The “Model Information” table in Output 17.1.2 shows that no validation method is used. The “Number of Observations” table confirms that all 33 sample observations are used in the analysis.

**Output 17.1.2** Model Information and Number of Observations

The PLSMOD Procedure

<table>
<thead>
<tr>
<th>Model Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source</td>
</tr>
<tr>
<td>EX1DATA</td>
</tr>
<tr>
<td>Factor Extraction Method</td>
</tr>
<tr>
<td>Partial Least Squares</td>
</tr>
<tr>
<td>PLS Algorithm</td>
</tr>
<tr>
<td>NIPALS</td>
</tr>
<tr>
<td>Validation Method</td>
</tr>
<tr>
<td>None</td>
</tr>
</tbody>
</table>

Number of Observations Read: 33
Number of Observations Used: 33

The table in **Output 17.1.3** indicates that only four or five factors are required to explain almost all the variation in both the predictors and the responses.

**Output 17.1.3** Amount of Variation Explained

<table>
<thead>
<tr>
<th>Number of Extracted Factors</th>
<th>Current Percentage Variation Accounted for by Partial Least Squares Factors</th>
<th>Total Percentage Variation Accounted for by Partial Least Squares Factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>77.67903</td>
<td>77.67903</td>
</tr>
<tr>
<td>2</td>
<td>20.62719</td>
<td>98.30622</td>
</tr>
<tr>
<td>3</td>
<td>1.00143</td>
<td>99.30766</td>
</tr>
<tr>
<td>4</td>
<td>0.24930</td>
<td>99.55696</td>
</tr>
<tr>
<td>5</td>
<td>0.13077</td>
<td>99.68773</td>
</tr>
<tr>
<td>6</td>
<td>0.08970</td>
<td>99.77742</td>
</tr>
<tr>
<td>7</td>
<td>0.05684</td>
<td>99.83426</td>
</tr>
<tr>
<td>8</td>
<td>0.06730</td>
<td>99.90156</td>
</tr>
<tr>
<td>9</td>
<td>0.01521</td>
<td>99.91676</td>
</tr>
<tr>
<td>10</td>
<td>0.02627</td>
<td>99.94304</td>
</tr>
</tbody>
</table>

In order to choose the optimal number of PLS factors, you can explore how well models that are based on data in training roles and have different numbers of factors fit the data in testing roles. To do so, you can use the PARTITION statement to assign observations to training and testing roles based on the values of the input variable named **Role**, as follows:

```plaintext
proc plsmod data=mycas.ex1Data nfac=10 cvtest(stat=press seed=12345);
   model tot_log tyr_log try_log = f1-f30;
   partition roleVar = Role(train='TRAIN' test='TEST');
run;
```

Output 17.1.4 shows the “Model Information” table and the “Number of Observations” table. The “Model Information” table indicates that test set validation is used, and it displays information about the options...
that are used in the model comparison test. The “Number of Observations” table confirms that there are 18 observations for the training role and 15 for the testing role.

**Output 17.1.4** Model Information and Number of Observations with Test Set Validation

![The PLSMOD Procedure](image)

**The PLSMOD Procedure**

<table>
<thead>
<tr>
<th>Model Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source</td>
</tr>
<tr>
<td>Factor Extraction Method</td>
</tr>
<tr>
<td>PLS Algorithm</td>
</tr>
<tr>
<td>Validation Method</td>
</tr>
<tr>
<td>Validation Testing Criterion</td>
</tr>
<tr>
<td>Number of Random Permutations</td>
</tr>
<tr>
<td>Random Number Seed for Permutation</td>
</tr>
</tbody>
</table>

- Number of Observations Read: 33
- Number of Observations Used: 33
- Number of Observations Used for Training: 18
- Number of Observations Used for Testing: 15

**Output 17.1.5** displays the results of the test set validation. They indicate that although five PLS factors produce the minimum predicted residual sum of squares, the residuals for four factors are insignificantly different from the residuals for five factors. Thus, the smaller model is preferred.

**Output 17.1.5** Test Set Validation for the Number of PLS Factors

![The PLSMOD Procedure](image)

**The PLSMOD Procedure**

<table>
<thead>
<tr>
<th>Test Set Validation for the Number of Extracted Factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Extracted Factors</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>10</td>
</tr>
</tbody>
</table>

- Minimum Root Mean PRESS: 0.500034
- Minimizing Number of Factors: 5
- Smallest Number of Factors with p > 0.1: 4
### Output 17.1.5 continued

<table>
<thead>
<tr>
<th>Number of Extracted Factors</th>
<th>Model Effects</th>
<th>Response Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Current</td>
<td>Total</td>
</tr>
<tr>
<td>Current</td>
<td>Total</td>
<td>Current</td>
</tr>
<tr>
<td>1</td>
<td>81.16545</td>
<td>48.33854</td>
</tr>
<tr>
<td>2</td>
<td>16.81131</td>
<td>32.54654</td>
</tr>
<tr>
<td>3</td>
<td>1.76391</td>
<td>11.44380</td>
</tr>
<tr>
<td>4</td>
<td>0.19507</td>
<td>3.83631</td>
</tr>
</tbody>
</table>

### References


Chapter 18
The QTRSELECT Procedure

Contents

Overview: QTRSELECT Procedure .................................................. 828
PROC QTRSELECT Features ......................................................... 828
PROC QTRSELECT Compared with Other SAS Procedures ................. 829
Using CAS Sessions and CAS Engine Librefs ................................ 830
Getting Started: QTRSELECT Procedure ........................................ 831
Syntax: QTRSELECT Procedure .................................................... 839
PROC QTRSELECT Statement ....................................................... 839
BY Statement ................................................................. 840
CLASS Statement ............................................................ 840
CODE Statement ............................................................. 841
DISPLAY Statement ............................................................ 842
DISPLAYOUT Statement ......................................................... 843
EFFECT Statement ............................................................... 843
MODEL Statement .............................................................. 845
OUTPUT Statement .............................................................. 847
PARTITION Statement .......................................................... 849
SELECTION Statement .......................................................... 850
WEIGHT Statement ............................................................. 851
Details: QTRSELECT Procedure .................................................. 851
Quantile Regression .............................................................. 851
Criteria Used in Model Selection ................................................. 853
Diagnostic Statistics ............................................................... 856
Classification Variables and the SPLIT Option ................... 857
Using Validation and Test Data ................................................. 858
Computational Method .......................................................... 861
Displayed Output ............................................................... 862
ODS Table Names ............................................................... 865
ODS Graphics .................................................................. 866
Examples: QTRSELECT Procedure .............................................. 867
Example 18.1: Simulation Study ................................................ 867
Example 18.2: Growth Charts for Body Mass Index .................. 869
Example 18.3: Pollution and Mortality ...................................... 871
References .................................................................. 879
Overview: QTRSELECT Procedure

The QTRSELECT procedure fits and performs model selection for quantile regression models in SAS Viya. Quantile regression uses a linear function model to fit the quantiles of a response variable conditional on the explanatory variables. The model does not assume a particular parametric distribution for the response. The models that PROC QTRSELECT supports can contain main effects that consist of both continuous and classification variables and interaction effects of these variables. The models can also include constructed effects such as splines. The procedure offers a number of effect-selection methods, including stepwise methods. It also offers extensive capabilities for customizing the model selection by using a wide variety of selection and stopping criteria, including significance-level-based criteria, information criteria, and modern validation-based criteria. PROC QTRSELECT also provides a variety of quantile regression diagnostics that are conditional on the selected model.

PROC QTRSELECT Features

The main features of the QTRSELECT procedure are as follows:

- **Model specification**
  - supports quantile regression for single or multiple quantile levels
  - supports multiple parameterizations for classification effects
  - supports any degree of interactions (crossed effects) and nested effects
  - supports hierarchical model selection strategy among effects that have a containment relationship
  - supports partitioning of data into training, validation, and testing roles
  - provides a WEIGHT statement for weighted analysis

- **Selection control**
  - provides multiple effect-selection methods
  - offers selection of individual levels of classification effects
  - provides effect selection based on a variety of selection criteria
  - provides stopping rules based on a variety of model evaluation criteria
  - supports stopping and selection rules based on external validation

- **Display and output**
  - provides a CODE statement to produce SAS code that can score a new data set
  - produces output data tables that contain predicted values, residuals, and confidence limits
  - uses ODS Graphics to create model selection plots as part of its output. For more information about ODS Graphics, see the section “ODS Graphics” on page 866

The QTRSELECT procedure supports the following effect-selection methods. For a more detailed description of these methods, see the section “SELECTION Statement” on page 36 in Chapter 3, “Shared Concepts.”
• Forward selection starts with no effects in the model and adds effects.
• Backward elimination starts with all effects in the model and deletes effects.
• Stepwise regression is similar to forward selection, except that effects already in the model do not necessarily stay there.

Because the QTRSELECT procedure runs on CAS, it also does the following:

• enables you to run on a cluster of machines that distribute the data and the computations
• enables you to run in single-machine mode on CAS
• exploits all the available cores and concurrent threads. For information about how PROC QTRSELECT uses threads, see the section “Multithreading” on page 81 in Chapter 3, “Shared Concepts.”

PROC QTRSELECT Compared with Other SAS Procedures

The QTRSELECT procedure provides quantile regression functionality that is comparable to that of the HPQUANTSELECT, QUANTSELECT, and QUANTREG procedures in SAS/STAT software.

PROC QTRSELECT Compared with the HPQUANTSELECT Procedure

The functionality of the QTRSELECT procedure closely resembles that of the HPQUANTSELECT procedure, which is a high-performance procedure. The QTRSELECT procedure is the next generation of the HPQUANTSELECT procedure, and it was developed specifically for SAS Viya. Both procedures are designed to run on a cluster of machines that distribute the data and the computations.

Both the QTRSELECT and HPQUANTSELECT procedures fit and perform effect selection for quantile regression models. The models can contain main effects that consist of both continuous and classification variables and interaction effects of these variables. The QTRSELECT procedure is additionally capable of constructing complex effects including univariate spline and polynomial expansions, and producing effect-selection plots by using ODS Graphics.

With the QTRSELECT and HPQUANTSELECT procedures, you request model selection by using the SELECTION statement. Both procedures offer the same methods of effect selection.

PROC QTRSELECT Compared with the QUANTSELECT Procedure

Both the QTRSELECT and QUANTSELECT procedures fit and perform model selection for quantile regression models. The models can contain main effects that consist of both continuous and classification variables and interaction effects of these variables. Both procedures support the forward, backward, and stepwise effect-selection methods and the ability to use separate validation and test data as specified in the PARTITION statement.

PROC QTRSELECT provides confidence limits and Wald tests for parameters, and prediction limits for quantiles.

PROC QUANTSELECT provides LASSO and adaptive LASSO effect-selection methods and effect selection for quantile process regression.
PROC QTRSELECT Compared with the QUANTREG Procedure

The QTRSELECT procedure chooses and fits quantile regression models. The QUANTREG procedure fits quantile regression models, but it does not offer model selection.

PROC QTRSELECT provides a variety of effect-selection methods, fit statistics, graphical effect-selection summary, and a cloud-enabled interior-point algorithm. It also supports partitioning the data into training, validation, and testing roles.

PROC QUANTREG provides simplex and smoothing algorithms, a rank-score test, a likelihood test, Markov-chain-marginal-bootstrap covariance estimation, many diagnostic and fit plots, and quantile process regression.

Using CAS Sessions and CAS Engine Librefs

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:

```
proc options option=(CASHOST CASPORT);
run;
```

In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:

```
cas mysess;
libname mycas cas sessref=mysess;
```

The CAS statement creates the CAS session named mysess, and the LIBNAME statement creates the mycas CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the CASHOST and CASPORT of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the mysess session, you can terminate it by using the TERMINATE option in the CAS statement as follows:

```
cas mysess terminate;
```

For more information about the CAS and LIBNAME statements, see the section “Introduction to Shared Concepts” on page 10 in Chapter 3, “Shared Concepts.”
The following example is modeled on the example in the section “Getting Started: QUANTSELECT Procedure” in the SAS/STAT User’s Guide. The Sashelp.baseball data set contains salary and performance information for Major League Baseball (MLB) players, excluding pitchers, who played in at least one game in both the 1986 and 1987 seasons. The salaries (Time Inc. 1987) are for the 1987 season, and the performance measures are for the 1986 season (Reichler 1987).

The following statements display the variables in the data set. Figure 18.1 shows the results.

```sas
proc contents varnum data=sashelp.baseball;
   ods select position;
run;
```

**Figure 18.1** Sashelp.Baseball Data Set

The CONTENTS Procedure

<table>
<thead>
<tr>
<th>#</th>
<th>Variable</th>
<th>Type</th>
<th>Len</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Name</td>
<td>Char</td>
<td>18</td>
<td>Player’s Name</td>
</tr>
<tr>
<td>2</td>
<td>Team</td>
<td>Char</td>
<td>14</td>
<td>Team at the End of 1986</td>
</tr>
<tr>
<td>3</td>
<td>nAtBat</td>
<td>Num</td>
<td>8</td>
<td>Times at Bat in 1986</td>
</tr>
<tr>
<td>4</td>
<td>nHits</td>
<td>Num</td>
<td>8</td>
<td>Hits in 1986</td>
</tr>
<tr>
<td>5</td>
<td>nHome</td>
<td>Num</td>
<td>8</td>
<td>Home Runs in 1986</td>
</tr>
<tr>
<td>6</td>
<td>nRuns</td>
<td>Num</td>
<td>8</td>
<td>Runs in 1986</td>
</tr>
<tr>
<td>7</td>
<td>nRBI</td>
<td>Num</td>
<td>8</td>
<td>RBIs in 1986</td>
</tr>
<tr>
<td>8</td>
<td>nBB</td>
<td>Num</td>
<td>8</td>
<td>Walks in 1986</td>
</tr>
<tr>
<td>9</td>
<td>YrMajor</td>
<td>Num</td>
<td>8</td>
<td>Years in the Major Leagues</td>
</tr>
<tr>
<td>10</td>
<td>CrAtBat</td>
<td>Num</td>
<td>8</td>
<td>Career Times at Bat</td>
</tr>
<tr>
<td>11</td>
<td>CrHits</td>
<td>Num</td>
<td>8</td>
<td>Career Hits</td>
</tr>
<tr>
<td>12</td>
<td>CrHome</td>
<td>Num</td>
<td>8</td>
<td>Career Home Runs</td>
</tr>
<tr>
<td>13</td>
<td>CrRuns</td>
<td>Num</td>
<td>8</td>
<td>Career Runs</td>
</tr>
<tr>
<td>14</td>
<td>CrRbi</td>
<td>Num</td>
<td>8</td>
<td>Career RBIs</td>
</tr>
<tr>
<td>15</td>
<td>CrBB</td>
<td>Num</td>
<td>8</td>
<td>Career Walks</td>
</tr>
<tr>
<td>16</td>
<td>League</td>
<td>Char</td>
<td>8</td>
<td>League at the End of 1986</td>
</tr>
<tr>
<td>17</td>
<td>Division</td>
<td>Char</td>
<td>8</td>
<td>Division at the End of 1986</td>
</tr>
<tr>
<td>18</td>
<td>Position</td>
<td>Char</td>
<td>8</td>
<td>Position(s) in 1986</td>
</tr>
<tr>
<td>19</td>
<td>nOuts</td>
<td>Num</td>
<td>8</td>
<td>Put Outs in 1986</td>
</tr>
<tr>
<td>20</td>
<td>nAssts</td>
<td>Num</td>
<td>8</td>
<td>Assists in 1986</td>
</tr>
<tr>
<td>21</td>
<td>nError</td>
<td>Num</td>
<td>8</td>
<td>Errors in 1986</td>
</tr>
<tr>
<td>22</td>
<td>Salary</td>
<td>Num</td>
<td>8</td>
<td>1987 Salary in $ Thousands</td>
</tr>
<tr>
<td>23</td>
<td>Div</td>
<td>Char</td>
<td>16</td>
<td>League and Division</td>
</tr>
<tr>
<td>24</td>
<td>logSalary</td>
<td>Num</td>
<td>8</td>
<td>Log Salary</td>
</tr>
</tbody>
</table>

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”
You can load the Sashelp.Baseball data set into your CAS session by using your CAS engine libref with the following DATA step:

```summary
data mycas.baseball;
  set sashelp.baseball;
run;
```

These statements assume that your CAS engine libref is named `mycas`, as in the section “Using CAS Sessions and CAS Engine Librefs” on page 830, but you can substitute any appropriately defined CAS engine libref.

Suppose you want to investigate how the MLB players’ salaries for the 1987 season depend on performance measures for the players’ previous season and MLB career. You might worry that some players who are outliers could dominate your least squares analysis. To address this concern, you can use the following statements to obtain a median regression model, which is equivalent to the 50th conditional percentile or the quantile regression model at quantile level 0.5:

```summary
proc qtrselect data=mycas.baseball;
  class league division;
  model Salary = nAtBat nHits nHome nRuns nRBI nBB yrMajor crAtBat crHits crHome crRuns crRbi crBB league division nOuts nAssts nError;
run;
```

If you do not use the `SELECTION` statement, the QTRSELECT procedure fits the full model that is specified by the `MODEL` statement without any effect selection.

The `qtrselect` Procedure

<table>
<thead>
<tr>
<th>Class Level Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class</td>
</tr>
<tr>
<td>League</td>
</tr>
<tr>
<td>Division</td>
</tr>
</tbody>
</table>

Figure 18.2 displays the “Number of Observations,” “Class Level Information,” and “Dimensions” tables. The “Number of Observations” table shows that, of the 322 observations, PROC QTRSELECT uses only 263 observations for model fitting and ignores 59 incomplete observations.

The “Class Level Information” table shows level information for two CLASS effects that the `CLASS` statement identifies: League and Division. League has two levels: American League and National League. Division also has two levels: East Division and West Division.

The “Dimensions” table shows that the `MODEL` statement identifies 19 effects for model fitting besides the intercept effect. Because the 19 effects include two CLASS effects and each level of a CLASS effect corresponds to a parameter, the 19 effects contain a total of 21 parameters.
Figure 18.3  Fit Statistics

The QTRSELECT Procedure

Quantile Level = 0.5

| Parameter          | DF | Estimate | Standard Error | t Value | Pr > |t| |
|--------------------|----|----------|----------------|---------|------|---|
| Intercept          | 1  | -67.75322| 39.95908       | -1.70   | 0.0912|
| nAtBat             | 1  | -1.57112 | 0.44700        | -3.51   | 0.0005|
| nHits              | 1  | 8.82192  | 1.94990        | 4.52    | <.0001|
| nHome              | 1  | -5.91757 | 4.91015        | -1.21   | 0.2293|
| nRuns              | 1  | -5.17076 | 2.14914        | -2.41   | 0.0169|
| nRBI               | 1  | 0.77547  | 2.15469        | 0.36    | 0.7192|
| nBB                | 1  | 5.28866  | 1.67603        | 3.16    | 0.0018|
| YrMajor            | 1  | 6.61877  | 6.61798        | 1.00    | 0.3182|
| CrAtBat            | 1  | -0.04463 | 0.15485        | -0.29   | 0.7734|
| CrHits             | 1  | 0.07896  | 0.73594        | 0.11    | 0.9146|
| CrHome             | 1  | 3.78231  | 1.90065        | 1.99    | 0.0477|
| CrRuns             | 1  | 1.23105  | 0.77137        | 1.60    | 0.1118|
| CrRbi              | 1  | -0.70695 | 0.76888        | -0.92   | 0.3588|
| CrBB               | 1  | -0.68911 | 0.41382        | -1.67   | 0.0971|
| League American    | 1  | -34.39136| 24.37175       | -1.41   | 0.1595|
| League National    | 0  | 0        | .              | .       | .    |
| Division East      | 1  | 60.30856 | 27.28730       | 2.21    | 0.0280|
| Division West      | 0  | 0        | .              | .       | .    |
| nOuts              | 1  | 0.23273  | 0.12110        | 1.92    | 0.0558|
| nAssts             | 1  | 0.09824  | 0.18888        | 0.52    | 0.6035|
| nError             | 1  | -0.81574 | 3.51436        | -0.23   | 0.8166|

Figure 18.3 displays the “Fit Statistics” table, which shows the values of model fitting criteria for the fitted median model. For more information about model fitting criteria for quantile regression, see the section “Details: QTRSELECT Procedure” on page 851.

Figure 18.4  Parameter Estimates

Parameter Estimates

| Parameter          | DF | Estimate | Standard Error | t Value | Pr > |t| |
|--------------------|----|----------|----------------|---------|------|---|
| Intercept          | 1  | -67.75322| 39.95908       | -1.70   | 0.0912|
| nOuts              | 1  | 0.23273  | 0.12110        | 1.92    | 0.0558|
| nAssts             | 1  | 0.09824  | 0.18888        | 0.52    | 0.6035|
| nError             | 1  | -0.81574 | 3.51436        | -0.23   | 0.8166|

Figure 18.4 displays the “Parameter Estimates” table, which shows the parameter estimates of the fitted median model. You can see that, of the 19 effective parameters whose degrees of freedom are not zero, the fitted model contains 13 insignificant parameters whose 95% confidence intervals cover zeros. Because more than half of the 19 effective parameters are insignificant, you might worry that the model is overfitted.
It is well known that both overfitting and underfitting harm the prediction performance of a model. You can prevent overfitting and underfitting by using a good effect-selection technique. The following statements apply the forward selection method and the SL (significance level) criterion to choose a parsimonious model for the mycas.baseball data table:

```
proc qtrselect data=mycas.baseball;
   class league division;
   model Salary = nAtBat nHits nHome nRuns nRBI nBB
                    yrMajor crAtBat crHits crHome crRuns crRbi
                    crBB league division nOuts nAssts nError
            / clb;
   selection method=forward(select=sl sle=0.1);
run;
```

The CLB option in the MODEL statement requests 95% confidence limits for the parameter estimates. The SLE=0.1 option in the SELECTION statement specifies the significance level for entry. A candidate effect can enter the model at a certain selection step only if the following conditions are met:

- Its $p$-value is the smallest among all the valid candidate effects.
- Its $p$-value is smaller than 0.1 (the significance level for entry).

For more information about using significance levels in effect selection, see the section “Statistical Tests for Significance Level” on page 855.

**Figure 18.5** Selection Information

<table>
<thead>
<tr>
<th>The QTRSELECT Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Selection Information</strong></td>
</tr>
<tr>
<td>Selection Method</td>
</tr>
<tr>
<td>Select Criterion</td>
</tr>
<tr>
<td>Stop Criterion</td>
</tr>
<tr>
<td>Effect Hierarchy Enforced</td>
</tr>
<tr>
<td>Entry Significance Level (SLE)</td>
</tr>
<tr>
<td>Stop Horizon</td>
</tr>
</tbody>
</table>

Figure 18.5 displays the “Selection Information” table. The “Selection Information” provides details about the method and criteria used to perform the model selection. The requested selection method is the forward selection method where the decisions about what effects to add at any step and when to terminate the selection are both based on the significance level criterion.
Figure 18.6 Selection Summary

The QTRSELECT Procedure

Quantile Level = 0.5
Selection Details

<table>
<thead>
<tr>
<th>Step</th>
<th>Effect Entered</th>
<th>Number Effects In</th>
<th>p Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Intercept</td>
<td>1</td>
<td>.</td>
</tr>
<tr>
<td>1</td>
<td>CrHome</td>
<td>2 &lt;.0001</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>nHits</td>
<td>3 &lt;.0001</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>CrHits</td>
<td>4 &lt;.0001</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>nOuts</td>
<td>5 0.0185</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>nAtBat</td>
<td>6 0.0182</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Division</td>
<td>7 0.0118</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>nBB</td>
<td>8 0.0647</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>nRuns</td>
<td>9 0.0558</td>
<td></td>
</tr>
</tbody>
</table>

Figure 18.6 displays the “Selection Summary” table. Each row in the “Selection Summary” table shows the effect that enters the model at the corresponding step of the effect selection process together with its p-value for adding the effect into the model at that step.

Figure 18.7 Stopping and Selection Reasons

Selection stopped because no candidate for entry is significant at the 0.1 level.

The model at step 8 is selected.

Selected Effects: Intercept nAtBat nHits nRuns nBB CrHits CrHome Division nOuts

Figure 18.7 displays the “Stop Reason,” “Selection Reason,” and “Selected Effects” tables. The “Stop Reason” and “Selection Reason” tables indicate that effect selection stopped because no candidate for entry was significant at the 0.1 level after step 8. The “Selected Effects” table lists the effects that are included in the selected model.

Figure 18.8 Details of the Selected Model

The QTRSELECT Procedure

Quantile Level = 0.5
Selected Model

<table>
<thead>
<tr>
<th>Objective Function</th>
<th>26568</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>0.39232</td>
</tr>
<tr>
<td>Adj R1</td>
<td>0.37318</td>
</tr>
<tr>
<td>AIC</td>
<td>2445.64547</td>
</tr>
<tr>
<td>AICC</td>
<td>2446.35693</td>
</tr>
<tr>
<td>SBC</td>
<td>2477.79485</td>
</tr>
<tr>
<td>ACL</td>
<td>101.01768</td>
</tr>
</tbody>
</table>
## Parameter Estimates

| Parameter  | DF | Estimate   | Standard Error | 95% Confidence Limits | t Value | Pr > |t| |
|------------|----|------------|----------------|------------------------|---------|------|---|
| Intercept  | 1  | -130.65536 | 33.04546       | -195.73336             | -3.95   | <.0001 |
| nAtBat     | 1  | -1.20522   | 0.41690        | -2.02624               | -2.89   | 0.0042 |
| nHits      | 1  | 7.76667    | 1.81045        | 11.33207               | 4.29    | <.0001 |
| nRuns      | 1  | -3.92180   | 1.87567        | -7.61566               | -2.09   | 0.0375 |
| nBB        | 1  | 3.92049    | 1.86565        | 5.97532                | 3.76    | 0.0002 |
| CrHits     | 1  | 0.17697    | 0.06856        | 0.31198                | 2.58    | 0.0104 |
| CrHome     | 1  | 1.66939    | 0.73083        | 3.10866                | 2.28    | 0.0232 |
| Division East | 1    | 65.14327 | 24.48038     | 113.35364              | 2.66    | 0.0083 |
| Division West | 0   | 0         | .             | .                      | .       | .     |
| nOuts      | 1  | 0.23719    | 0.11403        | 0.46176                | 2.08    | 0.0385 |

The “Fit Statistics” and “Parameter Estimates” tables in Figure 18.8 give details of the final selected model. You can see that all nine effective parameters (excluding Division West) are significant at the 5% significance level, corresponding to the 95% confidence limits.

Like the sample median, a median regression model is robust to extreme observations, because it depends only on a small middle subset of all the observations in the data table. However, it is less representative of the entire conditional distribution of the response variable. You might want to further investigate the mycas.baseball data table at other quantile levels. The following statements select quantile regression models at the quantile levels 0.1 and 0.9, which correspond to the 10% and 90% conditional percentiles of the players’ salaries:

```plaintext
proc qtrselect data=mycas.baseball alpha=0.1;
  class league division;
  model Salary = nAtBat nHits nHome nRuns nRBI nBB
                   yrMajor crAtBat crHits crHome crRuns crRbi
                   crBB league division nOuts nAssts nError
    / quantile=0.1 0.9 stb clb;
  selection method=backward(select=sl sls=0.1);
run;
```

The ALPHA=0.1 option in the PROC statement sets the significance level to 0.1. Combined with the CLB option in the MODEL statement, the ALPHA=0.1 option requests 90% confidence limits for parameter estimates. The QUANTILE= option in the MODEL statement specifies two quantile levels, 0.1 and 0.9, for fitting quantile regression models. The METHOD=BACKWARD option in the SELECTION statement specifies the backward elimination method for effect selection.
### Figure 18.9 continued

| Parameter     | DF | Estimate  | Standardized Estimate | Standard Error | 90% Confidence Limits | t Value | Pr > |t| |
|---------------|----|-----------|-----------------------|----------------|------------------------|---------|------|---|
| Intercept     | 1  | 4.75224   | 0                     | 18.94983       | -26.53111               | 0.25    | 0.8022 |
| nAtBat        | 1  | -0.73670  | -0.23293              | 0.17958        | -1.03316                | -4.10   | <.0001|
| nHits         | 1  | 2.69396   | 0.26272               | 0.63510        | 1.64551                 | 4.24    | <.0001|
| nBB           | 1  | 1.81807   | 0.08668               | 0.40595        | 1.14791                 | 4.48    | <.0001|
| CrRuns        | 1  | 0.65476   | 0.48600               | 0.09560        | 0.49694                 | 6.85    | <.0001|
| CrBB          | 1  | -0.44622  | -0.26842              | 0.16254        | -0.71454                | -2.75   | 0.0065|
| Division East | 1  | 28.35406  | 0.03148               | 10.68922       | 10.70774                | 2.65    | 0.0085|
| Division West | 0  | 0         | 0                     | .              | .                      |        |      |
| nAssts        | 1  | 0.14958   | 0.04811               | 0.05897        | 0.05223                 | 2.54    | 0.0118|

Figure 18.9 displays the “Selected Effects” and “Parameter Estimates” tables at quantile level 0.1.

### Figure 18.10 Parameter Estimates at Quantile Level 0.9

| Parameter     | DF | Estimate  | Standardized Estimate | Standard Error | 90% Confidence Limits | t Value | Pr > |t| |
|---------------|----|-----------|-----------------------|----------------|------------------------|---------|------|---|
| Intercept     | 1  | 20.39804  | 0                     | 58.17164       | -75.63745               | 0.35    | 0.7261|
| nHits         | 1  | 2.30897   | 0.22517               | 0.55640        | 1.39042                 | 4.15    | <.0001|
| nBB           | 1  | 3.09799   | 0.14770               | 1.44414        | 0.71386                 | 2.15    | 0.0329|
| CrAtBat       | 1  | -0.44914  | -2.28028              | 0.14651        | -0.69101                | -3.07   | 0.0024|
| CrHits        | 1  | 2.48064   | 3.56823               | 0.51725        | 1.62672                 | 4.80    | <.0001|
| CrHome        | 1  | 6.29896   | 1.20138               | 1.37134        | 4.03502                 | 4.59    | <.0001|
| CrRbi         | 1  | -2.12293  | -1.54553              | 0.76546        | -3.38662                | -2.77   | 0.0060|
| League American| 1 | -103.28955| -0.11451              | 33.68480       | -158.89975              | -3.07   | 0.0024|
| League National| 0 | 0         | 0                     | .              | .                      |        |      |
| Division East | 1  | 107.46694 | 0.11932               | 50.82797       | 23.55512                | 2.11    | 0.0355|
| Division West | 0  | 0         | 0                     | .              | .                      |        |      |
| nOuts         | 1  | 0.39766   | 0.24676               | 0.12820        | 0.18601                 | 3.10    | 0.0021|

Figure 18.9 displays the “Selected Effects” and “Parameter Estimates” tables at quantile level 0.9.

You might want to compute the 90th percentile predictions for players’ salaries and find out which players were overpaid based on the quantile regression model at quantile level 0.9. The following statements repeat the backward elimination method at quantile level 0.9, compute and sort the overpaid players’ salaries, and output the observations for the top 10 overpaid players in the mycas.baseball data table:

```plaintext
proc qtrselect data=mycas.baseball alpha=0.1;
  class league division;
  model Salary = nAtBat nHits nHome nRuns nRBI nBB
                 yrMajor crAtBat crHits crHome crRuns crRbi
                 crBB league division nOuts nAssts nError
     / quantile=0.9 clb;
  selection method=backward(select=s1 sls=0.1);
```
output out=mycas.BaseballOverpaid copyvar=(Name Salary) r=Overpaid p=PredictedSalary lclm uclm;
run;
data BaseballOverpaid;
set mycas.BaseballOverpaid;
run;
proc sort data=BaseballOverpaid;
by descending Overpaid;
run;
proc print data=BaseballOverpaid(obs=10);
var Name Salary Overpaid PredictedSalary lclm uclm;
run;
The LCLM and UCLM options, respectively, request lower and upper bounds of $100(1 - \alpha)$% confidence intervals for the expected conditional quantile predictions of players’ salaries at quantile level 0.9.

Figure 18.11 Top 10 Overpaid Baseball Players at Quantile Level 0.9

<table>
<thead>
<tr>
<th>Obs</th>
<th>Name</th>
<th>Salary</th>
<th>Overpaid</th>
<th>PredictedSalary</th>
<th>LCLM</th>
<th>UCLM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Smith, Ozzie</td>
<td>1940.0</td>
<td>1084.54</td>
<td>855.46</td>
<td>611.09</td>
<td>1099.83</td>
</tr>
<tr>
<td>2</td>
<td>Wiggins, Alan</td>
<td>700.0</td>
<td>220.74</td>
<td>479.26</td>
<td>345.93</td>
<td>612.60</td>
</tr>
<tr>
<td>3</td>
<td>Murray, Eddie</td>
<td>2460.0</td>
<td>213.10</td>
<td>2246.90</td>
<td>1982.06</td>
<td>2511.74</td>
</tr>
<tr>
<td>4</td>
<td>Strawberry, Darryl</td>
<td>1220.0</td>
<td>187.64</td>
<td>1032.36</td>
<td>914.99</td>
<td>1149.72</td>
</tr>
<tr>
<td>5</td>
<td>Gibson, Kirk</td>
<td>1300.0</td>
<td>177.24</td>
<td>1122.76</td>
<td>1011.97</td>
<td>1233.56</td>
</tr>
<tr>
<td>6</td>
<td>Trevino, Alex</td>
<td>512.5</td>
<td>149.70</td>
<td>362.80</td>
<td>273.17</td>
<td>452.43</td>
</tr>
<tr>
<td>7</td>
<td>Ramirez, Rafael</td>
<td>875.0</td>
<td>141.09</td>
<td>733.91</td>
<td>599.52</td>
<td>868.31</td>
</tr>
<tr>
<td>8</td>
<td>Romero, Ed</td>
<td>375.0</td>
<td>128.71</td>
<td>246.29</td>
<td>144.88</td>
<td>347.70</td>
</tr>
<tr>
<td>9</td>
<td>Mattingly, Don</td>
<td>1975.0</td>
<td>124.82</td>
<td>1850.18</td>
<td>1557.81</td>
<td>2142.55</td>
</tr>
<tr>
<td>10</td>
<td>Puhl, Terry</td>
<td>900.0</td>
<td>104.34</td>
<td>795.66</td>
<td>625.51</td>
<td>965.81</td>
</tr>
</tbody>
</table>

Output 18.11 shows the information about the top 10 overpaid players according to the final selected quantile regression model at quantile level 0.9. Ozzie Smith is in first place. This might be because, although Smith was known for his defensive brilliance, the model weights offensive performance measures much more than defensive performance measures.
Syntax: QTRSELECT Procedure

The following statements are available in the QTRSELECT procedure:

```
PROC QTRSELECT < options > ;
    BY variables ;
    CLASS variable < (options) > . . . < variable < (options) > > < / global-options > ;
    CODE < options > ;
    DISPLAY < table-list > < / options > ;
    DISPLAYOUT table-spec-list < / options > ;
    EFFECT name = effect-type (variables < / options > ) ;
    MODEL dependent = < effects > < / model-options > ;
    OUTPUT OUT = CAS-libref.data-table < keyword = name > . . . < keyword = name > > < / options > ;
    PARTITION partition-options ;
    SELECTION < METHOD = method < (method-options) > > < options > ;
    WEIGHT variable ;
```

The PROC QTRSELECT statement and a single MODEL statement are required. All other statements are optional. The CLASS statement can appear multiple times. If a CLASS statement is specified, it must precede the MODEL statement.

The rest of this section provides detailed syntax information about each of the preceding statements, beginning with the PROC QTRSELECT statement. The remaining statements are described in alphabetical order.

PROC QTRSELECT Statement

```
PROC QTRSELECT < options > ;
```

The PROC QTRSELECT statement invokes the procedure. Table 18.1 summarizes the options in the PROC QTRSELECT statement by function.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Basic Options</strong></td>
<td></td>
</tr>
<tr>
<td>DATA=</td>
<td>Specifies the input data table</td>
</tr>
<tr>
<td><strong>Other Options</strong></td>
<td></td>
</tr>
<tr>
<td>ALPHA=</td>
<td>Sets the significance level to be used for the construction of confidence intervals</td>
</tr>
<tr>
<td>COV=SPARSITY</td>
<td>Specifies the sparsity-function method for estimating the covariance matrix of the parameter estimates</td>
</tr>
<tr>
<td>NOCLPRINT</td>
<td>Limits or suppresses the display of class levels</td>
</tr>
</tbody>
</table>

You can specify the following options:
**ALPHA=** *number*

sets the significance level to be used for the construction of confidence intervals. The value must be between 0 and 1; the default value of 0.05 results in 95% intervals. This option affects the keywords LCLM, UCLM, and STDP in the OUTPUT statement and the CLB option in the MODEL statement.

**COV=** **SPARSITY** *(BF | HS)*

specifies the sparsity-function bandwidth method for estimating the covariance matrix of the parameter estimates. You can specify the following suboptions:

- **BF** uses the Bofinger bandwidth method.
- **HS** uses the Hall-Sheather bandwidth method.

By default, COV=SPARSITY(HS). For more information, see the section “Details: QTRSELECT Procedure” on page 851.

**DATA=** *CAS-libref.data-table*

names the input data table for PROC QTRSELECT to use. The default is the most recently created data table. *CAS-libref.data-table* is a two-level name, where

- **CAS-libref** refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to the data, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about *CAS-libref*, see the section “Using CAS Sessions and CAS Engine Librefs” on page 830.
- **data-table** specifies the name of the input data table.

**NOCLPRINT<=** *number>*

suppresses the display of the “Class Level Information” table if you do not specify *number*. If you specify *number*, the values of the classification variables are displayed for only those variables whose number of levels is less than *number*. Specifying *number* helps reduce the size of the “Class Level Information” table if some classification variables have a large number of levels.

---

**BY Statement**

**BY** *variables* ;

You can specify a BY statement in PROC QTRSELECT to obtain separate analyses of observations in groups that are defined by the values of the BY variables. If you specify more than one BY statement, only the last one specified is used. For more information, see the discussion of BY-group processing in *SAS Language Reference: Concepts*.

---

**CLASS Statement**

**CLASS** *variable* *(options)*...*variable* *(options)* *(global-options)* ;

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. You can list the response variable for binary models in the CLASS statement, but this is not required.
Table 18.2 summarizes the values that you can use for either an option or a global-option. The options are fully documented in the section “CLASS Statement” on page 12 in Chapter 3, “Shared Concepts.”

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESCENDING</td>
<td>Reverses the sort order</td>
</tr>
<tr>
<td>MISSING</td>
<td>Treats missing values as valid levels</td>
</tr>
<tr>
<td>ORDER=</td>
<td>Specifies the sort order for the levels</td>
</tr>
<tr>
<td>PARAM=</td>
<td>Specifies the parameterization of the variable</td>
</tr>
<tr>
<td>REF=</td>
<td>Specifies the reference level of the variable</td>
</tr>
<tr>
<td>SPLIT</td>
<td>Allows design columns for a variable to enter or leave the model independently</td>
</tr>
</tbody>
</table>

**CODE Statement**

```sas
CODE <options>;```

The CODE statement writes SAS DATA step code for computing predicted values of the fitted model to a file, to a catalog entry, or to a CAS table. To score new data, you can then include the file or the catalog entry in a DATA step, or you can specify the CAS table in the `runCodeTable` action in the `dataStep` action set (for more information, see SAS Viya: System Programming Guide).

Table 18.3 summarizes the options available in the CODE statement.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMMENT</td>
<td>Adds comments to the generated code</td>
</tr>
<tr>
<td>FILE=</td>
<td>Names the file in which to save the generated code</td>
</tr>
<tr>
<td>FORMATWIDTH=</td>
<td>Specifies the numeric format width for the regression coefficients</td>
</tr>
<tr>
<td>INDENTSIZE=</td>
<td>Specifies the number of spaces to indent the generated code</td>
</tr>
<tr>
<td>LABELID=</td>
<td>Specifies a number used to construct names and labels</td>
</tr>
<tr>
<td>LINESIZE=</td>
<td>Specifies the line size for the generated code</td>
</tr>
<tr>
<td>NOTRIM</td>
<td>Compares formatted values, including blank padding</td>
</tr>
<tr>
<td>OUT=</td>
<td>Names an output CAS table in which to save the generated code</td>
</tr>
</tbody>
</table>

For more information about the syntax of the CODE statement, see the section “CODE Statement” on page 16 in Chapter 3, “Shared Concepts.”

The predicted values in the SAS DATA step code are computed based on the parameter estimates for the selected models. If you specify multiple quantile levels by using the QUANTILES option in the MODEL statement, then one variable of the predicted values is generated for each specified quantile level.
Chapter 18: The QTRSELECT Procedure

DISPLAY Statement

DISPLAY < table-list > < / options > ;

The DISPLAY statement enables you to specify a list of display tables to display or exclude. This statement
is similar to the ODS SELECT, ODS EXCLUDE, and ODS TRACE statements. However, the DISPLAY
statement can improve performance when a large number of tables could be generated (such as in BY-group
processing). The procedure processes the DISPLAY statement on a CAS server and thus sends only a
subset of ODS tables to the SAS client. Because ODS statements are processed on a SAS client, first all the
generated display tables are sent to the client, and then the client creates a subset.

If you use both DISPLAY and ODS statements together, the DISPLAY statement takes precedence over the
ODS statements. Note that the ODS EXCLUDE statement processes tables that are sent to the client after
they have been filtered by the DISPLAY statement. In some cases, it might appear that the ODS EXCLUDE
statement is taking precedence because it can further filter the tables. For more information about ODS, see

You can specify the table-list as a list of table names, paths, partial pathnames, and regular expressions.

The table names that you can specify are listed in the section “ODS Table Names” on page 865.
A path is a table name that is prefixed with dot-separated grouping information. For example, a
SelectionSummary table that a procedure produces during a selection routine might have the path By-
group1.Summary:SelectionSummary. A partial pathname does not include all groups; for example, Selection-
Summary and Summary:SelectionSummary are partial pathnames for Bygroup1.Summary:SelectionSummary.

When you specify a table name or partial pathname, all display tables whose paths end in the specified name
are selected for display or exclusion. For example, both SelectionSummary and Summary:SelectionSummary
select Bygroup1.Summary:SelectionSummary.

A regular expression is enclosed in forward slashes (/). For example, specifying “/tions/” selects all pathnames
that contain the substring “tions”; in particular, the Bygroup1.Summary:SelectionSummary table is selected.
Specifying “!/tions/” selects all pathnames that do not contain the substring “tions”; in particular, the
Bygroup1.Summary:SelectionSummary table is not selected.

You can specify the following options after a slash (/):

CASESENSITIVE

performs a case-sensitive comparison of table names in the table-list to display table names when
tables are subsetted for display. To preserve case, you must enclose table names in the table-list in
quotation marks.

EXCLUDE

displays all display tables except those that you specify in the table-list.

EXCLUDEALL

suppresses display of all tables. This option takes precedence over the other options.

TRACE

displays the display table names, labels, and paths.
DISPLAYOUT Statement

DISPLAYOUT table-spec-list < / options > ;

The DISPLAYOUT statement enables you to create CAS output tables from your displayed output. This statement is similar to the ODS OUTPUT statement. For more information about ODS, see SAS Output Delivery System: Procedures Guide.

The table-spec-list specifies a list of CAS output tables to create. Each entry in the list has either a key=value format or a key format:

key=value specifies key as the ODS table name, path, or partial pathname, and specifies value as the CAS output table name.

key specifies key as the ODS table name and also as the CAS output table name.

The ODS table names that you can specify are listed in the section “ODS Table Names” on page 865. You cannot specify the ODS table named OutputCasTables in the table-spec-list.

Table names and partial pathnames are discussed under the DISPLAY statement. The DISPLAYOUT statement does not support regular expressions.

You can specify the following options after a slash (/):

INCLUDEALL creates output CAS tables for all display tables. The name of the created output CAS table is the same as the corresponding display table name. If you specify this option, the table-spec-list specification is ignored.

NOREPLACE does not replace any existing CAS output table of the same name.

REPEATED replicates all CAS output tables on all nodes.

EFFECT Statement

EFFECT name=effect-type (variables < / options >) ;

The EFFECT statement enables you to construct special collections of columns for design matrices. These collections are referred to as constructed effects to distinguish them from the usual model effects that are formed from continuous or classification variables, as discussed in the section “GLM Parameterization of Classification Variables and Effects” on page 54 in Chapter 3, “Shared Concepts.”

You can specify the following effect-types:

COLLECTION specifies a collection effect that defines one or more variables as a single effect that has multiple degrees of freedom. The variables in a collection are considered as a unit for purposes of estimation and inference.
**MULTIMEMBER | MM** specifies a multimember classification effect whose levels are determined by one or more variables that appear in a CLASS statement.

**POLYNOMIAL | POLY** specifies a multivariate polynomial effect in the specified numeric variables.

**SPLINE** specifies a regression spline effect whose columns are univariate spline expansions of one or more variables. A spline expansion replaces the original variable with an expanded or larger set of new variables.

Table 18.4 summarizes the *options* available in the EFFECT statement.

### Table 18.4 EFFECT Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Collection Effects Options</strong></td>
<td></td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays the constituents of the collection effect</td>
</tr>
<tr>
<td><strong>Multimember Effects Options</strong></td>
<td></td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays the levels of the multimember effect</td>
</tr>
<tr>
<td>NOEFFECT</td>
<td>Specifies that observations whose levels are all missing for the multimember variables should have 0 values in the corresponding design matrix columns</td>
</tr>
<tr>
<td>STDIZE</td>
<td>Standardizes the design matrix entries so that each observation has a sum of 1</td>
</tr>
<tr>
<td>WEIGHT=</td>
<td>Specifies the weight variable for the contributions of each classification effect</td>
</tr>
<tr>
<td><strong>Polynomial Effects Options</strong></td>
<td></td>
</tr>
<tr>
<td>DEGREE=</td>
<td>Specifies the degree of the polynomial</td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays details of the specified polynomial</td>
</tr>
<tr>
<td>MDEGREE=</td>
<td>Specifies the maximum degree of any variable in a term of the polynomial</td>
</tr>
<tr>
<td>NOSEPARATE</td>
<td>Treats the polynomial as a single effect with multiple degrees of freedom</td>
</tr>
<tr>
<td>STANDARDIZE=</td>
<td>Specifies centering and scaling suboptions for the variables that define the polynomial</td>
</tr>
<tr>
<td><strong>Spline Effects Options</strong></td>
<td></td>
</tr>
<tr>
<td>BASIS=</td>
<td>Specifies the type of basis (B-spline basis or truncated power function basis) for the spline effect</td>
</tr>
<tr>
<td>DATABOUNDARY</td>
<td>Uses the extremes of the data as boundary knots for a B-spline basis</td>
</tr>
<tr>
<td>DEGREE=</td>
<td>Specifies the degree of the spline effect</td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays the knots and locations for each spline basis function</td>
</tr>
<tr>
<td>KNOTMAX=</td>
<td>Requests equally spaced right-side boundary knots starting at the variables’ maximum and ending at the KNOTMAX= value</td>
</tr>
<tr>
<td>KNOTMETHOD=</td>
<td>Specifies how to construct the knots for the spline effect</td>
</tr>
<tr>
<td>KNOTMIN=</td>
<td>Requests equally spaced left-side boundary knots starting at the KNOTMIN= value and ending at the variables’ minimum value</td>
</tr>
<tr>
<td>NATURALLCUBIC</td>
<td>Specifies a natural cubic spline basis for the spline effect</td>
</tr>
</tbody>
</table>
Table 18.4  continued

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEPARATE</td>
<td>Treats the spline basis for each variable as a separate effect when multiple variables are specified</td>
</tr>
<tr>
<td>SPLIT</td>
<td>Treats each design matrix column as a separate effect for selection methods</td>
</tr>
</tbody>
</table>

For more information about the syntax of these **effect-types** and how columns of constructed effects are computed, see the section “EFFECT Statement” on page 21 in Chapter 3, “Shared Concepts.”

**MODEL Statement**

```latex
MODEL dependent=effects / model-options;
```

The MODEL statement names the dependent variable and the explanatory effects, including covariates, main effects, interactions, and nested effects. If you omit the explanatory effects, the procedure fits an intercept-only model.

After the keyword MODEL, the dependent (response) variable is specified, followed by an equal sign. The explanatory effects follow the equal sign.

For information about constructing the model effects, see the section “Specification and Parameterization of Model Effects” on page 51 in Chapter 3, “Shared Concepts.”

The **model-options** control other aspects of model formation and inference. Table 18.5 summarizes these options.

Table 18.5  MODEL Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLB</td>
<td>Requests confidence limits</td>
</tr>
<tr>
<td>INCLUDE=</td>
<td>Includes effects in all models for model selection</td>
</tr>
<tr>
<td>INFORMATIVE</td>
<td>Models missing values by using extra indicator variables</td>
</tr>
<tr>
<td>NOINT</td>
<td>Suppresses the intercept</td>
</tr>
<tr>
<td>QUANTILES</td>
<td>Specifies quantile levels for the quantile regression</td>
</tr>
<tr>
<td>START=</td>
<td>Includes effects in the initial model for model selection</td>
</tr>
<tr>
<td>STB</td>
<td>Displays standardized estimates</td>
</tr>
</tbody>
</table>

**Model Options**

You can specify the following **options** in the MODEL statement after a slash (/):
requests the $100(1 - \alpha)\%$ upper and lower confidence limits for the parameter estimates. By default, the $95\%$ limits are computed; you can use the ALPHAV= option in the PROC QTRSELECT statement to change the $\alpha$ level.

`INCLUDE=n`  
`INCLUDE=single-effect`  
`INCLUDE=(effects)`  
forces effects to be included in all models. If you specify `INCLUDE=n`, then the first $n$ effects listed in the `MODEL` statement are included in all models. If you specify `INCLUDE=single-effect` or `INCLUDE=(single-effect)`, then the specified effects are forced into all models.

`INFORMATIVE`  
models missing values by using extra model effects. These effects consist of dummy variables that take the value 1 when the value of a continuous model variable involved in the effect is missing, and take the value 0 otherwise. The missing value in the original model effect is replaced by the average value of the effect for the nonmissing values. For continuous-by-class effects, such as $A*x$, where $A$ is a classification variable and $x$ is a continuous variable, informative missingness creates multiple dummy columns and substitutes the effect mean of $x$ that corresponds to the respective level of $A$. Missing values for classification variables are treated as valid levels. For more information about informative missingness, see the section “Informative Missingness” on page 78 in Chapter 3, “Shared Concepts.”

`NOINT`  
suppresses the intercept term that is otherwise included in the model.

`QUANTILES<(quantile-level-options)>=<number-list>`  
`QUANTILE<(quantile-level-options)>=<number-list>`  
`Q<(quantile-level-options)>=<number-list>`  
specifies the quantile levels for the quantile regression. You can request any values of quantile levels in $(0, 1)$ by specifying a `number-list`. By default, the QTRSELECT procedure uses `QUANTILES=0.5`, which corresponds to median regression.

You can also specify the following `quantile-level-options`:

`NTAU=n`  
`NQ=n`  
specifies the following $n$ quantile levels for the quantile regression:

$$\left\{ \frac{1}{n + 1}, \ldots, \frac{n}{n + 1} \right\}$$

If you specify both a `QUANTILES=number-list` and `NTAU=n`, the QTRSELECT procedure uses all the specified quantile levels.

`SORT`  
sorts all the specified quantile levels in ascending order. This option affects the order of the prediction variables in the SAS DATA step code that is created by the CODE statement, and the order of the appropriate keyword variables in the SAS data set that is created by the OUTPUT statement.
START=n
START=single-effect
START= (effects)

begins the selection process in the FORWARD and STEPWISE selection methods from the initial model that you designate. If you specify START=n, then the starting model consists of the first n effects listed in the MODEL statement. If you specify START=single-effect or START= (single-effect), then the starting model consists of these specified effects.

STB

produces standardized regression coefficients. A standardized regression coefficient is computed by dividing a parameter estimate by the ratio of the sample standard deviation of the dependent variable to the sample standard deviation of the regressor. If you use the INCLUDE= option to force some effects to be in the model, then the QTRSELECT procedure computes the sample standard deviation against all the effects that are forced in, as follows. Let $X_1$ denote the design submatrix of $p_1$ regressors that consists of all the effects that are forced in, and let $z$ denote the dependent variable or any regressor. Then the sample standard deviation of $z$ is computed as

$$s_z = \sqrt{\frac{z' [I - X_1 (X_1' X_1)^{-1} X_1'] z}{n - p_1}}$$

**OUTPUT Statement**

```plaintext
OUTPUT OUT=CAS-libref.data-table
   <COPYVARS=(variables)>
   <keyword =name> . . . <keyword =name> ;
```

The OUTPUT statement creates a data table that contains observationwise statistics, which are computed after the model is fitted. The variables in the input data table are not included in the output data table, in order to avoid data duplication for large data tables; however, variables that you specify in the COPYVARS= option are included.

The output statistics are computed based on the parameter estimates for the selected model. If you specify multiple quantile levels by using the QUANTILES option in the MODEL statement, then for each appropriate keyword that is specified in the OUTPUT statement, one variable is generated for each specified quantile level. For observations in which only the response variable is missing, predicted values are computed even though these observations do not affect the model fit. This enables, for example, predicted values to be computed for new observations.

You must specify the following option:

```
OUT=CAS-libref.data-table
```

names the output data table for PROC QTRSELECT to use. You must specify this option before any other options. `CAS-libref.data-table` is a two-level name, where

- **CAS-libref** refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to where the data table is to be stored, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about **CAS-libref**, see the section “Using CAS Sessions and CAS Engine Librefs” on page 830.
**data-table** specifies the name of the output data table.

You can also specify the following syntax elements:

**COPYVAR=variable**
**COPYVARS=(variables)**
transfers one or more *variables* from the input data table to the output data table.

**keyword <=name >**
specifies the statistics to include in the output data table and optionally names the new variables that contain the statistics. Specify a keyword for each desired statistic (see the following list of keywords), followed optionally by an equal sign and a variable to contain the statistic.

If you specify **keyword=name**, the new variable that contains the requested statistic has the specified name. If you omit the optional =name after a **keyword**, then a default name is used.

You can specify the following values for **keyword** to request statistics that are available with all selection methods:

**PREDICTED<=name >**
**PRED<=name >**
**P<=name >**
requests predicted values for the response variable. The default name is Pred.

**RESIDUAL<=name >**
**RESID<=name >**
**R<=name >**
requests the residual, calculated as ACTUAL – PREDICTED. The default name is Residual.

**ROLE<=name >**
requests a numeric variable that indicates the role played by each observation in fitting the model. The default name is _ROLE_. For each observation, the interpretation of this variable is shown in Table 18.6.

### Table 18.6 Role Interpretation

<table>
<thead>
<tr>
<th>Value</th>
<th>Observation Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Not used</td>
</tr>
<tr>
<td>1</td>
<td>Training</td>
</tr>
<tr>
<td>2</td>
<td>Validation</td>
</tr>
<tr>
<td>3</td>
<td>Testing</td>
</tr>
</tbody>
</table>

If you do not partition the input data by using a **PARTITION** statement, then the role variable value is 1 for observations used in fitting the model, and 0 for observations that have at least one missing or invalid value for the response, regressor, frequency, or weight variables.

If you specify multiple quantile levels by using the **QUANTILES** option in the **MODEL** statement, then for each appropriate keyword that is specified in the **OUTPUT** statement, one variable is generated for each specified quantile level. These variables appear in the same order as the specified quantile levels. For
example, the following statements generate the mycas.out data table, which contains the two predicted quantile variables p1 and p2:

```sas
proc qtrselect data=mycas.one;
  model y = x1-x4 /quantiles=0.5 0.3;
  output out=mycas.out pred=p;
run;
```

The variable p1 is for quantile level 0.5, and the variable p2 is for quantile level 0.3.

By using the SORT suboption in the QUANTILES option, the following statements generate the mycas.out data table in sorted order:

```sas
proc qtrselect data=mycas.one;
  model y = x1-x4 /quantiles(sort)=0.5 0.3;
  output out=mycas.out pred=p;
run;
```

The variable p1 is for quantile level 0.3, and the variable p2 is for quantile level 0.5, because the sorted quantile levels are (0.3 0.5).

In addition to the preceding statistics, you can also use the *keywords* listed in Table 18.7 in the OUTPUT statement to obtain additional statistics. For computational formulas, see the section “Diagnostic Statistics” on page 856. All the statistics available in the OUTPUT statement are conditional on the selected model and do not take into account the variability introduced when you do model selection.

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCLM</td>
<td>Lower bound of a $100(1 - \alpha)%$ confidence interval for the quantile prediction variable</td>
</tr>
<tr>
<td>STDP</td>
<td>Standard error of the quantile prediction variable</td>
</tr>
<tr>
<td>UCLM</td>
<td>Upper bound of a $100(1 - \alpha)%$ confidence interval for the quantile prediction variable</td>
</tr>
</tbody>
</table>

**PARTITION Statement**

```
PARTITION partition-option ;
```

The PARTITION statement specifies how observations in the input data set are logically partitioned into disjoint subsets for model training, validation, and testing. For more information, see the section “Using Validation and Test Data” on page 80 in Chapter 3, “Shared Concepts.” Either you can designate a variable in the input data table and a set of formatted values of that variable to determine the role of each observation, or you can specify proportions to use for randomly assigning observations to each role.

You must specify exactly one of the following *partition-options*:
FRACTION(< TEST=fraction > < VALIDATE=fraction > < SEED=number >) randomly assigns specified proportions of the observations in the input data table to the roles. You specify the proportions for testing and validation by using the TEST= and VALIDATE= suboptions. If you specify both the TEST= and VALIDATE= suboptions, then the sum of the specified fractions must be less than 1 and the remaining fraction of the observations are assigned to the training role. The SEED= option specifies an integer that is used to start the pseudorandom number generator for random partitioning of data for training, testing, and validation. If you do not specify SEED=number or if number is less than or equal to 0, the seed is generated by reading the time of day from the computer’s clock.

ROLE=variable (< TEST='value' > < TRAIN='value' > < VALIDATE='value' >) ROLEVAR=variable (< TEST='value' > < TRAIN='value' > < VALIDATE='value' >) names the variable in the input data table whose values are used to assign roles to each observation. This variable cannot also appear as an analysis variable in other statements or options. The TEST=, TRAIN=, and VALIDATE= suboptions specify the formatted values of this variable that are used to assign observation roles. If you do not specify the TRAIN= suboption, then all observations whose role is not determined by the TEST= or VALIDATE= suboption are assigned to the training role.

**SELECTION Statement**

```
SELECTION < METHOD=method < (method-options) > ><options> ;
```

The SELECTION statement performs model selection by examining whether effects should be added to or removed from the model according to rules that are defined by model selection methods. The statement is fully documented in the section “SELECTION Statement” on page 36 in Chapter 3, “Shared Concepts.”

The QTRSELECT procedure supports the following values of the METHOD= option in the SELECTION statement:

- **BACKWARD** specifies the backward elimination method, which starts with all effects in the model and deletes effects.
- **FORWARD** specifies the forward selection method, which starts with no effects in the model and adds effects.
- **NONE** specifies no model selection.
- **STEPWISE** specifies the stepwise regression method, which is similar to the forward selection method except that effects already in the model do not necessarily stay there.

By default, METHOD=STEPWISE.

The DETAILS=ALL and DETAILS=STEPS options produce the “Fit Statistics,” and “Parameter Estimates” tables, which provide information about the model that is selected at each step of the selection process.

By default, the QTRSELECT procedure displays the stop horizon steps in the “Selection Summary” table, and uses these steps for choosing the final model of the model selection process. Therefore, it is possible that the final model occurs after the step that has the best stop criterion value. To exclude the stop horizon steps for choosing the final model, you can specify the following option:
HIDESTOPSTEPS

HSS

hides the stop horizon steps in the “Selection Summary” table, and excludes these steps in choosing the final model.

If you specify the PLOTS=CRITERIA or PLOTS=ALL option, then a plot of the fit criteria by the selection step is created for the AIC, AICC, SBC, R1, and adjusted R1 statistics. If you also specify a PARTITION statement, then the same type of plot is created for the ACL criteria for training, validation, and testing roles. If you specify the PLOTS=FITBYROLE option and a PARTITION statement, then a plot of the ACL criterion by the selection step for each role is created.

WEIGHT Statement

WEIGHT variable;

The variable in the WEIGHT statement is used as a weight to perform a weighted analysis of the data. Observations that have nonpositive or missing weights are not included in the analysis. If a WEIGHT statement is not included, all observations that are used in the analysis are assigned a weight of 1.

The QTRSELECT procedure uses each valid weight as the scale factor of its relevant observation. Let $X$ denote the design matrix before weights are used, and let $W$ denote the diagonal matrix whose diagonal elements are the weights. Then PROC QTRSELECT uses the weighted design matrix $WX$ for its computation. For example, the weighted version of $X'X$ is $X'WX$ but not $X'WX$.

Details: QTRSELECT Procedure

Quantile Regression

This section describes the basic concepts and notations for quantile regression and quantile regression model selection.

Let $\{(y_i, x_i) : i = 1, \ldots, n\}$ denote a data set of observations, where $y_i$ are responses and $x_i$ are regressors. Koenker and Bassett (1978) define the regression quantile at quantile level $\tau \in (0, 1)$ as any solution to the minimization problem

$$
\min_{\beta \in \mathbb{R}^p} \sum_{i=1}^{n} \rho_{\tau}(y_i - x_i' \beta)
$$

where $\rho_{\tau}(r) = \tau r^+ + (1 - \tau)r^-$ is a check loss function in which $r^+ = \max(r, 0)$ and $r^- = \max(-r, 0)$. If you specify weights $w_i, i = 1, \ldots, n$, in the WEIGHT statement, then weighted quantile regression is carried out by solving

$$
\min_{\beta \in \mathbb{R}^p} \sum_{i=1}^{n} \rho_{\tau}(w_i (y_i - x_i' \beta))
$$
The QTRSELECT procedure fits a quantile regression model by using a predictor-corrector interior point algorithm, which was originally designed to solve support vector machine classifiers for large data sets (Gertz and Griffin 2005, 2010).

Linear Model with Independent and Identically Distributed Errors

A linear model with independent and identically distributed (iid) errors assumes that the distribution of the response \( Y_i \) conditional on \( x_i \) follows the linear model,

\[
Y_i = x_i'\beta + \epsilon_i
\]

where \( \epsilon_i \) for \( i = 1, \ldots, n \) are iid in the distribution function \( F \). General linear regression uses this model for deriving its statistical inferences and model selection criteria. Although quantile regression can analyze heterogeneous data without the iid errors assumption, this model is helpful for deriving some quantile regression model selection criteria. For more information about using a linear model with iid errors in deriving the quasi-likelihood information criteria, see the section “Criteria Used in Model Selection” on page 853.

Linear-in-Parameter Model with Non-iid Settings

The more general form of a linear quantile regression model is

\[
Q_Y(\tau|x) = x'\beta(\tau)
\]

where the iid assumption is not necessary.

Let \( s_i(\tau) = 1/f_i(F_i^{-1}(\tau)) \) denote the sparsity function of the \( i \)th observation. Under some regularity conditions, the asymptotic form of the general form of quantile regression estimates is

\[
\sqrt{n}(\hat{\beta}(\tau) - \beta(\tau)) \rightarrow N(0, \tau(1 - \tau)H^{-1}\Omega H^{-1})
\]

where \( H = \lim_{n \to \infty} n^{-1} \sum x_i x_i' / s_i(\tau) \) and \( \Omega = \lim_{n \to \infty} n^{-1} \sum x_i x_i' \).

The sparsity function of the \( i \)th observation, \( s_i(\tau) \), can be estimated as

\[
\hat{s}_i(\tau) = \frac{\hat{F}_i^{-1}(\tau + h_n) - \hat{F}_i^{-1}(\tau - h_n)}{2h_n}
\]

where \( \hat{F}_i^{-1}(\tau \pm h_n) = x_i'\hat{\beta}(\tau \pm h_n) \) are the quantile predictions of the \( i \)th observation at quantile levels \( (\tau \pm h_n) \).

Accordingly, the covariance matrix of \( \hat{\beta}(\tau) \) can be estimated as

\[
\text{COV}(\hat{\beta}(\tau)) = n^{-1}(1 - \tau)\hat{H}_n^{-1}\hat{\Omega}_n \hat{H}_n^{-1}
\]

where \( \hat{H}_n = n^{-1} \sum (x_i x_i' / \hat{s}_i(\tau)) \) and \( \hat{\Omega}_n = n^{-1} \sum x_i x_i' \).

You can specify the bandwidth method for computing \( h_n \) by using either the COV=SPARSITY(BF) option for the Bofinger bandwidth method or the COV=SPARSITY(HF) option for the Hall-Sheather bandwidth method:

- The Bofinger bandwidth optimizes the mean square error for standard density estimation:

\[
h_n = n^{-1/5} \left( 4.5 \nu^2(\tau) \right)^{1/5}
\]
The Hall-Sheather bandwidth is based on Edgeworth expansions for studentized quantiles:

\[ h_n = n^{-1/3} z_\alpha 2^{2/3} (1.5v(\tau))^{1/3} \]

\( z_\alpha \) satisfies \( T(z_\alpha, df) = 1 - \alpha/2 \) for the construction of \( 1 - \alpha \) confidence intervals, where \( T \) is the cumulative distribution function for the \( t \) distribution and \( df \) is the residual degrees of freedom.

The quantity

\[ v(\tau) = \frac{s(\tau)}{s^{(2)}(\tau)} = \frac{f^2}{2(f'(f)/f)^2 + [(f'(f)/f)^2 - f''(f)/f]} \]

is not sensitive to \( f \) and can be estimated by assuming \( f \) is Gaussian as

\[ \hat{v}(\tau) = \frac{\exp(-q^2)}{2\pi(q^2 + 1)} \]

where \( q = \Phi^{-1}(\tau) \).

**More Statistics for Parameter Estimates**

Let \( \hat{\beta}_j(\tau) \) denote the \( j \)th parameter estimate, and let \( \hat{\beta}(\tau) = \left( \hat{\beta}_1(\tau), \ldots, \hat{\beta}_p(\tau) \right) \) denote the vector of the parameter estimates.

PROC QTRSELECT outputs the standard error, \( t \) value, and \( \Pr > |t| \) probability for each \( \hat{\beta}_j(\tau) \) in the parameter estimates table. If you specify the CLB option in the MODEL statement, PROC QTRSELECT also outputs confidence limits in the parameter estimates table. Table 18.8 summarizes these statistics for \( \hat{\beta}_j(\tau) \).

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard error: ( \hat{\sigma}_j )</td>
<td>( \sqrt{\text{COV} \left( \hat{\beta}(\tau) \right)_{jj}} )</td>
</tr>
<tr>
<td>( (1 - \alpha)% ) confidence limits</td>
<td>( \hat{\beta}<em>j(\tau) \pm t</em>{1,1-\frac{\alpha}{2}} \hat{\sigma}_j )</td>
</tr>
<tr>
<td>( t ) value</td>
<td>( \hat{\beta}_j(\tau)/\hat{\sigma}_j )</td>
</tr>
<tr>
<td>( \Pr &gt;</td>
<td>t</td>
</tr>
</tbody>
</table>

Here \( \text{COV} \left( \hat{\beta}(\tau) \right)_{jj} \) is the \( (j, j) \) element of \( \text{COV} \left( \hat{\beta}(\tau) \right) \), and \( t_{1,1-\frac{\alpha}{2}} \) denotes the \( (1 - \frac{\alpha}{2}) \)-level student’s \( t \) score with 1 degree of freedom.

**Criteria Used in Model Selection**

The QTRSELECT procedure supports the following fit statistics that you can use as criteria for the CHOOSE=, SELECT=, and STOP= options in the SELECTION statement:
ADJR1 specifies the adjusted R1 statistic.

AIC specifies Akaike’s information criterion (Akaike 1969; Koenker 2005).

AICC specifies the corrected Akaike’s information criterion (Hurvich and Tsai 1989).

BIC | SBC specifies the Schwarz Bayesian information criterion (Schwarz 1978; Koenker 2005).

R1 specifies the R1 statistic (Koenker and Machado 1999). The R1 statistic is not valid for the STOP= or CHOOSE= option.

SL specifies the significance level that is used to assess an effect’s contribution to the fit when it is added to or removed from a model. SL is not valid for the CHOOSE= option.

VALIDATE specifies the average check loss over the validation data.

Table 18.9 provides formulas and definitions for these fit statistics.

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Definition or Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>Number of observations</td>
</tr>
<tr>
<td>p</td>
<td>Number of parameters, including the intercept</td>
</tr>
<tr>
<td>r_i(τ)</td>
<td>Residual for the ith observation; ( r_i(τ) = y_i - x_i' \hat{β}(τ) )</td>
</tr>
<tr>
<td>D(τ)</td>
<td>Total sum of check losses; ( D(τ) = \sum_{i=1}^{n} ρ_τ(r_i) ). D(τ) is labeled as Objective Function in the “Fit Statistics” table.</td>
</tr>
<tr>
<td>D_0(τ)</td>
<td>Total sum of check losses for intercept-only model if the intercept is a forced-in effect; otherwise for empty model.</td>
</tr>
<tr>
<td>ACL(τ)</td>
<td>Average check loss; ( ACL(τ) = \frac{n}{n}D(τ) )</td>
</tr>
<tr>
<td>R1(τ)</td>
<td>Counterpart of linear regression R square for quantile regression; ( R1(τ) = 1 - \frac{D(τ)}{D_0(τ)} ) if intercept is a forced-in effect; otherwise ( 1 - \frac{D(τ)}{D_0(τ)} ).</td>
</tr>
<tr>
<td>ADJR1(τ)</td>
<td>Adjusted R1; ( 1 - \frac{(n-1)D(τ)}{(n-p)D_0(τ)} ) if intercept is a forced-in effect; otherwise ( 1 - \frac{D(τ)}{nD_0(τ)} ).</td>
</tr>
<tr>
<td>AIC(τ)</td>
<td>( 2n \ln (ACL(τ)) + 2p )</td>
</tr>
<tr>
<td>AICC(τ)</td>
<td>( 2n \ln (ACL(τ)) + \frac{2pn}{n-p-1} )</td>
</tr>
<tr>
<td>SBC(τ)</td>
<td>( 2n \ln (ACL(τ)) + p \ln(n) )</td>
</tr>
</tbody>
</table>

The ADJR1(τ) criterion is equivalent to the generalized approximate cross validation (GACV) criterion for quantile regression (Yuan 2006). The GACV criterion is defined as

\[
GACV(τ) = \frac{D(τ)}{(n - p)}
\]

which is proportional to \( 1 - ADJR1(τ) \).
Quasi-likelihood Information Criteria

Given the quantile level $\tau$, assume that the distribution of $Y_i$ conditional on $x_i$ follows the linear model

$$Y_i = x_i' \beta + \epsilon_i$$

where $\epsilon_i$ for $i = 1, \ldots, n$ are iid in distribution $F$. Further assume that $F$ is an asymmetric Laplace distribution whose density function is

$$f_\tau(r) = \frac{\tau(1 - \tau)}{\sigma} \exp\left(-\frac{\rho_\tau(r)}{\sigma}\right)$$

where $\sigma$ is the scale parameter. Then, the negative log-likelihood function is

$$l_\tau(\beta, \sigma) = n \log(\sigma) + \sigma^{-1} \sum_{i=1}^{n} \rho_\tau(y_i - x_i' \beta) - n \log(\tau(1 - \tau))$$

Under these settings, the maximum likelihood estimate (MLE) of $\beta$ is the same as the relevant level-$\tau$ quantile regression solution $\hat{\beta}(\tau)$, and the MLE for $\sigma$ is

$$\hat{\sigma}(\tau) = n^{-1} \sum_{i=1}^{n} \rho_\tau(y_i - x_i' \hat{\beta}(\tau))$$

where $\hat{\sigma}(\tau)$ equals the level-$\tau$ average check loss $ACL(\tau)$ for the quantile regression solution.

Because the general form of Akaike’s information criterion (AIC) is $AIC = -2l + 2p$, the quasi-likelihood AIC for quantile regression is

$$AIC(\tau) = 2n \ln(ACL(\tau)) + 2p$$

where $p$ is the degrees of freedom for the fitted model.

Similarly, the quasi-likelihood AICC (corrected AIC) and SBC (Schwarz Bayesian information criterion) can be formulated as follows:

$$AICC(\tau) = 2n \ln(ACL(\tau)) + \frac{2pn}{n - p - 1}$$

$$SBC(\tau) = 2n \ln(ACL(\tau)) + p \ln(n)$$

In fact, the quasi-likelihood AIC, AICC, and SBC are fairly robust, and you can use them to select effects for data sets without the iid assumption in asymmetric Laplace distribution. For a simulation study that applies SBC for effect selection, see “Example 18.1: Simulation Study” on page 867. The study generates a data table by using a naive instrumental model (Chernozhukov and Hansen 2008).

Statistical Tests for Significance Level

The QTRSELECT procedure supports the significance level (SL) criterion for effect selection. Consider the general form of a quantile regression model:

$$Q_Y(\tau | x_1, x_2) = x_1' \beta_1(\tau) + x_2' \beta_2(\tau)$$

At each step of an effect-selection process, a candidate effect can be represented as $x_2$, and the significance level of the candidate effect can be calculated by testing the null hypothesis: $H_0 : \beta_2(\tau) = 0$. 
The QTRSELECT procedure supports the Wald test for computing candidate significance levels. Let \( \hat{\beta}(\tau) = (\hat{\beta}_1(\tau), \hat{\beta}_2(\tau))' \) be the parameter estimates for the extended model, and denote the estimated covariance matrix of \( \hat{\beta}(\tau) \) as

\[
\hat{\Sigma}(\tau) = \begin{bmatrix}
\hat{\Sigma}_{11}(\tau) & \hat{\Sigma}_{12}(\tau) \\
\hat{\Sigma}_{21}(\tau) & \hat{\Sigma}_{22}(\tau)
\end{bmatrix}
\]

where \( \hat{\Sigma}_{22}(\tau) \) is the covariance matrix for \( \hat{\beta}_2(\tau) \). Then the Wald test score is defined as

\[
\hat{\beta}_2(\tau) \hat{\Sigma}_{22}^{-1}(\tau) \hat{\beta}_2(\tau)
\]

Under the null hypothesis that the reduced model is the true model, the Wald score follows a \( \chi^2 \) distribution with degrees of freedom \( df = df_2 - df_1 \), where \( df_1 \) and \( df_2 \) are the degrees of freedom for the reduced model and the extended model, respectively.

When you use SL as a criterion for effect selection, the algorithm for estimating sparsity function depends on whether an effect is being considered as an add or a drop candidate. For testing an add candidate effect, the sparsity values, \( \hat{s}_i(\tau)'s \), are estimated by using the reduced model that does not include the add candidate effect. For testing a drop candidate effect, the sparsity values are estimated on the extended model that does not exclude the drop candidate effect. Then, these estimated sparsity function values are used to compute the covariance matrix of the parameter estimates for the extended model. However, for the model that is selected at each step, the sparsity function for estimating standard errors and confidence limits of the parameter estimates is estimated on that model itself, but not on the model that was selected at the preceding step.

Because the null hypotheses usually do not hold, the SLENTRY and SLSTAY values cannot reliably be viewed as probabilities. One way to address this difficulty is to replace hypothesis testing with a means of selecting a model with information criteria or out-of-sample prediction criteria.

**Diagnostic Statistics**

This section gathers the formulas for the statistics available in the OUTPUT statement. All the statistics available in the OUTPUT statement are conditional on the selected model and do not take into account the variability that is introduced by doing model selection.

The model to be fit is \( Q_Y(\tau|x) = x'\beta(\tau) \), and the parameter estimate \( \hat{\beta}(\tau) \) is the solution that minimizes \( \sum_{i=1}^{n} \rho_\tau (y_i - x_i'\beta) \). The subscript \( i \) is for the \( i \)th observation. The subscript \( j \) is for the \( j \)th quantile level of the specified QUANTILES levels in the MODEL statement. \( \hat{\Sigma}(\tau) \) denotes the covariance estimation for \( \hat{\beta}(\tau) \).

The ALPHA= option in the PROC QTRSELECT statement sets the \( \alpha \) value for the confidence limit statistics. The degrees of freedom for \( t_2 \) are \( n - p \).

Table 18.10 contains the diagnostic statistics and their formulas. Each statistic is computed for each observation.
### Classification Variables and the SPLIT Option

PROC QTRSELECT supports the ability to split classification variables when you are doing model selection. You use the SPLIT option in the CLASS statement to specify that the columns of the design matrix that correspond to effects that contain a split classification variable can enter or leave a model independently of the other design columns of that effect. The following statements illustrate the use of the SPLIT option:

```
data mycas.splitExample;
  length c2 $6;
  drop i;
  do i=1 to 1000;
    c1 = 1 + mod(i, 6);
    if i < 200 then c2 = 'low';
    else if i < 500 then c2 = 'medium';
    else c2 = 'high';
    x1 = ranuni(1);
    x2 = ranuni(1);
    s = x1 + 3*(c2='low') + 10*(c1=3) + 5*(c1=5) + rannor(1);
    y = x1 + s + rannor(1);
    output;
  end;
run;
```

```
proc qtrselect data=mycas.splitExample;
  class c1(split) c2(order=freq);
  model y = c1 c2 x1 x2;
  selection method=forward;
run;
```

The “Class Level Information” table shown in Figure 18.12 is produced by default whenever you specify a CLASS statement.

### Table 18.10 Formulas and Definitions for Diagnostic Statistics

<table>
<thead>
<tr>
<th>MODEL Option or Statistic</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRED&lt;sub&gt;j&lt;/sub&gt;</td>
<td>( \hat{y}_{ji} = x'_i \hat{\beta}(\tau_j) )</td>
</tr>
<tr>
<td>RES&lt;sub&gt;j&lt;/sub&gt;</td>
<td>( y_i - \hat{y}_{ji} )</td>
</tr>
<tr>
<td>STDP&lt;sub&gt;j&lt;/sub&gt;</td>
<td>( \sqrt{x'_i \hat{\Sigma}(\tau_j) x_i} )</td>
</tr>
<tr>
<td>LCLM&lt;sub&gt;j&lt;/sub&gt;</td>
<td>( \hat{y}<em>{ji} - t</em>{\alpha/2} \text{STDP}_{ji} )</td>
</tr>
<tr>
<td>UCLM&lt;sub&gt;j&lt;/sub&gt;</td>
<td>( \hat{y}<em>{ji} + t</em>{\alpha/2} \text{STDP}_{ji} )</td>
</tr>
</tbody>
</table>
The QTRSELECT Procedure

Class Level Information

<table>
<thead>
<tr>
<th>Class</th>
<th>Levels</th>
</tr>
</thead>
<tbody>
<tr>
<td>c1</td>
<td>6 1 2 3 4 5 6</td>
</tr>
<tr>
<td>c2</td>
<td>high medium low</td>
</tr>
</tbody>
</table>

The SPLIT option has been specified for the classification variable c1. This permits the parameters associated with the effect c1 to enter or leave the model individually. The “Parameter Estimates” table in Figure 18.13 shows that for this example the parameters that correspond only to levels 3 and 5 of c1 are in the selected model.

![Figure 18.13 Parameter Estimates](image)

Using Validation and Test Data

When you have sufficient data, you can subdivide your data into three parts called the training, validation, and test data. The selection process fits models to the training data and uses the validation data to find the prediction errors for the models that are obtained in this way. This prediction error on the validation data can be used to decide when to terminate the selection process or to decide what effects to include as the selection process proceeds. Finally, after a selected model has been obtained, the test data can be used to assess how the selected model generalizes on data that played no role in selecting the model.

In some cases you might want to use only training and test data. For example, you might want to use an information criterion to decide what effects to include and when to terminate the selection process. In this case no validation data are required, but test data can still help you assess the predictive performance of the selected model. In other cases you might decide to use validation data during the selection process but forgo assessing the selected model on test data. Hastie, Tibshirani, and Friedman (2001) note that it is difficult to give a general rule for how many observations you should assign to each role. They state that a typical split might be 50% for training and 25% each for validation and testing.

You use a PARTITION statement to logically subdivide the DATA= data table into separate roles. You can name the fractions of the data that you want to reserve as test data and validation data. For example, the following statements randomly subdivide the mycas.inData data table, reserving 50% for training and 25% each for validation and testing:
**Using Validation and Test Data**

```latex
proc qtrselect data=mycas.inData;
  partition fraction(test=0.25 validate=0.25);
  ...
run;
```

In some cases you might need to exercise more control over the partitioning of the input data table. You can do this by naming both a variable in the input data table and a formatted value of that variable that correspond to each role. For example, the following statements assign roles to the observations in the mycas.inData data table based on the value of the variable in that data table. Observations in which the value of group is “group 1” are assigned to testing, and those whose value is “group 2” are assigned to training. All other observations are ignored.

```latex
proc qtrselect data=mycas.inData;
  partition roleVar=group(test='group 1' train='group 2')
  ...
run;
```

After you reserve observations for training, validation, and testing, a model fit of the training data is scored on the validation and test data, and the average check loss (ACL) is computed separately for each of these subsets. The ACL for each data role is the sum of check losses for observations in that role divided by the number of observations in that role.

The following statements illustrate the use of the ROLEVAR= option:

```latex
%let seed=321;
%let n=600;
%let p=10;

data mycas.roleExample;
  array x{&p} x1-x&p;
  length r $8;
  drop i j k;

  do i=1 to &n;
    do j=1 to &p;
      x{j} = ranuni(&seed);
    end;
    y = x1 + x2 + x3 + ranuni(&seed);
    k = mod(i,3);
    if k=0 then r = 'train';
    else if k=1 then r = 'validate';
    else if k=2 then r = 'test';
    output;
  end;
run;

data mycas.roleExample;
  array x{&p} x1-x&p;
  length r $8;
  drop i j k;

  do i=1 to &n;
    do j=1 to &p;
      x{j} = ranuni(&seed);
    end;
    y = x1 + x2 + x3 + ranuni(&seed);
    k = mod(i,3);
    if k=0 then r = 'train';
    else if k=1 then r = 'validate';
    else if k=2 then r = 'test';
    output;
  end;
run;

proc qtrselect data=mycas.roleExample;
  model y = x1-x&p;
  selection method=forward(select=validate stop=sbc);
  partition roleVar=r(train='train' validate='validate' test='test');
run;
```
The “Number of Observations” table shown in Figure 18.14 displays the number of observations used for training, validation, and testing.

**Figure 18.14 Number of Observations**

<table>
<thead>
<tr>
<th>The QTRSELECT Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Observations Read</td>
</tr>
<tr>
<td>Number of Observations Used</td>
</tr>
<tr>
<td>Number of Observations Used for Training</td>
</tr>
<tr>
<td>Number of Observations Used for Validation</td>
</tr>
<tr>
<td>Number of Observations Used for Testing</td>
</tr>
</tbody>
</table>

The “Selection Summary” table shown in Figure 18.15 displays the validation ACL values for each step of the selection process.

**Figure 18.15 Selection Summary**

<table>
<thead>
<tr>
<th>Quantile Level = 0.5 Selection Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
</tbody>
</table>

* Optimal Value Of Criterion

The “Fit Statistics” table shown in Figure 18.16 displays the training ACL, the validation ACL, and the testing ACL for the final model.
Using the Validation ACL as the SELECT= Criterion

If you provided observations for validation and specified a model selection method that uses the SELECT= criterion, then you can specify SELECT=VALIDATE as a suboption of the METHOD= option in the SELECTION statement. After each step, the selection process computes the validation ACL values for all the current candidate models. The candidate model that has the smallest validation ACL value usually serves as the model for the next selection step.

Using the Validation ACL as the STOP= Criterion

If you provided observations for validation, then you can specify STOP=VALIDATE as a suboption of the METHOD= option in the SELECTION statement. At step $k$ of the selection process, the best candidate effect to enter or leave the current model is determined. Here, “best candidate” means the effect that gives the best value of the SELECT= criterion; this criterion does not need to be based on the validation data. The validation ACL for the model with this candidate effect added or removed is computed. If this validation ACL is greater than the validation ACL for the model at step $k$, then the selection process terminates at step $k$.

Using the Validation ACL as the CHOOSE= Criterion

When you specify the CHOOSE=VALIDATE suboption of the METHOD= option in the SELECTION statement, the validation ACL is computed for the models at each step of the selection process. The smallest model at any step that yields the smallest validation ACL is selected.

Computational Method

Multithreading

The QTRSELECT procedure allocates data to different threads and calculates crossproduct matrices by accumulating the contributions from all threads. PROC QTRSELECT also uses multiple threads to compute matrix inverses and select candidates during model selection. For more information about how PROC QTRSELECT uses threads, see the section “Multithreading” on page 81 in Chapter 3, “Shared Concepts.”
Displayed Output

The following sections describe the output produced by PROC QTRSELECT. The output is organized into various tables, which are discussed in their order of appearance.

Selection Information

When you specify the SELECTION statement, the QTRSELECT procedure produces by default a series of tables that display information about the model selection. The “Selection Information” table informs you about the model selection method; select, stop, and choose criteria; and other parameters that govern the selection. You can suppress this table by specifying DETAILS=NONE in the SELECTION statement.

Number of Observations

The “Number of Observations” table displays the number of observations read from the input data table and the number of observations used in the analysis. If you use a PARTITION statement, the table also displays the number of observations used for each data role.

Class Level Information

The “Class Level Information” table lists the levels of every variable that you specify in the CLASS statement. You should check this information to make sure that the data are correct. You can adjust the order of the CLASS variable levels by specifying the ORDER= option in the CLASS statement. You can suppress the “Class Level Information” table completely or partially by specifying the NOCLPRINT= option in the PROC QTRSELECT statement.

If the classification variables use a nonsingular parameterization, the “Class Level Information” table also displays the reference value for each variable.

Dimensions

The “Dimensions” table displays the number of effects and the number of parameters from which the selected model is chosen. If you use split classification variables, then this table also displays the number of effects after splitting is taken into account.

Entry and Removal Candidates

When you specify the DETAILS=ALL or DETAILS=STEPS option in the SELECTION statement, the QTRSELECT procedure produces “Entry Candidates” and “Removal Candidates” tables that display the effect names and values of the criterion used to select entering or departing effects at each step of the selection process. The effects are displayed in sorted order from best to worst of the selection criterion.

Selection Summary

When you specify the SELECTION statement, the QTRSELECT procedure produces the “Selection Summary” table, which displays information about the sequence of steps of the selection process. For each step, the effect that was entered or dropped is displayed along with the statistics used to select the effect, stop the selection, and choose the selected model. For all criteria that you can use for model selection, the steps at which the optimal values of these criteria occur are also indicated.
You can suppress the display of the “Selection Summary” table by specifying DETAILS=NONE in the SELECTION statement.

**Stop Reason**

The “Stop Reason” table displays the reason why the selection stopped. To facilitate programmatic use of this table, an integer code is assigned to each reason and is included if you use an ODS OUTPUT statement or a DISPLAYOUT statement to output this table. The reasons and their associated codes follow:

<table>
<thead>
<tr>
<th>Code</th>
<th>Stop Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>All eligible effects are in the model.</td>
</tr>
<tr>
<td>2</td>
<td>All eligible effects have been removed.</td>
</tr>
<tr>
<td>3</td>
<td>Specified maximum number of steps done.</td>
</tr>
<tr>
<td>4</td>
<td>The model contains the specified maximum number of effects.</td>
</tr>
<tr>
<td>5</td>
<td>The model contains the specified minimum number of effects (for backward selection).</td>
</tr>
<tr>
<td>6</td>
<td>The stopping criterion is at a local optimum.</td>
</tr>
<tr>
<td>7</td>
<td>No suitable add or drop candidate could be found.</td>
</tr>
<tr>
<td>8</td>
<td>Adding or dropping any effect does not improve the selection criterion.</td>
</tr>
<tr>
<td>9</td>
<td>No candidate meets the appropriate SLE or SLS significance level.</td>
</tr>
<tr>
<td>10</td>
<td>Stepwise selection is cycling.</td>
</tr>
<tr>
<td>11</td>
<td>The model is an exact fit.</td>
</tr>
<tr>
<td>12</td>
<td>Dropping an effect would result in an empty model.</td>
</tr>
</tbody>
</table>

You can suppress the display of the “Stop Reason” table by specifying DETAILS=NONE in the SELECTION statement.

**Selection Reason**

When you specify the SELECTION statement, the QTRSELECT procedure produces a simple table that explains why the final model was selected.

You can suppress the display of the “Selection Reason” table by specifying DETAILS=NONE in the SELECTION statement.

**Selected Effects**

When you specify the SELECTION statement, the QTRSELECT procedure produces a simple table that lists which effects were selected for the final model.

**Fit Statistics**

The “Fit Statistics” table displays fit statistics for the selected model. The statistics include the following:

- Objective Function, total sum of check losses. Objective Function is denoted as $D(\tau)$ in Table 18.9.

- $R1$, a measure between 0 and 1 that indicates the portion of the (corrected) total check losses attributed to the fit rather than left to residuals. It is calculated as $1 - \frac{D(\tau)}{D_0(\tau)}$. It is the quantile regression counterpart of the linear regression R square.
Chapter 18: The QTRSELECT Procedure

- Adj R1, the adjusted R1, a version of R1 that has been adjusted for degrees of freedom. It is calculated as

\[
\text{Adj R1} = 1 - \frac{n - i}{n - p} (1 - \text{R1})
\]

where \( i = 1 \) if the intercept is forced in and \( i = 0 \) otherwise, \( n \) is the number of observations used to fit the model, and \( p \) is the number of parameters in the model.

- the fit criteria AIC, AICC, and SBC if they are used in the selection process. For the formulas to evaluate these criteria, see Table 18.9.

- the average check loss (ACL) on the training, validation, and test data

You can request the “Fit Statistics” tables for the model at each step of the selection process by specifying the DETAILS= option in the SELECTION statement.

Parameter Estimates

The “Parameter Estimates” table displays the parameters in the selected model and their estimates. The information displayed for each parameter in the selected model includes the following:

- the parameter label that includes the effect name and level information for effects that contain classification variables
- the degrees of freedom (DF) for the parameter. There is one degree of freedom unless the model is not full rank.
- the parameter estimate
- the standard error, which is the estimate of the standard deviation of the parameter estimate
- \( t \) Value, the \( t \) test that the parameter is 0. This is computed as the parameter estimate divided by the standard error.
- the Pr > |t|, the probability that a \( t \) statistic would obtain a greater absolute value than that observed given that the true parameter is 0. This is the two-tailed significance probability.

When you do model selection, these \( p \)-values are usually liberal because they are not adjusted for the fact that the terms in the model have been selected.

You can request a “Parameter Estimates” table for the model at each step of the selection process by specifying the DETAILS= option in the SELECTION statement.

Timing

The “Timing” table displays the amount of time (in seconds) and the percentage of the time that PROC QTRSELECT required to perform different tasks in the analysis.
**OutputCasTables Table**

The OutputCasTables table is a special table that has information about each CAS table that is created during a CAS action execution. The information for each CAS table consists of the CAS table name, the caslib in which the table resides, and the number of columns and rows in the CAS table. Because this table is not a typical ODS table that contains analytical results, you cannot include it in the `table-spec-list` in the `DISPLAYOUT` statement.

---

**ODS Table Names**

Each table that the QTRSELECT procedure creates has a name associated with it. You must use this name to refer to the table when you use ODS statements. These names are listed in Table 18.11.

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Required Statement / Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>Candidates</td>
<td>Swap candidates at step</td>
<td>`SELECTION DETAILS=ALL</td>
</tr>
<tr>
<td>ClassInfo</td>
<td>Level information from the <code>CLASS</code> statement</td>
<td><code>CLASS</code></td>
</tr>
<tr>
<td>Dimensions</td>
<td>Model dimensions</td>
<td>Default output</td>
</tr>
<tr>
<td>EntryCandidates</td>
<td>Candidates for entry at step</td>
<td>`SELECTION DETAILS=ALL</td>
</tr>
<tr>
<td>FitStatistics</td>
<td>Fit statistics</td>
<td>Default output</td>
</tr>
<tr>
<td>ModelInfo</td>
<td>Information about the modeling environment</td>
<td>Default output</td>
</tr>
<tr>
<td>NObs</td>
<td>Number of observations read and used</td>
<td>Default output</td>
</tr>
<tr>
<td>OutputCasTables</td>
<td>See the section “OutputCasTables Table” on page 865</td>
<td><code>OUTPUT DISPLAYOUT</code></td>
</tr>
<tr>
<td>ParameterEstimates</td>
<td>Solutions for the parameter estimates associated with effects in the <code>MODEL</code> statement</td>
<td>Default output</td>
</tr>
<tr>
<td>RemovalCandidates</td>
<td>Candidates for removal at step</td>
<td>`SELECTION DETAILS=ALL</td>
</tr>
<tr>
<td>SelectedEffects</td>
<td>List of selected effects</td>
<td><code>SELECTION</code></td>
</tr>
<tr>
<td>SelectionInfo</td>
<td>Information about selection settings</td>
<td>Default output</td>
</tr>
<tr>
<td>SelectionReason</td>
<td>Reason for selecting the final model</td>
<td><code>SELECTION</code></td>
</tr>
<tr>
<td>SelectionSummary</td>
<td>Summary information about the model selection steps</td>
<td><code>SELECTION</code></td>
</tr>
<tr>
<td>StopReason</td>
<td>Reason selection was terminated</td>
<td><code>SELECTION</code></td>
</tr>
<tr>
<td>Timing</td>
<td>Timing breakdown by task</td>
<td><code>SELECTION DETAILS</code></td>
</tr>
</tbody>
</table>
Chapter 18: The QTRSELECT Procedure

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in the “Statistical Graphics Using ODS” chapter in SAS/STAT User’s Guide.

Before you create graphs, ODS Graphics must be enabled (for example, with the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section “Enabling and Disabling ODS Graphics” in that chapter.

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section “A Primer on ODS Statistical Graphics” in that chapter.

When ODS Graphics is enabled, then the SELECTION statement can produce plots to help evaluate the selection process. For information about these plots, see the section “SELECTION Statement” on page 36 in Chapter 3, “Shared Concepts.”

PROC QTRSELECT assigns a name to each graph that it creates using ODS. You can use these names to reference the graphs when using ODS. The names are listed in Table 18.12.

<table>
<thead>
<tr>
<th>ODS Graph Name</th>
<th>Plot Description</th>
<th>PLOTS Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoefficientPanel</td>
<td>Coefficients and CHOOSE= criterion by step</td>
<td>COEFFICIENTS</td>
</tr>
<tr>
<td>ChooseCriterionPlot</td>
<td>CHOOSE= criterion by step</td>
<td>COEFFICIENTS(UNPACK)</td>
</tr>
<tr>
<td>CoefficientPlot</td>
<td>Coefficients by step</td>
<td>COEFFICIENTS(UNPACK)</td>
</tr>
<tr>
<td>CriterionPanel</td>
<td>Fit criteria by step</td>
<td>CRITERIA</td>
</tr>
<tr>
<td>AdjR1Plot</td>
<td>Adjusted R1 by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>AICCPPlot</td>
<td>Corrected Akaike’s information criterion by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>AICPlot</td>
<td>Akaike’s information criterion by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>SBCPlot</td>
<td>Schwarz Bayesian information criterion by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>TEST_ACLPlot</td>
<td>Average check loss on testing data by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>TRAIN_ACLPlot</td>
<td>Average check loss on training data by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>VAL_ACLPlot</td>
<td>Average check loss on validation data by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>FitByRolePlot</td>
<td>Progression of average check loss by role</td>
<td>FITBYROLE</td>
</tr>
</tbody>
</table>
Example 18.1: Simulation Study

This example is based on the section “Simulation Study” (Examples: QUANTSELECT Procedure, SAS/STAT User’s Guide). This simulation study shows how you can use the forward selection method to select quantile regression models for single quantile levels. The following statements simulate a data set from a naive instrumental model (Chernozhukov and Hansen 2008):

```sas
%let seed=321;
%let p=20;
%let n=3000;

data analysisData;
array x{&p} x1-x&p;
do i=1 to &n;
   U = ranuni(&seed);
   x1 = ranuni(&seed);
   x2 = ranexp(&seed);
   x3 = abs(rannor(&seed));
   y = x1*(U-0.1) + x2*(U*U-0.25) + x3*(exp(U)-exp(0.9));
do j=4 to &p;
   x{j} = ranuni(&seed);
end;
output;
end;
runt;
```

Variable U in the data set indicates the true quantile level of the response y conditional on $x = (x_1, \ldots, x_p)'$. Let $Q_Y(\tau|x) = x'\beta(\tau)$ denote the underlying quantile regression model, where $\beta(\tau) = (\beta_1(\tau), \ldots, \beta_p(\tau))'$. Then, the true parameter functions are

\[
\begin{align*}
\beta_1(\tau) &= \tau - 0.1 \\
\beta_2(\tau) &= \tau^2 - 0.25 \\
\beta_3(\tau) &= \exp(\tau) - \exp(0.9) \\
\beta_4(\tau) &= \cdots = \beta_p(\tau) = 0
\end{align*}
\]

It is easy to see that, at $\tau = 0.1$, only $\beta_2(0.1) = -0.24$ and $\beta_3(0.1) = \exp(0.1) - \exp(0.9) \approx -1.354432$ are nonzero parameters. Therefore, an effective effect-selection method should select $x_2$ and $x_3$ and drop all the other effects in this data set at $\tau = 0.1$. By the same rationale, $x_1$ and $x_3$ should be selected at $\tau = 0.5$ with $\beta_1(0.5) = 0.4$ and $\beta_3(0.5) \approx -0.810882$, and $x_1$ and $x_2$ should be selected at $\tau = 0.9$ with $\beta_1(0.9) = 0.8$ and $\beta_2(0.9) = 0.56$.

The following statements load the analysisData data set into your CAS session by using the mycas engine libref:
data mycas.analysisData;
  set analysisData;
run;

The following statements use PROC QTRSELECT with the forward selection method. The STB option and the CLB option in the MODEL statement request the standardized parameter estimates and the confidence limits of parameter estimates, respectively.

    proc qtrselect data=mycas.analysisData;
      model y= x1-x&p / quantile=0.1 0.5 0.9 clb;
      selection method=forward;
      output out=mycas.out p=pred;
    run;

Output 18.1.1 shows that, by default, the CHOOSE= and STOP= options are both set to SBC.

Output 18.1.1  Selection Information

The QTRSELECT Procedure

<table>
<thead>
<tr>
<th>Selection Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selection Method</td>
</tr>
<tr>
<td>Select Criterion</td>
</tr>
<tr>
<td>Stop Criterion</td>
</tr>
<tr>
<td>Effect Hierarchy Enforced</td>
</tr>
<tr>
<td>Stop Horizon</td>
</tr>
</tbody>
</table>

Output 18.1.2, Output 18.1.3, and Output 18.1.4 display the selected effects and the parameter estimates for \( \tau = 0.1 \), \( \tau = 0.5 \), and \( \tau = 0.9 \), respectively. You can see that the forward selection method correctly selects active effects for all three quantile levels.

Output 18.1.2  Parameter Estimates at \( \tau = 0.1 \)

| Parameter DF Estimate Standard Error 95% Confidence Limits t Value Pr > |t| |
|-----------------|---------|-----------------|------------------|-------|--------|
| Intercept       | 1       | 0.01179         | 0.01192          | -0.01158 | 0.03516 | 0.99   | 0.3225 |
| x2              | 1       | -0.22871        | 0.00946          | -0.24725 | -0.21017 | -24.19 | <.0001 |
| x3              | 1       | -1.37991        | 0.01556          | -1.41042 | -1.34939 | -88.67 | <.0001 |

Output 18.1.3  Parameter Estimates at \( \tau = 0.5 \)

| Parameter DF Estimate Standard Error 95% Confidence Limits t Value Pr > |t| |
|-----------------|---------|-----------------|------------------|-------|--------|
| Intercept       | 1       | 0.00946         | -0.01158         | 0.02046 | 0.3225  | 0.99   | 0.3225 |
| x2              | 1       | -0.24725        | -0.24725         | -0.21017 | -24.19  | <.0001 |
| x3              | 1       | -1.41042        | -1.41042         | -1.34939 | -88.67  | <.0001 |

Output 18.1.4  Parameter Estimates at \( \tau = 0.9 \)

| Parameter DF Estimate Standard Error 95% Confidence Limits t Value Pr > |t| |
|-----------------|---------|-----------------|------------------|-------|--------|
| Intercept       | 1       | 0.01556         | -0.01158         | 0.03112 | 0.3225  | 0.99   | 0.3225 |
| x2              | 1       | -0.21017        | -0.24725         | -0.21017 | -24.19  | <.0001 |
| x3              | 1       | -1.34939        | -1.41042         | -1.34939 | -88.67  | <.0001 |
Example 18.2: Growth Charts for Body Mass Index

This example is modeled on an example in the section “Getting Started: QUANTREG Procedure” in the SAS/STAT User’s Guide. This example highlights the use of the QTRSELECT procedure for multiple-level quantile regression by creating growth charts for men’s body mass index (BMI).

BMI, which is defined as the ratio of weight (kg) to squared height (m$^2$), is a standard measure for categorizing individuals as overweight or underweight. The percentiles of BMI for specified ages are of particular interest. This example draws smooth BMI quantile curves conditional on Age, which can serve as BMI growth charts in medical diagnoses to identify BMI percentiles for subjects.

The Sashelp.BMI-Men data set records 1999–2000 and 2001–2002 survey results for men’s BMI. These survey results are published by the National Center for Health Statistics. The data set contains 3,264 observations for two variables, BMI and Age.

The following statements display the variables in the data set. Output 18.2.1 shows the results.

```
proc contents varnum data=sashelp.BMI-Men;
   ods select position;
run;
```

Output 18.2.1 Sashelp.BMI-Men Data Set

The CONTENTS Procedure

---

### Example 18.2: Growth Charts for Body Mass Index (continued)

| Parameter | DF | Estimate | Standard Error | 95% Confidence Limits | t Value | Pr > |t| |
|-----------|----|----------|----------------|-----------------------|---------|-------|---|
| Intercept | 1  | 0.01178  | 0.03418        | -0.05524              | 0.07879 | 0.34  | 0.7304 |
| x1        | 1  | 0.42584  | 0.06237        | 0.30355               | 0.54814 | 6.83  | <.0001 |
| x3        | 1  | -0.86332 | 0.04765        | -0.95674              | -0.76989| -18.12| <.0001 |

Output 18.1.4 Parameter Estimates at $\tau = 0.9$

**Selected Effects:** Intercept x1 x2

| Parameter | DF | Estimate | Standard Error | 95% Confidence Limits | t Value | Pr > |t| |
|-----------|----|----------|----------------|-----------------------|---------|-------|---|
| Intercept | 1  | -0.00774 | 0.03292        | -0.07228              | 0.05680 | -0.24 | 0.8142 |
| x1        | 1  | 0.78294  | 0.05134        | 0.68228               | 0.88360 | 15.25 | <.0001 |
| x2        | 1  | 0.57644  | 0.03422        | 0.50935               | 0.64354 | 16.85 | <.0001 |
Chapter 18: The QTRSELECT Procedure

The following statements load the Sashelp.BMIMen data set into your CAS session by using the mycas engine libref:

```r
data mycas.BMIMen;
set sashelp.BMIMen;
SqrtAge = sqrt(Age);
InveAge = 1/Age;
LogBMI = log(BMI);
run;
```

The logarithm of BMI is used as the response. The following statements fit quantile regression models for the mycas.BMIMen data table at 10 quantile levels:

```r
%let quantile=0.03 0.05 0.1 0.25 0.5 0.75 0.85 0.90 0.95 0.97;
%let nq=10;

proc qtrselect data=mycas.BMIMen;
model logBMI = InveAge SqrtAge Age SqrtAge*Age Age*Age*Age / quantile=&quantile;
output out=mycas.BMIOut copyvars=(BMI Age) pred=P_LogBMI;
run;
```

The OUTPUT statement outputs the mean predicted quantiles for the 10 specified quantile levels. The PRED= option in the OUTPUT statement specifies the variable names for the quantile predictions. For example, p1 is for quantile level 0.03, and p2 is for quantile level 0.05.

The following statements define and apply a SAS macro function to create a quantile curves plot for the mycas.BMIOut data table:

```r
%let BMIcolor=red olive orange blue brown gray violet black gold green;

%macro plotBMI;
  data BMIPred;
  set mycas.BMIOut;
  %do j=1 %to &nq;
    predBMI&j = exp(P_LogBMI&j);
  %end;
  label %do j=1 %to &nq;
    predBMI&j=%qscan(&quantile,&j,%str( ))
    %end;;
  run;

  proc sort data=BMIPred;
    by Age;
  run;

  proc sgplot data=BMIPred;
    %do j=1 %to &nq;
      series y=predBMI&j x=Age/lineattrs=(thickness=2
        color=%qscan(&BMIcolor,&j,%str( )));
    %end;
  scatter y=BMI x=Age/markerattrs=(size=5);
  run;
%mend;

%plotBMI;
```
Output 18.2.2 shows the BMI quantile curves, which can serve as BMI growth charts. For example, the percentiles of any observations (small blue circles) that are located between the top 0.95 quantile (gold) curve and the 0.97 quantile (green) curve are between the 95th percentile and the 97th percentile. By using this rule, you can measure the percentile range for any observations of interest.

**Example 18.3: Pollution and Mortality**

This example shows how you can produce the ODS graphical summary for the effect selection process, and how you can use the PARTITION statement and other options to control the effect selection process. The data for this example come from a study by McDonald and Schwing (1973). The data set contains 60 observations, 15 covariates, and one response variable. The response variable is the total age-adjusted mortality rate for Standard Metropolitan Statistical Areas in the United States in 1959–1961.

The following DATA step creates the Mortality data set:

```r
data mortality;
  input index aap ajant ajult size65 nph nsch25 nfek ppsm snwp nowk nin3k
  hpi nopi sdpi datm DeathRate;
  label index="the index"
```

---

**Output 18.2.2 Growth Chart for Body Mass Index**

![Growth Chart for Body Mass Index](image)
data Chapter18; 
  input aap ajant ajult size65 nph nsch25 nfek ppsm snwp nowk nin3k hpi nopi datm DeathRate ;
  datalines;
  1  36 27 71 8.1 3.34 11.4 81.5 3243 8.8 42.6 11.7 21 15 59 59 921.870  
  2  35 23 72 11.1 3.14 11.0 78.8 4281 3.6 50.7 14.4 8 10 39 57 997.875  
  3  44 29 74 10.4 3.21 9.8 81.6 4260 0.8 39.4 12.4 6 6 33 54 962.354  
  4  47 45 79 6.5 3.41 11.1 77.5 3125 27.1 50.2 20.6 18 8 24 56 982.291  
  5  43 35 77 7.6 3.44 9.6 84.6 6441 24.4 43.7 14.3 43  
  ... more lines ...  
  11 42 56 1003.502 58 45 24 70 11.8 3.25 11.1 79.8 3678 1.0 44.8 14.0 7 3 8 56 895.696 59 42 83 76 9.7 3.22 9.0 76.2 9699 4.8 42.2 14.5 8 8 49 54 911.817 60 38 28 72 8.9 3.48 10.7 79.8 3451 11.7 37.5 13.0 14 13 39 58 954.442  
;
Example 18.3: Pollution and Mortality

The following statements load the `mycas.Mortality` data set into your CAS session. The DATA step assumes that your CAS engine libref is names `mycas`, but you can substitute any appropriately defined CAS engine libref:

```plaintext
data mycas.Mortality;
  set Mortality;
run;
```

The following statements fit a median model for mortality rate conditional on a set of climate, demographic, and pollution covariates by using the forward selection method:

```plaintext
ods graphics on;
proc qtrselect data=mycas.Mortality;
  partition fraction(validate=0.3 seed=8);
  model DeathRate = aap aap*aap ajant ajant*ajant ajult
                       ajult*ajult size65 size65*size65 nph nph*nph nsch25
                       nsch25*nsch25 nfek nfek*nfek ppsm ppsm*ppsm snwp snwp*snwp
                       nowk nowk*nowk nin3k nin3k*nin3k hpi hpi*hpi nopi
                       nopi*nopi sdpi sdpi*sdpi datm datm*datm
    / quantile=0.5;
  selection method=forward(choose=validate) sh=8 hier=single
    plot=all;
  output out=mycas.OutData copyvar=DeathRate p=Pred role=Role;
run;
```

Because linear terms alone might not be sufficient to fit this model, quadratic terms are also added in the MODEL statement. The FRACTION option of the PARTITION statement requests that 30% of the observations be used for validation and the remaining 70% of the observations be used for training. The SEED suboption in the FRACTION option specifies the random seed for partitioning the data. The HIER=SINGLE option in the MODEL statement forces the effect selection process to ignore quadratic effect candidates if their corresponding main effects are not in the model. The OUTPUT statement creates a SAS data set named `mycas.OutData`, which contains the variable `Role` for observation roles and the variable `Pred` for median prediction conditional on final selected effects.

Output 18.3.1 shows the selection summary. You can see that the best model is at step 10 for validation ACL and step 14 for the SBC.
## Output 18.3.1  Selection Summary

### The QTRSELECT Procedure

#### Quantile Level = 0.5

#### Selection Details

<table>
<thead>
<tr>
<th>Step</th>
<th>Effect</th>
<th>Entered</th>
<th>Number Effects In</th>
<th>SBC</th>
<th>ACL</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Intercept</td>
<td></td>
<td>1</td>
<td>277.8052</td>
<td>22.3061</td>
</tr>
<tr>
<td>1</td>
<td>snwp</td>
<td>2</td>
<td>255.0004</td>
<td>17.3987</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>sdpi</td>
<td>3</td>
<td>244.3580</td>
<td>15.4860</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>nopi</td>
<td>4</td>
<td>237.7798</td>
<td>15.2963</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>nowk</td>
<td>5</td>
<td>236.3809</td>
<td>14.4011</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>ajult</td>
<td>6</td>
<td>234.2327</td>
<td>15.5494</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>aap</td>
<td>7</td>
<td>231.1990</td>
<td>13.7808*</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>snwp*snwp</td>
<td>8</td>
<td>229.9437</td>
<td>13.9181</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>nowk*nowk</td>
<td>9</td>
<td>232.0149</td>
<td>14.4375</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>nin3k</td>
<td>10</td>
<td>230.4295</td>
<td>17.6828</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>nph</td>
<td>11</td>
<td>231.8665</td>
<td>19.2914</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>nsch25</td>
<td>12</td>
<td>232.7794</td>
<td>19.2858</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>nsch25*nsch25</td>
<td></td>
<td>228.5889</td>
<td>22.7282</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>size65</td>
<td>14</td>
<td>227.4682*</td>
<td>27.4452</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>datm</td>
<td>15</td>
<td>228.8842</td>
<td>29.1958</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>nfek</td>
<td>16</td>
<td>228.0792</td>
<td>31.9374</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>datm*datm</td>
<td>17</td>
<td>228.2483</td>
<td>32.2392</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>size65*size65</td>
<td>18</td>
<td>228.5158</td>
<td>36.5253</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>nph*nph</td>
<td>19</td>
<td>229.5477</td>
<td>36.4157</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>nopi*nopi</td>
<td>20</td>
<td>230.3984</td>
<td>38.8426</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>ppsm</td>
<td>21</td>
<td>227.8287</td>
<td>39.7214</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>hpi</td>
<td>22</td>
<td>230.5357</td>
<td>40.6773</td>
<td></td>
</tr>
</tbody>
</table>

* Optimal Value Of Criterion
Output 18.3.2 shows the selected effects and the relevant estimates.

**Output 18.3.2 Parameter Estimates**

*Selected Effects: intercept aap ajult snwp nowk nopi sdpi*

**The QTRSELECT Procedure**

Quantile Level = 0.5
Selected Model

<table>
<thead>
<tr>
<th>Parameter Estimates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
</tr>
</tbody>
</table>
| Intercepts 1 1052 162.78569 6.46 <.0001
| aap 1 1.41886 1.33368 1.06 0.2947
| ajult 1 -2.04810 2.85863 -0.72 0.4785
| snwp 1 4.09570 1.52556 2.68 0.0110
| nowk 1 -1.87116 1.68126 -1.11 0.2733
| nopi 1 -0.18440 0.51884 -0.36 0.7244
| sdpi 1 0.39674 0.08245 4.81 <.0001
Output 18.3.3 shows the progression of the standardized parameter estimates as the selection process proceeds.

**Output 18.3.3 Coefficient Panel**

**Coefficient Progression for DeathRate**
Quantile Level = 0.5

The diagram shows the progression of the standardized coefficient estimates for variables such as `nsch25`, `datm * datm`, `nph`, `nph * nph`, `size65 * size65`, and `nph * nph` as the selection process proceeds. The y-axis represents the standardized coefficient, and the x-axis shows the effect sequence from `Intercept` to `21+ hqi`.
Output 18.3.4 shows the progression of the average check losses for training data and validation data as the selection process proceeds.

**Output 18.3.4** Average Check Loss Plot

![Progression of ACL by Role for DeathRate](Image)
Output 18.3.5 shows the progression of six effect selection criteria as the selection process proceeds.

**Output 18.3.5 Criterion Panel**

Image of a graph showing the progression of criteria such as Adj R1, AIC, AICC, SBC, and ACL (Train) and ACL (Validate) for the DeathRate with Quantile Level = 0.5.
Output 18.3.6 shows the first 10 observations of the OUTPUT data set.

<table>
<thead>
<tr>
<th>Obs</th>
<th>Role</th>
<th>Pred</th>
<th>DeathRate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>934.933</td>
<td>921.870</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>982.291</td>
<td>982.291</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>902.719</td>
<td>934.700</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>913.280</td>
<td>912.347</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>972.209</td>
<td>970.467</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>892.894</td>
<td>860.101</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>970.780</td>
<td>959.221</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>888.544</td>
<td>871.338</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>943.251</td>
<td>952.529</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>903.261</td>
<td>844.053</td>
</tr>
</tbody>
</table>

References


# Chapter 19
The REGSELECT Procedure

## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview: REGSELECT Procedure</td>
<td>882</td>
</tr>
<tr>
<td>PROC REGSELECT Features</td>
<td>882</td>
</tr>
<tr>
<td>PROC REGSELECT Compared with Other SAS Procedures</td>
<td>883</td>
</tr>
<tr>
<td>Using CAS Sessions and CAS Engine Librefs</td>
<td>884</td>
</tr>
<tr>
<td>Getting Started: REGSELECT Procedure</td>
<td>885</td>
</tr>
<tr>
<td>Syntax: REGSELECT Procedure</td>
<td>893</td>
</tr>
<tr>
<td>PROC REGSELECT Statement</td>
<td>893</td>
</tr>
<tr>
<td>BY Statement</td>
<td>894</td>
</tr>
<tr>
<td>CLASS Statement</td>
<td>894</td>
</tr>
<tr>
<td>CODE Statement</td>
<td>895</td>
</tr>
<tr>
<td>DISPLAY Statement</td>
<td>895</td>
</tr>
<tr>
<td>DISPLAYOUT Statement</td>
<td>896</td>
</tr>
<tr>
<td>EFFECT Statement</td>
<td>897</td>
</tr>
<tr>
<td>FREQ Statement</td>
<td>899</td>
</tr>
<tr>
<td>MODEL Statement</td>
<td>899</td>
</tr>
<tr>
<td>OUTPUT Statement</td>
<td>901</td>
</tr>
<tr>
<td>PARTITION Statement</td>
<td>903</td>
</tr>
<tr>
<td>SELECTION Statement</td>
<td>904</td>
</tr>
<tr>
<td>WEIGHT Statement</td>
<td>905</td>
</tr>
<tr>
<td>Details: REGSELECT Procedure</td>
<td>905</td>
</tr>
<tr>
<td>Criteria Used in Model Selection</td>
<td>905</td>
</tr>
<tr>
<td>Diagnostic Statistics</td>
<td>907</td>
</tr>
<tr>
<td>Classification Variables and the SPLIT Option</td>
<td>908</td>
</tr>
<tr>
<td>Using Validation and Test Data</td>
<td>909</td>
</tr>
<tr>
<td>Computational Method</td>
<td>910</td>
</tr>
<tr>
<td>Joint Tests and Type 3 Tests</td>
<td>911</td>
</tr>
<tr>
<td>Displayed Output</td>
<td>911</td>
</tr>
<tr>
<td>ODS Table Names</td>
<td>915</td>
</tr>
<tr>
<td>ODS Graphics</td>
<td>917</td>
</tr>
<tr>
<td>Examples: REGSELECT Procedure</td>
<td>918</td>
</tr>
<tr>
<td>Example 19.1: Model Selection with Validation</td>
<td>918</td>
</tr>
<tr>
<td>References</td>
<td>926</td>
</tr>
</tbody>
</table>
Overview: REGSELECT Procedure

The REGSELECT procedure fits and performs model selection for ordinary least squares regression models in SAS Viya.

The models that PROC REGSELECT supports can contain main effects that consist of both continuous and classification variables and interaction effects of these variables. The models can also include constructed effects such as splines. The procedure offers a number of effect-selection methods, including stepwise methods and modern LASSO methods. It also offers extensive capabilities for customizing the model selection by using a wide variety of selection and stopping criteria, from computationally efficient significance-level-based criteria to modern, computationally intensive validation-based criteria. PROC REGSELECT also provides a variety of regression diagnostics that are conditional on the selected model.

PROC REGSELECT Features

The main features of the REGSELECT procedure are as follows:

- **Model specification**
  - supports multiple parameterizations for classification effects
  - supports any degree of interaction (crossed effects) and nested effects
  - supports a hierarchy among effects
  - supports partitioning of data into training, validation, and testing roles
  - provides a FREQ statement for grouped analysis
  - provides a WEIGHT statement for weighted analysis

- **Selection control**
  - provides multiple effect-selection methods
  - offers selection of individual levels of classification effects
  - provides effect selection based on a variety of selection criteria
  - provides stopping rules based on a variety of model evaluation criteria
  - supports stopping and selection rules based on external validation and leave-one-out cross validation

- **Display and output**
  - produces output data tables that contain predicted values, residuals, studentized residuals, confidence limits, and influence statistics
  - uses ODS Graphics to create model selection plots as part of its output. For more information about ODS Graphics, see the section “ODS Graphics” on page 917.

The REGSELECT procedure supports the following effect-selection methods. For a more detailed description of these methods, see the section “SELECTION Statement” on page 36 in Chapter 3, “Shared Concepts.”
• Forward selection starts with no effects in the model and adds effects.

• Backward elimination starts with all effects in the model and deletes effects.

• Stepwise regression is similar to forward selection, except that effects already in the model do not necessarily stay there.

• Forward-swap selection is an extension of the forward selection method. Before any addition step, PROC REGSELECT makes all pairwise swaps of one effect in the model and one effect out of the current model that improve the selection criterion. When the selection criterion is R square, this method is the same as the MAXR method in the REG procedure in SAS/STAT software.

• Least angle regression, like forward selection, starts with no effects in the model and adds effects. The parameter estimates at any step are “shrunk” when compared to the corresponding least squares estimates.

• LASSO adds and deletes parameters based on a version of ordinary least squares in which the sum of the absolute regression coefficients is constrained. PROC REGSELECT also supports adaptive LASSO selection, in which weights are applied to each of the parameters in forming the LASSO constraint.

Hybrid versions of the LAR and LASSO methods are also supported. They use LAR or LASSO to select the model, but they estimate the regression coefficients by ordinary weighted least squares.

Because the REGSELECT procedure runs on CAS, it also does the following:

• enables you to run on a cluster of machines that distribute the data and the computations

• enables you to run in single-machine mode on CAS

• exploits all the available cores and concurrent threads. For information about how PROC REGSELECT uses threads, see the section “Multithreading” on page 81 in Chapter 3, “Shared Concepts.”

---

**PROC REGSELECT Compared with Other SAS Procedures**

The REGSELECT procedure provides regression modeling functionality that is comparable to that of the HPREG, GLMSELECT, and REG procedures in SAS/STAT software.

**PROC REGSELECT Compared with the HPREG Procedure**

The functionality of the REGSELECT procedure closely resembles that of the HPREG procedure, which is a high-performance procedure. The REGSELECT procedure is the next generation of the HPREG procedure, and it was developed specifically for SAS Viya. Both procedures are designed to run on a cluster of machines that distribute the data and the computations.

Both the REGSELECT and HPREG procedures fit and perform model selection for ordinary least squares regression models, which you can specify as general linear models that include classification variables. You request model selection by using the SELECTION statement.

The REGSELECT and HPREG procedures offer the same methods of effect selection, but the REGSELECT procedure also produces selection plots by using ODS Graphics. The REGSELECT and HPREG procedures
offer the same methods of customizing the model selection, and the same set of regression diagnostics. The main differences in functionality are that the REGSELECT procedure provides the ability to specify constructed effects with the EFFECT statement, and it computes Type 3 tests of effects.

**PROC REGSELECT Compared with the GLMSELECT Procedure**

The functionality of the REGSELECT procedure also closely resembles that of the GLMSELECT procedure. Both procedures fit and perform model selection for ordinary least squares regression models, which you can specify as general linear models that include classification variables. Both procedures offer the same methods of effect selection (including the LAR and LASSO methods), the ability to use external validation data and cross validation as selection criteria, and extensive options to customize the selection process. Both procedures provide the ability to specify constructed effects with the EFFECT statement.

Unlike the GLMSELECT procedure, the REGSELECT procedure does not perform model selection by default. If you request model selection by using the SELECTION statement, then the default selection method is stepwise selection based on the Schwarz Bayesian information criterion (SBC). This default matches the default method in PROC GLMSELECT.

With the REGSELECT procedure—but not with the GLMSELECT procedure—you can request observationwise residual and influence diagnostics in the OUTPUT statement and variance inflation and tolerance statistics for the parameter estimates. If the fitted model has been obtained by performing model selection, then these statistics are conditional on the selected model and do not take into account the variability introduced by the selection process.

**PROC REGSELECT Compared with the REG Procedure**

A major functional difference between the REGSELECT procedure and the REG procedure is that the REGSELECT procedure enables you to specify general linear models that include classification variables. In this respect it is similar to the GLM and GLMSELECT procedures in SAS/STAT.

Unlike the REG procedure, the REGSELECT procedure supports the LAR and LASSO methods, the ability to use external validation data and cross validation as selection criteria, and extensive options to customize the selection process. PROC REGSELECT does not support the all-subset-based methods that you find in PROC REG, nor does it support the MINR method.

Like the REG procedure, PROC REGSELECT does not perform model selection by default. If you request model selection by using the SELECTION statement in PROC REGSELECT, then the default selection method is stepwise selection based on the Schwarz Bayesian information criterion (SBC).

With the REGSELECT procedure, as with the REG procedure, you can request observationwise residual and influence diagnostics in the OUTPUT statement, and variance inflation and tolerance statistics for the parameter estimates. If the fitted model has been obtained by performing model selection, then these statistics are conditional on the selected model and do not take into account the variability introduced by the selection process.

**Using CAS Sessions and CAS Engine Librefs**

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator
if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:

```sas
proc options option=(CASHOST CASPORT);
run;
```

In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:

```sas
cas mysess;
libname mycas cas sessref=mysess;
```

The CAS statement creates the CAS session named `mysess`, and the LIBNAME statement creates the `mycas` CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the CASHOST and CASPORT of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the `mysess` session, you can terminate it by using the TERMINATE option in the CAS statement as follows:

```sas
cas mysess terminate;
```

For more information about the CAS and LIBNAME statements, see the section “Introduction to Shared Concepts” on page 10 in Chapter 3, “Shared Concepts.”

---

**Getting Started: REGSELECT Procedure**

The following example is closely modeled on the example in the section “Getting Started: GLMSELECT Procedure” in the *SAS/STAT User's Guide*.

The Sashelp.Baseball data set contains salary and performance information for Major League Baseball players who played at least one game in both the 1986 and 1987 seasons, excluding pitchers. The salaries (*Sports Illustrated*, April 20, 1987) are from the 1987 season, and the performance measures are from 1986 (Collier Books, *The 1987 Baseball Encyclopedia Update*). The following step displays (in Figure 19.1) the variables in the data set:

```sas
proc contents varnum data=sashelp.baseball;
   ods select position;
run;
```
Suppose you want to investigate whether you can model the players’ salaries from the 1987 season based on performance measures for the previous season. The aim is to obtain a parsimonious model that does not overfit these particular data, making the model useful for prediction. This example shows how you can use PROC REGSELECT as a starting point for such an analysis. Because the variation of salaries is much greater for the higher salaries, it is appropriate to apply a log transformation to the salaries before you do the model selection.

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

You can load the Sashelp.Baseball data set into your CAS session by using your CAS engine libref with the following DATA step:

```sas
data mycas.baseball;
  set sashelp.baseball;
run;
```

These statements assume that your CAS engine libref is named Mycas, as in the section “Using CAS Sessions and CAS Engine Librefs” on page 884, but you can substitute any appropriately defined CAS engine libref.
The following statements select a model by using the default settings for stepwise selection. ODS Graphics must be enabled before you can request plots. For more information about ODS Graphics, see the section “ODS Graphics” on page 917.

```latex
ods graphics on;

proc regselect data=mycas.baseball;
   class league division;
   model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                    yrMajor crAtBat crHits crHome crRuns crRbi
                    crBB league division nOuts nAssts nError;
   selection method=stepwise plots=all;
run;
```

The output and graphics from this analysis are presented in Figure 19.2 through Figure 19.6.

**Figure 19.2** Selection Information, Number of Observations, Class Level Information, and Dimensions

<table>
<thead>
<tr>
<th>The REGSELECT Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Selection Information</strong></td>
</tr>
<tr>
<td>Selection Method</td>
</tr>
<tr>
<td>Select Criterion</td>
</tr>
<tr>
<td>Stop Criterion</td>
</tr>
<tr>
<td>Effect Hierarchy Enforced</td>
</tr>
<tr>
<td>Stop Horizon</td>
</tr>
<tr>
<td><strong>Number of Observations Read</strong></td>
</tr>
<tr>
<td><strong>Number of Observations Used</strong></td>
</tr>
<tr>
<td><strong>Class Level Information</strong></td>
</tr>
<tr>
<td>Class</td>
</tr>
<tr>
<td>League</td>
</tr>
<tr>
<td>Division</td>
</tr>
<tr>
<td><strong>Dimensions</strong></td>
</tr>
<tr>
<td>Number of Effects</td>
</tr>
<tr>
<td>Number of Parameters</td>
</tr>
</tbody>
</table>

The “Selection Information” table provides details about the method and criteria used to perform the model selection. The requested selection method is a variant of the traditional stepwise selection in which the decisions about what effects to add or drop at any step and when to terminate the selection are both based on the Schwarz Bayesian information criterion (SBC). The effect in the current model whose removal yields the maximal decrease in the SBC value is dropped, provided that this lowers the SBC value. When no further decrease in the SBC value can be obtained by dropping an effect in the model, the effect whose addition to the model yields the lowest SBC value is added and the whole process is repeated. The method terminates when dropping or adding any effect increases the SBC value.
Chapter 19: The REGSELECT Procedure

Figure 19.2 displays the “Number of Observations,” “Class Level Information,” and “Dimensions” tables. The “Number of Observations” table shows that of the 322 observations in the input data, only 263 observations are used in the analysis because there are observations that contain incomplete data. The “Class Level Information” table lists the levels of the classification variables division and league. When you specify effects that contain classification variables, the number of parameters is usually larger than the number of effects. The “Dimensions” table shows the number of effects and the number of parameters that are considered.

The “Stepwise Selection Summary” table in Figure 19.3 shows the effect that was added or dropped at each step of the selection process together with fit statistics for the model at each step. In this case, both selection and stopping are based on the SBC.

![Figure 19.3 Selection Summary Table]

The REGSELECT Procedure

Selection Details

<table>
<thead>
<tr>
<th>Step</th>
<th>Effect Entered</th>
<th>Effect Removed</th>
<th>Number Effects In</th>
<th>SBC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Intercept</td>
<td></td>
<td>1</td>
<td>-57.2041</td>
</tr>
<tr>
<td>1</td>
<td>CrRuns</td>
<td>2</td>
<td>-194.3166</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>nHits</td>
<td>3</td>
<td>-252.5794</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>YrMajor</td>
<td>4</td>
<td>-262.7322</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>CrRuns</td>
<td>3</td>
<td>-262.8353</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>nBB</td>
<td>4</td>
<td>-269.7804*</td>
<td></td>
</tr>
</tbody>
</table>

* Optimal Value Of Criterion

Figure 19.4 displays the “Stop Reason,” “Selection Reason,” and “Selected Effects” tables. Note that these tables are displayed without any titles. The “Stop Reason” table indicates that selection stopped because adding or removing any effect would worsen the SBC value that is used as the selection criterion. In this case, because no CHOOSE= criterion is specified in the SELECTION statement, the final model is the selected model; this is indicated in the “Selection Reason” table. The “Selected Effects” table lists the effects in the selected model.

![Figure 19.4 Stopping and Selection Reasons]

Stepwise selection stopped because adding or removing an effect does not improve the SBC criterion.

The model at step 5 is selected.

Selected Effects: Intercept nHits nBB YrMajor
The coefficient panel in Figure 19.5 enables you to visualize the selection process. In this plot, standardized coefficients of all the effects that are selected at some step of the stepwise method are plotted as a function of the step number. This enables you to assess the relative importance of the effects that are selected at any step of the selection process and to know when effects entered the model. The lower plot in the panel shows how the criterion that is used to choose the selected model changes as effects enter or leave the model.

**Figure 19.5 Coefficient Progression**

The criterion panel in Figure 19.6 provides a graphical view of the progression of the fit criteria as the selection process evolves.
Chapter 19: The REGSELECT Procedure

**Figure 19.6** Criterion Panel

**Fit Criteria for logSalary**

The “Analysis of Variance,” “Fit Statistics,” and “Parameter Estimates” tables shown in Figure 19.7 display details of the selected model.

**Figure 19.7** Details of the Selected Model

The REGSELECT Procedure

**Selected Model**

<table>
<thead>
<tr>
<th>Source</th>
<th>DF</th>
<th>Sum of Squares</th>
<th>Mean Square</th>
<th>F Value</th>
<th>Pr &gt; F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>3</td>
<td>120.52553</td>
<td>40.17518</td>
<td>120.12</td>
<td>&lt;.0001</td>
</tr>
<tr>
<td>Error</td>
<td>259</td>
<td>86.62820</td>
<td>0.33447</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Corrected Total</td>
<td>262</td>
<td>207.15373</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Finally, a table is displayed that shows the amount of time (in seconds) that PROC REGSELECT required to perform the different tasks in the analysis.

### Figure 19.8 Procedure Timing

<table>
<thead>
<tr>
<th>Task</th>
<th>Seconds</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setup and Parsing</td>
<td>0.03</td>
<td>35.04%</td>
</tr>
<tr>
<td>Levelization</td>
<td>0.05</td>
<td>50.94%</td>
</tr>
<tr>
<td>Model Initialization</td>
<td>0.00</td>
<td>1.56%</td>
</tr>
<tr>
<td>SSCP Computation</td>
<td>0.00</td>
<td>4.56%</td>
</tr>
<tr>
<td>Model Selection</td>
<td>0.00</td>
<td>2.89%</td>
</tr>
<tr>
<td>Cleanup</td>
<td>0.00</td>
<td>4.70%</td>
</tr>
<tr>
<td>Total</td>
<td>0.09</td>
<td>100.00%</td>
</tr>
</tbody>
</table>

You might want to examine regression diagnostics for the selected model to investigate whether collinearity among the selected parameters or the presence of outlying or high-leverage observations might be affecting the fit. The following statements include some options and statements to obtain these diagnostics:

```plaintext
proc regselect data=mycas.baseball;
  class league division;
  model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                 yrMajor crAtBat crHits crHome crRuns crRbi
                 crBB league division nOuts nAssts nError / vif clb;
  selection method=stepwise;
  output out=mycas.baseballOut
    p=predictedLogSalary r h cookd rstudent copyvars=(name);
run;
```

The VIF and CLB options in the MODEL statement request variance inflation factors and 95% confidence limits, respectively, for the parameter estimates. Figure 19.9 shows the “Parameter Estimates” table, which displays these requested statistics. The variance inflation factors (VIF) measure the inflation in the variances of the parameter estimates due to collinearities that exist among the regressor (independent) variables.
Although there are no formal criteria for deciding whether a VIF is large enough to affect the predicted values, the VIF values for the selected effects in this example are small enough to indicate that there are no collinearity issues among the selected regressors.

**Figure 19.9** Parameter Estimates with Additional Statistics

The REGSELECT Procedure

Selected Model

| Parameter  | DF  | Estimate       | Standard Error | t Value | Pr > |t|  | Variance Inflation | 95% Confidence Limits |
|------------|-----|----------------|----------------|---------|-------|---|-------------------|------------------------|
| Intercept  | 1   | 4.013911       | 0.111290       | 36.07   | <.0001|   |                    | 3.79476 4.23306       |
| nHits      | 1   | 0.007929       | 0.000994       | 7.98    | <.0001|   | 1.49642 0.00597    | 0.00989                |
| nBB        | 1   | 0.007280       | 0.002049       | 3.55    | 0.0005|   | 1.52109 0.00325    | 0.01131                |
| YrMajor    | 1   | 0.100663       | 0.007551       | 13.33   | <.0001|   | 1.02488 0.08579    | 0.11553                |

By default, SAS Viya statistical procedures do not include all variables from the input data table in output data tables. The COPYVARS= option in the OUTPUT statement specifies that the variable name in the input data table be added as an identification variable in the *baseballOut* data table that is produced by the OUTPUT statement. In addition to this variable, the OUTPUT statement requests that predicted values, raw residuals, leverage values, Cook’s D statistics, and studentized residuals be added to the output data table. Note that default names are used for these statistics, except for the predicted values for which a specified name, predictedLogSalary, is supplied. The following statements use PROC PRINT to display five observations in this output data table:

```sas
proc print data=mycas.baseballOut(obs=5);
run;
```

**Figure 19.10** First 5 Observations of the baseballOut Data Set

<table>
<thead>
<tr>
<th>Obs</th>
<th>predictedLogSalary</th>
<th>Residual</th>
<th>COOKD</th>
<th>H</th>
<th>RSTUDENT</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4.73980</td>
<td>.</td>
<td>.</td>
<td>0.016087</td>
<td></td>
<td>Allanson, Andy</td>
</tr>
<tr>
<td>2</td>
<td>6.50852</td>
<td>-0.29392</td>
<td>.</td>
<td>.000730178</td>
<td>0.011060</td>
<td>-0.51031 Dawson, Andre</td>
</tr>
<tr>
<td>3</td>
<td>4.66148</td>
<td>-0.41299</td>
<td>.0002516874</td>
<td>0.018999</td>
<td>-0.72031 Newman, Al</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>6.52518</td>
<td>0.47788</td>
<td>.003069150</td>
<td>0.017361</td>
<td>0.83308 Thornton, Andre</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5.65468</td>
<td>0.65524</td>
<td>.002020832</td>
<td>0.006219</td>
<td>1.13715 Van Slyke, Andy</td>
<td></td>
</tr>
</tbody>
</table>
Syntax: REGSELECT Procedure

The following statements are available in the REGSELECT procedure:

```latex
PROC REGSELECT <options> ;
  BY variables ;
  CLASS variable <(options)> . . . <variable <(options)> > </global-options> ;
  CODE <options> ;
  DISPLAY <table-list> </options> ;
  DISPLAYOUT table-spec-list </options> ;
  EFFECT name=effect-type(variables </options>) ;
  FREQ variable ;
  MODEL dependent = <effects> </model-options> ;
  OUTPUT OUT=CAS-libref.data-table <keyword <name>> . . . <keyword <name>> ;
  PARTITION partition-options ;
  SELECTION <METHOD=method <(method-options)>> <options> ;
  WEIGHT variable ;
```

The PROC REGSELECT statement and a single MODEL statement are required. All other statements are optional. The CLASS statement can appear multiple times. If a CLASS statement is specified, it must precede the MODEL statement.

The rest of this section provides detailed syntax information about each of the preceding statements, beginning with the PROC REGSELECT statement. The remaining statements are described in alphabetical order.

**PROC REGSELECT Statement**

```latex
PROC REGSELECT <options> ;
```

The PROC REGSELECT statement invokes the procedure. Table 19.1 summarizes the options in the PROC REGSELECT statement by function.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Basic Options</strong></td>
<td></td>
</tr>
<tr>
<td>DATA=</td>
<td>Specifies the input data set</td>
</tr>
<tr>
<td><strong>Other Options</strong></td>
<td></td>
</tr>
<tr>
<td>NOCLPRINT</td>
<td>Limits or suppresses the display of class levels</td>
</tr>
<tr>
<td>ALPHA=</td>
<td>Sets the significance level to be used for the construction of confidence intervals</td>
</tr>
</tbody>
</table>
You can specify the following options:

**ALPHA=number**
sets the significance level to be used for the construction of confidence intervals. The value must be between 0 and 1; the default value of 0.05 results in 95% intervals. This option affects the keywords LCL, LCLM, UCL, and UCLM in the OUTPUT statement and the CLB option in the MODEL statement.

**DATA=** *CAS-libref.data-table*
names the input data table for PROC REGSELECT to use. The default is the most recently created data table. *CAS-libref.data-table* is a two-level name, where

*CAS-libref* refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to the data, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about *CAS-libref*, see the section “Using CAS Sessions and CAS Engine Librefs” on page 884.

*data-table* specifies the name of the input data table.

**NOCLPRINT<=number>**
suppresses the display of the “Class Level Information” table if you do not specify *number*. If you specify *number*, the values of the classification variables are displayed for only those variables whose number of levels is less than *number*. Specifying *number* helps reduce the size of the “Class Level Information” table if some classification variables have a large number of levels.

---

**BY Statement**

```
BY variables;
```

You can specify a BY statement in PROC REGSELECT to obtain separate analyses of observations in groups that are defined by the values of the BY variables. If you specify more than one BY statement, only the last one specified is used. For more information, see the discussion of BY-group processing in *SAS Language Reference: Concepts*.

---

**CLASS Statement**

```
CLASS variable <(options)> . . . <variable <(options)>> < / global-options> ;
```

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. You can list the response variable for binary models in the CLASS statement, but this is not required. Table 19.2 summarizes the values that you can use for either an option or a global-option. The options are fully documented in the section “CLASS Statement” on page 12 in Chapter 3, “Shared Concepts.”
Table 19.2  CLASS Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESCENDING</td>
<td>Reverses the sort order</td>
</tr>
<tr>
<td>MISSING</td>
<td>Treats missing values as valid levels</td>
</tr>
<tr>
<td>ORDER=</td>
<td>Specifies the sort order for the levels</td>
</tr>
<tr>
<td>PARAM=</td>
<td>Specifies the parameterization of the variable</td>
</tr>
<tr>
<td>REF=</td>
<td>Specifies the reference level of the variable</td>
</tr>
<tr>
<td>SPLIT</td>
<td>Allows design columns for a variable to enter or leave the model independently</td>
</tr>
</tbody>
</table>

**CODE Statement**

```sas
CODE < options > ;
```

The CODE statement writes SAS DATA step code for computing predicted values of the fitted model to a file, to a catalog entry, or to a CAS table. To score new data, you can then include the file or the catalog entry in a DATA step, or you can specify the CAS table in the `runCodeTable` action in the `dataStep` action set (for more information, see *SAS Viya: System Programming Guide*).

Table 19.3 summarizes the options available in the CODE statement.

Table 19.3  CODE Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMMENT</td>
<td>Adds comments to the generated code</td>
</tr>
<tr>
<td>FILE=</td>
<td>Names the file in which to save the generated code</td>
</tr>
<tr>
<td>FORMATWIDTH=</td>
<td>Specifies the numeric format width for the regression coefficients</td>
</tr>
<tr>
<td>INDENTSIZE=</td>
<td>Specifies the number of spaces to indent the generated code</td>
</tr>
<tr>
<td>LABELID=</td>
<td>Specifies a number used to construct names and labels</td>
</tr>
<tr>
<td>LINESIZE=</td>
<td>Specifies the line size for the generated code</td>
</tr>
<tr>
<td>NOTRIM</td>
<td>Compares formatted values, including blank padding</td>
</tr>
<tr>
<td>OUT=</td>
<td>Names an output CAS table in which to save the generated code</td>
</tr>
</tbody>
</table>

For more information about the syntax of the CODE statement, see the section “CODE Statement” on page 16 in Chapter 3, “Shared Concepts.”

**DISPLAY Statement**

```sas
DISPLAY < table-list > < / options > ;
```

The DISPLAY statement enables you to specify a list of display tables to display or exclude. This statement is similar to the ODS SELECT, ODS EXCLUDE, and ODS TRACE statements. However, the DISPLAY statement can improve performance when a large number of tables could be generated (such as in BY-group processing). The procedure processes the DISPLAY statement on a CAS server and thus sends only a
subset of ODS tables to the SAS client. Because ODS statements are processed on a SAS client, first all the
generated display tables are sent to the client, and then the client creates a subset.

If you use both DISPLAY and ODS statements together, the DISPLAY statement takes precedence over the
ODS statements. Note that the ODS EXCLUDE statement processes tables that are sent to the client after
they have been filtered by the DISPLAY statement. In some cases, it might appear that the ODS EXCLUDE
statement is taking precedence because it can further filter the tables. For more information about ODS, see

You can specify the **table-list** as a list of table names, paths, partial pathnames, and regular expressions.

The table names that you can specify are listed in the section “ODS Table Names” on page 915. A path is a table name that is prefixed with dot-separated grouping information. For example, a
SelectionSummary table that a procedure produces during a selection routine might have the path By-

A partial pathname does not include all groups; for example, Selection- and Summary.SelectionSummary are partial pathnames for Bygroup1.Summary.SelectionSummary.

When you specify a table name or partial pathname, all display tables whose paths end in the specified name
are selected for display or exclusion. For example, both SelectionSummary and Summary.SelectionSummary select Bygroup1.Summary.SelectionSummary.

A regular expression is enclosed in forward slashes (/). For example, specifying “/tions/” selects all pathnames
that contain the substring “tions”; in particular, the Bygroup1.Summary.SelectionSummary table is selected.
Specifying “/?tions/” selects all pathnames that do not contain the substring “tions”; in particular, the
Bygroup1.Summary.SelectionSummary table is not selected.

You can specify the following **options** after a slash (/):

- **CASESENSITIVE**
  - performs a case-sensitive comparison of table names in the **table-list** to display table names when
tables are subsetted for display. To preserve case, you must enclose table names in the **table-list** in
  quotation marks.

- **EXCLUDE**
  - displays all display tables except those that you specify in the **table-list**.

- **EXCLUDEALL**
  - suppresses display of all tables. This option takes precedence over the other options.

- **TRACE**
  - displays the display table names, labels, and paths.

---

**DISPLAYOUT Statement**

**DISPLAYOUT** table-spec-list < / options > ;

The DISPLAYOUT statement enables you to create CAS output tables from your displayed output. This
statement is similar to the ODS OUTPUT statement. For more information about ODS, see *SAS Output

The **table-spec-list** specifies a list of CAS output tables to create. Each entry in the list has either a **key=value**
format or a **key** format:
key=value  specifies key as the ODS table name, path, or partial pathname, and specifies value as the CAS output table name.

key  specifies key as the ODS table name and also as the CAS output table name.

The ODS table names that you can specify are listed in the section “ODS Table Names” on page 915. You cannot specify the ODS table named OutputCasTables in the table-spec-list.

Table names and partial pathnames are discussed under the DISPLAY statement. The DISPLAYOUT statement does not support regular expressions.

You can specify the following options after a slash (/):

INCLUDEALL  creates output CAS tables for all display tables. The name of the created output CAS table is the same as the corresponding display table name. If you specify this option, the table-spec-list specification is ignored.

NOREPLACE  does not replace any existing CAS output table of the same name.

REPEATED  replicates all CAS output tables on all nodes.

**EFFECT Statement**

```
EFFECT name=effect-type (variables < / options>); 
```

The EFFECT statement enables you to construct special collections of columns for design matrices. These collections are referred to as constructed effects to distinguish them from the usual model effects that are formed from continuous or classification variables, as discussed in the section “GLM Parameterization of Classification Variables and Effects” on page 54 in Chapter 3, “Shared Concepts.”

You can specify the following effect-types:

- **COLLECTION**  specifies a collection effect that defines one or more variables as a single effect that has multiple degrees of freedom. The variables in a collection are considered as a unit for purposes of estimation and inference.

- **MULTIMEMBER | MM**  specifies a multimember classification effect whose levels are determined by one or more variables that appear in a CLASS statement.

- **POLYNOMIAL | POLY**  specifies a multivariate polynomial effect in the specified numeric variables.

- **SPLINE**  specifies a regression spline effect whose columns are univariate spline expansions of one or more variables. A spline expansion replaces the original variable with an expanded or larger set of new variables.

Table 19.4 summarizes the options available in the EFFECT statement.
# Table 19.4  EFFECT Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Collection Effects Options</strong></td>
<td></td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays the constituents of the collection effect</td>
</tr>
<tr>
<td><strong>Multimember Effects Options</strong></td>
<td></td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays the levels of the multimember effect</td>
</tr>
<tr>
<td>NOEFFECT</td>
<td>Specifies that observations whose levels are all missing for the multimember variables should have 0 values in the corresponding design matrix columns</td>
</tr>
<tr>
<td>STDIZE</td>
<td>Standardizes the design matrix entries so that each observation has a sum of 1</td>
</tr>
<tr>
<td>WEIGHT=</td>
<td>Specifies the weight variable for the contributions of each classification effect</td>
</tr>
<tr>
<td><strong>Polynomial Effects Options</strong></td>
<td></td>
</tr>
<tr>
<td>DEGREE=</td>
<td>Specifies the degree of the polynomial</td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays details of the specified polynomial</td>
</tr>
<tr>
<td>MDEGREE=</td>
<td>Specifies the maximum degree of any variable in a term of the polynomial</td>
</tr>
<tr>
<td>NOSEPARATE</td>
<td>Treats the polynomial as a single effect with multiple degrees of freedom</td>
</tr>
<tr>
<td>STANDARDIZE=</td>
<td>Specifies centering and scaling suboptions for the variables that define the polynomial</td>
</tr>
<tr>
<td><strong>Spline Effects Options</strong></td>
<td></td>
</tr>
<tr>
<td>BASIS=</td>
<td>Specifies the type of basis (B-spline basis or truncated power function basis) for the spline effect</td>
</tr>
<tr>
<td>DATABOUNDARY</td>
<td>Uses the extremes of the data as boundary knots for a B-spline basis</td>
</tr>
<tr>
<td>DEGREE=</td>
<td>Specifies the degree of the spline effect</td>
</tr>
<tr>
<td>DETAILS</td>
<td>Displays the knots and locations for each spline basis function</td>
</tr>
<tr>
<td>KNOTMAX=</td>
<td>Requests equally spaced right-side boundary knots starting at the variables’ maximum and ending at the KNOTMAX= value</td>
</tr>
<tr>
<td>KNOTMETHOD=</td>
<td>Specifies how to construct the knots for the spline effect</td>
</tr>
<tr>
<td>KNOTMIN=</td>
<td>Requests equally spaced left-side boundary knots starting at the KNOTMIN= value and ending at the variables’ minimum value</td>
</tr>
<tr>
<td>NATURALCUBIC</td>
<td>Specifies a natural cubic spline basis for the spline effect</td>
</tr>
<tr>
<td>SEPARATE</td>
<td>Treats the spline basis for each variable as a separate effect when multiple variables are specified</td>
</tr>
<tr>
<td>SPLIT</td>
<td>Treats each design matrix column as a separate effect for selection methods</td>
</tr>
</tbody>
</table>

For more information about the syntax of these `effect-types` and how columns of constructed effects are computed, see the section “EFFECT Statement” on page 21 in Chapter 3, “Shared Concepts.”
**FREQ Statement**

```
FREQ variable ;
```

The `variable` in the FREQ statement identifies a numeric variable in the data set that contains the frequency of occurrence of each observation. PROC REGSELECT treats each observation as if it appears \( f \) times, where \( f \) is the value of the FREQ `variable` for the observation. If \( f \) is not an integer, it is truncated to an integer. If \( f \) is less than 1 or missing, the observation is not used in the analysis. When the FREQ statement is not specified, each observation is assigned a frequency of 1.

**MODEL Statement**

```
MODEL dependent=<effects> / <model-options> ;
```

The MODEL statement names the dependent variable and the explanatory effects, including covariates, main effects, interactions, and nested effects. If you omit the explanatory effects, the procedure fits an intercept-only model.

After the keyword MODEL, the dependent (response) variable is specified, followed by an equal sign. The explanatory effects follow the equal sign.

For information about constructing the model effects, see the section “Specification and Parameterization of Model Effects” on page 51 in Chapter 3, “Shared Concepts.”

The `model-options` control other aspects of model formation and inference. Table 19.5 summarizes these options.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model Options</td>
<td></td>
</tr>
<tr>
<td>CLB</td>
<td>Requests confidence limits</td>
</tr>
<tr>
<td>INCLUDE=</td>
<td>Includes effects in all models for model selection</td>
</tr>
<tr>
<td>INFORMATIVE</td>
<td>Models missing values by using extra indicator variables</td>
</tr>
<tr>
<td>NOINT</td>
<td>Suppresses the intercept</td>
</tr>
<tr>
<td>SS3</td>
<td>Displays the Type 3 or joint tests of effects</td>
</tr>
<tr>
<td>START=</td>
<td>Includes effects in the initial model for model selection</td>
</tr>
<tr>
<td>STB</td>
<td>Displays standardized estimates</td>
</tr>
<tr>
<td>TOL</td>
<td>Displays tolerance values for the estimates</td>
</tr>
<tr>
<td>VIF</td>
<td>Displays variance inflation factors for the estimates</td>
</tr>
</tbody>
</table>

**Model Options**

You can specify the following `options` in the MODEL statement after a slash (`/`):
CLB
requests the $100(1 - \alpha)\%$ upper and lower confidence limits for the parameter estimates. By default, the $95\%$ limits are computed; you can use the ALPHA= option in the PROC REGSELECT statement to change the $\alpha$ level. The CLB option is not supported when you specify METHOD=LAR or METHOD=LASSO in the SELECTION statement.

INCLUDE=\(n\)
INCLUDE=\(single\text{-}effect\)
INCLUDE=(\(effects\))
forces effects to be included in all models. If you specify INCLUDE=\(n\), then the first \(n\) effects listed in the MODEL statement are included in all models. If you specify INCLUDE=\(single\text{-}effect\) or INCLUDE=(\(single\text{-}effect\)), then the specified effects are forced into all models. The INCLUDE= option is not available when you specify METHOD=LAR or METHOD=LASSO in the SELECTION statement.

INFORMATIVE
models missing values by using extra model effects. These effects consist of dummy variables that take the value 1 when the value of a continuous model variable involved in the effect is missing, and take the value 0 otherwise. The missing value in the original model effect is replaced by the average value of the effect for the nonmissing values. For continuous-by-class effects, such as \(A*x\), where \(A\) is a classification variable and \(x\) is a continuous variable, informative missingness creates multiple dummy columns and substitutes the effect mean of \(x\) that corresponds to the respective level of \(A\). Missing values for classification variables are treated as valid levels. For more information about informative missingness, see the section “Informative Missingness” on page 78 in Chapter 3, “Shared Concepts.”

NOINT
suppresses the intercept term that is otherwise included in the model.

SS3
displays a “Model Anova” table that contains tests computed with Type III sums of squares for each effect. For more information, see the section “Joint Tests and Type 3 Tests” on page 911.

START=\(n\)
START=\(single\text{-}effect\)
START=(\(effects\))
is used to begin the selection process in the FORWARD, FORWARDSWAP, and STEPWISE selection methods from the initial model that you designate. If you specify START=\(n\), then the starting model consists of the first \(n\) effects listed in the MODEL statement. If you specify START=\(single\text{-}effect\) or START=(\(single\text{-}effect\)), then the starting model consists of these specified effects. The START= option is not available when you specify METHOD=BACKWARD, METHOD=LAR, or METHOD=LASSO in the SELECTION statement.

STB
produces standardized regression coefficients. A standardized regression coefficient is computed by dividing a parameter estimate by the ratio of the sample standard deviation of the dependent variable to the sample standard deviation of the regressor.
TOL
produces tolerance values for the estimates. Tolerance for a parameter is defined as $1 - R^2$, where $R^2$ is obtained from the regression of the parameter on all other parameters in the model. This option is not supported when you specify METHOD=LAR or METHOD=LASSO in the SELECTION statement.

VIF
produces variance inflation factors along with the parameter estimates. Variance inflation is the reciprocal of tolerance. The VIF option is not supported when you specify METHOD=LAR or METHOD=LASSO in the SELECTION statement.

**OUTPUT Statement**

```plaintext
OUTPUT OUT=CAS-libref.data-table
  < COPYVARS=(variables) >
  < keyword <=name>>...< keyword <=name>> ;
```

The OUTPUT statement creates a data table that contains observationwise statistics, which are computed after the model is fitted. The variables in the input data table are not included in the output data table, in order to avoid data duplication for large data tables; however, variables that you specify in the COPYVARS= option are included.

The output statistics are computed based on the parameter estimates for the selected model. For observations in which only the response variable is missing, predicted values are computed even though these observations do not affect the model fit. This enables, for example, predicted values to be computed for new observations.

You must specify the following option:

**OUT=**

names the output data table for PROC REGSELECT to use. You must specify this option before any other options. `CAS-libref.data-table` is a two-level name, where

- **CAS-libref** refers to a collection of information that is defined in the LIBNAME statement and includes the `caslib`, which includes a path to where the data table is to be stored, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about `CAS-libref`, see the section “Using CAS Sessions and CAS Engine Librefs” on page 884.
- **data-table** specifies the name of the output data table.

You can also specify the following syntax elements:

**COPYVAR=variable**

**COPYVARS=**

transfers one or more variables from the input data table to the output data table.

**keyword <=name>**

specifies the statistics to include in the output data table and optionally names the new variables that contain the statistics. Specify a keyword for each desired statistic (see the following list of keywords), followed optionally by an equal sign and a variable to contain the statistic.
If you specify \texttt{keyword=\textit{name}}, the new variable that contains the requested statistic has the specified name. If you omit the optional \texttt{=\textit{name}} after a \texttt{keyword}, then a default name is used.

You can specify the following values for \texttt{keyword} to request statistics that are available with all selection methods:

- \textbf{PREDICTED}
- \textbf{PRED}
- \textbf{P}
  
  requests predicted values for the response variable. The default is \texttt{Pred}.

- \textbf{RESIDUAL}
- \textbf{RESID}
- \textbf{R}
  
  requests the residual, calculated as \texttt{ACTUAL} – \texttt{PREDICTED}. The default is \texttt{Residual}.

- \textbf{ROLE}
  
  requests a numeric variable that indicates the role played by each observation in fitting the model. The default is \texttt{_ROLE_}. For each observation, the interpretation of this variable is shown in Table 19.6.

<table>
<thead>
<tr>
<th>Table 19.6</th>
<th>Role Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>Observation Role</td>
</tr>
<tr>
<td>0</td>
<td>Not used</td>
</tr>
<tr>
<td>1</td>
<td>Training</td>
</tr>
<tr>
<td>2</td>
<td>Validation</td>
</tr>
<tr>
<td>3</td>
<td>Testing</td>
</tr>
</tbody>
</table>

If you do not partition the input data by using a \texttt{PARTITION} statement, then the role variable value is 1 for observations used in fitting the model, and 0 for observations that have at least one missing or invalid value for the response, regressor, frequency, or weight variables.

In addition to the preceding statistics, you can also use the \texttt{keywords} listed in Table 19.7 in the OUTPUT statement to obtain additional statistics. These statistics are not available if you specify \texttt{METHOD=LAR} or \texttt{METHOD=LASSO} in the \texttt{SELECTION} statement. For computational formulas, see the section “Diagnostic Statistics” on page 907. All the statistics available in the OUTPUT statement are conditional on the selected model and do not take into account the variability introduced when you do model selection.

<table>
<thead>
<tr>
<th>Table 19.7</th>
<th>Keywords for OUTPUT Statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keyword</td>
<td>Description</td>
</tr>
<tr>
<td>COOKD</td>
<td>Cook’s D influence statistic</td>
</tr>
<tr>
<td>COVRATIO</td>
<td>Standard influence of observation on covariance of betas</td>
</tr>
<tr>
<td>DFFIT</td>
<td>Standard influence of observation on predicted value</td>
</tr>
<tr>
<td>H</td>
<td>Leverage, (x_i(X'X)^{-1}x_i')</td>
</tr>
</tbody>
</table>
### Table 19.7  continued

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCL</td>
<td>Lower bound of a $100(1 - \alpha)$% confidence interval for an individual prediction. This includes the variance of the error and the variance of the parameter estimates.</td>
</tr>
<tr>
<td>LCLM</td>
<td>Lower bound of a $100(1 - \alpha)$% confidence interval for the expected value (mean) of the dependent variable.</td>
</tr>
<tr>
<td>LIKEDIST</td>
<td>Likelihood displacement, which measures the change in the error sum of squares when the current observation is deleted.</td>
</tr>
<tr>
<td>PRESS</td>
<td>$i$th residual divided by $(1 - h)$, where $h$ is the leverage and where the model has been refit without the $i$th observation.</td>
</tr>
<tr>
<td>RSTUDENT</td>
<td>A studentized residual with the current observation deleted.</td>
</tr>
<tr>
<td>STDI</td>
<td>Standard error of the individual predicted value.</td>
</tr>
<tr>
<td>STDP</td>
<td>Standard error of the mean predicted value.</td>
</tr>
<tr>
<td>STDR</td>
<td>Standard error of the residual.</td>
</tr>
<tr>
<td>STUDENT</td>
<td>Studentized residuals, which are the residuals divided by their standard errors.</td>
</tr>
<tr>
<td>UCL</td>
<td>Upper bound of a $100(1 - \alpha)$% confidence interval for an individual prediction.</td>
</tr>
<tr>
<td>UCLM</td>
<td>Upper bound of a $100(1 - \alpha)$% confidence interval for the expected value (mean) of the dependent variable.</td>
</tr>
</tbody>
</table>

---

#### PARTITION Statement

**PARTITION** partition-option ;

The **PARTITION** statement specifies how observations in the input data set are logically partitioned into disjoint subsets for model training, validation, and testing. For more information, see the section “Using Validation and Test Data” on page 80 in Chapter 3, “Shared Concepts.” Either you can designate a variable in the input data table and a set of formatted values of that variable to determine the role of each observation, or you can specify proportions to use for randomly assigning observations to each role.

You must specify exactly one of the following **partition-options**:

**FRACTION**(TEST=fraction > < VALIDATE=fraction > < SEED=number >)

randomly assigns specified proportions of the observations in the input data table to the roles. You specify the proportions for testing and validation by using the TEST= and VALIDATE= suboptions. If you specify both the TEST= and VALIDATE= suboptions, then the sum of the specified fractions must be less than 1 and the remaining fraction of the observations are assigned to the training role. The SEED= option specifies an integer that is used to start the pseudorandom number generator for random partitioning of data for training, testing, and validation. If you do not specify SEED=number or if number is less than or equal to 0, the seed is generated by reading the time of day from the computer’s clock.
ROLE=variable (<TEST='value'> <TRAIN='value'> <VALIDATE='value'>)
ROLEVAR=variable (<TEST='value'> <TRAIN='value'> <VALIDATE='value'>)

names the variable in the input data table whose values are used to assign roles to each observation. This variable cannot also appear as an analysis variable in other statements or options. The TEST=, TRAIN=, and VALIDATE= suboptions specify the formatted values of this variable that are used to assign observation roles. If you do not specify the TRAIN= suboption, then all observations whose role is not determined by the TEST= or VALIDATE= suboption are assigned to the training role.

### SELECTION Statement

**SELECTION <METHOD=method <(method-options)> > <options> ;**

The SELECTION statement performs model selection by examining whether effects should be added to or removed from the model according to rules that are defined by model selection methods. The statement is fully documented in the section “SELECTION Statement” on page 36 in Chapter 3, “Shared Concepts.”

The REGSELECT procedure supports the following values of the METHOD= option in the SELECTION statement:

- **BACKWARD** specifies the backward elimination method, which starts with all effects in the model and deletes effects.
- **FORWARD** specifies the forward selection method, which starts with no effects in the model and adds effects.
- **FORWARDSWAP** specifies forward-swap selection, which is an extension of the forward selection method. Before any addition step, PROC REGSELECT makes all pairwise swaps of one effect in the model and one effect out of the current model that improve the selection criterion. When the selection criterion is $R^2$, this method is the same as the MAXR method in the REG procedure in SAS/STAT software.
- **LAR** specifies the least angle regression method. Like forward selection, this method starts with no effects in the model and adds effects. The parameter estimates at any step are “shrunk” when compared to the corresponding least squares estimates. If the model contains classification variables, then these classification variables are split. For more information about split classification variables, see Classification Variables and the SPLIT Option.
- **LASSO** specifies the LASSO method, which adds and deletes parameters based on a version of ordinary least squares in which the sum of the absolute regression coefficients is constrained. If the model contains classification variables, then these classification variables are split. For more information about split classification variables, see Classification Variables and the SPLIT Option.
- **NONE** specifies no model selection.
- **STEPWISE** specifies the stepwise regression method, which is similar to the forward selection method except that effects already in the model do not necessarily stay there.

By default, METHOD=STEPWISE.
The DETAILS=ALL and DETAILS=STEPS options produce the “ANOVA,” “Fit Statistics,” and “Parameter Estimates” tables, which provide information about the model that is selected at each step of the selection process.

If you specify the PLOTS=CRITERIA or PLOTS=ALL option, then a plot of the fit criteria by the selection step is created. The default fit criteria that are shown in this plot are the ADJRSQ, AIC, AICC, ASE, and SBC statistics. If you also specify a PARTITION statement, then the ASE statistics for all roles that are defined in the PARTITION statement are included. If you use the CP or PRESS statistic as the SELECT=, STOP=, or CHOOSE= criterion in the SELECTION statement, then these statistics are also added to the criterion panel.

If you specify the PLOTS=FITBYROLE or PLOTS=ALL option and a PARTITION statement, then a plot of the ASE by the selection step for each role is created.

**WEIGHT Statement**

```
WEIGHT variable;
```

The variable in the WEIGHT statement is used as a weight to perform a weighted analysis of the data. Observations that have nonpositive or missing weights are not included in the analysis. If a WEIGHT statement is not included, all observations that are used in the analysis are assigned a weight of 1.

**Details: REGSELECT Procedure**

**Criteria Used in Model Selection**

The REGSELECT procedure supports a variety of fit statistics that you can specify as criteria for the CHOOSE=, SELECT=, and STOP= options in the SELECTION statement. The following statistics are available:

- **ADJRSQ**: Adjusted R-square statistic (Darlington 1968; Judge et al. 1985)
- **AIC**: Akaike’s information criterion (Akaike 1969; Judge et al. 1985)
- **AICC**: Corrected Akaike’s information criterion (Hurvich and Tsai 1989)
- **BIC | SBC**: Schwarz Bayesian information criterion (Schwarz 1978; Judge et al. 1985)
- **CP**: Mallows’ $C_p$ statistic (Mallows 1973; Hocking 1976)
- **PRESS**: Predicted residual sum of squares statistic
- **RSQUARE**: R-square statistic (Darlington 1968; Judge et al. 1985)
- **SL**: Significance used to assess the contribution of an effect to the fit when it is added to or removed from a model
- **VALIDATE**: Average square error over the validation data
When you use SL as a criterion for effect selection, the definition depends on whether an effect is being considered as a drop candidate or an add candidate. Assume that the current model has \( p \) parameters, excluding the intercept. If you denote its residual sum of squares by \( \text{RSS}_p \), add an effect with \( k \) degrees of freedom, and denote the residual sum of squares of the resulting model by \( \text{RSS}_{p+k} \), then the \( F \) statistic for entry with \( k \) numerator degrees of freedom and \( n - (p + k) - 1 \) denominator degrees of freedom is given by

\[
F = \frac{(\text{RSS}_p - \text{RSS}_{p+k})/k}{\text{RSS}_{p+k}/(n - (p + k) - 1)}
\]

where \( n \) is number of observations used in the analysis. The significance level for adding an effect is the \( p \)-value of this \( F \) statistic, and the effect is deemed significant if it is smaller than the SLENTRY limit. (For more information about the SLENTRY= option, see the section “SELECTION Statement” on page 36 in Chapter 3, “Shared Concepts.”) Among several such add candidates, the effect with the smallest \( p \)-value (most significant) is deemed best.

If you drop an effect that has \( k \) degrees of freedom and denote the residual sum of squares of the resulting model by \( \text{RSS}_{p-k} \), then the \( F \) statistic for removal with \( k \) numerator degrees of freedom and \( n - p - k \) denominator degrees of freedom is given by

\[
F = \frac{(\text{RSS}_{p-k} - \text{RSS}_p)/k}{\text{RSS}_p/(n - p - k)}
\]

where \( n \) is number of observations used in the analysis. The significance level for removal of an effect is the \( p \)-value of this \( F \) statistic, and the effect is deemed not significant if this \( p \)-value is larger than the SLSTAY limit. (For more information about the SLSTAY= option, see the section “SELECTION Statement” on page 36 in Chapter 3, “Shared Concepts.”) Among several such removal candidates, the effect with the largest \( p \)-value (least significant) is deemed the best removal candidate.

It is known that the “\( F \)-to-enter” and “\( F \)-to-delete” statistics do not follow an \( F \) distribution (Draper, Guttman, and Kanemasu 1971). Hence the SLENTRY and SLSTAY values cannot reliably be viewed as probabilities. One way to address this difficulty is to replace hypothesis testing as a means of selecting a model with information criteria or out-of-sample prediction criteria. Although Harrell (2001) points out that information criteria were developed for comparing only prespecified models, Burnham and Anderson (2002) note that AIC criteria have routinely been used for several decades to perform model selection in time series analysis.

Table 19.8 provides formulas and definitions for these fit statistics.

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Definition or Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>( n )</td>
<td>Number of observations</td>
</tr>
<tr>
<td>( p )</td>
<td>Number of parameters, including the intercept</td>
</tr>
<tr>
<td>( \hat{\sigma}^2 )</td>
<td>Estimate of pure error variance from fitting the full model</td>
</tr>
<tr>
<td>SST</td>
<td>Total sum of squares corrected for the mean for the dependent variable</td>
</tr>
<tr>
<td>SSE</td>
<td>Error sum of squares</td>
</tr>
<tr>
<td>ASE</td>
<td>( \frac{\text{SSE}}{n} )</td>
</tr>
<tr>
<td>MSE</td>
<td>( \frac{\text{SSE}}{n - p} )</td>
</tr>
</tbody>
</table>
Diagnostic Statistics

This section gathers the formulas for the statistics available in the OUTPUT statement. All the statistics available in the OUTPUT statement are conditional on the selected model and do not take into account the variability introduced by doing model selection.

The model to be fit is $Y = X\beta + \epsilon$, and the parameter estimate is denoted by $b = (X'X)^{-1}X'Y$. The subscript $i$ denotes values for the $i$th observation, and the parenthetical subscript $(i)$ means that the statistic is computed by using all observations except the $i$th observation.

The ALPHA= option in the PROC REGSELECT statement is used to set the $\alpha$ value for the confidence limit statistics.

Table 19.9 contains the diagnostic statistics and their formulas. Each statistic is computed for each observation.

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Definition or Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R^2$</td>
<td>$1 - \frac{SSE}{SST} \frac{(n - 1)R^2}{n - p}$</td>
</tr>
<tr>
<td>ADJRSQ</td>
<td>$1 - \frac{n}{n - p}$</td>
</tr>
<tr>
<td>AIC</td>
<td>$n \log \left( \frac{SSE}{n} \right) + 2p + n + 2$</td>
</tr>
<tr>
<td>AICC</td>
<td>$n \log \left( \frac{SSE}{n} \right) + \frac{n(n + p)}{n - p - 2}$</td>
</tr>
<tr>
<td>CP ($C_p$)</td>
<td>$\frac{SSE}{\hat{\sigma}^2} + 2p - n$</td>
</tr>
<tr>
<td>PRESS</td>
<td>$\sum_{i=1}^{n} \frac{r_i^2}{(1 - h_i)^2}$ where $r_i$ = residual at observation $i$ and $h_i$ = leverage of observation $i = x_i(X'X)^{-1}x_i'$</td>
</tr>
<tr>
<td>RMSE</td>
<td>$\sqrt{\text{MSE}}$</td>
</tr>
<tr>
<td>SBC</td>
<td>$n \ln \left( \frac{\text{SSE}}{n} \right) + p \ln(n)$</td>
</tr>
</tbody>
</table>

Table 19.9  Formulas and Definitions for Diagnostic Statistics

<table>
<thead>
<tr>
<th>MODEL Option or Statistic</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRED ($\hat{Y}_i$)</td>
<td>$X_i b$</td>
</tr>
<tr>
<td>RES ($r_i$)</td>
<td>$Y_i - \hat{Y}_i$</td>
</tr>
<tr>
<td>H ($h_i$)</td>
<td>$x_i(X'X)^{-1}x_i'$</td>
</tr>
<tr>
<td>STDP</td>
<td>$\sqrt{h_i\hat{\sigma}^2}$</td>
</tr>
<tr>
<td>STDI</td>
<td>$\sqrt{(1 + h_i)\hat{\sigma}^2}$</td>
</tr>
<tr>
<td>STDR</td>
<td>$\sqrt{(1 - h_i)\hat{\sigma}^2}$</td>
</tr>
</tbody>
</table>
### Table 19.9  continued

<table>
<thead>
<tr>
<th>MODEL Option or Statistic</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCL</td>
<td>$\hat{Y}<em>i - t</em>{\frac{\alpha}{2}} \text{STDI}$</td>
</tr>
<tr>
<td>LCLM</td>
<td>$\hat{Y}<em>i - t</em>{\frac{\alpha}{2}} \text{STDP}$</td>
</tr>
<tr>
<td>UCL</td>
<td>$\hat{Y}<em>i + t</em>{\frac{\alpha}{2}} \text{STDI}$</td>
</tr>
<tr>
<td>UCLM</td>
<td>$\hat{Y}<em>i + t</em>{\frac{\alpha}{2}} \text{STDP}$</td>
</tr>
<tr>
<td>STUDENT</td>
<td>$\text{STDR}_i$</td>
</tr>
<tr>
<td>RSTUDENT</td>
<td>$\hat{\sigma}(i)\sqrt{1 - h_i}$</td>
</tr>
<tr>
<td>COOKD</td>
<td>$\frac{1}{p} \frac{\text{STUDENT}^2 \text{STDP}^2}{\text{STDR}^2}$</td>
</tr>
<tr>
<td>COVRATIO</td>
<td>$\frac{\det(\hat{\sigma}(i)(x'_i x_i)^{-1})}{\det(\hat{\sigma}^2(X X)^{-1})}$</td>
</tr>
<tr>
<td>DFFITS</td>
<td>$\frac{(\hat{\sigma}(i)\sqrt{h_i})}{r_i}$</td>
</tr>
<tr>
<td>PRESS(predr$_i$)</td>
<td>$\frac{1}{1 - h_i}$</td>
</tr>
</tbody>
</table>

### Classification Variables and the SPLIT Option

PROC REGSELECT supports the ability to split classification variables when you are doing model selection. You use the SPLIT option in the CLASS statement to specify that the columns of the design matrix that correspond to effects that contain a split classification variable can enter or leave a model independently of the other design columns of that effect. The following statements illustrate the use of the SPLIT option:

```plaintext
data mycas.splitExample;
    length c2 $ 6;
    drop i;
    do i=1 to 1000;
        c1 = 1 + mod(i,6);
        if i < 200 then c2 = 'low';
        else if i < 500 then c2 = 'medium';
        else c2 = 'high';
        x1 = ranuni(1);
        x2 = ranuni(1);
        y = x1+3*(c2 = 'low') + 10*(c1=3) + 5*(c1=5) + rannor(1);
    output;
    end;
run;

proc regselect data=mycas.splitExample;
    class c1(split) c2(order=freq);
    model y = c1 c2 x1 x2;
    selection method=forward;
run;
```
The “Class Level Information” table shown in Figure 19.11 is produced by default whenever you specify a CLASS statement.

Figure 19.11  Class Levels
The REGSELECT Procedure

<table>
<thead>
<tr>
<th>Class Level Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class</td>
</tr>
<tr>
<td>c1</td>
</tr>
<tr>
<td>c2</td>
</tr>
</tbody>
</table>

The SPLIT option has been specified for the classification variable c1. This permits the parameters associated with the effect c1 to enter or leave the model individually. The “Parameter Estimates” table in Figure 19.12 shows that for this example the parameters that correspond only to levels 3 and 5 of c1 are in the selected model.

Figure 19.12  Parameter Estimates

| Parameter Estimates | Standard Error | t Value | Pr > |t| |
|---------------------|----------------|---------|-------|---|
| Intercept           | 1              | 2.827805 | 0.094738 | 29.85 | <.0001 |
| c1_3                | 1              | 10.161378 | 0.087606 | 115.99 | <.0001 |
| c1_5                | 1              | 5.018482  | 0.087593 | 57.29  | <.0001 |
| c2 high             | 1              | -3.134740 | 0.084808 | -36.96 | <.0001 |
| c2 medium           | 1              | -2.923335 | 0.092537 | -31.59 | <.0001 |
| c2 low              | 0              | 0        | .      | .     |
| x1                  | 1              | 1.315202  | 0.109440 | 12.02  | <.0001 |

Using Validation and Test Data

When you have sufficient data, you can subdivide your data into three parts called the training, validation, and test data. The selection process fits models to the training data and uses the validation data to find the prediction errors for the models that are obtained in this way. This prediction error on the validation data can be used to decide when to terminate the selection process or to decide what effects to include as the selection process proceeds. Finally, after a selected model has been obtained, the test data can be used to assess how the selected model generalizes on data that played no role in selecting the model.

In some cases you might want to use only training and test data. For example, you might want to use an information criterion to decide what effects to include and when to terminate the selection process. In this case no validation data are required, but test data can still help you assess the predictive performance of the selected model. In other cases you might decide to use validation data during the selection process but forgo assessing the selected model on test data. Hastie, Tibshirani, and Friedman (2001) note that it is difficult to give a general rule for how many observations you should assign to each role. They state that a typical split might be 50% for training and 25% each for validation and testing.

You use a PARTITION statement to logically subdivide the DATA= data table into separate roles. You can name the fractions of the data that you want to reserve as test data and validation data. For example, the
following statements randomly subdivide the `inData` data table, reserving 50% for training and 25% each for validation and testing:

```plaintext
proc regselect data=mycas.inData;
   partition fraction(test=0.25 validate=0.25);
   ...
run;
```

In some cases you might need to exercise more control over the partitioning of the input data table. You can do this by naming both a variable in the input data table and a formatted value of that variable that correspond to each role. For example, the following statements assign roles to the observations in the `inData` data table based on the value of the variable `group` in that data table. Observations in which the value of `group` is “group 1” are assigned to testing, and those whose value is “group 2” are assigned to training. All other observations are ignored.

```plaintext
proc regselect data=mycas.inData;
   partition roleVar=group(test='group 1' train='group 2')
   ...
run;
```

After you reserve observations for training, validation, and testing, a model fit of the training data is scored on the validation and test data, and the average squared error (ASE) is computed separately for each of these subsets. The ASE for each data role is the error sum of squares for observations in that role divided by the number of observations in that role.

**Using the Validation ASE as the STOP= Criterion**

If you have provided observations for validation, then you can specify `STOP=VALIDATE` as a suboption of the `METHOD=` option in the `SELECTION` statement. At step $k$ of the selection process, the best candidate effect to enter or leave the current model is determined. Here, “best candidate” means the effect that gives the best value of the `SELECT=` criterion; this criterion does not need to be based on the validation data. The validation ASE for the model with this candidate effect added or removed is computed. If this validation ASE is greater than the validation ASE for the model at step $k$, then the selection process terminates at step $k$.

**Using the Validation ASE as the CHOOSE= Criterion**

When you specify the `CHOOSE=VALIDATE` suboption of the `METHOD=` option in the `SELECTION` statement, the validation ASE is computed for the models at each step of the selection process. The smallest model at any step that yields the smallest validation ASE is selected.

**Computational Method**

**Multithreading**

The `REGSELECT` procedure allocates data to different threads and calculates crossproduct matrices by accumulating the contributions from all threads. PROC `REGSELECT` also uses multiple threads to compute matrix inverses and select candidates during model selection. For more information about how PROC `REGSELECT` uses threads, see the section “Multithreading” on page 81 in Chapter 3, “Shared Concepts.”
Linear hypotheses for $\beta$ are expressed in matrix form as

$$H_0: L\beta = c$$

where $L$ is a matrix of coefficients for the linear hypotheses and $c$ is a vector of constants. The vector of regression coefficients $\hat{\beta}$ includes slope parameters and intercept parameters. The $F$ statistic for testing $H_0$ is computed as

$$F = (L\hat{\beta} - c)'[L\hat{\Sigma}(\hat{\beta})L']^{-1}(L\hat{\beta} - c)/r$$

where $\hat{\Sigma}(\hat{\beta}) = \hat{\delta}^2(X'WX)^{-1}$ is the estimated covariance matrix of $\hat{\beta}$, with $\hat{\delta}^2$ being the weighted residual sum of squares divided by the residual degrees of freedom $d$. Under $H_0$, $F$ has an asymptotic $F$ distribution with $r$ and $d$ degrees of freedom, where $r$ is the rank of $L$.

For models that use less-than-full-rank parameterization (as specified by the PARAM=GLM option in the CLASS statement), a Type 3 test of an effect of interest (main effect or interaction) is a test of the Type III estimable functions that are defined for that effect. When the model contains no missing cells, the Type 3 test of a main effect is equivalent to testing the hypothesis of equal marginal means. For more information about Type III estimable functions, see the chapter “The GLM Procedure” and the section “The Four Types of Estimable Functions” in SAS/STAT User’s Guide. Also see Littell, Freund, and Spector (1991).

For models that use full-rank parameterization, all parameters are estimable when there are no missing cells, so it is unnecessary to define estimable functions. The standard test of an effect of interest in this case is the joint test that the values of the parameters associated with that effect are 0. For a model that uses effects parameterization (as specified by the PARAM=EFFECT option in the CLASS statement), the joint test for a main effect is equivalent to testing the equality of marginal means. For a model that uses reference parameterization (as specified by the PARAM=REF option in the CLASS statement), the joint test is equivalent to testing the equality of cell means at the reference level of the other model effects. For more information about the coding scheme and the associated interpretation of results, see Muller and Fetterman (2002, Chapter 14).

If there is no interaction term, the Type 3 test of an effect for a model that uses GLM parameterization is the same as the joint test of the effect for the model that uses full-rank parameterization. In this situation, the joint test is also called the Type 3 test. For a model that contains an interaction term and no missing cells, the Type 3 test of a component main effect under GLM parameterization is the same as the joint test of the component main effect under effect parameterization. Both test the equality of cell means. But this Type 3 test differs from the joint test under reference parameterization, which tests the equality of cell means at the reference level of the other component main effect. If some cells are missing, you can obtain meaningful tests only by testing a Type III estimation function, so in this case you should use GLM parameterization.

The results of a Type 3 test or a joint test do not depend on the order in which you specify the terms in the MODEL statement.

**Displayed Output**

The following sections describe the output produced by PROC REGSELECT. The output is organized into various tables, which are discussed in their order of appearance.
**Selection Information**

When you specify the `SELECTION` statement, the REGSELECT procedure produces by default a series of tables that display information about the model selection. The “Selection Information” table informs you about the model selection method; select, stop, and choose criteria; and other parameters that govern the selection. You can suppress this table by specifying `DETAILS=NONE` in the `SELECTION` statement.

**Number of Observations**

The “Number of Observations” table displays the number of observations read from the input data table and the number of observations used in the analysis. If you specify a `FREQ` statement, this table also displays the sum of frequencies read and used. If you use a `PARTITION` statement, the table also displays the number of observations used for each data role.

**Class Level Information**

The “Class Level Information” table lists the levels of every variable that you specify in the `CLASS` statement. You should check this information to make sure that the data are correct. You can adjust the order of the `CLASS` variable levels by specifying the `ORDER=` option in the `CLASS` statement. You can suppress the “Class Level Information” table completely or partially by specifying the `NOCLPRINT=` option in the `PROC REGSELECT` statement.

If the classification variables use a nonsingular parameterization, the “Class Level Information” table also displays the reference value for each variable.

**Dimensions**

The “Dimensions” table displays the number of effects and the number of parameters from which the selected model is chosen. If you use split classification variables, then this table also displays the number of effects after splitting is taken into account.

**Entry and Removal Candidates**

When you specify the `DETAILS=ALL` or `DETAILS=STEPS` option in the `SELECTION` statement, the REGSELECT procedure produces “Entry Candidates” and “Removal Candidates” tables that display the effect names and values of the criterion used to select entering or departing effects at each step of the selection process. The effects are displayed in sorted order from best to worst of the selection criterion.

**Selection Summary**

When you specify the `SELECTION` statement, the REGSELECT procedure produces the “Selection Summary” table, which displays information about the sequence of steps of the selection process. For each step, the effect that was entered or dropped is displayed along with the statistics used to select the effect, stop the selection, and choose the selected model. For all criteria that you can use for model selection, the steps at which the optimal values of these criteria occur are also indicated.

You can suppress the display of the “Selection Summary” table by specifying `DETAILS=NONE` in the `SELECTION` statement.
Stop Reason

The “Stop Reason” table displays the reason why the selection stopped. To facilitate programmatic use of this table, an integer code is assigned to each reason and is included if you use an ODS OUTPUT statement or a DISPLAYOUT statement to output this table. The reasons and their associated codes follow:

<table>
<thead>
<tr>
<th>Code</th>
<th>Stop Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>All eligible effects are in the model.</td>
</tr>
<tr>
<td>2</td>
<td>All eligible effects have been removed.</td>
</tr>
<tr>
<td>3</td>
<td>Specified maximum number of steps done.</td>
</tr>
<tr>
<td>4</td>
<td>The model contains the specified maximum number of effects.</td>
</tr>
<tr>
<td>5</td>
<td>The model contains the specified minimum number of effects.</td>
</tr>
<tr>
<td>6</td>
<td>The stopping criterion is at a local optimum.</td>
</tr>
<tr>
<td>7</td>
<td>No suitable add or drop candidate could be found.</td>
</tr>
<tr>
<td>8</td>
<td>Adding or dropping any effect does not improve the selection</td>
</tr>
<tr>
<td>9</td>
<td>No candidate meets the appropriate SLE or SLS significance</td>
</tr>
<tr>
<td>10</td>
<td>Stepwise selection is cycling.</td>
</tr>
<tr>
<td>11</td>
<td>The model is an exact fit.</td>
</tr>
<tr>
<td>12</td>
<td>Dropping an effect would result in an empty model.</td>
</tr>
</tbody>
</table>

You can suppress the display of the “Stop Reason” table by specifying DETAILS=NONE in the SELECTION statement.

Selection Reason

When you specify the SELECTION statement, the REGSELECT procedure produces a simple table that explains why the final model was selected.

You can suppress the display of the “Selection Reason” table by specifying DETAILS=NONE in the SELECTION statement.

Selected Effects

When you specify the SELECTION statement, the REGSELECT procedure produces a simple table that lists which effects were selected for the final model.

ANOVA

The “ANOVA” table displays an analysis of variance for the selected model. This table includes the following:

- the Source of the variation, Model for the fitted regression, Error for the residual error, and C Total for the total variation after correcting for the mean. The Uncorrected Total Variation is produced when you specify the NOINT option.
- the degrees of freedom (DF) associated with the source
- the Sum of Squares for the term
the Mean Square, which is the sum of squares divided by the degrees of freedom

- the $F$ Value for testing the hypothesis that all parameters are 0 except for the intercept. This is formed by dividing the mean square for Model by the mean square for Error.

- the Prob$>F$, which is the probability of getting a greater $F$ statistic than that observed if the hypothesis is true. When you do model selection, these $p$-values are usually liberal because they are not adjusted for the fact that the terms in the model have been selected.

You can request an “ANOVA” table for the model at each step of the selection process by specifying the DETAILS= option in the SELECTION statement.

**Fit Statistics**

The “Fit Statistics” table displays fit statistics for the selected model. The statistics include the following:

- Root MSE, an estimate of the standard deviation of the error term. It is calculated as the square root of the mean square error.

- R-square, a measure between 0 and 1 that indicates the portion of the (corrected) total variation attributed to the fit rather than left to residual error. It is calculated as SS(Model) divided by SS(Total) and is also called the *coefficient of determination*. It is the square of the multiple correlation—in other words, the square of the correlation between the dependent variable and the predicted values.

- Adj R-Sq, the adjusted R-square, which is a version of R-square that has been adjusted for degrees of freedom. It is calculated as

$$
\bar{R}^2 = 1 - \frac{(n - i)(1 - R^2)}{n - p}
$$

where $i$ is equal to 1 if there is an intercept and 0 otherwise, $n$ is the number of observations used to fit the model, and $p$ is the number of parameters in the model.

- the fit criteria AIC, AICC, BIC, CP, and PRESS if they are used in the selection process. See Table 19.8 for the formulas to evaluate these criteria.

- the average square errors (ASE) on the training, validation, and test data

You can request a “Fit Statistics” table for the model at each step of the selection process by specifying the DETAILS= option in the SELECTION statement.

**Parameter Estimates**

The “Parameter Estimates” table displays the parameters in the selected model and their estimates. The information displayed for each parameter in the selected model includes the following:

- the parameter label that includes the effect name and level information for effects that contain classification variables
- the degrees of freedom (DF) for the parameter. There is one degree of freedom unless the model is not full rank.
- the parameter estimate
- the standard error, which is the estimate of the standard deviation of the parameter estimate
- $t$ Value, the $t$ test that the parameter is 0. This is computed as the parameter estimate divided by the standard error.
- the Pr > |t|, the probability that a $t$ statistic would obtain a greater absolute value than that observed given that the true parameter is 0. This is the two-tailed significance probability.

When you do model selection, these $p$-values are usually liberal because they are not adjusted for the fact that the terms in the model have been selected.

You can request a “Parameter Estimates” table for the model at each step of the selection process by specifying the DETAILS= option in the SELECTION statement.

**Model Analysis of Variance (Type III)**

When you specify the SS3 option in the MODEL statement, the REGSELECT procedure produces the “Model Analysis of Variance (Type III)” table. This table displays tests that all parameters for a particular effect are equal to zero. These tests are computed by using a Type III sum of squares, which yields a balanced test of each effect, adjusted for every other effect. For more information, see the section “Joint Tests and Type 3 Tests” on page 911.

**Timing**

The “Timing” table displays the amount of time (in seconds) and the percentage of the time that PROC REGSELECT required to perform different tasks in the analysis.

**OutputCasTables Table**

The OutputCasTables table is a special table that has information about each CAS table that is created during a CAS action execution. The information for each CAS table consists of the CAS table name, the caslib in which the table resides, and the number of columns and rows in the CAS table. Because this table is not a typical ODS table that contains analytical results, you cannot include it in the `table-spec-list` in the DISPLAYOUT statement.

**ODS Table Names**

Each table that the REGSELECT procedure creates has a name associated with it. You must use this name to refer to the table when you use the DISPLAY statement, the DISPLAYOUT statement, or ODS statements. These names are listed in Table 19.10.
<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Required Statement / Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANOVA</td>
<td>Selected model ANOVA table</td>
<td>Default output</td>
</tr>
<tr>
<td>BSplineDetails</td>
<td>B-spline basis details</td>
<td>EFFECT SPLINE BASIS=BSPLINE and DETAILS</td>
</tr>
<tr>
<td>Candidates</td>
<td>Swap candidates at step</td>
<td>SELECTION DETAILS=ALL</td>
</tr>
<tr>
<td>ClassInfo</td>
<td>Level information from the CLASS statement</td>
<td>CLASS</td>
</tr>
<tr>
<td>CollectionLevelInfo</td>
<td>Level information for collection effects</td>
<td>EFFECT COLLECTION DETAILS</td>
</tr>
<tr>
<td>Dimensions</td>
<td>Model dimensions</td>
<td>Default output</td>
</tr>
<tr>
<td>EntryCandidates</td>
<td>Candidates for entry at step</td>
<td>SELECTION DETAILS=ALL</td>
</tr>
<tr>
<td>FitStatistics</td>
<td>Fit statistics</td>
<td>Default output</td>
</tr>
<tr>
<td>MMLevelInfo</td>
<td>Level information for multimember effects</td>
<td>EFFECT MULTIMEMBER DETAILS</td>
</tr>
<tr>
<td>ModelAnova</td>
<td>Model analysis of variance (Type III)</td>
<td>MODEL SS3</td>
</tr>
<tr>
<td>NObs</td>
<td>Number of observations read and used</td>
<td>Default output</td>
</tr>
<tr>
<td>OutputCasTables</td>
<td>See the section “OutputCasTables Table” on page 915</td>
<td>OUTPUT DISPLAYOUT</td>
</tr>
<tr>
<td>ParameterEstimates</td>
<td>Solutions for the parameter estimates associated with effects in the MODEL statement</td>
<td>Default output</td>
</tr>
<tr>
<td>PolyDetails</td>
<td>Number of variables and columns, polynomial degree, and standardization method</td>
<td>EFFECT POLYNOMIAL DETAILS</td>
</tr>
<tr>
<td>PolyScaling</td>
<td>Centering and scaling details</td>
<td>EFFECT POLYNOMIAL DETAILS</td>
</tr>
<tr>
<td>RemovalCandidates</td>
<td>Candidates for removal at step</td>
<td>SELECTION DETAILS=ALL</td>
</tr>
<tr>
<td>SelectedEffects</td>
<td>List of selected effects</td>
<td>SELECTION</td>
</tr>
<tr>
<td>SelectionInfo</td>
<td>Information about selection settings</td>
<td>Default output</td>
</tr>
<tr>
<td>SelectionReason</td>
<td>Reason for selecting the final model</td>
<td>SELECTION</td>
</tr>
<tr>
<td>SelectionSummary</td>
<td>Summary information about the model selection steps</td>
<td>SELECTION</td>
</tr>
</tbody>
</table>
Table 19.10  continued

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Required Statement / Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>SplineKnots</td>
<td>Knot and boundary knot values</td>
<td>EFFECT SPLINE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BASIS=BSPLINE and DETAILS</td>
</tr>
<tr>
<td>StopReason</td>
<td>Reason selection was terminated</td>
<td>SELECTION</td>
</tr>
<tr>
<td>Timing</td>
<td>Timing breakdown by task</td>
<td>Default output</td>
</tr>
<tr>
<td>TPFSplineDetails</td>
<td>Truncated power function spline basis details</td>
<td>EFFECT SPLINE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BASIS=TPF or</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BASIS=NATURALCUBIC and DETAILS</td>
</tr>
</tbody>
</table>

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in the “Statistical Graphics Using ODS” chapter in SAS/STAT User’s Guide.

Before you create graphs, ODS Graphics must be enabled (for example, with the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section “Enabling and Disabling ODS Graphics” in that chapter.

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section “A Primer on ODS Statistical Graphics” in that chapter.

When ODS Graphics is enabled, the SELECTION statement can produce plots to help evaluate the selection process. For information about these plots, see the section “SELECTION Statement” on page 36 in Chapter 3, “Shared Concepts.”

PROC REGSELECT assigns a name to each graph that it creates using ODS. You can use these names to reference the graphs when using ODS. The names are listed in Table 19.11.

Table 19.11  Graphs Produced by PROC REGSELECT

<table>
<thead>
<tr>
<th>ODS Graph Name</th>
<th>Plot Description</th>
<th>PLOTS Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoefficientPanel</td>
<td>Coefficients and CHOOSE= criterion by step</td>
<td>COEFFICIENTS</td>
</tr>
<tr>
<td>ChooseCriterionPlot</td>
<td>CHOOSE= criterion by step</td>
<td>COEFFICIENTS(UNPACK)</td>
</tr>
<tr>
<td>CoefficientPlot</td>
<td>Coefficients by step</td>
<td>COEFFICIENTS(UNPACK)</td>
</tr>
</tbody>
</table>
### Table 19.11 continued

<table>
<thead>
<tr>
<th>ODS Graph Name</th>
<th>Plot Description</th>
<th>PLOTS= Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>CriterionPanel</td>
<td>Fit criteria by step</td>
<td>CRITERIA</td>
</tr>
<tr>
<td>AdjRSqPlot</td>
<td>Max-rescaled (adjusted) R-square by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>AICCCPlot</td>
<td>Corrected Akaike’s information criterion by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>AICPlot</td>
<td>Akaike’s information criterion by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>CPPPlot</td>
<td>Mallows’ $C_p$ statistic by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>PRESSPlot</td>
<td>PRESS criterion by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>SBCPlot</td>
<td>Schwarz Bayesian information criterion by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>TEST_ASEPlot</td>
<td>Average square error on testing data by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>TRAIN_ASEPlot</td>
<td>Average square error on training data by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>VAL_ASEPlot</td>
<td>Average square error on validation data by step</td>
<td>CRITERIA(UNPACK)</td>
</tr>
<tr>
<td>FitByRolePlot</td>
<td>Progression of average square error by role</td>
<td>FITBYROLE</td>
</tr>
</tbody>
</table>

### Examples: REGSELECT Procedure

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

#### Example 19.1: Model Selection with Validation

This example is based on the example “Using Validation and Cross Validation” in the documentation for the GLMSELECT procedure in SAS/STAT software. The current example shows how you can use validation data to monitor and control variable selection. It also demonstrates the use of split classification variables.

The following DATA step produces analysis data that contain a variable that you can use to assign observations to the training, validation, and testing roles. In this case, each role has 5,000 observations.

```sas
data mycas.analysisData;
  drop i j c3Num;
  length c3$ 7;
```
array x(20) x1-x20;

do i=1 to 15000;
   do j=1 to 20;
      x{j} = ranuni(1);
   end;

   c1 = 1 + mod(i,8);
   c2 = ranbin(1,3,.6);

   if i < 50 then do; c3 = 'tiny'; c3Num=1;end;
   else if i < 250 then do; c3 = 'small'; c3Num=1;end;
   else if i < 600 then do; c3 = 'average'; c3Num=2;end;
   else if i < 1200 then do; c3 = 'big'; c3Num=3;end;
   else do; c3 = 'huge'; c3Num=5;end;

   yTrue = 10 + x1 + 2*x5 + 3*x10 + 4*x20 + 3*x1*x7 + 8*x6*x7
          + 5*(c1=3)*c3Num + 8*(c1=7);

   error = 5*rannor(1);

   y = yTrue + error;

   if mod(i,3)=1 then Role = 'TRAIN';
   else if mod(i,3)=2 then Role = 'VAL';
   else Role = 'TEST';

   output;
end;
run;

By construction, the true model consists of the main effects $x_1$, $x_5$, $x_{10}$, $x_{20}$, and $c_1$ and the interaction effects $x_1*x_7$, $x_6*x_7$, and $c_1*c_3$. Furthermore, you can see that only levels 3 and 7 of the classification variable $c_1$ are systematically related to the response.

Because the error term for each observation is five times a value drawn from a standard normal distribution, the expected error variance is 25. For the data in each role, you can compute an estimate of this error variance by forming the average square error (ASE) for the observations in the role. Output 19.1.1 shows the ASE for each role that you can compute with the following statements:

```sas
proc summary data=mycas.analysisData;
   class role;
   ways 1;
   var error;
   output out=ASE uss=uss n=n;
data ASE; set ASE;
   OracleASE = uss / n;
   label OracleASE = 'Oracle ASE';
   keep Role OracleASE;
proc print data=ASE label noobs;
run;
```
The ASE values shown Output 19.1.1 are labeled as “Oracle ASE” because you need to know the true underlying model if you want to compute these values from the response and underlying regressors. In a modeling context, a good predictive model produces values that are close to these oracle values. An overfit model produces a smaller ASE for the training data but higher values for the validation and test data. An underfit model exhibits higher values for all data roles.

Suppose you suspect that the dependent variable depends on both main effects and two-way interactions. You can use the following statements to select a model:

```plaintext
ods graphics on;
proc regselect data=mycas.analysisData;
   partition roleVar=role(train='TRAIN' validate='VAL' test='TEST');
   class c1 c2 c3;
   model y = c1|c2|c3|x1|x2|x3|x4|x5|x6|x7|x8|x9|x10|x11|x12|x13|x14|x15|x16|x17|x18|x19|x20 @2 /stb;
   selection method = stepwise(select=sl sle=0.1 sls=0.15 choose=validate)
      hierarchy=single details=steps plots(startstep=5)=all;
run;
```

A PARTITION statement assigns observations to training, validation, and testing roles based on the values of the input variable named role. The SELECTION statement requests stepwise selection based on significance level, where the SLE and SLS values are set to use the defaults of PROC REGSELECT. The CHOOSE=VALIDATE option selects the model that yields the smallest ASE value for the validation data.

The “Number of Observations” table in Output 19.1.2 confirms that there are 5,000 observations for each data role. The “Dimensions” table shows that the selection is from 278 effects with a total of 661 parameters.

```
Output 19.1.2  Number of Observations, Class Levels, and Dimensions

The REGSELECT Procedure

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Observations Read</td>
<td>15000</td>
</tr>
<tr>
<td>Number of Observations Used</td>
<td>15000</td>
</tr>
<tr>
<td>Number of Observations Used for Training</td>
<td>5000</td>
</tr>
<tr>
<td>Number of Observations Used for Validation</td>
<td>5000</td>
</tr>
<tr>
<td>Number of Observations Used for Testing</td>
<td>5000</td>
</tr>
</tbody>
</table>
```
Output 19.1.2 continued

Class Level Information

<table>
<thead>
<tr>
<th>Class</th>
<th>Levels</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>c1</td>
<td>8</td>
<td>1 2 3 4 5 6 7 8</td>
</tr>
<tr>
<td>c2</td>
<td>4</td>
<td>0 1 2 3</td>
</tr>
<tr>
<td>c3</td>
<td>5</td>
<td>average big huge small tiny</td>
</tr>
</tbody>
</table>

Dimensions

<table>
<thead>
<tr>
<th>Dimensions</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Effects</td>
<td>278</td>
</tr>
<tr>
<td>Number of Parameters</td>
<td>661</td>
</tr>
</tbody>
</table>

Output 19.1.3 shows the “Selection Summary” table. You see that 18 steps are done, at which point all effects in the model are significant at the SLS value of 0.15 and all the remaining effects if added individually would not be significant at the SLE significance level of 0.1. However, because you specified the CHOOSE=VALIDATE option, the model at step 18 is not used as the selected model. Instead the model at step 10 (where the validation ASE achieves a local minimum value) is selected. The “Stop Reason,” “Selection Reason,” and “Selected Effects” in Output 19.1.4 tables provide this information.

Output 19.1.3 Selection Summary

The REGSELECT Procedure

Selection Details

<table>
<thead>
<tr>
<th>Selection Summary</th>
<th>Effect Entered</th>
<th>Number Effects In</th>
<th>Validation ASE</th>
<th>p Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 Intercept</td>
<td>1</td>
<td>98.3895</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 c1</td>
<td>2</td>
<td>34.8572 &lt;.0001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 x7</td>
<td>3</td>
<td>32.5531 &lt;.0001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 x6</td>
<td>4</td>
<td>31.0646 &lt;.0001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 x20</td>
<td>5</td>
<td>29.7078 &lt;.0001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 x6*x7</td>
<td>6</td>
<td>29.2210 &lt;.0001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 x10</td>
<td>7</td>
<td>28.6683 &lt;.0001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7 x1</td>
<td>8</td>
<td>28.3250 &lt;.0001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 x5</td>
<td>9</td>
<td>27.9766 &lt;.0001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9 c3</td>
<td>10</td>
<td>27.8288 &lt;.0001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 c1*c3</td>
<td>11</td>
<td>25.9701* &lt;.0001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11 x10*c1</td>
<td>12</td>
<td>26.0696 0.0109</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12 x4</td>
<td>13</td>
<td>26.1594 0.0128</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13 x4*x10</td>
<td>14</td>
<td>26.1814 0.0035</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14 x20*c1</td>
<td>15</td>
<td>26.3294 0.0156</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15 x1*c3</td>
<td>16</td>
<td>26.3945 0.0244</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16 x1*x7</td>
<td>17</td>
<td>26.3632 0.0270</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17 x7*x10</td>
<td>18</td>
<td>26.4120 0.0313</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18 x1*x20</td>
<td>19</td>
<td>26.4330 0.0871</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* Optimal Value Of Criterion
Output 19.1.4 Stopping and Selection Reasons

Selection stopped because all candidates for removal are significant at the 0.15 level and no candidate for entry is significant at the 0.1 level.

The model at step 10 is selected where Validation ASE is 25.9701.

Selected Effects: Intercept c1 c3 c1*c3 x1 x5 x6 x7 x6*x7 x10 x20

You can see that the selected effects include all the main effects in the true model and two of the three true interaction terms. Furthermore, the selected model does not include any variables that are not in the true model. Note that these statements are not true of the larger model at the final step of the selection process.

When you enable ODS Graphics, the PLOTS=ALL option in the SELECTION statement produces plots that illuminate the selection process. The STARTSTEP=5 suboption in the PLOTS= option requests that only steps 5 and beyond be displayed in the plots. This enables you to more easily assess how the fit criteria change near the selected step.

The upper plot in the coefficient panel in Output 19.1.5 shows how the standardized parameter estimates change with the selection step after step 5. The lower plot in the panel shows how the average square error on the validation data, which you specified as the CHOOSE= criterion, changes with the selection step.

Output 19.1.5 Coefficient Panel
The criterion panel in Output 19.1.6 shows how the various fit criteria change as the stepwise selection method proceeds. This plot shows that if you had used the SBC statistic as the CHOOSE= criterion, then the selected model would still be the model at step 10. However, the AIC or AICC statistic continues to decline for all steps of the selection process. If you had used either of these criteria as the CHOOSE= criterion, then the final model at step 18 would have been selected.

Output 19.1.6 Criterion Panel

Output 19.1.7 shows how the average square error compares for the training, test, and validation data. You see that the average square error on both the test and validation data starts to increase beyond the selected step. This indicates that the selected model does not overfit the training data.
Output 19.1.8 shows the fit statistics of the selected model. You can see that the ASE values for the training, validation, and test data are all similar, indicating a reasonable predictive model. In this case, where the true model is known, you can see that all three ASE values are close to the oracle values for the true model, as shown in Output 19.1.1.

Output 19.1.8  Fit Statistics for the Selected Model

<table>
<thead>
<tr>
<th>Statistics</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root MSE</td>
<td>5.03976</td>
</tr>
<tr>
<td>R-Square</td>
<td>0.74483</td>
</tr>
<tr>
<td>Adj R-Sq</td>
<td>0.74246</td>
</tr>
<tr>
<td>AIC</td>
<td>21222</td>
</tr>
<tr>
<td>AICC</td>
<td>21223</td>
</tr>
<tr>
<td>SBC</td>
<td>16527</td>
</tr>
<tr>
<td>ASE (Train)</td>
<td>25.16041</td>
</tr>
<tr>
<td>ASE (Validate)</td>
<td>25.97010</td>
</tr>
<tr>
<td>ASE (Test)</td>
<td>25.83436</td>
</tr>
</tbody>
</table>

Because you specified the DETAILS=STEPS option in the SELECTION statement, you can see the fit statistics for the model at each step of the selection process. Output 19.1.9 shows these fit statistics for the
Example 19.1: Model Selection with Validation

Final model at step 18. You see that for this model, the ASE value for the training data is smaller than the ASE values for the validation and test data. This indicates an overfit model that might not generalize well to new data. You see that the ASE values for the validation and test data are now worse in comparison to the oracle values than the values for the selected model at step 10.

**Output 19.1.9** Fit Statistics for the Model at Step 18

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Root MSE</td>
<td>5.01386</td>
</tr>
<tr>
<td>R-Square</td>
<td>0.74862</td>
</tr>
<tr>
<td>Adj R-Sq</td>
<td>0.74510</td>
</tr>
<tr>
<td>AIC</td>
<td>21194</td>
</tr>
<tr>
<td>AICC</td>
<td>21196</td>
</tr>
<tr>
<td>SBC</td>
<td>16648</td>
</tr>
<tr>
<td>ASE (Train)</td>
<td>24.78688</td>
</tr>
<tr>
<td>ASE (Validate)</td>
<td>26.43304</td>
</tr>
<tr>
<td>ASE (Test)</td>
<td>26.07078</td>
</tr>
</tbody>
</table>

Output 19.1.10 shows part of the “Parameter Estimates” table for the selected model at step 10 that includes the estimates for the main effect c1. Because you specified the STB option in the MODEL statement, this table includes standardized estimates.

**Output 19.1.10** Part of the Parameter Estimates Table for the Selected Model

| Parameter | DF | Estimate | Standardized Estimate | Standard Error | t Value | Pr > |t|
|-----------|----|----------|-----------------------|----------------|---------|------|
| Intercept | 1  | 8.558686 | 0                     | 3.581926       | 2.39    | 0.0169 |
| c1 1      | 1  | 13.177235| 0.438872              | 5.043039       | 2.61    | 0.0090 |
| c1 2      | 1  | -2.800921| -0.093285             | 4.602421       | -0.61   | 0.5428 |
| c1 3      | 1  | 9.689426 | 0.322709              | 5.043363       | 1.92    | 0.0548 |
| c1 4      | 1  | 8.182340 | 0.272515              | 5.042136       | 1.62    | 0.1047 |
| c1 5      | 1  | 3.097216 | 0.103154              | 5.041119       | 0.61    | 0.5390 |
| c1 6      | 1  | 0.543642 | 0.018106              | 5.043466       | 0.11    | 0.9142 |
| c1 7      | 1  | 11.857267| 0.394910              | 5.042602       | 2.35    | 0.0187 |
| c1 8      | 0  | 0        | 0                     | 0.0            | ...     | ...   |

The magnitudes of the standardized estimates and the t statistics of the parameters of the effect c1 reveal that only levels 3 and 7 of this effect contribute appreciably to the model. This suggests that you might obtain a more parsimonious model with similar or better predictive power if parameters that correspond to the levels of c1 can enter or leave the model independently. You request this by specifying the SPLIT option in the CLASS statement, as shown in the following statements:

```plaintext
proc regselect data=mycas.analysisData;
    partition roleVar=role(train='TRAIN' validate='VAL' test='TEST');
    class c1(split) c2 c3;
    model y = c1|c2|c3|x1|x2|x3|x4|x5|x6|x7|x8|x9|x10|x11|x12|x13|x14|x15|x16|x17|x18|x19|x20 @2 /stb;
    selection method = stepwise(select=sl sle=0.1 sls=0.15 choose=validate)
                      hierarchy=single details=steps;
run;
```
Output 19.1.11 shows the “Dimensions” table. You can see that because the columns in the design matrix that correspond to the levels of $c_1$ are treated as separate effects, the selection is now from 439 effects, even though the number of parameters is unchanged.

**Output 19.1.11  Dimensions with $c_1$ Split**

<table>
<thead>
<tr>
<th>Dimensions</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Effects</td>
<td>439</td>
</tr>
<tr>
<td>Number of Parameters</td>
<td>661</td>
</tr>
</tbody>
</table>

Output 19.1.12 shows the selected effects. You can see that, as anticipated, the selected model now depends on only levels 3 and 7 of $c_1$.

**Output 19.1.12  Selected Effects with $c_1$ Split**

Selected Effects: Intercept $c_1_3$ $c_1_7$ $c_3$ $c_1_3*c_3$ $x_1$ $x_5$ $x_6$ $x_7$ $x_6*x_7$ $x_{10}$ $x_{20}$

Finally, the fit statistics for the selected model are shown in Output 19.1.13.

**Output 19.1.13  Fit Statistics for the Selected Model with $c_1$ Split**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Root MSE</td>
<td>5.04060</td>
</tr>
<tr>
<td>R-Square</td>
<td>0.74325</td>
</tr>
<tr>
<td>Adj R-Sq</td>
<td>0.74238</td>
</tr>
<tr>
<td>AIC</td>
<td>21195</td>
</tr>
<tr>
<td>AICC</td>
<td>21195</td>
</tr>
<tr>
<td>SBC</td>
<td>16311</td>
</tr>
<tr>
<td>ASE (Train)</td>
<td>25.31622</td>
</tr>
<tr>
<td>ASE (Validate)</td>
<td>25.98055</td>
</tr>
<tr>
<td>ASE (Test)</td>
<td>25.76059</td>
</tr>
</tbody>
</table>

If you compare the ASE values for this model in Output 19.1.13 with the oracle values in Output 19.1.1 and the values for the model without splitting $c_1$ in Output 19.1.8, you see that this more parsimonious model produces the best predictive performance for the test data of all the models considered in this example.

**References**


# Chapter 20
## The SPC Procedure

<table>
<thead>
<tr>
<th>Contents</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview: SPC Procedure</td>
<td>930</td>
</tr>
<tr>
<td>Uses of Shewhart Charts</td>
<td>930</td>
</tr>
<tr>
<td>Characteristics of Shewhart Charts</td>
<td>931</td>
</tr>
<tr>
<td>Classification of Shewhart Charts</td>
<td>933</td>
</tr>
<tr>
<td>PROC SPC Compared with the SHEWHART Procedure</td>
<td>934</td>
</tr>
<tr>
<td>Using CAS Sessions and CAS Engine Librefs</td>
<td>934</td>
</tr>
<tr>
<td>Getting Started: SPC Procedure</td>
<td>935</td>
</tr>
<tr>
<td>Syntax: SPC Procedure</td>
<td>938</td>
</tr>
<tr>
<td>PROC SPC Statement</td>
<td>938</td>
</tr>
<tr>
<td>BY Statement</td>
<td>939</td>
</tr>
<tr>
<td>BOXCHART Statement</td>
<td>940</td>
</tr>
<tr>
<td>CCHART Statement</td>
<td>940</td>
</tr>
<tr>
<td>IRCHART Statement</td>
<td>941</td>
</tr>
<tr>
<td>MCHART Statement</td>
<td>942</td>
</tr>
<tr>
<td>MRCHART Statement</td>
<td>943</td>
</tr>
<tr>
<td>NPCHART Statement</td>
<td>943</td>
</tr>
<tr>
<td>PCHART Statement</td>
<td>944</td>
</tr>
<tr>
<td>RCHART Statement</td>
<td>945</td>
</tr>
<tr>
<td>SCHART Statement</td>
<td>946</td>
</tr>
<tr>
<td>UCHART Statement</td>
<td>946</td>
</tr>
<tr>
<td>XCHART Statement</td>
<td>947</td>
</tr>
<tr>
<td>XRCHART Statement</td>
<td>948</td>
</tr>
<tr>
<td>XSCHART Statement</td>
<td>949</td>
</tr>
<tr>
<td>Dictionary of Options</td>
<td>949</td>
</tr>
<tr>
<td>Details: SPC Procedure</td>
<td>956</td>
</tr>
<tr>
<td>Methods of Estimating the Standard Deviation</td>
<td>956</td>
</tr>
<tr>
<td>Constructing Charts for Individual Measurements and Moving Ranges</td>
<td>959</td>
</tr>
<tr>
<td>Constructing Charts for Means</td>
<td>960</td>
</tr>
<tr>
<td>Constructing Charts for Medians</td>
<td>961</td>
</tr>
<tr>
<td>Constructing Charts for Ranges</td>
<td>963</td>
</tr>
<tr>
<td>Constructing Charts for Standard Deviations</td>
<td>964</td>
</tr>
<tr>
<td>Constructing Charts for Numbers of Nonconformities (c Charts)</td>
<td>965</td>
</tr>
<tr>
<td>Constructing Charts for Number Nonconforming (np Charts)</td>
<td>967</td>
</tr>
<tr>
<td>Constructing Charts for Proportion Nonconforming (p Charts)</td>
<td>969</td>
</tr>
<tr>
<td>Constructing Charts for Nonconformities per Unit (u Charts)</td>
<td>971</td>
</tr>
<tr>
<td>Percentile Definitions</td>
<td>972</td>
</tr>
</tbody>
</table>
Overview: SPC Procedure

The SPC procedure performs Shewhart control chart analysis in SAS Viya.

The Shewhart control chart is a graphical and analytical tool for determining whether a process is in a state of statistical control. You can use the SPC procedure to compute many different types of control charts, including all commonly used charts for variables and attributes. In addition, you can use PROC SPC to do the following:

- analyze multiple process variables and produce a table that identifies processes that are out of statistical control
- adjust control limits to compensate for unequal subgroup sizes
- estimate control limits from the data, compute control limits from specified values for population parameters (known standards), or read limits from an input data table
- perform tests for special causes based on runs patterns (Western Electric rules)
- estimate the process standard deviation by using various methods (variable charts only)
- specify process specification limits and compute capability indices for processes that are in statistical control
- save chart statistics and control limits in output data tables that can be used as input for the SHEWHART procedure in SAS/QC software

Uses of Shewhart Charts

The Shewhart chart is named after Walter A. Shewhart (1891–1967), a physicist at Bell Telephone Laboratories, who introduced the method in 1924 and elaborated on it in his book *Economic Control of Quality of Manufactured Product* (1931). The concepts that underlie the control chart are that the natural variability in any process can be quantified by using a set of control limits and that variation exceeding these limits signals a change in the process. Although control charts were developed to improve manufacturing processes, they are used in many other fields, including finance and health care.
In industry, the Shewhart chart is the most commonly applied statistical quality control method of studying the variation in output from a manufacturing process. Shewhart charts are typically used to distinguish variation due to special causes from variation due to common causes. Special causes, also referred to as assignable causes, are local, sporadic problems such as the failure of a particular machine or a mistakenly recorded measurement. Common causes are problems inherent in the system as a whole, such as inadequate product design, inherited defective material, and excessive humidity.

After the special causes have been identified and eliminated, the process is said to be in statistical control. When statistical control has been established, Shewhart charts can be used to monitor the process for the occurrence of future special causes and to measure and reduce the effects of common causes.

Deming (1982) emphasized that the improvement of a process can begin only after statistical control has been established. Deming also noted that control chart techniques are applicable to quality improvement in service industries as well as manufacturing industries.

**Characteristics of Shewhart Charts**

Figure 20.1 illustrates a typical Shewhart chart.

![A Shewhart Control Chart](image_url)
All Shewhart charts have the following characteristics:

- Each point represents a summary statistic that is computed from a sample of measurements of a quality characteristic. For example, the summary statistic might be the average value of a critical dimension of five items selected at random, or it might be the proportion of nonconforming items in a sample of 100 items.

- The samples from which the summary statistics are computed are referred to as rational subgroups or subgroup samples. The organization of the data into subgroups is critical to the interpretation of a Shewhart chart. Shewhart (1931) advocated selecting rational subgroups so that variation within subgroups is minimized and variation among subgroups is maximized; this makes the chart more sensitive to shifts in the process level. Various approaches to subgrouping are discussed by Grant and Leavenworth (1988), Montgomery (1996), and Kume (1985).

- The horizontal axis of a Shewhart chart identifies the subgroup samples. Frequently, the samples are indexed according to the order in which they were taken or the time at which they were taken. Subgroup samples can also be assigned labels that indicate some other type of classification (such as lot number).

- The central line on a Shewhart chart indicates the average (expected value) of the summary statistic when the process is in statistical control.

- The upper and lower control limits, labeled UCL and LCL, respectively, indicate the range of variation to be expected in the summary statistic when the process is in statistical control. The control limits are commonly computed as 3σ limits, which represent three standard errors of variation in the summary statistic above and below the central line. However, the limits can also be determined using a multiple of the standard error other than three.

The control limits are also determined by the subgroup sample size, because the standard error of the summary statistic is a function of sample size. If the sample size is constant across subgroups, the control limits are typically horizontal lines, as in Figure 20.1. However, if the sample size varies from subgroup to subgroup, the limits are usually adjusted to compensate for the effect of sample size, resulting in steplike boundaries.

Control limits can be estimated from the data that you are analyzing, or they can be standard, previously determined values. Estimated limits are often used when you are establishing statistical control, and standard limits are often used when you are maintaining statistical control.

- A point outside the control limits signals the presence of a special cause of variation. In addition, tests for special causes (also referred to as Western Electric rules and runs tests) can signal an out-of-control condition if a statistically unusual pattern of points is observed in the control chart. For example, one pattern that is commonly used to diagnose the existence of a trend is seven consecutive steadily increasing points.

When the process is in statistical control, a point can fall outside the control limits purely by chance, resulting in a false out-of-control signal. However, when the Shewhart chart correctly signals the presence of a special cause, you must take additional action to determine the nature of the problem and eliminate it.

---

1 In this context, σ always stands for the standard error of the subgroup summary statistic that is plotted on the chart. Elsewhere in this section, σ is also used to denote the standard deviation of a process, also referred to as the population standard deviation. This dual usage is standard practice.

2 The term standard deviation is also used by some authors to refer to this quantity; see, for example, Montgomery (1996). This section uses the term standard error for the dispersion of the distribution of a statistic and the term standard deviation for the dispersion of a distribution of individual measurements.
Classification of Shewhart Charts

Shewhart charts are broadly classified according to the type of data that you are analyzing:

- Shewhart charts for variables are used when you are measuring the quality characteristic of a process on a continuous scale.
- Shewhart charts for attributes are used when you are measuring the quality characteristic of a process by counting the number of nonconformities (defects) in an item or the number of nonconforming (defective) items in a sample.

Shewhart charts for variables are further classified according to the subgroup summary statistic that is plotted on the chart:

- $\bar{X}$ and $R$ charts display subgroup means (averages) and ranges. You usually present the two charts on the same page, aligning the $\bar{X}$ chart above the $R$ chart to facilitate the simultaneous analysis of the central tendency and variability of the process.
- $\bar{X}$ and $s$ charts display subgroup means (averages) and standard deviations. You usually present the two charts on the same page, aligning the $\bar{X}$ chart above the $s$ chart.
- Median and range charts display subgroup medians and ranges. You usually present the two charts on the same page, aligning the median chart above the $R$ chart.
- Charts for individual measurements and moving ranges display individual measurements and moving ranges of two or more successive measurements. In this case the subgroup sample consists of a single observation.
- Box charts display a box plot (box-and-whisker plot) for each subgroup and control limits for the subgroup means or medians. This type of chart facilitates detailed analysis of the subgroup distributions and is applicable to large subgroup sample sizes (10 or more).

Likewise, Shewhart charts for attributes are classified according to the subgroup summary statistic that is plotted on the chart:

- A $p$ chart displays the proportion of nonconforming (defective) items in a subgroup sample.
- An $np$ chart displays the number of nonconforming (defective) items in a subgroup sample.
- A $u$ chart displays the number of nonconformities (defects) per unit in a subgroup sample that consists of an arbitrary number of units.
- A $c$ chart displays the number of nonconformities (defects) in a unit (here, a subgroup sample typically consists of one unit).

You can produce all the preceding types of Shewhart charts by using the SPC procedure.
PROC SPC Compared with the SHEWHART Procedure

The SPC procedure performs univariate control chart analysis that is comparable to that of the SHEWHART procedure in SAS/QC. Whereas PROC SHEWHART produces graphical control charts, PROC SPC produces tables that summarize signals of process instability. PROC SPC also produces output data tables that you can read into SAS data sets and use as input to PROC SHEWHART.

PROC SPC is intended to perform control chart analysis on many process variables in parallel. Its input data format and syntax are more conducive to processing large numbers of processes than those of PROC SHEWHART. You can use PROC SPC to produce concise summaries of process health for a large number of processes. After you identify potential problems, you can use PROC SHEWHART to produce control chart graphs of selected processes.

Using CAS Sessions and CAS Engine Librefs

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:

```
proc options option=(CASHOST CASPORT);
run;
```

In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:

```
cas mysess;
libname mycas cas sessref=mysess;
```

The CAS statement creates the CAS session named mysess, and the LIBNAME statement creates the mycas CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the CASHOST and CASPORT of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the mysess session, you can terminate it by using the TERMINATE option in the CAS statement as follows:

```
cas mysess terminate;
```

For more information about the CAS and LIBNAME statements, see the section “Introduction to Shared Concepts” on page 10 in Chapter 3, “Shared Concepts.”
A brewery monitors its bottling process to ensure that each bottle is filled with the proper amount of beer. The following statements create a SAS data set named Beer that contains the amount of beer, recorded in fluid ounces, for 23 sample batches:

```sas
data Beer;
  input Batch size @;
  do i=1 to size;
    input Amount @@;
    output;
  end;
  drop i size;
  label Batch = 'Batch Number';
datalines;
 1  5 12.01 11.97 11.93 11.98 12.00
 2  5 11.88 11.98 11.93 12.03 11.92
 3  5 11.93 11.99 12.00 12.03 11.95
 4  5 11.98 11.94 12.02 11.90 11.97
 5  5 12.02 12.02 11.98 12.04 11.90
 6  4 11.98 11.98 12.00 11.93
 7  5 11.93 11.95 12.02 11.91 12.03
 8  5 12.00 11.98 12.02 11.89 12.01
 9  5 11.98 11.93 11.99 12.02 11.91
10  5 11.97 12.02 12.05 12.01 11.97
11  5 12.02 12.01 11.97 12.02 11.94
12  5 11.93 11.83 11.99 12.02 12.01
13  5 12.01 11.98 11.94 12.04 12.01
14  5 11.98 11.96 12.02 12.00 12.00
15  5 11.97 11.99 12.03 11.95 11.96
16  5 11.99 11.95 11.96 12.03 12.01
17  4 11.99 11.97 12.03 12.01
18  5 11.94 11.96 11.98 12.03 11.97
19  5 11.97 11.87 11.90 12.01 11.95
20  5 11.96 11.94 11.96 11.98 12.05
21  3 12.06 12.07 11.98
22  5 12.01 11.98 11.96 11.97 12.00
23  5 12.00 12.02 12.03 11.99 11.96
;```

The format of the Beer data set makes it suitable for analysis by PROC SHEWHART. The process variable is Amount, and the subgroup variable is Batch. Figure 20.2 shows a partial listing of the data set.
The following DATA step modifies `Beer` so that it is suitable for analysis by PROC SPC:

```plaintext
data Beer;
  length processname $16 subgroupname $8;
  processname='Amount';
  subgroupname='Batch';
  set Beer(rename=(Amount=process Batch=subgroup));
run;
```

The modified data set is shown in Figure 20.3.
The PROC SPC data format is more conducive than the PROC SHEWHART format to analyzing many processes at once. Because process names are recorded in the process name variable, you do not have to list the processes to be analyzed in the procedure syntax.

The following statements create a data table called mycas.AllProcesses that contains the beer bottle data along with data from seven other SAS data sets (not shown here) that appear in the SHEWHART procedure chapter of SAS/QC User’s Guide. These statements assume that the CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

```
data mycas.AllProcesses;
  set Beer Detergent Disks Partgaps Times Turbine Wafers Wire;
  format subgroup;
run;
```

The following statements use PROC SPC to perform a means and ranges chart analysis of the eight processes:

```
proc spc data=mycas.AllProcesses;
  xrchart;
run;
```

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

The XRCHART statement performs an $\bar{X}$ and $R$ chart analysis on the process data, calculating control charts for subgroup means and subgroup ranges. By default, control limits are calculated from the data. Figure 20.4 shows the resulting output.

![Figure 20.4 PROC SPC Output](image)

**The SPC Procedure**

<table>
<thead>
<tr>
<th>processname</th>
<th>subgroupname</th>
<th>Number of Subgroups</th>
<th>Number &gt; UCL</th>
<th>Number &lt; LCL</th>
<th>Number &gt; UCL</th>
<th>Number &lt; LCL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amount</td>
<td>Batch</td>
<td>23</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Breakstrength</td>
<td>Sample</td>
<td>25</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Delay</td>
<td>Day</td>
<td>11</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Diameter</td>
<td>Batch</td>
<td>45</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>KWatts</td>
<td>Day</td>
<td>40</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Partgap</td>
<td>Sample</td>
<td>41</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Time</td>
<td>Lot</td>
<td>25</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Weight</td>
<td>Lot</td>
<td>28</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

The default output from PROC SPC is an exception summary table. For each process, the table shows the number of subgroups that were analyzed and the number of subgroup summary statistics that fell outside the control limits of each chart.

The table in Figure 20.4 shows signals of unusual variation in process Delay (from the Times data set) and process Diameter (from the Wafers data set). When unusual variation is signaled in a process, you should investigate to find and correct any special causes of variation.
Syntax: SPC Procedure

PROC SPC < options > ;
   BY variables ;
   BOXCHART < / options > ;
   CCHART < / options > ;
   IRCHART < / options > ;
   MCHART < / options > ;
   MRCHART < / options > ;
   NPCHART < / options > ;
   PCHART < / options > ;
   RCHART < / options > ;
   SCHART < / options > ;
   UCHART < / options > ;
   XCHART < / options > ;
   XRCHART < / options > ;
   XSCHART < / options > ;

The PROC SPC statement invokes the procedure.

You use the chart statements to perform control chart analysis. You can specify a BY statement to obtain separate analyses for each BY group.

PROC SPC Statement

PROC SPC < options > ;

The PROC SPC statement is required in order to invoke the SPC procedure. By itself, it does not perform any statistical process control analysis.

You can specify the following options in the PROC SPC statement:

DATA=CAS-libref.data-table

names the input data table for PROC SPC to use. The default is the most recently created data table. CAS-libref.data-table is a two-level name, where

CAS-libref refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to the data, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about CAS-libref, see the section “Using CAS Sessions and CAS Engine Librefs” on page 934.

data-table specifies the name of the input data table.

For more information, see the section “DATA= Data Table” on page 980.
**LIMITS=** `CAS-libref.data-table`

names an input data table that contains preestablished control limits or the parameters from which control limits can be computed. Each observation in a **LIMITS=** data table provides control limit information for a process. `CAS-libref.data-table` is a two-level name, where `CAS-libref` refers to the caslib and session identifier, and `data-table` specifies the name of the input data table. For more information about this two-level name, see the DATA= option and the section “Using CAS Sessions and CAS Engine Librefs” on page 934.

You can create this data table as an **OUTLIMITS=** data table in a previous run of PROC SPC. For more information, see the section “LIMITS= Data Table” on page 981.

**NTHREADS=n**

specifies the number of threads per node to be used to perform the analysis. You can specify a number between 1 and 64, inclusive. By default, n is the maximum number of threads available on each node.

**PROCESSNAME=** `variable`

specifies the variable in the input data table that contains the names of processes to be analyzed. Each unique combination of process name and subgroup name in the input data table identifies the data for one control chart. By default, PROC SPC looks for a variable named ProcessName that contains the process names.

**PROCESSVALUE=** `variable`

specifies the variable in the input data table that contains the process measurements to be analyzed. By default, PROC SPC looks for a variable named Process that contains the process measurements.

**SUBGROUPNAME=** `variable`

specifies the variable in the input data table that contains the names of subgroup variables. Each unique combination of process name and subgroup name in the input data table identifies the data for one control chart. By default, PROC SPC looks for a variable named SubgroupName that contains the subgroup variable names.

**SUBGROUPVALUE=** `variable`

specifies the variable in the input data table that contains the subgroup values. Subgroup values identify rational subgroups for control chart analysis. By default, PROC SPC looks for a variable named Subgroup that contains the subgroup values.

**BY Statement**

```sas
BY variables;
```

You can specify a BY statement in PROC SPC to obtain separate analyses of observations in groups that are defined by the values of the BY variables. If you specify more than one BY statement, only the last one specified is used. For more information, see the discussion of BY-group processing in SAS Language Reference: Concepts.
BOXCHART Statement

BOXCHART < / options > ;

The BOXCHART statement produces control charts for subgroup means or medians, along with statistics that describe the distribution of data within each subgroup. Table 20.1 lists the options that you can specify in a BOXCHART statement. For detailed descriptions of these options, see the section “Dictionary of Options” on page 949.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLN</td>
<td>Includes all subgroups in the analysis, regardless of sample size</td>
</tr>
<tr>
<td>CIINDICES</td>
<td>Computes capability index confidence limits</td>
</tr>
<tr>
<td>CONTROLSTAT=</td>
<td>Specifies whether control limits are computed for subgroup means or medians</td>
</tr>
<tr>
<td>EXCHART</td>
<td>Produces results only for charts in which exceptions occur</td>
</tr>
<tr>
<td>LIMITN=</td>
<td>Specifies the nominal sample size for fixed control limits</td>
</tr>
<tr>
<td>MEDCENTRAL=</td>
<td>Specifies the method of estimating the process mean ( \mu )</td>
</tr>
<tr>
<td>NO3SIGMACHECK</td>
<td>Applies tests for special causes when control limits other than 3( \sigma ) limits are in effect</td>
</tr>
<tr>
<td>NOPRINT</td>
<td>Suppresses ODS output</td>
</tr>
<tr>
<td>OUTLIMITS=</td>
<td>Creates an output data table that contains control limits</td>
</tr>
<tr>
<td>OUTTABLE=</td>
<td>Creates an output data table that contains subgroup summary statistics and control limits</td>
</tr>
<tr>
<td>PCTLDEF=</td>
<td>Specifies the percentile definition for box-and-whisker computation</td>
</tr>
<tr>
<td>SIGMAS=</td>
<td>Specifies the width of the control limits in terms of a multiple of the standard error of the subgroup summary statistic</td>
</tr>
<tr>
<td>SMETHOD=</td>
<td>Specifies the method of estimating the process standard deviation ( \sigma )</td>
</tr>
<tr>
<td>SPECS=</td>
<td>Specifies an input data table that contains process specification limits and computes process capability indices</td>
</tr>
<tr>
<td>TEST2RUN=</td>
<td>Specifies the length of the pattern for Test 2</td>
</tr>
<tr>
<td>TEST3RUN=</td>
<td>Specifies the length of the pattern for Test 3</td>
</tr>
<tr>
<td>TESTNSTD</td>
<td>Applies tests for special causes to standardized chart statistics</td>
</tr>
<tr>
<td>TESTOVERLAP</td>
<td>Performs tests for special causes on overlapping patterns of points</td>
</tr>
<tr>
<td>TESTS=</td>
<td>Requests tests for special causes for the box chart</td>
</tr>
</tbody>
</table>

CCHART Statement

CCHART < / options > ;

The CCHART statement produces \( c \) charts for the numbers of nonconformities (defects) in subgroup samples. Table 20.2 lists the options that you can specify in a CCHART statement. For detailed descriptions of these options, see the section “Dictionary of Options” on page 949.
**Table 20.2**  CCHART Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLN</td>
<td>Includes all subgroups in the analysis, regardless of sample size</td>
</tr>
<tr>
<td>EXCHART</td>
<td>Produces results only for charts in which exceptions occur</td>
</tr>
<tr>
<td>LIMITN=</td>
<td>Specifies the nominal sample size for fixed control limits</td>
</tr>
<tr>
<td>NO3SIGMACHECK</td>
<td>Applies tests for special causes when control limits other than $3\sigma$ limits are in effect</td>
</tr>
<tr>
<td>NOPRINT</td>
<td>Suppresses ODS output</td>
</tr>
<tr>
<td>OUTLIMITS=</td>
<td>Creates an output data table that contains control limits</td>
</tr>
<tr>
<td>OUTTABLE=</td>
<td>Creates an output data table that contains subgroup summary statistics and control limits</td>
</tr>
<tr>
<td>SIGMAS=</td>
<td>Specifies the width of the control limits in terms of a multiple of the standard error of the subgroup summary statistic</td>
</tr>
<tr>
<td>SUBGROUPN=</td>
<td>Specifies a variable in the input data table that contains subgroup sample sizes</td>
</tr>
<tr>
<td>TEST2RUN=</td>
<td>Specifies the length of the pattern for Test 2</td>
</tr>
<tr>
<td>TEST3RUN=</td>
<td>Specifies the length of the pattern for Test 3</td>
</tr>
<tr>
<td>TESTNSTD</td>
<td>Applies tests for special causes to standardized chart statistics</td>
</tr>
<tr>
<td>TESTOVERLAP</td>
<td>Performs tests for special causes on overlapping patterns of points</td>
</tr>
<tr>
<td>TESTS=</td>
<td>Requests tests for special causes for the $c$ chart</td>
</tr>
</tbody>
</table>

**IRCHART Statement**

`IRCHART < / options > ;`

The IRCHART statement produces control charts for individual measurements and moving ranges. Table 20.3 lists the options that you can specify in an IRCHART statement. For detailed descriptions of these options, see the section “Dictionary of Options” on page 949.

**Table 20.3**  IRCHART Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIINDICES</td>
<td>Computes capability index confidence limits</td>
</tr>
<tr>
<td>EXCHART</td>
<td>Produces results only for charts in which exceptions occur</td>
</tr>
<tr>
<td>NO3SIGMACHECK</td>
<td>Applies tests for special causes when control limits other than $3\sigma$ limits are in effect</td>
</tr>
<tr>
<td>NOPRINT</td>
<td>Suppresses ODS output</td>
</tr>
<tr>
<td>OUTLIMITS=</td>
<td>Creates an output data table that contains control limits</td>
</tr>
<tr>
<td>OUTTABLE=</td>
<td>Creates an output data table that contains subgroup summary statistics and control limits</td>
</tr>
<tr>
<td>SIGMAS=</td>
<td>Specifies the width of the control limits in terms of a multiple of the standard error of the subgroup summary statistic</td>
</tr>
<tr>
<td>SMETHOD=</td>
<td>Specifies the method of estimating the process standard deviation $\sigma$</td>
</tr>
</tbody>
</table>
Table 20.3  continued

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPECS=</td>
<td>Specifies an input data table that contains process specification limits and computes process capability indices</td>
</tr>
<tr>
<td>TEST2RUN=</td>
<td>Specifies the length of the pattern for Test 2</td>
</tr>
<tr>
<td>TEST3RUN=</td>
<td>Specifies the length of the pattern for Test 3</td>
</tr>
<tr>
<td>TESTNSTD</td>
<td>Applies tests for special causes to standardized chart statistics</td>
</tr>
<tr>
<td>TESTOVERLAP</td>
<td>Performs tests for special causes on overlapping patterns of points</td>
</tr>
<tr>
<td>TESTS=</td>
<td>Requests tests for special causes for the individual measurements chart</td>
</tr>
</tbody>
</table>

MCHART Statement

**MCHART < / options > ;**

The MCHART statement produces control charts for subgroup medians. Table 20.4 lists the options that you can specify in an MCHART statement. For detailed descriptions of these options, see the section “Dictionary of Options” on page 949.

Table 20.4  MCHART Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLN</td>
<td>Includes all subgroups in the analysis, regardless of sample size</td>
</tr>
<tr>
<td>CIINDICES</td>
<td>Computes capability index confidence limits</td>
</tr>
<tr>
<td>EXCHART</td>
<td>Produces results only for charts in which exceptions occur</td>
</tr>
<tr>
<td>LIMITN=</td>
<td>Specifies the nominal sample size for fixed control limits</td>
</tr>
<tr>
<td>MEDCENTRAL=</td>
<td>Specifies the method of estimating the process mean $\mu$</td>
</tr>
<tr>
<td>NO3SIGMACHECK</td>
<td>Applies tests for special causes when control limits other than $3\sigma$ limits are in effect</td>
</tr>
<tr>
<td>NOPRINT</td>
<td>Suppresses ODS output</td>
</tr>
<tr>
<td>OUTLIMITS=</td>
<td>Creates an output data table that contains control limits</td>
</tr>
<tr>
<td>OUTTABLE=</td>
<td>Creates an output data table that contains subgroup summary statistics and control limits</td>
</tr>
<tr>
<td>SIGMAS=</td>
<td>Specifies the width of the control limits in terms of a multiple of the standard error of the subgroup summary statistic</td>
</tr>
<tr>
<td>SMETHOD=</td>
<td>Specifies the method of estimating the process standard deviation $\sigma$</td>
</tr>
<tr>
<td>SPECS=</td>
<td>Specifies an input data table that contains process specification limits and computes process capability indices</td>
</tr>
<tr>
<td>TEST2RUN=</td>
<td>Specifies the length of the pattern for Test 2</td>
</tr>
<tr>
<td>TEST3RUN=</td>
<td>Specifies the length of the pattern for Test 3</td>
</tr>
<tr>
<td>TESTNSTD</td>
<td>Applies tests for special causes to standardized chart statistics</td>
</tr>
<tr>
<td>TESTOVERLAP</td>
<td>Performs tests for special causes on overlapping patterns of points</td>
</tr>
<tr>
<td>TESTS=</td>
<td>Requests tests for special causes for the median chart</td>
</tr>
</tbody>
</table>
The MRCHART statement produces control charts for subgroup medians and ranges. Table 20.5 lists the options that you can specify in an MRCHART statement. For detailed descriptions of these options, see the section “Dictionary of Options” on page 949.

Table 20.5  MRCHART Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLN</td>
<td>Includes all subgroups in the analysis, regardless of sample size</td>
</tr>
<tr>
<td>CIINDICES</td>
<td>Computes capability index confidence limits</td>
</tr>
<tr>
<td>EXCHART</td>
<td>Produces results only for charts in which exceptions occur</td>
</tr>
<tr>
<td>LIMITN=</td>
<td>Specifies the nominal sample size for fixed control limits</td>
</tr>
<tr>
<td>MEDCENTRAL=</td>
<td>Specifies the method of estimating the process mean $\mu$</td>
</tr>
<tr>
<td>NO3SIGMACHECK</td>
<td>Applies tests for special causes when control limits other than $3\sigma$ limits are in effect</td>
</tr>
<tr>
<td>NOPRINT</td>
<td>Suppresses ODS output</td>
</tr>
<tr>
<td>OUTLIMITS=</td>
<td>Creates an output data table that contains control limits</td>
</tr>
<tr>
<td>OUTTABLE=</td>
<td>Creates an output data table that contains subgroup summary statistics and control limits</td>
</tr>
<tr>
<td>SIGMAS=</td>
<td>Specifies the width of the control limits in terms of a multiple of the standard error of the subgroup summary statistic</td>
</tr>
<tr>
<td>SMETHOD=</td>
<td>Specifies the method of estimating the process standard deviation $\sigma$</td>
</tr>
<tr>
<td>SPECS=</td>
<td>Specifies an input data table that contains process specification limits and computes process capability indices</td>
</tr>
<tr>
<td>TEST2RUN=</td>
<td>Specifies the length of the pattern for Test 2</td>
</tr>
<tr>
<td>TEST3RUN=</td>
<td>Specifies the length of the pattern for Test 3</td>
</tr>
<tr>
<td>TESTNSTD</td>
<td>Applies tests for special causes to standardized chart statistics</td>
</tr>
<tr>
<td>TESTOVERLAP</td>
<td>Performs tests for special causes on overlapping patterns of points</td>
</tr>
<tr>
<td>TESTS=</td>
<td>Requests tests for special causes for the median chart</td>
</tr>
<tr>
<td>TESTS2=</td>
<td>Requests tests for special causes for the $R$ chart</td>
</tr>
</tbody>
</table>

The NPCHART statement produces control charts for the numbers of nonconforming items in subgroup samples. Table 20.6 lists the options that you can specify in an NPCHART statement. For detailed descriptions of these options, see the section “Dictionary of Options” on page 949.
Table 20.6  NPCHART Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLN</td>
<td>Includes all subgroups in the analysis, regardless of sample size</td>
</tr>
<tr>
<td>EXCHART</td>
<td>Produces results only for charts in which exceptions occur</td>
</tr>
<tr>
<td>LIMITN=</td>
<td>Specifies the nominal sample size for fixed control limits</td>
</tr>
<tr>
<td>NO3SIGMACHECK</td>
<td>Applies tests for special causes when control limits other than 3σ limits are in effect</td>
</tr>
<tr>
<td>NOPRINT</td>
<td>Suppresses ODS output</td>
</tr>
<tr>
<td>OUTLIMITS=</td>
<td>Creates an output data table that contains control limits</td>
</tr>
<tr>
<td>OUTTABLE=</td>
<td>Creates an output data table that contains subgroup summary statistics and control limits</td>
</tr>
<tr>
<td>SIGMAS=</td>
<td>Specifies the width of the control limits in terms of a multiple of the standard error of the subgroup summary statistic</td>
</tr>
<tr>
<td>SUBGROUPN=</td>
<td>Specifies a variable in the input data table that contains subgroup sample sizes</td>
</tr>
<tr>
<td>TEST2RUN=</td>
<td>Specifies the length of the pattern for Test 2</td>
</tr>
<tr>
<td>TEST3RUN=</td>
<td>Specifies the length of the pattern for Test 3</td>
</tr>
<tr>
<td>TESTNSTD</td>
<td>Applies tests for special causes to standardized chart statistics</td>
</tr>
<tr>
<td>TESTOVERLAP</td>
<td>Performs tests for special causes on overlapping patterns of points</td>
</tr>
<tr>
<td>TESTS=</td>
<td>Requests tests for special causes for the np chart</td>
</tr>
</tbody>
</table>

PCHART Statement

\texttt{PCHART < /options> ;}

The PCHART statement produces control charts for the proportions of nonconforming items in subgroup samples. Table 20.7 lists the options that you can specify in a PCHART statement. For detailed descriptions of these options, see the section “Dictionary of Options” on page 949.

Table 20.7  PCHART Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLN</td>
<td>Includes all subgroups in the analysis, regardless of sample size</td>
</tr>
<tr>
<td>EXCHART</td>
<td>Produces results only for charts in which exceptions occur</td>
</tr>
<tr>
<td>LIMITN=</td>
<td>Specifies the nominal sample size for fixed control limits</td>
</tr>
<tr>
<td>NO3SIGMACHECK</td>
<td>Applies tests for special causes when control limits other than 3σ limits are in effect</td>
</tr>
<tr>
<td>NOPRINT</td>
<td>Suppresses ODS output</td>
</tr>
<tr>
<td>OUTLIMITS=</td>
<td>Creates an output data table that contains control limits</td>
</tr>
<tr>
<td>OUTTABLE=</td>
<td>Creates an output data table that contains subgroup summary statistics and control limits</td>
</tr>
<tr>
<td>SIGMAS=</td>
<td>Specifies the width of the control limits in terms of a multiple of the standard error of the subgroup summary statistic</td>
</tr>
</tbody>
</table>
The RCHART statement produces control charts for subgroup ranges. Table 20.8 lists the options that you can specify in an RCHART statement. For detailed descriptions of these options, see the section “Dictionary of Options” on page 949.

### Table 20.8  RCHART Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLN</td>
<td>Includes all subgroups in the analysis, regardless of sample size</td>
</tr>
<tr>
<td>CIINDICES</td>
<td>Computes capability index confidence limits</td>
</tr>
<tr>
<td>EXCHART</td>
<td>Produces results only for charts in which exceptions occur</td>
</tr>
<tr>
<td>LIMITN=</td>
<td>Specifies the nominal sample size for fixed control limits</td>
</tr>
<tr>
<td>NO3SIGMACHECK</td>
<td>Applies tests for special causes when control limits other than 3σ limits are in effect</td>
</tr>
<tr>
<td>NOPRINT</td>
<td>Suppresses ODS output</td>
</tr>
<tr>
<td>OUTLIMITS=</td>
<td>Creates an output data table that contains control limits</td>
</tr>
<tr>
<td>OUTTABLE=</td>
<td>Creates an output data table that contains subgroup summary statistics and control limits</td>
</tr>
<tr>
<td>SIGMAS=</td>
<td>Specifies the width of the control limits in terms of a multiple of the standard error of the subgroup summary statistic</td>
</tr>
<tr>
<td>SMETHOD=</td>
<td>Specifies the method of estimating the process standard deviation σ</td>
</tr>
<tr>
<td>SPECS=</td>
<td>Specifies an input data table that contains process specification limits and computes process capability indices</td>
</tr>
<tr>
<td>TEST2RUN=</td>
<td>Specifies the length of the pattern for Test 2</td>
</tr>
<tr>
<td>TEST3RUN=</td>
<td>Specifies the length of the pattern for Test 3</td>
</tr>
<tr>
<td>TESTNSTD</td>
<td>Applies tests for special causes to standardized chart statistics</td>
</tr>
<tr>
<td>TESTOVERLAP</td>
<td>Performs tests for special causes on overlapping patterns of points</td>
</tr>
<tr>
<td>TESTS2=</td>
<td>Requests tests for special causes for the R chart</td>
</tr>
</tbody>
</table>
**SCHART Statement**

```
SCHART < / options> ;
```

The SCHART statement produces control charts for subgroup standard deviations. Table 20.9 lists the options that you can specify in an SCHART statement. For detailed descriptions of these options, see the section “Dictionary of Options” on page 949.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLN</td>
<td>Includes all subgroups in the analysis, regardless of sample size</td>
</tr>
<tr>
<td>CIINDICES</td>
<td>Computes capability index confidence limits</td>
</tr>
<tr>
<td>EXCHART</td>
<td>Produces results only for charts in which exceptions occur</td>
</tr>
<tr>
<td>LIMITN=</td>
<td>Specifies the nominal sample size for fixed control limits</td>
</tr>
<tr>
<td>NO3SIGMACHECK</td>
<td>Applies tests for special causes when control limits other than $3\sigma$ limits are in effect</td>
</tr>
<tr>
<td>NOPRINT</td>
<td>Suppresses ODS output</td>
</tr>
<tr>
<td>OUTLIMITS=</td>
<td>Creates an output data table that contains control limits</td>
</tr>
<tr>
<td>OUTTABLE=</td>
<td>Creates an output data table that contains subgroup summary statistics and control limits</td>
</tr>
<tr>
<td>SIGMAS=</td>
<td>Specifies the width of the control limits in terms of a multiple of the standard error of the subgroup summary statistic</td>
</tr>
<tr>
<td>SMETHOD=</td>
<td>Specifies the method of estimating the process standard deviation $\sigma$ and computes process capability indices</td>
</tr>
<tr>
<td>SPECS=</td>
<td>Specifies an input data table that contains process specification limits</td>
</tr>
<tr>
<td>TEST2RUN=</td>
<td>Specifies the length of the pattern for Test 2</td>
</tr>
<tr>
<td>TEST3RUN=</td>
<td>Specifies the length of the pattern for Test 3</td>
</tr>
<tr>
<td>TESTNSTD</td>
<td>Applies tests for special causes to standardized chart statistics</td>
</tr>
<tr>
<td>TESTOVERLAP</td>
<td>Performs tests for special causes on overlapping patterns of points</td>
</tr>
<tr>
<td>TESTS2=</td>
<td>Requests tests for special causes for the $s$ chart</td>
</tr>
</tbody>
</table>

**UCHART Statement**

```
UCHART < / options> ;
```

The UCHART statement produces control charts for the numbers of nonconformities per inspection unit in subgroup samples. Table 20.10 lists the options that you can specify in a UCHART statement. For detailed descriptions of these options, see the section “Dictionary of Options” on page 949.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLN</td>
<td>Includes all subgroups in the analysis, regardless of sample size</td>
</tr>
<tr>
<td>EXCHART</td>
<td>Produces results only for charts in which exceptions occur</td>
</tr>
</tbody>
</table>
**Table 20.10  continued**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIMITN=</td>
<td>Specifies the nominal sample size for fixed control limits</td>
</tr>
<tr>
<td>NO3SIGMACHECK</td>
<td>Applies tests for special causes when control limits other than 3σ limits</td>
</tr>
<tr>
<td></td>
<td>are in effect</td>
</tr>
<tr>
<td>NOPRINT</td>
<td>Suppresses ODS output</td>
</tr>
<tr>
<td>OUTLIMITS=</td>
<td>Creates an output data table that contains control limits</td>
</tr>
<tr>
<td>OUTTABLE=</td>
<td>Creates an output data table that contains subgroup summary statistics</td>
</tr>
<tr>
<td></td>
<td>and control limits</td>
</tr>
<tr>
<td>SIGMAS=</td>
<td>Specifies the width of the control limits in terms of a multiple of the</td>
</tr>
<tr>
<td></td>
<td>standard error of the subgroup summary statistic</td>
</tr>
<tr>
<td>SUBGROUPN=</td>
<td>Specifies a variable in the input data table that contains subgroup sample</td>
</tr>
<tr>
<td></td>
<td>sizes</td>
</tr>
<tr>
<td>TEST2RUN=</td>
<td>Specifies the length of the pattern for Test 2</td>
</tr>
<tr>
<td>TEST3RUN=</td>
<td>Specifies the length of the pattern for Test 3</td>
</tr>
<tr>
<td>TESTNSTD</td>
<td>Applies tests for special causes to standardized chart statistics</td>
</tr>
<tr>
<td>TESTOVERLAP</td>
<td>Performs tests for special causes on overlapping patterns of points</td>
</tr>
<tr>
<td>TESTS=</td>
<td>Requests tests for special causes for the u chart</td>
</tr>
</tbody>
</table>

**XCHART Statement**

**XCHART< /options>;**

The XCHART statement produces control charts for subgroup means. Table 20.11 lists the options that you can specify in an XCHART statement. For detailed descriptions of these options, see the section “Dictionary of Options” on page 949.

**Table 20.11  XCHART Statement Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLN</td>
<td>Includes all subgroups in the analysis, regardless of sample size</td>
</tr>
<tr>
<td>CIINDICES</td>
<td>Computes capability index confidence limits</td>
</tr>
<tr>
<td>EXCHART</td>
<td>Produces results only for charts in which exceptions occur</td>
</tr>
<tr>
<td>LIMITN=</td>
<td>Specifies the nominal sample size for fixed control limits</td>
</tr>
<tr>
<td>NO3SIGMACHECK</td>
<td>Applies tests for special causes when control limits other than 3σ limits</td>
</tr>
<tr>
<td></td>
<td>are in effect</td>
</tr>
<tr>
<td>NOPRINT</td>
<td>Suppresses ODS output</td>
</tr>
<tr>
<td>OUTLIMITS=</td>
<td>Creates an output data table that contains control limits</td>
</tr>
<tr>
<td>OUTTABLE=</td>
<td>Creates an output data table that contains subgroup summary statistics</td>
</tr>
<tr>
<td></td>
<td>and control limits</td>
</tr>
<tr>
<td>SIGMAS=</td>
<td>Specifies the width of the control limits in terms of a multiple of the</td>
</tr>
<tr>
<td></td>
<td>standard error of the subgroup summary statistic</td>
</tr>
<tr>
<td>SMETHOD=</td>
<td>Specifies the method of estimating the process standard deviation σ</td>
</tr>
</tbody>
</table>
### XRCHART Statement

**XRCHART < / options > ;**

The XRCHART statement produces control charts for subgroup means and ranges. Table 20.12 lists the options that you can specify in an XRCHART statement. For detailed descriptions of these options, see the section “Dictionary of Options” on page 949.

#### Table 20.12  XRCHART Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLN</td>
<td>Includes all subgroups in the analysis, regardless of sample size</td>
</tr>
<tr>
<td>CIINDICES</td>
<td>Computes capability index confidence limits</td>
</tr>
<tr>
<td>EXCHART</td>
<td>Produces results only for charts in which exceptions occur</td>
</tr>
<tr>
<td>LIMITN=</td>
<td>Specifies the nominal sample size for fixed control limits</td>
</tr>
<tr>
<td>NO3SIGMACHECK</td>
<td>Applies tests for special causes when control limits other than 3σ limits are in effect</td>
</tr>
<tr>
<td>NOPRINT</td>
<td>Suppresses ODS output</td>
</tr>
<tr>
<td>OUTLIMITS=</td>
<td>Creates an output data table that contains control limits</td>
</tr>
<tr>
<td>OUTTABLE=</td>
<td>Creates an output data table that contains subgroup summary statistics and control limits</td>
</tr>
<tr>
<td>SIGMAS=</td>
<td>Specifies the width of the control limits in terms of a multiple of the standard error of the subgroup summary statistic</td>
</tr>
<tr>
<td>SMETHOD=</td>
<td>Specifies the method of estimating the process standard deviation σ</td>
</tr>
<tr>
<td>SPECS=</td>
<td>Specifies an input data table that contains process specification limits and computes process capability indices</td>
</tr>
<tr>
<td>TEST2RUN=</td>
<td>Specifies the length of the pattern for Test 2</td>
</tr>
<tr>
<td>TEST3RUN=</td>
<td>Specifies the length of the pattern for Test 3</td>
</tr>
<tr>
<td>TESTNSTD</td>
<td>Applies tests for special causes to standardized chart statistics</td>
</tr>
<tr>
<td>TESTOVERLAP</td>
<td>Performs tests for special causes on overlapping patterns of points</td>
</tr>
<tr>
<td>TESTS=</td>
<td>Requests tests for special causes for the $\bar{X}$ chart</td>
</tr>
<tr>
<td>TESTS2=</td>
<td>Requests tests for special causes for the $R$ chart</td>
</tr>
</tbody>
</table>
The XSCHART statement produces control charts for subgroup means and standard deviations. Table 20.13 lists the options that you can specify in an XSCHART statement. For detailed descriptions of these options, see the section “Dictionary of Options” on page 949.

**Table 20.13** XSCHART Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLN</td>
<td>Includes all subgroups in the analysis, regardless of sample size</td>
</tr>
<tr>
<td>CIINDICES</td>
<td>Computes capability index confidence limits</td>
</tr>
<tr>
<td>EXCHART</td>
<td>Produces results only for charts in which exceptions occur</td>
</tr>
<tr>
<td>LIMITN=</td>
<td>Specifies the nominal sample size for fixed control limits</td>
</tr>
<tr>
<td>NO3SIGMACHECK</td>
<td>Applies tests for special causes when control limits other than 3σ limits are in effect</td>
</tr>
<tr>
<td>NOPRINT</td>
<td>Suppresses ODS output</td>
</tr>
<tr>
<td>OUTLIMITS=</td>
<td>Creates an output data table that contains control limits</td>
</tr>
<tr>
<td>OUTTABLE=</td>
<td>Creates an output data table that contains subgroup summary statistics and control limits</td>
</tr>
<tr>
<td>SIGMAS=</td>
<td>Specifies the width of the control limits in terms of a multiple of the standard error of the subgroup summary statistic</td>
</tr>
<tr>
<td>SMETHOD=</td>
<td>Specifies the method of estimating the process standard deviation σ</td>
</tr>
<tr>
<td>SPECS=</td>
<td>Specifies an input data table that contains process specification limits and computes process capability indices</td>
</tr>
<tr>
<td>TEST2RUN=</td>
<td>Specifies the length of the pattern for Test 2</td>
</tr>
<tr>
<td>TEST3RUN=</td>
<td>Specifies the length of the pattern for Test 3</td>
</tr>
<tr>
<td>TESTNSTD</td>
<td>Applies tests for special causes to standardized chart statistics</td>
</tr>
<tr>
<td>TESTOVERLAP</td>
<td>Performs tests for special causes on overlapping patterns of points</td>
</tr>
<tr>
<td>TESTS=</td>
<td>Requests tests for special causes for the $\bar{x}$ chart</td>
</tr>
<tr>
<td>TESTS2=</td>
<td>Requests tests for special causes for the s chart</td>
</tr>
</tbody>
</table>

**Dictionary of Options**

The following entries provide detailed descriptions of options that are available in the chart statements: BOXCHART, CCHART, IRCHART, MCHART, MRCHART, NPCHART, PCHART, RCHART, SCHART, UCHART, XCHART, XRCHART, and XSCHART.

**ALLN**

includes all subgroups in the analysis, regardless of whether the subgroup sample size equals the nominal control limit sample size $n$ that you specify in the `LIMITN=` option or the variable `LIMITN_` in the `LIMITS=` data table. Use the ALLN option in conjunction with the `LIMITN=` option or the variable `LIMITN_`. 
requests capability index confidence limits that are based on subgroup summary data and are calculated using “effective degrees of freedom,” as described by Bissell (1990). These confidence limits are approximate. When you specify the CIINDICES option, the calculated confidence limits are displayed in the “CapabilityIndices” table and are included in the OUTLIMITS= data table, if one is produced.

**TYPE=keyword**

specifies the type of confidence limit. Valid *keywords* are LOWER, UPPER, and TWOSIDED. By default, TYPE=TWOSIDED.

**ALPHA=value**

specifies the default confidence level to compute confidence limits. The percentage for the confidence limits is \((1 - \text{value}) \times 100\). For example, ALPHA=.05 results in a 95% confidence limit. The default value is .05, and the possible range of values is from 0 to 1.

The CIINDICES option is ignored unless you also specify the SPECS= option. These options are available only in the chart statements that produce charts for variables: BOXCHART, IRCHART, MCHART, MRCHART, RCHART, SCHART, XCHART, XRCHART, and XSCHART. For more information about capability indices, see the section “Capability Indices” on page 979.

**CONTROLSTAT=MEAN | MEDIAN**

specifies whether the control limits that are displayed in a box chart are computed for subgroup means or for subgroup medians. By default, CONTROLSTAT=MEAN. This option is available only in the BOXCHART statement.

**EXCHART**

includes a control chart in the results only when exceptions occur—specifically, when the control limits are exceeded or when any tests that you request by using the TESTS= or TESTS2= option are positive.

**LIMITN=n**

specifies a nominal sample size for the control limits.

If you specify LIMITN=n, the control limits are computed for the fixed value \(n\), and they do not vary with the subgroup sample sizes. Moreover, only those subgroups whose sample size is equal to \(n\) are included in the analysis. You can specify ALLN in conjunction with LIMITN=n to force all subgroups to be included, regardless of subgroup sample size.

If you do not specify LIMITN=n and the subgroup sample sizes are constant, the default value of \(n\) is the constant subgroup sample size.

Depending on which chart statement you use, there are restrictions on the value of \(n\) that you can specify in the LIMITN= option. For the MRCHART, RCHART, and XRCHART statements, \(2 \leq n \leq 25\). For the SCHART and XSCHART statements, \(n \geq 2\). For the BOXCHART, MCHART, and XCHART statements, \(n \geq 1\). If the estimate of the process standard deviation is based on subgroup ranges (see the SMETHOD= option and the section “Methods of Estimating the Standard Deviation” on page 956), \(n < 26\). For the CCHART and UCHART statements, \(n > 0\), and \(n\) can assume fractional values (for all other chart statements, \(n\) must be a whole number). For the PCHART and NPCHART statements, \(n \geq 1\).

Note the difference between the LIMITN= option and the SUBGROUPN= option that is available in the CCHART, NPCHART, PCHART, and UCHART statements. The LIMITN= option specifies a
nominal sample size for the control limits, whereas the SUBGROUPN= option provides the sample sizes for the data.

**NOTE:** As an alternative to specifying the LIMITN= option, you can read the nominal control limit sample size from the variable _LIMITN_ in a LIMITS= data table.

**MEDCENTRAL=AVGMEAN | AVGMED | MEDMED**

identifies a method of estimating the process mean $\mu$, which is represented by the central line on a median chart. The method that corresponds to each keyword is shown in Table 20.14.

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVGMEAN</td>
<td>Average of subgroup means</td>
</tr>
<tr>
<td>AVGMED</td>
<td>Average of subgroup medians</td>
</tr>
<tr>
<td>MEDMED</td>
<td>Median of subgroup medians</td>
</tr>
</tbody>
</table>

By default, MEDCENTRAL=AVGMED. This option is available only in the MCHART and MRCHART statements, and in the BOXCHART statement when you specify the CONTROL-STAT=MEDIAN option.

**NO3SIGMACHECK**

suppresses the check for $3\sigma$ limits when you request tests for special causes. This enables tests for special causes to be applied when you use SIGMAS= option to specify control limits other than the default $3\sigma$ limits. You should not use the NO3SIGMACHECK option for standard control chart applications, because the standard tests for special causes assume $3\sigma$ limits.

**NOPRINT**

suppresses the display of ODS output.

**OUTLIMITS=**

creates an output data table that saves the control limits. You can use an OUTLIMITS= data table as an input LIMITS= data table in a subsequent run of the procedure. CAS-libref.data-table is a two-level name, where CAS-libref refers to the caslib and session identifier, and data-table specifies the name of the output data table. For more information about this two-level name, see the DATA= option and the section “Using CAS Sessions and CAS Engine Librefs” on page 934.

For more information about OUTLIMITS= data tables, see the section “OUTLIMITS= Data Table” on page 982.

**OUTTABLE=**

creates an output SAS data table that saves the information plotted on the chart, including the subgroup variable values and their corresponding summary statistics and control limits. CAS-libref.data-table is a two-level name, where CAS-libref refers to the caslib and session identifier, and data-table specifies the name of the output data table. For more information about this two-level name, see the DATA= option and the section “Using CAS Sessions and CAS Engine Librefs” on page 934.

For more information about OUTTABLE= data tables, see the section “OUTTABLE= Data Table” on page 987.
**PCTLDEF=**\textit{index}

specifies one of five definitions that are used to calculate percentiles in the construction of box-and-whisker plots that you request in the BOXCHART statement. The \textit{index} can be 1, 2, 3, 4, or 5. The five corresponding percentile definitions are discussed in the section “Percentile Definitions” on page 972. By default, PCTLDEF=5. This option is available only in the BOXCHART statement.

**SIGMAS=**\textit{k}

specifies the width of the control limits in terms of the multiple \textit{k} of the standard error of the subgroup summary statistic. The value of \textit{k} must be positive. By default, \textit{k} = 3, and the control limits are 3\textit{\sigma} limits.

**SMETHOD=**MAD | MMR< (\textit{n}) > | MVGRANGE< (\textit{n}) > | RMSDF | RMVLUE | RNOWEIGHT | SMVLUE | SNOWEIGHT

specifies the method of estimating the process standard deviation, \(\sigma\), as summarized in Table 20.15.

**Table 20.15** Methods of Estimating the Process Standard Deviation

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAD</td>
<td>Calculates a median absolute deviation estimate of (\sigma)</td>
</tr>
<tr>
<td>MMR</td>
<td>Calculates a median moving range estimate of (\sigma)</td>
</tr>
<tr>
<td>MVGRANGE</td>
<td>Estimates (\sigma) based on a moving range of \textit{n} subgroup means</td>
</tr>
<tr>
<td>RMSDF</td>
<td>Calculates a root mean square estimate of (\sigma)</td>
</tr>
<tr>
<td>RMVLUE</td>
<td>Calculates a minimum variance linear unbiased estimate of (\sigma) based on subgroup ranges</td>
</tr>
<tr>
<td>RNOWEIGHT</td>
<td>Estimates (\sigma) as an unweighted average of unbiased subgroup estimates of (\sigma) based on subgroup ranges</td>
</tr>
<tr>
<td>SMVLUE</td>
<td>Calculates a minimum variance linear unbiased estimate of (\sigma) based on subgroup standard deviations</td>
</tr>
<tr>
<td>SNOWEIGHT</td>
<td>Estimates (\sigma) as an unweighted average of unbiased subgroup estimates of (\sigma) based on subgroup standard deviations</td>
</tr>
</tbody>
</table>

Table 20.16 lists the SMETHOD= option values that are valid for each chart statement. The default value for each statement is shown in bold.

**Table 20.16** Valid SMETHOD= Values by Chart Statement

<table>
<thead>
<tr>
<th>Chart Statement</th>
<th>Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOXCHART</td>
<td>RMSDF, RMVLUE, RNOWEIGHT, SMVLUE, <strong>SNOWEIGHT</strong></td>
</tr>
<tr>
<td>ICHART</td>
<td>MAD, MMR, MVGRANGE</td>
</tr>
<tr>
<td>MCHART</td>
<td>RMSDF, RMVLUE, RNOWEIGHT, SMVLUE, SNOWEIGHT</td>
</tr>
<tr>
<td>MRCHART</td>
<td>RMVLUE, <strong>RNOWEIGHT</strong></td>
</tr>
<tr>
<td>RCHART</td>
<td>RMVLUE, <strong>RNOWEIGHT</strong></td>
</tr>
<tr>
<td>SCHART</td>
<td>RMSDF, SMVLUE, <strong>SNOWEIGHT</strong></td>
</tr>
<tr>
<td>XCHART</td>
<td>RMSDF, RMVLUE, <strong>RNOWEIGHT</strong>, SMVLUE, SNOWEIGHT</td>
</tr>
<tr>
<td>XRCHART</td>
<td>MVGRANGE, RMVLUE, <strong>RNOWEIGHT</strong></td>
</tr>
<tr>
<td>XSCCHART</td>
<td>MVGRANGE, RMSDF, SMVLUE, <strong>SNOWEIGHT</strong></td>
</tr>
</tbody>
</table>
When you use the MMR or MVGRANGE method, you can specify the number \( n \) of consecutive subgroup summary statistics from which moving ranges are computed by appending the number in parentheses to the keyword. For example, you can specify SMETHOD=MVGRANGE(4) to use moving ranges of four subgroup statistics. Values in the range \( 2 \leq n \leq 25 \) are valid. By default, \( n = 2 \).

For more information about the different methods of estimating the process standard deviation, see the section “Methods of Estimating the Standard Deviation” on page 956.

**SPECS=** `CAS-libref.data-table`

specifies an input data table that contains process specification limits. `CAS-libref.data-table` is a two-level name, where `CAS-libref` refers to the caslib and session identifier, and `data-table` specifies the name of the input data table. For more information about this two-level name, see the DATA= option and the section “Using CAS Sessions and CAS Engine Librefs” on page 934. PROC SPC calculates capability indices for processes whose specifications are given in `data-table`.

This option is available only in the chart statements that produce charts for variables: BOXCHART, IRCHART, MCHART, MRCHART, RCHART, SCHRAT, XCHART, XRCHART, and XSCHART. For more information, see the sections “SPECS= Data Table” on page 982 and “Capability Indices” on page 979.

**SUBGROUPN=** `variable`

specifies a variable in the input data table whose values specify the subgroup sample sizes. This option is available only in the CCHART, NPCHART, PCHART, and UCHART statements. You must specify this option in the NPCHART, PCHART, and UCHART statements. For the CCHART statement, the default subgroup sample size is 1.

**TEST2RUN=** `run-length`

specifies the length of the pattern for Test 2 that you request in the TESTS= and TESTS2= options. The `run-length` values that you can specify are 7, 8, 9, 11, 14, and 20. The form of the test for each `run-length` value is shown in the following table. By default, TEST2RUN=9. For more information, see the section “Tests for Special Causes” on page 973.

<table>
<thead>
<tr>
<th><code>run-length</code></th>
<th>Number of Points on One Side of the Central Line</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>7 in a row</td>
</tr>
<tr>
<td>8</td>
<td>8 in a row</td>
</tr>
<tr>
<td>9</td>
<td>9 in a row</td>
</tr>
<tr>
<td>11</td>
<td>At least 10 out of 11 in a row</td>
</tr>
<tr>
<td>14</td>
<td>At least 12 out of 14 in a row</td>
</tr>
<tr>
<td>20</td>
<td>At least 16 out of 20 in a row</td>
</tr>
</tbody>
</table>

**TEST3RUN=** `run-length`

specifies the length of the pattern for Test 3 that you request in the TESTS= and TESTS2= options. Test 3 searches for a pattern of steadily increasing or decreasing values, where the length of the pattern is at least the value of `run-length`. The values that you can specify for `run-length` are 6, 7, and 8. By default, TEST3RUN=6. For more information, see the section “Tests for Special Causes” on page 973.
TESTNSTD
applies the tests for special causes that you request in the TESTS= and TESTS2= options to standardized test statistics when the subgroup sample sizes are not constant. This method was suggested by Nelson (1994). For more information, see the section “Tests for Special Causes” on page 973. By default, the tests are not applied to data that have varying subgroup sample sizes.

TESTOVERLAP
applies tests for special causes (which you request in the TESTS= or TESTS2= option) to overlapping patterns of points.

This option modifies the way in which the search for a subsequent pattern is done when a pattern is encountered. If you omit this option, the search begins with the first subgroup after the current pattern ends. If you specify this option, the search begins with the second subgroup in the current pattern.

The following statements request Test 3, which detects six subgroup means in a row that steadily increase or decrease:

```
proc spc;
   xchart / test=3;
   xchart / test=3 testoverlap;
run;
```

Suppose that the subgroup means are steadily increasing for subgroups 5, 6, 7, 8, 9, 10, and 11. The first XRCHART statement signals that Test 3 is positive at subgroup 10 but not at subgroup 11, because the search for the next pattern begins with subgroup 11. The second XRCHART statement signals that Test 3 is positive at subgroup 10 and subgroup 11, because the search for the next pattern begins with subgroup 6 and thus finds a second pattern that ends with subgroup 11. For more information, see the section “Tests for Special Causes” on page 973.

**CAUTION:** Specifying TESTOVERLAP affects the interpretation of the standard tests for special causes, because a particular point can contribute to more than one positive test. Typically, you should not use this option.

TESTS=INDEX-LIST
requests one or more tests for special causes, which are also known as runs tests, pattern tests, and Western Electric rules. These tests detect particular nonrandom patterns in the points that are plotted on the primary control chart. The occurrence of a pattern, referred to as a signal, suggests the presence of a special cause of variation.

Each pattern is defined in terms of Zones A, B, and C, which are constructed by dividing the interval between the control limits into six equally spaced subintervals. Zone A is the union of the subintervals immediately below the upper control limit and immediately above the lower control limit. Zone C is the union of the subintervals immediately above and below the central line. Zone B is the union of the subintervals between Zones A and C. For an illustration of test zones, see Figure 20.9.

Table 20.17 lists the standard tests that you can request by specifying this option. The tests are indexed according to the sequence used by Nelson (1984, 1985).
Table 20.17 Standard Tests for Special Causes

<table>
<thead>
<tr>
<th>Index</th>
<th>Pattern Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>One point beyond Zone A (outside the control limits)</td>
</tr>
<tr>
<td>2</td>
<td>Nine points in a row in Zone C or beyond on one side of the central line (see the TEST2RUN= option)</td>
</tr>
<tr>
<td>3</td>
<td>Six points in a row that steadily increase or decrease (see the TEST3RUN= option)</td>
</tr>
<tr>
<td>4</td>
<td>Fourteen points in a row that alternate up and down</td>
</tr>
<tr>
<td>5</td>
<td>Two out of three points in a row in Zone A or beyond</td>
</tr>
<tr>
<td>6</td>
<td>Four out of five points in a row in Zone B or beyond</td>
</tr>
<tr>
<td>7</td>
<td>Fifteen points in a row in Zone C on either or both sides of the central line</td>
</tr>
<tr>
<td>8</td>
<td>Eight points in a row on either or both sides of the central line, with no points in Zone C</td>
</tr>
</tbody>
</table>

You can specify any combination of the eight *indices* by using an explicit list or an implicit list, as in the following example:

```
proc spc;
   xrchart / tests=1 2 3 4;
   xrchart / tests=1 to 4;
run;
```

The TESTS= option is available in all chart statements except the RCHART and SCHART statements. Use only tests 1, 2, 3, and 4 in the CCHART, NPCHART, PCHART, and UCHART statements. By default, the TESTS= option is not applied in any chart statement unless the control limits are 3σ limits. You can use the NO3SIGMACHECK option to request tests for special causes when you use the SIGMAS= option to specify control limits other than 3σ limits.

For more information about the TESTS= option, see the section “Tests for Special Causes” on page 973. Related options include TEST2RUN=, TEST3RUN=, TESTNSTD, TESTOVERLAP, and TESTS2=.

**TESTS2=**index-list

requests one or more tests for special causes for *R* charts or *s* charts. The syntax for the TESTS2= option is identical to the syntax for the TESTS= option. The TESTS2= option is available in the MRCHART, RCHART, SCHART, XRCHART, and XSCHART statements. For more information about the TESTS2= option, see the section “Tests for Special Causes” on page 973. Related options include TEST2RUN=, TEST3RUN=, TESTNSTD, TESTOVERLAP, and TESTS=.
Details: SPC Procedure

Methods of Estimating the Standard Deviation

You can specify the SMETHOD= option to choose among several methods of estimating the process standard deviation for control charts for variables. The following sections describe the methods in detail.

MAD Method

If you specify SMETHOD=MAD, a median absolute deviation estimator is computed for $\sigma$. This method is described by Boyles (1997) and computed as

$$\hat{\sigma} = \text{median}\{|X_i - \bar{X}_i|, 1 \leq i \leq N\}/0.6745$$

where $\bar{X}_i$ is the sample median.

The MAD method is available only in the IRCHART statement.

MMR Method

If you specify SMETHOD=MMR, a median moving range estimator is computed for $\sigma$. This estimator is described by Boyles (1997) and computed as

$$\hat{\sigma} = \bar{R}/0.954$$

where $\bar{R}$ is the median of the nonmissing moving ranges.

The MMR method is available only in the IRCHART statement.

MVGRANGE Method

If you specify SMETHOD=MVGRANGE, $\sigma$ is estimated by using a moving range of subgroup averages (or individual measurements in an IRCHART statement). The estimate of $\sigma$ is

$$\hat{\sigma} = \bar{R}/d_2(n)$$

where $\bar{R}$ is the average of the moving ranges; $n$ is the number of consecutive subgroup averages that are used to compute each moving range; and the unbiasing factor $d_2(n)$ is defined so that if the subgroup averages are normally distributed, the expected value of $R_i$ is

$$E(R_i) = d_2(n_i)\sigma$$

This method is described by the American Society for Testing and Materials (1976).

The method is appropriate for constructing the three-way control chart that Wheeler (1995) advocated for this situation. A three-way control chart is useful when sampling (within-group) variation is not the only source of variation. A three-way control chart contains a chart of subgroup means, a moving range chart of the subgroup means, and a chart of subgroup ranges. When you specify the SMETHOD=MVGRANGE option, the XRCHART statement produces the appropriate charts of subgroup means and subgroup ranges, and the XSCHART statement produces the appropriate charts of subgroup means and subgroup standard deviations.

The MVGRANGE method is available in the IRCHART, XRCHART, and XSCHART statements. It is the default in the IRCHART statement.
RMSDF Method

If you specify SMETHOD=RMSDF, a weighted root mean square estimate is computed for \( \sigma \),

\[
\hat{\sigma} = \frac{\sqrt{(n_1 - 1)s_1^2 + \cdots + (n_N - 1)s_N^2}}{c_4(n) \sqrt{n_1 + \cdots + n_N - N}}
\]

where \( n = n_1 + \cdots + n_N - (N - 1) \). The weights are the degrees of freedom \( n_i - 1 \). A subgroup standard deviation \( s_i \) is included in the calculation only if \( n_i \geq 2 \), and \( N \) is the number of subgroups for which \( n_i \geq 2 \).

If the unknown standard deviation \( \sigma \) is constant across subgroups, the root mean square estimate is more efficient than the minimum variance linear unbiased estimate (MVLUE). However, in process control applications, it is generally not assumed that \( \sigma \) is constant, and if \( \sigma \) varies across subgroups, the root mean square estimate tends to be more inflated than the MVLUE.

The RMSDF method is available in the BOXCHART, MCHART, SCHART, XCHART, and XSCHART statements.

RMVLUE Method

If you specify SMETHOD=RMVLUE, a minimum variance linear unbiased estimate (MVLUE) is computed for \( \sigma \); see Burr (1969, 1976) and Nelson (1989, 1994). The MVLUE is a weighted average of \( N \) unbiased estimates of \( \sigma \) of the form \( R_i/d_2(n_i) \), and it is computed as

\[
\hat{\sigma} = \frac{f_1R_1/d_2(n_1) + \cdots + f_NR_N/d_2(n_N)}{f_1 + \cdots + f_N}
\]

where

\[
f_i = \frac{[d_2(n_i)]^2}{[d_3(n_i)]^2}
\]

A subgroup range \( R_i \) is included in the calculation only if \( n_i \geq 2 \), and \( N \) is the number of subgroups for which \( n_i \geq 2 \). The unbiasing factor \( d_3(n_i) \) is defined so that, if the observations are normally distributed, the expected value of \( \sigma R_i \) is \( d_3(n_i)\sigma \). The MVLUE assigns greater weight to estimates of \( \sigma \) from subgroups that have larger sample sizes, and it is intended for situations where the subgroup sample sizes vary. If the subgroup sample sizes are constant, the RMVLUE estimate reduces to the RNOWEIGHT estimate.

The RMVLUE method is available in the BOXCHART, MCHART, MRCHART, RCHART, XCHART, and XRCHART statements.

RNOWEIGHT Method

If you specify SMETHOD=RNOWEIGHT, the estimate of \( \sigma \) is

\[
\hat{\sigma} = \frac{R_1/d_2(n_1) + \cdots + R_N/d_2(n_N)}{N}
\]

where \( N \) is the number of subgroups for which \( n_i \geq 2 \), and \( R_i \) is the sample range of the observations \( x_{i1}, \ldots, x_{in_i} \) in the \( i \)th subgroup:

\[
R_i = \max_{1 \leq j \leq n_i} (x_{ij}) - \min_{1 \leq j \leq n_i} (x_{ij})
\]
A subgroup range $R_i$ is included in the calculation only if $n_i \geq 2$. The unbiasing factor $d_2(n_i)$ is defined so that, if the observations are normally distributed, the expected value of $R_i$ is $d_2(n_i)\sigma$. Thus, $\hat{\sigma}$ is the unweighted average of $N$ unbiased estimates of $\sigma$. This method is described by the American Society for Testing and Materials (1976).

The RNOWEIGHT method is available in the BOXCHART, MCHART, MRCHART, RCHART, XCHART, and XRCHART statements. It is the default in the MCHART, MRCHART, RCHART, XCHART, and XRCHART statements.

### SMVLUE Method

If you specify SMETHOD=SMVLUE, a minimum variance linear unbiased estimate (MVLUE) is computed for $\sigma$; see Burr (1969, 1976) and Nelson (1989, 1994). This estimate is a weighted average of $N$ unbiased estimates of $\sigma$ of the form $s_i/c_4(n_i)$, and it is computed as

$$\hat{\sigma} = \frac{h_1 s_1/c_4(n_1) + \cdots + h_N s_N/c_4(n_N)}{h_1 + \cdots + h_N}$$

where

$$h_i = \frac{[c_4(n_i)]^2}{1 - [c_4(n_i)]^2}$$

A subgroup standard deviation $s_i$ is included in the calculation only if $n_i \geq 2$, and $N$ is the number of subgroups for which $n_i \geq 2$. The MVLUE assigns greater weight to estimates of $\sigma$ from subgroups with larger sample sizes, and it is intended for situations where the subgroup sample sizes vary. If the subgroup sample sizes are constant, the SMVLUE estimate reduces to the SNOWEIGHT estimate.

The SMVLUE method is available in the BOXCHART, MCHART, SCHART, XCHART, and XSCHART statements.

### SNOWEIGHT Method

If you specify SMETHOD=SNOWEIGHT, the estimate for $\sigma$ is

$$\hat{\sigma} = \frac{s_1/c_4(n_1) + \cdots + s_N/c_4(n_N)}{N}$$

where $N$ is the number of subgroups for which $n_i \geq 2$, $s_i$ is the sample standard deviation of the $i$th subgroup,

$$s_i = \sqrt{\frac{1}{n_i - 1} \sum_{j=1}^{n_i} (x_{ij} - \bar{X}_i)^2}$$

and

$$c_4(n_i) = \frac{\Gamma(n_i/2)\sqrt{2/(n_i - 1)}}{\Gamma((n_i - 1)/2)}$$

Here $\Gamma(\cdot)$ denotes the gamma function, and $\bar{X}_i$ denotes the $i$th subgroup mean. A subgroup standard deviation $s_i$ is included in the calculation only if $n_i \geq 2$. If the observations are normally distributed, the expected value of $s_i$ is $c_4(n_i)\sigma$. Thus, $\hat{\sigma}$ is the unweighted average of $N$ unbiased estimates of $\sigma$. This method is described by the American Society for Testing and Materials (1976).

The SNOWEIGHT method is available in the BOXCHART, MCHART, SCHART, XCHART, and XSCHART statements. It is the default in the BOXCHART, SCHART, and XSCHART statements.
Constructing Charts for Individual Measurements and Moving Ranges

You produce control charts for individual measurements and moving ranges by using the IRCHART statement. The following notation is used in this section:

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>μ</td>
<td>Process mean (expected value of the population of measurements)</td>
</tr>
<tr>
<td>σ</td>
<td>Process standard deviation (standard deviation of the population of measurements)</td>
</tr>
<tr>
<td>Xi</td>
<td>The ith individual measurement</td>
</tr>
<tr>
<td>X̄</td>
<td>Mean of the individual measurements, computed as ( (X_1 + \cdots + X_N)/N ), where ( N ) is the number of individual measurements</td>
</tr>
<tr>
<td>n</td>
<td>Number of consecutive measurements that are used to calculate the moving ranges (by default, ( n = 2 ))</td>
</tr>
<tr>
<td>Ri</td>
<td>Moving range that is computed for the ith subgroup (corresponding to the ith individual measurement). If ( i &lt; n ), then ( R_i ) is assigned a missing value. Otherwise, ( R_i = \max(X_i, X_{i-1}, \ldots, X_{i-n+1}) - \min(X_i, X_{i-1}, \ldots, X_{i-n+1}) )</td>
</tr>
<tr>
<td>( \bar{R} )</td>
<td>Average of the nonmissing moving ranges, computed as ( \frac{R_n + R_{n+1} + \cdots + R_N}{N + 1 - n} )</td>
</tr>
<tr>
<td>( d_2(n) )</td>
<td>Expected value of the range of ( n ) independent normally distributed variables with unit standard deviation</td>
</tr>
<tr>
<td>( d_3(n) )</td>
<td>Standard error of the range of ( n ) independent observations from a normal population with unit standard deviation</td>
</tr>
</tbody>
</table>

Subgroup Summary Statistics

Each point on an individual measurements chart indicates the value of a measurement \( (X_i) \).

Each point on a moving range chart indicates the value of a moving range \( (R_i) \). For \( n = 2 \), for example, if the first three measurements are 3.4, 3.7, and 3.6, then the first moving range is missing, the second moving range is \( |3.7 - 3.4| = 0.3 \), and the third moving range is \( |3.6 - 3.7| = 0.1 \).

Central Lines

By default, the central line on an individual measurements chart indicates an estimate for \( \mu \), which is computed as \( \bar{X} \). If you specify a known value \( (\mu_0) \) for \( \mu \), the central line indicates the value of \( \mu_0 \).

The central line on a moving range chart indicates an estimate for the expected moving range, which is computed as \( d_2(n)\hat{\sigma} \), where \( \hat{\sigma} = \bar{R}/d_2(n) \). If you specify a known value \( (\sigma_0) \) for \( \sigma \), the central line indicates the value of \( d_2(n)\sigma_0 \).
Control Limits

The control limits are computed as a specified multiple \( k \) of the standard errors of \( X_i \) and \( R_i \) above and below the central line. The default limits are computed using \( k = 3 \) (these are referred to as 3\( \sigma \) limits).

Table 20.19 provides the formulas for the limits.

<table>
<thead>
<tr>
<th>Table 20.19</th>
<th>Limits for Individual Measurements and Moving Range Charts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Individual Measurements Chart</td>
<td></td>
</tr>
<tr>
<td>LCL = lower control limit = ( \bar{X} - k \hat{\sigma} )</td>
<td></td>
</tr>
<tr>
<td>UCL = upper control limit = ( \bar{X} + k \hat{\sigma} )</td>
<td></td>
</tr>
<tr>
<td>Moving Range Chart</td>
<td></td>
</tr>
<tr>
<td>LCL = lower control limit = ( \text{max}(d_2(n)\hat{\sigma} - kd_3(n)\hat{\sigma}, 0) )</td>
<td></td>
</tr>
<tr>
<td>UCL = upper control limit = ( d_2(n)\hat{\sigma} + kd_3(n)\hat{\sigma} )</td>
<td></td>
</tr>
</tbody>
</table>

The formulas assume that the measurements are normally distributed. If the standard values \( \mu_0 \) and \( \sigma_0 \) are available for \( \mu \) and \( \sigma \), replace \( \bar{X} \) with \( \mu_0 \) and \( \hat{\sigma} \) with \( \sigma_0 \) in Table 20.19.

You can specify parameters for the limits as follows:

- Specify \( k \) by using the SIGMAS= option or the variable _SIGMAS_ in a LIMITS= data table.
- Specify \( n \) by using the LIMITN= option or the variable _LIMITN_ in a LIMITS= data table.
- Specify \( \mu_0 \) by using the variable _MEAN_ in the LIMITS= data table.
- Specify \( \sigma_0 \) by using the variable _STDDEV_ in the LIMITS= data table.

Constructing Charts for Means

You produce control charts for means by using the BOXCHART, XCHART, XRCHART, or XSCHART statement. The following notation is used in this section:

- \( \mu \): Process mean (expected value of the population of measurements)
- \( \sigma \): Process standard deviation (standard deviation of the population of measurements)
- \( \bar{X}_i \): Mean of measurements in \( i \)th subgroup
- \( n_i \): Sample size of \( i \)th subgroup
- \( R_i \): Range of measurements in \( i \)th subgroup
- \( \bar{X} \): Weighted average of subgroup means
**Subgroup Summary Statistic**

Each point on an $\bar{X}$ chart indicates the value of a subgroup mean ($\bar{X}_i$). For example, if the 10th subgroup contains the values 12, 15, 19, 16, and 14, the subgroup summary statistic is

$$\bar{X}_{10} = \frac{12 + 15 + 19 + 16 + 14}{5} = 15.2$$

**Central Line**

By default, the central line on an $\bar{X}$ chart indicates an estimate for $\mu$, which is computed as

$$\hat{\mu} = \bar{X} = \frac{n_1 \bar{X}_1 + \cdots + n_N \bar{X}_N}{n_1 + \cdots + n_N}$$

If you specify a known value ($\mu_0$) for $\mu$, the central line indicates the value of $\mu_0$.

**Control Limits**

The limits are computed as a specified multiple ($k$) of the standard error of $\bar{X}_i$ above and below the central line. The default limits are computed using $k = 3$ (these are referred to as $3\sigma$ limits).

Table 20.21 provides the formulas for the limits.

<table>
<thead>
<tr>
<th>Table 20.21</th>
<th>Limits for $\bar{X}$ Charts</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCL = lower limit = $\bar{X} - k\hat{\sigma} \sqrt{n_i}$</td>
<td></td>
</tr>
<tr>
<td>UCL = upper limit = $\bar{X} + k\hat{\sigma} \sqrt{n_i}$</td>
<td></td>
</tr>
</tbody>
</table>

Note that the limits vary with $n_i$. If the standard values $\mu_0$ and $\sigma_0$ are available for $\mu$ and $\sigma$, respectively, replace $\bar{X}$ with $\mu_0$ and $\hat{\sigma}$ with $\sigma_0$ in Table 20.21.

You can specify parameters for the limits as follows:

- Specify $k$ by using the `SIGMAS=` option or the variable `_SIGMAS_` in a `LIMITS=` data table.
- Specify a constant nominal sample size $n_i \equiv n$ for the control limits by using the `LIMITN=` option or the variable `_LIMITN_` in a `LIMITS=` data table.
- Specify $\mu_0$ by using the variable `_MEAN_` in a `LIMITS=` data table.
- Specify $\sigma_0$ by using the variable `_STDDEV_` in a `LIMITS=` data table.

**Constructing Charts for Medians**

You produce control charts for medians by using the `MCHART` or `MRCHART` statement, and by using the `BOXCHART` statement when you specify `CONTROLSTAT=MEAN`. The following notation is used in this section:
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mu$</td>
<td>Process mean (expected value of the population of measurements)</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>Process standard deviation (standard deviation of the population of measurements)</td>
</tr>
<tr>
<td>$n_i$</td>
<td>Sample size of $i$th subgroup</td>
</tr>
<tr>
<td>$N$</td>
<td>Number of subgroups</td>
</tr>
<tr>
<td>$x_{ij}$</td>
<td>$j$th measurement in the $i$th subgroup, $j = 1, 2, \ldots, n_i$</td>
</tr>
<tr>
<td>$x_{i(j)}$</td>
<td>$j$th-largest measurement in the $i$th subgroup. Then $x_{i(1)} \leq x_{i(2)} \leq \cdots \leq x_{i(n_i)}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\bar{X}$</td>
<td>Weighted average of subgroup means</td>
</tr>
<tr>
<td>$M_i$</td>
<td>Median of the measurements in the $i$th subgroup: $M_i = \begin{cases} x_{i((n_i+1)/2)} &amp; \text{if } n_i \text{ is odd} \ (x_{i(n_i/2)} + x_{i((n_i+1)/2)})/2 &amp; \text{if } n_i \text{ is even} \end{cases}$</td>
</tr>
<tr>
<td>$\bar{M}$</td>
<td>Average of the subgroup medians: $\bar{M} = (n_1 M_1 + \cdots + n_N M_N)/(n_1 + \cdots + n_N)$</td>
</tr>
<tr>
<td>$\tilde{M}$</td>
<td>Median of the subgroup medians. Denote the $j$th-largest median by $M_{(j)}$ so that $M_{(1)} \leq M_{(2)} \leq \cdots \leq M_{(N)}$. Then $\tilde{M} = \begin{cases} M_{(N+1)/2} &amp; \text{if } N \text{ is odd} \ (M_{(N/2)} + M_{(N/2+1)})/2 &amp; \text{if } N \text{ is even} \end{cases}$</td>
</tr>
<tr>
<td>$e_M(n)$</td>
<td>Standard error of the median of $n$ independent, normally distributed variables with unit standard deviation (the value of $e_M(n)$ can be calculated using the STDMED function in a DATA step)</td>
</tr>
</tbody>
</table>

**Subgroup Summary Statistics**

Each point on a median chart indicates the value of a subgroup median ($M_i$). For example, if the 10th contains the values 12, 15, 19, 16, and 14, the subgroup summary statistic is $M_{10} = 15$.

**Central Line**

The value of the central line indicates an estimate of $\mu$, which is computed as follows:

- $\bar{X}$ by default
- $\bar{M}$ when you specify MEDCENTRAL=AVGMEAN
- $\tilde{M}$ when you specify MEDCENTRAL=MEDMED
Control Limits

The control limits are computed as a specified multiple \( k \) of the standard error of \( M_i \) above and below the central line. The default limits are computed using \( k = 3 \) (these are referred to as 3\( \sigma \) limits).

Table 20.23 provides the formulas for the limits.

<table>
<thead>
<tr>
<th>Table 20.23 Limits for Median Charts</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCL = lower limit = ( \bar{M} - k\hat{\sigma}_M(n_i) )</td>
</tr>
<tr>
<td>UCL = upper limit = ( \bar{M} + k\hat{\sigma}_M(n_i) )</td>
</tr>
</tbody>
</table>

Note that the limits vary with \( n_i \). In Table 20.23, replace \( \bar{M} \) with \( \bar{X} \) if you specify MEDCENTRAL=AVGMEAN, and replace \( \bar{M} \) with \( \bar{M} \) if you specify MEDCENTRAL=MEDMED. The formulas assume that the data are normally distributed.

You can specify parameters for the limits as follows:

- Specify \( k \) by using the SIGMAS= option or the variable _SIGMAS_ in a LIMITS= data table.
- Specify a constant nominal sample size \( n_i \equiv n \) for the control limits by using the LIMITN= option or the variable _LIMITN_ in a LIMITS= data table.
- Specify \( \mu_0 \) by using the variable _MEAN_ in the LIMITS= data table.
- Specify \( \sigma_0 \) by using the variable _STDDEV_ in the LIMITS= data table.

Constructing Charts for Ranges

You produce control charts for ranges by using the MRCHART, RCHART, or XRCHART statement. The following notation is used in this section:

\[
\begin{align*}
\sigma & \quad \text{Process standard deviation (standard deviation of the population of measurements)} \\
R_i & \quad \text{Range of measurements in } i\text{th subgroup} \\
n_i & \quad \text{Sample size of } i\text{th subgroup} \\
d_2(n) & \quad \text{Expected value of the range of } n \text{ independent normally distributed variables with unit standard deviation} \\
d_3(n) & \quad \text{Standard error of the range of } n \text{ independent observations from a normal population with unit standard deviation}
\end{align*}
\]

Subgroup Summary Statistics

Each point on an R chart indicates the value of a subgroup range \( R_i \). For example, if the 10th subgroup contains the values 12, 15, 19, 16, and 14, the subgroup summary statistic is \( R_{10} = 19 - 12 = 7 \).
Central Line

By default, the central line for the $i$th subgroup indicates an estimate of the expected value of $R_i$, which is computed as $d_2(n_i)\hat{\sigma}$, where $\hat{\sigma}$ is an estimate of $\sigma$. If you specify a known value ($\sigma_0$) for $\sigma$, the central line indicates the value of $d_2(n_i)\sigma_0$. Note that the central line varies with $n_i$.

Control Limits

The control limits are computed as a specified multiple ($k$) of the standard error of $R_i$ above and below the central line. The default limits are computed using $k = 3$ (these are referred to as $3\sigma$ limits).

Table 20.25 provides the formulas for the limits.

<table>
<thead>
<tr>
<th>Table 20.25</th>
<th>Control Limits for $R$ Charts</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LCL = lower limit = $\max(d_2(n_i)\hat{\sigma} - kd_3(n_i)\hat{\sigma}, 0)$</td>
</tr>
<tr>
<td></td>
<td>UCL = upper limit = $d_2(n_i)\hat{\sigma} + kd_3(n_i)\hat{\sigma}$</td>
</tr>
</tbody>
</table>

The formulas assume that the data are normally distributed. Note that the control limits vary with $n_i$. If a standard value $\sigma_0$ is available for $\sigma$, replace $\hat{\sigma}$ with $\sigma_0$ in Table 20.25.

You can specify parameters for the limits as follows:

- Specify $k$ by using the SIGMAS= option or the variable _SIGMAS_ in a LIMITS= data table.
- Specify a constant nominal sample size $n_i \equiv n$ for the control limits by using the LIMITN= option or the variable _LIMITN_ in a LIMITS= data table.
- Specify $\sigma_0$ by using the variable _STDDEV_ in a LIMITS= data table.

Constructing Charts for Standard Deviations

You produce control charts for standard deviations by using the SCHART or XSCHART statement. The following notation is used in this section:

- $\sigma$ Process standard deviation (standard deviation of the population of measurements)
- $s_i$ Standard deviation of measurements in $i$th subgroup
  
  \[
  s_i = \sqrt{(1/(n_i - 1))((x_{i1} - \bar{X}_i)^2 + \cdots + (x_{in_i} - \bar{X}_i)^2)}
  \]

- $n_i$ Sample size of $i$th subgroup
- $c_4(n)$ Expected value of the standard deviation of $n$ independent normally distributed variables with unit standard deviation
- $c_5(n)$ Standard error of the standard deviation of $n$ independent observations from a normal population with unit standard deviation
Subgroup Summary Statistics

Each point on an $s$ chart indicates the value of a subgroup standard deviation ($s_i$). For example, if the 10th subgroup contains the values 12, 15, 19, 16, and 13, the subgroup summary statistic is

$$s_{10} = \sqrt{\frac{(12 - 15)^2 + (15 - 15)^2 + (19 - 15)^2 + (16 - 15)^2 + (13 - 15)^2}{4}} = 2.739$$

Central Line

By default, the central line for the $i$th subgroup indicates an estimate for the expected value of $s_i$, which is computed as $c_4(n_i)\hat{\sigma}$, where $\hat{\sigma}$ is an estimate of $\sigma$. If you specify a known value ($\sigma_0$) for $\sigma$, the central line indicates the value of $c_4(n_i)\sigma_0$. Note that the central line varies with $n_i$.

Control Limits

The control limits are computed as a specified multiple ($k$) of the standard error of $s_i$ above and below the central line. The default limits are computed using $k = 3$ (these are referred to as $3\sigma$ limits).

Table 20.27 provides the formulas for the limits.

<table>
<thead>
<tr>
<th>Table 20.27</th>
<th>Control Limits for $s$ Charts</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCL = lower limit = $\max(c_4(n_i)\hat{\sigma} - k\sigma_5(n_i)\hat{\sigma}, 0)$</td>
<td></td>
</tr>
<tr>
<td>UCL = upper limit = $c_4(n_i)\hat{\sigma} + k\sigma_5(n_i)\hat{\sigma}$</td>
<td></td>
</tr>
</tbody>
</table>

The formulas assume that the data are normally distributed. If a standard value $\sigma_0$ is available for $\sigma$, replace $\hat{\sigma}$ with $\sigma_0$ in Table 20.27. Note that the upper and lower limits vary with $n_i$.

You can specify parameters for the limits as follows:

- Specify $k$ by using the SIGMAS= option or the variable _SIGMAS_ in a LIMITS= data table.
- Specify a constant nominal sample size $n_i \equiv n$ for the control limits by using the LIMITN= option or the variable _LIMITN_ in a LIMITS= data table.
- Specify $\sigma_0$ by using the variable _STDDEV_ in a LIMITS= data table.

Constructing Charts for Numbers of Nonconformities (c Charts)

You produce control charts for numbers of nonconformities by using the CCHART statement.

The following notation is used in this section:

| $u$ | Expected number of nonconformities per unit that the process produces |
| $u_i$ | Number of nonconformities per unit in the $i$th subgroup |
| $c_i$ | Total number of nonconformities in the $i$th subgroup |
$n_i$  Number of inspection units in the $i$th subgroup. Typically, $n_i = 1$ and $u_i = c_i$ for $c$ charts. In general, $u_i = c_i / n_i$.

$\bar{u}$  Average number of nonconformities per unit taken across subgroups. The quantity $\bar{u}$ is computed as a weighted average:

$$\bar{u} = \frac{n_1u_1 + \cdots + n_Nu_N}{n_1 + \cdots + n_N} = \frac{c_1 + \cdots + c_N}{n_1 + \cdots + n_N}$$

$N$  Number of subgroups

### Subgroup Summary Statistics

Each point on a $c$ chart represents the total number of nonconformities ($c_i$) in a subgroup. For example, Figure 20.5 displays three sections of pipeline that are inspected for defective welds (indicated by an X). Each section represents a subgroup that consists of a number of inspection units, which are 1,000-foot sections. The number of units in the $i$th subgroup is denoted by $n_i$, which is the subgroup sample size. The value of $n_i$ can be fractional; Figure 20.5 shows $n_3 = 2.5$ units in the third subgroup.

**Figure 20.5** Terminology for $c$ Charts and $u$ Charts

<table>
<thead>
<tr>
<th>Subgroup 1</th>
<th>Subgroup 2</th>
<th>Subgroup 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>2.0</td>
<td>1.0</td>
<td>2.5</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>0.5</td>
<td>2.0</td>
<td>0.8</td>
</tr>
</tbody>
</table>

The number of nonconformities in the $i$th subgroup is denoted by $c_i$. The number of nonconformities per unit in the $i$th subgroup is denoted by $u_i = c_i / n_i$. In Figure 20.5, the number of welds per inspection unit in the third subgroup is $u_3 = 2 / 2.5 = 0.8$.

A $u$ chart that is created by the UCHART statement plots the quantity $u_i$ for the $i$th subgroup (see the section “Constructing Charts for Nonconformities per Unit (u Charts)” on page 971). An advantage of a $u$ chart is that the value of the central line at the $i$th subgroup does not depend on $n_i$. This is not the case for a $c$ chart, so a $u$ chart is often preferred when the number of units $n_i$ is not constant across subgroups.
Central Line

On a np chart, the central line indicates an estimate for $n_i \bar{u}$, which is computed as $n_i \bar{u}$. If you specify a known value ($u_0$) for $u$, the central line indicates the value of $n_i u_0$.

Note that the central line varies with subgroup sample size $n_i$. When $n_i = 1$ for all subgroups, the central line has the constant value $\bar{c} = (c_1 + \cdots + c_N)/N$.

Control Limits

The control limits are computed as a specified multiple ($k$) of the standard error of $c_i$ above and below the central line. The default limits are computed using $k = 3$ (these are referred to as $3\sigma$ limits).

The lower and upper control limits, LCL and UCL, respectively, are given by

$$LCL = \max\left(n_i \bar{u} - k \sqrt{n_i \bar{u}}, 0\right)$$

$$UCL = n_i \bar{u} + k \sqrt{n_i \bar{u}}$$

The lower and upper control limits vary with the number of inspection units per subgroup $n_i$. If $n_i = 1$ for all subgroups, the control limits have constant values, as follows:

$$LCL = \max\left(\bar{c} - k \sqrt{c}, 0\right)$$

$$UCL = \bar{c} + k \sqrt{c}$$

If a standard value $u_0$ is available for $u$, replace $\bar{u}$ with $u_0$ in the formulas for the control limits. You can specify parameters for the limits as follows:

- Specify $k$ by using the SIGMAS= option or the variable _SIGMAS_ in a LIMITS= data table.
- Specify a constant nominal sample size $n_i \equiv n$ for the control limits by using the LIMITN= option or the variable _LIMITN_ in a LIMITS= data table.
- Specify $u_0$ by using the variable _U_ in a LIMITS= data table.

Constructing Charts for Number Nonconforming (np Charts)

You produce control charts for number nonconforming by using the NPCHART statement.

The following notation is used in this section:

- $p$: Expected proportion of nonconforming items that the process produces
- $p_i$: Proportion of nonconforming items in the $i$th subgroup
- $X_i$: Number of nonconforming items in the $i$th subgroup
- $n_i$: Number of items in the $i$th subgroup
\[ \bar{p} = \frac{n_1 p_1 + \cdots + n_N p_N}{n_1 + \cdots + n_N} = \frac{X_1 + \cdots + X_N}{n_1 + \cdots + n_N} \]

\( N \)  Number of subgroups

Subgroup Summary Statistics

Each point on an \( np \) chart represents the observed number \( (X_i) \) of nonconforming items in a subgroup. For example, in Figure 20.6 the first subgroup contains 12 items, of which 3 are nonconforming. The summary statistic for the first subgroup is \( X_1 = 3 \).

**Figure 20.6** Proportions versus Counts

Note that a \( p \) chart displays the proportion of nonconforming items \( p_i \). You can use the PCHART statement to create \( p \) charts; for more information, see the section “Constructing Charts for Proportion Nonconforming (p Charts)” on page 969.

Central Line

By default, the central line on an \( np \) chart indicates an estimate for \( n_i \bar{p} \), which is computed as \( n_i \bar{p} \). If you specify a known value \( (p_0) \) for \( p \), the central line indicates the value of \( n_i p_0 \). Note that the central line varies with \( n_i \).

Control Limits

The control limits are computed as a specified multiple \( (k) \) of the standard error of \( X_i \) above and below the central line. The default limits are computed using \( k = 3 \) (these are referred to as 3\( \sigma \) limits).
The lower and upper control limits, LCL and UCL, respectively, are computed as

\[
\text{LCL} = \max \left( n_i \bar{p} - k \sqrt{n_i \bar{p}(1 - \bar{p})}, 0 \right)
\]
\[
\text{UCL} = \min \left( n_i \bar{p} + k \sqrt{n_i \bar{p}(1 - \bar{p})}, n_i \right)
\]

You can specify parameters for the limits as follows:

- Specify \( k \) by using the SIGMAS= option or the variable _SIGMAS_ in a LIMITS= data table.
- Specify a constant nominal sample size \( n \) for the control limits by using the LIMITN= option or the variable _LIMITN_ in a LIMITS= data table.
- Specify \( p_0 \) by using the variable _P_ in the LIMITS= data table.

---

**Constructing Charts for Proportion Nonconforming (p Charts)**

You produce control charts for proportion nonconforming by using the PCHART statement.

The following notation is used in this section:

- \( p \) Expected proportion of nonconforming items that the process produces
- \( p_i \) Proportion of nonconforming items in the \( i \)th subgroup
- \( X_i \) Number of nonconforming items in the \( i \)th subgroup
- \( n_i \) Number of items in the \( i \)th subgroup
- \( \bar{p} \) Average proportion of nonconforming items taken across subgroups:
  \[
  \bar{p} = \frac{n_1 p_1 + \cdots + n_N p_N}{n_1 + \cdots + n_N} = \frac{X_1 + \cdots + X_N}{n_1 + \cdots + n_N}
  \]
- \( N \) Number of subgroups

**Subgroup Summary Statistics**

Each point on a \( p \) chart represents the observed proportion \( (p_i = X_i/n_i) \) of nonconforming items in a subgroup. For example, in Figure 20.7 the second subgroup contains 16 items, of which 2 are nonconforming. The summary statistic for the second subgroup is \( p_2 = 2/16 = 0.125 \).
Figure 20.7 Proportions versus Counts

Note that an $np$ chart displays the number (count) of nonconforming items $X_i$. You can use the NPCHART statement to create $np$ charts; for more information, see the section “Constructing Charts for Number Nonconforming ($np$ Charts)” on page 967.

Central Line

By default, the central line on a $p$ chart indicates an estimate of $p$ that is computed as \( \bar{p} \). If you specify a known value ($p_0$) for $p$, the central line indicates the value of $p_0$.

Control Limits

The control limits are computed as a specified multiple ($k$) of the standard error of $p_i$ above and below the central line. The default limits are computed using $k = 3$ (these are referred to as 3\(\sigma\) limits).

The lower and upper control limits, LCL and UCL, respectively, are computed as

$$
\text{LCL} = \max \left( \bar{p} - k \sqrt{\bar{p}(1 - \bar{p})/n_i}, 0 \right)
$$

$$
\text{UCL} = \min \left( \bar{p} + k \sqrt{\bar{p}(1 - \bar{p})/n_i}, 1 \right)
$$

You can specify parameters for the limits as follows:

- Specify $k$ by using the SIGMAS= option or the variable _SIGMAS_ in a LIMITS= data table.
- Specify a constant nominal sample size $n_i$ for the control limits by using the LIMITN= option or the variable _LIMITN_ in a LIMITS= data table.
- Specify $p_0$ by using the variable _P_ in a LIMITS= data table.
**Constructing Charts for Nonconformities per Unit (u Charts)**

You produce control charts for nonconformities per unit by using the UCHART statement.

The following notation is used in this section:

- \( u \) Expected number of nonconformities per unit that the process produces
- \( u_i \) Number of nonconformities per unit in the \( i \)th subgroup. In general, \( u_i = c_i / n_i \).
- \( c_i \) Total number of nonconformities in the \( i \)th subgroup
- \( n_i \) Number of inspection units in the \( i \)th subgroup
- \( \bar{u} \) Average number of nonconformities per unit taken across subgroups. The quantity \( \bar{u} \) is computed as a weighted average:

\[
\bar{u} = \frac{n_1u_1 + \cdots + n_Nu_N}{n_1 + \cdots + n_N} = \frac{c_1 + \cdots + c_N}{n_1 + \cdots + n_N}
\]

- \( N \) Number of subgroups

**Subgroup Summary Statistics**

Each point on a \( u \) chart indicates the number of nonconformities per unit (\( u_i \)) in a subgroup. For example, Figure 20.8 displays three sections of pipeline that are inspected for defective welds (indicated by an X). Each section represents a subgroup that consists of a number of inspection units, which are 1,000-foot sections. The number of units in the \( i \)th subgroup is denoted by \( n_i \), which is the subgroup sample size.

**Figure 20.8** Terminology for \( c \) Charts and \( u \) Charts

<table>
<thead>
<tr>
<th>One unit = 1000 feet</th>
<th>( n_i )</th>
<th>( c_i )</th>
<th>( u_i )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subgroup 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2.0 1 0.5</td>
</tr>
<tr>
<td>Subgroup 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.0 2 2.0</td>
</tr>
<tr>
<td>Subgroup 3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2.5 2 0.8</td>
</tr>
</tbody>
</table>
The number of nonconformities in the ith subgroup is denoted by \( c_i \). The number of nonconformities per unit in the ith subgroup is denoted by \( u_i = c_i / n_i \). In Figure 20.8, the number of defective welds per unit in the third subgroup is \( u_3 = 2 / 2.5 = 0.8 \).

A \( u \) chart plots the quantity \( u_i \) for the ith subgroup. A \( c \) chart plots the quantity \( c_i \) for the ith subgroup (for more information, see the section “Constructing Charts for Numbers of Nonconformities (c Charts)” on page 965). An advantage of a \( u \) chart is that the value of the central line at the ith subgroup does not depend on \( n_i \). This is not the case for a \( c \) chart, so a \( u \) chart is often preferred when the number of units \( n_i \) is not constant across subgroups.

Central Line

On a \( u \) chart, the central line indicates an estimate of \( u \), which is computed as \( \bar{u} \) by default. If you specify a known value (\( u_0 \)) for \( u \), the central line indicates the value of \( u_0 \).

Control Limits

The control limits are computed as a specified multiple (\( k \)) of the standard error of \( u_i \) above and below the central line. The default limits are computed using \( k = 3 \) (these are referred to as 3\( \sigma \) limits).

The lower and upper control limits, LCL and UCL, respectively, are given by

\[
\begin{align*}
\text{LCL} &= \max \left( \bar{u} - k \sqrt{\frac{\bar{u}}{n_i}} , 0 \right) \\
\text{UCL} &= \bar{u} + k \sqrt{\frac{\bar{u}}{n_i}}
\end{align*}
\]

The limits vary with \( n_i \).

You can specify parameters for the limits as follows:

- Specify \( k \) by using the SIGMAS= option or the variable _SIGMAS_ in a LIMITS= data table.
- Specify a constant nominal sample size \( n_i = n \) for the control limits by using the LIMITN= option or the variable _LIMITN_ in a LIMITS= data table.
- Specify \( u_0 \) by using the variable _U_ in a LIMITS= data table.

Percentile Definitions

You can use the PCTLDEF= option in the BOXCHART statement to specify one of five definitions for computing quantile statistics (percentiles). Let \( n \) equal the number of nonmissing values for a variable, and let \( x_1, x_2, \ldots, x_n \) represent the ordered values of the process variable. For the \( t \)th percentile, set \( p = t / 100 \), and express \( np \) as

\[ np = j + g \]

where \( j \) is the integer part of \( np \), and \( g \) is the fractional part of \( np \).

The \( t \)th percentile (call it \( y \)) can be defined in five ways, as described in the next five sections.
Tests for Special Causes

This section provides details about tests for special causes that you can apply by using the SPC procedure.

The SPC procedure provides eight standard tests for special causes, also referred to as rules for lack of control, supplementary rules, runs tests, runs rules, pattern tests, and Western Electric rules. These tests improve the sensitivity of the Shewhart chart to small changes in the process. You can also improve the sensitivity of the chart by increasing the rate of sampling, increasing the subgroup sample size, and using control limits that represent less than three standard errors of variation from the central line. However, increasing the sampling rate and sample size is often impractical, and tightening the control limits increases the chances of falsely signaling an out-of-control condition. By detecting particular nonrandom patterns in

PCTLDEF=1

This definition uses the weighted average at $x_{np}$.

$$y = (1 - g)x_j + gx_{j+1}$$

where $x_0$ is taken to be $x_1$.

PCTLDEF=2

This definition uses the observation numbered closest to $np$.

$$y = x_i$$

where $i$ is the integer part of $np + 1/2$.

PCTLDEF=3

This definition uses the empirical distribution function:

$$y = x_j \quad \text{if } g = 0$$
$$y = x_{j+1} \quad \text{if } g > 0$$

PCTLDEF=4

This definition uses the weighted average aimed at $x_{p(n+1)}$.

$$y = (1 - g)x_j + gx_{j+1}$$

where $(n + 1)p = j + g$, and where $x_{n+1}$ is taken to be $x_n$.

PCTLDEF=5

This definition uses the empirical distribution function with averaging:

$$y = (x_j + x_{j+1})/2 \quad \text{if } g = 0$$
$$y = x_{j+1} \quad \text{if } g > 0$$
the subgroup summary statistics, the tests can provide greater sensitivity and useful diagnostic information while incurring a reasonable probability of a false signal.

The patterns that the eight standard tests detect are defined in Table 20.32 and Table 20.33 and illustrated in Figure 20.9 and Figure 20.10. All eight tests were developed for use with fixed \( 3\sigma \) limits. The tests are indexed according to the numbering sequence used by Nelson (1984, 1985). You can request any combination of the eight tests by specifying the test indices in the TESTS= option in the BOXCHART, CCHART, IRCHART, MCHART, MRCHART, NPCHART, PCHART, UCHART, XCHART, XRCHART, and XSCHART statements.

The following restrictions apply to the tests:

- Only Tests 1, 2, 3, and 4 are recommended for \( c \), \( np \), \( p \), and \( u \) charts that you create using the CCHART, NPCHART, PCHART, and UCHART statements, respectively. In these four cases, you should not use Test 2 unless the process distribution is symmetric or nearly symmetric.

- By default, the TESTS= option is not applied when the control limits are not \( 3\sigma \) limits or vary with subgroup sample size. You can use the NO3SIGMACHECK option to request tests for special causes when the SIGMAS= option specifies control limits other than \( 3\sigma \) limits. This is not recommended for standard control chart applications, because the standard tests for special causes are based on \( 3\sigma \) limits. You can apply tests for special causes when control limits vary with subgroup sample size by using the LIMITN= or TESTNSTD option (for more information, see the section “Applying Tests with Varying Subgroup Sample Sizes” on page 978).

### Table 20.32 Definitions of Tests 1 to 4

<table>
<thead>
<tr>
<th>Test Index</th>
<th>Pattern Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>One point beyond Zone A (outside the control limits)</td>
</tr>
<tr>
<td>2</td>
<td>Nine points in a row in Zone C or beyond on one side of the central line (see Note 1 after Figure 20.9)</td>
</tr>
<tr>
<td>3</td>
<td>Six points in a row that steadily increase or steadily decrease (see Note 2 after Figure 20.9)</td>
</tr>
<tr>
<td>4</td>
<td>Fourteen points in a row that alternate up and down</td>
</tr>
</tbody>
</table>
Figure 20.9 Examples of Tests 1 to 4

Notes:

1. You can specify the number of points in Test 2 as 7, 8, 9, 11, 14, or 20 in the TEST2RUN= option.

2. You can specify the number of points in Test 3 as 6, 7, or 8 in the TEST3RUN= option.
Table 20.33  Definitions of Tests 5 to 8

<table>
<thead>
<tr>
<th>Test Index</th>
<th>Pattern Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Two out of three points in a row in Zone A or beyond</td>
</tr>
<tr>
<td>6</td>
<td>Four out of five points in a row in Zone B or beyond</td>
</tr>
<tr>
<td>7</td>
<td>Fifteen points in a row in Zone C on either or both sides of the central line</td>
</tr>
<tr>
<td>8</td>
<td>Eight points in a row on either or both sides of the central line, with no points in Zone C</td>
</tr>
</tbody>
</table>

Figure 20.10  Examples of Tests 5 to 8
Interpreting Tests for Special Causes

Nelson (1984, 1985) makes the following comments about interpreting the tests for special causes:

- When a process is in statistical control, the chance of a false signal for each test is less than 5 in 1,000.
- Test 1 is positive if there is a shift in the process mean, if there is an increase in the process standard deviation, or if there is a "single aberration in the process such as a mistake in calculation, an error in measurement, bad raw material, a breakdown of equipment, and so on" (Nelson 1985).
- Test 2 signals a shift in the process mean. The use of nine points (rather than seven, as in Grant and Leavenworth 1988 for the pattern that defines Test 2 makes the chance of a false signal comparable to that of Test 1. (To control the number of points for the pattern in test 2, use the TEST2RUN= option in the chart statement.)
- Test 3 signals a drift in the process mean. Nelson (1985) states that causes can include “tool wear, depletion of chemical baths, deteriorating maintenance, improvement in skill, and so on.”
- Test 4 signals “a systematic effect such as produced by two machines, spindles, operators or vendors used alternately” (Nelson 1985).
- Tests 1, 2, 3, and 4 should be applied routinely; the combined chance of a false signal from one or more of these tests is less than 1 in 100. Nelson (1985) describes these tests as “a good set that will react to many commonly occurring special causes.”
- In the case of charts for variables, the first four tests should be augmented by Tests 5 and 6 when earlier warning is desired. The chance of a false signal increases to 2 in 100.
- Tests 7 and 8 indicate stratification (observations in a subgroup have multiple sources with different means). Test 7 is positive when the observations in the subgroup always have multiple sources. Test 8 is positive when the subgroups are taken from one source at a time.

Nelson (1985) also comments that “the probabilities quoted for getting false signals should not be considered to be very accurate” because the probabilities are based on assumptions of normality and independence that might not be satisfied. Consequently, he recommends that the tests “should be viewed as simply practical rules for action rather than tests having specific probabilities associated with them.” Nelson cautions that “it is possible, though unlikely, for a process to be out of control yet not show any signals from these eight tests.”

Modifying Standard Tests for Special Causes

Some textbooks and references present slightly different versions of Tests 2 and 3. You can use the following options to request these modifications:

- TEST2RUN=run-length specifies the length of the pattern for Test 2. The form of the test for each run-length is shown in the following table. The default run-length is 9.
Chapter 20: The SPC Procedure

<table>
<thead>
<tr>
<th>run-length</th>
<th>Number of Points on One Side of the Central Line</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>7 in a row</td>
</tr>
<tr>
<td>8</td>
<td>8 in a row</td>
</tr>
<tr>
<td>9</td>
<td>9 in a row</td>
</tr>
<tr>
<td>11</td>
<td>At least 10 out of 11 in a row</td>
</tr>
<tr>
<td>14</td>
<td>At least 12 out of 14 in a row</td>
</tr>
<tr>
<td>20</td>
<td>At least 16 out of 20 in a row</td>
</tr>
</tbody>
</table>

- TEST3RUN=run-length specifies the length of the pattern for Test 3. The run-length values that you can specify are 6, 7, and 8. The default run-length is 6.

The Western Electric Company (now AT&T) Statistical Quality Control Handbook (1956) and Montgomery (1996) discuss a test that is signaled by eight points in a row in Zone C or beyond (on one side of the central line). You can request this test by specifying TESTS=2 and TEST2RUN=8. The Handbook also discusses tests that correspond to Tests 1, 5, 6, 7, and 8.

Kume (1985) recommends a number of tests for special causes that can be regarded as modifications of Tests 2 and 3:

- seven points in a row on one side of the central line. Specify TESTS=2 and TEST2RUN=7.
- at least 10 out of 11 points in a row on one side of the central line. Specify TESTS=2 and TEST2RUN=11.
- at least 12 out of 14 points in a row on one side of the central line. Specify TESTS=2 and TEST2RUN=14.
- at least 16 out of 20 points in a row on one side of the central line. Specify TESTS=2 and TEST2RUN=20.
- seven points in a row that steadily increase or decrease. Specify TESTS=3 and TEST3RUN=7.

Applying Tests with Varying Subgroup Sample Sizes

Nelson (1989, 1994) describes the use of standardization to apply the tests for special causes to data that involve varying subgroup samples. This approach applies the tests to the standardized subgroup statistics, setting the control limits at $\pm 3$ and the zone boundaries at $\pm 1$ and $\pm 2$. For example, for an $\bar{X}$ chart with subgroup means $\bar{X}_i$ and varying subgroup sample sizes $n_i$, the tests are applied to the standardized values $z_i = (\bar{X}_i - \bar{X}) / (s / \sqrt{n_i})$, where $\bar{X}$ estimates the process mean and $s$ estimates the process standard deviation. You can request this method by using the TESTNSTD option.

Alternatively, you can handle varying subgroup sample sizes by specifying a nominal sample size in the LIMITN= option. If you specify LIMITN=n, the control limits are computed for the fixed value $n$, and they do not vary with the subgroup sample sizes. Moreover, subgroup summary statistics are computed only for those subgroups whose sample size is equal to $n$; only those subgroups are included in the results. You can specify the ALLN option in conjunction with the LIMITN= option to force subgroup summary statistics to be computed for all subgroups, regardless of subgroup sample size.
Capability Indices

This section provides formulas for process capability indices, which are computed for stable processes when you use the SPECS= option to specify a data table that contains process specification limits. The estimate $\hat{\sigma}$ is computed as described in the section “Methods of Estimating the Standard Deviation” on page 956.

The Index $C_p$

The process capability index $C_p$ is computed as

$$C_p = \frac{(USL - LSL)}{6\hat{\sigma}}$$

If you do not specify both LSL and USL, the variable _CP_ is assigned a missing value.

The Index CPL

The process capability index CPL is computed as

$$CPL = \frac{(\bar{X} - LSL)}{3\hat{\sigma}}$$

If you do not specify LSL, the variable _CPL_ is assigned a missing value.

The Index CPU

The process capability index CPU is computed as

$$CPU = \frac{(USL - \bar{X})}{3\hat{\sigma}}$$

If you do not specify USL, the variable _CPU_ is assigned a missing value.

The Index $C_{pk}$

The process capability index $C_{pk}$ is computed as

$$C_{pk} = \min(USL - \bar{X}, \bar{X} - LSL)/3\hat{\sigma}$$

If you specify only USL, the index $C_{pk}$ is computed as

$$C_{pk} = (USL - \bar{X})/3\hat{\sigma}$$

and if you specify only LSL, the index $C_{pk}$ is computed as

$$C_{pk} = (\bar{X} - LSL)/3\hat{\sigma}$$
The Index $C_{pm}$

The process capability index $C_{pm}$ is computed as

$$C_{pm} = \frac{\min(T - LSL, USL - T)}{3\sqrt{\hat{\sigma}^2 + (\overline{X} - T)^2}}$$

where $T$ is the target value that is specified in the Target variable in the SPECS= data table.

When you specify a single specification limit (SL) and target, $C_{pm}$ is computed as

$$C_{pm} = \frac{|T - SL|}{3\sqrt{\hat{\sigma}^2 + (\overline{X} - T)^2}}$$

You should verify that a process is in statistical control before interpreting its capability indices, and you should verify that the data are normally distributed. A process must be stable and predictable before you can effectively address its ability to produce output within specifications. PROC SPC computes capability indices only for processes that do not show signs of being out of statistical control.

Some references use different notation and names for capability indices. For example, the manual ASQC Automotive Division/AIAG (1990) uses the term “process capability indices” for the indices that are computed by PROC SPC, and it uses the term “process performance indices” for indices that are computed by using the overall sample standard deviation as the estimate of $\hat{\sigma}$.

Input Data Tables

Data= Data Table

The SPC procedure accepts a primary input data table that you specify in the DATA= option in the PROC SPC statement. This data table contains measurements of one or more process quality characteristics. Data to be analyzed by using the CCHART, IRCHART, NPCHART, PCHART, or UCHART statement should contain one observation per subgroup. Data to be analyzed by using other chart statements usually contain multiple observations per subgroup.

The DATA= data table must contain the following variables:

- the process name variable, which contains the names of the process variables. You specify it in the PROCESSNAME= option in the PROC SPC statement.
- the process value variable, which contains the process measurements. You specify it in the PROCESSVALUE= option in the PROC SPC statement.
- the subgroup name variables, which contains the names of the subgroup variables. You specify it in the SUBGROUPNAME= option in the PROC SPC statement.
- the subgroup value variable, which contains the subgroup values. You specify it in the SUBGROUPVALUE= option in the PROC SPC statement.
An input data table that contains attribute data (see the section “Classification of Shewhart Charts” on page 933) can also include a variable that contains the number of inspection units per subgroup. You specify this variable in the SUBGROUPN= option in the chart statement. It is required in the NPCHART, PCHART, and UCHART statements and optional in the CCHART statement.

The DATA= data table can also contain BY variables (see the section “BY Statement” on page 939).

**LIMITS= Data Table**

You can read preestablished control limits or parameters from which the control limits can be calculated from a LIMITS= data table that you specify by using the LIMITS= option in the PROC SPC statement. The LIMITS= data table can be an OUTLIMITS= data table that was created in a previous run of the SPC procedure. Such data tables always contain the variables that a LIMITS= data table requires. For more information about the variables that are included in an OUTLIMITS= data table, see the section “OUTLIMITS= Data Table” on page 982.

The LIMITS= data table must contain the following variables:

- the process name variable, which contains the names of the process variables. You specify it in the PROCESSNAME= option in the PROC SPC statement.
- the subgroup name variable, which contains the names of the subgroup variables. You specify it in the SUBGROUPNAME= option in the PROC SPC statement.

For each control chart type, Table 20.34 lists the variables that are required to specify parameters from which control limits are calculated and the variables that contain preestablished control limits.

<table>
<thead>
<tr>
<th>Chart Type</th>
<th>Parameters</th>
<th>Limits</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td><em>U</em></td>
<td><em>LCLC</em>, <em>C</em>, <em>UCLC</em></td>
</tr>
<tr>
<td>Individual measurements</td>
<td><em>MEAN</em>, <em>STDDEV</em></td>
<td><em>LCLI</em>, <em>MEAN</em>, <em>UCLI</em></td>
</tr>
<tr>
<td>Moving range</td>
<td><em>STDDEV</em></td>
<td><em>LCLR</em>, <em>R</em>, <em>UCLR</em></td>
</tr>
<tr>
<td>Medians</td>
<td><em>MEAN</em>, <em>STDDEV</em></td>
<td><em>LCLM</em>, <em>MEAN</em>, <em>UCLM</em></td>
</tr>
<tr>
<td>np</td>
<td><em>P</em></td>
<td><em>LCLNP</em>, <em>NP</em>, <em>UCLNP</em></td>
</tr>
<tr>
<td>p</td>
<td><em>P</em></td>
<td><em>LCLP</em>, <em>P</em>, <em>UCLP</em></td>
</tr>
<tr>
<td>R</td>
<td><em>STDDEV</em></td>
<td><em>LCLR</em>, <em>R</em>, <em>UCLR</em></td>
</tr>
<tr>
<td>s</td>
<td><em>STDDEV</em></td>
<td><em>LCLS</em>, <em>S</em>, <em>UCLS</em></td>
</tr>
<tr>
<td>u</td>
<td><em>U</em></td>
<td><em>LCLU</em>, <em>U</em>, <em>UCLU</em></td>
</tr>
<tr>
<td>$\bar{X}$</td>
<td><em>MEAN</em>, <em>STDDEV</em></td>
<td><em>LCLX</em>, <em>MEAN</em>, <em>UCLX</em></td>
</tr>
</tbody>
</table>

**Notes:**

1. You can include the variable _LIMITN_, with or without parameters and limits variables, to specify different nominal subgroup sizes for different processes.

2. The special missing value V indicates variables that vary with subgroup sample size. Their values are calculated from the data. Table 20.36 identifies the variables that are assigned the missing value V in an OUTLIMITS= data table when subgroup sample sizes vary.
Chapter 20: The SPC Procedure

3. The variables _SIGMAS_ and _TYPE_ are optional, but they are recommended to maintain a complete set of control limit information. The variable _TYPE_ must be a character variable of length 8; valid values are ‘ESTIMATE’, ‘STANDARD’, ‘STDMU’, and ‘STDSIGMA’.

4. If you specify a BY statement, the LIMITS= data table must also contain the specified BY variables.

**SPECS= Data Table**

You can read process specification limits from which capability indices can be calculated from a SPECS= data table that you specify by using the SPECS= option in a chart statement for variables.

The SPECS= data table must include the process name variable, which, as the name suggests, contains the names of the process variables. You specify it in the PROCESSNAME= option in the PROC SPC statement. Table 20.35 lists the specification limit variables that the data table can include.

<table>
<thead>
<tr>
<th>Table 20.35</th>
<th>SPECS= Data Table Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variable</td>
<td>Description</td>
</tr>
<tr>
<td>LSL</td>
<td>Lower specification limit</td>
</tr>
<tr>
<td>Target</td>
<td>Target value</td>
</tr>
<tr>
<td>USL</td>
<td>Upper specification limit</td>
</tr>
</tbody>
</table>

PROC SPC computes capability indices for each process that is included in the DATA= data table and whose specification limits are provided in the SPECS= data table. For more information about how capability indices are computed, see the section “Capability Indices” on page 979.

**Output Data Tables**

**OUTLIMITS= Data Table**

An OUTLIMITS= data table contains control limits and control limit parameters. It contains one observation for each process. The following variables are always included in an OUTLIMITS= data table:

- the process name variable, which contains the names of the process variables. You specify it in the PROCESSNAME= option in the PROC SPC statement.
- the subgroup name variable, which contains the names of the subgroup variables. You specify it in the SUBGROUPNAME= option in the PROC SPC statement.

In addition, if you specify any BY variables, they are included in the OUTLIMITS= data table.

The other variables that are included in an OUTLIMITS= data table depend on which chart statement produced that data table. Table 20.36 lists the variables that are always included in OUTLIMITS= data tables for the different chart statements.
### Table 20.36  Variables in the OUTLIMITS= Data Table

<table>
<thead>
<tr>
<th>Variable</th>
<th>Varying</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>LCLM</em>, <em>LCLR</em>, <em>LCLS</em>, <em>LCLX</em>, <em>LIMITN</em>, <em>MEAN</em>, <em>R</em>, <em>S</em>, <em>SIGMAS</em>, <em>STDDEV</em>, <em>TYPE</em>, <em>UCLM</em>, <em>UCLR</em>, <em>UCLS</em>, <em>UCLX</em></td>
<td>✓</td>
<td>Lower control limit for subgroup median</td>
</tr>
<tr>
<td><em>LIMITN</em></td>
<td>✓</td>
<td>Nominal sample size associated with the control limits</td>
</tr>
<tr>
<td><em>MEAN</em></td>
<td>Process mean (value of central line on box chart)</td>
<td></td>
</tr>
<tr>
<td><em>R</em></td>
<td>✓</td>
<td>Value of central line on R chart</td>
</tr>
<tr>
<td><em>S</em></td>
<td>✓</td>
<td>Value of central line on s chart</td>
</tr>
<tr>
<td><em>SIGMAS</em></td>
<td>Multiple ( k ) of standard error of ( \bar{X}_i ) or ( M_i )</td>
<td></td>
</tr>
<tr>
<td><em>STDDEV</em></td>
<td>Process standard deviation</td>
<td></td>
</tr>
<tr>
<td><em>TYPE</em></td>
<td>Type (estimate or standard value) of <em>MEAN</em> and <em>STDDEV</em></td>
<td></td>
</tr>
<tr>
<td><em>UCLM</em>, <em>UCLR</em>, <em>UCLS</em>, <em>UCLX</em></td>
<td>✓</td>
<td>Upper control limit for subgroup median</td>
</tr>
</tbody>
</table>

### CCHART Statement

| C, _LCLC_, _LIMITN_, _SIGMAS_, _U_, _UCLC_ | ✓ | Value of central line on c chart \((n_i \bar{u} \text{ or } n_i u_0)\) |
| _LIMITN_ | ✓ | Sample size associated with the control limits |
| _SIGMAS_ | Multiple \( k \) of standard error of \( c_i \) |
| _TYPE_ | Type (estimate or standard value) of _U_ |
| _U_ | Average number of nonconformities per unit \((\bar{u} \text{ or } u_0)\) |
| _UCLC_ | ✓ | Upper control limit for number of nonconformities |

### IRCHART Statement

| _LCLI_, _LCLR_, _LIMITN_, _MEAN_, _R_, _SIGMAS_, _STDDEV_, _TYPE_, _UCLI_, _UCLR_ | Lower control limit for individual measurements |
| _LIMITN_ | Number of consecutive measurements that are used to compute moving ranges |
| _MEAN_ | Process mean |
| _R_ | Value of central line on moving range chart |
| _SIGMAS_ | Multiple \( k \) of standard error of individual measurement or moving range |
| _STDDEV_ | Process standard deviation |
| _TYPE_ | Type (estimate or standard value) of _MEAN_ and _STDDEV_ |
| _UCLI_, _UCLR_ | Upper control limit for individual measurements |
| _UCLR_ | Upper control limit for moving ranges |

### MCHART Statement

<p>| <em>LCLM</em>, <em>LCLR</em>, <em>LCLS</em> | ✓ | Lower control limit for subgroup median |
| <em>LCLS</em> | Lower control limit for subgroup range |
| <em>LCLM</em>, <em>LCLR</em> | Lower control limit for subgroup standard deviation |</p>
<table>
<thead>
<tr>
<th>Variable</th>
<th>Varying</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>LIMITN</em></td>
<td>✓</td>
<td>Sample size associated with the control limits</td>
</tr>
<tr>
<td><em>MEAN</em></td>
<td></td>
<td>Value of central line on median chart ($\bar{M}$, $\tilde{M}$, $\bar{X}$, or $\mu_0$)</td>
</tr>
<tr>
<td><em>R</em></td>
<td>✓</td>
<td>Value of central line on $R$ chart</td>
</tr>
<tr>
<td><em>S</em></td>
<td>✓</td>
<td>Value of central line on $s$ chart</td>
</tr>
<tr>
<td><em>SIGMAS</em></td>
<td></td>
<td>Multiple ($k$) of standard error of $M_i$</td>
</tr>
<tr>
<td><em>STDDEV</em></td>
<td></td>
<td>Process standard deviation</td>
</tr>
<tr>
<td><em>TYPE</em></td>
<td></td>
<td>Type (estimate or standard value) of <em>MEAN</em> and <em>STDDEV</em></td>
</tr>
<tr>
<td><em>UCLM</em></td>
<td>✓</td>
<td>Upper control limit for subgroup median</td>
</tr>
<tr>
<td><em>UCLR</em></td>
<td>✓</td>
<td>Upper control limit for subgroup range</td>
</tr>
<tr>
<td><em>UCLS</em></td>
<td>✓</td>
<td>Upper control limit for subgroup standard deviation</td>
</tr>
</tbody>
</table>

**MRCHART Statement**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Varying</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>LCLM</em></td>
<td>✓</td>
<td>Lower control limit for subgroup median</td>
</tr>
<tr>
<td><em>LCLR</em></td>
<td>✓</td>
<td>Lower control limit for subgroup range</td>
</tr>
<tr>
<td><em>LIMITN</em></td>
<td>✓</td>
<td>Sample size associated with the control limits</td>
</tr>
<tr>
<td><em>MEAN</em></td>
<td></td>
<td>Estimate of process mean ($\bar{M}$, $\tilde{M}$, $\bar{X}$, or $\mu_0$)</td>
</tr>
<tr>
<td><em>R</em></td>
<td>✓</td>
<td>Value of central line on range chart</td>
</tr>
<tr>
<td><em>SIGMAS</em></td>
<td></td>
<td>Multiple ($k$) of standard error of $M_i$ or $R_i$</td>
</tr>
<tr>
<td><em>STDDEV</em></td>
<td></td>
<td>Process standard deviation</td>
</tr>
<tr>
<td><em>TYPE</em></td>
<td></td>
<td>Type (estimate or standard value) of <em>MEAN</em> and <em>STDDEV</em></td>
</tr>
<tr>
<td><em>UCLM</em></td>
<td>✓</td>
<td>Upper control limit for subgroup median</td>
</tr>
<tr>
<td><em>UCLR</em></td>
<td>✓</td>
<td>Upper control limit for subgroup range</td>
</tr>
</tbody>
</table>

**NPCHART Statement**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Varying</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>LCLNP</em></td>
<td>✓</td>
<td>Lower control limit for number of nonconforming items</td>
</tr>
<tr>
<td><em>LIMITN</em></td>
<td>✓</td>
<td>Sample size associated with the control limits</td>
</tr>
<tr>
<td><em>NP</em></td>
<td>✓</td>
<td>Average number of nonconforming items ($n_i \bar{p}$ or $n_i p_0$)</td>
</tr>
<tr>
<td><em>P</em></td>
<td>✓</td>
<td>Average proportion of nonconforming items ($\tilde{p}$ or $p_0$)</td>
</tr>
<tr>
<td><em>SIGMAS</em></td>
<td></td>
<td>Multiple ($k$) of standard error of $X_i$</td>
</tr>
<tr>
<td><em>TYPE</em></td>
<td></td>
<td>Type (standard or estimate) of <em>NP</em></td>
</tr>
<tr>
<td><em>UCLNP</em></td>
<td>✓</td>
<td>Upper control limit for number of nonconforming items</td>
</tr>
</tbody>
</table>

**PCHART Statement**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Varying</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>LCLP</em></td>
<td>✓</td>
<td>Lower control limit for proportion of nonconforming items</td>
</tr>
<tr>
<td><em>LIMITN</em></td>
<td>✓</td>
<td>Nominal sample size associated with the control limits</td>
</tr>
<tr>
<td><em>P</em></td>
<td></td>
<td>Average proportion of nonconforming items ($\tilde{p}$ or $p_0$)</td>
</tr>
<tr>
<td><em>SIGMAS</em></td>
<td></td>
<td>Multiple ($k$) of standard error of $p_i$</td>
</tr>
<tr>
<td><em>TYPE</em></td>
<td></td>
<td>Type (standard or estimate) of <em>P</em></td>
</tr>
<tr>
<td><em>UCLP</em></td>
<td>✓</td>
<td>Upper control limit for proportion of nonconforming items</td>
</tr>
</tbody>
</table>

**RCHART Statement**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Varying</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>LCLR</em></td>
<td>✓</td>
<td>Lower control limit for subgroup range</td>
</tr>
<tr>
<td><em>LCLX</em></td>
<td>✓</td>
<td>Lower control limit for subgroup mean</td>
</tr>
<tr>
<td><em>LIMITN</em></td>
<td>✓</td>
<td>Sample size associated with the control limits</td>
</tr>
</tbody>
</table>
### Table 20.36  continued

<table>
<thead>
<tr>
<th>Variable</th>
<th>Varying</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>MEAN</em></td>
<td></td>
<td>Process mean ((\bar{X}))</td>
</tr>
<tr>
<td><em>R</em></td>
<td>✓</td>
<td>Value of central line on (R) chart</td>
</tr>
<tr>
<td><em>SIGMAS</em></td>
<td></td>
<td>Multiple ((k)) of standard error of (R_i)</td>
</tr>
<tr>
<td><em>STDDEV</em></td>
<td></td>
<td>Process standard deviation</td>
</tr>
<tr>
<td><em>TYPE</em></td>
<td></td>
<td>Type (estimate or standard value) of <em>MEAN</em> and <em>STDDEV</em></td>
</tr>
<tr>
<td><em>UCLR</em></td>
<td>✓</td>
<td>Upper control limit for subgroup range</td>
</tr>
<tr>
<td><em>UCLX</em></td>
<td>✓</td>
<td>Upper control limit for subgroup mean</td>
</tr>
</tbody>
</table>

**SCHART Statement**

| _LCLS_   | ✓       | Lower control limit for subgroup standard deviation |
| _LCLX_   | ✓       | Lower control limit for subgroup mean |
| _LIMITN_ | ✓       | Sample size associated with the control limits |
| _MEAN_   |         | Process mean (\(\bar{X}\) or \(\mu_0\)) |
| _S_      | ✓       | Value of central line on \(s\) chart |
| _SIGMAS_ |         | Multiple \((k)\) of standard error of \(\bar{X}_i\) or \(s_i\) |
| _STDDEV_ |         | Process standard deviation |
| _TYPE_   |         | Type (estimate or standard value) of _MEAN_ and _STDDEV_ |
| _UCLS_   | ✓       | Upper control limit for subgroup standard deviation |
| _UCLX_   | ✓       | Upper control limit for subgroup mean |

**UCHART Statement**

| _CLU_     | ✓       | Lower control limit for number of nonconformities per unit |
| _LIMITN_ | ✓       | Sample size associated with the control limits |
| _SIGMAS_ |         | Multiple \((k)\) of standard error of \(u_i\) |
| _TYPE_   |         | Type (estimate or standard value) of _U_ |
| _U_      |         | Value of central line of \(u\) chart (\(\bar{u}\) or \(u_0\)) |
| _UCLU_   | ✓       | Upper control limit for number of nonconformities per unit |

**XCHART Statement**

| _LCLR_   | ✓       | Lower control limit for subgroup range |
| _LCLS_   | ✓       | Lower control limit for subgroup standard deviation |
| _LCLX_   | ✓       | Lower control limit for subgroup mean |
| _LIMITN_ | ✓       | Sample size associated with the control limits |
| _MEAN_   |         | Process mean (\(\bar{X}\) or \(\mu_0\)) |
| _R_      | ✓       | Value of central line on \(R\) chart |
| _S_      | ✓       | Value of central line on \(s\) chart |
| _SIGMAS_ |         | Multiple \((k)\) of standard error of \(\bar{X}_i\) |
| _STDDEV_ |         | Process standard deviation |
| _TYPE_   |         | Type (estimate or standard value) of _MEAN_ and _STDDEV_ |
| _UCLR_   | ✓       | Upper control limit for subgroup range |
| _UCLS_   | ✓       | Upper control limit for subgroup standard deviation |
| _UCLX_   | ✓       | Upper control limit for subgroup mean |
### Table 20.36 continued

<table>
<thead>
<tr>
<th>Variable</th>
<th>Varying</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XRCHART Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td><em>LCLR</em></td>
<td>✓</td>
<td>Lower control limit for subgroup range</td>
</tr>
<tr>
<td><em>LCLX</em></td>
<td>✓</td>
<td>Lower control limit for subgroup mean</td>
</tr>
<tr>
<td><em>LIMITN</em></td>
<td>✓</td>
<td>Nominal sample size associated with the control limits</td>
</tr>
<tr>
<td><em>MEAN</em></td>
<td></td>
<td>Process mean ($\bar{X}$ or $\mu_0$)</td>
</tr>
<tr>
<td><em>R</em></td>
<td>✓</td>
<td>Value of central line on $R$ chart</td>
</tr>
<tr>
<td><em>SIGMAS</em></td>
<td></td>
<td>Multiple ($k$) of standard error of $\bar{X}_i$ or $R_i$</td>
</tr>
<tr>
<td><em>STDDEV</em></td>
<td></td>
<td>Process standard deviation</td>
</tr>
<tr>
<td><em>TYPE</em></td>
<td>✓</td>
<td>Type (estimate or standard value) of <em>MEAN</em> and <em>STDDEV</em></td>
</tr>
<tr>
<td><em>UCLR</em></td>
<td>✓</td>
<td>Upper control limit for subgroup range</td>
</tr>
<tr>
<td><em>UCLX</em></td>
<td>✓</td>
<td>Upper control limit for subgroup mean</td>
</tr>
</tbody>
</table>

| XSCHART Statement | | |
| _LCLS_ | ✓ | Lower control limit for subgroup standard deviation |
| _LCLX_ | ✓ | Lower control limit for subgroup mean |
| _LIMITN_ | ✓ | Nominal sample size associated with the control limits |
| _MEAN_ | | Process mean ($\bar{X}$ or $\mu_0$) |
| _S_ | ✓ | Value of central line on $s$ chart |
| _SIGMAS_ | | Multiple ($k$) of standard error of $\bar{X}_i$ or $s_i$ |
| _STDDEV_ | | Process standard deviation |
| _TYPE_ | ✓ | Type (estimate or standard value) of _MEAN_ and _STDDEV_ |
| _UCLS_ | ✓ | Upper control limit for subgroup standard deviation |
| _UCLX_ | ✓ | Upper control limit for subgroup mean |

**Notes:**

1. If the control limits vary with subgroup sample size, the special missing value V is assigned to the variables that are indicated by a check mark (✓) in the Varying column of Table 20.36.

2. For all charts for variables, the OUTLIMITS= data table includes control limits for means or medians and for ranges or standard deviations. This enables you to use the OUTLIMITS= data table as a LIMITS= data table for different charts for variables.

3. For the BOXCHART statement, the variables _LCLM_ and _UCLM_ are included if you specify CONTROLSTAT=MEDIAN; otherwise, the variables _LCLX_ and _UCLX_ are included. The variables _LCLR_ , _R_ , and _UCLR_ are included if you specify SMETHOD=RMVLUE or RNOWEIGHT; otherwise, the variables _LCLS_ , _S_ , and _UCLS_ are included.

4. For the MCHART and XCHART statements, the variables _LCLS_ , _S_ , and _UCLS_ are included if you specify SMETHOD=RMSDF, SMVLUE, or SNOWEIGHT; otherwise, the variables _LCLR_ , _R_ , and _UCLR_ are included.

If you specify the SPECS= option, the OUTLIMITS= data table also includes the variables that are listed in Table 20.37.
Table 20.37  Additional OUTLIMITS= Data Table Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>LSL</em></td>
<td>Lower specification limit</td>
</tr>
<tr>
<td><em>TARGET</em></td>
<td>Target value</td>
</tr>
<tr>
<td><em>USL</em></td>
<td>Upper specification limit</td>
</tr>
<tr>
<td><em>CP</em></td>
<td>Process capability index $C_p$</td>
</tr>
<tr>
<td><em>CPL</em></td>
<td>Process capability index CPL</td>
</tr>
<tr>
<td><em>CPU</em></td>
<td>Process capability index CPU</td>
</tr>
<tr>
<td><em>CPK</em></td>
<td>Process capability index $C_{pk}$</td>
</tr>
<tr>
<td><em>CPM</em></td>
<td>Process capability index $C_{pm}$</td>
</tr>
</tbody>
</table>

**NOTE:** Capability indices are computed only for processes that are in statistical control.

If you specify both the SPECS= and CIINDICES options, the OUTLIMITS= data table includes the additional variables that are listed in Table 20.38.

Table 20.38  Additional OUTLIMITS= Data Table Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>CP_LCL</em></td>
<td>Lower confidence limit for $C_p$</td>
</tr>
<tr>
<td><em>CP_UCL</em></td>
<td>Upper confidence limit for $C_p$</td>
</tr>
<tr>
<td><em>CPL_LCL</em></td>
<td>Lower confidence limit for CPL</td>
</tr>
<tr>
<td><em>CPL_UCL</em></td>
<td>Upper confidence limit for CPL</td>
</tr>
<tr>
<td><em>CPU_LCL</em></td>
<td>Lower confidence limit for CPU</td>
</tr>
<tr>
<td><em>CPU_UCL</em></td>
<td>Upper confidence limit for CPU</td>
</tr>
<tr>
<td><em>CPK_LCL</em></td>
<td>Lower confidence limit for $C_{pk}$</td>
</tr>
<tr>
<td><em>CPK_UCL</em></td>
<td>Upper confidence limit for $C_{pk}$</td>
</tr>
<tr>
<td><em>CPM_LCL</em></td>
<td>Lower confidence limit for $C_{pm}$</td>
</tr>
<tr>
<td><em>CPM_UCL</em></td>
<td>Upper confidence limit for $C_{pm}$</td>
</tr>
</tbody>
</table>

**NOTE:** Confidence limits are computed for $C_{pm}$ only when both an LSL and a USL are specified and when the target value is halfway between the LSL and the USL.

**OUTTABLE= Data Table**

An OUTTABLE= data table contains subgroup summary statistics, control limits, and related information. The following variables are always included in an OUTTABLE= data table:

- _VAR_, which contains the process variable name
- the subgroup name variable, which contains subgroup variable names. You specify the name of the subgroup name variable in the SUBGROUPNAME= option in the PROC statement.
- the subgroup variable, which contains subgroup variable values. You specify the name of the subgroup variable in the SUBGROUPVALUE= option in the PROC statement.
In addition, if you specify any BY variables, they are included in the OUTTABLE= data table.

The other variables that are included in an OUTTABLE= data table depend on which chart statement produced that data table. Table 20.39 lists the variables that are included in OUTTABLE= data tables for the different chart statements.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOXCHART Statement</td>
<td></td>
</tr>
<tr>
<td><em>EXLIM</em></td>
<td>Control limit exceeded on box chart</td>
</tr>
<tr>
<td><em>LCLM</em></td>
<td>Lower control limit for median</td>
</tr>
<tr>
<td><em>LCLX</em></td>
<td>Lower control limit for mean</td>
</tr>
<tr>
<td><em>LIMITN</em></td>
<td>Nominal sample size associated with the control limits</td>
</tr>
<tr>
<td><em>MEAN</em></td>
<td>Process mean</td>
</tr>
<tr>
<td><em>SIGMAS</em></td>
<td>Multiple (k) of the standard error associated with the control limits</td>
</tr>
<tr>
<td><em>STDDEV</em></td>
<td>Process standard deviation (( \hat{\sigma} ) or ( \sigma_0 ))</td>
</tr>
<tr>
<td><em>SUBMAX</em></td>
<td>Subgroup maximum</td>
</tr>
<tr>
<td><em>SUBMED</em></td>
<td>Subgroup median</td>
</tr>
<tr>
<td><em>SUBMIN</em></td>
<td>Subgroup minimum</td>
</tr>
<tr>
<td><em>SUBN</em></td>
<td>Subgroup sample size</td>
</tr>
<tr>
<td><em>SUBQ1</em></td>
<td>Subgroup first quartile (25th percentile)</td>
</tr>
<tr>
<td><em>SUBQ3</em></td>
<td>Subgroup third quartile (75th percentile)</td>
</tr>
<tr>
<td><em>SUBX</em></td>
<td>Subgroup mean</td>
</tr>
<tr>
<td><em>TESTS</em></td>
<td>Tests for special causes signaled on box chart</td>
</tr>
<tr>
<td><em>UCLM</em></td>
<td>Upper control limit for median</td>
</tr>
<tr>
<td><em>UCLX</em></td>
<td>Upper control limit for mean</td>
</tr>
</tbody>
</table>

| CCHART Statement |                                                                                |
| _C_             | Average number of nonconformities                                             |
| _EXLIM_         | Control limit exceeded on c chart                                            |
| _LCLC_          | Lower control limit for number of nonconformities                            |
| _LIMITN_        | Nominal sample size associated with the control limits                        |
| _SIGMAS_        | Multiple (k) of the standard error associated with the control limits         |
| _SUBC_          | Subgroup number of nonconformities                                           |
| _SUBN_          | Subgroup sample size                                                         |
| _TESTS_         | Tests for special causes signaled on c chart                                 |
| _UCLC_          | Upper control limit for number of nonconformities                            |

| IRCHART Statement |                                                                                |
| _EXLIM_          | Control limit exceeded on individual measurements chart                       |
| _EXLIMR_         | Control limit exceeded on moving range chart                                  |
| _LCLI_           | Lower control limit for individual measurements                              |
| _LCLR_           | Lower control limit for moving range                                          |
| _LIMITN_         | Number of consecutive measurements that are used to compute moving ranges    |
| _MEAN_           | Process mean                                                                 |
| _R_              | Average range                                                                |
### Table 20.39  continued

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>SIGMAS</em></td>
<td>Multiple ((k)) of the standard error associated with the control limits</td>
</tr>
<tr>
<td><em>STDDEV</em></td>
<td>Process standard deviation ((\hat{\sigma} \text{ or } \sigma_0))</td>
</tr>
<tr>
<td><em>SUBI</em></td>
<td>Individual measurement</td>
</tr>
<tr>
<td><em>SUBR</em></td>
<td>Moving range</td>
</tr>
<tr>
<td><em>TESTS</em></td>
<td>Tests for special causes signaled on individual measurements chart</td>
</tr>
<tr>
<td><em>UCLI</em></td>
<td>Upper control limit for individual measurements</td>
</tr>
<tr>
<td><em>UCLR</em></td>
<td>Upper control limit for moving range</td>
</tr>
</tbody>
</table>

#### MCHART Statement

| _EXLIM_  | Control limit exceeded on median chart |
| _LCLM_   | Lower control limit for median |
| _LIMITN_ | Nominal sample size associated with the control limits |
| _MEAN_   | Estimate of process mean \((\bar{M}, \hat{M}, \bar{X}, \text{ or } \mu_0)\) |
| _SIGMAS_ | Multiple \((k)\) of the standard error associated with the control limits |
| _STDDEV_ | Process standard deviation \((\hat{\sigma} \text{ or } \sigma_0)\) |
| _SUBMED_ | Subgroup median |
| _SUBN_   | Subgroup sample size |
| _TESTS_  | Tests for special causes signaled on median chart |
| _UCLM_   | Upper control limit for median |

#### MRCHART Statement

| _EXLIM_  | Control limit exceeded on median chart |
| _EXLIMR_ | Control limit exceeded on range chart |
| _LCLM_   | Lower control limit for median |
| _LCLR_   | Lower control limit for range |
| _LIMITN_ | Nominal sample size associated with the control limits |
| _MEAN_   | Average range |
| _R_      | Estimate of process mean \((\bar{M}, \hat{M}, \bar{X}, \text{ or } \mu_0)\) |
| _SIGMAS_ | Multiple \((k)\) of the standard error associated with the control limits |
| _STDDEV_ | Process standard deviation \((\hat{\sigma} \text{ or } \sigma_0)\) |
| _SUBM_   | Subgroup median |
| _SUBN_   | Subgroup sample size |
| _SUBR_   | Subgroup range |
| _TESTS_  | Tests for special causes signaled on median chart |
| _TESTS2_ | Tests for special causes signaled on range chart |
| _UCLM_   | Upper control limit for mean |
| _UCLR_   | Upper control limit for range |

#### NPCHART Statement

<p>| <em>EXLIM</em>  | Control limit exceeded on (np) chart |
| <em>LCLNP</em>  | Lower control limit for number of nonconforming items |
| <em>LIMITN</em> | Nominal sample size associated with the control limits |
| <em>NP</em>     | Average number of nonconforming items |</p>
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>SIGMAS</em></td>
<td>Multiple ((k)) of the standard error of (X_i) associated with the control limits</td>
</tr>
<tr>
<td><em>SUBNP</em></td>
<td>Subgroup number of nonconforming items</td>
</tr>
<tr>
<td><em>SUBN</em></td>
<td>Subgroup sample size</td>
</tr>
<tr>
<td><em>TESTS</em></td>
<td>Tests for special causes signaled on (np) chart</td>
</tr>
<tr>
<td><em>UCLNP</em></td>
<td>Upper control limit for number of nonconforming items</td>
</tr>
</tbody>
</table>

**PCHART Statement**

| _EXLIM_   | Control limit exceeded on \(p\) chart                                       |
| _LCLP_    | Lower control limit for proportion of nonconforming items                   |
| _LIMITN_  | Nominal sample size associated with the control limits                       |
| _P_       | Average proportion of nonconforming items                                   |
| _SIGMAS_  | Multiple \((k)\) of the standard error of \(p_i\) associated with the control limits |
| _SUBP_    | Subgroup proportion of nonconforming items                                  |
| _SUBN_    | Subgroup sample size                                                         |
| _TESTS_   | Tests for special causes signaled on \(p\) chart                           |
| _UCLP_    | Upper control limit for proportion of nonconforming items                   |

**RCHART Statement**

| _EXLIM_   | Control limit exceeded on \(R\) chart                                        |
| _LCLR_    | Lower control limit for range                                                 |
| _LIMITN_  | Nominal sample size associated with the control limits                        |
| _R_       | Average range                                                                 |
| _SIGMAS_  | Multiple \((k)\) of the standard error associated with the control limits     |
| _STDDEV_  | Process standard deviation \((\hat{\sigma} \text{ or } \sigma_0)\)                |
| _SUBN_    | Subgroup sample size                                                          |
| _SUBR_    | Subgroup range                                                                |
| _TESTS2_  | Tests for special causes signaled on \(R\) chart                             |
| _UCLR_    | Upper control limit for range                                                 |

**SCHART Statement**

<p>| <em>EXLIM</em>   | Control limit exceeded on (s) chart                                        |
| <em>LCLS</em>    | Lower control limit for standard deviation                                    |
| <em>LIMITN</em>  | Nominal sample size associated with the control limits                        |
| <em>S</em>       | Average standard deviation                                                    |
| <em>SIGMAS</em>  | Multiple ((k)) of the standard error associated with the control limits     |
| <em>STDDEV</em>  | Process standard deviation ((\hat{\sigma} \text{ or } \sigma_0))                |
| <em>STDDEV</em>  | Process standard deviation ((\hat{\sigma} \text{ or } \sigma_0))                |
| <em>SUBN</em>    | Subgroup sample size                                                          |
| <em>SUBS</em>    | Subgroup standard deviation                                                   |
| <em>TESTS2</em>  | Tests for special causes signaled on (s) chart                             |
| <em>UCLS</em>    | Upper control limit for standard deviation                                    |</p>
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>UCHART Statement</strong></td>
<td></td>
</tr>
<tr>
<td><em>EXLIM</em></td>
<td>Control limit exceeded on ( u ) chart</td>
</tr>
<tr>
<td><em>LCLU</em></td>
<td>Lower control limit for number of nonconformities per unit</td>
</tr>
<tr>
<td><em>LIMITN</em></td>
<td>Nominal sample size associated with the control limits</td>
</tr>
<tr>
<td><em>SIGMAS</em></td>
<td>Multiple ( (k) ) of the standard error associated with the control limits</td>
</tr>
<tr>
<td><em>SUBU</em></td>
<td>Subgroup number of nonconformities per unit</td>
</tr>
<tr>
<td><em>SUBN</em></td>
<td>Subgroup sample size</td>
</tr>
<tr>
<td><em>TESTS</em></td>
<td>Tests for special causes signaled on ( u ) chart</td>
</tr>
<tr>
<td><em>U</em></td>
<td>Average number of nonconformities per unit</td>
</tr>
<tr>
<td><em>UCLU</em></td>
<td>Upper control limit for number of nonconformities per unit</td>
</tr>
<tr>
<td><strong>XCHART Statement</strong></td>
<td></td>
</tr>
<tr>
<td><em>EXLIM</em></td>
<td>Control limit exceeded on ( \bar{X} ) chart</td>
</tr>
<tr>
<td><em>LCLX</em></td>
<td>Lower control limit for mean</td>
</tr>
<tr>
<td><em>LIMITN</em></td>
<td>Nominal sample size associated with the control limits</td>
</tr>
<tr>
<td><em>MEAN</em></td>
<td>Process mean</td>
</tr>
<tr>
<td><em>SIGMAS</em></td>
<td>Multiple ( (k) ) of the standard error associated with the control limits</td>
</tr>
<tr>
<td><em>STDDEV</em></td>
<td>Process standard deviation ( (\hat{\sigma} \text{ or } \sigma_0) )</td>
</tr>
<tr>
<td><em>SUBN</em></td>
<td>Subgroup sample size</td>
</tr>
<tr>
<td><em>SUBX</em></td>
<td>Subgroup mean</td>
</tr>
<tr>
<td><em>TESTS</em></td>
<td>Tests for special causes signaled on ( \bar{X} ) chart</td>
</tr>
<tr>
<td><em>UCLX</em></td>
<td>Upper control limit for mean</td>
</tr>
<tr>
<td><strong>XRCHART Statement</strong></td>
<td></td>
</tr>
<tr>
<td><em>EXLIM</em></td>
<td>Control limit exceeded on ( \bar{X} ) chart</td>
</tr>
<tr>
<td><em>EXLIMR</em></td>
<td>Control limit exceeded on ( R ) chart</td>
</tr>
<tr>
<td><em>LCLR</em></td>
<td>Lower control limit for range</td>
</tr>
<tr>
<td><em>LCLX</em></td>
<td>Lower control limit for mean</td>
</tr>
<tr>
<td><em>LIMITN</em></td>
<td>Nominal sample size associated with the control limits</td>
</tr>
<tr>
<td><em>MEAN</em></td>
<td>Process mean</td>
</tr>
<tr>
<td><em>R</em></td>
<td>Average range</td>
</tr>
<tr>
<td><em>SIGMAS</em></td>
<td>Multiple ( (k) ) of the standard error associated with the control limits</td>
</tr>
<tr>
<td><em>STDDEV</em></td>
<td>Process standard deviation ( (\hat{\sigma} \text{ or } \sigma_0) )</td>
</tr>
<tr>
<td><em>SUBN</em></td>
<td>Subgroup sample size</td>
</tr>
<tr>
<td><em>SUBR</em></td>
<td>Subgroup range</td>
</tr>
<tr>
<td><em>SUBX</em></td>
<td>Subgroup mean</td>
</tr>
<tr>
<td><em>TESTS</em></td>
<td>Tests for special causes signaled on ( \bar{X} ) chart</td>
</tr>
<tr>
<td><em>TESTS2</em></td>
<td>Tests for special causes signaled on ( R ) chart</td>
</tr>
<tr>
<td><em>UCLR</em></td>
<td>Upper control limit for range</td>
</tr>
<tr>
<td><em>UCLX</em></td>
<td>Upper control limit for mean</td>
</tr>
<tr>
<td><strong>XSCHART Statement</strong></td>
<td></td>
</tr>
<tr>
<td><em>EXLIM</em></td>
<td>Control limit exceeded on ( \bar{X} ) chart</td>
</tr>
<tr>
<td><em>EXLIMS</em></td>
<td>Control limit exceeded on ( s ) chart</td>
</tr>
</tbody>
</table>
Table 20.39  continued

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>LCLS</em></td>
<td>Lower control limit for standard deviation</td>
</tr>
<tr>
<td><em>LCLX</em></td>
<td>Lower control limit for mean</td>
</tr>
<tr>
<td><em>LIMITN</em></td>
<td>Nominal sample size associated with the control limits</td>
</tr>
<tr>
<td><em>MEAN</em></td>
<td>Process mean</td>
</tr>
<tr>
<td><em>S</em></td>
<td>Average standard deviation</td>
</tr>
<tr>
<td><em>SIGMAS</em></td>
<td>Multiple (k) of the standard error associated with the control limits</td>
</tr>
<tr>
<td><em>STDDEV</em></td>
<td>Process standard deviation ((\hat{\sigma} \text{ or } \sigma_0))</td>
</tr>
<tr>
<td><em>SUBN</em></td>
<td>Subgroup sample size</td>
</tr>
<tr>
<td><em>SUBS</em></td>
<td>Subgroup standard deviation</td>
</tr>
<tr>
<td><em>SUBX</em></td>
<td>Subgroup mean</td>
</tr>
<tr>
<td><em>TESTS</em></td>
<td>Tests for special causes signaled on (\bar{X}) chart</td>
</tr>
<tr>
<td><em>TESTS2</em></td>
<td>Tests for special causes signaled on (s) chart</td>
</tr>
<tr>
<td><em>UCLS</em></td>
<td>Upper control limit for standard deviation</td>
</tr>
<tr>
<td><em>UCLX</em></td>
<td>Upper control limit for mean</td>
</tr>
</tbody>
</table>

Notes:

1. The variable _TESTS_ is saved if you specify the TESTS= option. The \(k\)th character of a value of _TESTS_ is \(k\) if Test \(k\) is positive at that subgroup. For example, if you request all eight tests and Tests 2 and 8 are positive for a given subgroup, the value of _TESTS_ has a 2 for the second character, an 8 for the eighth character, and blanks for the other six characters.

2. The variable _TESTS2_ is saved if you specify the TESTS2= option.

3. The variables _EXLIM_, _EXLIMR_, _EXLIMS_, _TESTS_, and _TESTS2_ are character variables of length 8. All other variables are numeric.

ODS Table Names

Each table that the SPC procedure creates has a name associated with it. You must use this name to refer to the table when you use ODS statements. The name of this table and a short description of its contents are listed in Table 20.40.

Table 20.40  ODS Table Produced by PROC SPC

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Statements</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>BoxchartExceptionSummary</td>
<td>Summary of exceptions</td>
<td>BOXCHART</td>
<td>Default output</td>
</tr>
<tr>
<td>CapabilityIndices</td>
<td>Specification limits and capability indices</td>
<td>BOXCHART, IRCHART, MCHART, MRCHART, RCHART, SCHART, XCHART, XRCHART, XSCHART</td>
<td>SPECS=</td>
</tr>
</tbody>
</table>
Table 20.40  continued

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>CChartExceptionSummary</td>
<td>Summary of exceptions</td>
<td>CCHART</td>
<td>Default output</td>
</tr>
<tr>
<td>IRChartExceptionSummary</td>
<td>Summary of exceptions</td>
<td>IRCHART</td>
<td>Default output</td>
</tr>
<tr>
<td>MChartExceptionSummary</td>
<td>Summary of exceptions</td>
<td>MCHART</td>
<td>Default output</td>
</tr>
<tr>
<td>MRChartExceptionSummary</td>
<td>Summary of exceptions</td>
<td>MRCHART</td>
<td>Default output</td>
</tr>
<tr>
<td>NPChartExceptionSummary</td>
<td>Summary of exceptions</td>
<td>NPCHART</td>
<td>Default output</td>
</tr>
<tr>
<td>PChartExceptionSummary</td>
<td>Summary of exceptions</td>
<td>PCHART</td>
<td>Default output</td>
</tr>
<tr>
<td>RChartExceptionSummary</td>
<td>Summary of exceptions</td>
<td>RCHART</td>
<td>Default output</td>
</tr>
<tr>
<td>SChartExceptionSummary</td>
<td>Summary of exceptions</td>
<td>SCHART</td>
<td>Default output</td>
</tr>
<tr>
<td>UChartExceptionSummary</td>
<td>Summary of exceptions</td>
<td>UCHART</td>
<td>Default output</td>
</tr>
<tr>
<td>XChartExceptionSummary</td>
<td>Summary of exceptions</td>
<td>XCHART</td>
<td>Default output</td>
</tr>
<tr>
<td>XRChartExceptionSummary</td>
<td>Summary of exceptions</td>
<td>XRCHART</td>
<td>Default output</td>
</tr>
<tr>
<td>XSChartExceptionSummary</td>
<td>Summary of exceptions</td>
<td>XSCHART</td>
<td>Default output</td>
</tr>
</tbody>
</table>

Examples: SPC Procedure

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

Example 20.1: Saving Results in Output Data Tables

The following statements perform $\bar{X}$ and $R$ chart analysis on the data table mycas.AllProcesses that is created in the section “Getting Started: SPC Procedure” on page 935 and save the results in OUTTABLE= and OUTLIMITS= data tables:

```sas
proc spc data=mycas.AllProcesses;
   xrchart / noprint
       outtable = mycas.AllTable
       outlimits = mycas.AllLimits;
run;
```

The NOPRINT option suppresses the display of the exception summary table. Output 20.1.1 shows a partial listing of the mycas.AllTable data table.
Output 20.1.1 Partial Listing of the mycas.AllTable Data Table

<table>
<thead>
<tr>
<th><em>VAR</em></th>
<th>subgroupname</th>
<th>subgroup</th>
<th><em>SIGMAS</em></th>
<th><em>LIMITN</em></th>
<th><em>SUBN</em></th>
<th><em>LCLX</em></th>
<th><em>SUBX</em></th>
<th><em>MEAN</em></th>
<th><em>UCLX</em></th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>Lot</td>
<td>1</td>
<td>3</td>
<td>6</td>
<td>6</td>
<td>7.94314</td>
<td>8.00833</td>
<td>8.00307</td>
<td>8.06299</td>
</tr>
<tr>
<td>Time</td>
<td>Lot</td>
<td>2</td>
<td>3</td>
<td>6</td>
<td>6</td>
<td>7.94314</td>
<td>8.02167</td>
<td>8.00307</td>
<td>8.06299</td>
</tr>
<tr>
<td>Time</td>
<td>Lot</td>
<td>3</td>
<td>3</td>
<td>6</td>
<td>6</td>
<td>7.94314</td>
<td>7.97833</td>
<td>8.00307</td>
<td>8.06299</td>
</tr>
<tr>
<td>Time</td>
<td>Lot</td>
<td>4</td>
<td>3</td>
<td>6</td>
<td>6</td>
<td>7.94314</td>
<td>8.00667</td>
<td>8.00307</td>
<td>8.06299</td>
</tr>
<tr>
<td>Time</td>
<td>Lot</td>
<td>5</td>
<td>3</td>
<td>6</td>
<td>6</td>
<td>7.94314</td>
<td>8.01833</td>
<td>8.00307</td>
<td>8.06299</td>
</tr>
<tr>
<td>Time</td>
<td>Lot</td>
<td>6</td>
<td>3</td>
<td>6</td>
<td>6</td>
<td>7.94314</td>
<td>8.00667</td>
<td>8.00307</td>
<td>8.06299</td>
</tr>
<tr>
<td>Time</td>
<td>Lot</td>
<td>7</td>
<td>3</td>
<td>6</td>
<td>6</td>
<td>7.94314</td>
<td>7.98500</td>
<td>8.00307</td>
<td>8.06299</td>
</tr>
<tr>
<td>Time</td>
<td>Lot</td>
<td>8</td>
<td>3</td>
<td>6</td>
<td>6</td>
<td>7.94314</td>
<td>8.03000</td>
<td>8.00307</td>
<td>8.06299</td>
</tr>
<tr>
<td>Time</td>
<td>Lot</td>
<td>9</td>
<td>3</td>
<td>6</td>
<td>6</td>
<td>7.94314</td>
<td>8.03000</td>
<td>8.00307</td>
<td>8.06299</td>
</tr>
<tr>
<td>Time</td>
<td>Lot</td>
<td>10</td>
<td>3</td>
<td>6</td>
<td>6</td>
<td>7.94314</td>
<td>7.99167</td>
<td>8.00307</td>
<td>8.06299</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><em>STDDEV</em></th>
<th><em>EXLIM</em></th>
<th><em>LCLR</em></th>
<th><em>SUBR</em></th>
<th><em>R</em></th>
<th><em>UCLR</em></th>
<th><em>EXLIMR</em></th>
</tr>
</thead>
<tbody>
<tr>
<td>0.048927</td>
<td>0</td>
<td>0.16</td>
<td>0.124</td>
<td>0.24847</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.048927</td>
<td>0</td>
<td>0.09</td>
<td>0.124</td>
<td>0.24847</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.048927</td>
<td>0</td>
<td>0.07</td>
<td>0.124</td>
<td>0.24847</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.048927</td>
<td>0</td>
<td>0.10</td>
<td>0.124</td>
<td>0.24847</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.048927</td>
<td>0</td>
<td>0.16</td>
<td>0.124</td>
<td>0.24847</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.048927</td>
<td>0</td>
<td>0.12</td>
<td>0.124</td>
<td>0.24847</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.048927</td>
<td>0</td>
<td>0.15</td>
<td>0.124</td>
<td>0.24847</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.048927</td>
<td>0</td>
<td>0.11</td>
<td>0.124</td>
<td>0.24847</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.048927</td>
<td>0</td>
<td>0.15</td>
<td>0.124</td>
<td>0.24847</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.048927</td>
<td>0</td>
<td>0.10</td>
<td>0.124</td>
<td>0.24847</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Output 20.1.2 displays the mycas.AllLimits data table.

Output 20.1.2 mycas.AllLimits Data Table

<table>
<thead>
<tr>
<th>processname</th>
<th>subgroupname</th>
<th><em>SIGMAS</em></th>
<th><em>LIMITN</em></th>
<th><em>LCLX</em></th>
<th><em>MEAN</em></th>
<th><em>UCLX</em></th>
<th><em>STDDEV</em></th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>Lot</td>
<td>3</td>
<td>6</td>
<td>7.94</td>
<td>8.00</td>
<td>8.06</td>
<td>0.049</td>
</tr>
<tr>
<td>Diameter</td>
<td>Batch</td>
<td>3</td>
<td>5</td>
<td>34.98</td>
<td>34.99</td>
<td>35.01</td>
<td>0.010</td>
</tr>
<tr>
<td>Partgap</td>
<td>Sample</td>
<td>3</td>
<td>5</td>
<td>241.94</td>
<td>260.00</td>
<td>278.05</td>
<td>13.454</td>
</tr>
<tr>
<td>Breakstrength</td>
<td>Sample</td>
<td>3</td>
<td>V</td>
<td>V</td>
<td>59.98</td>
<td>V</td>
<td>2.111</td>
</tr>
<tr>
<td>Amount</td>
<td>Batch</td>
<td>3</td>
<td>V</td>
<td>V</td>
<td>59.98</td>
<td>V</td>
<td>2.111</td>
</tr>
<tr>
<td>Delay</td>
<td>Day</td>
<td>3</td>
<td>V</td>
<td>V</td>
<td>9.53</td>
<td>V</td>
<td>8.512</td>
</tr>
<tr>
<td>KWatts</td>
<td>Day</td>
<td>3</td>
<td>20</td>
<td>3362.10</td>
<td>3497.56</td>
<td>3633.02</td>
<td>201.937</td>
</tr>
<tr>
<td>Weight</td>
<td>Lot</td>
<td>3</td>
<td>5</td>
<td>21.49</td>
<td>24.61</td>
<td>27.74</td>
<td>2.330</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><em>LCLR</em></th>
<th><em>R</em></th>
<th><em>UCLR</em></th>
<th><em>TYPE</em></th>
</tr>
</thead>
<tbody>
<tr>
<td>0.000</td>
<td>0.124</td>
<td>0.25</td>
<td>ESTIMATE</td>
</tr>
<tr>
<td>0.000</td>
<td>0.022</td>
<td>0.05</td>
<td>ESTIMATE</td>
</tr>
<tr>
<td>0.000</td>
<td>31.293</td>
<td>66.17</td>
<td>ESTIMATE</td>
</tr>
<tr>
<td>V</td>
<td>V</td>
<td>V</td>
<td>ESTIMATE</td>
</tr>
<tr>
<td>V</td>
<td>V</td>
<td>V</td>
<td>ESTIMATE</td>
</tr>
<tr>
<td>V</td>
<td>V</td>
<td>V</td>
<td>ESTIMATE</td>
</tr>
<tr>
<td>312.779</td>
<td>754.225</td>
<td>1195.67</td>
<td>ESTIMATE</td>
</tr>
<tr>
<td>0.000</td>
<td>5.420</td>
<td>11.46</td>
<td>ESTIMATE</td>
</tr>
</tbody>
</table>
Note that limits variables for three of the processes have the special missing value $V$, because the data for those processes have varying subgroup sample sizes.

The mycas.AllLimits data table has the same format as a LIMITS= data table, so it could provide limits for these processes in a later run of PROC SPC.

Example 20.2: Computing Process Capability Indices

You can use the SPC procedure to compute process capability indices for stable processes. The following statements create a data table that contains process specification limits for the processes in the data table mycas.AllProcesses:

```sas
data mycas.AllSpecs;
    length processname $16;
    input processname LSL Target USL;
    datalines;
    Amount  11.95  12.00  12.05
    Breakstrength  52.0  60.0
    Delay  0 30
    Diameter  34.98  35.00  35.01
    KWatts  3400  3500  4000
    Partgap  250  260  270
    Time  7.85  8.0  8.15
    Weight  23.0  24.0  25.0
;
```

Not all processes have both lower and upper specification limits. For example, the Breakstrength process variable measures the breaking strength of metal wire and has no upper specification limit.

The following statements read the specification limits and compute capability indices for processes that the $\bar{X}$ and $R$ chart analysis indicates are in statistical control:

```sas
proc spc data=mycas.AllProcesses;
    xrchart / exchart specs=mycas.AllSpecs;
run;
```

The EXCHART option includes only processes whose exceptions are signaled in the exception summary table. The resulting output is shown in Output 20.2.1.

Output 20.2.1 Exception Summary and Capability Indices

The SPC Procedure

<table>
<thead>
<tr>
<th>Means and Ranges Chart Summary for ALLPROCESSES</th>
</tr>
</thead>
<tbody>
<tr>
<td>processname</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Delay</td>
</tr>
<tr>
<td>Diameter</td>
</tr>
</tbody>
</table>
Output 20.2.1 continued

<table>
<thead>
<tr>
<th>processname</th>
<th>subgroupname</th>
<th>LSL</th>
<th>Target</th>
<th>USL</th>
<th>Cp</th>
<th>CPL</th>
<th>CPU</th>
<th>Cpk</th>
<th>Cpm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amount</td>
<td>Batch</td>
<td>11.95000</td>
<td>12.00000</td>
<td>12.05000</td>
<td>0.379318</td>
<td>0.228958</td>
<td>0.529679</td>
<td>0.228958</td>
<td>0.345768</td>
</tr>
<tr>
<td>Breakstrength</td>
<td>Sample</td>
<td>52.000000</td>
<td>60.000000</td>
<td></td>
<td></td>
<td>1.259258</td>
<td>1.259258</td>
<td>1.262873</td>
<td></td>
</tr>
<tr>
<td>KWatts</td>
<td>Day</td>
<td>3400.0000</td>
<td>3500.0000</td>
<td>4000.0000</td>
<td>0.495204</td>
<td>0.161038</td>
<td>0.161038</td>
<td>0.165056</td>
<td>0.165056</td>
</tr>
<tr>
<td>Partgap</td>
<td>Sample</td>
<td>250.00000</td>
<td>260.00000</td>
<td>270.00000</td>
<td>0.247761</td>
<td>0.247640</td>
<td>0.247640</td>
<td>0.247761</td>
<td>0.247761</td>
</tr>
<tr>
<td>Time</td>
<td>Lot</td>
<td>7.850000</td>
<td>8.000000</td>
<td>8.150000</td>
<td>1.021941</td>
<td>1.042834</td>
<td>1.001048</td>
<td>1.019939</td>
<td>1.019939</td>
</tr>
<tr>
<td>Weight</td>
<td>Lot</td>
<td>23.000000</td>
<td>24.000000</td>
<td>25.000000</td>
<td>0.143037</td>
<td>0.230790</td>
<td>0.055284</td>
<td>0.055284</td>
<td>0.138324</td>
</tr>
</tbody>
</table>

When you use the EXCHART option in combination with the SPECS= option, each process is listed in either the exception summary table or the capability indices table, but not both.

Example 20.3: Reading Preestablished Control Limits

You can apply the control limits that are saved in the data table mycas.AllLimits to new data for those processes. The exception summary table in the section “Getting Started: SPC Procedure” on page 935 shows unusual variation in the processes Delay and Diameter, whereas the other six processes are stable. The data table mycas.StableProcesses contains additional data for the stable processes. The following statements read the control limits that are saved in the mycas.AllLimits data table and apply them to these new data:

```sas
proc spc data=mycas.StableProcesses limits=mycas.AllLimits;
   xrchart;
run;
```

The resulting exception summary table is shown in Output 20.3.1.

Output 20.3.1 Exception Summary of Additional Process Data

<table>
<thead>
<tr>
<th>processname</th>
<th>subgroupname</th>
<th>Number of Subgroups</th>
<th>Number &gt; UCL</th>
<th>Number &lt; LCL</th>
<th>Number &gt; UCL</th>
<th>Number &lt; LCL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amount</td>
<td>Batch</td>
<td>35</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Breakstrength</td>
<td>Sample</td>
<td>28</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KWatts</td>
<td>Day</td>
<td>55</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Partgap</td>
<td>Sample</td>
<td>36</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time</td>
<td>Lot</td>
<td>19</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Weight</td>
<td>Lot</td>
<td>40</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The analysis of the additional process data shows unusual variation in the Partgap process.
Example 20.4: Applying Tests for Special Causes

The following statements produce a data table for 100 simulated process measurements, called Process001 through Process100:

```sas
data mycas.Random;
   length processname $16 subgroupname $16;
   do i = 1 to 100;
      processname = 'Process' || left( put( i, z3. ) );
      subgroupname = 'Subgroup' || left( put( i, z3. ) );
      do subgroup = 1 to 30;
         do j = 1 to 5;
            process = rannor(123);
            output;
         end;
      end;
   end;
   drop i j;
run;
```

The following statements use PROC SPC to do an $\bar{X}$ chart analysis of the simulated data and apply the first through fourth tests for special causes:

```sas
proc spc data=mycas.Random;
   xchart / exchart
      tests = 1 to 4
      outtable = mycas.RandomTests;
run;
```

The TESTS= option requests the tests for special causes. Output 20.2 shows the resulting exception summary table.
Output 20.4.1  Results of Tests for Special Causes

The SPC Procedure

<table>
<thead>
<tr>
<th>processname</th>
<th>subgroupname</th>
<th>Number of Subgroups</th>
<th>Number &gt; UCL</th>
<th>Number &lt; LCL</th>
<th>Test 1 Signals</th>
<th>Test 2 Signals</th>
<th>Test 3 Signals</th>
<th>Test 4 Signals</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process006</td>
<td>Subgroup006</td>
<td>30</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Process011</td>
<td>Subgroup011</td>
<td>30</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Process015</td>
<td>Subgroup015</td>
<td>30</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Process021</td>
<td>Subgroup021</td>
<td>30</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Process031</td>
<td>Subgroup031</td>
<td>30</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Process032</td>
<td>Subgroup032</td>
<td>30</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Process034</td>
<td>Subgroup034</td>
<td>30</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Process047</td>
<td>Subgroup047</td>
<td>30</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Process048</td>
<td>Subgroup048</td>
<td>30</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Process066</td>
<td>Subgroup066</td>
<td>30</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Process069</td>
<td>Subgroup069</td>
<td>30</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Process074</td>
<td>Subgroup074</td>
<td>30</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Process078</td>
<td>Subgroup078</td>
<td>30</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Process079</td>
<td>Subgroup079</td>
<td>30</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Process080</td>
<td>Subgroup080</td>
<td>30</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Process081</td>
<td>Subgroup081</td>
<td>30</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Process089</td>
<td>Subgroup089</td>
<td>30</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Process092</td>
<td>Subgroup092</td>
<td>30</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Process099</td>
<td>Subgroup099</td>
<td>30</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Nineteen of the 100 processes are flagged for at least one exception: a subgroup mean outside the control limits or a positive test result. Because these data were simulated from a normal distribution, they represent processes that are in statistical control. Therefore, you should regard all these exceptions as false positives.

The combined chance of a false signal when Tests 1–4 are applied together is less than 1 in 100 (see the section “Interpreting Tests for Special Causes” on page 977). Therefore, finding 21 positive tests in an analysis of 3,000 total subgroups is not unexpected.

Example 20.5: Producing Control Charts with PROC SHEWHART

The exception summary table that is displayed in Output 20.5.1 shows that the process variable Process047 had two signals of Test 3, indicating a drift in the process mean. The following statements read the data for Process047 from the data table mycas.RandomTests and run PROC SHEWHART to produce an $\bar{X}$ chart of the Process047 data:

```
proc shewhart table=mycas.RandomTests(where=(VAR eq 'Process047'));
  xchart Process047 * Subgroup /
    tests = 1 to 4
    markers;
run;
```

The $\bar{X}$ chart for Process047 is shown in Output 20.5.1.
The sawtooth pattern of points in Output 20.5.1 indicates an alternating instability in the process. However, it is known that the simulated data are stable, and therefore this must be a random effect. Control chart analysis detects unusual variation in a process measurement, but if you analyze enough data, you will find unusual variation even in a stable process.
References


# Chapter 21
The TREESPLIT Procedure

## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview: TREESPLIT Procedure</td>
<td>1002</td>
</tr>
<tr>
<td>PROC TREESPLIT Features</td>
<td>1002</td>
</tr>
<tr>
<td>Using CAS Sessions and CAS Engine Librefs</td>
<td>1003</td>
</tr>
<tr>
<td>Getting Started: TREESPLIT Procedure</td>
<td>1004</td>
</tr>
<tr>
<td>Syntax: TREESPLIT Procedure</td>
<td>1011</td>
</tr>
<tr>
<td>PROC TREESPLIT Statement</td>
<td>1011</td>
</tr>
<tr>
<td>AUTOTUNE Statement</td>
<td>1018</td>
</tr>
<tr>
<td>CLASS Statement</td>
<td>1025</td>
</tr>
<tr>
<td>CODE Statement</td>
<td>1026</td>
</tr>
<tr>
<td>FREQ Statement</td>
<td>1026</td>
</tr>
<tr>
<td>GROW Statement</td>
<td>1026</td>
</tr>
<tr>
<td>MODEL Statement</td>
<td>1028</td>
</tr>
<tr>
<td>OUTPUT Statement</td>
<td>1028</td>
</tr>
<tr>
<td>PARTITION Statement</td>
<td>1029</td>
</tr>
<tr>
<td>PRUNE Statement</td>
<td>1030</td>
</tr>
<tr>
<td>VIICODE Statement</td>
<td>1032</td>
</tr>
<tr>
<td>WEIGHT Statement</td>
<td>1032</td>
</tr>
<tr>
<td>Details: TREESPLIT Procedure</td>
<td>1033</td>
</tr>
<tr>
<td>Building a Decision Tree</td>
<td>1033</td>
</tr>
<tr>
<td>Splitting Criteria</td>
<td>1036</td>
</tr>
<tr>
<td>Primary and Surrogate Splitting Rules</td>
<td>1038</td>
</tr>
<tr>
<td>Pruning</td>
<td>1039</td>
</tr>
<tr>
<td>Scoring</td>
<td>1044</td>
</tr>
<tr>
<td>Measures of Model Fit</td>
<td>1045</td>
</tr>
<tr>
<td>Variable Importance</td>
<td>1045</td>
</tr>
<tr>
<td>Hyperparameter Tuning</td>
<td>1049</td>
</tr>
<tr>
<td>Displayed Output</td>
<td>1052</td>
</tr>
<tr>
<td>ODS Table Names</td>
<td>1054</td>
</tr>
<tr>
<td>ODS Graphics</td>
<td>1055</td>
</tr>
<tr>
<td>INPUT and TARGET Statement Syntax</td>
<td>1056</td>
</tr>
<tr>
<td>Examples: TREESPLIT Procedure</td>
<td>1057</td>
</tr>
<tr>
<td>Example 21.1: Creating a Binary Classification Tree with Validation Data</td>
<td>1057</td>
</tr>
<tr>
<td>Example 21.2: Creating a Regression Tree</td>
<td>1064</td>
</tr>
<tr>
<td>Example 21.3: Assessing Variable Importance</td>
<td>1067</td>
</tr>
<tr>
<td>References</td>
<td>1068</td>
</tr>
</tbody>
</table>
Overview: TREESPLIT Procedure

The TREESPLIT procedure builds tree-based statistical models for classification and regression in SAS Viya. The procedure produces a classification tree, which models a categorical response, or a regression tree, which models a continuous response. Both types of trees are called decision trees, because the model is expressed as a series of if-then statements. For each type of tree, you specify a response variable (also called a target variable), whose values you want PROC TREESPLIT to predict, and one or more input variables (called predictor variables), whose values the procedure uses to predict the values of the target variable.

The predictor variables for tree models can be categorical or continuous. The set of all possible combinations of the predictor variables is called the predictor space. The model is based on partitioning the predictor space into nonoverlapping segments, which correspond to the terminal nodes (called leaves) of the tree. Partitioning is done repeatedly, starting with the root node, which contains all the data, and continuing until a stopping criterion is met. At each step, the parent node is split into child nodes by selecting a predictor variable and a split value for that variable that minimize the variability, according to a specified measure (or the default measure), in the response variable across the child nodes. Various measures, such as the Gini index, entropy, and residual sum of squares, can be used to assess candidate splits for each node. The selected predictor variable and its split value are called the primary splitting rule.

Tree models are built from training data for which the response values are known, and these models are subsequently used to score (classify or predict) response values for new data. For classification trees, the most frequent response level of the training observations in a leaf is used to classify observations in that leaf. For regression trees, the average response of the training observations in a leaf is used to predict the response for observations in that leaf. The splitting rules that define the leaves provide the information that is needed to score new data; these rules consist of the primary splitting rules, surrogate rules, and default rules for each node.

The process of building a decision tree begins with growing a large, full tree. The full tree can overfit the training data, resulting in a model that does not adequately generalize to new data. To prevent overfitting, the full tree is often pruned back to a smaller subtree that balances the goals of fitting training data and predicting new data. Two commonly applied approaches for finding the best subtree are cost-complexity pruning (Breiman et al. 1984) and C4.5 pruning (Quinlan 1993). For more information, see the section “Building a Decision Tree” on page 1033.

Compared with other regression and classification methods, tree models have the advantage that they are easy to interpret and visualize, especially when the tree is small. Tree-based methods scale well to large data, and they offer various methods of handling missing values, including surrogate splits.

However, tree models have limitations. Regression tree models fit response surfaces that are constant over rectangular regions of the predictor space, so they often lack the flexibility needed to capture smooth relationships between the predictor variables and the response. Another limitation of tree models is that small changes in the data can lead to very different splits, and this undermines the interpretability of the model (Hastie, Tibshirani, and Friedman 2009; Kuhn and Johnson 2013).

PROC TREESPLIT Features

The main features of the TREESPLIT procedure are as follows:
Using CAS Sessions and CAS Engine Librefs

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:

```sas
proc options option=(CASHOST CASPORT);
run;
```

In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:

```sas
cas mysess;
libname mycas cas sessref=mysess;
```

The CAS statement creates the CAS session named `mysess`, and the LIBNAME statement creates the `mycas` CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the CASHOST and CASPORT of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the `mysess` session, you can terminate it by using the TERMINATE option in the CAS statement as follows:

```sas
proc cas terminate session=mysess;
```

The TREESPLIT procedure uses ODS Graphics to create plots as part of its output. For information about the statistical graphics available with the TREESPLIT procedure, see the PLOTS options in the PROC TREESPLIT statement and the section “ODS Graphics” on page 1055.

Using CAS Sessions and CAS Engine Librefs

- provides a computationally efficient strategy for generating candidate splits
- provides the cost-complexity, C4.5, and reduced-error methods of pruning trees
- supports the use of validation data for selecting the best subtree during pruning
- supports the use of \(k\)-fold cross validation for cost-complexity pruning when validation data are not present
- provides various methods of handling missing values, including surrogate rules
- creates tree diagrams and plots for pruning analysis
- computes statistics for assessing model fit
- computes measures of variable importance
- produces a file that contains SAS DATA step code for scoring new data
- provides an output data table that contains leaf assignments and predicted values for observations
Getting Started: TREESPLIT Procedure

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

This example explains basic features of the TREESPLIT procedure for building a classification tree. The data contain measurements of 13 chemical attributes for 178 samples of wine. Each wine is derived from one of three cultivars that are grown in the same area of Italy, and the goal of the analysis is a model that classifies samples into cultivar groups. The data are available from the UCI Irvine Machine Learning Repository (Lichman 2013).

The following statements create a data set named `Wine` that contains the measurements:

```sas
data Wine;
%let url = http://archive.ics.uci.edu/ml/machine-learning-databases;
infile "&url/wine/wine.data" url delimiter=',';
input Cultivar Alcohol Malic Ash Alkan Mg TotPhen Flav NFPhen Cyanins Color Hue ODRatio Proline;
label Cultivar = "Cultivar"
   Alcohol = "Alcohol"
   Malic = "Malic Acid"
   Ash = "Ash"
   Alkan = "Alkalinity of Ash"
   Mg = "Magnesium"
   TotPhen = "Total Phenols"
   Flav = "Flavonoids"
   NFPhen = "Nonflavonoid Phenols"
   Cyanins = "Proanthocyanins"
   Color = "Color Intensity"
   Hue = "Hue"
   ODRatio = "OD280/OD315 of Diluted Wines"
   Proline = "Proline";
run;
```

The following DATA step loads the `mycas.Wine` data into your CAS session. This DATA step assumes that your CAS engine libref is named `mycas`, but you can substitute any appropriately defined CAS engine libref.

---

1 Disclaimer: SAS may reference other websites or content or resources for use at Customer’s sole discretion. SAS has no control over any websites or resources that are provided by companies or persons other than SAS. Customer acknowledges and agrees that SAS is not responsible for the availability or use of any such external sites or resources, and does not endorse any advertising, products, or other materials on or available from such websites or resources. Customer acknowledges and agrees that SAS is not liable for any loss or damage that may be incurred by Customer or its end users as a result of the availability or use of those external sites or resources, or as a result of any reliance placed by Customer or its end users on the completeness, accuracy, or existence of any advertising, products, or other materials on, or available from, such websites or resources.
The following statements print the first 10 observations of Wine, as shown in Figure 21.1:

```
proc print data=Wine(obs=10); run;
```

The variable Cultivar is a nominal categorical variable that has levels 1, 2, and 3, and the 13 attribute variables are continuous.

The following statements use the TREESPLIT procedure to create a classification tree:

```
ods graphics on;

proc treesplit data=mycas.Wine seed=54321;
  class Cultivar;
  model Cultivar = Alcohol Malic Ash Alkan Mg TotPhen Flav NFPhen Cyanins Color Hue ODRatio Proline;
  grow entropy;
  prune costcomplexity(leaves=SE);
run;
```

The MODEL statement specifies Cultivar as the response variable and the variables to the right of the equal sign as the predictor variables. The inclusion of Cultivar in the CLASS statement designates it as a categorical response variable and requests a classification tree. All the predictor variables are treated as continuous variables because none are included in the CLASS statement.

The GROW and PRUNE statements control two fundamental aspects of building classification and regression trees: growing and pruning. You use the GROW statement to specify the criterion for recursively splitting parent nodes into child nodes as the tree is grown. For classification trees, the default criterion is entropy; for more information, see the section “Splitting Criteria” on page 1036.

By default, the growth process continues until the tree reaches a maximum depth of 10 (you can specify a different limit by using the MAXDEPTH= option). The result is often a large tree that overfits the data and is likely to perform poorly in predicting future data. A recommended strategy for avoiding this problem is to prune the tree to a smaller subtree that minimizes prediction error. You can use the PRUNE statement to specify the method of pruning. The default method is cost complexity.
The default output includes several informational tables, which are shown in Figure 21.2 through Figure 21.7. The “Model Information” table in Figure 21.2 provides information about the model and the methods that are used to grow and prune the tree.

**Figure 21.2** Model Information

<table>
<thead>
<tr>
<th>The TREESPLIT Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Model Information</strong></td>
</tr>
<tr>
<td>Split Criterion: Entropy</td>
</tr>
<tr>
<td>Pruning Method: Cost Complexity</td>
</tr>
<tr>
<td>Max Branches per Node: 2</td>
</tr>
<tr>
<td>Max Tree Depth: 10</td>
</tr>
<tr>
<td>Tree Depth Before Pruning: 4</td>
</tr>
<tr>
<td>Tree Depth After Pruning: 2</td>
</tr>
<tr>
<td>Number of Leaves Before Pruning: 9</td>
</tr>
<tr>
<td>Number of Leaves After Pruning: 4</td>
</tr>
</tbody>
</table>

The “Observation Information” table in Figure 21.3 provides the numbers of observations that are read and used. These numbers are the same in this example because there are no missing values in the predictor variables and the ASSIGNMISSING= option is not set to NONE.

**Figure 21.3** Observation Information

<table>
<thead>
<tr>
<th>Training</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Observations Read: 178</td>
</tr>
<tr>
<td>Number of Observations Used: 178</td>
</tr>
</tbody>
</table>

The plot in Figure 21.4 is a tool for selecting the tuning parameter for cost-complexity pruning. The parameter (indicated on the lower horizontal axis) indexes a sequence of progressively smaller subtrees that are nested within the large tree. The parameter value 0 corresponds to the fully grown tree, and positive values control the trade-off between complexity (number of leaves) and fit to the training data, as measured by average misclassification rate.
Figure 21.4 shows that the minimum average misclassification rate, which is obtained by 10-fold cross validation, is 0.0844. On the plot, this value is indicated by a filled-in circle. Information about the 1-SE misclassification rate is also included because the LEAVES=SE suboption was specified.

Breiman’s 1-SE rule chooses the parameter that corresponds to the smallest subtree for which the misclassification rate is less than one standard error above the minimum misclassification rate (Breiman et al. 1984). The parameter value that corresponds to the 1-SE rule is indicated by a star. The value 0.143 corresponds to a tree that only has three leaves. The dotted vertical line in Figure 21.4 indicates the chosen tree.

The tree diagram in Figure 21.5, which is produced by default when ODS Graphics is enabled, provides an overview of the tree as a classifier.
The tree is constructed starting with all the observations in the root node (labeled 0). This node is split into one internal node and one leaf node (1 and 2, respectively), and node 1 is further split into leaf nodes (3 and 4).

The color of the bar in each leaf node indicates the most frequent level of Cultivar among the observations in that node; this is also the classification level that is assigned to all observations in that node. The height of the bar indicates the proportion of observations in the node that have the most frequent level. The width of the link between parent and child nodes is proportional to the number of observations in the child node.

The diagram in Figure 21.6 provides more detail about the nodes and splits.
The detailed tree diagram displays a box for each node; the box contains six lines of information, separated by a horizontal line. The proportion of each level of the predictor variable is shown below the horizontal line, and the level that has the highest proportion is also displayed above the horizontal line. Also displayed above the horizontal line are the node identifier and the number of observations that are assigned to the node.

The root node (node 0) contains 178 samples. Because no MAXBRANCH= option is specified in the preceding statements, PROC TREESPLIT divides each node into two child nodes (MAXBRANCH=2 by default). At node 0, PROC TREESPLIT determines that the impurity of the root node is maximally decreased (as measured by the entropy criterion, which is the default) by splitting the 178 observations such that all samples for which \( ODRatio \geq 2.4985 \) are assigned to node 1 and all samples for which \( ODRatio < 2.4985 \) are assigned to node 2. This is the primary splitting rule for node 0. In this training phase, 109 samples are assigned to node 1 and 69 samples are assigned to node 2.

Figure 21.6 also indicates that node 1 contains 0 samples with level 2. The legend shows that level 2 corresponds to \textit{Cultivar}=3, so node 1 contains no samples of the third cultivar. Similarly, the diagram indicates...
that node 2 contains 0 samples with level 1, and the legend shows that level 1 corresponds to Cultivar=1, so node 2 contains no samples of the first cultivar.

The primary splitting rule for node 1 consists of the variable Alcohol and the split value 12.74. All samples for which Alcohol ≥ 12.74 are assigned to node 3 (which then contains 64 samples), and all samples for which Alcohol < 12.74 are assigned to node 4 (which then contains 45 samples).

The resulting classification tree yields simple rules for predicting the cultivar. For example, a sample for which ODRatio ≥ 2.4985 and Alcohol < 12.74 is predicted to be from the second cultivar (node 4 indicates that level 3 has the highest proportion of observations, and the legend shows that level 3 corresponds to Cultivar=2).

Figure 21.6 displays the entire tree that begins with the root node and has a depth of three levels. You can use the PLOTS=ZOOMEDTREE option in the PROC TREESPLIT statement to request diagrams that begin with other nodes and have specified depths.

The table in Figure 21.7 displays fit statistics for the tree model.

<table>
<thead>
<tr>
<th>Number of Leaves</th>
<th>Misclassification Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>4</td>
</tr>
</tbody>
</table>

The misclassification rate is the total proportion of the 178 wine samples that were misclassified. The following numbers of samples were misclassified in the terminal nodes:

- node 2: 21
- node 3: 5
- node 4: 0

So the total misclassification rate is \((21 + 5 + 0)/178 \approx 0.1461\).
Syntax: TRESPLIT Procedure

The following statements and options are available in the TRESPLIT procedure:

```plaintext
PROC TRESPLIT < options > ;
    AUTOTUNE < options > ;
    CLASS variables ;
    CODE < options > ;
    FREQ variable ;
    GROW criterion < options > ;
    MODEL response = variable . . . ;
    OUTPUT OUT=CAS-libref.data-table output-options ;
    PARTITION < partition-options > ;
    PRUNE prune-method < (prune-options) > ;
    VIICODE < options > ;
    WEIGHT variable ;
```

The PROC TRESPLIT statement and the MODEL statement are required. If any variables are character or are to be treated as categorical, at least one CLASS statement is required. Variables that appear after the equal sign (=) in the MODEL statement are predictor variables that model the response variable. By default, all variables that appear in the MODEL statement are treated as continuous variables. A CLASS statement causes a variable to be treated as categorical. Specifying a variable in a CLASS statement but not in a MODEL statement causes the variable to be implicitly added to the MODEL statement.

The following sections describe the PROC TRESPLIT statement and then describe the other statements in alphabetical order.

### PROC TRESPLIT Statement

```plaintext
PROC TRESPLIT < options > ;
```

The PROC TRESPLIT statement invokes the procedure. Table 21.1 summarizes the options in the PROC TRESPLIT statement.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BINMETHOD=</td>
<td>Specifies how to bin interval inputs prior to training</td>
</tr>
<tr>
<td>CVCC</td>
<td>Requests a table of the results of the cost-complexity pruning based on cross validation</td>
</tr>
<tr>
<td>INMODEL=</td>
<td>Specifies a saved tree model to use to score a new table</td>
</tr>
<tr>
<td>NUMBIN=</td>
<td>Specifies the number of bins to use for continuous variables</td>
</tr>
<tr>
<td>NOPRINT</td>
<td>Suppresses ODS output</td>
</tr>
<tr>
<td>NSURROGATES=</td>
<td>Specifies the number of surrogate rules to create</td>
</tr>
<tr>
<td>OUTMODEL=</td>
<td>Specifies the data table to which the decision tree model is to be saved</td>
</tr>
</tbody>
</table>
Table 21.1 continued

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PLOTS=</td>
<td>Specifies options for plots</td>
</tr>
<tr>
<td>PRINTTARGET</td>
<td>Create tables that contain information about columns in the output</td>
</tr>
<tr>
<td>PRUNINGTABLE</td>
<td>Creates a table that contains the pruning information</td>
</tr>
<tr>
<td>RBAIMP</td>
<td>Creates a variable importance table by using random branch assignment</td>
</tr>
<tr>
<td>SEED=</td>
<td>Specifies the random number seed to use for autotuning</td>
</tr>
<tr>
<td>VII=</td>
<td>Calculates the importance of the specified types of variable interactions</td>
</tr>
</tbody>
</table>

Splitting Options

| ASSIGNMISSING= | Specifies how to handle missing values in a predictor variable |
| CLUSTERSPLIT   | Requests that an alternative method of determining splits be used |
| MAXBRANCH=     | Specifies the maximum number of child nodes per parent node |
| MAXDEPTH=      | Specifies the maximum tree depth                     |
| MINLEAFSIZE=   | Specifies the minimum number of observations per child node |
| MINUSEINSEARCH=| Specifies the minimum number of observations to use with the USEINSEARCH policy for handling missing values |
| SPLITONCE      | Specifies that a variable can be used to split only once per branch |

You can specify the following options:

**ASSIGNMISSING=** BRANCH | NONE | MACSMALL | POPULAR | SIMILAR | USEINSEARCH

specifies how to handle missing values of predictor variables during training and how to handle missing values and unknown levels of predictor variables after all surrogate rules have been applied during scoring. An unknown level of a categorical predictor variable is a level that does not exist in the training data but is encountered during scoring. During scoring, unknown levels are treated as missing values.

Both this option and the NSURROGATES= option affect how missing values are handled in model training and scoring. For more information about how these rules are applied, see the section “Primary and Surrogate Splitting Rules” on page 1038.

You can specify one of the following values:

**BRANCH** during the training phase, assigns any observation that has a missing value for the predictor variable to a specially created child node (branch). If all observations have nonmissing values for a predictor variable, then no branch is created to contain observations with missing values. In the scoring phase, if an observation has a missing value for a predictor variable and no special branch was created to contain observations with missing values, or if an observation has an unknown level for a predictor variable, then the observation is assigned to the child node that contains the most training observations.

**NONE** during the training phase, excludes any observation that has a missing value for any predictor variable. In the scoring phase, if an observation has a missing value or an unknown level for a categorical predictor variable, then the observation is assigned to the child node that contains the most training observations. If an observation has
a missing value for a continuous predictor, then the observation is assigned to the child node with the smallest numerical values of the predictor.

**MACSMALL**
during the training phase, treats a missing value in a categorical predictor variable as a separate, legitimate value. If all observations have nonmissing values for a categorical predictor variable, then no branch is selected to contain observations with missing values. In the scoring phase, if an observation has a missing value for a categorical predictor variable and no branch is selected to contain observations with missing values, or if an observation has an unknown level for a predictor variable, then the observation is assigned to the child node that contains the most training observations.

In both the training and the scoring phases, missing values in continuous predictor variables are treated as the smallest possible value.

**POPULAR**
during the training phase, assigns any observation that has a missing value in the predictor variable to the child node that has the most training observations. In the scoring phase, if an observation has a missing value or an unknown level for a predictor variable, then the observation is assigned to the child node that contains the most training observations.

**SIMILAR**
during the training phase, assigns any observation that has a missing value in the predictor variable to the child node whose observations are most similar to it. This similarity is determined using the chi-square criterion for categorical responses or the $F$-test criterion for continuous responses. If all observations have nonmissing values for a predictor variable, then no branch is selected to contain observations with missing values. In the scoring phase, if an observation has a missing value for a predictor variable and no branch is selected to contain observations with missing values, or if an observation has an unknown level for a predictor variable, then the observation is assigned to the child node that contains the most training observations.

**USEINSEARCH**
during the training phase, treats a missing value in a predictor variable as a separate, legitimate value. If all observations have nonmissing values for a predictor variable, then no branch is selected to contain observations with missing values. In the scoring phase, if an observation has a missing value for a predictor variable and no branch is selected to contain observations with missing values, or if an observation has an unknown level for a predictor variable, then the observation is assigned to the child node that contains the most training observations.

By default, ASSIGNMISSING=USEINSEARCH.

**BINMETHOD=BUCKET | QUANTILE**
specifies how to bin interval input variables prior to growing the decision tree. The number of bins that are created is determined by the NUMBIN= option.

You can specify one of the following values:

**BUCKET**
 bins interval input variables into fixed-width bins. The width of each bin for a particular variable is calculated by subtracting the smallest value among all observations from the largest value among all observations, and then dividing that result by the number of bins.
QUANTILE bins interval input variables into bins according to their quantile. The width of the bins for a particular variable is not fixed, but the number of observations in each bin is approximately equal.

By default, BINMETHOD=BUCKET.

CLUSTERSPLIT determines the splits at each node using clustering on each input variable, and then chooses the splitting variable on the basis of which variable and split optimize the criterion that is specified in the GROW statement.

CVCC

CVCCOSTCOMPLEXITY requests a table of the results of cost-complexity pruning based on cross validation. For each penalty parameter in the cross validation, the table provides the penalty parameter, the minimum, the maximum, and the average error. The error is the misclassification rate when the response variable is categorical and is the average square error (ASE) when the response variable is continuous. You can use the PLOTS=CVCC option to request a plot of the information in this table.

DATA=CAS-libref.data-table

names the input data table for PROC TREESPLIT to use. CAS-libref.data-table is a two-level name, where

CAS-libref refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to the data, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about CAS-libref, see the section “Using CAS Sessions and CAS Engine Librefs” on page 1003.

data-table specifies the name of the input data table.

INMODEL=<CAS-libref.data-table>

specifies the data table that you have previously saved as a tree model by using the OUTMODEL= option in a previous run of PROC TREESPLIT. CAS-libref.data-table is a two-level name, where CAS-libref refers to the caslib and session identifier, and data-table specifies the name of the input data table. For more information about this two-level name, see the DATA= option and the section “Using CAS Sessions and CAS Engine Librefs” on page 1003.

When you use the INMODEL= option, the OUTPUT statement is required and any other options in the PROC TREESPLIT statement, except for NOPRINT and PLOTS, are ignored.

The data table used in the INMODEL= option must include the attributes that are associated with the table produced by the OUTMODEL= option, or the TREESPLIT procedure will error.

MAXBRANCH=b

specifies the maximum number of child nodes per parent node in the tree. PROC TREESPLIT tries to create this number of children unless it is impossible (for example, if a split variable does not have enough levels).

By default, MAXBRANCH=2.
**MAXDEPTH=** *number*

specifies the maximum depth of the tree to be grown. The number of levels in a tree is equal to the depth plus one. The default is calculated by the following equation, where *b* is the value of the **MAXBRANCH=** option in the PROC TREESPLIT statement:

\[
\text{MaxDepth} = \left\lceil \frac{10}{\log_2 (b)} \right\rceil
\]

This value can be tuned with the AUTOTUNE statement.

**MINLEAFSIZE=** *number*

specifies the minimum *number* of observations in the training data that each child of a split must contain in order for the split to be considered.

By default, **MINLEAFSIZE=5**.

**MINUSEINSEARCH=** *number*

specifies a threshold for using missing values in the split search when **ASSIGNMISSING=USEINSEARCH** as the missing value policy. If the number of observations that have missing values for the splitting variable is greater than or equal to *number*, then PROC TREESPLIT uses the USEINSEARCH policy for missing values.

By default, **MINUSERINSEARCH=1**.

**NOPRINT**

suppresses the generation of ODS output.

**NUMBIN=** *number*

specifies the *number* of bins to use for binning interval predictor variables. PROC TREESPLIT bins continuous predictors to a fixed bin size. This option controls the number of bins and thereby also the size of the bins.

By default, **NUMBIN=20**. This value can be tuned with the AUTOTUNE statement.

**NSURROGATES=** *number*

specifies the number of surrogate rules to create for each splitting rule, where *number* is an integer greater than 0. Surrogate rules are backup splitting rules that are used when the variable that corresponds to the primary splitting rule is missing.

Both this option and the ASSIGNMISSING= affect training and scoring. For more information, see the section “Primary and Surrogate Splitting Rules” on page 1038.

By default, **NSURROGATES=0**.

**OUTMODEL=** *CAS-libref.data-table*

specifies the data table to which to save the decision tree model. *CAS-libref.data-table* is a two-level name, where *CAS-libref* refers to the caslib and session identifier, and *data-table* specifies the name of the output data table. For more information about this two-level name, see the DATA= option and the section “Using CAS Sessions and CAS Engine Librefs” on page 1003.

The data table that results from this option contains information about each node in the decision tree model, including the splitting variables, the child nodes, the number of observations at each node, and the predicted response at each node. This table also has attributes attached that contain information about the input variables, the target variable, and the decision tree model.
Chapter 21: The TREESPLIT Procedure

The `PLOTS` statement controls the plots that are produced through ODS Graphics. When you specify only one `plot-request`, you can omit the parentheses around it.

You can specify the following `global-plot-option`:

- **ONLY**
  - suppresses the default plots. Only plots that you specifically request are displayed.

You can specify the following `plot-requests`:

- **ALL**
  - produces all appropriate plots.

- **CVCC**
  - produces a plot that is used to determine the tuning parameter for cost-complexity pruning when \( k \)-fold cross validation is used. This plot displays the average misclassification rate when the response variable is a classification variable and displays the average square error (ASE) when the response variable is a continuous variable.

- **NONE**
  - suppresses the default plots. Only plots that you specifically request are displayed.

- **PRUNEUNTIL**
  - plots the metric that is used to select the final subtree.
  
  This option is enabled by default except when you specify the OFF option in the `PRUNE` statement (which turns off pruning completely). Specifying the PRUNEUNTIL option has no effect if you also specify the OFF option in the prune statement.

- **WHOLETREE**
  - produces a plot to visualize the entire finished (grown and pruned) tree. This option is enabled by default.

  You can specify the following values:

  - **LINKSTYLE=CURVED | ORTHOGONAL | STRAIGHT**
    - specifies the style of links between nodes and leaves in the tree. You can specify the following `link-styles`:

    - **CURVED**
      - requests curved links between the nodes and their children.

    - **ORTHOGONAL**
      - requests that links go straight down partway from a node to its children, create a horizontal line at the base of the vertical line, and then go straight down from that line to each child.

    - **STRAIGHT**
      - requests that links go straight from the nodes to their children.

    By default, LINKSTYLE=CURVED.

  - **LINKWIDTH=CONSTANT | PROPORTIONAL**
    - specifies the width of links between nodes and leaves in the tree. You can specify the following values:
CONSTANT requests that all links have the same thickness.

PROPORTIONAL requests that links have a thickness proportional to the total number of observations that go between the node and each child.

By default, LINKWIDTH=PROPORTIONAL.

NOLEGEND turns off the legend.

ZOOMEDTREE < (zoomed-tree-options)> produces a plot to visualize a portion of the finished (grown and pruned) tree. This option is enabled by default.

You can specify the following values:

DEPTH=depth creates a plot down to depth for each node-id specified in the NODES= option.

By default, DEPTH=2.

LINKSTYLE=CURVED | ORTHOGONAL | STRAIGHT specifies the style of links between nodes and leaves in the tree. You can specify the following values:

CURVED requests curved links between the nodes and their children.

ORTHOGONAL requests that links go straight down partway from a node to its children, create a horizontal line at the base of the vertical line, and then go straight down from that line to each child.

STRAIGHT requests that links go straight from the nodes to their children.

By default, LINKSTYLE=CURVED.

LINKWIDTH=CONSTANT | PROPORTIONAL specifies the width of links between nodes and leaves in the tree. You can specify the following values:

CONSTANT requests that all links have the same thickness.

PROPORTIONAL requests that links have a thickness proportional to the total number of observations that go between the node and each child.

By default, LINKWIDTH=PROPORTIONAL.

NODES=(node-id < node-id < ... >>) requests a plot for a subtree that is rooted at each node specified by a node-id. The values of node-id are alphanumeric strings that are displayed within the nodes in the plot that is created by the WHOLETREE option. PROC TREESPLIT creates one plot for each node-id that you specify. The default node ID is “0,” the root of the entire tree.
Chapter 21: The TREESPLIT Procedure

NOLEGEND
suppresses the display of the legend

PRINTTARGET
outputs tables that indicate generated columns in the OUT= table from the OUTPUT statement. For a continuous response, PROC TREESPLIT generates an output table named PredName, which indicates the name of the predicted value column. For a categorical response, PROC TREESPLIT generates an output table named PredIntoName, which indicates the name of the predicted value column, and also an output table named PredProbName, which indicates the names of the predicted probability columns.

PRUNINGTABLE
outputs a table of the pruning results.

RBAIMP
creates a variable importance table by using random branch assignment (RBA). This table is created in addition to the normal variable importance table that is calculated using the residual sum of squares (RSS) error. For more information about RBA and RSS variable importance, see the section “Variable Importance” on page 1045.

SEED=number
specifies the initial seed for random number generation for cross validation. The value of number must be an integer. By default, the seed is generated by reading the time of day from the computer’s clock.

SPLITONCE
splits predictor variables only once on a branch. When you specify this option, a variable cannot be split more than once on the path from the root to any particular terminal node. However, a variable can be used more than once across branches.

VII=2 | 3
INTERACTIONIMP=2 | 3
calculates the variable interaction importance, which is described in the section “Variable Interaction Importance” on page 1047. You can specify the following values:

2 calculates the importance of all two-way variable interactions.
3 calculates the importance of all three-way and all two-way variable interactions.

If you do not specify this option, then the variable interaction importance is not calculated.

AUTOTUNE Statement

AUTOTUNE < options > ;

The AUTOTUNE statement searches for the best combination of values of the MAXDEPTH= and NUMBIN= options in the PROC TREESPLIT statement and the criterion in the GROW statement. If you specify the AUTOTUNE statement, then you cannot also specify k-fold cross validation with cost-complexity pruning.

Table 21.2 summarizes the options you can specify in the AUTOTUNE statement.
### Table 21.2 AUTOTUNE Statement Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EVALHISTORY=</td>
<td>Specifies how to report the evaluation history of the tuner.</td>
</tr>
<tr>
<td>FRACTION=</td>
<td>Specifies the fraction of observations to use for validation.</td>
</tr>
<tr>
<td>KFOLD=</td>
<td>Specifies the number of folds for k-fold cross validation.</td>
</tr>
<tr>
<td>MAXBAYES=</td>
<td>Specifies the maximum number of points in the kriging model.</td>
</tr>
<tr>
<td>MAXEVALS=</td>
<td>Specifies the maximum number of evaluations.</td>
</tr>
<tr>
<td>MAXITER=</td>
<td>Specifies the maximum number of iterations when SEARCHMETHOD=GA or BAYESIAN</td>
</tr>
<tr>
<td>MAXTIME=</td>
<td>Specifies the maximum time for all iterations.</td>
</tr>
<tr>
<td>MAXTRAININGTIME=</td>
<td>Specifies the maximum time for a model training.</td>
</tr>
<tr>
<td>NCONVITER=</td>
<td>Specifies the number of convergence iterations.</td>
</tr>
<tr>
<td>NOGRIDSHUFFLE</td>
<td>Requests that the grid points not be shuffled.</td>
</tr>
<tr>
<td>NPARALLEL=</td>
<td>Specifies the number of parallel sessions.</td>
</tr>
<tr>
<td>NSUBSESSIONWORKERS=</td>
<td>Specifies the number of workers in parallel sessions.</td>
</tr>
<tr>
<td>OBJECTIVE=</td>
<td>Specifies the objective function.</td>
</tr>
<tr>
<td>POPSIZE=</td>
<td>Specifies the population size when SEARCHMETHOD=GA or BAYESIAN</td>
</tr>
<tr>
<td>SAMPLESIZE=</td>
<td>Specifies the sample size when SEARCHMETHOD=LHS or RANDOM.</td>
</tr>
<tr>
<td>SEARCHMETHOD=</td>
<td>Specifies the search method that the optimizer uses.</td>
</tr>
<tr>
<td>SECONDOBJECTIVE=</td>
<td>Specifies the second objective to use for tuning.</td>
</tr>
<tr>
<td>TARGETEVENT=</td>
<td>Specifies the target event for ROC-based calculations.</td>
</tr>
<tr>
<td>TRAINFRACTION=</td>
<td>Specifies the fraction of observations to use for training.</td>
</tr>
<tr>
<td>TUNINGPARAMETERS=</td>
<td>Specifies the custom tuning parameters.</td>
</tr>
<tr>
<td>USEPARAMETERS=</td>
<td>Specifies how to handle the TUNINGPARAMETERS= option.</td>
</tr>
</tbody>
</table>

**EVALHISTORY=ALL | LOG | NONE | TABLE**

specifies how to report the evaluation history of the tuner.

You can specify one of the following values:

- **ALL** reports each evaluation in the log and creates the EvaluationHistory ODS table.

- **LOG** prints the following information to the log for each evaluation: evaluation number, objective value, best objective value up to that point, evaluation time, and elapsed time since the beginning of the tuning process.

- **NONE** suppresses reporting of evaluations in the log and does not create the EvaluationHistory ODS table.

- **TABLE** creates the EvaluationHistory ODS table, which contains all evaluated points. The table contains columns for the evaluation number, all tuning parameters, and the objective function value.

By default, EVALHISTORY=TABLE
**FRACTION=**number

specifies the fraction of all data to be used for validation, where number must be between 0.01 and 0.99, inclusive. If you specify this option, the tuner uses a single-partition validation for finding the objective value (validation error estimate). Using this option might not be advisable for small or unbalanced data tables, where the random assignment of the validation subset might not provide a good estimate of error. For large, balanced data tables, a single-partition validation is usually sufficient for estimating error; a single partition is more efficient than cross validation in terms of the total execution time.

If a PARTITION statement is specified, the validation partition defined in that statement is used, and this option is ignored. You cannot specify this option in combination with the KFOLD= option.

If the TRAINFRACTION= option is used, then by default the value of the FRACTION= option is one minus the value of the TRAINFRACTION= option. Otherwise, by default, FRACTION=0.3.

**KFOLD=**number

specifies the number of folds (partitions) in the cross validation process, where number must be between 2 and 20, inclusive. If you specify this option, the tuner uses cross validation to find the objective value. In cross validation, each model evaluation requires number of training executions (on number–1 data folds) and number of scoring executions (on one hold-out fold). Thus, the evaluation time is increased by approximately a factor of number. For small to medium data tables or for unbalanced data tables, cross validation provides on average a better representation of error across the entire data table (a better generalization error).

If you do not specify either this option or the FRACTION= option, then the default of FRACTION=0.3 is used. If a PARTITION statement is specified, the validation partition defined in that statement is used, and this option is ignored. You cannot specify this option in combination with the FRACTION= option.

**MAXBAYES=**number

specifies the maximum number of points in the Kriging model. This option is only honored when SEARCHMETHOD=BAYESIAN, and has a minimum value of 10.

By default, MAXBAYES=100.

**MAXEVALS=**number

specifies the maximum number of configuration evaluations allowed for the tuner, where number must be an integer greater than or equal to 3. When the number of evaluations is reached, the tuner terminates the search and returns the results. To produce a single objective function value (validation error estimate), each configuration evaluation requires either a single model training and scoring execution on a validation partition, or a number of training and scoring executions equal to the value of the KFOLD= option for cross validation. The MAXEVALS= option might lead to termination before the value of the MAXITER= option or the MAXTIME= option is reached.

By default, MAXEVALS=50.

**MAXITER=**number

specifies the maximum number of iterations of the optimization tuner, where number must be greater than or equal to 1. Each iteration normally involves a number of objective evaluations up to the value of the POPSIZE= option. The MAXITER= option is used only when SEARCHMETHOD=GA or SEARCHMETHOD=BAYESIAN; it is ignored when SEARCHMETHOD=RANDOM or SEARCHMETHOD=LHS. The MAXITER= option might lead to termination before the value of the MAXEVALS= option or the MAXTIME= option is reached.
By default, MAXITER=5.

**MAXTIME=number**
specifies the maximum time (in seconds) allowed for the tuner, where *number* must be greater than or equal to 1. When this value is reached, the tuner terminates the search and returns results. The actual run time for optimization might be longer because it includes the remaining time needed to finish the current evaluation. For long-running model training (large data tables), the actual run time might significantly exceed *number*. The MAXTIME= option might lead to termination before the value of the MAXEVALS= option or the MAXITER= option is reached.

By default, MAXTIME=36000.

**MAXTRAINTIME=number**
specifies the maximum time allowed for a single model train. The model train is terminated if it exceeds this time, and the objective value is set to missing.

By default, there is no maximum time allowed for a single train.

**NCONVITER=number**
specifies the number of convergence iterations after which tuning is terminated.

By default, NCONVITER=4.

**NOGRIDSHUFFLE**
requests that the points that are generated by the grid search method not be shuffled before execution.

**NPARALLEL=number**
specifies the number of evaluations to be performed in parallel, where *number* must be greater than or equal to 0.

By default, NPARALLEL=0, which indicates that the value of *number* is determined as follows:

- If SEARCHMETHOD=GA or SEARCHMETHOD=BAYESIAN, then the number of parallel evaluations is equal to the value of the POPSIZE= option minus 1.
- If SEARCHMETHOD=LHS or SEARCHMETHOD=RANDOM, then the number of parallel evaluations is equal to the value of the SAMPLESIZE= option, with a maximum value of 32.

**NSUBSESSIONWORKERS=number**
specifies the number of workers to use in parallel subsessions. When alternative configurations are evaluated in parallel, a number of subsessions is created by the tuner, with each using multiple workers. By default, the number of workers that are used in a parallel subsession is determined as described in the section “Determining the Number of Parallel Evaluations” on page 1051.

**OBJECTIVE=function**
specifies which measure of model performance the tuner uses as the objective function.

You can specify one of the following values for *function*. Some values can be specified only when the target variable is of a particular type, as shown in parentheses.

**ASE**
uses average squared error as the objective function.

**AUC**
uses area under the curve as the objective function (nominal type only).

**F05**
uses the F0.5 coefficient as the objective function (nominal type only).
F1 uses the F1 coefficient as the objective function (nominal type only).

GAMMA uses the gamma coefficient as the objective function (nominal type only).

GINI uses the Gini coefficient as the objective function (nominal type only).

KS uses the Kolmogorov-Smirnov coefficient as the objective function (nominal type only).

MAE uses the mean absolute error as the objective function (interval type only).

MCE uses the misclassification rate as the objective function (nominal type only).

MCLL uses the multiclass log loss as the objective function (nominal type only).

MISC uses the misclassification error percentage as the objective function (nominal type only).

MSE uses the mean squared error as the objective function (interval type only).

MSLE uses the mean squared logarithmic error as the objective function (interval type only).

RASE uses the root average squared error as the objective function.

RMAE uses the root mean absolute error as the objective function (interval type only).

RMSLE uses the root mean squared logarithmic error as the objective function (interval type only).

TAU uses the tau coefficient as the objective function (nominal type only).

By default, OBJECTIVE=MISC for nominal targets, and OBJECTIVE=MSE for interval targets.

POPSIZE=number specifies the maximum population size, where population is the number of configuration evaluations in one iteration (population). The number must be greater than or equal to 1. In some cases, the tuner algorithm might generate a number of new configurations that is smaller than number. The POPSIZE= option is used only when SEARCHMETHOD=GA or SEARCHMETHOD=BAYESIAN; it is ignored when SEARCHMETHOD=RANDOM or SEARCHMETHOD=LHS.

By default, POPSIZE=10.

SAMPLESIZE=number specifies the total number of evaluations, where number must be greater than or equal to 1. This option is only used when SEARCHMETHOD=RANDOM or SEARCHMETHOD=LHS; it is ignored when SEARCHMETHOD=GA or SEARCHMETHOD=BAYESIAN.

By default, SAMPLESIZE=50.

SEARCHMETHOD=BAYESIAN | GA | GRID | LHS | RANDOM specifies the search method to use for tuning. You can specify the following values:

BAYESIAN builds a kriging surrogate model to approximate the objective value and uses this surrogate model to generate new alternative configurations at each iteration. The kriging model is continuously updated during the search process.

GA uses an initial Latin hypercube sample that seeds a genetic algorithm to generate a new population of alternative configurations at each iteration.
GRID uses all combinations of selected values of the hyperparameters.

LHS uses a Latin hypercube to generate a single sample of configurations that is uniform in each tuning parameter but random in combinations.

RANDOM generates a single sample of purely random configurations.

By default, SEARCHMETHOD=GA.

SECONDOBJECTIVE=NONE | SCORETIME | TRAINTIME
specifies the type of the second objective to use for tuning. You can specify the following values:

NONE does not use a second objective for tuning.

SCORETIME specifies the model scoring time as the second objective for tuning.

TRAINTIME specifies the model training time as the second objective for tuning.

By default, SECONDOBJECTIVE=NONE.

TARGETEVENT=string
specifies the target event to use for calculating the selected objective function. This option is ignored when the value of the OBJECTIVE= option is not AUC, F1, F05, GINI, GAMMA, TAU, or KS.

If you do not specify the TARGETEVENT= option, the tuner selects one of the target levels and uses it for calculating the specified objective function.

TRAINFRACTION=number
specifies the fraction of all data to be used for training, where number must be between 0.01 and 0.99, inclusive. If you specify this option, the tuner uses a single-partition validation to find the objective value (validation error estimate). Using this option might not be advisable for small or unbalanced data tables, where the random assignment of the validation subset might not provide a good estimate of error. For large, balanced data tables, a single-partition validation is usually sufficient for estimating error; a single partition is more efficient than cross validation in terms of the total execution time.

If a PARTITION statement is specified, the validation partition defined in that statement is used, and this option is ignored. You cannot specify this option in combination with the KFOLD= option.

If the FRACTION= option is used, then by default the value of the TRAINFRACTION= option is one minus the value of the FRACTION= option. Otherwise, by default, TRAINFRACTION=0.7.

TUNINGPARAMETERS=(suboption | ... | < suboption >)
TUNEPARMS=(suboption | ... | < suboption >)
specifies which parameters to tune and which ranges to tune over. If USEPARAMETERS=STANDARD, this option is ignored.

You can specify one or more of the following suboptions:

CRITERION (VALUES=value-list INIT=value EXCLUDE)
specifies information about the splitting criteria to use for tuning the decision tree. For more information about the criteria, see the GROW statement.

You can specify the following additional suboptions:
VALUES=value-list
specifies a list of splitting criteria to consider during tuning, where value-list is a space-separated list that can include one or more of the following values: CHAID, CHISQUARE, FTEST, GAIN, GINI, and VARIANCE.

INIT=value
specifies the initial splitting criterion for the tuner to use. You can specify the following values: CHAID, CHISQUARE, FTEST, GAIN, GINI, IGR, RSS, or VARIANCE.

By default, INIT=IGR for nominal targets and INIT=VARIANCE for interval targets.

EXCLUDE
excludes the splitting criterion from the tuning process.

MAXDEPTH (LB=number UB=number VALUES=value-list INIT=number EXCLUDE)
specifies information about the maximum depth to grow the decision tree to use for tuning the decision tree. For more information, see the MAXDEPTH= option.

You can specify the following additional suboptions:

LB=number
specifies the minimum depth to consider during tuning. If you specify this suboption, you cannot specify the VALUES= suboption.

By default, LB=1.

UB=number
specifies the maximum depth to consider during tuning. If you specify this suboption, you cannot specify the VALUES= suboption.

By default, UB=19.

VALUES=value-list
specifies a list of depth values to consider during tuning, where value-list is a space-separated list of positive integers. If you specify this suboption, you cannot specify either the LB= or UB= suboption.

INIT=number
specifies the initial depth for the tuner to use.

By default, INIT=10.

EXCLUDE
excludes depth from the tuning process.

NUMBIN (LB=number UB=number VALUES=value-list INIT=number EXCLUDE)
specifies information about the number of bins in which to bin the interval inputs while tuning the decision tree. For more information, see the NUMBIN= option.

You can specify the following additional suboptions:
**CLASS Statement**

**LB=** *number*

specifies the minimum number of bins to consider during tuning. If you specify this suboption, you cannot specify the VALUES= suboption.

By default, LB=20.

**UB=** *number*

specifies the maximum number of bins to consider during tuning. If you specify this suboption, you cannot specify the VALUES= suboption.

By default, UB=200.

**VALUES=** *value-list*

specifies a list of numbers of bins to consider during tuning, where *value-list* is a space-separated list of positive integers. If you specify this suboption, you cannot specify either the LB= or UB= suboption.

**INIT=** *number*

specifies the initial number of bins for the tuner to use.

By default, INIT=20.

**EXCLUDE**

excludes the number of bins from the tuning process.

**USEPARAMETERS=** *tuning-parameter-option*

specifies which set of parameters to tune.

You can specify the following *tuning-parameter-options*:

- **STANDARD** tunes using the default bounds and initial values for all parameters.
- **CUSTOM** tunes only the parameters that are specified in the TUNINGPARAMETERS= option.
- **COMBINED** tunes the parameters that are specified in the TUNINGPARAMETERS= option and uses default bounds and initial values to tune all other parameters.

---

**CLASS Statement**

**CLASS** *variables* ;

The CLASS statement causes the specified *variables* to be treated as categorical variables in the analysis. These variables enter the analysis not through their values but through levels to which the unique values are mapped. For more information about these mappings, see the section “Levelization of Classification Variables” on page 49.

You can specify only one CLASS statement.

**NOTE:** All classification levels are padded or truncated to 32 characters.
Chapter 21: The TREESPLIT Procedure

CODE Statement

```sas
CODE < options > ;
```

The CODE statement writes SAS DATA step code for computing predicted values of the fitted model to a file, to a catalog entry, or to a CAS table. To score new data, you can then include the file or the catalog entry in a DATA step, or you can specify the CAS table in the `runCodeTable` action in the `dataStep` action set (for more information, see SAS Viya: System Programming Guide).

Table 21.3 summarizes the options available in the CODE statement.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMMENT</td>
<td>Adds comments to the generated code</td>
</tr>
<tr>
<td>FILE=</td>
<td>Names the file in which to save the generated code</td>
</tr>
<tr>
<td>FORMATWIDTH=</td>
<td>Specifies the numeric format width for the regression coefficients</td>
</tr>
<tr>
<td>INDENTSIZE=</td>
<td>Specifies the number of spaces to indent the generated code</td>
</tr>
<tr>
<td>LABELID=</td>
<td>Specifies a number used to construct names and labels</td>
</tr>
<tr>
<td>LINESIZE=</td>
<td>Specifies the line size for the generated code</td>
</tr>
<tr>
<td>NOTRIM</td>
<td>Compares formatted values, including blank padding</td>
</tr>
<tr>
<td>OUT=</td>
<td>Names an output CAS table in which to save the generated code</td>
</tr>
</tbody>
</table>

For more information about the syntax of the CODE statement, see the section “CODE Statement” on page 16 in Chapter 3, “Shared Concepts.”

FREQ Statement

```sas
FREQ variable ;
```

The `variable` in the FREQ statement identifies a numeric variable in the data set that contains the frequency of occurrence of each observation. PROC TREESPLIT treats each observation as if it appears `f` times, where `f` is the value of the FREQ `variable` for the observation. If `f` is not an integer, it is truncated to an integer. If `f` is less than 1 or missing, the observation is not used in the analysis. When the FREQ statement is not specified, each observation is assigned a frequency of 1.

GROW Statement

```sas
GROW criterion < (options) > ;
```

The GROW statement specifies the `criterion` by which to split a parent node into child nodes. As it grows the tree, PROC TREESPLIT calculates the specified `criterion` for each predictor variable and then splits on the predictor variable that optimizes the specified `criterion`.

For categorical responses, the available criteria are CHAID, CHISQUARE, ENTROPY, GINI, and IGR; the default is IGR. For continuous responses, the available criteria are CHAID, FTEST, and RSS; the default is RSS.
For either categorical or continuous responses, you can specify the following criterion:

**CHAID < (options) >**

for categorical predictor variables, CHAID uses the value (as specified in the ALPHA= option) of a chi-square statistic (for a classification tree) or an $F$ statistic (for a regression tree) to merge similar levels of the predictor variable until the number of children in the proposed split reaches the number that you specify in the MAXBRANCH= option. The $p$-values for the final split determine the variable on which to split.

For continuous predictor variables, CHAID chooses the best single split until the number of children in the proposed split reaches the value that you specify in the MAXBRANCH= option.

You can specify the following options:

**ALPHA=** value

specifies the maximum $p$-value for a split to be considered.

By default, ALPHA=0.3.

**BONFERRONI**

requests a Bonferroni adjustment to the $p$-value for a variable after the split has been determined.

By default, no adjustment is made.

For categorical responses only, you can specify the following criteria:

**CHISQUARE < (options) >**

uses a chi-square statistic to split each variable and then uses the $p$-values that correspond to the resulting splits to determine the splitting variable.

You can specify the following options:

**ALPHA=** value

specifies the maximum $p$-value for a split to be considered.

By default, ALPHA=0.3.

**BONFERRONI**

requests a Bonferroni adjustment to the $p$-value for a variable after the split has been determined.

By default, no adjustment is made.

**ENTROPY < option >**

**GAIN < option >**

uses the gain in information (decrease in entropy) to split each variable and then to determine the split.

You can specify the following option:

**MINENTROPY=** number

**MINGAIN=** number

specifies the minimum gain value to validate a split.
GINI
uses the decrease in the Gini index to split each variable and then to determine the split.

IGR
uses the entropy metric to split each variable and then uses the information gain ratio to determine the split.

The default criterion for categorical responses is IGR.

For continuous responses only, you can specify the following criteria:

FTEST <[options]>
uses an F statistic to split each variable and then uses the resulting p-value to determine the split variable.

You can specify the following options:

ALPHA=value
specifies the maximum p-value for a split to be considered.

By default, ALPHA=0.3.

BONFERRONI
requests a Bonferroni adjustment to the p-value for a variable after the split has been determined.

By default, no adjustment is made.

RSS
VARIANCE
uses the change in response variance to split each variable and then to determine the split.

The default criterion for continuous responses is RSS.

The value of criterion can be tuned with the AUTOTUNE statement.

MODEL Statement

MODEL response = variable;

The MODEL statement causes PROC TREESPLIT to create a tree model by using response as the response variable and one or more variables as predictors. By default, variables are treated as a continuous predictors if they are numeric variables, or as categorical variables if they also appear in a CLASS statement.

NOTE: Specifying a character variable in a MODEL statement without previously declaring it in a CLASS statement results in an error.

OUTPUT Statement

OUTPUT OUT=CAS-libref.data-table <option>;

The OUTPUT statement creates an output data table that contains the results of PROC TREESPLIT.

You must specify the following argument:
OUT=CAS-libref.data-table
names the output data table for PROC TREESPLIT to use. You must specify this option before any other options. CAS-libref.data-table is a two-level name, where

CAS-libref refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to where the data table is to be stored, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about CAS-libref, see the section “Using CAS Sessions and CAS Engine Librefs” on page 1003.

data-table specifies the name of the output data table.

You can also specify the following options:

COPYVAR=variable
COPYVARS=(variables)
lists one or more variables from the input data table to be transferred to the output data table.

ROLE< =name >
generates a numeric variable that indicates the role played by each observation in fitting the model. By default, the variable is named _ROLE_. You can add an optional =name to change the name of this generated variable. For each observation, the interpretation of this variable is shown in Table 21.4.

<table>
<thead>
<tr>
<th>Value</th>
<th>Observation Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Not used</td>
</tr>
<tr>
<td>1</td>
<td>Training</td>
</tr>
<tr>
<td>2</td>
<td>Validation</td>
</tr>
<tr>
<td>3</td>
<td>Testing</td>
</tr>
</tbody>
</table>

If you do not partition the input data by using a PARTITION statement, then the value of the role variable is 1 for all observations.

PARTITION Statement

PARTITION partition-option ;

The PARTITION statement specifies how observations in the input data set are logically partitioned into disjoint subsets for model training, validation, and testing. For more information, see the section “Using Validation and Test Data” on page 80 in Chapter 3, “Shared Concepts.” Either you can designate a variable in the input data table and a set of formatted values of that variable to determine the role of each observation, or you can specify proportions to use for randomly assigning observations to each role.

You must specify exactly one of the following partition-options:
FRACTION(<TEST=fraction> <VALIDATE=fraction> <SEED=number>)
randomly assigns specified proportions of the observations in the input data table to the roles. You
specify the proportions for testing and validation by using the TEST= and VALIDATE= suboptions.
If you specify both the TEST= and VALIDATE= suboptions, then the sum of the specified fractions
must be less than 1 and the remaining fraction of the observations are assigned to the training role. The
SEED= option specifies an integer that is used to start the pseudorandom number generator for random
partitioning of data for training, testing, and validation. If you do not specify SEED=number or if
number is less than or equal to 0, the seed is generated by reading the time of day from the computer’s
clock.

ROLE=variable (<TEST=value> <TRAIN=value> <VALIDATE=value>)
ROLEVAR=variable (<TEST=value> <TRAIN=value> <VALIDATE=value>)
names the variable in the input data table whose values are used to assign roles to each observation.
This variable cannot also appear as an analysis variable in other statements or options. The TEST=,
TRAIN=, and VALIDATE= suboptions specify the formatted values of this variable that are used to
assign observation roles. If you do not specify the TRAIN= suboption, then all observations whose
role is not determined by the TEST= or VALIDATE= suboption are assigned to the training role.

PRUNE Statement

PRUNE <prune-method> (prune-options) ;

The PRUNE statement specifies the pruning method and related options.
You can specify the following prune-methods. The default prune-method is COSTCOMPLEXITY.

C45 <(CONFIDENCE=confidence-level)>
requests C4.5 pruning (Quinlan 1993), which is based on the upper confidence limit for the error rate.
For more information, see the section “Pruning” on page 1039. This pruning method is available only
for classification trees (which have a categorical response). PROC TREESPLIT uses the error rate
from the training data only.

You can specify the following prune-option:

CONFIDENCE=confidence-level
specifies the pruning confidence level, which must be a positive number in the range of [0, 1].
The default confidence level is 0.25.

COSTCOMPLEXITY <(prune-options)>
requests cost-complexity pruning (Breiman et al. 1984; Quinlan 1987; Zhang and Singer 2010). You
can specify this pruning method both for classification trees (which have a categorical response) and
for regression trees (which have a continuous response).

By default, cost-complexity pruning uses k-fold cross validation. If you specify a validation data set by
using the PARTITION statement, PROC TREESPLIT uses the validation data for subtree selection
and does not perform k-fold cross validation.

You can specify one the following prune-options:
**ALPHA=number**  
selects the subtree whose cost-complexity value \( \alpha \) is equal to \( \text{number} \) instead of using \( k \)-fold cross validation. You cannot specify this option with either the KFOLD= or the LEAVES= options.

**KFOLD=number**  
specifies the \( \text{number} \) of folds for \( k \)-fold cross validation. This option is ignored if you specify the PARTITION statement. You can specify this option and the LEAVES=SE value at the same time.

By default, KFOLD=10.

**LEAVES=number | ALL | SE**  
selects the subtree according to one of the following values:

- **number** selects the subtree that has the \( \text{number} \) of leaves, or the subtree whose number of leaves is less than and closest to \( \text{number} \) if no subtree with exactly \( \text{number} \) of leaves exists, instead of using \( k \)-fold cross validation.
- **ALL** selects the largest tree instead of using \( k \)-fold cross validation.
- **SE** selects the subtree by performing \( k \)-fold cross validation and using the Breiman’s 1-SE rule instead of the minimum error rate. You can specify this value and the KFOLD= option at the same time.

**OFF**  
**NONE**  
turns off pruning completely. No pruning is performed, and no pruning plots are generated.

**REDUCEDERROR < (prune-options) >**  
**REP < (prune-options) >**  
requests reduced-error pruning (Quinlan 1986). Reduced-error pruning has two stages: subtree sequence generation and subtree selection. The validation data are used for both stages. The PARTITION statement is required. For more information, see the section “Pruning” on page 1039.

PROC TREESPLIT generates a pruning plot that shows the requested error metric as a function of the number of leaves in the subtree.

You can specify the following **prune-options**:

**LEAVES=number | ALL**  
selects the subtree that has the requested \( \text{number} \) of leaves, or if no subtree with exactly that \( \text{number} \) of leaves is available, selects the subtree whose number of leaves is less than and closest to \( \text{number} \). When LEAVES=ALL, the largest tree is selected.

By default, PRUNE=COSTCOMPLEXITY.
VIICODE Statement

```
VIICODE < options > ;
```

The VIICODE statement writes SAS DATA step code to a file or to a catalog entry. The SAS DATA step code creates new variables on the basis of the detected variable interactions.

You can specify the following `options` in addition to all options in the CODE statement. For more information about the syntax of the CODE statement, see the section “CODE Statement” on page 16 in Chapter 3, “Shared Concepts.”

ADD

requests that the newly created variables be of the form $V + W$.

LIMIT=`number`

specifies the maximum `number` of new variables to create. By default, LIMIT=200.

MISS

requests that the generated code handle missing values.

MULTIPLY

requests that the newly created variables be of the form $V \times W$.

SUBTRACT

requests that the newly created variables be of the form $V - W$.

THRESHOLD=`number`

requests that interactions with an importance less than `number` times the maximum interaction importance be ignored, where `number` must be between 0 and 1. By default, THRESHOLD=0.0001.

If the VII= option is not specified in the PROC TREESPLIT statement, then the VIICODE statement is ignored. For more information about variable interaction importance, see the section “Variable Interaction Importance” on page 1047.

WEIGHT Statement

```
WEIGHT variable ;
```

The `variable` in the WEIGHT statement is used as a weight to perform a weighted analysis of the data. Observations that have nonpositive or missing weights are not included in the analysis. If a WEIGHT statement is not included, all observations that are used in the analysis are assigned a weight of 1.
Building a Decision Tree

Algorithms for building a decision tree use the training data to split the predictor space (the set of all possible combinations of values of the predictor variables) into nonoverlapping regions. These regions correspond to the terminal nodes of the tree, which are also known as leaves.

The splitting is done by recursive partitioning, starting with all the observations, which are represented by the node at the top of the tree. The algorithm splits this parent node into two or more child nodes in such a way that the values (or levels) of the response variable within each child region are as similar as possible. The splitting process is then repeated for each of the child nodes, and the recursion continues until a stopping criterion is satisfied. At that point, the tree is considered to be fully built (grown).

At each step, the split is determined by finding the best predictor variable and the best split value (or possibly a set of split values when the value of the MAXBRANCH= option is greater than 2) that optimize a specified criterion in the response variable across the child nodes to which the parent node is split. For more information about the splitting methods available in the TREESPLIT procedure, see the section “Splitting Criteria” on page 1036.

The best predictor variable and split value (or set of split values) is called the primary splitting rule. You can also request one or more surrogate splitting rules, which are determined in a fashion similar to the primary splitting rule and are applied when the primary splitting rule’s variable is missing. Each nonterminal node has a primary splitting rule and zero or more surrogate splitting rules. For more information about surrogate splitting rules, see the section “Primary and Surrogate Splitting Rules” on page 1038.

Each region, which corresponds to a terminal node, is described by a set of primary and surrogate splitting rules that determine the observations that are assigned to the region. During scoring these rules are applied to determine the predicted values for new observations. For classification trees, the predicted value for a scored observation is the level of the response variable that is most common in the region in which the scored observation exists. For regression trees, the predicted value for a scored observation is the mean of the response variable across training observations values in the region in which the scored observation exists.

Figure 21.8, Figure 21.9, and Figure 21.10 illustrate this process. Figure 21.8 shows a classification tree of depth 2.
Figure 21.8  First Two Splits for the mycas.hmeq Data Table

Classification Tree for BAD

BAD  1  0
All 4,172 observations in the data are initially assigned to node 0 at the top of the tree, which represents the entire predictor space. PROC TREESPLIT splits this space into two nonoverlapping regions, represented by node 1 and node 2. PROC TREESPLIT determines that the predictor variable and its split value that optimize the growing criterion across the child nodes are DebtInc and 43.77156, respectively. This split variable and split value constitute the primary splitting rule for node 0. Observations in which DebtInc < 43.77156 are assigned to node 2, and observations in which DebtInc ≥ 43.77156 are assigned to node 1.

Figure 21.9 presents a scatter plot of the predictor space for the primary splitting rule for node 0. The split of the variable DebtInc divides the predictor space into node 1 and node 2; these nodes are represented by the two rectangular regions, which have different ratios of events to nonevents for the response variable.

Next, the algorithm determines the primary splitting rule for node 2 and splits the region that is represented by node 2 into two nonoverlapping regions, represented by node 3 and node 4. PROC TREESPLIT chooses the variable Delinq and its values to optimize the growing criterion. Observations in which values of the categorical predictor variable Delinq are equal to 0, 1, 2, 3, or 4 are assigned to node 3, and observations in which the values of Delinq are equal to 5, 6, 7, 8, or 10 are assigned to node 4.

Figure 21.10 presents a scatter plot of the predictor space for the primary splitting rule of node 2. The split of the variable Delinq further divides the predictor space of node 2 into two regions: node 3 and node 4, which have different ratios of events to nonevents for the response variable.

Node 3 has a very high proportion of observations for which Bad is equal to 0. In contrast, Bad is equal to 1 for almost all the observations in node 4.

Note that several observations have the same values for Debtinc or Delinq, so the scatter plot gives the perception of fewer observations than there actually are.
This example illustrates recursive binary splitting, in which each parent node is split into two child nodes. By default, the TREESPLIT procedure creates at most two child nodes per split. You can use the MAXBRANCH= option to specify the maximum possible number of child nodes you want per split of a parent node. Based on the splitting criterion you specify, PROC TREESPLIT determines the best number of child nodes up to the value of the MAXBRANCH= option.

### Splitting Criteria

The goal of recursive partitioning, as described in the section “Building a Decision Tree” on page 1033, is to subdivide the predictor space in such a way that the values of the response variable for the observations within a terminal node are similar as possible. The TREESPLIT procedure provides two types of criteria for splitting a parent node: criteria that maximize a decrease in node impurity, as defined by an impurity function, and criteria that are defined by a statistical test. You can specify the criterion in the GROW statement. If you do not specify a criterion in the GROW statement, the entropy criterion is used for classification trees and the RSS criterion is used for regression trees.

### Criteria Based on Impurity

The entropy, Gini index, and RSS criteria decrease impurity. The impurity of a parent node \( \tau \) is defined as \( i(\tau) \), a nonnegative number that is equal to zero for a pure node (a node in which all the observations have the same value of the response variable). Nodes in which the observations have very different values of the response variable have a large impurity.
The TREESPLIT procedure selects the best split variable and the best split value to produce the highest reduction in impurity,

\[ \Delta i(s, \tau) = i(\tau) - \sum_{b=1}^{B} p(\tau_b|\tau)i(\tau_b) \]

where \( \tau_b \) denotes the \( b \)th child node, \( p(\tau_b|\tau) \) is the proportion of observations in \( \tau \) that are assigned to \( \tau_b \), and \( B \) is the number of branches after splitting \( \tau \).

**Impurity Reduction Criteria for Classification Trees**

The following impurity reduction criteria available for classification trees are based on different impurity functions \( i(\tau) \):

- Entropy criterion: The entropy impurity of node \( \tau \) is defined as

\[ i(\tau) = -\sum_{j=1}^{J} p_j \log_2 p_j \]

where \( p_j \) is the proportion of observations that have the \( j \)th response value.

- Gini index criterion: This criterion defines \( i(\tau) \) as the Gini index that corresponds to the ASE of a class response and is given by

\[ i(\tau) = -\sum_{j=1}^{J} p_j^2 \]

For more information, see Hastie, Tibshirani, and Friedman (2009).

**Impurity Reduction Criterion for Regression Trees**

Only one impurity reduction criterion, the RSS criterion, is available for regression trees. This criterion, also called the ANOVA criterion, defines \( i(\tau) \) as the residual sum of squares,

\[ i(\tau) = \frac{1}{N(\tau)} \sum_{i=1}^{N(\tau)} (Y_i - \bar{Y})^2 \]

where \( N(\tau) \) is the number of observations in \( \tau \), \( Y_i \) is the response value of observation \( i \), and \( \bar{Y} \) is the average response of the observations in \( \tau \).

**Criteria Based on Statistical Test**

The chi-square, \( F \) test, and CHAID criteria are defined by statistical tests. These criteria calculate the worth of a split by testing for a significant difference in the response variable across the branches that are defined by a split. The worth is defined as \(-\log(p)\), where \( p \) is the \( p \)-value of the test. You can adjust the \( p \)-values for these criteria by specifying the BONFERRONI option in the GROW statement.
**Statistical Criterion for Classification Trees**

In the chi-square criterion for categorical response variables, the worth is based on the $p$-value for the Pearson chi-square test, which compares the frequencies of the levels of the response across the child nodes.

**Statistical Criterion for Regression Trees**

In the $F$-test criterion for continuous response variables, the worth is based on the $F$ test for the null hypothesis that the means of the response values are identical across the child nodes. The test statistic is

$$F = \frac{SS_{between}/(B - 1)}{SS_{within}/(N(\tau) - B)}$$

where

$$SS_{between} = \sum_{b=1}^{B} N(\tau_b)(\overline{Y}(\tau_b) - \overline{Y}(\tau))^2$$

$$SS_{within} = \sum_{b=1}^{B} \sum_{i=1}^{N(\tau_b)} (Y_{bi} - \overline{Y}(\tau_b))^2$$

**Statistical Criterion for Both Classification and Regression Trees**

Available for both categorical and continuous response variables, CHAID, first described by Kass (1980), is an approach that regards every possible split as representing a test. CHAID tests the hypothesis of no association between the values of the response (target) variable and the branches of a node. The Bonferroni adjusted probability is defined as $\alpha m$, where $\alpha$ is the significance level of a test and $m$ is the number of independent tests.

**Primary and Surrogate Splitting Rules**

During training, PROC TREESPLIT creates a primary splitting rule for each nonterminal node by examining all the node’s observations’ variables and their values and calculating which variable and which of its values to split on such that the observations within each resulting child node are as similar as possible. If you specify a value, $x$, for the NSURROGATES= option, then PROC TREESPLIT creates $x$ surrogate rules in a similar fashion, each time using a different variable that has a similar predictive ability. The TREESPLIT procedure uses the method of Breiman et al. (1984) to determine surrogate rules. Surrogate rules are ordered by agreement, which is measured by the proportion of observations that have a nonmissing value in the rule’s predictor variable that two rules assign to the same branch. For example, if the primary splitting rule assigns 100 observations to a child node, surrogate rule A assigns 40 observations to the same child node, and surrogate rule B assigns 50 observations to the same child node, then surrogate rule B is applied before surrogate rule A. Surrogate rules enable you to make better use of the data.
During scoring, the primary and surrogate splitting rules are applied as follows:

1. The primary splitting rule is applied if the primary rule’s variable is not missing; otherwise,
2. The surrogate rule that has the highest agreement with the primary rule is applied if the first surrogate rule’s variable is not missing; otherwise,
3. Each subsequent splitting rule (ordered by agreement) is applied if the preceding rule has a missing value.
4. If all of the surrogate rules’ variables are missing, then the default splitting rule (as specified in the ASSIGNMISSING= option) is used. The default splitting rule enables all data to be scored even if the primary rule and all surrogate rules cannot be used on a particular observation.

Table 21.5 shows an example set of rules for a particular node in a decision tree where \(X\), \(Y\), and \(Z\) are three continuous predictors.

<table>
<thead>
<tr>
<th>Rule</th>
<th>Assign to Branch 1</th>
<th>Assign to Branch 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary</td>
<td>(X \geq 1)</td>
<td>(X &lt; 1)</td>
</tr>
<tr>
<td>Surrogate 1</td>
<td>(X) is missing, (Y &lt; 0)</td>
<td>(X) is missing, (Y \geq 0)</td>
</tr>
<tr>
<td>Surrogate 2</td>
<td>(X) and (Y) are missing, (Z \geq 100)</td>
<td>(X) and (Y) are missing, (Z &lt; 100)</td>
</tr>
<tr>
<td>Default</td>
<td>None</td>
<td>(X, Y,) and (Z) are missing</td>
</tr>
</tbody>
</table>

By default, NSURROGATES=0. If a variable is used as a surrogate, you can see exactly how it is used in the SAS DATA step code that is generated when you specify the CODE statement.

**Pruning**

The TREESPLIT procedure creates a classification or regression tree by first growing a tree as described in the section “Splitting Criteria” on page 1036. This usually results in a large tree that provides a good fit to the training data. The problem with this tree is its potential for overfitting the data: the tree can be tailored too specifically to the training data and not generalize well to new data. The solution is to find a smaller subtree that results in a low error rate on the validation data.

It is often prohibitively expensive to evaluate the error on all possible subtrees of the full tree. A more practical strategy is to focus on a sequence of nested trees that are obtained by successively pruning leaves from the tree. Figure 21.11 shows an example of pruning in which node 3’s leaves (terminal nodes 4 and 5) are removed to create a nested subtree of the full tree. In the nested subtree, node 3 is now a leaf that contains all the observations that were previously in nodes 4 and 5. This process is repeated until only the root node remains.
Many different methods have been proposed for pruning in this manner. These methods address both how to select which nodes to prune to create the sequence of subtrees and how then to select the optimal subtree from this sequence as the final tree. You can use the PRUNE statement in PROC TREESPLIT to specify which pruning method to apply and related options. Several well-known pruning methods, described in this section, are available, and you can override the final selected tree based on your preferences or domain knowledge.

### Cost-Complexity Pruning

Cost-complexity pruning is a widely used pruning method that was originally proposed by Breiman et al. (1984). You can request cost-complexity pruning for either a categorical or continuous response variable by specifying the following statement:

```
prune costcomplexity;
```

The cost-complexity pruning method helps prevent overfitting by making a trade-off between the complexity (size) of a tree and the error rate. Thus large trees with a low error rate are penalized in favor of smaller trees. The cost complexity of a tree $T$ is defined as

$$CC(T) = R(T) + \alpha |T|$$

where $R(T)$ represents $T$’s error rate, $|T|$ represents the number of leaves on $T$, and the complexity parameter $\alpha$ represents the cost of each leaf. For a categorical response variable, the misclassification rate is used for the error rate, $R(T)$; for a continuous response variable, the residual sum of squares (RSS), also called the sum of square errors (SSE), is used for the error rate. Only the training data are used to evaluate cost complexity. Breiman et al. (1984) show that for each value of $\alpha$, there is a subtree of $T$ that minimizes cost complexity. When $\alpha = 0$, this is the full tree, $T_0$. As $\alpha$ increases, the corresponding subtree becomes progressively smaller, and the subtrees are in fact nested. Then, at some value of $\alpha$, the root node has the minimal cost complexity for any $\alpha$ greater than or equal to that value. Because there are a finite number of
possible subtrees, each subtree corresponds to an interval of values of $\alpha$; that is,

- $[0, \alpha_1)$ = interval where $T_0$ (the full tree) has minimal cost complexity
- $[\alpha_1, \alpha_2)$ = interval where $T_1$ has minimal cost complexity
- $\vdots$
- $[\alpha_m, \infty)$ = interval where $T_m$ (the root node) has minimal cost complexity

PROC TREESPLIT uses weakest-link pruning, as described by Breiman et al. (1984), to create the sequence of $\alpha_1, \ldots, \alpha_m$ values and the corresponding sequence of nested subtrees, $T_1, \ldots, T_m$.

Finding the optimal subtree from this sequence is then a question of determining the optimal value of the complexity parameter $\alpha$. This is performed either by using the validation partition (when you use the PARTITION statement to reserve a validation holdout sample) or by using cross validation. In the first case, the subtree in the pruning sequence that has the lowest validation error rate is selected as the final tree. When there is no validation partition, $k$-fold cross validation can be applied to cost-complexity pruning to select a subtree that generalizes well and does not overfit the training data (Breiman et al. 1984; Zhang and Singer 2010). The algorithm proceeds as follows after creating the sequence of subtrees and $\alpha$ values by using the entire set of training data as described earlier:

1. Randomly divide the training observations into $k$ approximately equal-sized parts, or folds.
2. Define a sequence of $\beta$ values as the geometric mean of the endpoints of the $[\alpha_i, \alpha_{i+1})$ intervals (that is, $\beta_i = \sqrt{\alpha_i \times \alpha_{i+1}}$) to represent the intervals.
3. For each of the $k$ folds, hold out the current fold for validation and use the remaining $k - 1$ folds for the training data in the following steps:
   a) Grow a tree as is done using the full training data set with the same splitting criterion.
   b) Using the $\beta_1, \ldots, \beta_m$ values that are calculated in step 2, create a sequence of subtrees for each $\beta_i$ as described in the pruning steps given earlier, but now using $\beta_i$ as a fixed value for $\alpha$ and minimizing the cost complexity, $CC(T)$, to select a subtree at each pruning step.
   c) For each $\beta_i$, set $T_{ij}$ to be the subtree that has the minimum cost complexity from the sequence for the $j$th fold.
   d) Calculate the error for each $T_{ij}$ by using the current ($j$th) fold (the one omitted from the training).
4. Now the error rate can be averaged across folds, and the $\beta_i$ that has the smallest average error is selected. The tree $T_i$ from pruning the complete training data that corresponds to the selected $\beta_i$ is the final selected subtree.

Because the $k$-fold cross validation is done sequentially, the process can take a long time for deep trees or for large amounts of data. Consider using the PARTITION statement for pruning large amounts of data.

The TREESPLIT procedure provides two plots that you can use to tune and evaluate the pruning process: the cost-complexity analysis plot and the cost-complexity pruning plot.

When performing cost-complexity pruning with cross validation (that is, no PARTITION statement is specified), it is recommended that you examine the cost-complexity analysis plot that is created by default. This plot displays the error as a function of the complexity parameters $\beta_i$, and it uses a vertical reference line.
to indicate the $\beta_i$ that minimizes the error. You can use this plot to examine alternative choices for $\beta_i$. For example, you might prefer to select a smaller tree that has only a slightly higher error rate. You can specify LEAVES=SE to use the 1-SE rule developed by Breiman et al. (1984).

When you specify validation data by using the PARTITION statement, the cost-complexity pruning plot displays the error rate $R(T)$ as a function of the number of leaves $|T|$ for both the training and validation data. This plot indicates the final selected tree, the tree with the minimum $R(T)$ for the validation data, by using a vertical reference line. Like the cost-complexity analysis plot that is produced when you perform cross validation, this plot can help you identify a smaller tree that has only a slightly higher validation error rate. For an example of this plot, see Output 21.1.5 in “Example 21.1: Creating a Binary Classification Tree with Validation Data” on page 1057.

You can use the LEAVES= option in the PRUNE statement to select a tree that has a specified number of leaves. Alternatively, you can use the ALPHA= option in the PRUNE statement to select a tree that has a specified value of the cost-complexity value $\alpha$.

### C4.5 Pruning

Quinlan (1987) first introduced pessimistic pruning as a method of pruning classification trees. In this method, the estimate of the true error rate is increased by using a statistical correction in order to prevent overfitting. C4.5 pruning (Quinlan 1993) evolved from pessimistic pruning to employ an even more pessimistic (that is, higher) estimate of the true error rate. An advantage of methods such as pessimistic and C4.5 pruning is that they enable you to use all the data for training instead of requiring a holdout sample. In C4.5 pruning, the upper confidence limit of the true error rate based on the binomial distribution is used to estimate the error rate. PROC TREESPLIT implements a C4.5 algorithm variant that uses the beta distribution in place of the binomial distribution to estimate the upper confidence limit. This pruning method is available only for categorical response variables and it uses only training data for tree pruning. It is implemented by the following statement:

```
prune C45;
```

The C4.5 pruning method follows these steps:

1. Grow a tree from the training data table, and call this full, unpruned tree $T_0$.
2. Set $i = 0$, and do the following until $T_i$ is only the root node:
   
   a) For each leaf (terminal node) in the tree $T_i$, solve the following equation for $p_l$ (which is the adjusted prediction error rate for leaf $l$):

   $$\alpha = 1 - \frac{\Gamma(N_l + 1)}{\Gamma(F_l + 1) \Gamma(N_l - F_l)} \int_0^{p_l} v^{F_l} (1 - v)^{N_l - F_l - 1} dv$$

   Here the confidence level $\alpha$ is the value of the CONFIDENCE= option in the PRUNE statement, $F_l$ is the number of failures (misclassified observations) at leaf $l$, $N_l$ is the number of observations at leaf $l$, and the function $\Gamma(x)$ is defined as

   $$\Gamma(x) = \int_0^{\infty} v^{x-1} e^{-v} dv$$
b) Given these values of $p_l$, use the following formula for the prediction error $E_i$ of tree $T_i$:

$$E_i = \sum_{l \in T_i} N_l p_l$$

c) For each node in tree $T_i$ that has only leaves as children, create a candidate subtree by pruning those leaves.

d) For each candidate subtree, use the equations from steps 2 and 3 to calculate its prediction error. Then select the candidate subtree that has the largest decrease (or smallest increase) in prediction error, $E_i$. Let this be the next subtree in the sequence, $T_{i+1}$.

e) Set $i = i + 1$

3. Calculate the change in error between each pair of consecutive subtrees, $\Delta_i = E_i - E_{i-1}$ for each $i = 1, \ldots, m$.

4. Find the smallest integer $j$ such that $\Delta_j > 0$.

5. Select the subtree $T_{j-1}$ as the final subtree.

**Reduced-Error Pruning**

Quinlan’s reduced-error pruning (1987) performs pruning and subtree selection based on minimizing the error rate in the validation partition at each pruning step and then in the overall subtree sequence. The error rate is based on the misclassification rate for a categorical response variable and on the ASE for a continuous response. The following PRUNE statement implements reduced-error pruning:

```
prune reducederror;
```

Reduced-error pruning creates a sequence of subtrees from the largest tree, $T_0$, to the root node, $T_m$. The subtree that has the smallest validation error is then selected as the final subtree. Pruning could be stopped as soon as the error begins to increase in the validation data as originally described by Quinlan; continuing to prune to create a subtree sequence back to the root node enables you to select a smaller tree that still has an acceptable error rate, as discussed in the next section.

**User Specification of Subtree**

You might want to select a different tree from the one selected by default when you use cost-complexity or reduced-error pruning to create the sequence of subtrees. For example, you might have a subtree that has a slightly larger error but is smaller, and thus simpler, than the subtree that has the minimum error according to reduced-error pruning. You can override the selected subtree and instead select the subtree that has $n$ leaves and was created by cost-complexity or reduced-error pruning, where $n$ is specified in the LEAVES= option in the PRUNE statement. In addition, if you are using cost-complexity pruning, you can override the selected subtree by using the ALPHA= option in the PRUNE statement.

Alternatively, you might want to select the largest tree that is created in one of the following ways:

- Specify LEAVES=ALL in the PRUNE statement to still see the statistics for the sequence of subtrees that are created according to the specified pruning error measure, even though the largest (unpruned) tree is selected as the final subtree.

- Specify the following statement to select the largest tree with no pruning performed:

```
prune none;

Statistics are not calculated and plots are not created for a sequence of subtrees.

---

**Scoring**

After you create a tree model, you can apply it to training or test data for model assessment or to new data for making predictions. The process of applying a model to a data table is called *scoring*. You can score data as described in the following sections.

**Scoring the Input Data Table**

Usually, the purpose of scoring training data is to diagnose the model. The training data table is the data table that you specify with the `DATA=` option. To score the training data, use the `OUTPUT` statement to create an output data table that contains one observation for each observation in the training data. You can specify the output data table by using the `OUT=` option in the `OUTPUT` statement.

In the following example, the input data table (`mycas.hmeq`) is scored after the tree model has been created:

```sas
proc treesplit data=mycas.hmeq;
    class Bad Delinq Derog Job Ninq Reason;
    model Bad = Delinq Derog Job Ninq Reason;
    output out=mycas.scored;
run;
```

For classification trees, the scored data table also contains one new variable for each level of the response variable. These new variables have the prefix “`P_`”; for all observations in the same leaf, these new variables represent the proportion of the training observations in that leaf that have that particular response level. For example, if the name of the categorical response variable is `Color` and it has two levels, `'Blue'` and `'Green'`, then the scored data table contains the variable `P_ColorBlue` (which provides the proportion of training data in this leaf that have the response level `'Blue'`) and the variable `P_ColorGreen` (which provides the proportion of training data in this leaf that have the response level `'Green'`).

For regression trees, the scored data table contains exactly one new variable with the prefix “`P_`”, which represents the average value of the response variable for all observations in the same leaf. For example, if the name of the continuous response variable is `logSalary`, then the scored data table contains one new variable, `P_logSalary`, which represents the average value of the response variable `logSalary` in the training data for observations in the same leaf.

**Scoring Using DATA Step Code**

You can use the `CODE` statement to generate SAS DATA step code that you can use to score new data. The following example uses PROC TREESPLIT to produce SAS DATA step code:

```sas
proc treesplit data=mycas.hmeq;
    class Bad Delinq Derog Job Ninq Reason;
    model Bad = Delinq Derog Job Ninq Reason;
    code file="treesplit_data_step.sas";
run;
```
Measures of Model Fit

The TREESPLIT procedure measures model fit based on the misclassification rate for classification trees or the average square error for regression trees, as follows:

- The misclassification rate is calculated from the number of incorrectly predicted observations:

\[
\text{Misc} = \frac{1}{N_0} \sum \begin{cases} 
0 & \text{if prediction is correct} \\
1 & \text{otherwise}
\end{cases}
\]

- The average square error (ASE) for regression trees is defined as

\[
\text{ASE} = \frac{\text{RSS}}{N_0}
\]

Variable Importance

A training data table can contain a large number of predictor variables, some of which are useful for predicting the response variable and others of which are not. You can use the TREESPLIT procedure to select the most useful predictor variables on the basis of variable importance. (See “Example 21.3: Assessing Variable Importance” on page 1067.) Variable importance is an indication of which predictor variables are the most useful for predicting the response variable.

The most important variables might not be the ones near the top of the tree. The TREESPLIT procedure implements several methods for computing variable importance, which are described in the following subsections. By default, PROC TREESPLIT calculates the variable importance by three methods (count-based, change in the residual sum of square errors, and relative importance), and outputs the results from these variable importance methods in one table. You can request that the TREESPLIT procedure also calculate the variable importance by random branch assignment (RBA) by specifying the RBAIMP option.

Count-Based Importance Method

PROC TREESPLIT calculates two count-based variable importance methods:

- Count-based variable importance simply counts the number of times in the tree that a particular variable is used in a split.

- If the NSURROGATES= option is used, then surrogate-count-based variable importance also counts the number of times that a variable is used in a surrogate splitting rule.
Chapter 21: The TREESPLIT Procedure

Residual Sum of Squares (RSS) Importance Method

This method measures variable importance based on the change of RSS when a split is found at a node. The change for variable \( v \) is

\[
\Delta_d^v = RSS_d - \sum_i RSS_d^i
\]

where

- \( d \) denotes the node
- \( i \) denotes the index of a child that this node includes
- \( RSS_d \) is the RSS if the node is treated as a leaf
- \( RSS_d^i \) is the RSS of the node after it has been split

If the change in RSS is negative (which is possible when you measure the validation set), then the change is set to 0.

The residual sum of squares for regression trees is defined as

\[
RSS = \sum_\lambda \sum_{i \in \lambda} (y_i - \hat{y}_{\lambda}^T)^2
\]

where

- \( i \) is an observation on leaf \( \lambda \)
- \( y_i \) is the predicted value of the response variable of observation \( i \)
- \( \hat{y}_{\lambda}^T \) is the actual value of the response variable on leaf \( \lambda \)

The residual sum of squares for classification trees is defined as

\[
RSS = \sum_\lambda \sum_{\Phi} N_{\phi}^\lambda \left[ \sum_{\tau \neq \Phi} \left( P_{\tau}^\lambda \right)^2 + \left( 1 - P_{\Phi}^\lambda \right)^2 \right]
\]

where

- \( \Phi \) is the actual response level
- \( N_{\Phi}^\lambda \) is the number of observations on leaf \( \lambda \) that have response level \( \Phi \)
- \( P_{\tau}^\lambda \) is the posterior probability for the response level \( \tau \) on leaf \( \lambda \)
- \( P_{\Phi}^\lambda \) is the posterior probability for the actual response level \( \Phi \) on leaf \( \lambda \)
If surrogate rules are in effect, they are also credited with a portion of the change in RSS. The credit is proportional to the agreement between the primary and surrogate splitting rules at the node. The agreement at node $d$ (denoted by $\kappa_d$) is defined as

$$\kappa_d = \sum_i \frac{N_i}{N_d}$$

where $N_d$ is the number of nonmissing observations and $N_i$ is the number of observations that were assigned to $i$ by both the primary and surrogate rules.

The change in RSS from the surrogate rules for variable $v$ is defined as

$$\Delta^v_d = \kappa_d \left( \text{RSS}_d - \sum_i \text{RSS}_i^d \right)$$

The RSS-based importance for a single variable, $v$, is then defined as

$$\sum_{d=1}^{D} \Delta^v_d$$

where $D$ is the total number of nodes in which $v$ was used as the splitting variable.

**Relative Variable Importance Method**

The relative variable importance metric is a number between 0 and 1, which is calculated in two steps:

1. PROC TREESPLIT finds the maximum RSS-based variable importance.
2. For each variable, PROC TREESPLIT calculates the relative variable importance as the RSS-based importance of this variable divided by the maximum RSS-based importance among all the variables.

The RSS and relative importance are calculated from the validation data. If no validation data exist, they are calculated instead from the training data.

**Variable Interaction Importance**

In some cases, interactions of variables are of more interest than a single variable. When you specify the VII= option in the PROC TREESPLIT statement, the procedure computes variable interaction importance as follows.

The two-way interaction importance for variables $v$ and $w$ is

$$\sum_{d,e} \frac{\Delta^v_d + \Delta^w_e}{\Delta^v_d}$$

where

- $d$ and $e$ denote nodes, with node $d$ being the parent of node $e$
- $v$ and $w$ are variables, with $v$ being the splitting variable for node $d$ and $w$ being the splitting variable for node $e$
• this summation is across all parent-child node combinations with respect to the variables v and w

The three-way interaction importance for variables v, w, and x is

\[ \sum_{d,e,f} \frac{\Delta_d^v + \Delta_e^w + \Delta_f^x}{\Delta_d^v + \Delta_e^w} \]

where

• d, e, and f denote nodes, with node d being the parent of node e, which is the parent of node f
• v, w, and x are variables, with v being the splitting variable for node d, w being the splitting variable for node e, and x being the splitting variable for node f
• this summation is across all parent-child-grandchild node combinations with respect to the variables v, w, and x

When comparing variable importance values, you should only compare a two-way interaction with another two-way interaction, or a three-way interaction with another three-way interaction. Comparing a two-way interaction to a three-way interaction is not meaningful, because the values for the importance are not on the same scale.

In addition to calculating the variable interaction importance, you can also generate SAS DATA step code to create interaction variables and save the code to a file by using the VIICODE statement. When you run the generated SAS DATA step code on the original data or on new data, interaction variables are created. Created interaction variables are of the form \( W \odot V, W \odot V, \) or \( W \odot V, \) where the values used in the calculations come from the variables whose interactions are determined to be important. The values used are as follows:

• For continuous variables, the raw value of the variable is used.
• For categorical variables, the proportion of observations in the training set that is assigned to a specific branch is used.

Random Branch Assignment Importance Method

The random branch assignment (RBA) method computes the importance of an input variable v by comparing how well the data fit the predictions before and after they are modified. Neville and Tan (2014) motivate and introduce the RBA method of variable importance. To modify the predictions, the TREESPLIT procedure replaces all splitting rules that use variable v by a rule that randomly assigns an observation to a branch. The probability of assigning an observation to a branch is proportional to the number of observations that are assigned to the branch in the current data. The current data are the training data when RBA is computed during training. Otherwise, the current data are those being scored on an existing model.

The RBA importance can be expressed mathematically as

\[ I_{RBA}(v) \propto \sum_{i=1}^{n} \text{Loss}(y_i, \hat{y}_i) - \sum_{i=1}^{n} \text{Loss}(y_i, \hat{y}_i) \]
Hyperparameter Tuning

The quality of the predictive model that a machine learning algorithm creates depends on the values for various options that govern the training process; these options are called hyperparameters. The default values of these hyperparameters might not be suitable for all applications. In order to reduce the manual effort in adjusting these hyperparameters, you can use the AUTOTUNE statement to identify the best settings for them.

To tune hyperparameters, the AUTOTUNE statement directs the procedure to engage a search method (tuner) that searches for the best possible combination of values of these hyperparameters while trying to minimize or maximize an objective function. The objective function is a validation error or accuracy estimate—for example, MISC (misclassification error) for nominal targets or ASE (average square error) for interval targets. The tuning process involves multiple evaluations of the objective function (one for each model hyperparameter configuration it evaluates) and might include multiple iterations, depending on the specified search method. The tuning process also requires data partitioning in order to avoid overfitting to training data; models are trained on a training partition and validated on a validation partition to obtain the objective function value for each chosen hyperparameter configuration. The options for data partitioning are described in the next section, followed by descriptions of the default hyperparameter tuning process and determination of sizing for parallel tuning.

Data Partitioning

Each evaluation of the objective function can consist of one or several training and scoring (validation) executions, depending on whether single-partition validation or k-fold cross validation is used. In both cases, the tuner partitions the data unless the PARTITION statement is used (in this case, the tuner does not partition the training data, but instead uses the data roles that are specified in the PARTITION statement for training and validation).

Single-Partition Validation

By default, the tuner uses single-partition validation. In this process, the tuner partitions all the data into two subsets: one subset for model training and one subset for model validation. For each newly generated configuration of hyperparameters, a new model is trained on the training subset and then the validation subset is scored using the trained model to find the resulting objective function value.

The AUTOTUNE statement calls the sampling action set, which creates the training and validation partitions. The stratified action in the sampling action set is used for a target of nominal type (if all target levels can be included in both the training and validation partitions), and the srs action in the sampling action set is used for a target of interval type and for cases in which stratified sampling is not possible. By default, a validation partition of 30% is used and the remaining 70% is used for model training. The validation partition size can be adjusted by specifying the FRACTION= option in the AUTOTUNE statement. For more information about the sampling action set, see Chapter 17, “Sampling and Partitioning Action Set” (SAS Visual Statistics: Programming Guide).
Cross Validation
If $k$-fold cross validation is specified, the tuner partitions all the data into $k$ subsets (folds). For each fold, a new model is trained on all folds except the selected (holdout) fold (that is, it is trained on $k-1$ folds) and then validated using the selected (holdout) fold. The objective function value is averaged over the set of fold validation executions to obtain a single error estimate value. Cross validation is specified using the KFOLD= option in the AUTOTUNE statement.

For efficiency, the cross validation process might be terminated before all $k$ folds are evaluated. Cross validation is terminated under the following conditions: the validation score is 2 times worse than the current best score after the first fold, the validation score is 1.75 times worse than the current best score after two folds, or the validation score is 1.5 times worse than the current best score after three or more folds.

Hyperparameter Tuning Process
By default, the tuner chooses the model hyperparameter configurations to evaluate by using a hybrid optimization strategy that is based on a genetic algorithm (GA), which applies principles of natural selection and evolution to find an improved configuration. The tuner performs the following steps:

1. A default model configuration (default values of select model hyperparameters) is evaluated first and designated as Iteration 0. The objective function value is obtained by using either single-partition validation or $k$-fold cross validation and then recorded for comparison.

2. An initial set of hyperparameter configurations, also called a “population,” is generated using a technique called Latin hypercube sampling (LHS). Each configuration of hyperparameters in the Latin hypercube sample is evaluated, and the objective function value for each is again recorded for comparison. This set becomes Iteration 1. All or a portion of the sample configurations can be evaluated in parallel, as described in the section “Determining the Number of Parallel Evaluations” on page 1051.

3. The best model configurations from the Iteration 1 are used to generate the next population of model configurations, Iteration 2, which are then evaluated, in parallel when possible (see the section “Determining the Number of Parallel Evaluations” on page 1051). This process is repeated for the remaining iterations, as long as the maximum number of evaluations or the maximum time has not been reached. The default number of iterations is 5, and the default maximum time is 36,000 seconds (10 hours).

4. The best model configuration (the one that minimizes or maximizes the objective function) is reevaluated by executing a single model training and scoring, and information about the model training and scoring for this configuration is returned.

5. All evaluated model configurations are ranked, and the hyperparameter and objective function values of the top 10 configurations are returned in the TunerResults table.

You can specify four alternative search methods: Bayesian, random, LHS, and grid. The preceding steps are performed differently, depending on the search method that you use:

- For the Bayesian search method, the preceding steps are performed in the same sequence, with the following changes. Instead of using genetic operations to generate the new combinations of hyperparameter values, an internal surrogate model is created after the first iteration. This surrogate
model is then explored for potential candidate configurations, and the new set of configurations is created and evaluated in parallel. After each iteration, the internal surrogate model is updated with the new data and explored again for new potential configurations.

- For the random, LHS, or grid search method, step 3 in the preceding sequence is eliminated; a single sample of candidate configurations is generated and evaluated in step 2. This single sample of configurations is generated in one of the following ways:
  - randomly, for the random search method
  - by using a Latin hypercube sample, for the LHS search method
  - by creating all possible combinations of hyperparameter levels, for the grid search method. If the VALUELIST= option is specified for any hyperparameter, the levels are taken from that list. Otherwise, the tuner takes the lower bound, the upper bound, and the middle value between the two bounds and uses those values as the grid levels for each hyperparameter. The total number of grid configurations is the product of the numbers of levels for each hyperparameter. If the number of hyperparameters and the number of levels is greater than a very small value, then the total number of combinations can be many hundreds or thousands. If the MAXEVALS= option is specified, the tuner uses it to terminate execution of the grid search method before all grid points are evaluated.

If the tuner had partitioned the data for training and scoring (if the PARTITION statement is not specified), then the final (best) model configuration is trained on all the data and scored on all the data; as a result the final model that is returned and its reported validation score might not match the value that is observed during tuning, when the model was trained to a portion of the data and validated on the remaining validation partition.

**Determining the Number of Parallel Evaluations**

By default, the number of possible parallel evaluations during step 2 and step 3 (if performed) in the section “Hyperparameter Tuning Process” on page 1050 is determined by the population size for the GA or Bayesian search method or by the sample size for the random or LHS sampling method. The maximum number of parallel evaluations is limited by the total number of worker nodes that are connected to the server divided by the number of worker nodes that are used by the parallel sessions that are created. For example, if the server is configured with 100 worker nodes and four workers are used in the sessions, at most 25 parallel evaluations are performed by default. This limit can be overridden by a factor of 2, resulting in up to 50 parallel evaluations in this example scenario.

Specifically, the number of parallel evaluations is determined as follows:

1. If you do not specify the number of workers to use in parallel subsessions by using the NSUBSESSIONWORKERS= option in the AUTOTUNES statement, the number of workers is determined using the size of the data table: NSUBSESSIONWORKERS = 1 + nDataRows * nDataColumns * 2E–8. If determined on the basis of the data table size, the number of workers in each subsession is fairly aggressively set at one node per 50 million values. The efficiency benefits of tuning in parallel generally outweigh the time savings from distributed training if the data can be managed on fewer nodes. For example, if training a single model on four workers takes 1 minute versus 2 minutes on one worker, then training four models in parallel on one worker each requires only 2 minutes, whereas training four sequential models each on four workers would require 4 minutes.
2. The number of possible parallel evaluations is then determined either as one less than the population size for the GA or Bayesian search method, or as the sampling size for the random or LHS search method.

3. The number of parallel evaluations is then limited by the server configuration:
   - In single-machine mode, if the number of possible parallel evaluations is greater than 4 and not specified in the NPARALLEL= option in the AUTOTUNE statement, it is limited to 4. You can override this limit up to a value of 32 by specifying the NPARALLEL= option (the value of the NPARALLEL= option is reduced if it is greater than 32).
   - In distributed mode, the upper limit for the number of parallel evaluations is calculated as \( \frac{W}{n} \), where \( W \) is the number of workers connected to the server and \( n \) is the number of workers in the parallel subsessions. You can override this limit by a factor of 2 by specifying the NPARALLEL= option; the new limit is \( 2 \frac{W}{n} \) (the value of the NPARALLEL= option is reduced if it is greater than \( 2W/n \)).

You can use the AUTOTUNE statement to tune the following options in the PROC TREESPLIT statement:
   - the value of the NUMBIN= option for the number of bins for continuous predictors
   - the value of the MAXDEPTH= option for the maximum depth of the decision tree
   - the criterion on the GROW statement for how the nodes are split

---

**Displayed Output**

The following sections describe the output that PROC TREESPLIT produces. The output is organized into various tables, which are discussed in the order of appearance.

**Model Information**

The “Model Information” table contains the settings of the training parameters. This table also contains some basic information about the resulting decision tree. This table is produced by default.

**Number of Observations**

The “Number of Observations” table contains the number of observations that are read from the input data table and the number of observations that are used in the analysis. When you specify the PARTITION statement, the table also indicates the number of observations that are used in each partition. This table is produced by default.

**CV Statistics**

The “CV Statistics” table contains information about \( k \)-fold cross validated cost complexity pruning, such as the alpha values, the corresponding size of the tree, and the corresponding minimum, average, and maximum error. The table is generated when you specify the CVCC option in the PROC TREESPLIT statement, and you are also using \( k \)-fold cross validated cost complexity pruning.
Tree Performance

The “Tree Performance” table contains statistics that measure the model’s goodness of fit. If you specify the PARTITION statement, the table also indicates these statistics on all partitions of the data.

Variable Importance

The “Variable Importance” table displays variable importance based on residual sum of square errors, which is explained in the section “Residual Sum of Squares (RSS) Importance Method” on page 1046. This table is produced by default.

RBA Variable Importance

The “RBA Variable Importance” table displays variable importance based on the random branch assignment (RBA) method, which is explained in the section “Random Branch Assignment Importance Method” on page 1048. This table is produced by the RBAIMP option in the PROC TREESPLIT statement.

Reduced Error

The “Reduced Error” table contains the pruning results of reduced error pruning that uses validation data. This table is generated when you specify the PRUNINGTABLE option and are using reduced error pruning in the PRUNE statement with validation data.

Cost Complexity

The “Cost Complexity” table contains the pruning results of cost complexity pruning that uses validation data. This table is generated when you specify the PRUNINGTABLE option and are using cost complexity pruning in the PRUNE statement with validation data.

Output CAS Tables

When you specify the OUTPUT statement or the OUTMODEL= option in the PROC TREESPLIT statement to create output tables on your CAS server, the TREESPLIT procedure produces the output data table along with a table that lists the CAS library, the data table name, and the number of rows and columns in that data table.

Tuner Information

The “Tuner Information” table displays the setup values that the tuner uses. This table is produced by the AUTOTUNE statement.

Tuner Summary

The “Tuner Summary” table displays statistics about the tuning process. This table is produced by the AUTOTUNE statement.


**Tuner Timing**

The “Tuner Timing” table displays the total time spent on different tasks while tuning. This table is produced by the AUTOTUNE statement.

**Best Configuration**

The “Best Configuration” table displays the hyperparameters and objective function values for the best configuration. This table is produced by the AUTOTUNE statement.

**Tuner Results**

The “Tuner Results” table displays the values of the hyperparameters, the objective function for the default configuration (Iteration 0), and up to 10 best found configurations. This table is produced by the AUTOTUNE statement.

**Evaluation History**

The “Evaluation History” tables displays the values of the hyperparameters and the objective function for all configurations. This table is produced by the AUTOTUNE statement, either by default or when EVALHISTORY=ALL.

**Predicted Probability Names**

The “Predicted Probability Names” table indicates the names of the created variables in either the score code or the data table that is specified in the OUT= option in the OUTPUT statement. This table is produced when you specify the PRINTTARGET option in the PROC TREESPLIT statement.

**ODS Table Names**

PROC TREESPLIT assigns a name to each table it creates. You can use these names to refer to the table when you use the Output Delivery System (ODS) to select tables and create output data tables. Table 21.6 lists these names.

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>BestConfiguration</td>
<td>Hyperparameters and objective function values for the best configuration</td>
<td>AUTOTUNE</td>
<td>Default</td>
</tr>
<tr>
<td>CostComplexity</td>
<td>Information about the number of leaves and the error for cost-complexity pruning when validation data are used</td>
<td>PROC TREESPLIT</td>
<td>PRUNINGTABLE</td>
</tr>
<tr>
<td>CrossValidationValues</td>
<td>Results for cost-complexity pruning based on cross validation</td>
<td>PROC TREESPLIT</td>
<td>CVCC</td>
</tr>
</tbody>
</table>
Table 21.6  continued

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>EvaluationHistory</td>
<td>Values of the hyperparameters and the objective function for all configurations</td>
<td>AUTOTUNE</td>
<td>Default / EvalHistory=ALL</td>
</tr>
<tr>
<td>ModelInfo</td>
<td>Information about the modeling environment</td>
<td>PROC TREESPLIT</td>
<td>Default</td>
</tr>
<tr>
<td>NObs</td>
<td>Observation information</td>
<td>PROC TREESPLIT</td>
<td>Default</td>
</tr>
<tr>
<td>PredName</td>
<td>Predicted name information for interval targets</td>
<td>PROC TREESPLIT</td>
<td>PRINTTARGET</td>
</tr>
<tr>
<td>PredProbName</td>
<td>Predicted probability name information for nominal targets</td>
<td>PROC TREESPLIT</td>
<td>PRINTTARGET</td>
</tr>
<tr>
<td>PredIntoName</td>
<td>Predicted name information for nominal targets</td>
<td>PROC TREESPLIT</td>
<td>PRINTTARGET</td>
</tr>
<tr>
<td>RBAImportance</td>
<td>Random branch assignment variable importance</td>
<td>PROC TREESPLIT</td>
<td>RBAIMP</td>
</tr>
<tr>
<td>ReducedError</td>
<td>Information about the number of leaves and the error for reduced-error pruning when validation data are used</td>
<td>PROC TREESPLIT</td>
<td>PRUNINGTABLE</td>
</tr>
<tr>
<td>TreePerformance</td>
<td>Fit statistics for the selected tree</td>
<td>PROC TREESPLIT</td>
<td>Default</td>
</tr>
<tr>
<td>TunerInfo</td>
<td>Setup values used by the tuner</td>
<td>AUTOTUNE</td>
<td>Default</td>
</tr>
<tr>
<td>TunerResults</td>
<td>Values of the hyperparameters and the objective function for the default configuration (Iteration 0) and up to 10 best configurations found</td>
<td>AUTOTUNE</td>
<td>Default</td>
</tr>
<tr>
<td>TunerSummary</td>
<td>Statistics about the tuning process</td>
<td>AUTOTUNE</td>
<td>Default</td>
</tr>
<tr>
<td>TunerTiming</td>
<td>Total time spent on different tasks while tuning</td>
<td>AUTOTUNE</td>
<td>Default</td>
</tr>
<tr>
<td>VariableImportance</td>
<td>Variable importance</td>
<td>PROC TREESPLIT</td>
<td>Default</td>
</tr>
</tbody>
</table>

**ODS Graphics**

You can refer to every graph that is produced through ODS Graphics by name. The names of the graphs that PROC TREESPLIT generates are listed in Table 21.7, along with the relevant PLOTS= options.
Table 21.7  Graphs Produced by PROC TREESPLIT

<table>
<thead>
<tr>
<th>ODS Graph Name</th>
<th>Plot Description</th>
<th>PLOTS= Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>CrossValidationASEPlot</td>
<td>Cross validation cost-complexity ASE plot when the response is continuous</td>
<td>CVCC</td>
</tr>
<tr>
<td>CrossValidationMisclassPlot</td>
<td>Cross validation cost-complexity misclassification plot when the response is categorical</td>
<td>CVCC</td>
</tr>
<tr>
<td>PruningPlot</td>
<td>Plot of the error sum of squares, misclassification rate, or cost complexity when it is used for final tree selection</td>
<td>PRUNEUNTIL</td>
</tr>
<tr>
<td>WholeTreePlot</td>
<td>Overview plot of final tree</td>
<td>WHOLETREE</td>
</tr>
<tr>
<td>ZoomedTreePlot</td>
<td>Detailed plot of portion of final tree</td>
<td>ZOOMEDTREE</td>
</tr>
</tbody>
</table>

**INPUT and TARGET Statement Syntax**

In addition to the syntax that is described in the CLASS and MODEL statement sections, PROC TREESPLIT supports INPUT/TARGET syntax that you might be familiar with from other procedures. The INPUT/TARGET syntax cannot be used together with the CLASS/MODEL syntax.

This syntax requires one TARGET statement and one or more INPUT statements. If you use this syntax, then the PROC TREESPLIT statement, the TARGET statement, and the INPUT statement are required. Depending on the options in those statements, specified variables can be interval or nominal. By default, numeric INPUT variables are treated as interval (or continuous) predictors, and character INPUT variables are treated as nominal (or categorical) predictors.

```
INPUT variables < / option> ;
TARGET variable < / option> ;
```

**INPUT Statement**

```
INPUT variables < / option> ;
```

The INPUT statement specifies predictor variables for the decision tree or regression tree. The value of variable can be a range such as “var_1–var_1000” or the special “_ALL_” value to include all variables in the data tables. As with CLASS variables, all nominal INPUT variables are padded or truncated to 32 characters.

You cannot use an INPUT statement with a MODEL or CLASS statement.

You can specify the following option:

**LEVEL=INTERVAL | NOMINAL**

specifies whether the specified predictor variables are interval or nominal. You can specify the following values:
INTerval treats all numeric variables as interval predictors.

NOMINAL treats all variables as nominal predictors.

By default, numeric variables are treated as interval predictors, and character variables are treated as nominal predictors. Specifying LEVEL=NOMINAL forces all variables in that statement to be treated as nominal. You cannot specify LEVEL=INTERVAL for character variables.

**TARGET Statement**

```plaintext
TARGET variable < / options > ;
```

The TARGET statement names the variable whose values PROC TREESPLIT predicts. Missing values in the target are ignored except during scoring.

You cannot use a TARGET statement with a MODEL or CLASS statement.

You can specify the following option:

**LEVEL=INTERVAL | NOMINAL**

specifies whether the specified response variable is interval or nominal. You can specify the following values:

- **INTERVAL** treats the response as an interval variable and creates a regression tree.
- **NOMINAL** treats the response as a nominal variable and creates a decision tree.

By default, LEVEL=INTERVAL for numeric variables and LEVEL=NOMINAL for categorical variables.

---

**Examples: TREESPLIT Procedure**

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

**Example 21.1: Creating a Binary Classification Tree with Validation Data**

A common use of classification trees is to predict whether a mortgage applicant will default on a loan. The data table Hmeq, which is in the Sampsio library that SAS provides, contains observations for 5,960 mortgage applicants. A variable named Bad indicates whether the applicant, after being approved for a loan, paid off or defaulted on the loan.

This example uses Hmeq to build a tree model that is used to score the data and can be used to score data about new applicants. Table 21.8 describes the variables in Hmeq.
Table 21.8 Variables in the Home Equity (Hmeq) Data Table

<table>
<thead>
<tr>
<th>Variable</th>
<th>Role</th>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bad</td>
<td>Response</td>
<td>Binary</td>
<td>1 = applicant defaulted on the loan or is seriously delinquent</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 = applicant paid off the loan</td>
</tr>
<tr>
<td>CLAge</td>
<td>Predictor</td>
<td>Interval</td>
<td>Age of oldest credit line in months</td>
</tr>
<tr>
<td>CLNo</td>
<td>Predictor</td>
<td>Interval</td>
<td>Number of credit lines</td>
</tr>
<tr>
<td>DebtInc</td>
<td>Predictor</td>
<td>Interval</td>
<td>Debt-to-income ratio</td>
</tr>
<tr>
<td>Delinq</td>
<td>Predictor</td>
<td>Interval</td>
<td>Number of delinquent credit lines</td>
</tr>
<tr>
<td>Derog</td>
<td>Predictor</td>
<td>Interval</td>
<td>Number of major derogatory reports</td>
</tr>
<tr>
<td>Job</td>
<td>Predictor</td>
<td>Nominal</td>
<td>Occupational category</td>
</tr>
<tr>
<td>Loan</td>
<td>Predictor</td>
<td>Interval</td>
<td>Requested loan amount</td>
</tr>
<tr>
<td>MortDue</td>
<td>Predictor</td>
<td>Interval</td>
<td>Amount due on mortgage</td>
</tr>
<tr>
<td>nInq</td>
<td>Predictor</td>
<td>Interval</td>
<td>Number of recent credit inquiries</td>
</tr>
<tr>
<td>Reason</td>
<td>Predictor</td>
<td>Binary</td>
<td>'DebtCon' = debt consolidation</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>'HomeImp' = home improvement</td>
</tr>
<tr>
<td>Value</td>
<td>Predictor</td>
<td>Interval</td>
<td>Value of property</td>
</tr>
<tr>
<td>YoJ</td>
<td>Predictor</td>
<td>Interval</td>
<td>Years at present job</td>
</tr>
</tbody>
</table>

The response variable for the tree model is Bad, a classification variable that has two values (0 for payment of loan and 1 for default). The other variables are predictor variables for the model. The following statements load mycas.hmeq data into your CAS session and display the first 10 observations of the data table. For this example, the statements assume that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

```plaintext
/* Convert variable names to mixed case */
data mycas.hmeq;
  length Bad Loan MortDue Value 8 Reason Job $7
    YoJ Derog Delinq nInq CLNo DebtInc 8;
  set sampsio.hmeq;
run;

proc print data=mycas.hmeq(obs=10); run;
```
Example 21.1: Creating a Binary Classification Tree with Validation Data

Output 21.1.1 shows the first 10 observations of mycas.hmeq.

**Output 21.1.1 Partial Listing of the mycas.hmeq Data**

<table>
<thead>
<tr>
<th>Obs</th>
<th>Bad</th>
<th>Loan</th>
<th>MortDue</th>
<th>Value</th>
<th>Reason</th>
<th>Job</th>
<th>YoJ</th>
<th>Derog</th>
<th>Delinq</th>
<th>CLAge</th>
<th>nInq</th>
<th>CLNo</th>
<th>DebtInc</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1100</td>
<td>25860</td>
<td>39025</td>
<td>HomeImp</td>
<td>Other</td>
<td>10.5</td>
<td>0</td>
<td>0</td>
<td>94.367</td>
<td>1</td>
<td>9</td>
<td>.</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1500</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1800</td>
<td>48649</td>
<td>57037</td>
<td>HomeImp</td>
<td>Other</td>
<td>5.0</td>
<td>3</td>
<td>2</td>
<td>77.100</td>
<td>1</td>
<td>17</td>
<td>.</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>2000</td>
<td>.</td>
<td>62250</td>
<td>HomeImp</td>
<td>Sales</td>
<td>16.0</td>
<td>0</td>
<td>0</td>
<td>115.800</td>
<td>0</td>
<td>13</td>
<td>.</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>2000</td>
<td>45000</td>
<td>55000</td>
<td>HomeImp</td>
<td>Other</td>
<td>3.0</td>
<td>0</td>
<td>0</td>
<td>86.067</td>
<td>2</td>
<td>25</td>
<td>.</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>2200</td>
<td>24280</td>
<td>34687</td>
<td>HomeImp</td>
<td>Other</td>
<td>.</td>
<td>0</td>
<td>1</td>
<td>300.867</td>
<td>0</td>
<td>8</td>
<td>.</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>2300</td>
<td>28192</td>
<td>40150</td>
<td>HomeImp</td>
<td>Other</td>
<td>4.5</td>
<td>0</td>
<td>0</td>
<td>54.600</td>
<td>1</td>
<td>16</td>
<td>.</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>2400</td>
<td>50000</td>
<td>73395</td>
<td>HomeImp</td>
<td>ProfExe</td>
<td>5.0</td>
<td>1</td>
<td>0</td>
<td>.</td>
<td>1</td>
<td>0</td>
<td>.</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>2400</td>
<td>.</td>
<td>17180</td>
<td>HomeImp</td>
<td>Other</td>
<td>.</td>
<td>0</td>
<td>0</td>
<td>14.567</td>
<td>3</td>
<td>4</td>
<td>.</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>2500</td>
<td>15000</td>
<td>20200</td>
<td>HomeImp</td>
<td>18.0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>136.067</td>
<td>1</td>
<td>19</td>
<td>.</td>
</tr>
</tbody>
</table>

The following statements use the TREESPLIT procedure to create a classification tree:

```sas
ods graphics on;

proc treesplit data=mycas.hmeq maxdepth=5;
   class Bad Delinq Derog Job nInq Reason;
   model Bad = Delinq Derog Job nInq Reason CLAge CLNo DebtInc Loan MortDue Value YoJ;
   prune costcomplexity;
   partition fraction(validate=0.3 seed=123);
   code file='treesplexc.sas';
run;
```

The MAXDEPTH= option specifies that the maximum depth of the tree to be grown is 5.

Specifying Bad to the left of the equal sign in the MODEL statement indicates that it is the response variable. Because no GROW statement is included, PROC TREESPLIT defaults to using the entropy metric, which calculates the gain to grow the tree. The PRUNE statement requests cost-complexity pruning.

The PARTITION statement requests that the observations in Hmeq be partitioned into disjoint subsets for model training and validation. Observations are randomly selected for the validation subset with probability 0.3; the remaining observations are selected for the training subset.

The FILE= option in the CODE statement requests that SAS DATA step score code be saved to a file named treesplexc.sas.
The tree diagram in Output 21.1.2 provides an overview of the full tree.

**Output 21.1.2  Overview Diagram of Final Tree**

The observations in terminal nodes 9, C, D, F, and I are assigned a prediction of Bad=0, and those in terminal nodes 3, 8, A, G, and H are assigned a prediction of Bad=1. Node C contains the most observations, as indicated by the thickness of the link from its parent node.

The tree diagram in Output 21.1.3 is a detailed view of the top portion of the tree. You can use the PLOTS= option in the PROC TREESPLIT statement to control which nodes are displayed.
Output 21.1.3 Detailed Tree Diagram

By default, this view provides detailed splitting information about the first three levels of the tree, including the root level. The splitting rule above each node shows the splitting variable and splitting value; the rule determines which observations from the parent node are included in the node. The first row of the table inside the node provides the node identifier. The second row provides the number of training observations. The third row shows the predicted response for observations in that node if classification occurs at that point, along with the proportion of training observations with that observed response. Note that the legend shows what actual value of the response variable is represented by the value shown in the node. For example, in node 6, 93.39% of the 3,208 observations in the training data have an observed response value of Bad=0, as indicated by the value 2.
Output 21.1.4 displays the fit statistics for the final tree.

Output 21.1.4  Tree Performance

<table>
<thead>
<tr>
<th>Number of Leaves</th>
<th>Misclassification Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>6</td>
</tr>
<tr>
<td>Validation</td>
<td>6</td>
</tr>
</tbody>
</table>

Output 21.1.5 displays the pruning plot.

Output 21.1.5  Pruning Plot

This plot displays the misclassification rates for the training and validation data as the tree is pruned. The tree with 10 leaves is selected as the final tree because it has the lowest misclassification rate for the validation data.

Creating Score Code and Scoring New Data

In addition to seeing information about the tree model, you might be interested in applying the model to predict the response variable in other data tables where the response is unknown. You can use the score code file treesplexc.sas (which was created by the FILE= option in the CODE statement) to score new data anywhere that you can run SAS DATA step code. The following is an example of using the score code file treesplexc.sas to score the data in Hmeq and save the results in a SAS data table named Scored.
data scored;
  set sampsio.hmeq;
  %include 'treesplexc.sas';
run;

Output 21.1.6 shows a partial listing of Scored.

<table>
<thead>
<tr>
<th>Obs</th>
<th>BAD</th>
<th>LOAN</th>
<th>MORTDUE</th>
<th>VALUE</th>
<th>REASON</th>
<th>YOJ</th>
<th>DEROG</th>
<th>DELINQ</th>
<th>CLAGE</th>
<th>NINQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1100</td>
<td>25860</td>
<td>39025</td>
<td>HomeImp</td>
<td>Other</td>
<td>10.5</td>
<td>0</td>
<td>0</td>
<td>94.367</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1300</td>
<td>70053</td>
<td>68400</td>
<td>HomeImp</td>
<td>Other</td>
<td>7.0</td>
<td>0</td>
<td>2</td>
<td>121.833</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1500</td>
<td>13500</td>
<td>16700</td>
<td>HomeImp</td>
<td>Other</td>
<td>4.0</td>
<td>0</td>
<td>0</td>
<td>149.467</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1500</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>1700</td>
<td>97800</td>
<td>112000</td>
<td>HomeImp</td>
<td>Office</td>
<td>3.0</td>
<td>0</td>
<td>0</td>
<td>93.333</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1700</td>
<td>30548</td>
<td>40320</td>
<td>HomeImp</td>
<td>Other</td>
<td>9.0</td>
<td>0</td>
<td>0</td>
<td>101.466</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>1800</td>
<td>48649</td>
<td>57037</td>
<td>HomeImp</td>
<td>Other</td>
<td>5.0</td>
<td>3</td>
<td>2</td>
<td>77.100</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>1800</td>
<td>28502</td>
<td>43034</td>
<td>HomeImp</td>
<td>Other</td>
<td>11.0</td>
<td>0</td>
<td>0</td>
<td>88.766</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>2000</td>
<td>32700</td>
<td>46740</td>
<td>HomeImp</td>
<td>Other</td>
<td>3.0</td>
<td>0</td>
<td>0</td>
<td>216.933</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>2000</td>
<td>.</td>
<td>62250</td>
<td>HomeImp</td>
<td>Sales</td>
<td>16.0</td>
<td>0</td>
<td>0</td>
<td>115.800</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Obs</th>
<th>CLNO</th>
<th>DEBTINC</th>
<th><em>leaf_id</em></th>
<th>I_Bad</th>
<th>P_Bad1</th>
<th>P_Bad0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9</td>
<td>.</td>
<td>9 1</td>
<td>0.58906</td>
<td>0.41094</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>14</td>
<td>.</td>
<td>9 1</td>
<td>0.58906</td>
<td>0.41094</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>.</td>
<td>9 1</td>
<td>0.58906</td>
<td>0.41094</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>.</td>
<td>.</td>
<td>5 1</td>
<td>0.93421</td>
<td>0.06579</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>14</td>
<td>.</td>
<td>9 1</td>
<td>0.58906</td>
<td>0.41094</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td>37.1136</td>
<td>10 0</td>
<td>0.06568</td>
<td>0.93432</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>17</td>
<td>.</td>
<td>9 1</td>
<td>0.58906</td>
<td>0.41094</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>36.8849</td>
<td>10 0</td>
<td>0.06568</td>
<td>0.93432</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>12</td>
<td>.</td>
<td>9 1</td>
<td>0.58906</td>
<td>0.41094</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>13</td>
<td>.</td>
<td>9 1</td>
<td>0.58906</td>
<td>0.41094</td>
<td></td>
</tr>
</tbody>
</table>

The data table contains the 13 original variables and 4 new variables that are created by the score code. The variable P_BAD1 is the proportion of training observations in this leaf for which BAD=1; this variable can be interpreted as the probability of default. The variable I_BAD indicates the predicted value of BAD for the observation.

You can use the preceding statements to score new data by including the new data table in place of Hmeq in the SET statement. The new data table must contain the same variables as the data that are used to build the tree model, but not the unknown response variable that you now want to predict.
Example 21.2: Creating a Regression Tree

This example performs an analysis in which a linear regression model is fit. You can alternatively fit a regression tree to predict the salaries of Major League Baseball players based on their performance measures from the previous season by using almost identical code. Regression trees are piecewise constant models that, for relatively small data tables such as Sashelp.Baseball, provide succinct summaries of how the predictor variables determine the predictions. These models are usually easier to interpret than linear regression models. The Sashelp.Baseball data table contains salary and performance information for Major League Baseball players (excluding pitchers) who played at least one game in both the 1986 and 1987 seasons (Time Inc. 1987). You can load the Sashelp.Baseball data set into your CAS session by naming your CAS engine libref in the first statement of the following DATA step:

```
data mycas.baseball;
  set sashelp.baseball;
run;
```

The following statements create a regression tree model:

```
ods graphics on;
proc treesplit data=mycas.baseball maxdepth=3;
  class league division;
  model logSalary = nAtBat nHits nHome nRuns nRBI nBB
crAtBat crHits crHome crRuns crRbi
crBB league division nOuts nAssts nError;
  output out=mycas.treesplout;
  prune none;
run;
```

Because no GROW statement is specified, the tree is grown using the RSS criterion by default. Because no PRUNE statement is included, no pruning is performed. The OUTPUT statement requests generation of the data table mycas.treesplout, which contains the predicted salary from the tree model for each observation.

Much of the output for a regression tree is identical to the output for a classification tree. Where there are differences, tables and plots are displayed and discussed on the following pages.

Output 21.2.1 displays the full regression tree.

---

**Example 21.2: Creating a Regression Tree**

This example performs an analysis in which a linear regression model is fit. You can alternatively fit a regression tree to predict the salaries of Major League Baseball players based on their performance measures from the previous season by using almost identical code. Regression trees are piecewise constant models that, for relatively small data tables such as Sashelp.Baseball, provide succinct summaries of how the predictor variables determine the predictions. These models are usually easier to interpret than linear regression models. The Sashelp.Baseball data table contains salary and performance information for Major League Baseball players (excluding pitchers) who played at least one game in both the 1986 and 1987 seasons (Time Inc. 1987). You can load the Sashelp.Baseball data set into your CAS session by naming your CAS engine libref in the first statement of the following DATA step:

```
data mycas.baseball;
  set sashelp.baseball;
run;
```

The following statements create a regression tree model:

```
ods graphics on;
proc treesplit data=mycas.baseball maxdepth=3;
  class league division;
  model logSalary = nAtBat nHits nHome nRuns nRBI nBB
crAtBat crHits crHome crRuns crRbi
crBB league division nOuts nAssts nError;
  output out=mycas.treesplout;
  prune none;
run;
```

Because no GROW statement is specified, the tree is grown using the RSS criterion by default. Because no PRUNE statement is included, no pruning is performed. The OUTPUT statement requests generation of the data table mycas.treesplout, which contains the predicted salary from the tree model for each observation.

Much of the output for a regression tree is identical to the output for a classification tree. Where there are differences, tables and plots are displayed and discussed on the following pages.

Output 21.2.1 displays the full regression tree.
The final selected tree has eight leaves. In a regression tree, the shade of the leaves represents the predicted response value, which is the average observed logSalary for the observations in that leaf. Node E has the lowest predicted response value, indicated by the lightest shade of blue, and node 7 has the highest, indicated by the dark shade.

Output 21.2.2 shows details of the first three levels of the tree, including the root node.
As in Output 21.1.3, this diagram displays split variables and split values for the nodes, along with the exact predicted response value, which is the average observed response, in each node.

Output 21.2.3 displays the fit statistic for the final regression tree (the only fit statistic provided for a regression tree is the ASE).

**Output 21.2.3** Regression Tree Performance

<table>
<thead>
<tr>
<th>Fits Statistics for Selected Tree</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Leaves</td>
</tr>
<tr>
<td>Training</td>
</tr>
<tr>
<td>8</td>
</tr>
</tbody>
</table>
Output 21.2.4 is a partial display of the mycas.treesplout data table that is created when you specify the OUTPUT statement.

<table>
<thead>
<tr>
<th>Obs</th>
<th>P_logSalary</th>
<th>DT_PredStd</th>
<th>LeafID</th>
<th>Residual</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4.6555457426</td>
<td>0.3064936468</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>6.8573201516</td>
<td>0.4410266912</td>
<td>7</td>
<td>0.6427120531</td>
</tr>
<tr>
<td>3</td>
<td>4.6555457426</td>
<td>0.3064936468</td>
<td>14</td>
<td>0.4070505006</td>
</tr>
<tr>
<td>4</td>
<td>6.8573201516</td>
<td>0.4410266912</td>
<td>7</td>
<td>-0.145745307</td>
</tr>
<tr>
<td>5</td>
<td>5.7477251685</td>
<td>0.3269101703</td>
<td>11</td>
<td>-0.56219311</td>
</tr>
<tr>
<td>6</td>
<td>4.6555457426</td>
<td>0.3064936468</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>6.8573201516</td>
<td>0.4410266912</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>4.6555457426</td>
<td>0.3064936468</td>
<td>14</td>
<td>-0.089386386</td>
</tr>
<tr>
<td>9</td>
<td>6.8573201516</td>
<td>0.4410266912</td>
<td>7</td>
<td>0.2023084645</td>
</tr>
<tr>
<td>10</td>
<td>6.35986721</td>
<td>0.5351772705</td>
<td>8</td>
<td>-0.260205997</td>
</tr>
</tbody>
</table>

The variable P_logSalary contains the predicted salaries on the log scale. Note that all observations in the same leaf have the same predicted response. The OUT= data table can contain additional variables from the DATA= data table if you specify them by using the COPYVARS= option.

**Example 21.3: Assessing Variable Importance**

This example creates a classification tree model to determine important variables (parameters) during the manufacture of a semiconductor device. Some of the variables that are involved in the manufacturing process are: gTemp, the growth temperature of substrate; aTemp, the annealing temperature; Rot, rotation speed; Dopant, the atom used during device growth; and Usable, which indicates whether the device is usable.

The following statements create a data table named mycas.MBE_Data, which contains measurements for 20 devices:

```plaintext
data mycas.MBE_Data;
  label gTemp = 'Growth Temperature of Substrate';
  label aTemp = 'Anneal Temperature';
  label Rot = 'Rotation Speed';
  label Dopant = 'Dopant Atom';
  label Usable = 'Experiment Could Be Performed';
  input gTemp aTemp Rot Dopant $ 39-40 Usable $ 47-54;
datalines;
384.614 633.172 1.01933 C Unusable
363.874 512.942 0.72057 C Unusable
397.395 671.179 0.90419 C Unusable
389.962 653.940 1.01417 C Unusable
387.763 612.545 1.00417 C Unusable
394.206 617.021 1.07188 Si Usable
387.135 616.035 0.94740 Si Usable
428.783 745.345 0.99087 Si Unusable
399.365 600.932 1.23307 Si Unusable
455.502 648.821 1.01703 Si Unusable
```

This example shows how to assess the importance of variables in the context of predicting salaries on a log scale. The model output includes a scored predictor data table that can be used to analyze how different factors influence the predicted salaries.
The following statements create the tree model:

```
proc treesplit data=mycas.MBE_Data maxdepth=6;
  class Usable Dopant;
  model Usable = gTemp aTemp Rot Dopant;
  prune none;
run;
```

Output 21.3.1 shows the “Variable Importance” table.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Importance</th>
<th>Std Dev</th>
<th>Relative Importance</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dopant</td>
<td>2.7000</td>
<td>0</td>
<td>1.0000</td>
<td>1</td>
</tr>
<tr>
<td>aTemp</td>
<td>1.0500</td>
<td>0</td>
<td>0.3889</td>
<td>1</td>
</tr>
</tbody>
</table>

This table shows that the predictor gTemp has the largest importance value. This means that the growth temperature of the substrate is the most important consideration in determining the usability of the sample. The column Count indicates how many times the variable was used in for splits within the decision tree.

References


Part III

Utility
Chapter 22
The ASSESS Procedure

Overview: ASSESS Procedure

The ASSESS procedure assesses and compares supervised learning models in SAS Viya. For a supervised learning model that has a nominal target, the ASSESS procedure produces lift information and receiver operating characteristic (ROC) information. For a regression model, the ASSESS procedure performs a quantile binning of the predictions and then returns the summary statistics of the response variable for each bin. PROC ASSESS also calculates fit statistics such as average square error, mean square logarithmic error, mean absolute error, mean consequential error, and multiclass log loss.

The expected input to the ASSESS procedure is score data from supervised learning models.
PROC ASSESS Features

The ASSESS procedure can be used to select a champion model from several models or to evaluate a certain model during the model assessment phase. The following list summarizes some basic features of PROC ASSESS:

- executes analysis in parallel on multiple machines
- is highly multithreaded
- calculates lift and gain in each bucket based on percentile
- calculates ROC information based on cut size
- calculates popular fit statistics for regression and classification models
- performs analysis for each partition

Using CAS Sessions and CAS Engine Librefs

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:

```sas
proc options option=(CASHOST CASPORT);
run;
```

In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:

```sas
cas mysess;
libname mycas cas sessref=mysess;
```

The CAS statement creates the CAS session named `mysess`, and the LIBNAME statement creates the `mycas` CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the `CASHOST` and `CASPORT` of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the `mysess` session, you can terminate it by using the TERMINATE option in the CAS statement as follows:
cas mysess terminate;

For more information about the CAS and LIBNAME statements, see the section “Introduction to Shared Concepts” on page 10 in Chapter 3, “Shared Concepts.”

Getting Started: ASSESS Procedure

NOTE: Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

This example demonstrates how to use the ASSESS procedure to compute lift regression information and fit statistics.

The following SAS DATA step creates the SAS data table mycas.score, which is similar to score data from a linear regression, and loads the table into your CAS session:

```
data mycas.score;
  input _PartInd_ good p_good;
datalines;
0 0.8224 0.7590
0 0.6538 0.4632
0 0.7693 0.7069
0 0.7491 0.7087
0 0.7779 0.7209
1 0.7161 0.8389
1 0.6779 0.7209
1 0.6392 0.6077
1 0.8090 0.9096
1 0.6064 0.7355
;
```

These statements assume that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

In this data table, the variable good is the ground truth and the variable p_good is the prediction from modeling. The variable _PartInd_ assigns each observation to one of two partitions for the analysis.

The following statements call the ASSESS procedure to compute lift regression information and fit statistics.

```
proc assess data=mycas.score nbins=2;
  var p_good;
  target good;
  by _PartInd_;
run;
```

The VAR statement specifies that the variable p_good should be analyzed in the model assessment. The TARGET statement specifies that the variable good is the response variable. The BY statement specifies that PROC ASSESS should perform the analysis separately for each value of _PartInd_.

Figure 22.1 shows the minimum, maximum, and mean of the actual and predicted values for _PartInd_=0.
Figure 22.1 Lift Regression Information for the First Partition

The ASSESS Procedure

\[\_PartInd\_ = 0\]

<table>
<thead>
<tr>
<th>Variable</th>
<th>Depth</th>
<th>Number of Observations</th>
<th>Predicted</th>
<th>Actual</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Minimum</td>
<td>Maximum</td>
</tr>
<tr>
<td>p_good</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>3</td>
<td>0.708700</td>
<td>0.759000</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>2</td>
<td>0.463200</td>
<td>0.706900</td>
</tr>
</tbody>
</table>

Figure 22.2 shows the error metrics of the interval target for \_PartInd\_ = 0.

Figure 22.2 Fit Statistics for the First Partition

\[\_PartInd\_ = 0\]

<table>
<thead>
<tr>
<th>Number of Observations</th>
<th>Divisor of Average</th>
<th>Squared Error</th>
<th>Absolute Error</th>
<th>Squared Logarithmic Error</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Average</td>
<td>Root Average</td>
<td>Root Average</td>
<td>Root Mean</td>
</tr>
<tr>
<td></td>
<td>Mean</td>
<td>Mean</td>
<td>Mean</td>
<td>Mean</td>
</tr>
<tr>
<td></td>
<td>Mean</td>
<td>Root Mean</td>
<td>Root Mean</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>0.009825</td>
<td>0.099119</td>
<td>0.082760</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.287680</td>
<td>0.003829</td>
<td>0.061879</td>
</tr>
</tbody>
</table>

Figure 22.3 shows the minimum, maximum, and mean of the actual and predicted values for \_PartInd\_ = 1.

Figure 22.3 Lift Regression Information for the Second Partition

The ASSESS Procedure

\[\_PartInd\_ = 1\]

<table>
<thead>
<tr>
<th>Variable</th>
<th>Depth</th>
<th>Number of Observations</th>
<th>Predicted</th>
<th>Actual</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Minimum</td>
<td>Maximum</td>
</tr>
<tr>
<td>p_good</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>3</td>
<td>0.735500</td>
<td>0.909600</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>2</td>
<td>0.607700</td>
<td>0.620900</td>
</tr>
</tbody>
</table>

Figure 22.4 shows the error metrics for the interval target for \_PartInd\_ = 1.
Syntax: ASSESS Procedure

The following statements are available in the ASSESS procedure:

```plaintext
PROC ASSESS < options > ;
  INPUT variable ;
  TARGET variables < /options > ;
  FITSTAT PVAR=variables / PEVENT="event-list" < DLM="character"> ;
  FREQ variable ;
  BY variable ;
```

The PROC ASSESS, INPUT, and TARGET statements are required.

The following sections describe the PROC ASSESS statement and then describe the other statements in alphabetical order.

PROC ASSESS Statement

```plaintext
PROC ASSESS < options > ;
```

The PROC ASSESS statement invokes the ASSESS procedure.

You can specify the following options:

- **DATA=CAS-libref.data-table**
  names the input data table for PROC ASSESS to use. The default is the most recently created data table. `CAS-libref.data-table` is a two-level name, where

  - **CAS-libref** refers to a collection of information that is defined in the LIBNAME statement and includes the `caslib`, which includes a path to the data, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about `CAS-libref`, see the section “Using CAS Sessions and CAS Engine Librefs” on page 1074.

  - **data-table** specifies the name of the input data table.

---

**Figure 22.4** Fit Statistics for the Second Partition

<table>
<thead>
<tr>
<th>Number of Observations</th>
<th>Squared Error</th>
<th>Absolute Error</th>
<th>Squared Logarithmic Error</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Divisor of Average</td>
<td>Root Average</td>
<td>Mean</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>5</td>
<td>0.009222</td>
</tr>
</tbody>
</table>
FITSTATOUT=\texttt{CAS-libref.data-table}
specifies the name of the fit statistics results table. \texttt{CAS-libref.data-table} is a two-level name, where \texttt{CAS-libref} refers to the caslib and session identifier, and \texttt{data-table} specifies the name of the output data table. For more information about this two-level name, see the \texttt{DATA=} option and the section “Using CAS Sessions and CAS Engine Librefs” on page 1074.

If you specify this option, all ODS tables are suppressed. This option is valid only if you also include a FITSTAT statement.

LIFTOUT=\texttt{CAS-libref.data-table}
specifies the name of the lift results table. \texttt{CAS-libref.data-table} is a two-level name, where \texttt{CAS-libref} refers to the caslib and session identifier, and \texttt{data-table} specifies the name of the output data table. For more information about this two-level name, see the \texttt{DATA=} option and the section “Using CAS Sessions and CAS Engine Librefs” on page 1074.

If you specify this option, all ODS tables are suppressed.

MAXITER=\texttt{integer}
specifies the maximum number of iterations for the percentile algorithm, where \texttt{integer} must be an integer greater than or equal to 1. By default, MAXITER=5 times the value of the NBINS= option.

NBINS=\texttt{integer}
specifies the number of bins to be used in the lift calculation, where \texttt{integer} must be an integer greater than or equal to 2.

By default, NBINS=20.

NCUTS=\texttt{integer}
specifies the number of cuts to be used in the ROC calculation, where \texttt{integer} must be an integer greater than or equal to 2. The inverse of \texttt{integer} represents the step size of the quantile. For example, NCUTS=10 generates 10 intervals for ROC analysis, with each step size 0.1. You cannot specify this option when the response variable that is specified in the TARGET statement is interval.

By default, NCUTS=100.

NTHREADS=\texttt{number}
specifies the number of threads that are used in the computation, where \texttt{number} must be an integer between 1 and 64, inclusive. The default value is the number of CPUs available in the machine.

ROCOUT=\texttt{CAS-libref.data-table}
specifies the name of the ROC results table. \texttt{CAS-libref.data-table} is a two-level name, where \texttt{CAS-libref} refers to the caslib and session identifier, and \texttt{data-table} specifies the name of the output data table. For more information about this two-level name, see the \texttt{DATA=} option and the section “Using CAS Sessions and CAS Engine Librefs” on page 1074.

If you specify this option, all ODS tables are suppressed. This option is valid only for a nominal target (LEVEL=NOMINAL in the TARGET statement).

\textbf{BY Statement}

\texttt{BY \ variable \ ;}

The BY statement specifies one partition \texttt{variable} to be used in the predictive model. If you specify the BY statement, PROC ASSESS performs the analysis separately for each value of the BY variable. If you do not specify the BY statement, PROC ASSESS treats the entire input data as one group for analysis.
FITSTAT Statement

The FITSTAT statement reports the following error metrics for a nominal target: average square error, divisor of average square error, root average square error, mean consequential error, and multiclass log loss. (For an interval target, the following metrics are created without specifying the FITSTAT statement: average square error, divisor of average square error, root average square error, mean absolute error, root mean absolute error, mean square logarithmic error, and root mean square logarithmic error.)

You must specify the following arguments:

\textbf{PVAR=} \textit{variables} \\

specifies the posterior probability for each level in model prediction except the variable specified in the INPUT statement. Duplicate variables are not allowed. If you specify the variable that is specified in the INPUT statement, it is ignored.

\textbf{PEVENT=} \textit{"event-list"} \\

specifies the events that correspond to each variable in the \textbf{PVAR=} option. The \textit{"event-list"} cannot include the event that corresponds to the variable in the TARGET statement. You must specify the \textbf{PEVENT=} and \textbf{PVAR=} options one-to-one in the same order. The maximal length allowed of \textbf{PEVENT} is 5000 characters.

You can also specify the following option:

\textbf{DELIMITER=} \textit{"character"} \\

\textbf{DLM=} \textit{"character"} \\

specifies the delimiter that is used to separate events that are specified in the \textbf{PEVENT=} option. A delimiter is used when event names contain embedded spaces (for example, \textit{“Fuel Oil”}) or special characters (for example, \textit{“;”} or \textit{“*”}). You must specify the quotation marks around \textit{character}. You can specify only one delimiter; combinations of delimiters are not supported. The valid values of the \textbf{DELIMITER=} option are \textit{“ ”} (space), \textit{“;”} (semicolon), \textit{“*”} (asterisk), \textit{“.”} (period), and \textit{“,”} (comma).

By default, \textbf{DELIMITER=} \textit{“ ”} (space).

If a delimiter other than \textit{“ ”} is specified, PROC ASSESS treats any spaces (including empty spaces) to be valid characters. You are responsible for mapping events correctly in order to obtain correct results.

FREQ Statement

\textbf{FREQ \textit{variable} ;} \\

The \textit{variable} in the FREQ statement identifies a numeric variable in the data set that contains the frequency of occurrence of each observation. PROC ASSESS treats each observation as if it appears \(f\) times, where \(f\) is the value of the FREQ \textit{variable} for the observation. If \(f\) is not an integer, it is truncated to an integer. If \(f\) is less than 1 or missing, the observation is not used in the analysis. When the FREQ statement is not specified, each observation is assigned a frequency of 1.
INPUT Statement

INPUT variable ;
VAR variable ;

The INPUT statement specifies the name of one variable to be analyzed in model assessment. When the variable specified in the TARGET statement is nominal, variable is the posterior probability of the event to be analyzed; when the variable specified in the TARGET statement is interval, variable is the predicted value.

TARGET Statement

TARGET variable < /options > ;

The TARGET statement specifies the response variable in supervised learning. You can specify the following options:

LEVEL=INTERVAL | NOMINAL
specifies the variable type.

You can specify the following values:

INTERVAL specifies that the response variable is interval, which must be numeric.

NOMINAL specifies that the response variable is nominal, also known as a classification variable, which can be numeric or character.

By default, LEVEL=INTERVAL.

EVENT="string"
specifies the formatted value of response variable that represents the event. For an interval target (LEVEL=INTERVAL), the EVENT= option is ignored. For a nominal target (LEVEL=NOMINAL), the EVENT= option is required.

Details: ASSESS Procedure

Lift Information

In the lift information table, the number of events are first allocated to each bin based on a percentile analysis of the probability of the event. In distributed mode, the percentile analysis is based on a heuristic, so it is an estimation of the precise percentile cutoff points that are used for binning. Consequently, the resulting cutoff points might be different from the cutoff points that are based on sorting the whole data set. This difference is more likely to happen when the data set is small.

Events are allocated to bins in two ways in order to perform the analysis. One way is the baseline case, which allocates events evenly to bins with the assumption that no analytic models are applied. The second way is
the predicted case, which allocates events to each bin based on probabilities from modeling. In each case, the following metrics are calculated independently:

- captured response percentage, which is the number of events in each bin divided by the total number of events
- cumulative captured response percentage, which accumulates the captured response percentages over bins
- response percentage, which is the number of events divided by the number of observations in each bin
- cumulative response percentage, which accumulates the response percentages over bins

Lift is defined as the ratio of the captured response percentage from the predicted case to the captured response percentage from the baseline case. Gain is defined as the ratio of the cumulative response percentage from the predicted case to the cumulative response percentage from the baseline case, minus one.

**ROC Information**

In the receiver operating characteristic (ROC) information table, the confusion matrix is calculated based on the event in each cutoff point. More terminology and the derivation of a confusion matrix is available in Fawcett (2006), Powers (2011), and Derby (2013).

The following notation is necessary for discussion of ROC information:

- \( m \) is the total cutoff points
- \( n \) is the number of observations
- \( N \) is the sum of observation frequencies in the data
- \( w_i \) are the observation frequencies, so

\[
N = \sum_{i=1}^{n} w_i
\]

- \( a_k \) is true positive at cutoff point \( k, k \in [0, m - 1] \)
- \( b_k \) is false positive at cutoff point \( k, k \in [0, m - 1] \)
- \( c_k \) is false negative at cutoff point \( k, k \in [0, m - 1] \)

The definitions of \( C \) (area under the curve), Gini, gamma, and tau are as follows:

\[
C = \frac{\mu + \frac{\theta}{2}}{\rho}
\]
Chapter 22: The ASSESS Procedure

Gini = \frac{\mu - \omega}{\rho}

gamma = \frac{\mu - \omega}{\mu + \omega}

tau = \frac{\mu - \omega}{\frac{N}{2}(N - 1)}

where:

\theta = \sum_{k=1}^{m} ((a_{k-1} - a_k)(b_{k-1} - b_k))

\mu = \sum_{k=2}^{m} \left( (a_{k-1} - a_k) \sum_{j=1}^{k} (b_{j-1} - b_j) \right)

\omega = \sum_{k=1}^{m} \left( (a_{k-1} - a_k) \sum_{j=k+1}^{m} (b_{j-1} - b_j) \right)

\rho = a_0 b_0, a_m = 0 and b_m = 0

The F0.5 score at cutoff point k is extended from F1 and is defined as follows:

\frac{(1 + \beta^2) \frac{pr}{\beta^2 p + r}}{1 + \beta^2}

where \( p = \frac{a_k}{a_k + b_k} \), \( r = \frac{a_k}{a_k + c_k} \), and \( \beta = 0.5 \). For more information about F0.5, see Kaggle Inc. (2015).

The Kolmogorov-Smirnov chart and other metrics are available in Derby (2013).

**Fit Statistics**

The following notation is necessary for discussion of fit statistics:

- \( n \) is the number of observations
- \( r \) is the number of levels for a nominal target
- \( N \) is the sum of observation frequencies in the data
- \( w_i \) is the frequency of observation \( i \), so
  \[
  N = \sum_{i=1}^{n} w_i
  \]
- \( y_i \) is the actual target value of observation \( i \)
- \( \hat{y}_i \) is the predicted target value of observation \( i \)
- \( t_i \) is the level from the target in observation \( i \)
- \( \hat{t}_i \) is the predicted level from the target in observation \( i \)
- \( y_{i,j} \) is 1 if observation \( i \) is assigned to target level \( j \), otherwise is 0
- \( p_{i,j} \) is the predicted probability that observation \( i \) is assigned to target level \( j \)

For interval targets, PROC ASSESS calculates the following metrics:

- average square error
  \[
  ASE = \frac{1}{N} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 w_i
  \]
- root average square error
  \[
  RASE = \sqrt{ASE}
  \]
- mean square logarithmic error
  \[
  MSLE = \frac{1}{N} \sum_{i=1}^{n} (\log(\hat{y}_i + 1) - \log(y_i + 1))^2 w_i
  \]
- mean absolute error
  \[
  MAE = \frac{1}{N} \sum_{i=1}^{n} |y_i - \hat{y}_i| w_i
  \]

For nominal targets, PROC ASSESS calculates the following metrics:

- average square error
  \[
  ASE = \frac{1}{rN} \sum_{i=1}^{n} \sum_{j=1}^{r} (y_{i,j} - p_{i,j})^2 w_i
  \]
- root average square error
  \[
  RASE = \sqrt{ASE}
  \]
• mean consequential error (misclassification)

$$\text{MCE} = \frac{1}{N} \sum_{t_i \neq \hat{t}_i} w_i$$

• multi-class log loss

$$\text{logloss} = -\frac{1}{N} \sum_{i=1}^{n} \sum_{j=1}^{r} y_{i,j} \log(p_{i,j}) w_i$$

---

**Displayed Output**

The following sections describe the output that PROC ASSESS produces. The output is organized into various tables, which are discussed in the order of their appearance.

**Fit Statistics**

The “Fit Statistics” table contains the following error metrics for a nominal target if you specify the FITSTAT statement: average square error, divisor of average square error, root average square error, mean consequential error, and multiclass log loss.

For an interval target, the following metrics are created without specifying the FITSTAT statement: average square error, divisor of average square error, root average square error, mean absolute error, root mean absolute error, mean square logarithmic error, and root mean square logarithmic error.

**Lift Information**

The “Lift Information” table contains information about the lift and gain of each partition for a nominal target.

**ROC Information**

The “ROC Information” table contains information about the receiver operating characteristic (ROC) of each partition for a nominal target.

**Lift Regression Information**

The “Lift Regression Information” table contains information about lift regression for each partition for an interval target.

**ODS Table Names**

Each table created by the ASSESS procedure has a name associated with it, and you must use this name to refer to the table when you use ODS statements. The names of each table and a short description of the contents are listed in Table 22.1.
Table 22.1 ODS Tables Produced by PROC ASSESS

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>FitStat</td>
<td>Fit statistics</td>
<td>FITSTAT</td>
<td>PVAR= and PEVENT= for a nominal target; default for interval target</td>
</tr>
<tr>
<td>LIFTInfo</td>
<td>Lift information</td>
<td>PROC ASSESS</td>
<td>Default for nominal target</td>
</tr>
<tr>
<td>LIFTRegInfo</td>
<td>Lift regression information</td>
<td>PROC ASSESS</td>
<td>Default for interval target</td>
</tr>
<tr>
<td>ROCInfo</td>
<td>ROC information</td>
<td>PROC ASSESS</td>
<td>Default for nominal target</td>
</tr>
</tbody>
</table>

Examples: ASSESS Procedure

NOTE: Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

Example 22.1: Assess a Model That Has a Binary Target

The following example demonstrates how to use the ASSESS procedure to compute ROC information, lift information, and fit statistics for a model that has a nominal target.

The following DATA step creates a score data table from a previously trained model. For example, the model could be a tree-based model, a neural network, or a support vector machine.

```sas
data mycas.score2;
  length good_bad $4;
  input _PartInd_ good_bad p_good p_bad;
datalines;
0 good 0.6675 0.3325
0 good 0.5189 0.4811
0 good 0.6852 0.3148
0 bad 0.0615 0.9385
0 bad 0.3053 0.6947
0 bad 0.6684 0.3316
0 good 0.6422 0.3578
0 good 0.6752 0.3248
0 good 0.5396 0.4604
0 good 0.4983 0.5017
```
The following PROC ASSESS call uses five bins to do lift analysis and uses 0.2 as the incremental step size for ROC analysis:

```plaintext
proc assess data=mycas.score2 ncuts=5 nbins=5;
  var p_good;
  target good_bad / event="good" level=nominal;
  fitstat pvar=p_bad / pevent="bad" ;
  by _PartInd_; 
run;
```

Output 22.1.1 shows the lift and gain for each bin in the first partition.
### Output 22.1.1 Lift Regression Information for the First Partition

**The ASSESS Procedure**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Event</th>
<th>Depth</th>
<th>Number of Observations</th>
<th>Number of Events</th>
<th>Lift Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>p_good</td>
<td>good</td>
<td>0</td>
<td>0 0</td>
<td>0 0</td>
<td>0.00 0.00 . . .</td>
</tr>
<tr>
<td>20</td>
<td>3</td>
<td>3</td>
<td>27.27</td>
<td>27.27 1.363636</td>
<td>1.363636 100.00 100.00</td>
</tr>
<tr>
<td>40</td>
<td>3</td>
<td>2</td>
<td>18.18</td>
<td>45.45 0.909091</td>
<td>1.136364 66.67 83.33</td>
</tr>
<tr>
<td>60</td>
<td>3</td>
<td>3</td>
<td>27.27</td>
<td>72.73 1.363636</td>
<td>1.212121 100.00 88.89</td>
</tr>
<tr>
<td>80</td>
<td>3</td>
<td>3</td>
<td>27.27</td>
<td>100.00 1.363636</td>
<td>1.250000 100.00 91.67</td>
</tr>
<tr>
<td>100</td>
<td>3</td>
<td>0</td>
<td>0.00</td>
<td>100.00 0 1</td>
<td>0.00 73.33</td>
</tr>
</tbody>
</table>

**Gain**

0.363636
0.136364
0.212121
0.250000
0
Output 22.1.2 shows the ROC information, including the confusion matrix and its derivations, for each bin in the first partition.

**Output 22.1.2** ROC Information for the First Partition

| Variable | Event | Cutoff | TP  | FP  | FN  | TN  | FPR  | FDR  | TPR  | TNR  | ACC  | KS   | Youden Index | F1 Score | F0.5 Score |
|----------|-------|--------|-----|-----|-----|-----|------|------|------|------|------|-----|-------|-----------|----------|------------|
| p_good   | good  | 0.20000 | 11  | 2   | 0   | 0   | 1    | 1    | 0    | 0    | 0    | 0   | 0.846154 | 0.774648  |
|          |       | 0.40000 | 11  | 1   | 0   | 3   | 0.250000 | 0.083333 | 1    | 0.750000 | 0.933333 | 0    | 0.956522 | 0.932203  |
|          |       | 0.60000 | 6   | 1   | 3   | 0.250000 | 0.166667 | 0.454545 | 0.750000 | 0.533333 | 0    | 0.204545 | 0.588235 | 0.714286 |
|          |       | 0.80000 | 4   | 1   | 1   | 1   | 0    | 0    | 1    | 0    | 0    | 0   | 0.266667 | 0          | 0        |

**Output 22.1.3** Fit Statistics for the First Partition

<table>
<thead>
<tr>
<th>Fit Statistics</th>
<th>Squared Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Observations</td>
<td>15</td>
</tr>
<tr>
<td>Divisor of Average</td>
<td>15</td>
</tr>
<tr>
<td>Root Average</td>
<td>0.169127</td>
</tr>
<tr>
<td>Mean Consequential Error</td>
<td>0.411250</td>
</tr>
<tr>
<td>Mean Multiclass Log Loss</td>
<td>0.266667</td>
</tr>
<tr>
<td>Mean Multiclass Log Loss</td>
<td>0.514728</td>
</tr>
</tbody>
</table>
Example 22.1: Assess a Model That Has a Binary Target

Output 22.1.4 shows the lift and gain for each bin in the second partition.

Output 22.1.4  Lift Regression Information for the Second Partition

The ASSESS Procedure

_PartInd_=1

<table>
<thead>
<tr>
<th>Variable</th>
<th>Event</th>
<th>Depth</th>
</tr>
</thead>
<tbody>
<tr>
<td>p_good</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>good</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>

<p>| Number of | Number of | Lift Information |</p>
<table>
<thead>
<tr>
<th>Observations</th>
<th>Events</th>
<th>Lift</th>
<th>Response Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>20</td>
<td>3</td>
<td>3</td>
<td>27.27</td>
</tr>
<tr>
<td>40</td>
<td>3</td>
<td>3</td>
<td>27.27</td>
</tr>
<tr>
<td>60</td>
<td>3</td>
<td>3</td>
<td>27.27</td>
</tr>
<tr>
<td>80</td>
<td>3</td>
<td>0</td>
<td>0.00</td>
</tr>
<tr>
<td>100</td>
<td>2</td>
<td>2</td>
<td>18.18</td>
</tr>
</tbody>
</table>

Gain

0.363636
0.363636
0.363636
0.022727
0

Output 22.1.5 shows the ROC information, including the confusion matrix and its derivations, for each bin in the second partition.
Chapter 22: The ASSESS Procedure

Output 22.1.5 ROC Information for the Second Partition

<table>
<thead>
<tr>
<th>Variable</th>
<th>Event</th>
<th>Cutoff</th>
<th>TP</th>
<th>FP</th>
<th>FN</th>
<th>TN</th>
<th>FPR</th>
<th>FDR</th>
<th>TPR</th>
<th>TNR</th>
<th>ACC</th>
<th>KS</th>
<th>Youden Index</th>
<th>F1 Score</th>
<th>F0.5 Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>p_good</td>
<td>good</td>
<td>0.00</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0.214286</td>
<td>0</td>
<td>1</td>
<td>0.785714</td>
<td>0</td>
<td>0.880000</td>
<td>0.820896</td>
</tr>
<tr>
<td>0.200000</td>
<td>10</td>
<td>0.00</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1.230769</td>
<td>0.909091</td>
<td>0</td>
<td>0.714286</td>
<td>0</td>
<td>0.0909091</td>
<td>0.833333</td>
<td>0.793651</td>
<td></td>
</tr>
<tr>
<td>0.400000</td>
<td>9</td>
<td>0.00</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0.250000</td>
<td>0.818182</td>
<td>0</td>
<td>0.642857</td>
<td>0</td>
<td>0.1818182</td>
<td>0.782609</td>
<td>0.762712</td>
<td></td>
</tr>
<tr>
<td>0.600000</td>
<td>5</td>
<td>0.00</td>
<td>1</td>
<td>6</td>
<td>3</td>
<td>0</td>
<td>0.454545</td>
<td>1</td>
<td>0.571429</td>
<td>1</td>
<td>0.454545</td>
<td>0.625000</td>
<td>0.806452</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.800000</td>
<td>3</td>
<td>0.00</td>
<td>1</td>
<td>8</td>
<td>3</td>
<td>0</td>
<td>0.272727</td>
<td>1</td>
<td>0.428571</td>
<td>0</td>
<td>0.272727</td>
<td>0.428571</td>
<td>0.652174</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0.00</td>
<td>1</td>
<td>11</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1.214286</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Output 22.1.6 shows the error metrics of the nominal target in the second partition.

Output 22.1.6 Fit Statistics for the Second Partition

<table>
<thead>
<tr>
<th>Variable</th>
<th>Event</th>
<th>Cutoff</th>
<th>AUC</th>
<th>Gini</th>
<th>Gamma</th>
<th>Tau</th>
<th>Accuracy</th>
<th>Youden Index</th>
<th>F1 Score</th>
<th>F0.5 Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>p_good</td>
<td>good</td>
<td>0.00</td>
<td>0.636364</td>
<td>0.272727</td>
<td>0.428571</td>
<td>0.098901</td>
<td>0.214286</td>
<td>0.285714</td>
<td>0.357143</td>
<td>0.428571</td>
</tr>
<tr>
<td>0.200000</td>
<td>10</td>
<td>0.00</td>
<td>0.636364</td>
<td>0.272727</td>
<td>0.428571</td>
<td>0.098901</td>
<td>0.214286</td>
<td>0.285714</td>
<td>0.357143</td>
<td>0.428571</td>
</tr>
<tr>
<td>0.400000</td>
<td>9</td>
<td>0.00</td>
<td>0.636364</td>
<td>0.272727</td>
<td>0.428571</td>
<td>0.098901</td>
<td>0.214286</td>
<td>0.285714</td>
<td>0.357143</td>
<td>0.428571</td>
</tr>
<tr>
<td>0.600000</td>
<td>5</td>
<td>0.00</td>
<td>0.636364</td>
<td>0.272727</td>
<td>0.428571</td>
<td>0.098901</td>
<td>0.214286</td>
<td>0.285714</td>
<td>0.357143</td>
<td>0.428571</td>
</tr>
<tr>
<td>0.800000</td>
<td>3</td>
<td>0.00</td>
<td>0.636364</td>
<td>0.272727</td>
<td>0.428571</td>
<td>0.098901</td>
<td>0.214286</td>
<td>0.285714</td>
<td>0.357143</td>
<td>0.428571</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
<th>Event</th>
<th>Cutoff</th>
<th>Squared Error</th>
<th>Divisor of Observations</th>
<th>Average</th>
<th>Root Average</th>
<th>Mean Error</th>
<th>Multiclass Log Loss</th>
</tr>
</thead>
<tbody>
<tr>
<td>p_good</td>
<td>good</td>
<td>0.00</td>
<td>14</td>
<td>14</td>
<td>0.221587</td>
<td>0.470731</td>
<td>0.142857</td>
<td>0.636734</td>
</tr>
<tr>
<td>0.200000</td>
<td>10</td>
<td>0.00</td>
<td>0.636364</td>
<td>0.272727</td>
<td>0.428571</td>
<td>0.098901</td>
<td>0.285714</td>
<td>0.357143</td>
</tr>
<tr>
<td>0.400000</td>
<td>9</td>
<td>0.00</td>
<td>0.636364</td>
<td>0.272727</td>
<td>0.428571</td>
<td>0.098901</td>
<td>0.285714</td>
<td>0.357143</td>
</tr>
<tr>
<td>0.600000</td>
<td>5</td>
<td>0.00</td>
<td>0.636364</td>
<td>0.272727</td>
<td>0.428571</td>
<td>0.098901</td>
<td>0.285714</td>
<td>0.357143</td>
</tr>
<tr>
<td>0.800000</td>
<td>3</td>
<td>0.00</td>
<td>0.636364</td>
<td>0.272727</td>
<td>0.428571</td>
<td>0.098901</td>
<td>0.285714</td>
<td>0.357143</td>
</tr>
</tbody>
</table>
References


Chapter 23
The BINNING Procedure

Contents

Overview: BINNING Procedure .................................................. 1094
Bucket Binning ................................................................. 1094
Winsorized Binning ............................................................. 1094
Quantile Binning ............................................................... 1094
Cutpoint Binning ............................................................... 1094
Tree-Based Binning ............................................................. 1094
PROC BINNING Features ...................................................... 1095
Using CAS Sessions and CAS Engine Librefs ............................. 1095
Getting Started: BINNING Procedure ...................................... 1096
Bucket Binning ................................................................. 1096
Syntax: BINNING Procedure .................................................. 1098
PROC BINNING Statement .................................................... 1098
CODE Statement .............................................................. 1102
FREQ Statement .............................................................. 1102
INPUT Statement .............................................................. 1102
OUTPUT Statement ........................................................... 1103
TARGET Statement ............................................................ 1104
Details: BINNING Procedure .................................................. 1105
Binning Computation and Formulas ....................................... 1105
Tree-Based Binning ............................................................. 1107
Computing the Weight of Evidence and Information Value ............. 1107
Displayed Output .............................................................. 1108
ODS Table Names ............................................................. 1109
Examples: BINNING Procedure ............................................. 1109
Example 23.1: Quantile Binning ............................................. 1109
Example 23.2: Winsorized Binning ......................................... 1112
Example 23.3: Bucket Binning and Weight-of-Evidence Computation 1114
Example 23.4: Cutpoint Binning ............................................. 1115
Example 23.5: Tree-Based Binning ......................................... 1116
Overview: BINNING Procedure

The BINNING procedure performs binning in SAS Viya. Binning is a common step in the data preparation stage of the model-building process. You can use binning to classify missing variables, reduce the impact of outliers, and generate multiple effects. The generated effects are useful and contain certain nonlinear information about the original variables.

The BINNING procedure supports several binning methods, which are described in the following subsections. The BINNING procedure can also calculate the weight of evidence (WOE) and information value (IV) based on binning results.

Bucket Binning

Bucket binning creates equal-length bins and assigns each observation in the data to one of these bins. You can choose the number of bins during the binning; the default number of bins (the binning level) is 16. Only interval input variables are supported for this binning method.

Winsorized Binning

Winsorized binning is similar to bucket binning except that both tails are cut off to obtain a smooth binning result. This technique is often used to remove outliers during the data preparation stage. Only interval input variables are supported for this binning method.

Quantile Binning

PROC BINNING calculates the quantile (or percentile) cutpoints and uses them as the lower bound and upper bound in creating bins. As a result, each bin should have a similar number of observations. Because PROC BINNING always assigns observations that have the same value to the same bin, quantile binning might create unbalanced bins if any variable has tied values. Only interval input variables are supported for this binning method. For more information, see the section “Binning Computation and Formulas” on page 1105.

Cutpoint Binning

The cutpoint binning method enables you to create bins by specifying the bin upper bound. Only interval input variables are supported for this binning method.

Tree-Based Binning

The tree-based binning method enables you to create bins by creating decision trees or regression trees. This technique requires you to specify a target variable. If the target variable is nominal, PROC BINNING creates decision trees; if the target variable is interval, PROC BINNING creates regression trees.
PROC BINNING Features

The BINNING procedure has the following features:

- provides a bucket (equal-length) binning method
- provides a Winsorized binning method and Winsorized statistics
- provides a quantile binning method
- provides a cutpoints binning method
- provides a tree-based binning method
- calculates the weight of evidence (WOE) and information value (IV) based on binning results

Using CAS Sessions and CAS Engine Librefs

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:

```sas
proc options option=(CASHOST CASPORT);
run;
```

In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:

```sas
cas mysess;
libname mycas cas sessref=mysess;
```

The CAS statement creates the CAS session named `mysess`, and the LIBNAME statement creates the `mycas` CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the CASHOST and CASPORT of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the `mysess` session, you can terminate it by using the TERMINATE option in the CAS statement as follows:

```sas
cas mysess terminate;
```

For more information about the CAS and LIBNAME statements, see the section “Introduction to Shared Concepts” on page 10 in Chapter 3, “Shared Concepts.”
Bucket Binning

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

The following DATA step creates the data table mycas.bucket, which consists of 1,000 observations of an ID variable (id) and three continuous variables (x1–x3), in your CAS session:

```sas
data mycas.bucket;
  length id 8;
  do id=1 to 1000;
    x1 = ranuni(101);
    x2 = 10*ranuni(201);
    x3 = 100*ranuni(301);
    output;
  end;
run;
```

These statements assume that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

The following statements show how you can use the BINNING procedure to perform bucket binning:

```sas
proc binning data=mycas.bucket numbin=10 method=bucket;
  input x1-x3;
  output out=mycas.out;
run;
```

The DATA= option specifies the input data table. The NUMBIN= option requests that 10 bins be created for all binning variables. The METHOD= option requests that the bucket binning method be used. The INPUT statement names three continuous variables (x1–x3) as input variables for binning. The OUTPUT statement creates an OUTPUT data table to contain the results of PROC BINNING.

Figure 23.1 displays the “Bin Details” table. This table shows the binning variable, bin ID, bin lower bound, bin upper bound, bin width, number of observations in that bin, and some statistics of that bin (such as mean, standard deviation, minimum, and maximum).
### The BINNING Procedure

#### Bin Details

<table>
<thead>
<tr>
<th>Variable</th>
<th>Bin ID</th>
<th>Lower Bound</th>
<th>Upper Bound</th>
<th>Bin Width</th>
<th>Number of Observations</th>
<th>Mean</th>
<th>Standard Deviation</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>Missing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>-Infy</td>
<td>0.0999</td>
<td></td>
<td>108</td>
<td>0.0472</td>
<td>0.0289</td>
<td>0.0005</td>
<td>0.0995</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.0999</td>
<td>0.1994</td>
<td>0.0995</td>
<td>94</td>
<td>0.1523</td>
<td>0.0313</td>
<td>0.1002</td>
<td>0.1993</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.1994</td>
<td>0.2989</td>
<td>0.0995</td>
<td>90</td>
<td>0.2477</td>
<td>0.0302</td>
<td>0.2003</td>
<td>0.2986</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.2989</td>
<td>0.3983</td>
<td>0.0995</td>
<td>95</td>
<td>0.3455</td>
<td>0.0284</td>
<td>0.2994</td>
<td>0.3965</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0.3983</td>
<td>0.4978</td>
<td>0.0995</td>
<td>90</td>
<td>0.4508</td>
<td>0.0282</td>
<td>0.3984</td>
<td>0.4977</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>0.4978</td>
<td>0.5973</td>
<td>0.0995</td>
<td>115</td>
<td>0.5506</td>
<td>0.0311</td>
<td>0.4998</td>
<td>0.5969</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>0.5973</td>
<td>0.6968</td>
<td>0.0995</td>
<td>116</td>
<td>0.6461</td>
<td>0.0282</td>
<td>0.5980</td>
<td>0.6951</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>0.6968</td>
<td>0.7963</td>
<td>0.0995</td>
<td>105</td>
<td>0.7509</td>
<td>0.0290</td>
<td>0.6983</td>
<td>0.7950</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>0.7962</td>
<td>0.8957</td>
<td>0.0995</td>
<td>106</td>
<td>0.8478</td>
<td>0.0260</td>
<td>0.8003</td>
<td>0.8940</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>0.8957</td>
<td>Infty</td>
<td></td>
<td>81</td>
<td>0.9421</td>
<td>0.0262</td>
<td>0.8978</td>
<td>0.9952</td>
</tr>
<tr>
<td>x2</td>
<td>Missing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>-Infy</td>
<td>1.0215</td>
<td></td>
<td>99</td>
<td>0.4865</td>
<td>0.2964</td>
<td>0.0277</td>
<td>1.0152</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1.0215</td>
<td>2.0154</td>
<td>0.9939</td>
<td>110</td>
<td>1.5061</td>
<td>0.2823</td>
<td>1.0232</td>
<td>1.9713</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>2.0154</td>
<td>3.0093</td>
<td>0.9939</td>
<td>99</td>
<td>2.4672</td>
<td>0.2840</td>
<td>2.0172</td>
<td>3.0041</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>3.0093</td>
<td>4.0031</td>
<td>0.9939</td>
<td>85</td>
<td>3.4925</td>
<td>0.3028</td>
<td>3.0129</td>
<td>4.0023</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>4.0031</td>
<td>4.9970</td>
<td>0.9939</td>
<td>109</td>
<td>4.4883</td>
<td>0.2898</td>
<td>4.0033</td>
<td>4.9736</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>4.9970</td>
<td>5.9909</td>
<td>0.9939</td>
<td>98</td>
<td>5.4785</td>
<td>0.2910</td>
<td>5.0073</td>
<td>5.9874</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>5.9909</td>
<td>6.9847</td>
<td>0.9939</td>
<td>100</td>
<td>6.5686</td>
<td>0.2797</td>
<td>6.0550</td>
<td>6.9823</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>6.9847</td>
<td>7.9786</td>
<td>0.9939</td>
<td>94</td>
<td>7.4833</td>
<td>0.2893</td>
<td>6.9975</td>
<td>7.9784</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>7.9786</td>
<td>8.9725</td>
<td>0.9939</td>
<td>112</td>
<td>8.4685</td>
<td>0.2947</td>
<td>7.9802</td>
<td>8.9657</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>8.9725</td>
<td>Infty</td>
<td></td>
<td>94</td>
<td>9.4733</td>
<td>0.2697</td>
<td>8.9760</td>
<td>9.9663</td>
</tr>
<tr>
<td>x3</td>
<td>Missing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>-Infy</td>
<td>10.174</td>
<td></td>
<td>97</td>
<td>5.2253</td>
<td>2.5781</td>
<td>2.0424</td>
<td>10.051</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>30.112</td>
<td>40.082</td>
<td>9.9694</td>
<td>94</td>
<td>35.979</td>
<td>2.7377</td>
<td>30.274</td>
<td>40.076</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>40.082</td>
<td>50.051</td>
<td>9.9694</td>
<td>102</td>
<td>44.474</td>
<td>2.9092</td>
<td>40.153</td>
<td>49.758</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>50.051</td>
<td>60.021</td>
<td>9.9694</td>
<td>99</td>
<td>55.331</td>
<td>2.6685</td>
<td>50.111</td>
<td>59.963</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>60.021</td>
<td>69.990</td>
<td>9.9694</td>
<td>107</td>
<td>64.927</td>
<td>2.7211</td>
<td>60.307</td>
<td>69.680</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>69.990</td>
<td>79.959</td>
<td>9.9694</td>
<td>92</td>
<td>74.900</td>
<td>2.9279</td>
<td>69.996</td>
<td>79.880</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>79.959</td>
<td>89.929</td>
<td>9.9694</td>
<td>91</td>
<td>84.836</td>
<td>2.9643</td>
<td>79.975</td>
<td>89.848</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>89.929</td>
<td>Infty</td>
<td></td>
<td>115</td>
<td>95.022</td>
<td>3.0062</td>
<td>90.078</td>
<td>99.898</td>
</tr>
</tbody>
</table>
Syntax: BINNING Procedure

The following statements are available in the BINNING procedure:

```plaintext
PROC BINNING <options>;
  CODE FILE=filename;
  FREQ variable;
  INPUT variables <options>;
  OUTPUT OUT=CAS-libref.data-table <options>;
  TARGET variable / <options>;
```

The PROC BINNING statement and at least one INPUT statement are required. You can specify multiple INPUT statements. The TARGET statement is required to calculate the weight of evidence (WOE) and information value (IV) when the value of the METHOD= option is BUCKET, QUANTILE, WINSOR, or CUTPTS. The TARGET statement is also required when the value of the METHOD= option is TREE.

The following sections describe the PROC BINNING statement and then describe the other statements in alphabetical order.

PROC BINNING Statement

```plaintext
PROC BINNING <options>;
```

The PROC BINNING statement invokes the procedure. Table 23.1 summarizes important options in the PROC BINNING statement by function.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATA=</td>
<td>Specifies the input data table</td>
</tr>
<tr>
<td>NUMBIN=</td>
<td>Specifies the global number of bins for all binning variables</td>
</tr>
<tr>
<td>DISTINCTCOUNTLIMIT=</td>
<td>Specifies the maximum number of distinct levels for interval variables in order for them to be treated as nominal variables</td>
</tr>
<tr>
<td>METHOD=</td>
<td>Specifies which binning method to use</td>
</tr>
<tr>
<td>WOE(WOEADJUST=)</td>
<td>Computes the weight of evidence and information values</td>
</tr>
<tr>
<td>BINMISSING=</td>
<td>Specifies whether to use missing values of the input variables for the binning process</td>
</tr>
</tbody>
</table>

Table 23.1  PROC BINNING Statement Options
Table 23.1  continued

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MISSINGBINSTATS=</td>
<td>Specifies whether to use the bin that contains the missing values for binning analysis</td>
</tr>
<tr>
<td>MISSINGEVENTNONEVENT=</td>
<td>Specifies whether to consider missing values of the target variable as nonevent</td>
</tr>
</tbody>
</table>

You can specify the following options:

**DATA=** `CAS-libref.data-table`

names the input data table for PROC BINNING to use. The default is the most recently created data table. `CAS-libref.data-table` is a two-level name, where

- `CAS-libref` refers to a collection of information that is defined in the LIBNAME statement and includes the `caslib`, which includes a path to the data, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about `CAS-libref`, see the section “Using CAS Sessions and CAS Engine Librefs” on page 1095.

- `data-table` specifies the name of the input data table.

**METHOD=** `BUCKET | QUANTILE | WINSOR(RATE=) | CUTPTS(numlist) | TREE <options>`

specifies which binning method to use.

You can specify one of the following methods:

**BUCKET**

uses the bucket binning method. Only interval input variables are supported for this binning method.

**QUANTILE**

uses the quantile binning method. Only interval input variables are supported for this binning method.

**WINSOR(RATE=rate)**

uses the Winsorized binning method and specifies the `rate` that it uses. You must specify a `rate` between 0 and 0.5, exclusive. Only interval input variables are supported for this binning method.

**CUTPTS(numlist)**

uses the cutpoints binning method and specifies the lower and upper bounds of the bin. The numbers in `numlist` specify the upper bounds of each bin except the last bin (whose upper bound is positive infinity). Each number in `numlist` also implies the lower bound of the next bin; the lower bound of the first bin is negative infinity. Only interval input variables are supported for this binning method.
TREE <options>
uses the tree-based binning method. Both interval and nominal input variables are supported for
this binning method. You can specify the following options:

INITBIN=integer
specifies the initial number of bins for interval input variables to start the tree-based binning
process. The final number of bins created by PROC BINNING might be different than the
number you specify. The value of integer can be any integer between 1 and 10,000, inclusive.
By default, INITBIN=100.

INITMETHOD=BUCKET | QUANTILE
specifies the method to initialize the tree-based binning for interval input variables. You can
specify either of the following values:

   BUCKET         uses the bucket binning method.
   QUANTILE       uses the quantile binning method.

By default, the bucket binning method is used.

LEAFSIZE=integer
MINNOBSINBIN=integer
specifies the allowed minimum number of observations per leaf for tree-based binning. The
value of integer can be any integer between 1 and 30,000. There is no default. If you specify
this option, the number of observations in each leaf will be larger than the specified integer.

MAXNBINS=integer
specifies the desired maximum number of bins for all variables. The value of integer can be
any integer between 2 and 10,000, inclusive. If this option is specified, PROC BINNING
tries to make sure that the final number of bins for all variables is no more than the specified
integer. By default, MAXNBINS=5.

MINNBINS=integer
specifies the desired minimum number of bins for all variables. The value of integer can be
any integer between 1 and 10,000, inclusive. If this option is specified, PROC BINNING
tries to make sure that the final number of bins for all variables is no less than the specified
integer. By default, MINNBINS=1.

CRITERIA <options>
specifies the criteria for growing the tree for the tree-based binning method. You can specify
the following options:

   GAIN
   ENTROPY
   uses the gain in information (decrease in entropy) as the tree-growing criterion. This
criterion is available only for a nominal target.

   GAINRATIO
ENTROPYRATIO
uses the gain ratio in information (decrease in entropy ratio) as the tree-growing criterion. This criterion is available only for a nominal target. This is the default criterion for a nominal target.

GINI
uses the Gini statistic as the tree-growing criterion. This criterion is available only for a nominal target.

SSE
uses the sum of squared error as the tree-growing criterion. This criterion is available only for an interval target. This is the default criterion for an interval target.

NUMBIN=integer
specifies the global number of binning levels for all binning variables. The value of integer can be any integer between 2 and 1000, inclusive.

By default, NUMBIN=16.

In tree-based binning, this option is ignored.

DISTINCTCOUNTLIMIT=integer
MAXDISTINCTLEVELS=integer
specifies the maximum number of distinct levels that interval variables can have in order for them to be treated as nominal variables, where integer can be any integer between 2 and 30,000, inclusive. You can use the LEVEL= option in the INPUT statement to treat interval variables as nominal variables as long as their levels are less than integer.

By default, DISTINCTCOUNTLIMIT=5000.

WOE(WOEADJUST=number)
computes the weight of evidence and information values. In tree-based binning, this option is ignored. You can specify the following suboption:

WOEADJUST=number
specifies the adjustment factor for the weight-of-evidence calculation, where number is between 0 and 1, inclusive.

By default, WOEADJUST=0.5.

BINMISSING=TRUE | FALSE
specifies whether to use missing values of the input variables for the binning process. You can specify either of the following values:

TRUE places all missing values into a bin whose binId=0.
FALSE ignores all missing values.

By default, BINMISSING=TRUE.
MISSINGBINSTATS=TRUE | FALSE
specifies whether to use the bin that contains the missing values for binning analysis such as WOE and
IV. You can specify either of the following values:

TRUE uses the bin that contains the missing values (binId=0) for binning analysis.
FALSE ignores the bin that contains the missing values.

By default, MISSINGBINSTATS=TRUE.

MISSINGEVALNONEVENT=TRUE | FALSE
specifies whether to consider missing values of the target variable as nonevent. You can specify either
of the following values:

TRUE considers missing values of the target variable to be nonevent.
FALSE excludes missing values of the target variable from analysis.

By default, MISSINGEVALNONEVENT=FALSE.

---

**CODE Statement**

CODE FILE=filename;

The CODE statement generates score code and stores it in a file that can be used for scoring purposes. Only
one CODE statement is processed. If you specify multiple CODE statements, only the first one is used.

You must specify the following option:

FILE=filename
specifies the filename of the file to write the SAS score code to.

---

**FREQ Statement**

FREQ variable;

The variable in the FREQ statement identifies a numeric variable in the data set that contains the frequency
of occurrence of each observation. PROC BINNING treats each observation as if it appears $f$ times, where $f$
is the value of the FREQ variable for the observation. If $f$ is not an integer, it is truncated to an integer. If $f$ is
less than 1 or missing, the observation is not used in the analysis. When the FREQ statement is not specified,
each observation is assigned a frequency of 1.

---

**INPUT Statement**

INPUT variables</options> ;
The INPUT statement names one or more variables as input variables for binning. PROC BINNING does not support duplicate variables. If the INPUT statement contains a duplicate variable, PROC BINNING takes only the first variable and provides a warning message.

When the values of the METHOD= option is BUCKET, QUANTILE, WINSOR, or CUTPTS, the specified variables must be interval variables. If nominal variables are provided, PROC BINNING stops and returns an error message.

When the values of the METHOD= option is TREE, the specified variables can be either interval or nominal variables.

You can specify the following options after a slash (/):

**NUMBIN=integer**

specifies the number of binning levels for all binning variables in the current INPUT statement. The value of integer can be any integer between 2 and 1,000, inclusive.

The number of binning levels that you specify in an INPUT statement overwrites the global number of binning levels, which is specified in the NUMBIN= option in the PROC BINNING statement (or is 16 by default).

In tree-based binning, this option is ignored.

**LEVEL=INTERVAL | NOMINAL**

specifies whether to treat the specified variables as interval or nominal. You can specify the following values:

**INTERVAL**

**INT**

treats all variables that are specified in this INPUT statement as interval. If the INPUT statement contains any character variables, PROC BINNING stops and returns an error message.

**NOMINAL**

**NOM**

treats all variables that are specified in this INPUT statement as nominal. If you specify LEVEL=NOMINAL and the variables in the INPUT statement are interval when the value of the METHOD= option is BUCKET, QUANTILE, WINSOR, or CUTPTS, PROC BINNING prints a warning and continues to treat the variables in the INPUT statement as interval variables.

By default, LEVEL=INTERVAL.

---

**OUTPUT Statement**

**OUTPUT OUT=CAS-libref.data-table < options > ;**

The OUTPUT statement enables you to write an output table back to the CAS server by creating an output data table that contains the results of PROC BINNING. For tree-based binning of class inputs, the OUTPUT statement also enables you to write an output bin-mapping data table that contains the following three columns: class input variables, their levels, and their associated bin numbers. If you do not specify the OUTPUT statement, the write-back process is skipped. This is useful when you want only the ODS output. By default, the output table contains the new binned variables. If you specify the COPYVARS= option,
Chapter 23: The BINNING Procedure

the output table contains the “copied variables” and the “binned variables”. The output bin-mapping data table contains only three columns, and the COPYVARS= option does not apply to it. If you specify multiple OUTPUT statements, PROC BINNING stops and produces an error message.

NOTE: If an input variable value is missing, then the binning output level value is 0.

You must specify one of the following options:

**OUT=** *CAS-libref.data-table*

names the output data table for PROC BINNING to use. *CAS-libref.data-table* is a two-level name, where *CAS-libref* refers to the caslib and session identifier, and *data-table* specifies the name of the output data table. For more information about this two-level name, see the DATA= option and the section “Using CAS Sessions and CAS Engine Librefs” on page 1095.

**OUTLEVELBINMAP=** *CAS-libref.data-table*

names the output bin-mapping data table for PROC BINNING to use. *CAS-libref.data-table* is a two-level name, where *CAS-libref* refers to the caslib and session identifier, and *data-table* specifies the name of the output data table. For more information about this two-level name, see the DATA= option and the section “Using CAS Sessions and CAS Engine Librefs” on page 1095.

You can also specify the following option:

**COPYVAR=** *variable*

**COPYVARS=** *(variables)*

lists one or more *variables* from the input data table to be transferred to the output data table.

---

**TARGET Statement**

**TARGET** *variable* / <options> ;

When the value of the METHOD= option is BUCKET, QUANTILE, WINSOR, or CUTPTS, the TARGET statement names the *variable* that PROC BINNING uses to calculate the weight of evidence and information value. If you specify multiple TARGET statements, only the first one is used.

When the value of the METHOD= option is TREE, the TARGET statement names the *variable* that PROC BINNING uses to create trees.

You can specify the following options after a slash (/):

**EVENT=** "*category"*

specifies the target event category that PROC BINNING uses to calculate the weight of evidence and information value in a quoted string of characters. You must specify this option if the WOE option is specified in the PROC BINNING statement.

**LEVEL=** *INTERVAL | NOMINAL*

specifies whether to treat the target variable as interval or nominal. You can specify the following values:
**Details: BINNING Procedure**

### Binning Computation and Formulas

For variable $x$, assume that the data are represented by $\{x_i\}$, where $i = 1, 2, \ldots, n$. Let $\min(x) = \min_{i \in \{1, \ldots, n\}} x_i$, and let $\max(x) = \max_{i \in \{1, \ldots, n\}} x_i$. The range of the variable is $\text{range}(x) = \max(x) - \min(x)$.

The computations for the various binning methods are as follows, where `numbin` is the value of the NUMBIN= option in the PROC BINNING statement:

- For bucket binning, the length of the bucket is
  
  $$L = \frac{\max(x) - \min(x)}{n}$$

  The split points are
  
  $$s_k = \min(x) + L \times k$$

  where $k = 1, 2, \ldots, \text{numbin} - 1$.

  When the data are evenly distributed on the cloud, the time complexity for bucket binning is $O(n/(\text{nodes} \times \text{nThreads}))$, where $n$ is the number of observations, $\text{nodes}$ is the number of computer nodes in the cloud, and $\text{nThreads}$ is the number of CPUs on each node.

- For quantile binning, PROC BINNING calculates a quantile table $P$. Let $P = \{p_k\}$, where $k = 1, 2, \ldots, \text{numbin}$. Then $p_k$ is described as
  
  $$p_k = \begin{cases} 
  1.0/\text{numbin} + p_{k-1} & \text{if } 0 < k < \text{numbin} \\
  1.0 & \text{if } k = \text{numbin} 
  \end{cases}$$

  Quantile binning often requires data to be sorted in a particular way, and the sorting process usually consumes a significant amount of CPU time and memory. When the input data table is larger than the available memory, the sorting algorithm becomes more complicated. In distributed computing, data communications overhead also increases the sorting challenge. To avoid the time-consuming sorting process, the BINNING procedure uses an iterative projection method for quantile binning, which runs much faster than the sorting-based quantile binning method in most cases.
After calculating the quantile table, PROC BINNING uses an iterative projection method to compute quantiles (percentiles) and uses these quantiles to create bins.

Quantile binning aims to assign the same number of observations to each bin. As a result, each bin should have a similar number of observations. Because PROC BINNING always assigns observations that have the same value to the same bin, quantile binning might create unbalanced bins if any variable has tied values. For example, if an observation whose value is \(x\) is assigned to bin \(k\), then every observation whose value is \(x\) is assigned to bin \(k\) for this variable, and no observation whose value is \(x\) is assigned to the next bin, bin \(k + 1\). Therefore, bin \(k\) might have more observations than bin \(k + 1\), because the tied values at the boundaries between bin \(k\) and bin \(k + 1\) are all assigned to bin \(k\). That is, tied values at the boundaries between two bins are always assigned to the lower-numbered bin.

- Winsorized binning applies bucket binning to the Winsorized data table. Winsorization involves replacing lower and upper outliers (which are defined according to the RATE= suboption in the METHOD=WINSOR option in the PROC BINNING statement) by the lower and upper thresholds. For variable \(x\), PROC BINNING uses a simple bucket sorting method to obtain the basic information.

Let \(N\) be the number of buckets. For each bucket \(B_i, i = 1, 2, \ldots, N\), PROC BINNING keeps the following information:

- \(c_i\): count of \(x\) in \(B_i\)
- \(\text{min}_i\): minimum value of \(x\) in \(B_i\)
- \(\text{max}_i\): maximum value of \(x\) in \(B_i\)
- \(\sum x_i\): sum of \(x\) in \(B_i\)
- \(\sum x_i^2\): sum of \(x^2\) in \(B_i\)

The Winsorized statistics are computed first. After the minimum and maximum have been found, the bin lower and upper bounds are calculated the same way as in bucket binning.

Let the tail count, \(wc\), be \(\text{ceil}(\text{Rate} \times n)\), and find the smallest \(I\) such that \(\sum_{i=1}^{I} c_i \geq wc\). Then the left tail count is \(\text{lwc} = \sum_{i=1}^{I} c_i\). Find the next \(I\) such that \(\sum_{i=1}^{I} c_i > lwc\). Therefore, the minimum value is \(\text{WinsorMin} = \text{min}_{I}\). Similarly, find the largest \(I\) such that \(\sum_{i=1}^{N} c_i \geq wc\). The right tail count is \(\text{rwc} = \sum_{i=1}^{N} c_i\). Find the next \(I\) such that \(\sum_{i=1}^{I} c_i > rwc\). Then the maximum value is \(\text{WinsorMax} = \text{max}_{I}\). The mean is calculated by the formula

\[
\text{WinsorMean} = \frac{lwc \times \text{WinsorMin} + \sum_{i=I}^{I} x_i + rwc \times \text{WinsorMax}}{n}
\]

The trimmed mean is calculated by the formula

\[
\text{TrimmedMean} = \frac{\sum_{i=I}^{I} x_i}{n - lwc - rwc}
\]

**NOTE:** PROC BINNING reserves bin 0 for missing values.

**NOTE:** Whenever possible, PROC BINNING always tries to generate the specified number of bins. In some cases, some bins might have no observations in them.
Tree-Based Binning

Tree-based binning is different from other binning methods. The basic idea of tree-based binning is to bin (group) input variables by growing a one-level decision tree (or regression tree, when the target variable is not nominal) for each input variable. In the literature, the difference between a decision tree and a regression tree is often ignored (with some exceptions). Therefore, in the rest of this chapter, the term decision tree can be used to represent either a decision tree or a regression tree. To grow a decision tree, PROC BINNING uses a training data set to split the target variable space into nonoverlapping regions. These regions correspond to the terminal nodes of the tree, which are also known as leaves.

The tree growing is done by splitting; the splitting starts with all the observations, which are represented by the node at the top of the tree. The algorithm splits this parent node into child nodes in such a way that the values (levels) of the target variable within each child region are as similar as possible. The split is determined by finding the best split value that optimizes a specified criterion across the child nodes for each input variable. The goal of maximizing similarity is achieved by minimizing node impurity. For an interval target, the default and only available growing criterion is SSE (sum of squared error), which aims to minimize node impurity. For a nominal target, the default growing criterion is GAINRATIO (or ENTROPYRATIO as the alias), which measures the gain ratio in information (decrease in entropy ratio) with the goal of minimizing node impurity. For a nominal target, the other supported growing criteria are GAIN (or ENTROPY as the alias) and GINI, which measure the gain in information (decrease in entropy) and the Gini statistic with the goal to minimize node impurity.

Computing the Weight of Evidence and Information Value

PROC BINNING can compute the weight of evidence and the information value.

Weight of evidence (WOE) is a measure of how much the evidence supports or undermines a hypothesis. WOE measures the relative risk of an attribute of binning level. The value depends on whether the value of the target variable is a nonevent or an event. An attribute’s WOE is defined as follows:

\[
\text{WOE}_{\text{attribute}} = \ln \frac{p_{\text{attr}(\text{nonevent})}}{p_{\text{attr}(\text{event})}} = \ln \frac{N_{\text{attr}(\text{nonevent})}/N_{\text{tot}(\text{nonevent})}}{N_{\text{attr}(\text{event})}/N_{\text{tot}(\text{event})}}
\]

The definitions of the quantities in the preceding formula are as follows:

- \(N_{\text{attr}(\text{nonevent})}\): the number of nonevent records that exhibit the attribute
- \(N_{\text{tot}(\text{nonevent})}\): the total number of nonevent records
- \(N_{\text{attr}(\text{event})}\): the number of event records that exhibit the attribute
- \(N_{\text{tot}(\text{event})}\): the total number of event records

To avoid an undefined WOE, an adjustment factor, \(x\), is used:
Chapter 23: The BINNING Procedure

WOE\text{attribute} = \ln \left( \frac{N_{\text{attr} (\text{nonevent})} + x}{N_{\text{tot} (\text{nonevent})}} \right) / \left( \frac{N_{\text{attr} (\text{event})} + x}{N_{\text{tot} (\text{event})}} \right)

You can use the WOEADJUST= option to specify a value between [0, 1] for x. By default, x is 0.5.

The information value (IV) is a weighted sum of the WOE of the characteristic’s attributes. The weight is the difference between the conditional probability of an attribute for an event and the conditional probability of that attribute for a nonevent. In the following formula of IV, m is the number of bins of a variable:

\[ IV = \sum_{i=1}^{m} \left( \frac{N_{\text{attr} (\text{nonevent})}}{N_{\text{tot} (\text{nonevent})}} - \frac{N_{\text{attr} (\text{event})}}{N_{\text{tot} (\text{event})}} \right) \times \text{WOE}_i \]

An information value can be any real number. Generally speaking, the higher the information value, the more predictive an attribute is likely to be.

Displayed Output

The following sections describe the output that PROC BINNING produces. The output is organized into various tables.

Bin Details

The “Bin Details” table displays some parameters and data information, which includes the binning variable, bin ID, bin lower bound, bin upper bound, bin width, number of observations in that bin, and some statistics of that bin (such as mean, standard deviation, minimum, and maximum). If you specify the WOE option in the PROC BINNING statement, this table also includes the target event count, weight of evidence, and information value of that bin.

Trimmed Statistics

The “Trimmed Statistics” table displays the binning variable, trimmed minimum, maximum, mean, standard error mean, left tail, left tail percentage, right tail, and right tail percentage. PROC BINNING generates this table if you specify the METHOD=WINSOR option in the PROC BINNING statement.

Winsorized Statistics

The “Winsorized Statistics” table displays the binning variable, Winsorized minimum, maximum, mean, standard error mean, left tail, left tail percentage, right tail, and right tail percentage. PROC BINNING generates this table if you specify the METHOD=WINSOR option in the PROC BINNING statement.

Variable Information Value

The “Variable Information Value (IV)” table displays the information value for each variable. PROC BINNING generates this table if you specify the WOE option in the PROC BINNING statement.
Variable Transformation Information

The “Variable Transformation Information” table displays the transformation information for each variable, in tree-based binning. PROC BINNING generates this table if you specify the METHOD=TREE option in the PROC BINNING statement.

ODS Table Names

Each table created by the BINNING procedure has a name associated with it, and you must use this name to refer to the table when you use ODS statements. The names of each table and a short description of the contents are listed in Table 23.2.

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>BinDetails</td>
<td>Basic binning information and parameters</td>
<td>PROC BINNING</td>
<td>Default</td>
</tr>
<tr>
<td>InfoValue</td>
<td>Information value for each variable</td>
<td>PROC BINNING</td>
<td>WOE</td>
</tr>
<tr>
<td>Trim</td>
<td>Trimmed statistics for the given variables</td>
<td>PROC BINNING</td>
<td>METHOD=WINSOR</td>
</tr>
<tr>
<td>Winsor</td>
<td>Winsor statistics for the given variables</td>
<td>PROC BINNING</td>
<td>METHOD=WINSOR</td>
</tr>
<tr>
<td>VarTransInfo</td>
<td>Transformation information for the specified variables</td>
<td>PROC BINNING</td>
<td>METHOD=TREE</td>
</tr>
</tbody>
</table>

Examples: BINNING Procedure

NOTE: Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

Example 23.1: Quantile Binning

The following DATA step generates 1,000,000 observations of an ID variable (id) and two continuous variables (x1–x2). The mycas CAS library reference enables your client machine to communicate with the CAS session. These statements assume that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.
Chapter 23: The BINNING Procedure

```sas
data mycas.ex1;
  length id 8;
  do id=1 to 1000000;
    x1 = ranuni(101);
    x2 = 10*ranuni(201);
    output;
  end;
run;
```

The following statements demonstrate how to use PROC BINNING to perform the quantile binning:

```sas
proc binning data=mycas.ex1 numbin=10 method=quantile;
  input x1-x2;
  output out=mycas.out1;
run;
```

The DATA= option specifies the input data table. The NUMBIN= option requests that 10 bins be created for all binning variables. The METHOD= option requests that the quantile binning method be used. The INPUT statement names two continuous variables (x1–x2) as input variables for binning. The OUTPUT statement creates an OUTPUT data table to contain the results of PROC BINNING.

The “Bin Details” table in Output 23.1 shows the binning variable, bin ID, bin lower bound, bin upper bound, bin width, number of observations in that bin, and some statistics of that bin (such as mean, standard deviation, minimum, and maximum). When the binning method is quantile, PROC BINNING assigns the same number of observations to each bin for the input variables if possible.
Example 23.1: Quantile Binning

Output 23.1.1 Bin Details

The BINNING Procedure

<table>
<thead>
<tr>
<th>Variable</th>
<th>Bin ID</th>
<th>Lower Bound</th>
<th>Upper Bound</th>
<th>Bin Width</th>
<th>Number of Observations</th>
<th>Mean</th>
<th>Standard Deviation</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>Missing</td>
<td>0</td>
<td></td>
<td></td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>-Infy</td>
<td>0.0999</td>
<td></td>
<td>999995</td>
<td>0.0498</td>
<td>0.0288</td>
<td>224E-9</td>
<td>0.0999</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.0999</td>
<td>0.1994</td>
<td>0.0996</td>
<td>100000</td>
<td>0.1496</td>
<td>0.0287</td>
<td>0.0999</td>
<td>0.1994</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.1994</td>
<td>0.2992</td>
<td>0.0998</td>
<td>100000</td>
<td>0.2493</td>
<td>0.0288</td>
<td>0.1994</td>
<td>0.2992</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.2992</td>
<td>0.3995</td>
<td>0.1003</td>
<td>99999</td>
<td>0.3494</td>
<td>0.0290</td>
<td>0.2992</td>
<td>0.3995</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0.3995</td>
<td>0.4999</td>
<td>0.1004</td>
<td>100000</td>
<td>0.4497</td>
<td>0.0291</td>
<td>0.3995</td>
<td>0.4999</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>0.4999</td>
<td>0.5997</td>
<td>0.0998</td>
<td>100000</td>
<td>0.5498</td>
<td>0.0288</td>
<td>0.4999</td>
<td>0.5997</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>0.5997</td>
<td>0.7004</td>
<td>0.1006</td>
<td>100000</td>
<td>0.6500</td>
<td>0.0291</td>
<td>0.5997</td>
<td>0.7004</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>0.7004</td>
<td>0.8002</td>
<td>0.0999</td>
<td>100000</td>
<td>0.7504</td>
<td>0.0288</td>
<td>0.7004</td>
<td>0.8002</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>0.8002</td>
<td>0.9002</td>
<td>0.1000</td>
<td>100000</td>
<td>0.8503</td>
<td>0.0288</td>
<td>0.8002</td>
<td>0.9002</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>0.9002</td>
<td>Infy</td>
<td></td>
<td>100000</td>
<td>0.9502</td>
<td>0.0288</td>
<td>0.9002</td>
<td>1.0000</td>
</tr>
<tr>
<td>x2</td>
<td>Missing</td>
<td>0</td>
<td></td>
<td></td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>-Infy</td>
<td>0.9969</td>
<td></td>
<td>99999</td>
<td>0.4988</td>
<td>0.2872</td>
<td>911E-8</td>
<td>0.9969</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.9969</td>
<td>1.9947</td>
<td>0.9978</td>
<td>100000</td>
<td>1.4971</td>
<td>0.2873</td>
<td>0.9969</td>
<td>1.9947</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1.9947</td>
<td>2.9937</td>
<td>0.9990</td>
<td>100000</td>
<td>2.4942</td>
<td>0.2887</td>
<td>1.9947</td>
<td>2.9937</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2.9937</td>
<td>3.9946</td>
<td>1.0009</td>
<td>100000</td>
<td>3.4942</td>
<td>0.2889</td>
<td>2.9937</td>
<td>3.9946</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>3.9946</td>
<td>4.9985</td>
<td>1.0039</td>
<td>100000</td>
<td>4.4969</td>
<td>0.2901</td>
<td>3.9946</td>
<td>4.9985</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>4.9985</td>
<td>5.9970</td>
<td>0.9985</td>
<td>100000</td>
<td>5.4974</td>
<td>0.2881</td>
<td>4.9985</td>
<td>5.9970</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>5.9970</td>
<td>6.9927</td>
<td>0.9956</td>
<td>100000</td>
<td>6.4945</td>
<td>0.2876</td>
<td>5.9970</td>
<td>6.9927</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>6.9927</td>
<td>7.9985</td>
<td>1.0059</td>
<td>99999</td>
<td>7.4974</td>
<td>0.2895</td>
<td>6.9927</td>
<td>7.9985</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>7.9985</td>
<td>8.9994</td>
<td>1.0008</td>
<td>100000</td>
<td>8.4983</td>
<td>0.2890</td>
<td>7.9985</td>
<td>8.9994</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>8.9994</td>
<td>Infy</td>
<td></td>
<td>100002</td>
<td>9.4999</td>
<td>0.2895</td>
<td>8.9994</td>
<td>10.000</td>
</tr>
</tbody>
</table>
Example 23.2: Winsorized Binning

The following DATA step generates 10,000 observations of an ID variable (id) and three continuous variables (x1–x3). The mycas CAS library reference enables your client machine to communicate with the CAS session.

```sas
data mycas.ex2;
  length id 8;
  do id=1 to 10000;
    x1 = ranuni(101);
    x2 = 10*ranuni(201);
    x3 = 100*ranuni(301);
    output;
  end;
run;
```

For Winsorized binning, PROC BINNING provides bucket binning and basic Winsorized statistical information for the input data.

The following statements include the WINSOR(RATE=0.05) option and generate tables for Winsorized and trimmed statistics:

```sas
proc binning data=mycas.ex2 numbin=10 method=winsor(rate=0.05);
  input x1-x3;
  output out=mycas.out2;
run;
```

The preceding statements generate Output 23.2.1 through Output 23.2.3.
### Output 23.2.1 Bin Details

The BINNING Procedure

<table>
<thead>
<tr>
<th>Variable</th>
<th>Bin ID</th>
<th>Lower Bound</th>
<th>Upper Bound</th>
<th>Bin Width</th>
<th>Number of Observations</th>
<th>Mean</th>
<th>Standard Deviation</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>Missing</td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0.0767</td>
<td>0.0304</td>
<td>0.0475</td>
<td>0.1376</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>-Inf</td>
<td>0.1377</td>
<td></td>
<td>1405</td>
<td>0.1832</td>
<td>0.0259</td>
<td>0.1380</td>
<td>0.2276</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.3180</td>
<td>0.4082</td>
<td>0.0902</td>
<td>864</td>
<td>0.3625</td>
<td>0.0258</td>
<td>0.3180</td>
<td>0.4081</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0.4082</td>
<td>0.4983</td>
<td>0.0902</td>
<td>906</td>
<td>0.4519</td>
<td>0.0258</td>
<td>0.4982</td>
<td>0.9982</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>0.7688</td>
<td>0.8589</td>
<td>0.0902</td>
<td>950</td>
<td>0.8124</td>
<td>0.0263</td>
<td>0.7690</td>
<td>0.8589</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>0.8589</td>
<td>Infy</td>
<td></td>
<td>1394</td>
<td>0.9192</td>
<td>0.0307</td>
<td>0.8589</td>
<td>0.9491</td>
</tr>
<tr>
<td>x2</td>
<td>Missing</td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>1.3978</td>
<td>0.7827</td>
<td>0.2989</td>
<td>0.4948</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>-Inf</td>
<td>1.3978</td>
<td></td>
<td>1385</td>
<td>1.8522</td>
<td>0.2597</td>
<td>1.3997</td>
<td>2.2997</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>2.3008</td>
<td>3.2038</td>
<td>0.9030</td>
<td>888</td>
<td>2.7546</td>
<td>0.2623</td>
<td>2.3027</td>
<td>3.2037</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>4.1069</td>
<td>5.0099</td>
<td>0.9030</td>
<td>905</td>
<td>4.5591</td>
<td>0.2567</td>
<td>4.1072</td>
<td>5.0091</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>5.9129</td>
<td>6.8159</td>
<td>0.9030</td>
<td>908</td>
<td>6.3697</td>
<td>0.2596</td>
<td>5.9137</td>
<td>6.8156</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>7.7189</td>
<td>8.6220</td>
<td>0.9030</td>
<td>935</td>
<td>8.1767</td>
<td>0.2593</td>
<td>7.7201</td>
<td>8.6218</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>8.6220</td>
<td>Infy</td>
<td></td>
<td>1410</td>
<td>9.2308</td>
<td>0.3049</td>
<td>8.6222</td>
<td>9.5250</td>
</tr>
<tr>
<td>x3</td>
<td>Missing</td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>14.323</td>
<td>8.2442</td>
<td>5.3640</td>
<td>14.310</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>23.282</td>
<td>32.241</td>
<td>8.9589</td>
<td>902</td>
<td>27.757</td>
<td>2.6725</td>
<td>23.305</td>
<td>32.227</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>32.241</td>
<td>41.200</td>
<td>8.9589</td>
<td>898</td>
<td>36.908</td>
<td>2.5926</td>
<td>32.242</td>
<td>41.184</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>50.158</td>
<td>59.117</td>
<td>8.9589</td>
<td>901</td>
<td>54.656</td>
<td>2.5158</td>
<td>50.156</td>
<td>59.108</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>68.076</td>
<td>77.035</td>
<td>8.9589</td>
<td>877</td>
<td>72.557</td>
<td>2.5229</td>
<td>68.090</td>
<td>77.026</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>85.994</td>
<td>Infy</td>
<td></td>
<td>1398</td>
<td>92.050</td>
<td>3.0000</td>
<td>86.001</td>
<td>94.953</td>
</tr>
</tbody>
</table>

### Output 23.2.2 Winsorized Statistics

<table>
<thead>
<tr>
<th>Variable</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Mean</th>
<th>Standard Deviation</th>
<th>N</th>
<th>Percent Left Tail</th>
<th>Percent Right Tail</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>0.0475</td>
<td>0.9491</td>
<td>0.5018</td>
<td>0.2848</td>
<td>499</td>
<td>4.99</td>
<td>500 5.00</td>
</tr>
<tr>
<td>x2</td>
<td>0.4948</td>
<td>9.5250</td>
<td>5.0302</td>
<td>2.8663</td>
<td>499</td>
<td>4.99</td>
<td>500 5.00</td>
</tr>
<tr>
<td>x3</td>
<td>5.3640</td>
<td>94.953</td>
<td>49.793</td>
<td>28.373</td>
<td>499</td>
<td>4.99</td>
<td>500 5.00</td>
</tr>
</tbody>
</table>
Output 23.2.3  Trimmed Statistics

<table>
<thead>
<tr>
<th>Variable</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Mean</th>
<th>Standard Deviation</th>
<th>N</th>
<th>Percent Left Tail</th>
<th>N</th>
<th>Percent Right Tail</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>0.0475</td>
<td>0.9491</td>
<td>0.5021</td>
<td>0.2599</td>
<td>499</td>
<td>500</td>
<td>5.00</td>
<td></td>
</tr>
<tr>
<td>x2</td>
<td>0.4948</td>
<td>9.5250</td>
<td>5.0320</td>
<td>2.6201</td>
<td>499</td>
<td>4.99</td>
<td>500</td>
<td>5.00</td>
</tr>
<tr>
<td>x3</td>
<td>5.3640</td>
<td>94.953</td>
<td>49.748</td>
<td>25.916</td>
<td>499</td>
<td>4.99</td>
<td>500</td>
<td>5.00</td>
</tr>
</tbody>
</table>

Example 23.3: Bucket Binning and Weight-of-Evidence Computation

The following DATA step generates a data table that contains 10 observations of a target variable (y), three continuous variables (x0–x2), and some other variables. The mycas CAS library reference enables your client machine to communicate with the CAS session.

```plaintext
data mycas.ex3;
   input cl1 $ x0 x1 x2 y $ freq id;
datalines;
a 2 . 7 n 2 1
a 2 2 6 . 3 2
a 3 0 1 o 0 3
c 2 3 7 y . 4
c 2 . 4 n -5 5
a 3 6 7 n 3 6
b 1 4 4 y 4 7
b 2 5 6 y 3 8
b 1 6 4 o 1 9
b 2 3 2 n 3 10
;
```

The following statements show how you can use the BINNING procedure to perform bucket binning and compute the WOE and the information value (IV):

```plaintext
proc binning data=mycas.ex3 numbin=5 woe;
   input x1/numbin=4;
   input x2;
   target y/event="y";
   output out=mycas.out3;
run;
```

The DATA= option specifies the input data table. The WOE option enables computation of the weight of evidence and information values with WOEADJUST=0.5 by default. The first INPUT statement names one continuous variable (x1) as the first input variable for binning with four bins. The second INPUT statement names another continuous variable (x2) as the second input variable for binning with five bins, as specified in the NUMBIN= global option. The TARGET statement names the variable (y) that PROC BINNING uses to calculate the weight of evidence, and the EVENT= option specifies the target event category in a quoted string. The OUTPUT statement creates an OUTPUT data table to contain the results of PROC BINNING.

The preceding statements generate Output 23.3.1 through Output 23.3.2.
Example 23.4: Cutpoint Binning

Output 23.3.1 Bin Details

The BINNING Procedure

<table>
<thead>
<tr>
<th>Variable</th>
<th>Bin ID</th>
<th>Lower Bound</th>
<th>Upper Bound</th>
<th>Bin Width</th>
<th>Number of Observations</th>
<th>Mean</th>
<th>Standard Deviation</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Event Count</th>
<th>Weight of Evidence</th>
<th>Information Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>Missing</td>
<td>-Infty</td>
<td>1.5000</td>
<td></td>
<td>2</td>
<td>0</td>
<td>0.9163</td>
<td>0</td>
<td></td>
<td></td>
<td>0.3054</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>-Infty</td>
<td>1.5000</td>
<td></td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0.4055</td>
<td>0.0676</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1.5000</td>
<td>3.15000</td>
<td></td>
<td>2</td>
<td>2</td>
<td>3.3333</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>-1.386</td>
<td>0.6931</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>3.45000</td>
<td>1.5000</td>
<td></td>
<td>3</td>
<td>5.6667</td>
<td>0.5774</td>
<td>5</td>
<td>6</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>4.5000</td>
<td>Infty</td>
<td></td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x2</td>
<td>Missing</td>
<td>-Infty</td>
<td>2.2000</td>
<td></td>
<td>2</td>
<td>1</td>
<td>1.5000</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0.9163</td>
<td>0.3054</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>-Infty</td>
<td>2.2000</td>
<td></td>
<td>2</td>
<td>2</td>
<td>0.7071</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>2.2000</td>
<td>3.4000</td>
<td></td>
<td>2</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>3.4000</td>
<td>4.6000</td>
<td></td>
<td>3</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>4.6000</td>
<td>5.8000</td>
<td></td>
<td>4</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>5.8000</td>
<td>Infty</td>
<td></td>
<td>5</td>
<td>6</td>
<td>6.6000</td>
<td>6</td>
<td>7</td>
<td>2</td>
<td>-0.693</td>
<td>0.2310</td>
</tr>
</tbody>
</table>

Output 23.3.2 Variable Information Value

<table>
<thead>
<tr>
<th>Variable</th>
<th>Information Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>1.0662</td>
</tr>
<tr>
<td>x2</td>
<td>0.5365</td>
</tr>
</tbody>
</table>

Example 23.4: Cutpoint Binning

The following DATA step generates a data table that contains 10 observations of a target variable (y), two continuous variables (x1, x2), and some other variables:

data mycas.ex4;
input cl1 $ cl2 x1 x2 y freq id;
datalines;
a 2 3 7 9 2 1
a 2 2 6 8 3 2
a 3 0 1 5 0 3
c 2 3 7 4 . 4
c 2 . 4 8 -5 5
a 3 6 7 5 3 6
b 1 4 4 8 4 7
b 2 5 6 3 3 8
b 1 6 4 8 1 9
b 2 3 2 6 3 10;


Chapter 23: The BINNING Procedure

The following statements show how you can use the BINNING procedure to perform cutpoint binning:

```plaintext
proc binning data=mycas.ex4 numbin=4 method=cutpts;
    input x2/cutpts(2, 2.3, 4.5);
    input x1/numbin=3 cutpts(3.1, 5);
run;
```

The DATA= option specifies the input data table. The METHOD= option requests that the cutpoint binning method be used. The first INPUT statement names one continuous variable `x2` as the first input variable for binning, with four bins specified by the NUMBIN= global option. The second INPUT statement names another continuous variable `x1` as the second input variable for binning with three bins. For the first input variable `x2`, 2, 2.3, 4.5 (and infinity) are used as the upper bounds for its four bins. For the second input variable `x1`, 3.1, 5 (and infinity) are used as the upper bounds for its three bins.

Output 23.4.1 shows the “Bin Details” table.

```
Output 23.4.1  Bin Details

The BINNING Procedure

<table>
<thead>
<tr>
<th>Variable</th>
<th>Bin ID</th>
<th>Lower Bound</th>
<th>Upper Bound</th>
<th>Bin Width</th>
<th>Number of Observations</th>
<th>Mean</th>
<th>Standard Deviation</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>x2</td>
<td>Missing</td>
<td>-Infy</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>2.3000</td>
<td>0.3000</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>2.3000</td>
<td>4.5000</td>
<td>2.2000</td>
<td>3</td>
<td>4</td>
<td>0</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>4.5000</td>
<td>Infy</td>
<td></td>
<td>5</td>
<td>6.6000</td>
<td>0.5477</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>x1</td>
<td>Missing</td>
<td>-Infy</td>
<td>3.1000</td>
<td></td>
<td>5</td>
<td>2.2000</td>
<td>1.3038</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>3.1000</td>
<td>5.19000</td>
<td></td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>5</td>
<td>Infy</td>
<td></td>
<td>3</td>
<td>5.6667</td>
<td>0.5774</td>
<td>5</td>
<td>6</td>
</tr>
</tbody>
</table>
```

Example 23.5: Tree-Based Binning

The following statements show how you can use the BINNING procedure to perform tree-based binning:

```plaintext
data mycas.hmeq;
    set sampsio.hmeq;
run;

proc binning data=mycas.hmeq method=tree;
    target bad/level=int;
    input mortdue/level = int;
    input job/ level = nom;
    output outlevelbinmap=mycas.outlevel;
run;

proc print data=mycas.outlevel;
run;
```

The DATA= option specifies the input data table. The METHOD= option requests that tree-based binning method be used. The TARGET statement specifies `bad` as the target variable and requests that it be treated as
an interval variable. The first INPUT statement names one variable, `mortdue`, and requests that it be treated as interval. The second INPUT statement names one variable, `job`, and requests that it be treated as nominal.

Output 23.5.1 shows the “Bin Details” table. Compared to the “Bin Details” tables in previous examples, this table has an extra column, “N Levels”. The reason is that other binning methods can handle only interval variables, but tree-based binning can handle both interval and nominal variables. The “N Levels” column shows the number of levels of this variable that this bin contains. As a result, for interval variables, this column is empty.

### Output 23.5.1 Bin Details

The BINNING Procedure

<table>
<thead>
<tr>
<th>Variable</th>
<th>Bin ID</th>
<th>Lower Bound</th>
<th>Upper Bound</th>
<th>Bin Width</th>
<th>N Levels</th>
<th>Number of Observations</th>
<th>Mean</th>
<th>Standard Deviation</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>MORTDUE</td>
<td>Missing</td>
<td>Infy</td>
<td>37837</td>
<td></td>
<td></td>
<td>518</td>
<td>933</td>
<td>23701</td>
<td>9087.0</td>
<td>7750</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>37837</td>
<td>73611</td>
<td>35774</td>
<td></td>
<td>2309</td>
<td>56416</td>
<td>9712.4</td>
<td>37843</td>
<td>73600</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>73611</td>
<td>105410</td>
<td>31799</td>
<td></td>
<td>1283</td>
<td>87448</td>
<td>9032.3</td>
<td>73624</td>
<td>105328</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>105410</td>
<td>145158</td>
<td>39749</td>
<td></td>
<td>575</td>
<td>124127</td>
<td>11706</td>
<td>105422</td>
<td>145127</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>145158</td>
<td>Infy</td>
<td></td>
<td></td>
<td>342</td>
<td>191406</td>
<td>48765</td>
<td>145324</td>
<td>399550</td>
</tr>
<tr>
<td>JOB</td>
<td>Missing</td>
<td>Infy</td>
<td>1</td>
<td></td>
<td></td>
<td>1</td>
<td>279</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>109</td>
<td></td>
<td></td>
<td>1</td>
<td>109</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1</td>
<td>193</td>
<td></td>
<td></td>
<td>2</td>
<td>3155</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>2</td>
<td>3155</td>
<td></td>
<td></td>
<td>1</td>
<td>1276</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>1</td>
<td>948</td>
<td></td>
<td></td>
<td>5</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Output 23.5.2 shows the “Transformation Information” table.

### Output 23.5.2 Transformation Information

<table>
<thead>
<tr>
<th>Variable</th>
<th>N</th>
<th>N Miss</th>
<th>N Bins</th>
<th>Importance</th>
<th>Relative Importance</th>
</tr>
</thead>
<tbody>
<tr>
<td>MORTDUE</td>
<td>5442</td>
<td>518</td>
<td>6</td>
<td>8.6977</td>
<td>0.5006</td>
</tr>
<tr>
<td>JOB</td>
<td>5681</td>
<td>279</td>
<td>6</td>
<td>17.375</td>
<td>1</td>
</tr>
</tbody>
</table>

Output 23.5.3 shows the “Output Bin Mapping” data table.

### Output 23.5.3 Output Bin Mapping Data Table

<table>
<thead>
<tr>
<th>Obs</th>
<th>Variable</th>
<th>Level</th>
<th>BinId</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>JOB</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>JOB</td>
<td>Sales</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>JOB</td>
<td>Self</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>JOB</td>
<td>Mgr</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>JOB</td>
<td>Other</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>JOB</td>
<td>ProfExe</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>JOB</td>
<td>Office</td>
<td>5</td>
</tr>
</tbody>
</table>
Chapter 24
The CARDINALITY Procedure

Overview: CARDINALITY Procedure

The CARDINALITY procedure determines a variable’s cardinality or limited cardinality in SAS Viya. The cardinality of a variable is the number of its distinct values, and the limited cardinality of a variable is the number of its distinct values that do not exceed a specified threshold.

In order to decide whether to include a variable in a subsequent SAS analysis as a classification or interval variable, it is sufficient to compute only the limited cardinality (and not the full cardinality) because both tend to yield the same decision. The limited cardinality is computationally less expensive than the full cardinality, especially for big data.

The CARDINALITY procedure creates the following:

- a cardinality data table, which contains summary information for each variable and some additional statistics about numeric variables
- a details data table, which contains the levels of each variable
PROC CARDINALITY Features

The CARDINALITY procedure has the following features:

- treats all variables as classification variables and attempts to determine the highest levels of each variable not to exceed a specified limit
- performs a single pass to determine the limited cardinality of each variable
- runs with all the input variables in the data table or with a specified list of variables
- can be run multiple times with a WHERE clause to reveal more levels. The results are not affected by the number of units doing the work (distributed or threaded or both) because PROC CARDINALITY produces the highest levels of each variable without exceeding the specified limit.
- recommends a level (CLASS, INTERVAL, or ID) for each variable. You can override these recommended levels in subsequent steps.
- can be run multiple times to extract subsequent segments of the full histogram of any variable
- builds the levelization in an ascending or descending order with or without formats

PROC CARDINALITY Compared to Other SAS Procedures

Other SAS procedures, such as the SUMMARY procedure, can calculate the cardinality of variables, but some are expensive to compute and result in a huge data set, and others yield nondeterministic results (that is, the results depend on the number of working units).

Using CAS Sessions and CAS Engine Librefs

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:

    proc options option=(CASHOST CASPORT);
    run;

In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:
cas mysess;
libname mycas cas sessref=mysess;

The CAS statement creates the CAS session named mysess, and the LIBNAME statement creates the mycas CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the CASHOST and CASPORT of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the mysess session, you can terminate it by using the TERMINATE option in the CAS statement as follows:

    cas mysess terminate;

For more information about the CAS and LIBNAME statements, see the section “Introduction to Shared Concepts” on page 10 in Chapter 3, “Shared Concepts.”

---

**Getting Started: CARDINALITY Procedure**

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

This example uses the Iris data set as input to demonstrate how to use PROC CARDINALITY. The Iris data published by Fisher (1936) include the species of iris and the sepal length, sepal width, petal length, and petal width (which are measured in millimeters) on 50 iris specimens from each of three species: *Iris setosa*, *I. versicolor*, and *I. virginica*.

You can load the sashelp.iris data set into your CAS session by naming your CAS engine libref in the first statement of the following DATA step:

    data mycas.iris;
      set sashelp.iris;
    run;

These statements assume that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

The following statements show the basic usage:

    proc cardinality data=mycas.iris outcard=mycas.card
      outdetails=mycas.details maxlevels=10;
    run;

The DATA= option names mycas.iris as the input data table. Two output data tables are requested: the required OUTCARD= option requests that cardinality and summary information be stored in the mycas.card data table, and the OUTDETAILS= option requests that the levels found for each variable be stored in the mycas.details data table. The MAXLEVELS= option specifies 10 as the maximum number of levels to report.
The following statements produce the “Cardinality Output Data” table, shown in Figure 24.1. This table contains one row for each variable in the input data table. If you do not want to display all the variables in the input data table, you can use the VAR statement to specify which variables you want to display.

```
proc print data=mycas.card;
  var _varname_ _type_ _cardinality_ _more_;
run;
```

**Figure 24.1** Cardinality Output Data Table

<table>
<thead>
<tr>
<th>Obs</th>
<th><em>VARNAME</em></th>
<th><em>TYPE</em></th>
<th><em>CARDINALITY</em></th>
<th><em>MORE</em></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Species</td>
<td>C</td>
<td>3</td>
<td>N</td>
</tr>
<tr>
<td>2</td>
<td>SepalLength</td>
<td>N</td>
<td>10</td>
<td>Y</td>
</tr>
<tr>
<td>3</td>
<td>SepalWidth</td>
<td>N</td>
<td>10</td>
<td>Y</td>
</tr>
<tr>
<td>4</td>
<td>PetalLength</td>
<td>N</td>
<td>10</td>
<td>Y</td>
</tr>
<tr>
<td>5</td>
<td>PetalWidth</td>
<td>N</td>
<td>10</td>
<td>Y</td>
</tr>
</tbody>
</table>

Table 24.1 explains the columns in the “Cardinality Output Data Table.” This example works with a simple subset of the variables in both the cardinality and details data tables.

**Table 24.1** Variables from the Cardinality Data

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>VARNAME</em></td>
<td>Variable name</td>
</tr>
<tr>
<td><em>TYPE</em></td>
<td>Variable type (N for numeric or C for character)</td>
</tr>
<tr>
<td><em>CARDINALITY</em></td>
<td>Number of levels extracted (less than or equal to the value of the MAXLEVELS= option)</td>
</tr>
<tr>
<td><em>MORE</em></td>
<td>Indication of more unreported levels (Y to indicate more levels or N to indicate no more levels)</td>
</tr>
</tbody>
</table>

Figure 24.1 shows that the Species variable has _TYPE_ = C, indicating that it is a character variable; its _CARDINALITY_ value is 3, indicating that three levels (values) are reported in the details data; and _MORE_ = N, indicating that there are no unreported levels.

Figure 24.1 shows that the SepalLength variable is a numeric variable (_TYPE_ = N); its _CARDINALITY_ value is 10, indicating that 10 different values are reported; and _MORE_ = Y, indicating that there are still more levels that can be explored.

The following statements produce the details output data table, as shown in Figure 24.2:

```
data details;
  set mycas.details;
  where _varname_ in ('Species', 'SepalLength');
run;

proc print data=details;
  var _VARNAME_ _INDEX_ _FREQ_ _RAWNUM_ _RAWCHAR_;
run;
```
Table 24.2 explains the columns in the details output data table.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>VARNAME</em></td>
<td>Variable name</td>
</tr>
<tr>
<td><em>INDEX</em></td>
<td>Index of the level</td>
</tr>
<tr>
<td><em>FREQ</em></td>
<td>Frequency of the level</td>
</tr>
<tr>
<td><em>RAWNUM</em></td>
<td>Raw level of the variable if numeric</td>
</tr>
<tr>
<td><em>RAWCHAR</em></td>
<td>Raw level of the variable if character</td>
</tr>
</tbody>
</table>

The _Species_ variable has three rows in the details data table, one row for each of the reported levels (the _CARDINALITY_ variable in the “Cardinality Output Data Table” has the value 3). Each of these rows has _FREQ_ = 50, and the _RAWCHAR_ values are reported for each level. There are no more unreported levels for the variable _Species_ (_MORE_ = N); thus, no row has a missing value (.) in the _INDEX_ column for the _Species_ variable.

The “Details Output Data Table” reports the top 10 levels of the _SepalLength_ variable and shows that there are still as many as 105 unreported levels that are greater than 52. (For the _SepalLength_ variable, _CARDINALITY_ = 10 and _MORE_ = Y in the “Cardinality Output Data Table.”) Therefore, _SepalLength_ occupies 11 rows in the details data table (10 reported levels and one additional level that describes the group of all unreported levels.) The _INDEX_ of the unreported group of levels for the _SepalLength_ variable is missing, reminding you that many levels make up this group. Essentially, the last row lumps all the other _SepalLength_ rows into one level that has a missing _INDEX_ value. So you see 10 levels in addition to the levels greater than 52, a total of 11 rows for the _SepalLength_ variable. The last row, which includes a missing _INDEX_ value, contains enough information to run the procedure again with a WHERE clause to obtain the next set of details.
Syntax: CARDINALITY Procedure

The following statements are available in the CARDINALITY procedure:

```
PROC CARDINALITY OUTCARD= 'CAS-libref.data-table < options > ' ;
   FREQ variable ;
   VAR variables < / options > ;
```

The PROC CARDINALITY statement is required.

PROC CARDINALITY Statement

```
PROC CARDINALITY OUTCARD= 'CAS-libref.data-table < options > ' ;
```

The PROC CARDINALITY statement invokes the procedure. Table 24.3 summarizes the options available in the PROC CARDINALITY statement.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input and Output Data Sets</strong></td>
<td></td>
</tr>
<tr>
<td>DATA=</td>
<td>Specifies the input data table</td>
</tr>
<tr>
<td>OUTCARD=</td>
<td>Names the cardinality data table to contain the summary information for each variable</td>
</tr>
<tr>
<td>OUTDETAILS=</td>
<td>Names an optional data table to contain the detailed levels of all the variables</td>
</tr>
<tr>
<td><strong>Cardinality Options</strong></td>
<td></td>
</tr>
<tr>
<td>MAXLEVELS=</td>
<td>Specifies the maximum number of levels to consider</td>
</tr>
<tr>
<td>ORDER=</td>
<td>Specifies the order to be applied to all variables that have an unspecified order</td>
</tr>
</tbody>
</table>

You must specify the following option:

**OUTCARD= 'CAS-libref.data-table'**

specifies the output data table to contain the cardinality and summary information for each variable. **CAS-libref.data-table** is a two-level name, where **CAS-libref** refers to the caslib and session identifier, and **data-table** specifies the name of the output data table. For more information about this two-level name, see the DATA= option and the section “Using CAS Sessions and CAS Engine Librefs” on page 1120.

You can also specify the following options:

**DATA= 'CAS-libref.data-table'**

names the input data table for PROC CARDINALITY to use. **CAS-libref.data-table** is a two-level name, where **CAS-libref** refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to the data, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME
FREQ Statement

FREQ statement. For more information about CAS-libref, see the section “Using CAS Sessions and CAS Engine Librefs” on page 1120.

data-table specifies the name of the input data table.

MAXLEVELS=number
specifies the maximum number of levels of the variables in the input data to consider.

ORDER=ASC | DESC | ASCFMT | DESFMT
specifies the order to be applied to all variables whose order is not specified in a VAR statement.

You can specify the following values:

ASC orders variables in unformatted ascending order.
DESC orders variables in unformatted descending order.
ASCFMT orders variables in formatted ascending order.
DESFMT orders variables in formatted descending order.

By default, ORDER=ASC.

OUTDETAILS=CAS-libref.data-table
specifies the output data table to contain the levels that are found for each variable. CAS-libref.data-table is a two-level name, where CAS-libref refers to the caslib and session identifier, and data-table specifies the name of the input data table. For more information about this two-level name, see the DATA= option and the section “Using CAS Sessions and CAS Engine Librefs” on page 1120.

All the levels that are higher than the value of the MAXLEVELS= option are grouped together into one level that has a missing value (.) in the _INDEX_ column in the output data table. For this level, the corresponding raw and formatted levels contain the last values visible in the histogram. This is useful when you need the next segment of the histogram past the visible levels of a variable.

FREQ Statement

FREQ variable ;

The variable in the FREQ statement identifies a numeric variable in the data set that contains the frequency of occurrence of each observation. PROC CARDINALITY treats each observation as if it appears \( f \) times, where \( f \) is the value of the FREQ variable for the observation. If \( f \) is not an integer, it is truncated to an integer. If \( f \) is less than 1 or missing, the observation is not used in the analysis. When the FREQ statement is not specified, each observation is assigned a frequency of 1.

Except for the minimum and the maximum values, the statistics for each variable are affected by the value of variable. Missing and nonpositive values do not contribute to the other statistics.
VAR Statement

    VAR variables < / options > ;

The VAR statement enables you to specify a subset of variables in the input data table (which is specified in the DATA= option in the PROC CARDINALITY statement) and specify how to order them.

You can specify one or more VAR statements for each type of ordering that you want. You cannot specify a variable more than once in all the VAR statements.

ORDER=ASC | DESC | ASCFMT | DESFMT

requests a specific method of levelization. You can specify the following values:

- **ASC** orders the specified variables in unformatted ascending order.
- **DESC** orders the specified variables in unformatted descending order.
- **ASCFMT** orders the specified variables in formatted ascending order.
- **DESFMT** orders the specified variables in formatted descending order.

By default, the value of the ORDER= option in the PROC CARDINALITY statement is used. If the ORDER= option is not specified in the PROC CARDINALITY statement, ORDER=ASC by default if the variable is unformatted or ORDER=ASCFMT by default if the variable is formatted.

Details: CARDINALITY Procedure

Displayed Output

The output data tables have fixed variable names and labels. Each variable occupies one row in the cardinality data table; the number of rows that it can occupy in the details data table is less than or equal to the value of the MAXLEVELS= option plus 1. The following statements run PROC CARDINALITY and then PROC CONTENTS to examine contents of the CARD table:

```sas
proc cardinality data=mycas.iris outcard=mycas.card
   outdetails=mycas.details maxlevels=10;
run;

data card;
   set mycas.card;
run;

proc contents data=card order=varnum;
run;
```
The CONTENTS Procedure

Variables in Creation Order

<table>
<thead>
<tr>
<th>#</th>
<th>Variable</th>
<th>Type</th>
<th>Len</th>
<th>Format</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><em>VARNAME</em></td>
<td>Char</td>
<td>32</td>
<td>$</td>
<td>Variable name</td>
</tr>
<tr>
<td>2</td>
<td><em>FMTWIDTH</em></td>
<td>Num</td>
<td>8</td>
<td>BEST.</td>
<td>Width of the variable formatted value</td>
</tr>
<tr>
<td>3</td>
<td><em>TYPE</em></td>
<td>Char</td>
<td>1</td>
<td>$</td>
<td>Type of the raw values</td>
</tr>
<tr>
<td>4</td>
<td><em>RLEVEL</em></td>
<td>Char</td>
<td>10</td>
<td>$</td>
<td>Recommended level for analytics</td>
</tr>
<tr>
<td>5</td>
<td><em>ORDER</em></td>
<td>Char</td>
<td>8</td>
<td>$</td>
<td>Variable sort order</td>
</tr>
<tr>
<td>6</td>
<td><em>MORE</em></td>
<td>Char</td>
<td>1</td>
<td>$</td>
<td>Have more unreported levels</td>
</tr>
<tr>
<td>7</td>
<td><em>CARDINALITY</em></td>
<td>Num</td>
<td>8</td>
<td>BEST.</td>
<td>Number of levels</td>
</tr>
<tr>
<td>8</td>
<td><em>NOBS</em></td>
<td>Num</td>
<td>8</td>
<td>BEST.</td>
<td>Number of observations</td>
</tr>
<tr>
<td>9</td>
<td><em>SUMFREQS</em></td>
<td>Num</td>
<td>8</td>
<td>BEST.</td>
<td>Total summation of frequencies</td>
</tr>
<tr>
<td>10</td>
<td><em>NMISS</em></td>
<td>Num</td>
<td>8</td>
<td>BEST.</td>
<td>Number of missing values</td>
</tr>
<tr>
<td>11</td>
<td><em>MISSFMT</em></td>
<td>Char</td>
<td>32</td>
<td>$</td>
<td>Format of the missing value</td>
</tr>
<tr>
<td>12</td>
<td><em>VISIBLE</em></td>
<td>Num</td>
<td>8</td>
<td>BEST.</td>
<td>Percentage of the visible part of the report</td>
</tr>
<tr>
<td>13</td>
<td><em>MIN</em></td>
<td>Num</td>
<td>8</td>
<td>BEST.</td>
<td>Minimum numeric value</td>
</tr>
<tr>
<td>14</td>
<td><em>MAX</em></td>
<td>Num</td>
<td>8</td>
<td>BEST.</td>
<td>Maximum numeric value</td>
</tr>
<tr>
<td>15</td>
<td><em>MEAN</em></td>
<td>Num</td>
<td>8</td>
<td>BEST.</td>
<td>Mean</td>
</tr>
<tr>
<td>16</td>
<td><em>STDDEV</em></td>
<td>Num</td>
<td>8</td>
<td>BEST.</td>
<td>Standard deviation</td>
</tr>
<tr>
<td>17</td>
<td><em>SKEWNESS</em></td>
<td>Num</td>
<td>8</td>
<td>BEST.</td>
<td>Skewness</td>
</tr>
<tr>
<td>18</td>
<td><em>KURTOSIS</em></td>
<td>Num</td>
<td>8</td>
<td>BEST.</td>
<td>Kurtosis</td>
</tr>
<tr>
<td>19</td>
<td><em>MFREQ</em></td>
<td>Num</td>
<td>8</td>
<td>BEST.</td>
<td>Maximum frequency</td>
</tr>
<tr>
<td>20</td>
<td><em>MREQFOUNDLEVEL</em></td>
<td>Char</td>
<td>1</td>
<td>$</td>
<td>Found maximum frequency in visible part of the report</td>
</tr>
<tr>
<td>21</td>
<td><em>MFREQNUM</em></td>
<td>Num</td>
<td>8</td>
<td>BEST.</td>
<td>Numeric level with the maximum frequency</td>
</tr>
<tr>
<td>22</td>
<td><em>MREQCHR</em></td>
<td>Char</td>
<td>32</td>
<td>$</td>
<td>Character level with the maximum frequency</td>
</tr>
<tr>
<td>23</td>
<td><em>MREQCFMT</em></td>
<td>Char</td>
<td>32</td>
<td>$</td>
<td>Formatted level with the maximum frequency</td>
</tr>
<tr>
<td>24</td>
<td><em>LASTNUM</em></td>
<td>Num</td>
<td>8</td>
<td>BEST.</td>
<td>Last raw numeric value (cutoff)</td>
</tr>
<tr>
<td>25</td>
<td><em>LASTCHR</em></td>
<td>Char</td>
<td>32</td>
<td>$</td>
<td>Last raw character value (cutoff)</td>
</tr>
<tr>
<td>26</td>
<td><em>LASTCFMT</em></td>
<td>Char</td>
<td>32</td>
<td>$</td>
<td>Last formatted value (cutoff)</td>
</tr>
</tbody>
</table>

Note that even though the numbers of columns is fixed, the width of some of the columns in both data tables (for example, _RAWCHAR_ and _CFMT_) is related to the raw and formatted data values of each input variable. The width of these two variables is set to the maximum width over all the variables in the reports.
Figure 24.4  Contents of the Details Output Data Table

<table>
<thead>
<tr>
<th>#</th>
<th>Variable</th>
<th>Type</th>
<th>Len</th>
<th>Format</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><em>VARNAME</em></td>
<td>Char</td>
<td>32</td>
<td>$</td>
<td>Variable name</td>
</tr>
<tr>
<td>2</td>
<td><em>INDEX</em></td>
<td>Num</td>
<td>8</td>
<td>BEST.</td>
<td>Level index</td>
</tr>
<tr>
<td>3</td>
<td><em>FREQ</em></td>
<td>Num</td>
<td>8</td>
<td>BEST.</td>
<td>Level frequency</td>
</tr>
<tr>
<td>4</td>
<td>FREQPERCENT</td>
<td>Num</td>
<td>8</td>
<td>BEST.</td>
<td>Percentage of the frequency of current level</td>
</tr>
<tr>
<td>5</td>
<td>NMISSPERCENT</td>
<td>Num</td>
<td>8</td>
<td>BEST.</td>
<td>Percentage of non missing values</td>
</tr>
<tr>
<td>6</td>
<td><em>RAWNUM</em></td>
<td>Num</td>
<td>8</td>
<td>BEST.</td>
<td>Raw numeric value</td>
</tr>
<tr>
<td>7</td>
<td><em>RAWCHAR</em></td>
<td>Char</td>
<td>32</td>
<td>$</td>
<td>Raw character value</td>
</tr>
<tr>
<td>8</td>
<td><em>FMTWIDTH</em></td>
<td>Num</td>
<td>8</td>
<td>BEST.</td>
<td>Width of the variable formatted value</td>
</tr>
<tr>
<td>9</td>
<td><em>CFMT</em></td>
<td>Char</td>
<td>32</td>
<td>$</td>
<td>Formatted value of a variable</td>
</tr>
</tbody>
</table>

Examples: CARDINALITY Procedure

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

Example 24.1: Limited Cardinality of the Species Variable

This example illustrates how you can use PROC CARDINALITY to determine the limited cardinality of a variable. It uses the same input data table as is used in the section “Getting Started: CARDINALITY Procedure” on page 1121.

You can load the sashelp.iris data set into your CAS session by naming your CAS engine libref in the first statement of the following DATA step:

```sas
data mycas.iris;
  set sashelp.iris;
run;
```

The following statements run PROC CARDINALITY and plot the levels from the details data set for the variable Species:

```sas
proc cardinality data=mycas.iris outcard=mycas.card
  outdetails=mycas.details maxlevels=5;
run;
```

```sas
data sp;
  set mycas.details( where=(_varname_='Species'));
  label _cfmt_='Formatted value of the variable Species';
  if _index_ = . then do;
    _cfmt_=cats(">",left(_cfmt_));
  endif;
run;
```
Example 24.2: Limited Cardinality of the Sepal Length Variable

This example demonstrates how to use PROC CARDINALITY with more levels; it uses the same data table as is used in the section “Getting Started: CARDINALITY Procedure” on page 1121.

You can load the sashelp.iris data set into your CAS session by naming your CAS engine libref in the first statement of the following DATA step:

```plaintext
data mycas.iris;
  set sashelp.iris;
run;
```

The following statements run PROC CARDINALITY and display some of the contents of the cardinality data set:

```plaintext
proc cardinality data=mycas.iris outcard=mycas.card
  outdetails=mycas.details maxlevels=100;
run;

proc print data=mycas.card;
  var _varname_ _type_ _cardinality_ _more_ _visible_ _min_ _max_;
run;
```

Whether MAXLEVELS=100 or 5, the Species variable remains fully visible with a _CARDINALITY_ value of 3.

Output 24.1.1 shows that the variable Species has only three levels in the details data.

Output 24.1.1 Histogram of Species

---

Example 24.2: Limited Cardinality of the Sepal Length Variable

---

Output 24.1.1 Histogram of Species

---

Whether MAXLEVELS=100 or 5, the Species variable remains fully visible with a _CARDINALITY_ value of 3.
Output 24.2.1 shows that all the variables have a _VISIBILITY_ value of 100%.

**Output 24.2.1 Visibility of the Variables**

<table>
<thead>
<tr>
<th>Obs</th>
<th><em>VARNAME</em></th>
<th><em>TYPE</em></th>
<th><em>CARDINALITY</em></th>
<th><em>MORE</em></th>
<th><em>VISIBLE</em></th>
<th><em>MIN</em></th>
<th><em>MAX</em></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Species</td>
<td>C</td>
<td>3 N</td>
<td></td>
<td>100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>SepalLength</td>
<td>N</td>
<td>35 N</td>
<td></td>
<td>100</td>
<td>43</td>
<td>79</td>
</tr>
<tr>
<td>3</td>
<td>SepalWidth</td>
<td>N</td>
<td>23 N</td>
<td></td>
<td>100</td>
<td>20</td>
<td>44</td>
</tr>
<tr>
<td>4</td>
<td>PetalLength</td>
<td>N</td>
<td>43 N</td>
<td></td>
<td>100</td>
<td>10</td>
<td>69</td>
</tr>
<tr>
<td>5</td>
<td>PetalWidth</td>
<td>N</td>
<td>22 N</td>
<td></td>
<td>100</td>
<td>1</td>
<td>25</td>
</tr>
</tbody>
</table>

The following statements extract and display the levels of the variable SepalLength:

```r
data sp;
  label _cfmt_='Formatted Value of the Variable SepalLength';
  set mycas.details( where=(_varname_='SepalLength'));
  if _index_ = . then do;
    _cfmt_=cats(">",left(_cfmt_));
  end;
  _cfmt_=left(_cfmt_);
run;

proc sgplot data=sp;
  vbar _cfmt_ / freq=_freq_; 
run;
```

Output 24.2.2 shows the histogram.

**Output 24.2.2 Full Histogram of the SepalLength Variable**

The following statements specify MAXLEVELS=5, so only the leftmost five levels of the full histogram are displayed. All the other levels are combined into one level, which is reported in the details data table with a missing value in the _INDEX_ column.
Example 24.3: More Levels of the Sepal Length Variable

You can load the sashelp.iris data set into your CAS session by naming your CAS engine libref in the first statement of the following DATA step:

```plaintext
data mycas.iris;
  set sashelp.iris;
run;
```

You can dig deeper by requesting a subsequent segment of the histogram of any variable. This example further examines the SepalLength variable. The cutoff reported from Example 24.2 was 47, which means that all levels less than or equal to 47 have been reported. So you might want to run PROC CARDINALITY again in order to report levels greater than 47, as shown in the following statements:

```plaintext
proc cardinality data=mycas.iris outcard=mycas.card
  outdetails=mycas.details maxlevels=5;
run;

data sp;
  label _cfmt_='Formatted Value of the Variable SepalLength';
  set mycas.details(where=(varname_='SepalLength'));
  if _index_ = . then do;
    _cfmt_=cats(">",left(_cfmt_));
  end;
  _cfmt_=left(_cfmt_);
run;

proc sgplot data=sp;
  vbar _cfmt_ / freq=_freq_;  
run;
```

Output 24.2.3 shows the full histogram of the variable SepalLength. It shows all 35 levels of the variable.

Output 24.2.3 Limited Histogram of the SepalLength Variable

---

Example 24.3: More Levels of the Sepal Length Variable

You can load the sashelp.iris data set into your CAS session by naming your CAS engine libref in the first statement of the following DATA step:

```plaintext
data mycas.iris;
  set sashelp.iris;
run;
```

You can dig deeper by requesting a subsequent segment of the histogram of any variable. This example further examines the SepalLength variable. The cutoff reported from Example 24.2 was 47, which means that all levels less than or equal to 47 have been reported. So you might want to run PROC CARDINALITY again in order to report levels greater than 47, as shown in the following statements:
proc cardinality data=mycas.iris (where=(SepalLength > 47))
    outcard=mycas.card outdetails=mycas.details maxlevels=5;
    var SepalLength;
run;

data sp;
    set mycas.details;
    label _cfmt_="formatted value of the variable SepalLength > 47";
    if _index_ = . then do;
        _cfmt_=cats(">",left(_cfmt_));
    end;
    _cfmt_=left(_cfmt_);
run;

proc sgplot data=sp;
    vbar _cfmt_ / freq=_freq_; run;

Output 24.3.1 Histogram for SepalLength Levels Greater Than 47

The union of the detailed levels from Example 24.2 and this example cover 10 levels of the SepalLength variable.

Table 24.4 lists the variables that are related to the maximum frequency in the histogram.

Table 24.4 More Variables from the Cardinality Data

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>MFREQ</em></td>
<td>Maximum frequency</td>
</tr>
<tr>
<td><em>MFREQFOUNLEVEL</em></td>
<td>Maximum frequency found in the visible part of the report</td>
</tr>
<tr>
<td><em>MFREQNUM</em></td>
<td>Numeric level that has the maximum frequency</td>
</tr>
<tr>
<td><em>MFREQCHR</em></td>
<td>Character level that has the maximum frequency</td>
</tr>
<tr>
<td><em>MFREQFMT</em></td>
<td>Formatted level that has the maximum frequency</td>
</tr>
</tbody>
</table>
The maximum frequency can be attained in the visible part of the histogram, or it can be hidden with the others. If the level that has the maximum frequency is in the visible part, the _MFREQFOUNDLEVEL_ variable is set to Y (Yes); otherwise it is set to N (No). This information is reported here because it is useful for subsequent analysis.

The following statements illustrate the variables that pertain to the maximum frequency of each variable.

```plaintext
proc cardinality data=mycas.iris outcard=mycas.card
   outdetails=mycas.details maxlevels=5;
run;

proc print data=mycas.card;
   var _varname_ _mf:;
run;
```

**Output 24.3.2** Maximum Frequency Information

<table>
<thead>
<tr>
<th>Obs</th>
<th><em>VARNAME</em></th>
<th><em>MFREQ</em></th>
<th><em>MFREQFOUNDLEVEL</em></th>
<th><em>MFREQNUM</em></th>
<th><em>MFREQCHR</em></th>
<th><em>MFREQCFMT</em></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Species</td>
<td>50</td>
<td>Y</td>
<td>.</td>
<td>Setosa</td>
<td>Setosa</td>
</tr>
<tr>
<td>2</td>
<td>SepalLength</td>
<td>139</td>
<td>N</td>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>SepalWidth</td>
<td>131</td>
<td>N</td>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>PetalLength</td>
<td>126</td>
<td>N</td>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>PetalWidth</td>
<td>101</td>
<td>N</td>
<td>.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For the **Species** variable, the value of _MFREQ_ is 50. Because the **Species** variable has a limited cardinality of 3, _MFREQFOUNDLEVEL_ = Y. Also note that both _MFREQCHR_ and its formatted value in _MFREQCFMT_ are Setosa.

For the **SepalLength** variable, _MFREQ_ is 139 (which occurs for the value _MFREQNUM_ = .), and _MFREQFOUNDLEVEL_ = N (which means that the visible part of the histogram does not contain the level that has the maximum frequency).

---

**Example 24.4: A Variable with a User-Defined Format**

This example demonstrates how PROC CARDINALITY determines how to order variables that have user-defined formats. User-defined formats are considered to be prior knowledge about the variable and are used by default. (You can override this behavior by requesting a specific order in the ORDER= option.)

You can load the sashelp.cars data set into your CAS session by naming your CAS engine libref in the first statement of the following DATA step:

```plaintext
data mycas.cars;
   set sashelp.cars;
run;
```

The following statements create the format on the variable engineSize, run the procedure, and examine some of the variables in the cardinality data set:
proc format casfmtlib='myfmtlib';
  value engsize
    low  - <3  ='toys'
    3    - <6  ='granny'
    6    - high='usable';
run;

data mycas.cars;
  format engineSize engsize.;
  set mycas.cars;
run;

proc cardinality data=mycas.cars outcard=mycas.card
  outdetails=mycas.details maxlevels=5;
  var engineSize;
run;

title 'Cars data with a user-defined format';
proc print data=mycas.card;
  var _varname_ _order_ _more_ _cardinality_;
run;

Output 24.4.1 Summary of the Variable engineSize

Cars data with a user-defined format

<table>
<thead>
<tr>
<th>Obs</th>
<th><em>VARNAME</em></th>
<th><em>ORDER</em></th>
<th><em>MORE</em></th>
<th><em>CARDINALITY</em></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>engineSize</td>
<td>ASCFMT</td>
<td>N</td>
<td>3</td>
</tr>
</tbody>
</table>

The order that is used for the engineSize variable is ascending formatted (ASCFMT), because engineSize has a user-defined format (engsize7). Three levels were found (_CARDINALITY_ = 3). There are no more levels to report (_MORE_ = N). The full histogram is displayed. The following statements produce Output 24.4.2:

proc print data=mycas.details;
  var _varname_ _index_ _freq_ _cfmt_;
run;

Output 24.4.2 Details Data with a User-Defined Format for the Variable engineSize

Cars data with a user-defined format

<table>
<thead>
<tr>
<th>Obs</th>
<th><em>VARNAME</em></th>
<th><em>INDEX</em></th>
<th><em>FREQ</em></th>
<th>CFMT_</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>engineSize</td>
<td>1</td>
<td>242</td>
<td>granny</td>
</tr>
<tr>
<td>2</td>
<td>engineSize</td>
<td>2</td>
<td>178</td>
<td>toys</td>
</tr>
<tr>
<td>3</td>
<td>engineSize</td>
<td>3</td>
<td>8</td>
<td>usable</td>
</tr>
</tbody>
</table>

The order that is used for the engineSize variable is ascending formatted (ASCFMT), because engineSize has a user-defined format engsize7. All the levels are reported. There is no missing values in the _INDEX_ column for the engineSize variable.
Example 24.5: Forcing Another Order on the engineSize Variable

You can load the sashelp.cars data set into your CAS session by naming your CAS engine libref in the first statement of the following DATA step:

``` SAS
data mycas.cars;
  set sashelp.cars;
run;
```

The following statements specify the ORDER= option in the VAR statement in order to use a different levelization order:

``` SAS
title 'Cars data with a user-defined format with an ASC order';
proc cardinality data=mycas.cars outcard=mycas.card
  outdetails=mycas.details maxlevels=5;
  var engineSize /order=asc;
run;
```

``` SAS
proc print data=mycas.card ;
  var _varname_ _order_ _more_ _cardinality_;
run;
```

**Output 24.5.1** Summary of the engineSize Variable without a Format

**Cars data with a user-defined format with an ASC order**

<table>
<thead>
<tr>
<th>Obs</th>
<th><em>VARNAME</em></th>
<th><em>ORDER</em></th>
<th><em>MORE</em></th>
<th><em>CARDINALITY</em></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>EngineSize</td>
<td>ASC</td>
<td>Y</td>
<td>5</td>
</tr>
</tbody>
</table>

The order that is used in the engineSize variable is ascending unformatted, as requested by the ORDER=ASC option in the VAR statement. The reported levels are based on the numeric raw values of the engineSize variable. The format engsize7. is ignored. The raw values are more than 5 (the value of the MAXLEVEL= option). The details data table contains six rows: five for reported known levels and one for all the remaining unreported levels ). The cardinality data table reports _MORE_ = Y (Yes) and _CARDINALITY_ = 5 for the engineSize variable. The details of the variable engineSize are discussed next, starting with the details data set produced in the previous step:

``` SAS
data details(drop=_rawchar_);
  set mycas.details;
  if _index_ = . then do;
    _cfmt_ = cats">", put(_rawnum_ ,best12.));
    _rawnum_ = .;
  end;
  _cfmt_ = left(_cfmt_); 
run;
```

``` SAS
proc print data=details;
run;
```
Output 24.5.2 Details of the engineSize Variable without a Format

Cars data with a user-defined format with an ASC order

<table>
<thead>
<tr>
<th>Obs</th>
<th>VARNAME</th>
<th>INDEX</th>
<th>FREQ</th>
<th>FREQPERCENT</th>
<th>NMISSPERCENT</th>
<th>RAWNUM</th>
<th><em>FMTWIDTH</em></th>
<th>CFMT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>EngineSize</td>
<td>1</td>
<td>2</td>
<td>0.4672897196</td>
<td>0.4672897196</td>
<td>1.3</td>
<td>12</td>
<td>1.3</td>
</tr>
<tr>
<td>2</td>
<td>EngineSize</td>
<td>2</td>
<td>1</td>
<td>0.2336448598</td>
<td>0.2336448598</td>
<td>1.4</td>
<td>12</td>
<td>1.4</td>
</tr>
<tr>
<td>3</td>
<td>EngineSize</td>
<td>3</td>
<td>6</td>
<td>1.4018691589</td>
<td>1.4018691589</td>
<td>1.5</td>
<td>12</td>
<td>1.5</td>
</tr>
<tr>
<td>4</td>
<td>EngineSize</td>
<td>4</td>
<td>10</td>
<td>2.3364485981</td>
<td>2.3364485981</td>
<td>1.6</td>
<td>12</td>
<td>1.6</td>
</tr>
<tr>
<td>5</td>
<td>EngineSize</td>
<td>5</td>
<td>4</td>
<td>0.9345794393</td>
<td>0.9345794393</td>
<td>1.7</td>
<td>12</td>
<td>1.7</td>
</tr>
<tr>
<td>6</td>
<td>EngineSize</td>
<td>.</td>
<td>405</td>
<td>94.626168224</td>
<td>94.626168224</td>
<td>.</td>
<td>12</td>
<td>&gt;1.7</td>
</tr>
</tbody>
</table>

The order that is used in the `engineSize` variable is ascending unformatted, as requested by the ORDER=ASC option in the VAR statement. The reported levels are based on the numeric raw value of the `engineSize` variable in ascending order. The format `engsize7.` is ignored. Five raw values are displayed with one additional level, indicating that the histogram of the variable is partial and not full. The observation for which `_INDEX_ = .` indicates that there are still unreported levels in the raw values. A second pass with a WHERE clause can reveal the details of the levels.

References

Chapter 25
The PARTITION Procedure

Overview: PARTITION Procedure

The PARTITION procedure performs sampling in SAS Viya. It performs simple random sampling, stratified sampling, oversampling, or $k$-fold partitioning to produce a table that contains a subset of the observations or that contains partitioned observations.

The PARTITION procedure creates the following:

- one output data table, which contains the subset or partitioned data table
- one summary table, which contains the numbers of observations and variables in the output data table
- one frequency table, which contains the frequency information for the population and sample or for each fold for $k$-fold partitioning
Using CAS Sessions and CAS Engine Librefs

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:

```
proc options option=(CASHOST CASPORT);
run;
```

In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:

```
cas mysess;
libname mycas cas sessref=mysess;
```

The CAS statement creates the CAS session named `mysess`, and the LIBNAME statement creates the `mycas` CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the CASHOST and CASPORT of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the `mysess` session, you can terminate it by using the TERMINATE option in the CAS statement as follows:

```
cas mysess terminate;
```

For more information about the CAS and LIBNAME statements, see the section “Introduction to Shared Concepts” on page 10 in Chapter 3, “Shared Concepts.”

Getting Started: PARTITION Procedure

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”
This example performs stratified partitioning of 5,960 fictitious mortgages, with the BY variable BAD used as the stratum. The input data table mycas.hmeq includes information about the fictitious mortgages. Each observation represents an applicant for a home equity loan, and all applicants have an existing mortgage.

You can load the sampsio.hmeq data set into your CAS session by naming your CAS engine libref in the first statement of the following DATA step. This DATA step assumes that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

```latex
\begin{verbatim}
data mycas.hmeq;
  set sampsio.hmeq;
run;
\end{verbatim}
```

The following statements perform the partitioning:

```latex
\begin{verbatim}
proc partition data=mycas.hmeq samppct=10 samppct2=20 seed=1234 partind;
  by BAD;
  output out=mycas.out1 copyvars=(BAD loan derog mortdue value yoj delinq clage ninq clno debtinc);
run;
\end{verbatim}
```

The SAMPPCT=10 option requests that 10% of the input data be included in the training partition, and the SAMPPCT2=20 option requests that 20% of the input data be included in the testing partition. The SEED= option specifies 1234 as the random seed to be used in the partitioning process. The PARTIND option requests that the output data table, mycas.out1, include an indicator that shows whether each observation is selected to a partition (1 for training or 2 for testing) or not (0). The binary BY variable BAD indicates whether an applicant eventually defaulted or was ever seriously delinquent. The BY statement triggers stratified sampling, which enables you to sample each subpopulation in the BY variable (stratum) independently. The OUTPUT statement creates a new data table to contain the variables from the input data table that are listed in the COPYVARS= option and the partition indicator. The displayed output includes a frequency table (Figure 25.1) that shows the frequency of observations in each level of BAD.

**Figure 25.1 Frequency Information Table**

<table>
<thead>
<tr>
<th>Stratified Sampling Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Index</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>1</td>
</tr>
</tbody>
</table>
Syntax: PARTITION Procedure

The following statements are available in the PARTITION procedure:

```
PROC PARTITION <options> ;
   BY variable < variable ... variable > ;
   OUTPUT OUT=CAS-libref.data-table <options> ;
   DISPLAY <table-list> < /options> ;
   DISPLAYOUT table-spec-list < /options> ;
```

The BY statement is required for stratified sampling and oversampling. The OUTPUT statement is required for all sampling methods.

PROC PARTITION statement

```
PROC PARTITION <options> ;
```

The PROC PARTITION statement invokes the procedure.

You can specify the following `options` for all types of sampling:

- `DATA=CAS-libref.data-table` names the input data table for PROC PARTITION to use. `CAS-libref.data-table` is a two-level name, where
  - `CAS-libref` refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to the data, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about `CAS-libref`, see the section “Using CAS Sessions and CAS Engine Librefs” on page 1138.
  - `data-table` specifies the name of the input data table.

- `NTHREADS=number-of-threads` specifies the number of threads to be used. The default is the CPU count on each node.

- `PARTIND` adds to the output data table a partition indicator, _PartInd_, which indicates whether an observation is selected to a partition (1 or 2) or not (0).

- `SEED=random-seed` specifies an integer that is used to start the pseudorandom number generator. If you do not specify a seed or you specify a value less than or equal to 0, the seed is generated from reading the time of day from the computer’s clock. The SEED= option enables you to reproduce the same sample output.

You can specify the following `options` only for simple random sampling and stratified sampling:
**SAMPPCT=** `sample-percentage`

specifies what percentage of the data is to be sampled, where `sample-percentage` is a positive number less than or equal to 100. For example, `SAMPPCT=50.5` samples 50.5% of data. When you specify this option along with the `PARTIND` option, the percentage of the sample whose `_PartInd_` is 1 is specified in `sample-percentage`, and the percentage of the sample whose `_PartInd_` is 0 is 100 minus `sample-percentage`.

**SAMPPCT2=** `sample-percentage-2`

partitions the input data into three parts when specified along with the `SAMPPCT=` and `PARTIND` options. The percentage of the sample whose `_PartInd_` is 1 is specified in the `SAMPPCT=` option, the percentage of the sample whose `_PartInd_` is 2 is specified in the `SAMPPCT2=` option, and the percentage of the sample whose `_PartInd_` is 0 is 100 minus the sum of the values of the `SAMPPCT=` and `SAMPPCT2=` options. The sum of the `sample-percentages` specified in the `SAMPPCT=` and `SAMPPCT2=` options must be a positive number less than or equal to 100.

The following formulas are used to calculate sample sizes for partitions:

\[
\begin{align*}
n &= \text{round}((\text{sample-percentage} + \text{sample-percentage-2}) \times N) \\
n_1 &= \text{round}((\text{sample-percentage}/(\text{sample-percentage} + \text{sample-percentage-2})) \times n) \\
n_2 &= n - n_1
\end{align*}
\]

where \( N \) is the number of observations in the input data, \( n \) is the total number of observations in the partitions, \( n_1 \) is the number of observations in the first sample, and \( n_2 \) is the number of observations in the second sample.

You can specify the following options only for oversampling:

**EVENT=** "rare-event-level"

specifies the level (category) that corresponds to a rare event of the variable that is specified in the BY statement. If you have a format applied to the variable, you should specify the formatted event level in this option. If you specify this option, PROC PARTITION uses an oversampling technique to adjust the class distribution of the data, and the following two options are required.

**EVENTPROP=** `event-proportion`

specifies the proportion of rare events that you want in the sample, where `event-proportion` is a positive number less than or equal to 1. For example, `EVENTPROP=0.3` specifies that 30% of the sample are rare events and 70% are not rare.

**SAMPPCTEVT=** `sample-event-percentage`

specifies the sample percentage from the rare event level, where `sample-event-percentage` is a positive number less than or equal to 100. For example, `SAMPPCTEVT=50.5` specifies that you want to sample 50.5% of the rare events.

You can specify the following option only for \( k \)-fold partitioning:

**KFOLD=** \( k \)

specifies the number of folds you want to partition the data into, where \( k \) is a positive integer greater than or equal to 2.
BY Statement

```
BY variable < variable . . . variable > ;
```

The BY statement specifies classification variables to be used for stratification. There is no limit on the BY variables for stratified sampling or $k$-fold partitioning, but only one BY variable is allowed for oversampling.

DISPLAY Statement

```
DISPLAY < table-list > < / options > ;
```

The DISPLAY statement enables you to specify a list of display tables to display or exclude. This statement is similar to the ODS SELECT, ODS EXCLUDE, and ODS TRACE statements. However, the DISPLAY statement can improve performance when a large number of tables could be generated (such as in BY-group processing). The procedure processes the DISPLAY statement on a CAS server and thus sends only a subset of ODS tables to the SAS client. Because ODS statements are processed on a SAS client, first all the generated display tables are sent to the client, and then the client creates a subset.

If you use both DISPLAY and ODS statements together, the DISPLAY statement takes precedence over the ODS statements. Note that the ODS EXCLUDE statement processes tables that are sent to the client after they have been filtered by the DISPLAY statement. In some cases, it might appear that the ODS EXCLUDE statement is taking precedence because it can further filter the tables. For more information about ODS, see SAS Output Delivery System: Procedures Guide.

You can specify the `table-list` as a list of table names, paths, partial pathnames, and regular expressions.

The table names that you can specify are listed in the section “ODS Table Names” on page 1146. A path is a table name that is prefixed with dot-separated grouping information. For example, a SelectionSummary table that a procedure produces during a selection routine might have the path `Bygroup1.Summary.SelectionSummary`. A partial pathname does not include all groups; for example, `SelectionSummary` and `Summary.SelectionSummary` are partial pathnames for `Bygroup1.Summary.SelectionSummary`.

When you specify a table name or partial pathname, all display tables whose paths end in the specified name are selected for display or exclusion. For example, both `SelectionSummary` and `Summary.SelectionSummary` select `Bygroup1.Summary.SelectionSummary`.

A regular expression is enclosed in forward slashes (/). For example, specifying “/tions/” selects all pathnames that contain the substring “tions”; in particular, the `Bygroup1.Summary.SelectionSummary` table is selected. Specifying “!/tions/” selects all pathnames that do not contain the substring “tions”; in particular, the `Bygroup1.Summary.SelectionSummary` table is not selected.

You can specify the following `options` after a slash (/):

**CASESENSITIVE**

performs a case-sensitive comparison of table names in the `table-list` to display table names when tables are subsetted for display. To preserve case, you must enclose table names in the `table-list` in quotation marks.
**DISPLAYOUT Statement**

**DISPLAYOUT** table-spec-list < options > ;

The DISPLAYOUT statement enables you to create CAS output tables from your displayed output. This statement is similar to the ODS OUTPUT statement. For more information about ODS, see *SAS Output Delivery System: Procedures Guide*.

The table-spec-list specifies a list of CAS output tables to create. Each entry in the list has either a **key=value** format or a **key** format:

- **key=value** specifies **key** as the ODS table name, path, or partial pathname, and specifies **value** as the CAS output table name.
- **key** specifies **key** as the ODS table name and also as the CAS output table name.

The ODS table names that you can specify are listed in the section “**ODS Table Names**” on page 1146. You cannot specify the ODS table named OutputCasTables in the table-spec-list.

Table names and partial pathnames are discussed under the **DISPLAY** statement. The DISPLAYOUT statement does not support regular expressions.

You can specify the following **options** after a slash (/):

- **INCLUDEALL** creates output CAS tables for all display tables. The name of the created output CAS table is the same as the corresponding display table name. If you specify this option, the table-spec-list specification is ignored.

- **NOREPLACE** does not replace any existing CAS output table of the same name.

- **REPEATED** replicates all CAS output tables on all nodes.

**OUTPUT Statement**

**OUTPUT OUT=CAS-libref.data-table < options > ;**

The OUTPUT statement creates a data table to contain the sampled observations and stratification information for the sample.
You must specify the following option:

**OUT=** *CAS-libref.data-table*

names the output data table for PROC PARTITION to use. You must specify this option before any other options. *CAS-libref.data-table* is a two-level name, where

*CAS-libref* refers to a collection of information that is defined in the LIBNAME statement and includes the *caslib*, which includes a path to where the data table is to be stored, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about *CAS-libref*, see the section “Using CAS Sessions and CAS Engine Librefs” on page 1138.

*data-table* specifies the name of the output data table.

This table includes variables that are specified in the COPYVARS= option. In the K-fold partition case, the output includes one more column, _Fold_. In other cases, if you also specify the PARTIND option in the PROC PARTITION statement, the output includes one more column, _PartInd_. In the oversampling case, an additional column, _Freq_, is provided. It is calculated as the ratio of target level’s proportion in the population to its proportion in the sample. _Fold_, _PartInd_ and _Freq_ are reserved names. If they exist in the input data table and you specify them in the COPYVARS= option, you need to use the FOLDNAME=, PARTINDNAME= and FREQNAME= options to change the generated columns’ names.

You can also specify the following options:

**COPYVAR=** *variable*

**COPYVARS=(** *variables**)

lists one or more variables from the input data table to be transferred to the output data table.

If the COPYVARS= option is not specified, all the variables in the input data table are transferred to the output data table.

**FOLDNAME=** *fold-name*

renames the output data table’s _Fold_ column (which is generated when k-fold partitioning is requested) to the specified *fold-name*.

**FREQNAME=** *freq-name*

renames the output data table’s _Freq_ column (which is generated when oversampling is requested) to the specified *freq-name*.

**PARTINDNAME=** *partition-indicator-name*

renames the generated partition indicator, _PartInd_, in the output data table to the specified *partition-indicator-name*. 
Details: PARTITION Procedure

Folds

You can use the PARTITION procedure to partition the input data table into equal-size disjoint subsets for k-fold cross validation. When you specify the number of folds $k$ in the KFOLD= option, the _Fold_ column in the output data table shows which fold each observation is selected to. _Fold_ takes values between 1 and $k$.

Training, Validation, and Test Data

You can use the PARTITION procedure to partition the input data table into disjoint subsets for model training, validation, and testing. When you specify the percentages for training and testing partitions in the SAMPPCT= and SAMPPCT2= options, the _PartInd_ indicator shows whether each observation is selected to the training partition (1) or the testing partition (2), respectively. The remaining observations are assigned 0 for _PartInd_ and belong to the validation partition.

Classification Level

For classification variables, a classification level is an observed value that is distinct after beginning and ending white space have been removed. For example, the values “MyLevel” and “MYLEVEL” are treated as different levels in the data table. But, “a” and “a ” are treated as the same level.

Displayed Output

The following sections describe the output that PROC PARTITION produces. The output is organized into various tables, which are discussed in the order of their appearance.

Frequency Information Table

For simple random sampling, the “Simple Random Sampling Frequency” table lists the number of observations in the input data table and in the sample output data table.

For stratified sampling, the “Stratified Sampling Frequency” table lists the respective frequency in each stratum for the input data and the sample. If one BY variable is specified, each level of the BY variable represents a stratum; if two BY variables are specified, a combination of the levels of two BY variables represents a stratum.

For oversampling, the “Oversampling Frequency” table lists the number of observations in each stratum for the input data and the sample.
OutputCasTables Table

The OutputCasTables table is a special table that has information about each CAS table that is created during a CAS action execution. The information for each CAS table consists of the CAS table name, the caslib in which the table resides, and the number of columns and rows in the CAS table. Because this table is not a typical ODS table that contains analytical results, you cannot include it in the `table-spec-list` in the `DISPLAYOUT` statement.

ODS Table Names

Each table that the PARTITION procedure creates has a name associated with it. You must use this name to refer to the table when you use the DISPLAY statement or ODS statements. These names are listed in Table 25.1.

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>KFOLDFreq</td>
<td>Frequency table of input data size and fold sizes</td>
<td>PROC PARTITION</td>
<td>Default</td>
</tr>
<tr>
<td>OutputCasTables</td>
<td>See the section “OutputCasTables Table” on page 1146</td>
<td>OUTPUT OUT=</td>
<td></td>
</tr>
<tr>
<td>OVERFreq</td>
<td>Frequency table of input data size and sample size in different levels of the BY variable for the oversampling level of the BY variable (this table is produced for oversampling)</td>
<td>PROC PARTITION</td>
<td>Default</td>
</tr>
<tr>
<td>SRSFreq</td>
<td>Frequency table of input data size and sample size (this table is produced for simple random sampling)</td>
<td>PROC PARTITION</td>
<td>Default</td>
</tr>
<tr>
<td>STRAFreq</td>
<td>Frequency table of input data size and sample size in different stratification levels that are defined by BY variables (this table is produced for stratified sampling)</td>
<td>PROC PARTITION</td>
<td>Default</td>
</tr>
</tbody>
</table>
Examples: PARTITION Procedure

NOTE: Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

Example 25.1: Simple Random Sampling

This example demonstrates how to use PROC PARTITION to perform simple random sampling on the mycas.hmeq data table.

You can load the sampsio.hmeq data set into your CAS session by naming your CAS engine libref in the first statement of the following DATA step. This DATA step assumes that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

```sas
data mycas.hmeq;
  set sampsio.hmeq;
run;
```

The following statements perform the sampling:

```sas
proc partition data=mycas.hmeq samppct=10 seed=10 nthreads=1;
  output out=mycas.out2 copyvars=(job reason loan value delinq derog);
  display 'SRSFreq';
run;
```

```sas
proc print data=mycas.out2(obs=20);
run;
```

The SAMPPCT=10 option requests that 10% of the input data be sampled. The OUTPUT statement requests that the sampled data be stored in a table named mycas.out2, and the COPYVARS= option lists the variables to be copied from mycas.hmeq to mycas.out2. The DISPLAY statement requests that the SRSFreq ODS table be displayed.

Output 25.1.1 shows the number of observations in the mycas.hmeq data table and the number of samples in the mycas.out2 data table.

### Output 25.1.1 Frequency Information Table

<table>
<thead>
<tr>
<th>Simple Random Sampling Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Obs</td>
</tr>
<tr>
<td>----------------</td>
</tr>
<tr>
<td>5960</td>
</tr>
</tbody>
</table>

Output 25.1.2 shows the sample data, which are stored in the mycas.out2 data table.
Example 25.2: Stratified Sampling

This example demonstrates how to use PROC PARTITION to perform stratified sampling to partition the data; it uses the same data table as is used in Example 25.1.

You can load the sampsio.hmeq data set into your CAS session by naming your CAS engine libref in the first statement of the following DATA step. This DATA step assumes that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

```plaintext
data mycas.hmeq;
  set sampsio.hmeq;
run;
```

The following statements perform the partitioning:

```plaintext
proc partition data=mycas.hmeq samppct=10 samppct2=20 seed=10 partind nthreads=3;
  by BAD;
  output out=mycas.out3 copyvars=(job reason loan value delinq derog);
run;
```

```plaintext
proc print data=mycas.out3(obs=20);
run;
```

The SAMPPCT=10 option requests that 10% of the input data be included in the training partition, and the SAMPPCT2=20 option requests that 20% of the input data be included in the testing partition. The SEED= option specifies 10 as the random seed to be used in the partitioning process. The PARTIND option requests that the output data table, mycas.out3, include an indicator that shows whether each observation is selected.
to a partition (1 for training or 2 for testing) or not (0). The OUTPUT statement requests that the sampled data be stored in a table named mycas.out3, and the COPYVARS= option lists the variables to be copied from mycas.hmeq to mycas.out3.

Output 25.2.1 shows the frequency information for each level of the BY variable BAD in the mycas.out3 data table.

Output 25.2.1 Frequency Information Table

The PARTITION Procedure

<table>
<thead>
<tr>
<th>Stratified Sampling Frequency</th>
<th>Number of Obs</th>
<th>Sample Size 1</th>
<th>Sample Size 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Index BAD</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 0</td>
<td>4771</td>
<td>477</td>
<td>954</td>
</tr>
<tr>
<td>1 1</td>
<td>1189</td>
<td>119</td>
<td>238</td>
</tr>
</tbody>
</table>

Output 25.2.2 shows the first 20 output sample observations in mycas.out3; the _PartInd_ column shows which partition the observation is selected for (1 for training or 2 for testing) or none (0).

Output 25.2.2 Sample Output with Partition Indicator

<table>
<thead>
<tr>
<th>Obs</th>
<th>JOB</th>
<th>REASON</th>
<th>LOAN</th>
<th>VALUE</th>
<th>DELINQ</th>
<th>DEROG</th>
<th><em>PartInd</em></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Other</td>
<td>Homelmp</td>
<td>1100</td>
<td>39025</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td>1500</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>Other</td>
<td>Homelmp</td>
<td>1800</td>
<td>57037</td>
<td>2</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>Sales</td>
<td>Homelmp</td>
<td>2000</td>
<td>62250</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>Other</td>
<td>Homelmp</td>
<td>2000</td>
<td>55000</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>Other</td>
<td>Homelmp</td>
<td>2200</td>
<td>34687</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>Other</td>
<td>Homelmp</td>
<td>2300</td>
<td>40150</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>ProfExe</td>
<td>Homelmp</td>
<td>2400</td>
<td>73395</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>Other</td>
<td>Homelmp</td>
<td>2400</td>
<td>17180</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>Homelmp</td>
<td></td>
<td>2500</td>
<td>20200</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>11</td>
<td>ProfExe</td>
<td>Homelmp</td>
<td>2500</td>
<td>78600</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>ProfExe</td>
<td>DebtCon</td>
<td>2900</td>
<td>113000</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>13</td>
<td>Other</td>
<td>Homelmp</td>
<td>2900</td>
<td>67996</td>
<td>0</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>Other</td>
<td>Homelmp</td>
<td>3000</td>
<td>20300</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>15</td>
<td>Other</td>
<td>Homelmp</td>
<td>3000</td>
<td>193500</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>16</td>
<td>Other</td>
<td>Homelmp</td>
<td>3000</td>
<td>14100</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>17</td>
<td>Mgr</td>
<td>Homelmp</td>
<td>3000</td>
<td>71500</td>
<td>2</td>
<td>.</td>
<td>2</td>
</tr>
<tr>
<td>18</td>
<td></td>
<td></td>
<td>3100</td>
<td>70400</td>
<td>.</td>
<td>.</td>
<td>0</td>
</tr>
<tr>
<td>19</td>
<td>Other</td>
<td>Homelmp</td>
<td>3200</td>
<td>40834</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>20</td>
<td>Mgr</td>
<td>Homelmp</td>
<td>3200</td>
<td>.</td>
<td>2</td>
<td>.</td>
<td>0</td>
</tr>
</tbody>
</table>
Example 25.3: Oversampling

This example demonstrates how to use PROC PARTITION to perform oversampling; it uses the same data table as in Example 25.1.

You can load the sampsio.hmeq data set into your CAS session by naming your CAS engine libref in the first statement of the following DATA step. This DATA step assumes that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

```sas
data mycas.hmeq;
    set sampsio.hmeq;
run;
```

The following statements perform oversampling:

```sas
proc partition data=mycas.hmeq samppctevt=90 eventprop=0.5
    event="1" seed=10 nthreads=1;
    by BAD;
    ods output OVERFREQ=outFreq;
    output out=mycas.out4 copyvars=(job loan value delinq derog)
        freqname=_Freq2_;
run;
```

```sas
proc print data=mycas.out4(obs=20);
run;
```

The EVENTPROP=0.5 option specifies that 50% of the sample are rare events. The SAMPPCTEVT=90 option requests that 90% of the rare events be sampled. The EVENT="1" option specifies that the second level of the variable BAD corresponds to a rare event. The OUTPUT statement requests that the sampled data be stored in a table named mycas.out4, specifies the variables to be transferred from the input data table, requests that the _Freq_ column be renamed to _Freq2_.

Output 25.3.1 shows the number of observations in the sample and in each level of the BY variable BAD in the mycas.out4 data table.

**Output 25.3.1** Frequency Information Table

<table>
<thead>
<tr>
<th>The PARTITION Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Oversampling Frequency</strong></td>
</tr>
<tr>
<td>Number</td>
</tr>
<tr>
<td>Index</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>1</td>
</tr>
</tbody>
</table>

Output 25.3.2 shows the first 20 output sample observations in the mycas.out4 data table; the _Freq2_ column shows the ratio of target level’s proportion in the population to its proportion in the sample.
Example 25.4: k-Fold Partitioning

This example demonstrates how to use PROC PARTITION to perform stratified k-fold partitioning to partition the data; it uses the same data table as is used in Example 25.1.

You can load the sampsio.hmeq data set into your CAS session by naming your CAS engine libref in the first statement of the following DATA step. This DATA step assumes that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

```plaintext
data mycas.hmeq;
  set sampsio.hmeq;
run;
```

The following statements perform the partitioning:

```plaintext
proc partition data=mycas.hmeq kfold=10 seed=10 nthreads=3;
  by BAD;
  output out=mycas.out5 copyvars=(job reason loan value delinq derog);
run;
```

```plaintext
proc print data=mycas.out5(obs=20);
run;
```

The KFOLD=10 option requests that the input data be partitioned to 10 equal-size folds within each level of the BY variable BAD. The SEED= option specifies 10 as the random seed to be used in the partitioning process. The OUTPUT statement requests that the sampled data be stored in a table named mycas.out5, and
the COPYVARS= option lists the variables to be copied from mycas.hmeq to mycas.out5. The output data table, mycas.out5, includes an indicator that shows which fold each observation is selected to.

Output 25.4.1 shows the frequency information for each level of the BY variable BAD in the mycas.out5 data table.

### Output 25.4.1 Frequency Information Table

<table>
<thead>
<tr>
<th>Index</th>
<th>BAD</th>
<th>Number of Obs</th>
<th>Fold Size 1</th>
<th>Fold Size 2</th>
<th>Fold Size 3</th>
<th>Fold Size 4</th>
<th>Fold Size 5</th>
<th>Fold Size 6</th>
<th>Fold Size 7</th>
<th>Fold Size 8</th>
<th>Fold Size 9</th>
<th>Fold Size 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>4771</td>
<td>478</td>
<td>477</td>
<td>477</td>
<td>477</td>
<td>477</td>
<td>477</td>
<td>477</td>
<td>477</td>
<td>477</td>
<td>477</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1189</td>
<td>119</td>
<td>118</td>
<td>119</td>
<td>119</td>
<td>119</td>
<td>119</td>
<td>119</td>
<td>119</td>
<td>119</td>
<td>119</td>
</tr>
</tbody>
</table>

Output 25.4.2 shows the first 20 output sample observations in mycas.out5; the _Fold_ column shows which fold the observation is selected to.

### Output 25.4.2 k-Fold Partitioning Output Data Table

<table>
<thead>
<tr>
<th>Obs</th>
<th>JOB</th>
<th>REASON</th>
<th>LOAN</th>
<th>VALUE</th>
<th>DELINQ</th>
<th>DEROG</th>
<th><em>Fold</em></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Other</td>
<td>Homelmp</td>
<td>1100</td>
<td>39025</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td>1500</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>Other</td>
<td>Homelmp</td>
<td>1800</td>
<td>57037</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>Sales</td>
<td>Homelmp</td>
<td>2000</td>
<td>62250</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>Other</td>
<td>Homelmp</td>
<td>2000</td>
<td>55000</td>
<td>0</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>Other</td>
<td>Homelmp</td>
<td>2200</td>
<td>34687</td>
<td>1</td>
<td>0</td>
<td>9</td>
</tr>
<tr>
<td>7</td>
<td>Other</td>
<td>Homelmp</td>
<td>2300</td>
<td>40150</td>
<td>0</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>8</td>
<td>ProfExe</td>
<td>Homelmp</td>
<td>2400</td>
<td>73395</td>
<td>0</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>9</td>
<td>Other</td>
<td>Homelmp</td>
<td>2400</td>
<td>17180</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>10</td>
<td>Homelmp</td>
<td></td>
<td>2500</td>
<td>20200</td>
<td>0</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>11</td>
<td>ProfExe</td>
<td>Homelmp</td>
<td>2500</td>
<td>78600</td>
<td>0</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>12</td>
<td>ProfExe</td>
<td>DebtCon</td>
<td>2900</td>
<td>113000</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>13</td>
<td>Other</td>
<td>Homelmp</td>
<td>2900</td>
<td>67996</td>
<td>0</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>14</td>
<td>Other</td>
<td>Homelmp</td>
<td>3000</td>
<td>20300</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>15</td>
<td>Other</td>
<td>Homelmp</td>
<td>3000</td>
<td>193500</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>16</td>
<td>Other</td>
<td>Homelmp</td>
<td>3000</td>
<td>14100</td>
<td>0</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>17</td>
<td>Mgr</td>
<td>Homelmp</td>
<td>3000</td>
<td>71500</td>
<td>2</td>
<td>.</td>
<td>2</td>
</tr>
<tr>
<td>18</td>
<td></td>
<td></td>
<td>3100</td>
<td>70400</td>
<td>.</td>
<td>.</td>
<td>4</td>
</tr>
<tr>
<td>19</td>
<td>Other</td>
<td>Homelmp</td>
<td>3200</td>
<td>40834</td>
<td>0</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>20</td>
<td>Mgr</td>
<td>Homelmp</td>
<td>3200</td>
<td>.</td>
<td>2</td>
<td>.</td>
<td>4</td>
</tr>
</tbody>
</table>
Chapter 26
The VARIMPUTE Procedure

Overview: VARIMPUTE Procedure

The VARIMPUTE procedure performs variable imputation in SAS Viya. Imputation is a common step in data preparation.

The VARIMPUTE procedure can replace numeric missing values with a specified value, with the mean or median of the nonmissing values, or with some random value between the minimum value and the maximum value of the nonmissing values.

The VARIMPUTE procedure can replace nominal missing values with a specified value and with the nonmissing value which has the most frequent appearance for that variable.

The VARIMPUTE procedure also supports formatted values including user defined formats.

When PROC VARIMPUTE calculates the mean, median, mode, or a random value, it ignores any observation that has a value that is less than or equal to 0 for any variable that is named in the FREQ statement.
Using CAS Sessions and CAS Engine Librefs

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:

```plaintext
proc options option=(CASHOST CASPORT);
run;
```

In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:

```plaintext
cas mysess;
libname mycas cas sessref=mysess;
```

The CAS statement creates the CAS session named `mysess`, and the LIBNAME statement creates the `mycas` CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the CASHOST and CASPORT of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the `mysess` session, you can terminate it by using the TERMINATE option in the CAS statement as follows:

```plaintext
cas mysess terminate;
```

For more information about the CAS and LIBNAME statements, see the section “Introduction to Shared Concepts” on page 10 in Chapter 3, “Shared Concepts.”

Getting Started: VARIMPUTE Procedure

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

The VARIMPUTE procedure can use any or all of six methods to impute numeric missing values. This example uses all six imputation methods to manipulate a data table. The input data table `mycas.hmeq` includes information about fictitious mortgages. Each observation represents an applicant for a home equity loan, and all applicants have an existing mortgage.

You can load the `sampsio.hmeq` data table into your CAS session by naming your CAS engine libref in the first statement of the following DATA step:
These statements assume that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

The following statements perform the imputation:

```sas
proc varimpute data=mycas.hmeq seed=18000;
   input clage /ctech=mean;
   input delinq/ctech=median;
   input ninq/ctech=random;
   input reason/ntech=mode;
   input job/ntech=value valuescharacter=Office;
   input debtinc yoj/ctech=value cvalues=50,100;
   output out=mycas.out;
run;
```

The SEED= option specifies 18,000 as the random seed to be used in the random imputation process. The first INPUT statement (which includes the CTECH=MEAN option) requests that mean imputation be used for the variable clage. The second INPUT statement (which includes the CTECH=MEDIAN option) requests that median imputation be used for the variable delinq. The third INPUT statement (which includes the CTECH=RANDOM option) requests that random imputation be used for the variable ninq. The fourth INPUT statement (which includes the NTECH=MODE option) requests that mode imputation be used for the variable reason. The fifth INPUT statement (which includes the NTECH=VALUE option) requests that Office be imputed as the value of the variable job. The sixth INPUT statement (which includes the CTECH=VALUE option) requests that 50 be imputed as the value of variable debtinc and 100 be imputed as the value of variable yoj. The OUTPUT statement creates a new data table to contain the imputed values; it is shown in Output 26.1 and Output 26.2. The Variable column shows the original variable names from the input data table. The Imputation Method column shows the types of imputation methods: value (for character or numeric variables), median, random (between the minimum value and the maximum value of the nonmissing values), mean, and mode. For random imputation, the last column does not show any number, because the imputation is done by using a random seed. For other imputation methods, the last column shows the imputation values that are used to replace missing values. The Result Variable column shows the new variable names in the output data table.

**Figure 26.1** Imputed Values for Interval Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Imputation Method</th>
<th>Result Variable</th>
<th>N</th>
<th>Number of Missing Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLAGE</td>
<td>Mean</td>
<td>IM_CLAGE</td>
<td>5652</td>
<td>308</td>
</tr>
<tr>
<td>DELINQ</td>
<td>Median</td>
<td>IM_DELINQ</td>
<td>5380</td>
<td>580</td>
</tr>
<tr>
<td>NINQ</td>
<td>Random</td>
<td>IM_NINQ</td>
<td>5450</td>
<td>510</td>
</tr>
<tr>
<td>DEBTINC</td>
<td>Value</td>
<td>IM_DEBTINC</td>
<td>4693</td>
<td>1267</td>
</tr>
<tr>
<td>YOJ</td>
<td>Value</td>
<td>IM_YOJ</td>
<td>5445</td>
<td>515</td>
</tr>
</tbody>
</table>
PROC VARIMPUTE Statement

PROC VARIMPUTE DATA=\texttt{CAS-libref.data-table} \(<\text{options}>\) ;

\begin{itemize}
  \item \texttt{DATA=} \texttt{CAS-libref.data-table} names the input data table for PROC VARIMPUTE to use. The default is the most recently created data table. \texttt{CAS-libref.data-table} is a two-level name, where \texttt{CAS-libref} refers to a collection of information that is defined in the LIBNAME statement and includes the case\texttt{lib}, which includes a path to the data, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about \texttt{CAS-libref}, see the section \textit{“Using CAS Sessions and CAS Engine Librefs”} on page 1154.
  \item \texttt{data-table} specifies the name of the input data table.
\end{itemize}
**CODE Statement**

**FILE=filename ;**

The CODE statement generates SAS DATA step code that mimics the computations that are performed. Only one CODE statement is processed. If you specify multiple CODE statements, only the first one is used.

You can specify the following option:

**FILE=filename**

specifies the name of the file to write the SAS score code to.

**FREQ Statement**

**FREQ variable ;**

The variable in the FREQ statement identifies a numeric variable in the data set that contains the frequency of occurrence of each observation. PROC VARIMPUTE treats each observation as if it appears \( f \) times, where \( f \) is the value of the FREQ variable for the observation. If \( f \) is not an integer, it is truncated to an integer. If \( f \) is less than 1 or missing, the observation is not used in the analysis. When the FREQ statement is not specified, each observation is assigned a frequency of 1.

**INPUT Statement**

**INPUT variables </options> ;**

The INPUT statement names one or more input variables and specifies the method that is used to impute missing values.

You can specify the following options:

- **CONTINUOUSTECH=imputation-method**
- **CTECH=imputation-method**

specifies the imputation method for interval variables.

You can choose from the following methods:
**Chapter 26: The VARIMPUTE Procedure**

**MEAN** specifies that missing values of each variable are replaced by the algebraic mean of the nonmissing values of that variable.

**MEDIAN** specifies that missing values of each variable are replaced by the algebraic median of the nonmissing values of that variable.

**RANDOM** specifies that missing values of each variable are replaced by a random value that is drawn between the minimum value and the maximum values of the nonmissing values of that variable.

**VALUE CVALUES=** specifies that missing values for each numeric input variable are replaced by the corresponding numeric value in the value-list.

By default, **CONTINUOUSTECH=MEAN**.

**NOMINALTECH=** specifies the imputation method for nominal variables.

You can choose from the following **imputation-methods**:

**MODE** replaces missing values of each variable by the nonmissing value that appears most frequently for that variable.

**VALUE VALUESNUMERIC=** replaces missing values for each numeric input variable by the corresponding numeric value in the value-list.

**VALUE VALUESCHARACTER=** replaces missing values for each character input variable by the corresponding character value in the value-list.

By default, **NOMINALTECH=MODE**.

---

**OUTPUT Statement**

**OUTPUT OUT=** creates an output data table to contain the results of PROC VARIMPUTE.

You must specify the following option:

**OUT=** names the output data table for PROC VARIMPUTE to use. You must specify this option before any other options. **CAS-libref.data-table** is a two-level name, where

**CAS-libref** refers to a collection of information that is defined in the LIBNAME statement and includes the `caslib`, which includes a path to where the data table is to be stored, and a session identifier, which defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about **CAS-libref**, see the section “Using CAS Sessions and CAS Engine Librefs” on page 1154.

**data-table** specifies the name of the output data table.

The output data table contains the scored data.
You can also specify the following option:

- COPYVAR= variable
- COPYVARS= (variables)

  lists one or more variables from the input data table to be transferred to the output data table.

---

### Details: VARIMPUTE Procedure

#### Obtaining the Statistics for Imputation

PROC VARIMPUTE first computes the imputation value and then imputes with that value. The statistics mean, median, mode, minimum, and maximum are computed precisely. To compute the median, PROC VARIMPUTE uses the empirical distribution function. Let $n$ be the number of nonmissing values for a variable, and let $x_1, x_2, \ldots, x_n$ represent the ordered values of the variable. Let the 50th percentile be $y$, $p = 0.5$, and $np = j + g$ where $j$ is the integer part of $np$ and $g$ is the fractional part of $np$. Then, the median $y$ is

$$
\begin{align*}
  y &= x_j & \text{if } g = 0 \\
  y &= x_{j+1} & \text{if } g > 0
\end{align*}
$$

#### Random Imputation

If you specify CONTINUOUSTECH=RANDOM in the INPUT statement, PROC VARIMPUTE replaces missing values with a random value that is drawn between the minimum and maximum of the variable. For variable $x$, assume that the input data table is $\{x_i\}$, where $i = 1, 2, \ldots, n$. Let $\min(x) = \min_{i \in \{1 \ldots n\}} \{x_i\}$, and let $\max(x) = \max_{i \in \{1 \ldots n\}} \{x_i\}$. The random value is calculated as $R = \min(x) + (\max(x) - \min(x)) \times \text{rand('uniform')}$, where $\text{rand('uniform')}$ is a function that takes a seed as input and returns a random value from a uniform distribution between 0 and 1. You can specify the seed as a nonnegative integer in the SEED= option in the PROC VARIMPUTE statement. If you do not specify the SEED= option, the seed is generated from reading the time of day from the computer’s clock.

#### Displayed Output

The following sections describe the output that PROC VARIMPUTE produces. The output is organized into various tables, which are discussed in their order of appearance.

#### Imputation Requests

The first table displayed lists the imputation method, the number of variables that are imputed, and the seed value for the random imputation.
Imputed Values for Interval Variables

The “Imputed Values for Interval Variables” table lists the names of the interval variables, the imputation method, the numbers of nonmissing and missing values for each variable, and the imputation values that replace missing values for interval variables.

Imputed Values for Nominal Variables

The “Imputed Values for Nominal Variables” table lists the names of the nominal variables, the imputation method, the numbers of nonmissing and missing values for each variable, and the imputation values that replace missing values for nominal variables.

ODS Table Names

Each table that the VARIMPUTE procedure creates has a name associated with it. You must use this name to refer to the table when you use ODS statements. The name of each table and a short description of the contents are listed in Table 26.1.

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>ImputeInfo</td>
<td>Imputation method, number of variables that are imputed, and seed for the pseudorandom number generator</td>
<td>PROC VARIMPUTE</td>
<td>Default</td>
</tr>
<tr>
<td>VarImputeInfo (Interval Variables)</td>
<td>Variables, imputation method, number of nonmissing values, number of missing values, and imputation values</td>
<td>PROC VARIMPUTE</td>
<td>Default</td>
</tr>
<tr>
<td>VarImputeInfo (Nominal Variables)</td>
<td>Variables, imputation method, number of nonmissing values, number of missing values, and imputation values</td>
<td>PROC VARIMPUTE</td>
<td>Default</td>
</tr>
</tbody>
</table>

Examples: VARIMPUTE Procedure

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”
Example 26.1: Imputation of Missing Values for HMEQ Data Table

This example demonstrates how to use PROC VARIMPUTE to perform imputation; it uses the same data table as in the section “Getting Started: VARIMPUTE Procedure” on page 1154.

You can load the sampsiio.hmeq data table into your CAS session by naming your CAS engine libref in the first statement of the following DATA step:

```plaintext
data mycas.hmeq;
  set sampsiio.hmeq;
  id = _N_;
run;
```

The following statements perform the imputation:

```plaintext
proc varimpute data=mycas.hmeq seed=12345;
  input derog clno/ctech=value cvalues=5,20;
  input value /ctech=mean;
  input job/ntech=mode;
  input mortdue /ctech=median;
  input ninq /ctech=random;
  input reason/ntech=value valuescharacter=unknown;
  output out=mycas.out1 copyvar=(id);
run;
```

```plaintext
data out2; set mycas.out1; run;
proc sort data=out2; by id; run;
proc print data=out2(firstobs=110 obs=124);
run;
```

Output 26.1.1 shows the number of variables for which the missing observations are imputed and the random seed value for the random imputation method.

<table>
<thead>
<tr>
<th>Imputation Method</th>
<th>Number of Variables</th>
<th>Seed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Random</td>
<td>1</td>
<td>12345</td>
</tr>
<tr>
<td>Median</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Numeric Value</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Mode</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Character Value</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Output 26.1.2 shows the imputation results with interval variables.
Chapter 26: The VARIMPUTE Procedure

Output 26.1.2 Imputed Values for Interval Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Imputation Method</th>
<th>Result Variable</th>
<th>N</th>
<th>Number of Missing</th>
<th>Imputed Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>Mean</td>
<td>IM_VALUE</td>
<td>5848</td>
<td>112</td>
<td>101776</td>
</tr>
<tr>
<td>MORTDUE</td>
<td>Median</td>
<td>IM_MORTDUE</td>
<td>5442</td>
<td>518</td>
<td>65017</td>
</tr>
<tr>
<td>NINQ</td>
<td>Random</td>
<td>IM_NINQ</td>
<td>5450</td>
<td>510</td>
<td></td>
</tr>
<tr>
<td>DEROG</td>
<td>Value</td>
<td>IM_DEROG</td>
<td>5252</td>
<td>708</td>
<td>5</td>
</tr>
<tr>
<td>CLNO</td>
<td>Value</td>
<td>IM_CLNO</td>
<td>5738</td>
<td>222</td>
<td>20</td>
</tr>
</tbody>
</table>

Output 26.1.3 shows the imputation results for nominal variables.

Output 26.1.3 Imputed Values for Nominal Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Imputation Method</th>
<th>Result Variable</th>
<th>N</th>
<th>Number of Missing</th>
<th>Imputed Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>Mean</td>
<td>IM_VALUE</td>
<td>5848</td>
<td>112</td>
<td>101776</td>
</tr>
<tr>
<td>MORTDUE</td>
<td>Median</td>
<td>IM_MORTDUE</td>
<td>5442</td>
<td>518</td>
<td>65017</td>
</tr>
<tr>
<td>NINQ</td>
<td>Random</td>
<td>IM_NINQ</td>
<td>5450</td>
<td>510</td>
<td></td>
</tr>
<tr>
<td>DEROG</td>
<td>Value</td>
<td>IM_DEROG</td>
<td>5252</td>
<td>708</td>
<td>5</td>
</tr>
<tr>
<td>CLNO</td>
<td>Value</td>
<td>IM_CLNO</td>
<td>5738</td>
<td>222</td>
<td>20</td>
</tr>
</tbody>
</table>

Output 26.1.4 shows 15 output observations with the new imputed values for those input variable values that were missing.

Output 26.1.4 Output data table

<table>
<thead>
<tr>
<th>Obs</th>
<th>id</th>
<th>IM_VALUE</th>
<th>IM_MORTDUE</th>
<th>IM_NINQ</th>
<th>IM_DEROG</th>
<th>IM_CLNO</th>
<th>IM_JOB</th>
<th>IM_REASON</th>
</tr>
</thead>
<tbody>
<tr>
<td>110</td>
<td>110</td>
<td>79852.00</td>
<td>68815</td>
<td>0.0000</td>
<td>0</td>
<td>14</td>
<td>Other</td>
<td>Homelmp</td>
</tr>
<tr>
<td>111</td>
<td>111</td>
<td>80027.00</td>
<td>72021</td>
<td>0.0000</td>
<td>2</td>
<td>5</td>
<td>Other</td>
<td>Homelmp</td>
</tr>
<tr>
<td>112</td>
<td>112</td>
<td>63869.00</td>
<td>53969</td>
<td>0.0000</td>
<td>0</td>
<td>30</td>
<td>ProfEx</td>
<td>DebtCon</td>
</tr>
<tr>
<td>113</td>
<td>113</td>
<td>60250.00</td>
<td>65017</td>
<td>10.9178</td>
<td>5</td>
<td>20</td>
<td>Other</td>
<td>UNKNOWN</td>
</tr>
<tr>
<td>114</td>
<td>114</td>
<td>64074.00</td>
<td>53091</td>
<td>0.0000</td>
<td>1</td>
<td>17</td>
<td>Office</td>
<td>DebtCon</td>
</tr>
<tr>
<td>115</td>
<td>115</td>
<td>79087.00</td>
<td>66127</td>
<td>0.0000</td>
<td>1</td>
<td>15</td>
<td>Other</td>
<td>Homelmp</td>
</tr>
<tr>
<td>116</td>
<td>116</td>
<td>77182.00</td>
<td>65017</td>
<td>13.8192</td>
<td>5</td>
<td>20</td>
<td>Other</td>
<td>UNKNOWN</td>
</tr>
<tr>
<td>117</td>
<td>117</td>
<td>66000.00</td>
<td>39000</td>
<td>1.0000</td>
<td>0</td>
<td>19</td>
<td>Other</td>
<td>Homelmp</td>
</tr>
<tr>
<td>118</td>
<td>118</td>
<td>79247.00</td>
<td>71082</td>
<td>0.0000</td>
<td>0</td>
<td>14</td>
<td>Other</td>
<td>Homelmp</td>
</tr>
<tr>
<td>119</td>
<td>119</td>
<td>49044.00</td>
<td>65017</td>
<td>0.0000</td>
<td>0</td>
<td>0</td>
<td>Other</td>
<td>Homelmp</td>
</tr>
<tr>
<td>120</td>
<td>120</td>
<td>101776.05</td>
<td>57000</td>
<td>1.0000</td>
<td>5</td>
<td>13</td>
<td>Other</td>
<td>Homelmp</td>
</tr>
<tr>
<td>121</td>
<td>121</td>
<td>46000.00</td>
<td>34184</td>
<td>2.0000</td>
<td>0</td>
<td>22</td>
<td>Mgr</td>
<td>Homelmp</td>
</tr>
<tr>
<td>122</td>
<td>122</td>
<td>87000.00</td>
<td>66517</td>
<td>0.0000</td>
<td>0</td>
<td>34</td>
<td>ProfEx</td>
<td>DebtCon</td>
</tr>
<tr>
<td>123</td>
<td>123</td>
<td>50000.00</td>
<td>45500</td>
<td>0.0000</td>
<td>0</td>
<td>18</td>
<td>Other</td>
<td>Homelmp</td>
</tr>
<tr>
<td>124</td>
<td>124</td>
<td>146870.00</td>
<td>118000</td>
<td>1.0000</td>
<td>2</td>
<td>20</td>
<td>ProfEx</td>
<td>Homelmp</td>
</tr>
</tbody>
</table>
# Chapter 27
## The VARREDUCE Procedure

### Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview: VARREDUCE Procedure</td>
<td>1164</td>
</tr>
<tr>
<td>PROC VARREDUCE Features</td>
<td>1164</td>
</tr>
<tr>
<td>PROC VARREDUCE Compared to Other SAS Procedures</td>
<td>1165</td>
</tr>
<tr>
<td>Using CAS Sessions and CAS Engine Librefs</td>
<td>1165</td>
</tr>
<tr>
<td>Getting Started: VARREDUCE Procedure</td>
<td>1166</td>
</tr>
<tr>
<td>Unsupervised Variable Selection</td>
<td>1168</td>
</tr>
<tr>
<td>Supervised Variable Selection</td>
<td>1170</td>
</tr>
<tr>
<td>Syntax: VARREDUCE Procedure</td>
<td>1171</td>
</tr>
<tr>
<td>PROC VARREDUCE Statement</td>
<td>1171</td>
</tr>
<tr>
<td>CLASS Statement</td>
<td>1173</td>
</tr>
<tr>
<td>DISPLAY Statement</td>
<td>1174</td>
</tr>
<tr>
<td>DISPLAYOUT Statement</td>
<td>1175</td>
</tr>
<tr>
<td>FREQ Statement</td>
<td>1175</td>
</tr>
<tr>
<td>REDUCE Statement</td>
<td>1176</td>
</tr>
<tr>
<td>Details: VARREDUCE Procedure</td>
<td>1177</td>
</tr>
<tr>
<td>Missing Values</td>
<td>1177</td>
</tr>
<tr>
<td>Unsupervised Variable Selection</td>
<td>1178</td>
</tr>
<tr>
<td>Supervised Variable Selection</td>
<td>1178</td>
</tr>
<tr>
<td>Variable Selection for Regression</td>
<td>1178</td>
</tr>
<tr>
<td>Variable Selection for Classification</td>
<td>1179</td>
</tr>
<tr>
<td>Criteria Used in Model Selection</td>
<td>1179</td>
</tr>
<tr>
<td>Computational Method</td>
<td>1180</td>
</tr>
<tr>
<td>GLM Parameterization of Classification Variables and Effects</td>
<td>1181</td>
</tr>
<tr>
<td>Intercept</td>
<td>1182</td>
</tr>
<tr>
<td>Regression Effects</td>
<td>1182</td>
</tr>
<tr>
<td>Main Effects</td>
<td>1182</td>
</tr>
<tr>
<td>Interaction Effects</td>
<td>1183</td>
</tr>
<tr>
<td>Nested Effects</td>
<td>1184</td>
</tr>
<tr>
<td>Continuous-Nesting-Class Effects</td>
<td>1184</td>
</tr>
<tr>
<td>Continuous-by-Class Effects</td>
<td>1185</td>
</tr>
<tr>
<td>General Effects</td>
<td>1185</td>
</tr>
<tr>
<td>Displayed Output</td>
<td>1186</td>
</tr>
<tr>
<td>Number of Observations</td>
<td>1186</td>
</tr>
<tr>
<td>Selection Summary</td>
<td>1186</td>
</tr>
<tr>
<td>Selected Variables</td>
<td>1186</td>
</tr>
<tr>
<td>ODS Table Names</td>
<td>1186</td>
</tr>
</tbody>
</table>
Overview: VARREDUCE Procedure

The VARREDUCE procedure performs both supervised and unsupervised variable selection in SAS Viya. You can use the VARREDUCE procedure to read data in distributed form and perform variable selection in parallel.

The VARREDUCE procedure performs unsupervised variable selection by identifying a set of variables that jointly explain the maximum amount of data variance. Unlike principal component analysis (PCA), which reduces dimensionality by generating a set of new variables (variable extraction), the VARREDUCE procedure reduces dimensionality by selecting a subset of the original variables (variable selection). Thus, this technique preserves model interpretation.

The VARREDUCE procedure performs supervised variable selection by identifying a set of variables that jointly explain the maximum amount of variance contained in the response variables. The VARREDUCE procedure supports variable selection in both the regression setting and the classification (categorization) setting.

PROC VARREDUCE Features

The VARREDUCE procedure analyzes variance and reduces dimensionality by selecting the variables that contribute the most to the overall variance of the data in unsupervised variable selection (or to the overall variance of the response variables in supervised variable selection). The following list summarizes the basic features of the VARREDUCE procedure:

- Variable selection is based on covariance analysis
- Input data can be read in parallel.
- Computation of the Pearson correlation, covariance, or sums of squares and crossproducts matrix is distributed.
- Computation of the variable selection steps is distributed.
- All phases of analytic execution use of high degree of multithreading.
- Both supervised and unsupervised variable selection are supported.
- Multiple response variables are supported in variable selection for regression.
- The CLASS statement supports categorical inputs.
The REDUCE statement supports main and interaction effects.

The OUTCP option supports outputting a CORR, COV, or SSCP matrix.

PROC VARREDUCE Compared to Other SAS Procedures

This section compares the VARREDUCE procedure with the PCA and GENSELECT procedures in SAS Visual Statistics and with the FACTOR, PRINCOMP, GLMSELECT, and DISCRIM procedures in SAS/STAT software.

When PROC VARREDUCE performs unsupervised variable selection, it analyzes variance and reduces dimensionality by forward selection of the variables that contribute the most to the overall data variance. The output lists the variables in order of their contribution to data variance; the output can be used directly for reporting or for selecting variables for procedures that build models. In contrast, principal component analysis analyzes the variance and then projects the data space onto an orthogonal set of axes by a linear combination of the original variables. These new principal components best explain the data variance and can be used as input to procedures that build models. Principal component analysis can be done through the FACTOR, PRINCOMP, and PCA procedures, which reduce the number of inputs from the original set. The primary difference between principal component analysis (PCA) and PROC VARREDUCE is that PCA generates new variables, whereas PROC VARREDUCE reduces data dimensionality by selecting a subset of the original variables. This feature of PROC VARREDUCE is beneficial in applications where retaining the original variables is important for model exploration and interpretation.

When PROC VARREDUCE performs supervised variable selection, it analyzes the variance and reduces dimensionality by forward selection of the variables that contribute the most to explaining the overall variance of the response variables (targets). The output lists the variables in order of their contribution to explaining the response variance. The output can be used directly for reporting or for selecting variables for procedures that build models. When PROC VARREDUCE is used to perform supervised variable selection, it most resembles the GLMSELECT and GENSELECT procedures. However, PROC VARREDUCE allows multiple response variables, whereas PROC GLMSELECT and PROC GENSELECT do not. When the response variable is a classification variable and its levelization is done in a special format, PROC VARREDUCE analyzes the variance in the same way that linear discriminant analysis (LDA) does. LDA can be done through the DISCRIM procedure in SAS/STAT. Like principal component analysis, LDA generates new variables by linearly combining all original variables, whereas PROC VARREDUCE reduces data dimensionality by selecting a subset of the original variables.

Using CAS Sessions and CAS Engine Librefs

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:

- [The REDUCE statement supports main and interaction effects.]
- [The OUTCP option supports outputting a CORR, COV, or SSCP matrix.]

PROC VARREDUCE Compared to Other SAS Procedures

This section compares the VARREDUCE procedure with the PCA and GENSELECT procedures in SAS Visual Statistics and with the FACTOR, PRINCOMP, GLMSELECT, and DISCRIM procedures in SAS/STAT software.

When PROC VARREDUCE performs unsupervised variable selection, it analyzes variance and reduces dimensionality by forward selection of the variables that contribute the most to the overall data variance. The output lists the variables in order of their contribution to data variance; the output can be used directly for reporting or for selecting variables for procedures that build models. In contrast, principal component analysis analyzes the variance and then projects the data space onto an orthogonal set of axes by a linear combination of the original variables. These new principal components best explain the data variance and can be used as input to procedures that build models. Principal component analysis can be done through the FACTOR, PRINCOMP, and PCA procedures, which reduce the number of inputs from the original set. The primary difference between principal component analysis (PCA) and PROC VARREDUCE is that PCA generates new variables, whereas PROC VARREDUCE reduces data dimensionality by selecting a subset of the original variables. This feature of PROC VARREDUCE is beneficial in applications where retaining the original variables is important for model exploration and interpretation.

When PROC VARREDUCE performs supervised variable selection, it analyzes the variance and reduces dimensionality by forward selection of the variables that contribute the most to explaining the overall variance of the response variables (targets). The output lists the variables in order of their contribution to explaining the response variance. The output can be used directly for reporting or for selecting variables for procedures that build models. When PROC VARREDUCE is used to perform supervised variable selection, it most resembles the GLMSELECT and GENSELECT procedures. However, PROC VARREDUCE allows multiple response variables, whereas PROC GLMSELECT and PROC GENSELECT do not. When the response variable is a classification variable and its levelization is done in a special format, PROC VARREDUCE analyzes the variance in the same way that linear discriminant analysis (LDA) does. LDA can be done through the DISCRIM procedure in SAS/STAT. Like principal component analysis, LDA generates new variables by linearly combining all original variables, whereas PROC VARREDUCE reduces data dimensionality by selecting a subset of the original variables.

Using CAS Sessions and CAS Engine Librefs

SAS Cloud Analytic Services (CAS) is the analytic server and associated cloud services in SAS Viya. This section describes how to create a CAS session and set up a CAS engine libref that you can use to connect to the CAS session. It assumes that you have a CAS server already available; contact your system administrator if you need help starting and terminating a server. This CAS server is identified by specifying the host on which it runs and the port on which it listens for communications. To simplify your interactions with this CAS server, the host information and port information for the server are stored as SAS option values that are retrieved automatically whenever this CAS server needs to be accessed. You can examine the host and port values for the server at your site by using the following statements:
In addition to starting a CAS server, your system administrator might also have created a CAS session and a CAS engine libref for your use. You can define your own sessions and CAS engine librefs that connect to the CAS server as shown in the following statements:

```
cas mysess;
libname mycas cas sessref=mysess;
```

The CAS statement creates the CAS session named `mysess`, and the LIBNAME statement creates the `mycas` CAS engine libref that you use to connect to this session. It is not necessary to explicitly name the CASHOST and CASPORT of the CAS server in the CAS statement, because these values are retrieved from the corresponding SAS option values.

If you have created the `mysess` session, you can terminate it by using the TERMINATE option in the CAS statement as follows:

```
cas mysess terminate;
```

For more information about the CAS and LIBNAME statements, see the section “Introduction to Shared Concepts” on page 10 in Chapter 3, “Shared Concepts.”

### Getting Started: VARREDUCE Procedure

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

This example uses a data table which contains 100 observations that contain one character variable (C), one classification variable (y), and 10 continuous variables (x1–x10). This data table is used for both of the getting-started examples in the following sections.

The following DATA step stores the variables in the `mycas.getStarted` data table and loads the table into your CAS session:

```
data mycas.getStarted;
  input C$ y x1-x10;
datalines;
D 0 10.2 6 1.6 38 15 2.4 20 0.8 8.5 3.9
F 1 12.2 6 2.6 42 61 1.5 10 0.6 8.5 0.7
D 1 7.7 1 2.1 38 61 1 90 0.6 7.5 5.2
J 1 10.9 7 3.5 46 42 0.3 0 0.2 6 3.6
E 0 17.3 6 3.8 26 47 0.9 10 0.4 1.5 4.7
A 0 18.7 4 1.8 2 34 1.7 80 1 9.5 2.2
B 0 7.2 1 0.3 48 61 1.1 10 0.8 3.5 4
D 0 0.1 3 2.4 0 65 1.6 70 0.8 3.5 0.7
H 1 2.4 4 0.7 38 22 0.2 20 0 3 4.2
J 0 15.6 7 1.4 0 98 0.3 0 1 5 5.2
J 0 11.1 3 2.4 42 55 2.2 60 0.6 4.5 0.7
```
<p>| | | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>0</td>
<td>4</td>
<td>6</td>
<td>0.9</td>
<td>4</td>
<td>36</td>
<td>2.1</td>
<td>30</td>
<td>0.8</td>
<td>9</td>
</tr>
<tr>
<td>A</td>
<td>0</td>
<td>6.2</td>
<td>2</td>
<td>1.8</td>
<td>14</td>
<td>79</td>
<td>1.1</td>
<td>70</td>
<td>0.2</td>
<td>0</td>
</tr>
<tr>
<td>H</td>
<td>0</td>
<td>3.7</td>
<td>3</td>
<td>0.8</td>
<td>12</td>
<td>66</td>
<td>1.3</td>
<td>40</td>
<td>0.4</td>
<td>0.5</td>
</tr>
<tr>
<td>A</td>
<td>1</td>
<td>9.2</td>
<td>3</td>
<td>2.3</td>
<td>48</td>
<td>51</td>
<td>2.3</td>
<td>50</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>G</td>
<td>0</td>
<td>14</td>
<td>3</td>
<td>2.1</td>
<td>18</td>
<td>12</td>
<td>2.2</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>E</td>
<td>1</td>
<td>19.5</td>
<td>6</td>
<td>3.7</td>
<td>26</td>
<td>81</td>
<td>0.1</td>
<td>30</td>
<td>0.6</td>
<td>5</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>11</td>
<td>3</td>
<td>2.8</td>
<td>38</td>
<td>9</td>
<td>1.7</td>
<td>50</td>
<td>0.8</td>
<td>6.5</td>
</tr>
<tr>
<td>I</td>
<td>0</td>
<td>15.3</td>
<td>7</td>
<td>2.2</td>
<td>20</td>
<td>98</td>
<td>2.7</td>
<td>100</td>
<td>0.4</td>
<td>7</td>
</tr>
<tr>
<td>H</td>
<td>1</td>
<td>7.4</td>
<td>4</td>
<td>0.5</td>
<td>28</td>
<td>65</td>
<td>1.3</td>
<td>60</td>
<td>0.2</td>
<td>9.5</td>
</tr>
<tr>
<td>F</td>
<td>0</td>
<td>11.4</td>
<td>2</td>
<td>1.4</td>
<td>42</td>
<td>12</td>
<td>2.4</td>
<td>10</td>
<td>0.4</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>19.4</td>
<td>1</td>
<td>0.4</td>
<td>42</td>
<td>4</td>
<td>2.4</td>
<td>10</td>
<td>0</td>
<td>6.5</td>
</tr>
<tr>
<td>G</td>
<td>0</td>
<td>5.9</td>
<td>4</td>
<td>2.6</td>
<td>12</td>
<td>57</td>
<td>0.8</td>
<td>50</td>
<td>0.4</td>
<td>2</td>
</tr>
<tr>
<td>G</td>
<td>1</td>
<td>15.8</td>
<td>6</td>
<td>3.7</td>
<td>34</td>
<td>8</td>
<td>1.3</td>
<td>90</td>
<td>0.6</td>
<td>2.5</td>
</tr>
<tr>
<td>I</td>
<td>0</td>
<td>10</td>
<td>3</td>
<td>1.9</td>
<td>16</td>
<td>80</td>
<td>0.3</td>
<td>90</td>
<td>0.4</td>
<td>9.5</td>
</tr>
<tr>
<td>E</td>
<td>0</td>
<td>15.7</td>
<td>1</td>
<td>2.7</td>
<td>32</td>
<td>25</td>
<td>1.7</td>
<td>20</td>
<td>0.2</td>
<td>8.5</td>
</tr>
<tr>
<td>G</td>
<td>0</td>
<td>11</td>
<td>5</td>
<td>2.9</td>
<td>48</td>
<td>53</td>
<td>0.1</td>
<td>50</td>
<td>1</td>
<td>3.5</td>
</tr>
<tr>
<td>J</td>
<td>1</td>
<td>16.8</td>
<td>0</td>
<td>0.9</td>
<td>14</td>
<td>86</td>
<td>1.4</td>
<td>40</td>
<td>0.8</td>
<td>9</td>
</tr>
<tr>
<td>D</td>
<td>1</td>
<td>11</td>
<td>4</td>
<td>3.2</td>
<td>48</td>
<td>63</td>
<td>2.8</td>
<td>90</td>
<td>0.6</td>
<td>0</td>
</tr>
<tr>
<td>J</td>
<td>1</td>
<td>4.8</td>
<td>7</td>
<td>3.6</td>
<td>24</td>
<td>1</td>
<td>2.2</td>
<td>20</td>
<td>1</td>
<td>8.5</td>
</tr>
<tr>
<td>J</td>
<td>1</td>
<td>10.4</td>
<td>5</td>
<td>2</td>
<td>42</td>
<td>56</td>
<td>1</td>
<td>20</td>
<td>0</td>
<td>3.5</td>
</tr>
<tr>
<td>G</td>
<td>0</td>
<td>12.7</td>
<td>7</td>
<td>3.6</td>
<td>8</td>
<td>56</td>
<td>2.1</td>
<td>70</td>
<td>1</td>
<td>4.5</td>
</tr>
<tr>
<td>G</td>
<td>0</td>
<td>6.8</td>
<td>1</td>
<td>3.2</td>
<td>30</td>
<td>27</td>
<td>0.6</td>
<td>0</td>
<td>0.8</td>
<td>2</td>
</tr>
<tr>
<td>E</td>
<td>0</td>
<td>8.8</td>
<td>0</td>
<td>3.2</td>
<td>2</td>
<td>67</td>
<td>0.7</td>
<td>10</td>
<td>0.4</td>
<td>1</td>
</tr>
<tr>
<td>I</td>
<td>1</td>
<td>0.2</td>
<td>0</td>
<td>2.9</td>
<td>10</td>
<td>41</td>
<td>2.3</td>
<td>60</td>
<td>0.2</td>
<td>9</td>
</tr>
<tr>
<td>J</td>
<td>1</td>
<td>4.6</td>
<td>7</td>
<td>3.9</td>
<td>50</td>
<td>61</td>
<td>2.1</td>
<td>50</td>
<td>0.4</td>
<td>3</td>
</tr>
<tr>
<td>J</td>
<td>1</td>
<td>2.3</td>
<td>2</td>
<td>3.2</td>
<td>36</td>
<td>98</td>
<td>0.1</td>
<td>40</td>
<td>0.6</td>
<td>4.5</td>
</tr>
<tr>
<td>I</td>
<td>0</td>
<td>10.8</td>
<td>3</td>
<td>2.7</td>
<td>28</td>
<td>58</td>
<td>0.8</td>
<td>80</td>
<td>0.8</td>
<td>3</td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>9.3</td>
<td>2</td>
<td>3.3</td>
<td>44</td>
<td>44</td>
<td>0.3</td>
<td>50</td>
<td>0.8</td>
<td>5.5</td>
</tr>
<tr>
<td>F</td>
<td>0</td>
<td>9.2</td>
<td>6</td>
<td>0.6</td>
<td>4</td>
<td>64</td>
<td>0.1</td>
<td>0</td>
<td>0.6</td>
<td>4.5</td>
</tr>
<tr>
<td>D</td>
<td>0</td>
<td>7.4</td>
<td>0</td>
<td>2.9</td>
<td>14</td>
<td>0</td>
<td>0.2</td>
<td>30</td>
<td>0.8</td>
<td>7.5</td>
</tr>
<tr>
<td>G</td>
<td>0</td>
<td>18.3</td>
<td>3</td>
<td>3.1</td>
<td>8</td>
<td>60</td>
<td>0.3</td>
<td>60</td>
<td>0.2</td>
<td>7</td>
</tr>
<tr>
<td>F</td>
<td>0</td>
<td>5.3</td>
<td>4</td>
<td>0.2</td>
<td>48</td>
<td>63</td>
<td>2.3</td>
<td>80</td>
<td>0.2</td>
<td>8</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>2.6</td>
<td>5</td>
<td>2.2</td>
<td>24</td>
<td>4</td>
<td>1.3</td>
<td>20</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>F</td>
<td>0</td>
<td>13.8</td>
<td>4</td>
<td>3.6</td>
<td>4</td>
<td>7</td>
<td>1.1</td>
<td>10</td>
<td>0.4</td>
<td>3.5</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>12.4</td>
<td>6</td>
<td>1.7</td>
<td>30</td>
<td>44</td>
<td>1.1</td>
<td>60</td>
<td>0.2</td>
<td>6</td>
</tr>
<tr>
<td>I</td>
<td>0</td>
<td>1.3</td>
<td>1</td>
<td>1.3</td>
<td>8</td>
<td>53</td>
<td>1.1</td>
<td>70</td>
<td>0.6</td>
<td>7</td>
</tr>
<tr>
<td>F</td>
<td>0</td>
<td>18.2</td>
<td>7</td>
<td>1.7</td>
<td>26</td>
<td>92</td>
<td>2.2</td>
<td>30</td>
<td>1</td>
<td>8.5</td>
</tr>
<tr>
<td>J</td>
<td>0</td>
<td>5.2</td>
<td>2</td>
<td>2.2</td>
<td>18</td>
<td>12</td>
<td>1.4</td>
<td>90</td>
<td>0.8</td>
<td>4</td>
</tr>
<tr>
<td>G</td>
<td>1</td>
<td>9.4</td>
<td>2</td>
<td>0.8</td>
<td>22</td>
<td>86</td>
<td>0.4</td>
<td>30</td>
<td>0.4</td>
<td>1</td>
</tr>
<tr>
<td>J</td>
<td>1</td>
<td>10.4</td>
<td>2</td>
<td>1.7</td>
<td>26</td>
<td>31</td>
<td>2.4</td>
<td>10</td>
<td>0.2</td>
<td>7</td>
</tr>
<tr>
<td>J</td>
<td>0</td>
<td>13</td>
<td>1</td>
<td>1.8</td>
<td>14</td>
<td>11</td>
<td>2.3</td>
<td>50</td>
<td>0.6</td>
<td>5.5</td>
</tr>
<tr>
<td>A</td>
<td>0</td>
<td>17.9</td>
<td>4</td>
<td>3.1</td>
<td>46</td>
<td>58</td>
<td>2.6</td>
<td>90</td>
<td>0.6</td>
<td>1.5</td>
</tr>
<tr>
<td>D</td>
<td>1</td>
<td>19.4</td>
<td>6</td>
<td>3</td>
<td>20</td>
<td>50</td>
<td>2.8</td>
<td>100</td>
<td>0.2</td>
<td>9</td>
</tr>
<tr>
<td>I</td>
<td>0</td>
<td>19.6</td>
<td>3</td>
<td>3.6</td>
<td>22</td>
<td>19</td>
<td>1.2</td>
<td>0</td>
<td>0.6</td>
<td>5</td>
</tr>
<tr>
<td>I</td>
<td>1</td>
<td>6</td>
<td>2</td>
<td>1.5</td>
<td>30</td>
<td>30</td>
<td>2.2</td>
<td>20</td>
<td>0.4</td>
<td>8.5</td>
</tr>
<tr>
<td>G</td>
<td>0</td>
<td>13.8</td>
<td>1</td>
<td>2.7</td>
<td>0</td>
<td>52</td>
<td>2.4</td>
<td>20</td>
<td>0.8</td>
<td>6</td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>14.3</td>
<td>4</td>
<td>2.9</td>
<td>30</td>
<td>11</td>
<td>0.6</td>
<td>90</td>
<td>0.6</td>
<td>0.5</td>
</tr>
<tr>
<td>E</td>
<td>0</td>
<td>15.6</td>
<td>0</td>
<td>0.4</td>
<td>38</td>
<td>79</td>
<td>0.4</td>
<td>80</td>
<td>0.4</td>
<td>1</td>
</tr>
<tr>
<td>D</td>
<td>0</td>
<td>14</td>
<td>2</td>
<td>1</td>
<td>22</td>
<td>61</td>
<td>3</td>
<td>90</td>
<td>0.6</td>
<td>2</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>9.4</td>
<td>5</td>
<td>0.4</td>
<td>12</td>
<td>53</td>
<td>1.7</td>
<td>40</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>H</td>
<td>0</td>
<td>13.2</td>
<td>1</td>
<td>1.6</td>
<td>40</td>
<td>15</td>
<td>0.7</td>
<td>40</td>
<td>0.2</td>
<td>9</td>
</tr>
<tr>
<td>A</td>
<td>0</td>
<td>13.5</td>
<td>5</td>
<td>2.4</td>
<td>18</td>
<td>89</td>
<td>1.6</td>
<td>20</td>
<td>0.4</td>
<td>9.5</td>
</tr>
<tr>
<td>E</td>
<td>0</td>
<td>2.6</td>
<td>4</td>
<td>2.3</td>
<td>38</td>
<td>6</td>
<td>0.8</td>
<td>20</td>
<td>0.4</td>
<td>5</td>
</tr>
<tr>
<td>E</td>
<td>0</td>
<td>12.4</td>
<td>3</td>
<td>1.3</td>
<td>26</td>
<td>8</td>
<td>2.8</td>
<td>10</td>
<td>0.8</td>
<td>6</td>
</tr>
</tbody>
</table>
These statements assume that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

### Unsupervised Variable Selection

The following statements use PROC VARREDUCE for unsupervised variable selection:

```plaintext
proc varreduce data=mycas.getStarted technique=VarianceAnalysis;
    class C;
    reduce unsupervised C x1-x10 / maxeffects=5 varexp=0.99;
run;
```

The TECHNIQUE= option in the PROC VARREDUCE statement specifies variance analysis as the technique to be used for variable selection. In the REDUCE statement, the MAXEFFECTS= option specifies 5 as the maximum number of variables to select, and the VAREXP= option specifies 99% as the maximum percentage of the total variance to explain. The procedure stops when either of these two conditions is satisfied.
The output from this analysis is presented in Figure 27.1 through Figure 27.3.

Figure 27.1 displays the “Number of Observations” tables. This table shows that all 100 observations in the data table are used in the analysis.

**Figure 27.1** Number of Observations

The VARREDUCE Procedure

<table>
<thead>
<tr>
<th>Number of Observations Read</th>
<th>100</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Observations Used</td>
<td>100</td>
</tr>
</tbody>
</table>

Figure 27.2 and Figure 27.3 show the “Selection Summary” and “Selected Effects” tables. The “Selection Summary” table shows which variable (or level for CLASS variables) is selected in each step, in addition to showing the total variance that is explained by the variables selected so far. The “Selected Effects” table presents all the selected variables and their corresponding variable types.

**Figure 27.2** Selection Summary

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Parameter</th>
<th>Proportion of Variance Explained</th>
<th>SSE</th>
<th>MSE</th>
<th>AIC</th>
<th>AICC</th>
<th>BIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>x3</td>
<td>0.068102</td>
<td>18.637953</td>
<td>0.188262</td>
<td>7.105200</td>
<td>27.216339</td>
<td>2.971252</td>
</tr>
<tr>
<td>2</td>
<td>x7</td>
<td>0.132579</td>
<td>17.348424</td>
<td>0.177025</td>
<td>6.993502</td>
<td>26.094008</td>
<td>2.945605</td>
</tr>
<tr>
<td>3</td>
<td>x10</td>
<td>0.193293</td>
<td>16.134131</td>
<td>0.166331</td>
<td>6.860937</td>
<td>24.945494</td>
<td>2.919092</td>
</tr>
<tr>
<td>4</td>
<td>C I</td>
<td>0.252883</td>
<td>14.942348</td>
<td>0.155649</td>
<td>6.704199</td>
<td>23.767490</td>
<td>2.888406</td>
</tr>
<tr>
<td>5</td>
<td>C F</td>
<td>0.311964</td>
<td>13.760728</td>
<td>0.144850</td>
<td>6.521819</td>
<td>22.558528</td>
<td>2.852077</td>
</tr>
<tr>
<td>6</td>
<td>C J</td>
<td>0.368951</td>
<td>12.620982</td>
<td>0.134266</td>
<td>6.315361</td>
<td>21.320171</td>
<td>2.811671</td>
</tr>
<tr>
<td>7</td>
<td>C B</td>
<td>0.425720</td>
<td>11.485609</td>
<td>0.123501</td>
<td>6.081095</td>
<td>20.048690</td>
<td>2.763457</td>
</tr>
<tr>
<td>8</td>
<td>C G</td>
<td>0.481257</td>
<td>10.374861</td>
<td>0.112770</td>
<td>5.819386</td>
<td>18.744449</td>
<td>2.707799</td>
</tr>
<tr>
<td>9</td>
<td>C D</td>
<td>0.535098</td>
<td>9.290037</td>
<td>0.102176</td>
<td>5.529803</td>
<td>17.407018</td>
<td>2.644269</td>
</tr>
<tr>
<td>10</td>
<td>C H</td>
<td>0.588910</td>
<td>8.221792</td>
<td>0.091353</td>
<td>5.206788</td>
<td>16.030839</td>
<td>2.567305</td>
</tr>
<tr>
<td>11</td>
<td>C A</td>
<td>0.640665</td>
<td>7.186698</td>
<td>0.080749</td>
<td>4.852232</td>
<td>14.617801</td>
<td>2.478801</td>
</tr>
<tr>
<td>12</td>
<td>x5</td>
<td>0.691609</td>
<td>6.167815</td>
<td>0.070089</td>
<td>4.459345</td>
<td>13.161117</td>
<td>2.371965</td>
</tr>
</tbody>
</table>

**Figure 27.3** Selected Effects

<table>
<thead>
<tr>
<th>Selected Number</th>
<th>Selected Variable</th>
<th>Variable Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>x3</td>
<td>INTERVAL</td>
</tr>
<tr>
<td>2</td>
<td>x7</td>
<td>INTERVAL</td>
</tr>
<tr>
<td>3</td>
<td>x10</td>
<td>INTERVAL</td>
</tr>
<tr>
<td>4</td>
<td>C</td>
<td>CLASS</td>
</tr>
<tr>
<td>5</td>
<td>x5</td>
<td>INTERVAL</td>
</tr>
</tbody>
</table>
Supervised Variable Selection

The following statements use PROC VARREDUCE for supervised variable selection. The CLASS statement specifies that $y$ is a categorical response variable. The TECHNIQUE= option specifies discriminant analysis as the technique to be used for variable selection. The MAXEFFECTS= option in the REDUCE statement specifies 5 as the maximum number of variables to select.

```plaintext
proc varreduce data=mycas.getStarted technique=DiscriminantAnalysis;
    class C y;
    reduce supervised y = C x1-x10 / maxeffects=5;
run;
```

The output from this analysis is presented in Figure 27.4 through Figure 27.6.

Figure 27.4 shows that all 100 observations in the data table are used in the analysis.

Figure 27.5 and Figure 27.6 show the “Selection Summary” and “Selected Effects” tables. The “Selection Summary” table shows which variable (or level for CLASS variables) is selected in each step, in addition to showing the total variance that is explained by the variables selected so far. The “Selected Effects” table presents all the selected variables and their corresponding variable types.

### Figure 27.4 Number of Observations

The VARREDUCE Procedure

<table>
<thead>
<tr>
<th>Number of Observations Read</th>
<th>100</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Observations Used</td>
<td>100</td>
</tr>
</tbody>
</table>

### Figure 27.5 Selection Summary

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Parameter</th>
<th>Proportion of Variance Explained</th>
<th>SSE</th>
<th>MSE</th>
<th>AIC</th>
<th>AICC</th>
<th>BIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>C J</td>
<td>0.081099 0.918901 0.009282</td>
<td>0.015424</td>
<td>2.019590</td>
<td>-0.038525</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>x8</td>
<td>0.132303 0.867697 0.008854 0.001913</td>
<td>2.005456</td>
<td>-0.049809</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>x2</td>
<td>0.168747 0.831253 0.008570 0.004821</td>
<td>2.006669</td>
<td>-0.046665</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>C C</td>
<td>0.199152 0.800848 0.008342 0.002084</td>
<td>2.014475</td>
<td>-0.037877</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>x4</td>
<td>0.218366 0.781634 0.008228 0.013631</td>
<td>2.036240</td>
<td>-0.016110</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>x9</td>
<td>0.236868 0.763132 0.008118 0.029676</td>
<td>2.059346</td>
<td>0.005986</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Figure 27.6 Selected Effects

<table>
<thead>
<tr>
<th>Selected Effects</th>
<th>Selected Variable</th>
<th>Variable Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Variable</td>
<td>Type</td>
</tr>
<tr>
<td>1</td>
<td>C</td>
<td>CLASS</td>
</tr>
<tr>
<td>2</td>
<td>x8</td>
<td>INTERVAL</td>
</tr>
<tr>
<td>3</td>
<td>x2</td>
<td>INTERVAL</td>
</tr>
<tr>
<td>4</td>
<td>x4</td>
<td>INTERVAL</td>
</tr>
<tr>
<td>5</td>
<td>x9</td>
<td>INTERVAL</td>
</tr>
</tbody>
</table>
Syntax: VARREDUCE Procedure

The following statements are available in the VARREDUCE procedure:

```
PROC VARREDUCE < options > ;
  CLASS variable < (options) > . . . < variable < (options) > > </ global-options > ;
  DISPLAY < table-list > </ options > ;
  DISPLAYOUT table-spec-list </ options > ;
  REDUCE UNSUPERVISED effects </ reduce-options > ;
  REDUCE SUPERVISED response . . . < response > = effects </ reduce-options > ;
  FREQ variable ;
```

The `PROC VARREDUCE` statement and one `REDUCE` statement are required. The `CLASS` statement can appear multiple times. If a `CLASS` statement is specified, it must precede the `REDUCE` statement.

PROC VARREDUCE Statement

```
PROC VARREDUCE < options > ;
```

The `PROC VARREDUCE` statement invokes the procedure. Table 27.1 summarizes the important options in the `PROC VARREDUCE` statement by function. The options are then described fully in alphabetical order.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATA=</td>
<td>Specifies the input data table</td>
</tr>
<tr>
<td>NOPRINT</td>
<td>Suppresses ODS output</td>
</tr>
<tr>
<td>MATRIX=</td>
<td>Specifies the matrix to use to select variables</td>
</tr>
<tr>
<td>TECHNIQUE=</td>
<td>Selects the variable selection technique</td>
</tr>
<tr>
<td>OUTCP=</td>
<td>Outputs the CORR, COV, or SSCP matrix, which is specified in the MATRIX= option</td>
</tr>
</tbody>
</table>

You can specify the following options:

- `DATA=CAS-libref.data-table` names the input data table for PROC VARREDUCE to use. The default is the most recently created data table. `CAS-libref.data-table` is a two-level name, where

  - `CAS-libref` refers to a collection of information that is defined in the LIBNAME statement and includes the caslib, which includes a path to the data, and a session identifier, which
Chapter 27: The VARREDUCE Procedure

defaults to the active session but which can be explicitly defined in the LIBNAME statement. For more information about CAS-libref, see the section “Using CAS Sessions and CAS Engine Librefs” on page 1165.

data-table specifies the name of the input data table.

**MATRIX=CORR | COV | SSCP**
specifies which matrix to use to select variables.

You can specify the following values:

- **CORR** selects variables based on the Pearson correlation matrix. Assuming that $X$ and $Y$ are two variables, the correlation between $X$ and $Y$ is computed by

\[
\text{Corr}(X, Y) = \frac{E((X - E(X))(Y - E(Y)))}{\sqrt{E((X - E(X))^2)E((Y - E(Y))^2)}}
\]

- **COV** selects variables based on the covariance matrix. Assuming that $X$ and $Y$ are two variables, the covariance between $X$ and $Y$ is computed by

\[
\text{Cov}(X, Y) = E((X - E(X))(Y - E(Y)))
\]

- **SSCP** selects variables based on the sums of squares and crossproducts matrix. Assuming that $X$ and $Y$ are two variables and that $x$ and $y$ are their corresponding variable vectors, the SSCP between $X$ and $Y$ is computed by

\[
\text{SSCP}(X, Y) = x^T y
\]

By default, MATRIX=CORR.

**NOPRINT** suppresses the generation of ODS output.

**OUTCP=CAS-libref.data-table < / LIST < EPS = number >>** creates a data table that contains a symmetric matrix that depicts the relationships among variables and also creates a set of statistics about the input data table and variables. *CAS-libref.data-table* is a two-level name, where *CAS-libref* refers to the caslib and session identifier, and *data-table* specifies the name of the output data table. For more information about this two-level name, see the DATA= option and the section “Using CAS Sessions and CAS Engine Librefs” on page 1165. Depending on the value of the MATRIX= option, the symmetric matrix can be a correlation (CORR) matrix, a covariance (COV) matrix, or a sums of squares and crossproducts (SSCP) matrix.

When you specify the LIST option, the symmetric matrix is output in the list-of-list (LIL) format. In this format, the matrix is represented as a set of tuples $(i, j, x)$, where $x$ is an entry in the matrix and $i$ and $j$ denote its row and column indices, respectively. LIL format can be used when the output contains too many columns to fit in a data table. For example, in most database systems, the maximum number of columns in a table is usually limited to several thousand. If an output matrix contains more columns than the limit, you must use the LIST option to avoid errors that would arise from writing too many columns to the table. When LIL format is used, all 0 entries in the matrix are ignored in the output.

When you specify EPS= *number* in the LIST suboption, matrix entries whose absolute value is smaller than *number* are ignored in the output. This feature helps omit unreliable estimations and generate a compact representation of the matrix. When you do not specify the EPS= option, only the 0 entries in the matrix are ignored in the output.
TECHNIQUE=keyword
TECH=keyword

specifies the variable selection technique.

You can specify the following keywords:

VARIANCEANALYSIS | VAR performs variance analysis for variable selection.

DISCRIMINANTANALYSIS | DSC performs discriminant analysis for variable selection.

By default, TECHNIQUE=VAR.

You can use variance analysis for both supervised and unsupervised variable selection. You can use discriminant analysis only for supervised variable selection that uses one classification variable as the response. For more information, see the section “Variable Selection for Classification” on page 1179.

### CLASS Statement

CLASS variable < (options)> . . . < variable < (options)> > </global-options> ;

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. The CLASS statement must precede the REDUCE statement.

The VARREDUCE procedure supports the GLM method of CLASS variable parameterization; for more information, see the section “GLM Parameterization of Classification Variables and Effects” on page 1181.

You can specify options either as individual variable options, by enclosing the options in parentheses after the variable name, or as global-options, by placing them after a slash (/). Global-options are applied to all variables that are specified in the CLASS statement. If you specify more than one CLASS statement, the global-options that are specified in any one CLASS statement apply to all CLASS statements. However, individual CLASS variable options override the global-options.

Table 27.2 summarizes the values you can use for either an option or a global-option. The options are described in detail in the list that follows Table 27.2.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESCENDING</td>
<td>Reverses the sort order</td>
</tr>
<tr>
<td>MISSING</td>
<td>Treats missing values as valid levels</td>
</tr>
</tbody>
</table>

DESCENDING
DESC

reverses the sort order of the classification variable.

MISSING
treats missing values (“, .A”, . . . , .Z” for numeric variables and blanks for character variables) as valid values for the CLASS variable.
If you do not specify the MISSING option, observations that have missing values for CLASS variables are removed from the analysis.

**DISPLAY Statement**

```
DISPLAY < table-list > < / options > ;
```

The DISPLAY statement enables you to specify a list of display tables to display or exclude. This statement is similar to the ODS SELECT, ODS EXCLUDE, and ODS TRACE statements. However, the DISPLAY statement can improve performance when a large number of tables could be generated (such as in BY-group processing). The procedure processes the DISPLAY statement on a CAS server and thus sends only a subset of ODS tables to the SAS client. Because ODS statements are processed on a SAS client, first all the generated display tables are sent to the client, and then the client creates a subset.

If you use both DISPLAY and ODS statements together, the DISPLAY statement takes precedence over the ODS statements. Note that the ODS EXCLUDE statement processes tables that are sent to the client after they have been filtered by the DISPLAY statement. In some cases, it might appear that the ODS EXCLUDE statement is taking precedence because it can further filter the tables. For more information about ODS, see *SAS Output Delivery System: Procedures Guide*.

You can specify the `table-list` as a list of table names, paths, partial pathnames, and regular expressions.

The table names that you can specify are listed in the section “ODS Table Names” on page 1186. A path is a table name that is prefixed with dot-separated grouping information. For example, a SelectionSummary table that a procedure produces during a selection routine might have the path `Bygroup1.Summary.SelectionSummary`. A partial pathname does not include all groups; for example, `Selection-` and `Summary.SelectionSummary` are partial pathnames for `Bygroup1.Summary.SelectionSummary`.

When you specify a table name or partial pathname, all display tables whose paths end in the specified name are selected for display or exclusion. For example, both `SelectionSummary` and `Summary.SelectionSummary` select `Bygroup1.Summary.SelectionSummary`.

A regular expression is enclosed in forward slashes (/). For example, specifying “/tions/” selects all pathnames that contain the substring “tions”; in particular, the `Bygroup1.Summary.SelectionSummary` table is selected. Specifying “!/tions/” selects all pathnames that do not contain the substring “tions”; in particular, the `Bygroup1.Summary.SelectionSummary` table is not selected.

You can specify the following `options` after a slash (/):

**CASESENSITIVE**

performs a case-sensitive comparison of table names in the `table-list` to display table names when tables are subsetted for display. To preserve case, you must enclose table names in the `table-list` in quotation marks.

**EXCLUDE**

displays all display tables except those that you specify in the `table-list`.

**EXCLUDEALL**

suppresses display of all tables. This option takes precedence over the other options.
TRACE
displays the display table names, labels, and paths.

DISPLAYOUT Statement

DISPLAYOUT table-spec-list < / options > ;

The DISPLAYOUT statement enables you to create CAS output tables from your displayed output. This statement is similar to the ODS OUTPUT statement. For more information about ODS, see SAS Output Delivery System: Procedures Guide.

The table-spec-list specifies a list of CAS output tables to create. Each entry in the list has either a key=value format or a key format:

key=value specifies key as the ODS table name, path, or partial pathname, and specifies value as the CAS output table name.
key specifies key as the ODS table name and also as the CAS output table name.

The ODS table names that you can specify are listed in the section “ODS Table Names” on page 1186. You cannot specify the ODS table named OutputCasTables in the table-spec-list.

Table names and partial pathnames are discussed under the DISPLAY statement. The DISPLAYOUT statement does not support regular expressions.

You can specify the following options after a slash (/):

INCLUDEALL
creates output CAS tables for all display tables. The name of the created output CAS table is the same as the corresponding display table name. If you specify this option, the table-spec-list specification is ignored.

NOREPLACE
does not replace any existing CAS output table of the same name.

REPEATED
replicates all CAS output tables on all nodes.

FREQ Statement

FREQ variable ;

The variable in the FREQ statement identifies a numeric variable in the data set that contains the frequency of occurrence of each observation. PROC VARREDUCE treats each observation as if it appears \( f \) times, where \( f \) is the value of the FREQ variable for the observation. If \( f \) is not an integer, it is truncated to an integer. If \( f \) is less than 1 or missing, the observation is not used in the analysis. When the FREQ statement is not specified, each observation is assigned a frequency of 1.
**REDUCE Statement**

**REDUCE UNSUPERVISED** effects < / reduce-options > ;

**REDUCE SUPERVISED** response-variable < response-variable > . . . = effects < / reduce-options > ;

PROC VARREDUCE can be used for either of the following types of variable selection:

- For unsupervised variable selection, you specify the `effects` to be considered in the variable selection process. An `effect` can be an original variable in the input data table or a variable that is constructed from the original variables.

- For supervised variable selection, you specify both the `effects` and one or more `response-variables`. A `response-variable` can be an original variable in the input data table or a variable that is constructed from the original variables. PROC VARREDUCE supports variable selection both in a classification (categorization) context, in which you specify one `response-variable`, and in a regression context, in which you can specify more than one `response-variable`.

Table 27.3 summarizes the `reduce-options`, which control the number of variables to be selected.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIC</td>
<td>Uses Akaike’s information criterion to perform model selection</td>
</tr>
<tr>
<td>AICC</td>
<td>Uses the corrected Akaike’s information criterion to perform model selection</td>
</tr>
<tr>
<td>BIC</td>
<td>Uses Schwarz Bayesian information criterion to perform model selection</td>
</tr>
<tr>
<td>MAXITER</td>
<td>MAXSTEPS=</td>
</tr>
<tr>
<td>MAXEFFECTS=</td>
<td>Specifies the number of effects to select; the number must be greater than or equal to 1.</td>
</tr>
<tr>
<td>VARIANCEEXPLAINED</td>
<td>VAREXP=</td>
</tr>
<tr>
<td>MINVARIANCEINCREMENT</td>
<td>VARINC=</td>
</tr>
</tbody>
</table>

The `reduce-options` determine the number of variables to be selected. You can specify the following `reduce-options` as stopping criteria for the VARREDUCE procedure. When you specify more than one option, PROC VARREDUCE stops whenever one of the specified options is satisfied, or when the explained variance equals the total variance. In the latter case, the procedure prints the following message in the log: “Early stop: the proportion of the explained variance to the total variance equals 1.”

You can specify the following `reduce-options` only for supervised variable selection:
AIC
stops PROC VARREDUCE if the Akaike’s information criterion (AIC) value fails to decrease in three contiguous steps.

AICC
stops PROC VARREDUCE if the corrected Akaike’s information (AICC) value fails to decrease in three contiguous steps.

BIC
stops PROC VARREDUCE if the Schwarz Bayesian information criterion (BIC) value fails to decrease in three contiguous steps.

You can specify the following reduce-options for both supervised and unsupervised variable selection:

MAXITER=n
MAXSTEPS=n
stops PROC VARREDUCE after it runs n steps.

MAXEFFECTS=n
stops PROC VARREDUCE after n effects have been selected. Because individual levels of one classification variable can be selected in different steps of the variable selection process, selecting n effects might require more steps than are specified by the value of MAXITER= option.

VARIANCEEXPLAINED=fraction
VAREXP=fraction
stops PROC VARREDUCE when the fraction of the total variance can be explained by the selected variables.

MINVARIANCEINCREMENT=fraction
VARINC=fraction
stops PROC VARREDUCE when the minimum increment of the explained variance is less than fraction of the total variance.

Details: VARREDUCE Procedure

The performance of a learning model usually decreases in terms of accuracy and efficiency when the dimensionality of the input data is high. The problem is known as the “curse of dimensionality.” Variable selection techniques can reduce the dimensionality of data by removing irrelevant and redundant variables (Liu and Motoda 1998).

The VARREDUCE procedure performs both supervised and unsupervised variable selection. It selects variables by identifying a set of variables that can jointly explain the maximum amount of data variance.

Missing Values

Any observation that has missing values for the responses, frequency, or effects is excluded from the analysis; however, missing values are valid for responses and effects if the MISSING option is specified in the CLASS statement. Observations that have a frequency less than 1 are also excluded.
Unsupervised Variable Selection

When no response variable is specified, PROC VARREDUCE conducts unsupervised variable selection. Assume that \( k \) variables need to be selected. Let \( X \in \mathbb{R}^{n \times m} \) be a data table that contains \( n \) samples and \( m \) variables; let \( X = (X_1, X_2) \), where \( X_1 \in \mathbb{R}^{n \times k} \) contains the \( k \) selected variables and \( X_2 \in \mathbb{R}^{n \times (m-k)} \) contains the remaining \( m - k \) variables. PROC VARREDUCE selects the variables by minimizing the following equation:

\[
\min \text{Trace} \left( X_2^T \left( I - X_1 \left( X_1^T X_1 \right)^{-1} X_1^T \right) X_2 \right)
\]

\( \left( I - X_1 \left( X_1^T X_1 \right)^{-1} X_1^T \right) ^{1/2} \) projects \( X_2 \) to the null space of \( X_1 \). Therefore, the preceding equation measures the data variance that resides in the null space of \( X_1 \), which is the data variance that cannot be explained by the variables in \( X_1 \). Minimizing this equation leads to the selection of the variables that jointly explain the maximum amount of the variance in the original data.

Let \( C_{11} = X_1^T X_1 \), \( C_{12} = X_1^T X_2 \), and \( C_{21} = X_2^T X_1 \). The following equations hold:

\[
C = X^T X = \begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix}
\]

\[
X_2^T \left( I - X_1 \left( X_1^T X_1 \right)^{-1} X_1^T \right) X_2 = C_{22} - C_{21} C_{11}^{-1} C_{12}
\]

When all the variables are centralized to have a zero mean, \( C \) is the covariance matrix. This corresponds to specifying \texttt{MATRIX=COV} in the PROC VARREDUCE statement, which requests that the covariance matrix be used for variable selection. Similarly, if variables need to be both centralized and normalized to have unit length, you should specify \texttt{MATRIX=CORR} in the PROC VARREDUCE statement, which requests that the correlation matrix be used for variable selection. If neither centralization nor normalization should be applied, specify \texttt{MATRIX=SSCP} in the PROC VARREDUCE statement.

Principal component analysis (PCA) (Jolliffe 2002) also reduces dimensionality by preserving data variance. The key difference between PCA and PROC VARREDUCE is that PCA generates a small set of new variables (variable extraction) by linearly combining the original variables, whereas PROC VARREDUCE selects a small set of the original variables (variable selection). The variables returned by PROC VARREDUCE are the original variables. This feature is very important in applications where retaining the original variables is important for model exploration or interpretation (for example, genetic analysis and text mining).

Supervised Variable Selection

When response variables are specified in a REDUCE statement, PROC VARREDUCE conducts supervised variable selection, either in a regression context or in a classification (categorization) context.

Variable Selection for Regression

In a regression context, all response variables should be numerical. When the response is a classification variable, it needs to be levelized to multiple dummy variables, where each dummy variable corresponds to a
level of the classification variable. You can achieve this levelization by adding this variable to the variable list in the CLASS statement.

Let \( Y \in \mathbb{R}^{n \times t} \) be the response data that contain \( t \) response variables. Assume that \( k \) variables need to be selected. Let \( X \in \mathbb{R}^{n \times m} \) be a data table that contains \( n \) samples and \( m \) variables; let \( X = (X_1, X_2) \), where \( X_1 \in \mathbb{R}^{n \times k} \) contains the \( k \) selected variables and \( X_2 \in \mathbb{R}^{n \times (m-k)} \) contains the remaining \( m-k \) variables. PROC VARREDUCE selects the variables by minimizing the following equation:

\[
\min \text{Trace} \left( Y^T \left( I - X_1 \left( X_1^T X_1 \right)^{-1} X_1^T \right) Y \right)
\]

\( \left( I - X_1 \left( X_1^T X_1 \right)^{-1} X_1^T \right)^{1/2} Y \) projects \( Y \) onto the null space of \( X_1 \). Therefore, the equation measures the response variance that resides in the null space of \( X_1 \), which is the variance of the response variables that cannot be explained by the variables in \( X_1 \). Minimizing the equation leads to the selection of the variables that jointly explain the maximum amount of the variance of the response variables.

### Variable Selection for Classification

In a classification context, one classification variable is specified as the response, and each of its levels corresponds to a category of the classification problem. Let the classification variable be \( y \) with \( c \) levels \( \{1, \ldots, c\} \). Then \( y \) can be levelized in a special way to generate response data \( Y \in \mathbb{R}^{n \times c} \) as:

\[
Y_{i,j} = \begin{cases} 
\frac{1}{\sqrt{n}} \left( \sqrt{n_j} - \sqrt{n_i} \right), & y_i = j \\
-\frac{1}{\sqrt{n}} \sqrt{n_j}, & y_i \neq j
\end{cases}
\]

By using this \( Y \) in the variance analysis, PROC VARREDUCE selects variables by using the discriminant criterion that is specified in linear discriminant analysis (LDA) (Fisher 1936; Cooley and Lohnes 1971). LDA also reduces dimensionality. The key difference between LDA and PROC VARREDUCE is that LDA generates a small set of new variables (variable extraction) by linearly combining the original variables, whereas PROC VARREDUCE selects a small set of the original variables (variable selection).

### Criteria Used in Model Selection

The VARREDUCE procedure supports the following three fit statistics that you can specify as stopping criteria in the REDUCE statement. These statistics are produced only for the supervised variable selection:

- **AIC** Akaike’s information criterion (Akaike 1969; Judge et al. 1985)
- **AICC** Corrected Akaike’s information criterion (Hurvich and Tsai 1989)
- **BIC** Schwarz Bayesian information criterion (Schwarz 1978; Judge et al. 1985)

The VARREDUCE procedure supports multiple response variables; therefore, it computes the AIC, AICC, and BIC that are defined for multivariate regression. Besides the three criteria, it also computes the error sum of squares (SSE) and residual mean square error (MSE).

Table 27.4 provides formulas and definitions for these fit statistics.
### Table 27.4  Formulas and Definitions for Model Fit Summary Statistics

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Definition or Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>Number of observations</td>
</tr>
<tr>
<td>$p$</td>
<td>Number of parameters</td>
</tr>
<tr>
<td>$t$</td>
<td>Number of response variables in supervised selection; number of unselected variables in unsupervised selection</td>
</tr>
<tr>
<td>SSE</td>
<td>Error sum of squares</td>
</tr>
<tr>
<td>MSE</td>
<td>$\frac{\text{SSE}}{n-p}$</td>
</tr>
<tr>
<td>AIC</td>
<td>$\ln(\text{SSE}) + \frac{2pt + t(t+1)}{n}$</td>
</tr>
<tr>
<td>AICCC</td>
<td>$\ln(\text{SSE}) + \frac{(n+p)t}{n-p-t-1} - pt\ln(n)$</td>
</tr>
<tr>
<td>BIC</td>
<td>$\ln(\text{SSE}) + \frac{p\ln(n)}{n}$</td>
</tr>
</tbody>
</table>

### Computational Method

Given $m$ variables, finding the $k$ variables that minimize the proposed equations is a combinatorial problem, which is NP-hard (nondeterministic polynomial-time hard). To select $k$ variables, PROC VARREDUCE applies $k$ steps of a greedy search to generate a suboptimal solution for the problem.

Assume that $q$ features have been selected, that $X_1$ contains the $q$ selected variables, and that $X_2$ contains the remaining variables. PROC VARREDUCE selects the $q + 1$ variable, $F$, by minimizing the equation

$$\arg\min_{F} \text{Trace} \left( \hat{X}_2^T \left( I - \hat{X}_1 \left( \hat{X}_1^T \hat{X}_1 \right)^{-1} \hat{X}_1^T \right) \hat{X}_2 \right)$$

where $\hat{X}_1$ is the data table that contains the feature $F$ and the $q$ selected variables, and $\hat{X}_2$ is the data table that contains the remaining variables. Minimizing the preceding problem is equivalent to maximizing the following problem:

$$\left\| \frac{X_2^T}{2} \left( I - X_1 \left(X_1^T X_1 \right)^{-1} X_1^T \right) f \right\|_2^2$$

$$\left\| \left( I - X_1 \left(X_1^T X_1 \right)^{-1} X_1^T \right)^{\frac{1}{2}} f \right\|_2^2$$

In the preceding equation, $\left\| \frac{X_2^T}{2} \left( I - X_1 \left(X_1^T X_1 \right)^{-1} X_1^T \right) f \right\|_2^2$ is the summation of the squares of the covariance between the variable $f$ and all the unselected variables in the null space of $X_1$. And $\left\| \left( I - X_1 \left(X_1^T X_1 \right)^{-1} X_1^T \right)^{\frac{1}{2}} f \right\|_2^2$ is the square of the variance of $f$ in the null space of $X_1$, which is used as a normalization factor.
This problem can be solved efficiently. Assuming that \( m \gg k \), the time complexity for solving it is

\[
O \left( m^2 (n + k^2) \right)
\]

where \( m \) is the number of variables, \( n \) is the number of samples, and \( k \) is the number of selected variables. In the equation, \( m^2 n \) corresponds to the time for computing the covariance (or correlation or SSCP) matrix. And \( m^2 k^2 \) corresponds to the time for selecting \( k \) variables out of \( m \).

Similar analysis also applies to supervised variable selection with PROC VARREDUCE. The following problem is maximized for supervised variable selection:

\[
\| Y^T \left( I - X_1 (X_1^T X_1)^{-1} X_1^T \right) f \|^2_2
\]

\[
\| (I - X_1 (X_1^T X_1)^{-1} X_1^T) \frac{1}{2} f \|^2_2
\]

Here, \( Y \) is the response data table. Let \( c \) be the number of columns in \( Y \). The time complexity for selecting \( k \) variables by solving the preceding problem is

\[
O \left( k^2 (c + k) m + m^2 n \right)
\]

Note that for most data of very high dimensionality, \( c + k \ll m \).

PROC VARREDUCE is fully threaded and distributed. When there are \( p \) machines used for computing, the time complexity for unsupervised variable selection is

\[
\text{CPU} \left( \frac{m^2 (n + k^2)}{p} + m^2 \log p \right) + \text{NET} \left( m^2 \log p \right)
\]

and the time complexity for supervised variable selection is

\[
\text{CPU} \left( \frac{k^2 (c + k) m + m^2 n}{p} + m^2 \log p \right) + \text{NET} \left( m^2 \log p \right)
\]

where CPU corresponds to the time used for computing and NET corresponds to the time used for communication among computers.

### GLM Parameterization of Classification Variables and Effects

Table 27.5 shows the types of effects that are available in the VARREDUCE procedure; they are discussed in more detail in the following subsections. Let \( A, B, \) and \( C \) represent classification variables, and let \( X \) and \( Z \) represent continuous variables.

<table>
<thead>
<tr>
<th>Effect</th>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>Default</td>
<td>Intercept (unless the NOINT option is specified)</td>
</tr>
<tr>
<td>Regression</td>
<td>X Z</td>
<td>Continuous variables</td>
</tr>
<tr>
<td>Polynomial</td>
<td>X*Z</td>
<td>Interaction of continuous variables</td>
</tr>
</tbody>
</table>
Table 27.6 shows some examples of MODEL statements that use various types of effects.

<table>
<thead>
<tr>
<th>Specification</th>
<th>Type of Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>model Y=X;</td>
<td>Simple regression</td>
</tr>
<tr>
<td>model Y=X Z;</td>
<td>Multiple regression</td>
</tr>
<tr>
<td>model Y=X X*X;</td>
<td>Polynomial regression</td>
</tr>
<tr>
<td>model Y=A;</td>
<td>One-way analysis of variance (ANOVA)</td>
</tr>
<tr>
<td>model Y=A B C;</td>
<td>Main-effects ANOVA</td>
</tr>
<tr>
<td>model Y=A B A*B;</td>
<td>Factorial ANOVA with interaction</td>
</tr>
<tr>
<td>model y=A B(A) C(B A);</td>
<td>Nested ANOVA</td>
</tr>
<tr>
<td>model Y=A X;</td>
<td>Analysis of covariance (ANCOVA)</td>
</tr>
<tr>
<td>model Y=A X(A);</td>
<td>Separate-slopes regression</td>
</tr>
<tr>
<td>model Y=A X X*A;</td>
<td>Homogeneity-of-slopes regression</td>
</tr>
</tbody>
</table>

Intercept

By default, linear models that are created by the VARREDUCE procedure automatically include a column of 1s in X. This column corresponds to an intercept parameter. In many procedures, you can use the NOINT option in the MODEL statement to suppress this intercept. For example, the NOINT option is useful when the MODEL statement contains a classification effect and you want the parameter estimates to be in terms of the mean response for each level of that effect.

Regression Effects

Numeric variables or polynomial terms that involve them can be included in the model as regression effects (covariates). The actual values of such terms are included as columns of the relevant model matrices. You can use the bar operator along with a regression effect to generate polynomial effects. For example, X|X|X expands to X*X*X*X*X*X, which is a cubic model.

Main Effects

If a classification variable has m levels, the GLM parameterization generates m columns for its main effect in the model matrix. Each column is an indicator variable for a particular level. The order of the columns is the sort order of the values of their levels and can be controlled by the ORDER= option in the CLASS statement.
Table 27.7 is an example where \( \beta_0 \) denotes the intercept and A and B are classification variables that have two and three levels, respectively.

### Table 27.7  Example of Main Effects

<table>
<thead>
<tr>
<th>Data</th>
<th>I</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>( \beta_0 )</td>
<td>A1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

There are usually more columns for these effects than there are degrees of freedom to estimate them. In other words, the GLM parameterization of main effects is *singular*.

### Interaction Effects

Often a model includes interaction (crossed) effects to account for how the effect of a variable changes along with the values of other variables. With an interaction, the terms are first reordered to correspond to the order of the variables in the CLASS statement. Thus, \( B*A \) becomes \( A*B \) if A precedes B in the CLASS statement. Then, the GLM parameterization generates columns for all combinations of levels that occur in the data. The order of the columns is such that the rightmost variables in the interaction change faster than the leftmost variables (Table 27.8).

### Table 27.8  Example of Interaction Effects

<table>
<thead>
<tr>
<th>Data</th>
<th>I</th>
<th>A</th>
<th>B</th>
<th>A*B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

In the preceding matrix, main-effects columns are not linearly independent of crossed-effects columns. In fact, the column space for the crossed effects contains the space of the main effect.

When your model contains many interaction effects, you might be able to code them more parsimoniously by using the bar operator ( \( | \) ). The bar operator generates all possible interaction effects. For example, \( A | B | C \) expands to \( A B A*B C A*C B*C A*B*C \). To eliminate higher-order interaction effects, use the at sign ( \( @ \) ) in conjunction with the bar operator. For example, \( A | B | C | D@2 \) expands to \( A B A*B C A*C B*C D A*D B*D C*D \).
Nested Effects

Nested effects are generated in the same manner as crossed effects. Hence, the design columns that are generated by the following two statements are the same (but the ordering of the columns is different):

```plaintext
model Y=A B(A);
model Y=A A*B;
```

The nesting operator in the VARREDUCE procedure is more of a notational convenience than an operation that is distinct from crossing. Nested effects are typically characterized by the property that the nested variables do not appear as main effects. The order of the variables within nesting parentheses is made to correspond to the order of these variables in the CLASS statement. The order of the columns is such that variables outside the parentheses index faster than those inside the parentheses, and the rightmost nested variables index faster than the leftmost variables (Table 27.9).

<table>
<thead>
<tr>
<th>Data</th>
<th>I</th>
<th>A</th>
<th>B(A)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Continuous-Nesting-Class Effects

When a continuous variable nests or crosses with a classification variable, the design columns are constructed by multiplying the continuous values into the design columns for the classification effect (Table 27.10).

<table>
<thead>
<tr>
<th>Data</th>
<th>I</th>
<th>A</th>
<th>X(A)</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>A</td>
<td>β₀</td>
<td>A1</td>
</tr>
<tr>
<td>21</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>24</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>22</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>28</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>19</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>23</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

This model estimates a separate intercept and a separate slope for X within each level of A.
Continuous-by-Class Effects

Continuous-by-class effects generate the same design columns as continuous-nesting-class effects. Table 27.11 shows the construction of the $X*A$ effect. The two columns for this effect are the same as the columns for the $X(A)$ effect in Table 27.10.

<table>
<thead>
<tr>
<th>Data</th>
<th>I</th>
<th>X</th>
<th>A</th>
<th>$X*A$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$</td>
<td>$A$</td>
<td>$\beta_0$</td>
<td>$X$</td>
<td>$A_1$</td>
</tr>
<tr>
<td>21</td>
<td>1</td>
<td>1</td>
<td>21</td>
<td>1</td>
</tr>
<tr>
<td>24</td>
<td>1</td>
<td>1</td>
<td>24</td>
<td>1</td>
</tr>
<tr>
<td>22</td>
<td>1</td>
<td>1</td>
<td>22</td>
<td>1</td>
</tr>
<tr>
<td>28</td>
<td>2</td>
<td>1</td>
<td>28</td>
<td>0</td>
</tr>
<tr>
<td>19</td>
<td>2</td>
<td>1</td>
<td>19</td>
<td>0</td>
</tr>
<tr>
<td>23</td>
<td>2</td>
<td>1</td>
<td>23</td>
<td>0</td>
</tr>
</tbody>
</table>

You can use continuous-by-class effects together with pure continuous effects to test for homogeneity of slopes.

General Effects

An example that combines all the effects is $X_1*X_2*A*B*C(D,E)$. The continuous list comes first, followed by the crossed list, followed by the nested list in parentheses.

Effects might be renamed by the procedure to correspond to ordering rules. For example, $B*A(E,D)$ might be renamed $A*B(D,E)$ to satisfy the following:

- Classification variables that occur outside parentheses (crossed effects) are sorted in the order in which they appear in the CLASS statement.
- Variables within parentheses (nested effects) are sorted in the order in which they appear in the CLASS statement.

The sequencing of the parameters that are generated by an effect is determined by the variables whose levels are indexed faster:

- Variables in the crossed list index faster than variables in the nested list.
- Within a crossed or nested list, variables to the right index faster than variables to the left.

For example, suppose that a model includes four effects—$A$, $B$, $C$, and $D$—each having two levels, 1 and 2, and that the CLASS statement is as follows:

```plaintext
class A B C D;
```
Then the order of the parameters for the effect $B^*A(C \ D)$, which is renamed $A^*B(C \ D)$, is as follows:

$$A_1 B_1 C_1 D_1 \rightarrow A_1 B_2 C_1 D_1 \rightarrow A_2 B_1 C_1 D_1 \rightarrow A_2 B_2 C_1 D_1 \rightarrow$$
$$A_1 B_1 C_1 D_2 \rightarrow A_1 B_2 C_1 D_2 \rightarrow A_2 B_1 C_1 D_2 \rightarrow A_2 B_2 C_1 D_2 \rightarrow$$
$$A_1 B_1 C_2 D_1 \rightarrow A_1 B_2 C_2 D_1 \rightarrow A_2 B_1 C_2 D_1 \rightarrow A_2 B_2 C_2 D_1 \rightarrow$$
$$A_1 B_1 C_2 D_2 \rightarrow A_1 B_2 C_2 D_2 \rightarrow A_2 B_1 C_2 D_2 \rightarrow A_2 B_2 C_2 D_2 \rightarrow$$

Note that first the crossed effects $B$ and $A$ are sorted in the order in which they appear in the `CLASS` statement so that $A$ precedes $B$ in the parameter list. Then, for each combination of the nested effects in turn, combinations of $A$ and $B$ appear. The $B$ effect changes fastest because it is rightmost in the cross list. Then $A$ changes next fastest, and $D$ changes next fastest. The $C$ effect changes most slowly because it is leftmost in the nested list.

### Displayed Output

The following sections describe the output that PROC VARREDUCE produces by default. The output is organized into various tables, which are discussed in the order of appearance.

#### Number of Observations

The “Number of Observations” table displays the number of observations read from the input data table and the number of observations used in the analysis.

#### Selection Summary

The “Selection Summary” table displays for each iteration the name of the selected effect, the name of the selected level, and the total variance explained after the iteration.

#### Selected Variables

The “Selected Variables” table summarizes which variables were selected in the selection process. It also provides information about the variable type of each selected variable.

### ODS Table Names

Each table that the VARREDUCE procedure creates has a name associated with it. You must use this name to refer to the table when you use ODS statements. The name of each table and a short description of the contents are listed in Table 27.12.

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>NObs</td>
<td>Number of observations read and used; number of events and trials, if applicable</td>
<td>PROC VARREDUCE</td>
<td>Default</td>
</tr>
</tbody>
</table>
Examples: VARREDUCE Procedure

**NOTE:** Input data must be in a CAS table that is accessible in your CAS session. You must refer to this table by using a two-level name. The first level must be a CAS engine libref, and the second level must be the table name. For more information, see the sections “Using CAS Sessions and CAS Engine Librefs” on page 10 and “Loading a SAS Data Set onto a CAS Server” on page 11 in Chapter 3, “Shared Concepts.”

**Example 27.1: Analyzing Framingham Heart Study Data**

This example analyzes the data from the Framingham Heart Study for feature selection, which are available in a data set named Heart in the Sashelp library.

You can load the Sashelp.Heart data set into your CAS session by naming your CAS engine libref in the first statement of the following DATA step:

```sas
   data mycas.heart;
      set sashelp.heart;
   run;
```

These statements assume that your CAS engine libref is named `mycas`, but you can substitute any appropriately defined CAS engine libref.

The following statements invoke the VARREDUCE procedure:

```sas
   proc varreduce data=mycas.heart matrix=COV tech=DSC;
      ods output SelectionSummary=Summary;
      class Status Sex Chol_Status BP_Status Weight_Status Smoking_Status;
      reduce supervised Status = Sex AgeAtStart Height Weight Diastolic Systolic MRW Smoking Cholesterol Chol_Status BP_Status Weight_Status Smoking_Status/ maxiter=15 BIC;
      display 'SelectionSummary' 'SelectedEffects';
   run;

   proc sgplot data=Summary;
      series x=Iteration y=BIC;
   run;
```

The TECH=DSC option in the PROC VARREDUCE statement requests a discriminant analysis of the Heart data table for feature selection. The ODS OUTPUT statement stores the “Selection Summary” table as a
Chapter 27: The VARREDUCE Procedure

local file named Summary. The MATRIX=COV option in the PROC VARREDUCE statement requests that selections be done based on the covariance matrix. The BIC option specifies the stop criterion, and the MAXITER= option specifies 15 as the maximum number of iterations. The selection process terminates when the BIC statistic increases in the last three consecutive steps.

Output 27.1.1 and Output 27.1.2 display the selection summary from each iteration and the selected effects.

Output 27.1.1 Selection Summary

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Parameter</th>
<th>Proportion of Variance Explained</th>
<th>SSE</th>
<th>MSE</th>
<th>AIC</th>
<th>AICC</th>
<th>BIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AgeAtStart</td>
<td>0.191310</td>
<td>0.808690</td>
<td>0.00016052</td>
<td>-0.210355</td>
<td>1.789647</td>
<td>-0.210648</td>
</tr>
<tr>
<td>2</td>
<td>Smoking</td>
<td>0.220923</td>
<td>0.779077</td>
<td>0.00015467</td>
<td>-0.246867</td>
<td>1.753136</td>
<td>-0.246262</td>
</tr>
<tr>
<td>3</td>
<td>Systolic</td>
<td>0.245405</td>
<td>0.754595</td>
<td>0.00014984</td>
<td>-0.278002</td>
<td>1.722002</td>
<td>-0.276499</td>
</tr>
<tr>
<td>4</td>
<td>Sex Female</td>
<td>0.255479</td>
<td>0.744521</td>
<td>0.00014787</td>
<td>-0.290648</td>
<td>1.709358</td>
<td>-0.288247</td>
</tr>
<tr>
<td>5</td>
<td>Weight_Status Underweight</td>
<td>0.256772</td>
<td>0.743228</td>
<td>0.00014764</td>
<td>-0.291593</td>
<td>1.708415</td>
<td>-0.288294</td>
</tr>
<tr>
<td>6</td>
<td>BP_Status Normal</td>
<td>0.257880</td>
<td>0.742120</td>
<td>0.00014745</td>
<td>-0.292291</td>
<td>1.707720</td>
<td>-0.288093</td>
</tr>
<tr>
<td>7</td>
<td>Smoking_Status Non-smoker</td>
<td>0.258962</td>
<td>0.741038</td>
<td>0.00014727</td>
<td>-0.292957</td>
<td>1.707057</td>
<td>-0.287861</td>
</tr>
<tr>
<td>8</td>
<td>Chol_Status Borderline</td>
<td>0.259547</td>
<td>0.740453</td>
<td>0.00014718</td>
<td>-0.292952</td>
<td>1.707065</td>
<td>-0.286958</td>
</tr>
</tbody>
</table>

Output 27.1.2 Selected Effects

<table>
<thead>
<tr>
<th>Number</th>
<th>Selected Effects</th>
<th>Variable Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AgeAtStart</td>
<td>INTERVAL</td>
</tr>
<tr>
<td>2</td>
<td>Smoking</td>
<td>INTERVAL</td>
</tr>
<tr>
<td>3</td>
<td>Systolic</td>
<td>INTERVAL</td>
</tr>
<tr>
<td>4</td>
<td>Sex</td>
<td>CLASS</td>
</tr>
<tr>
<td>5</td>
<td>Weight_Status</td>
<td>CLASS</td>
</tr>
<tr>
<td>6</td>
<td>BP_Status</td>
<td>CLASS</td>
</tr>
<tr>
<td>7</td>
<td>Smoking_Status</td>
<td>CLASS</td>
</tr>
<tr>
<td>8</td>
<td>Chol_Status</td>
<td>CLASS</td>
</tr>
</tbody>
</table>

Output 27.1.3 shows the BIC curve change throughout iterations.
Example 27.2: Output a Correlation Matrix to a SAS Data File

This example shows how to output a correlation matrix to a SAS data file. The OUTCP= option creates an output data table named mycas.corr.

The following DATA step generates a data table that has 2,000 observations and contains both interval variables (x1–x2) and CLASS variables (a, c1, and c2):

```sas
data mycas.data1;
  array x{2};
  array c{2};
  do i=1 to 2000;
    a=int(ranuni(1)*2);
    do j=1 to 2;
      x{j}=ranuni(1);
      c{j}=int(ranuni(1)*2);
    end;
    output;
  end;
run;
```

These statements assume that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

The following statements invoke the VARREDUCE procedure:

```sas
title "Output the Correlation Matrix";

proc varreduce data=mycas.data1 matrix=corr outcp=mycas.corr;
  class a;
  reduce unsupervised a x1-x2 /maxsteps=4;
run;

proc print data=mycas.corr;
run;
```
Output 27.2.1 shows the content of the data file that PROC VARREDUCE generates.

**Output 27.2.1** Output the Correlation Matrix

<table>
<thead>
<tr>
<th>Obs</th>
<th><em>ID</em></th>
<th><em>TYPE</em></th>
<th><em>VAR</em></th>
<th><em>vID</em></th>
<th>v1</th>
<th>v2</th>
<th>v3</th>
<th>v4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>MEAN/FREQ</td>
<td></td>
<td></td>
<td>1015.00</td>
<td>985.00</td>
<td>0.50</td>
<td>0.50</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>N</td>
<td></td>
<td></td>
<td>2000.00</td>
<td>2000.00</td>
<td>2000.00</td>
<td>2000.00</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>CORR</td>
<td>a0</td>
<td>v1</td>
<td>1.00</td>
<td>-1.00</td>
<td>-0.00</td>
<td>-0.02</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>CORR</td>
<td>a1</td>
<td>v2</td>
<td>-1.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.02</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>CORR</td>
<td>x1</td>
<td>v3</td>
<td>1.00</td>
<td>0.00</td>
<td>1.00</td>
<td>-0.02</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>CORR</td>
<td>x2</td>
<td>v4</td>
<td>-0.02</td>
<td>0.02</td>
<td>-0.01</td>
<td>1.00</td>
</tr>
</tbody>
</table>

The _VAR_ column displays the names of all variables and the levels of the CLASS variables. Assuming that you have \( n \) effects (the total number of interval variables and the levels of CLASS variables), the _vID_ column contains \( n \) markers, v1 to vn, where vi denotes the ith effect. The column _TYPE_ defines the role of each row. When the _TYPE_ column displays MEAN/FREQ, the corresponding row contains either the mean for an interval variable or the frequency for a level of a CLASS variable. When the _TYPE_ column displays N, the corresponding row contains the number of samples. And when the _TYPE_ column displays CORR, COV, or SSCP, the corresponding row contains a row of the CORR, COV, or SSCP matrix. In this example, the CORR matrix is \( 4 \times 4 \), and it resides in the table in rows 3–6 and columns 7–10.

**Example 27.3: Output the Correlation Matrix in LIL Format**

This example shows how to output a correlation matrix in list-of-list (LIL) format. The OUTCP= option creates an output data table named mycas.corr_lil.

The following DATA step generates a data table that has 2,000 observations and contains both interval variables (x1–x2) and a CLASS variable (a):

```plaintext
data mycas.data2;
    array x{2};
    do i=1 to 2000;
        a=int(ranuni(1)*2);
        do j=1 to 2;
            x{j}=ranuni(1);
        end;
        output;
    end;
run;
```

These statements assume that your CAS engine libref is named mycas, but you can substitute any appropriately defined CAS engine libref.

The following statements invoke the VARREDUCE procedure:
Example 27.3: Output the Correlation Matrix in LIL Format

```sas
title "Output the Correlation Matrix in LIL Format";
proc varreduce data=mycas.data2 matrix=corr outcp=mycas.corr_lil/list eps=0.01;
   class a;
   reduce unsupervised a x1-x2 /maxsteps=4;
run;
proc print data=mycas.corr_lil;
run;
```

Output 27.3.1 shows the correlation matrix in LIL format.

**Output 27.3.1** Output the Correlation Matrix in LIL Format

<table>
<thead>
<tr>
<th>Obs</th>
<th><em>TYPE</em></th>
<th><em>ID</em></th>
<th><em>NAME1</em></th>
<th><em>NAME2</em></th>
<th><em>VAL</em></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>S</td>
<td>1</td>
<td>samples</td>
<td></td>
<td>2000.00</td>
</tr>
<tr>
<td>2</td>
<td>S</td>
<td>2</td>
<td>nVar</td>
<td></td>
<td>3.00</td>
</tr>
<tr>
<td>3</td>
<td>S</td>
<td>3</td>
<td>nEff</td>
<td></td>
<td>4.00</td>
</tr>
<tr>
<td>4</td>
<td>F</td>
<td>1</td>
<td>a</td>
<td>0</td>
<td>979.00</td>
</tr>
<tr>
<td>5</td>
<td>F</td>
<td>2</td>
<td>a</td>
<td>1</td>
<td>1021.00</td>
</tr>
<tr>
<td>6</td>
<td>M</td>
<td>3</td>
<td>x1</td>
<td></td>
<td>0.49</td>
</tr>
<tr>
<td>7</td>
<td>M</td>
<td>4</td>
<td>x2</td>
<td></td>
<td>0.50</td>
</tr>
<tr>
<td>8</td>
<td>R</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
</tr>
<tr>
<td>9</td>
<td>R</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>-1.00</td>
</tr>
<tr>
<td>10</td>
<td>R</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>1.00</td>
</tr>
<tr>
<td>11</td>
<td>R</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>0.03</td>
</tr>
<tr>
<td>12</td>
<td>R</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>-0.03</td>
</tr>
<tr>
<td>13</td>
<td>R</td>
<td>6</td>
<td>3</td>
<td>3</td>
<td>1.00</td>
</tr>
<tr>
<td>14</td>
<td>R</td>
<td>10</td>
<td>4</td>
<td>4</td>
<td>1.00</td>
</tr>
</tbody>
</table>

The column _TYPE_ defines the type of each row:

- When the _TYPE_ column displays S, the corresponding row contains the statistics of the data table. More specifically, when the _TYPE_ column displays S and the _NAME1_ column displays samples, the _VAL_ column in the corresponding row contains the number of samples in the data table. Similarly, when the _TYPE_ column displays S and the _NAME1_ column displays nVar, the _VAL_ column contains the number of variables. And when the _TYPE_ column displays S and the _NAME1_ column displays nEff, the _VAL_ column in the corresponding row contains the number of effects.

- When the _TYPE_ column displays F, the row contains the frequency of a level of a CLASS variable. In this case, the _NAME1_ column contains the name and level of the CLASS variable.

- When the _TYPE_ column displays M, the row contains the mean of an interval variable. In this case, the _NAME1_ column contains the name of the variable and the _NAME2_ column is empty.

- When the _TYPE_ column displays R, the row contains an entry in the correlation matrix. In this case, the _NAME1_ column contains the row ID, the _NAME2_ column contains the column ID, and the _VAL_ column contains the value.
• When the _TYPE_ column displays V or P, the corresponding row contains an entry of a COV matrix or an SSCP matrix, respectively.

Only entries in the lower triangle of the correlation matrix are written to the file, because the correlation matrix is symmetric. Also, any entry of the matrix whose value is smaller than 0.01 is ignored in the output (EPS = 0.01); this saves storage space.

References


Subject Index

2-D geometric anisotropic structure
LMIXED procedure, 516

ABC parameters
KCLUS procedure, 474

ABC statistics
KCLUS procedure, 474

AC1 agreement coefficient
FREQTAB procedure, 260

adapted LASSO selection
Shared Concepts, 69

adjusted odds ratio
FREQTAB procedure, 265

adjusted relative risks
FREQTAB procedure, 266

agreement, measures of
FREQTAB procedure, 254

Agresti-Caffo confidence limits
risk difference (FREQTAB), 228

Agresti-Coull confidence limits
proportions (FREQTAB), 219

Akaike’s information criterion
LMIXED procedure, 536

Akaike’s information criterion (finite sample corrected version)
LMIXED procedure, 536

alpha level
LMIXED procedure, 505, 515

ANCOVA effects
Shared Concepts, 55
VARREDUCE procedure, 1182

ANOVA (row mean scores) statistic
Mantel-Haenszel (FREQTAB), 263

ANOVA effects
Shared Concepts, 54
VARREDUCE procedure, 1182

ANOVA table
REGSELECT procedure, 913

ANTE(1) structure
LMIXED procedure, 516

antedependence structure (ARTE)
LMIXED procedure, 516

AR(1) structure
LMIXED procedure, 516

ASSESS procedure, 1073

B-spline basis, 33
computational method, 81, 82
convergence criterion, 44–46, 48
displayed output, 1084
fit statistics, 1082, 1084
function-based convergence criteria, 44, 45
gradient-based convergence criteria, 44, 46
input data tables, 1077
input variables, 1080
lift information, 1080, 1084
lift regression information, 1084
multithreading, 81
natural cubic spline basis, 35
ODS table names, 35, 1084
optimization techniques, 48, 82
parameter-based convergence criteria, 45, 48
ROC information, 1081, 1084
splines and spline bases, 31
target variables, 1080
test data, 80
trunncated power function (TPF) basis, 32
validation, 80

association statistics
LOGSELECT procedure, 597
association, measures of
FREQTAB procedure, 208
at sign (@) operator
Shared Concepts, 53, 56, 1183

autoregressive moving average structure
LMIXED procedure, 516

autoregressive structure (AR(1))
LMIXED procedure, 516

B-spline
spline basis (Shared Concepts), 33

B-spline basis
ASSESS procedure, 33
BINNING procedure, 33
CARDINALITY procedure, 33
CORRELATION procedure, 33
GAMMOD procedure, 33
GENSELECT procedure, 33
ICA procedure, 33
KCLUS procedure, 33
LMIXED procedure, 33
LOGSELETC procedure, 33
MBC procedure, 33
NLMOD procedure, 33
PARTITION procedure, 33
PCA procedure, 33
PHSELECT procedure, 33
splines and spline bases, 31
test data, 80
truncated power function (TPF) basis, 32
validation, 80
case-control studies
  odds ratio (FREQTAB), 241
categorical data analysis
  FREQTAB procedure, 114
cell count data
  FREQTAB procedure, 198
centering and scaling information
  ICA procedure, 443
  PCA procedure, 728
  PLSMOD procedure, 818
chi-square goodness-of-fit test
  FREQTAB procedure, 203
chi-square tests
  FREQTAB procedure, 202
Cholesky
covariance structure (LMIXED), 520
root (LMIXED), 520
Cholesky covariance matrix (CHOL)
  L MIXED procedure, 516
Cicchetti-Allison weights
  kappa coefficient (FREQTAB), 259
class level
  GAMMOD procedure, 315, 341
  GENSELECT procedure, 374, 413
  L MIXED procedure, 498, 534
  LOGSELECT procedure, 562, 595
  MODELMATRIX procedure, 665
  PHSELECT procedure, 753, 774
  PLSMOD procedure, 803, 818
  QTRSELECT procedure, 840, 862
  REGSELECT procedure, 894, 912
CLASS statement
  Shared Concepts, 49
  syntax (Shared Concepts), 12, 49
classification level
  PARTITION procedure, 1145
classification table
  HPLOGISTIC procedure, 586
  LOGSELECT procedure, 560, 597
classification trees, see decision trees
classification variables
  Shared Concepts, 49
Clopper-Pearson confidence limits
  proportions (FREQTAB), 218
cluster initialization method
  MBC procedure, 627
cluster summary
  KCLUS procedure, 471
cluster summary for mixed variables
  KCLUS procedure, 473
cluster summary for nominal variables
  KCLUS procedure, 473
Cochran’s Q test
  FREQTAB procedure, 254, 261
Cochran-Armitage test for trend
  FREQTAB procedure, 251
Cochran-Mantel-Haenszel statistics
  FREQTAB procedure, 261
CODE statement
  syntax (Shared Concepts), 16
cohort studies
  relative risks (FREQTAB), 244
colon (:) operator
  Shared Concepts, 53
column names of the design matrix
  MODELMA TRIX procedure, 664
common odds ratio
  exact confidence limits (FREQTAB), 269
  exact test (FREQTAB), 269
  logit (FREQTAB), 265
  Mantel-Haenszel (FREQTAB), 265
common relative risks
  logit (FREQTAB), 266
  Mantel-Haenszel (FREQTAB), 266
common risk difference
  FREQTAB procedure, 237
common subject
  L MIXED procedure, 532
complete separation
  GENSELECT procedure, 405
  LOGSELECT procedure, 583
components
  PLSMOD procedure, 788
compound symmetry structure (CS)
  L MIXED procedure, 516
computation
  BINNING procedure, 1105
computational method
  ASSESS procedure, 81, 82
  BINNING procedure, 81, 82
  CARDINALITY procedure, 81, 82
  CORRELATION procedure, 81, 82
  GAMMOD procedure, 81, 82
  GENSELECT procedure, 81, 82
  ICA procedure, 81, 82
  KCLUS procedure, 81, 82
  L MIXED procedure, 81, 82, 534
  LOGSELECT procedure, 81, 82
  MBC procedure, 81, 82
  MODELMA TRIX procedure, 664
  NLMOD procedure, 81, 82
  PARTITION procedure, 81, 82
  PCA procedure, 81, 82
  PHSELECT procedure, 81, 82
Subject Index

PLSMOD procedure, 81, 82
QTRSELECT procedure, 81, 82, 861
REGSELECT procedure, 81, 82, 910
TREESPLIT procedure, 81, 82
VARIMPUTE procedure, 81, 82
VARREDUCE procedure, 81, 82, 1180

concordant observations
FREQTAB procedure, 208

confidence limits
exact (FREQTAB), 126
measures of association (FREQTAB), 208
model parameters (GENSELECT), 382, 384
model parameters (LOGSELECT), 570
model parameters (MODELMATRIX), 663
model parameters (PHSELECT), 761
model parameters (QTRSELECT), 846
model parameters (REGSELECT), 900
proportions (FREQTAB), 218

confidence limits, SPC procedure
confidence level, 950
type, 950

constraints
boundary (LMIXED), 512, 514
contingency coefficient
FREQTAB procedure, 207
contingency tables
FREQTAB procedure, 114, 147
continuity-adjusted chi-square test
FREQTAB procedure, 205
continuous-by-class effects
Shared Concepts, 57
VARREDUCE procedure, 1185
continuous-nesting-class effects
Shared Concepts, 57
VARREDUCE procedure, 1184

convergence criteria
LMIXED procedure, 506, 507, 535

convergence criterion
ASSESS procedure, 44–46, 48
BINNING procedure, 44–46, 48
CARDINALITY procedure, 44–46, 48
CORRELATION procedure, 44–46, 48
GAMMOD procedure, 44–46, 48, 315
GENSELECT procedure, 44–46, 48, 373
ICA procedure, 44–46, 48
KCLUS procedure, 44–46, 48
LMIXED procedure, 44–46, 48
LOGSELECT procedure, 44–46, 48, 559
MBC procedure, 44–46, 48
NLMOD procedure, 44–46, 48
PARTITION procedure, 44–46, 48
PCA procedure, 44–46, 48
PHSELECT procedure, 44–46, 48, 752
PLSMOD procedure, 44–46, 48
QTRSELECT procedure, 44–46, 48
REGSELECT procedure, 44–46, 48
Shared Concepts, 44–46, 48
TREESPLIT procedure, 44–46, 48
VARIMPUTE procedure, 44–46, 48
VARREDUCE procedure, 44–46, 48

convergence status
GAMMOD procedure, 342
GENSELECT procedure, 414
LMIXED procedure, 536
LOGSELECT procedure, 595
MBC procedure, 638
PHSELECT procedure, 774

corrected sums of squares and crossproducts, 95
correlation
estimates (LMIXED), 524
PCA procedure, 728
principal components, 727, 728
correlation coefficients
limited combinations of, 99
printing, for each variable, 95
suppressing probabilities, 95
correlation matrix
GENSELECT procedure, 373, 416
LOGSELECT procedure, 560, 597
PHSELECT procedure, 752, 776

CORRELATION procedure, 89
B-spline basis, 33
computational method, 81, 82
details, 100
extamples, 104
function-based convergence criteria, 44, 45
gradient-based convergence criteria, 44, 46
input data tables, 95
multithreading, 81
natural cubic spline basis, 35
ODS table names, 35, 104
optimization techniques, 48, 82
OUTP= data table, 102
parameter-based convergence criteria, 45, 48
splines and spline bases, 31
syntax, 94
task tables, 94
test data, 80
truncated power function (TPF) basis, 32
validation, 80
correlation statistic
Mantel-Haenszel (FREQTAB), 263
cost complexity
TREESPLIT procedure, 1053
cost-complexity pruning
TREESPLIT procedure, 1030, 1040
covariance
PCA procedure, 728
principal components, 727, 728
covariance matrix
GENSELECT procedure, 373, 415
LOGSELECT procedure, 560, 597
PHSELECT procedure, 753, 776
covariance parameter estimates
LMIXED procedure, 536
covariance structure
antedependence (LMIXED), 519
autoregressive (LMIXED), 519
autoregressive moving average (LMIXED), 520
banded (LMIXED), 521
banded unstructured (LMIXED), 522
Cholesky type (LMIXED), 520
compound symmetry (LMIXED), 521
equi-correlation (LMIXED), 522
examples (LMIXED), 517
factor-analytic (LMIXED), 521
heterogeneous autoregressive (LMIXED), 520
heterogeneous compound symmetry (LMIXED), 521
heterogeneous Toeplitz (LMIXED), 522
heterogeneous uniform correlation (LMIXED), 522
Huynh-Feldt (LMIXED), 521
LMIXED procedure, 490, 516
positive (semi)definite, 520
simple (LMIXED), 521
Toeplitz (LMIXED), 521
uniform correlation (LMIXED), 522
unstructured (LMIXED), 522
unstructured, correlation (LMIXED), 522
variance components (LMIXED), 523
covariances, 95
Cramér’s V statistic
FREQTAB procedure, 207
Cronbach’s coefficient alpha, 101
calculating and printing, 95
dimensional reduction
dimensions
GENSELECT procedure, 443
LMIXED procedure, 499
LOGSELECT procedure, 596
MODELMATRIX procedure, 665
PHSELECT procedure, 776
PLSMOD procedure, 818
QTRSELECT procedure, 862
REGSELECT procedure, 912
direct product structure
LMIXED procedure, 516
discordant observations
FREQTAB procedure, 208
dispersion parameter
GAMMOD procedure, 338
DISPLAY statement
syntax (Shared Concepts), 19
displayed output
ASSESS procedure, 1084
BINNING procedure, 1108
CARDINALITY procedure, 1126
GAMMOD procedure, 341
GENSELECT procedure, 413
ICA procedure, 443
KCLUS procedure, 471
LMIXED procedure, 534
LOGSELECT procedure, 594
MBC procedure, 638
MODELMATRIX procedure, 664
NLMOD procedure, 695
PARTITION procedure, 1145
Subject Index

PCA procedure, 728
PHSELECT procedure, 773
PLSMOD procedure, 818
QTRSELECT procedure, 862
REGSELECT procedure, 911
TREESPLIT procedure, 1052
VARIMPUTE procedure, 1159
VARREDUCE procedure, 1186

DISPLAYOUT statement
- syntax (Shared Concepts), 20

distributed computing
- LMIXED procedure, 534
distribution function
- GAMMOD procedure, 326
- GENSELECT procedure, 382
distribution functions
- NLMOD procedure, 692
dot plots
- FREQTAB procedure, 184
double dash (\texttt{- -}) operator
- Shared Concepts, 54
dummy parameterization
- Shared Concepts, 54
- VARREDUCE procedure, 1181
effect
- name length (LMIXED), 498
effect details
- MODELMATRIX procedure, 665
effect parameterization
- Shared Concepts, 59

EFFECT statement
- collection effect (Shared Concepts), 23
- multimember effect (Shared Concepts), 23
- polynomial effect (Shared Concepts), 25
- spline effect (Shared Concepts), 28
- syntax (Shared Concepts), 21
eigenvalue decomposition method
- PCA procedure, 705, 724
eigenvalues
- ICA procedure, 444
- PCA procedure, 729
eigenvalues and eigenvectors
- PCA procedure, 724, 727, 729
eigenvectors
- PCA procedure, 729
der  nal PDF for Principal Component Analysis
empirical Bayes estimate
- NLMOD procedure, 688
equivalence tests
- binomial proportions (FREQTAB), 224
- relative risk (FREQTAB), 249
- risk difference (FREQTAB), 235
estimated number of clusters
- KCLUS procedure, 475
- estimates for smoothing components
  - GAMMOD procedure, 343
estimation
  - mixed model (LMIXED), 529
  - estimation methods
    - LMIXED procedure, 498
evaluation history
  - TREESPLIT procedure, 1054
  - exact confidence limits
    - odds ratio (FREQTAB), 243
    - proportion difference (FREQTAB), 231
    - proportions (FREQTAB), 218
    - ratio of proportions (FREQTAB), 247
    - relative risks (FREQTAB), 247
    - risk difference (FREQTAB), 231
  - exact \( p \)-values
    - FREQTAB procedure, 273
  - exact tests
    - computational algorithms (FREQTAB), 272
    - computational resources (FREQTAB), 274
    - FREQTAB procedure, 126, 271
    - Monte Carlo estimation (FREQTAB), 134
    - network algorithm (FREQTAB), 272
examples, ASSESS
  - multimember effect, 24
examples, BINNING
  - multimember effect, 24
examples, CARDINALITY
  - multimember effect, 24
examples, CORRELATION
  - multimember effect, 24
examples, GAMMOD
  - multimember effect, 24
examples, GENSELECT
  - multimember effect, 24
examples, ICA
  - multimember effect, 24
examples, KCLUS
  - multimember effect, 24
examples, LMXIED
  - holding covariance parameters fixed, 512
  - multimember effect, 24
  - specifying lower bounds, 512
  - split-plot design, 527
  - starting values from data set, 513
  - subject-specific R matrices, 524
  - unstructured covariance, G-side, 516
examples, LOGSELECT
  - multimember effect, 24
examples, MBC
  - multimember effect, 24
examples, NLMOD
  - multimember effect, 24
examples, NLMOD procedure
boundary specification, 682  
conditional model expression, 699  
enzyme data, 676  
gamma distribution, 693  
join point, 699  
plateau model, 699  
predicted values, 699  
segmented model, 699  
starting values, data table, 687  
starting values, grid, 687  
examples, PARTITION  
  multimember effect, 24  
examples, PCA  
  multimember effect, 24  
examples, PHSELECT  
  multimember effect, 24  
examples, PLSMOD  
  multimember effect, 24  
examples, QTRSELECT  
  multimember effect, 24  
examples, REGSELECT  
  multimember effect, 24  
examples, TREESPLIT  
  multimember effect, 24  
examples, VARIMPUTE  
  multimember effect, 24  
examples, VARREDUCE  
  multimember effect, 24  
expectation-maximization convergence criterion  
  MBC procedure, 627  
expectation-maximization iteration limit  
  MBC procedure, 628  
expectation-maximization technique  
  MBC procedure, 629  
explained variation of variables  
  PCA procedure, 728  
factor-analytic structures  
  LMIXED procedure, 516  
factors  
  PLSMOD procedure, 788  
Farrington-Manning test  
  risk difference (FREQTAB), 234  
finding the number of clusters  
  KCLUS procedure, 467  
Fisher’s exact test  
  FREQTAB procedure, 206  
fit criteria  
  QTRSELECT procedure, 853  
  REGSELECT procedure, 905  
  VARREDUCE procedure, 1179  
fit statistics  
  ASSESS procedure, 1082, 1084  
  GAMMOD procedure, 342  
  GENSELECT procedure, 415  
  LOGSELECT procedure, 597  
  MBC procedure, 638  
  PHSELECT procedure, 776  
  QTRSELECT procedure, 863  
  REGSELECT procedure, 914  
fit summary  
  MBC procedure, 639  
fitting algorithms  
  GAMMOD procedure, 336  
fixed effects  
  LMIXED procedure, 490  
fixed-effects parameters  
  LMIXED procedure, 526  
Fleiss-Cohen weights  
  kappa coefficient (FREQTAB), 259  
folds  
  PARTITION procedure, 1145  
formula  
  BINNING procedure, 1105  
forward selection  
  Shared Concepts, 62  
forward swap selection  
  Shared Concepts, 67  
Freeman-Halton test  
  FREQTAB procedure, 207  
FREQTAB procedure, 114  
  AC1 agreement coefficient, 260  
  adjusted odds ratio (Mantel-Haenszel), 265  
  adjusted relative risks (Mantel-Haenszel), 266  
  Agresti-Caffo confidence limits, 228  
  Agresti-Coull confidence limits, 219  
  ANOVA (row mean scores) statistic, 263  
  bar charts, 184  
  Barnard’s test, 236  
  binomial proportions, 217  
  Blaker confidence limits, 219  
  Bowker’s symmetry test, 254, 255  
  Breslow-Day test, 267  
  cell count data, 198  
  chi-square goodness-of-fit test, 203  
  chi-square tests, 202  
  Clopper-Pearson confidence limits, 218  
  Cochran’s Q test, 254, 261  
  Cochran-Armitage test for trend, 251  
  common odds ratio, 269  
  common risk difference, 237  
  computational resources (exact tests), 274  
  contingency coefficient, 207  
  continuity-adjusted chi-square test, 205  
  correlation statistic, 263  
  Cramér’s V statistic, 207  
  crosstabulation tables, 281  
  default tables, 147
subject index

displayed output, 278
dot plots, 184
equivalence tests, 224
equivalence tests (relative risk), 249
equivalence tests (risk difference), 235
exact confidence limits, 126
exact p-values, 273
exact tests, 126, 271
exact unconditional confidence limits, 231
Farrington-Manning test, 234
Fisher’s exact test, 206
Freeman-Halton test, 207
Gail-Simon test, 271
gamma statistic, 208, 209
general association statistic, 264
Hauck-Anderson confidence limits, 228
I-Square, 268
input data tables, 124
Jeffreys confidence limits, 219
Jonckheere-Terpstra test, 252
kappa coefficient, 254, 255
Kendall’s tau-b statistic, 208, 210
Klingenberg confidence limits, 238
lambda asymmetric, 208, 215
lambda symmetric, 208, 216
likelihood ratio chi-square test, 205
Likelihood ratio confidence limits, 219
Logit confidence limits, 220
ManTEL-Fleiss criterion, 264
ManTEL-Haenszel chi-square test, 205
ManTEL-Haenszel confidence limits, 237
ManTEL-Haenszel statistics, 261
maximum time (exact tests), 134
McNemar’s test, 254
measures of agreement, 254
measures of association, 208
Mid-p confidence limits, 220, 244
Miettinen-Nurminen confidence limits, 229
minimum risk confidence limits, 238
missing values, 198
Monte Carlo estimation (exact tests), 126, 134, 275
mosaic plots, 175
multiway tables, 280
network algorithm, 272
Newcombe confidence limits, 230, 234, 240
noninferiority tests, 222
noninferiority tests (relative risk), 248
noninferiority tests (risk difference), 232
odds ratio, 241
ODS graph names, 293
ODS table names, 288
one-way frequency tables, 279
ordering of levels, 125
output data sets, 136, 276
overall kappa coefficient, 260
Pearson chi-square test, 203
Pearson correlation coefficient, 208, 211
phi coefficient, 207
polychoric correlation coefficient, 208, 214
prevalence-adjusted bias-adjusted kappa, 259
relative risks, 244
risk difference, 226
score confidence limits, 229, 246
scores, 201
sensitivity, 225
simple kappa coefficient, 255
Somers’ D statistics, 208, 211
Spearman rank correlation coefficient, 208, 212
specificity, 225
standardized residuals, 204
Stuart’s tau-c statistic, 208, 210
summary score confidence limits, 240
superiority tests, 223
superiority tests (relative risk), 249
superiority tests (risk difference), 235
tetrachoric correlation coefficient, 214
uncertainty coefficients, 208, 216, 217
Wald confidence limits (risk difference), 230
weighted kappa coefficient, 254, 257
Wilson confidence limits, 220
Yule’s Q statistic, 209
Zelen’s exact test, 268
frequencies and descriptive statistics for nominal
variables
KCLUS procedure, 473
frequency information table
PARTITION procedure, 1145
frequency tables
FREQTAB procedure, 114, 147
one-way (FREQTAB), 279
function-based convergence criteria
ASSESS procedure, 44, 45
BINNING procedure, 44, 45
CARDINALITY procedure, 44, 45
CORRELATION procedure, 44, 45
GAMMOD procedure, 44, 45
GENSELECT procedure, 44, 45
ICA procedure, 44, 45
KCLUS procedure, 44, 45
LMIXED procedure, 44, 45
LOGSELECT procedure, 44, 45
LOGSELECT procedure, 44, 45
MBC procedure, 44, 45
NLMod procedure, 44, 45
PARTITION procedure, 44, 45
PCA procedure, 44, 45
PHSelect procedure, 44, 45
PLSMOD procedure, 44, 45
QTRSELECT procedure, 44, 45
REGSELECT procedure, 44, 45
Shared Concepts, 44, 45
TREESPLIT procedure, 44, 45
VARIMPUTE procedure, 44, 45
VARREDUCE procedure, 44, 45

G matrix
LMIXED procedure, 490, 514, 526
Gail-Simon test
FREQTAB procedure, 271
gamma distribution
NLMOD procedure, 685
gamma statistic
FREQTAB procedure, 208, 209
GAMMOD procedure, 303
B-spline basis, 33
class level, 315, 341
computational method, 81, 82
convergence criterion, 44–46, 48, 315
convergence status, 342
degrees of freedom, 337
dispersion parameter, 338
displayed output, 341
distribution function, 326
estimates for smoothing components, 343
fit statistics, 342
fitting algorithms, 336
function-based convergence criteria, 44, 45
generalized additive models, 333
generalized linear models, 333
gradient-based convergence criteria, 44, 46
input data tables, 315
iteration history, 342
link function, 327
low-rank approximation, 332
model evaluation criteria, 335
model inference, 338
model information, 341
model options summary, 321
multithreading, 81, 340
natural cubic spline basis, 35
number of observations, 341
ODS Graphics, 344
ODS table names, 35, 343
optimization algorithms, 340
optimization techniques, 48, 82
outer iteration, 336
output data tables, 329
OutputCasTables table, 343
parameter estimates, 343
parameter-based convergence criteria, 45, 48
penalized likelihood estimation, 334
performance iteration, 337
response level ordering, 322
response profile, 341
response variable options, 322
spline details, 341
splines and spline bases, 31
test data, 80
tests for smoothing components, 339, 343
thin-plate regression splines, 331
thin-plate smoothing splines, 331
truncated power function (TPF) basis, 32
validation, 80
weighting, 330
Gaussian distribution
NLMOD procedure, 685
general association statistic
Mantel-Haenszel (FREQTAB), 264
general distribution
NLMOD procedure, 685
general effects
Shared Concepts, 58
VARREDUCE procedure, 1185
general linear covariance structure
LMIXED procedure, 516
generalized additive models
GAMMOD procedure, 333
generalized inverse, 532
generalized linear models
GAMMOD procedure, 333
GENSELECT procedure, 364, 384
B-spline basis, 33
candidates for addition or removal, 414
class level, 374, 413
complete separation, 405
computational method, 81, 82
confidence limits, 382
convergence criterion, 44–46, 48, 373
convergence status, 414
correlation matrix, 373, 416
covariance matrix, 373, 415
dimensions, 415
displayed output, 413
distribution function, 382
existence of MLEs, 405
fit statistics, 415
function-based convergence criteria, 44, 45
gradient-based convergence criteria, 44, 46
infinite parameter estimates, 405
input data tables, 373
iteration history, 414
LASSO method, 406
LASSO selection, 406
link function, 384
maximum likelihood estimates, 405
model information, 413
<table>
<thead>
<tr>
<th>Model Options Summary</th>
<th>380</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multithreading</td>
<td>81</td>
</tr>
<tr>
<td>Natural Cubic Spline Basis</td>
<td>35</td>
</tr>
<tr>
<td>Number of Observations</td>
<td>413</td>
</tr>
<tr>
<td>ODS Graph Names</td>
<td>417</td>
</tr>
<tr>
<td>ODS Table Names</td>
<td>35, 416</td>
</tr>
<tr>
<td>Optimization Algorithms</td>
<td>412</td>
</tr>
<tr>
<td>Optimization Techniques</td>
<td>48, 82</td>
</tr>
<tr>
<td>Output Data Tables</td>
<td>387</td>
</tr>
<tr>
<td>OutputCasTables Table</td>
<td>416</td>
</tr>
<tr>
<td>Parameter Estimates</td>
<td>415</td>
</tr>
<tr>
<td>Parameter-Based Convergence Criteria</td>
<td>45, 48</td>
</tr>
<tr>
<td>Response Level Ordering</td>
<td>381</td>
</tr>
<tr>
<td>Response Profile</td>
<td>413</td>
</tr>
<tr>
<td>Response Variable Options</td>
<td>381</td>
</tr>
<tr>
<td>Selected Effects</td>
<td>415</td>
</tr>
<tr>
<td>Selection Information</td>
<td>414</td>
</tr>
<tr>
<td>Selection Reason</td>
<td>415</td>
</tr>
<tr>
<td>Selection Summary</td>
<td>414</td>
</tr>
<tr>
<td>Separation</td>
<td>405</td>
</tr>
<tr>
<td>Splines and Spline Bases</td>
<td>31</td>
</tr>
<tr>
<td>Stop Reason</td>
<td>415</td>
</tr>
<tr>
<td>Test Data</td>
<td>80</td>
</tr>
<tr>
<td>Truncated Power Function (TPF) Basis</td>
<td>32</td>
</tr>
<tr>
<td>Validation</td>
<td>80</td>
</tr>
</tbody>
</table>

**GLM Parameterization**

- Shared Concepts, 54, 1181

**Global Test**

- LOGSELECT procedure, 596

**Gradient**

- LMIXED procedure, 535

**Gradient-Based Convergence Criteria**

- ASSESS procedure, 44, 46
- BINNING procedure, 44, 46
- CARDINALITY procedure, 44, 46
- CORRELATION procedure, 44, 46
- GAMMOD procedure, 44, 46
- GENSELECT procedure, 44, 46
- ICA procedure, 44, 46
- KCLUS procedure, 44, 46
- LMIXED procedure, 44, 46
- LOGSELECT procedure, 44, 46
- MBC procedure, 44, 46
- NLMOD procedure, 44, 46
- PARTITION procedure, 44, 46
- PCA procedure, 44, 46
- PHSELECT procedure, 44, 46
- PLSMOD procedure, 44, 46
- QTRSELECT procedure, 44, 46
- REGSELECT procedure, 44, 46
- Shared Concepts, 44, 46
- TREESPLIT procedure, 44, 46
- VARIMPUTE procedure, 44, 46
- VARREDUCE procedure, 44, 46

**Group LASSO Selection**

- Shared Concepts, 69

**Hauck-Anderson Confidence Limits**

- Risk Difference (FREQTAB), 228

**Hessian**

- PHSELECT procedure, 753

**Hessian Matrix**

- LMIXED procedure, 535

**Heterogeneity**

- LMIXED procedure, 524

**Heterogeneous**

- AR(1) Structure (LMIXED), 516
- Compound-Symmetry Structure (LMIXED), 516
- Covariance Structure (LMIXED), 523
- Toeplitz Structure (LMIXED), 516

**Heterogeneous UC**

- LMIXED procedure, 516

**Heterogeneous Uniform Correlation Structure**

- LMIXED procedure, 522

**HPLOGISTIC Procedure**

- Classification Table, 586

**Huynh-Feldt**

- Structure (LMIXED), 516

**Hypothesis Tests**

- Exact (FREQTAB), 126

**I-Square**

- FREQTAB procedure, 268

**ICA Procedure**

- B-Spline Basis, 33
- Centering and Scaling Information, 443
- Computational Method, 81, 82, 436
- Convergence Criterion, 44–46, 48
- Deflationary Decorrelation Method, 428
- Demixing Matrix, 444
- Dewhiting Transformation Matrix, 444
- Dimension Reduction, 445
- Dimensions, 443
- Displayed Output, 443
- Eigenvalues, 444
- Examples, 445
- Function-Based Convergence Criteria, 44, 45
- Gradient-Based Convergence Criteria, 44, 46
- Input Data Tables, 436
- Introductory Example, 429
- Mixing Matrix, 444
- Model Information, 443
- Multithreading, 81
- Natural Cubic Spline Basis, 35
- Number of Observations, 443
<table>
<thead>
<tr>
<th>Subject Index</th>
<th>1203</th>
</tr>
</thead>
</table>
| ODS table names, 43, 444 | Jonckheere-Terpstra test  
FREQTAB procedure, 252 |
| optimization techniques, 48, 82 | kappa coefficient  
FREQTAB procedure, 254, 255  
weights (FREQTAB), 259 |
| output data tables, 439, 440 | KCLUS procedure, 452  
ABC parameters, 474  
ABC statistics, 474  
B-spline basis, 33  
cluster centroid output data tables, 461  
cluster summary, 471  
cluster summary for mixed variables, 473  
cluster summary for nominal variables, 473  
computational method, 81, 82  
convergence criterion, 44–46, 48  
descriptive statistics, 472  
displayed output, 471  
estimated number of clusters, 475  
finding the number of clusters, 467  
frequencies and descriptive statistics for nominal variables, 473  
function-based convergence criteria, 44, 45  
gradient-based convergence criteria, 44, 46  
input data tables, 458  
iteration history, 472  
missing values, 466  
model information, 471  
multithreading, 81  
natural cubic spline basis, 35  
number of observations, 471  
ODS table names, 43, 475  
optimization techniques, 48, 82  
output data tables, 466  
OutputCasTables, 475  
parameter-based convergence criteria, 45, 48  
splines and spline bases, 31  
standardization, 474  
test data, 80  
truncated power function (TPF) basis, 32  
validation, 80  
within-cluster statistics, 473  
Kendall’s tau-b statistic  
FREQTAB procedure, 208, 210  
Klingenberg confidence limits  
common risk difference (FREQTAB), 238  
Kronecker product structure  
LMIXED procedure, 516  
lag functionality  
NLMOD procedure, 690  
lambda asymmetric  
FREQTAB procedure, 208, 215  
lambda symmetric |
| parameter-based convergence criteria, 45, 48 | imputation information table  
VARIMPUTE procedure, 1160 |
| splines and spline bases, 31 | imputation requests table  
VARIMPUTE procedure, 1159 |
| symmetric decorrelation method, 428 | individual measurement and moving range charts  
central line, 959  
control limit equations, 960  
notation, 959  
subgroup summary statistics, 959 |
| test data, 80 | infinite parameter estimates  
GENSELECT procedure, 405  
LOGSELECT procedure, 583 |
| truncated power function (TPF) basis, 32 | information criteria  
QTRSELECT procedure, 855 |
| validation, 80 | information value  
BINNING procedure, 1107 |
| whitening transformation matrix, 444 | initial values  
LMIXED procedure, 511  
NLMOD procedure, 685 |
| imputation information table | input variables  
ASSESS procedure, 1080 |
| VARIMPUTE procedure, 1159 | interaction effects  
Shared Concepts, 56  
VARREDUCE procedure, 1183 |
| individual measurement and moving range charts | intercept  
Shared Concepts, 55  
VARREDUCE procedure, 1182 |
| central line, 959 | iteration details  
LMIXED procedure, 498 |
| control limit equations, 960 | iteration history  
GAMMOD procedure, 342  
GENSELECT procedure, 414  
KCLUS procedure, 472  
LMIXED procedure, 535  
LOGSELECT procedure, 595  
MBC procedure, 628, 638  
PHSELECT procedure, 774 |
| notation, 959 | iterations  
history (LMIXED), 535  
ITERGS method  
PCA procedure, 705, 725 |
| subgroup summary statistics, 959 | Jeffreys confidence limits  
proportions (FREQTAB), 219 |
FREQTAB procedure, 208, 216
LAR selection
Shared Concepts, 67
LASSO method
GENSELECT procedure, 406
LASSO selection
GENSELECT procedure, 406
LOGSELECT procedure, 584
PHSELECT procedure, 768
Shared Concepts, 68
latent variables
PLSMOD procedure, 788
latent vectors
PLSMOD procedure, 788
LD statistic
PHSELECT procedure, 771
least squares
NLMOD procedure, 691
least squares distribution
NLMOD procedure, 685
left-truncation variable
model parameters (PHSELECT), 761
levelization
Shared Concepts, 49
lift information
ASSESS procedure, 1080, 1084
lift regression information
ASSESS procedure, 1084
likelihood displacement
PHSELECT procedure, 771
likelihood ratio chi-square test
FREQTAB procedure, 205
Likelihood ratio confidence limits
proportions (FREQTAB), 219
linear covariance structure
LMIXED procedure, 516
link function
GAMMOD procedure, 327
GENSELECT procedure, 384
LOGSELECT procedure, 570
listwise deletion, 102
LMIXED procedure, 488
2-D geometric anisotropic structure, 516
Akaike’s information criterion, 536
Akaike’s information criterion (finite sample corrected version), 536
alpha level, 505, 515
ANTE(1) structure, 516
antedependence structure (ARTE), 516
AR(1) structure, 516
ARMA structure, 516
autoregressive moving average structure, 516
autoregressive structure (AR(1)), 516
B-spline basis, 33
banded Toeplitz structure, 516
BLUE, 532
BLUP, 532
boundary constraints, 512, 514
chi-square test, 505
Cholesky covariance matrix (CHOL), 516
Cholesky covariance structure, 520
Cholesky root, 520
class level, 498, 534
common subject, 532
compound symmetry structure (CS), 516
computational method, 81, 82, 534
certainty interval, 515
certainty limits, 505, 515
certainty criteria, 506–508, 535
certainty criterion, 44–46, 48
certainty status, 536
 covariance estimates, 524
covariance parameter estimates, 536
covariance structure, 490, 516, 517
degrees of freedom, 505
dimension information, 534
dimensions, 499
direct product structure, 516
displayed output, 534
distributed computing, 534
EBLUPs, 515, 532
effect name length, 498
estimation methods, 498
factor-analytic structures, 516
fitting information, 536
fixed effects, 490
fixed-effects parameters, 505, 526
function-based convergence criteria, 44, 45, 506
G matrix, 490, 514, 526
general linear covariance structure, 516
generalized inverse, 532
gradient, 535
gradient-based convergence criteria, 44, 46, 506, 507
grid search, 511
Hessian matrix, 535
heterogeneity, 524
heterogeneous AR(1) structure, 516
heterogeneous compound-symmetry structure, 516
heterogeneous covariance structures, 523
heterogeneous Toeplitz structure, 516
heterogeneous UC, 516
heterogeneous uniform correlation structure, 522
Huynh-Feldt structure, 516
infinite degrees of freedom, 505
initial values, 511
input data tables, 497
intercept effect, 505, 514
iteration details, 498
iteration history, 535
iterations, 535
Kronecker product structure, 516
linear covariance structure, 516
Matérn covariance structure, 516
matrix notation, 525
maximum likelihood (ML), 489
microarray data, 538
mixed model, 526
mixed model equations, 498, 531
mixed model theory, 525
model information, 499, 534
multithreading, 81, 534
natural cubic spline basis, 35
Newton-Raphson algorithm, 530
number of observations, 499, 534
ODS table names, 35, 537
optimization information, 535
optimization techniques, 48, 82
output data tables, 500, 510
OutputCasTables table, 537
parameter constraints, 512
parameter-based convergence criteria, 45, 48, 509
positive definiteness, 520
profiling residual variance, 499
R matrix, 490, 524, 526
random effects, 490, 514
random-effects parameters, 515, 526
repeated effects, 523
residual method, 505
residual variance tolerance, 499
restricted maximum likelihood (REML), 489
ridge, 530
Schwarz Bayesian information criterion, 536
spatial anisotropic exponential structure, 516
splines and spline bases, 31
split-plot design, 526
standard linear model, 490
statement positions, 495
subject effect, 515, 524
summary of commands, 495
table names, 537
test data, 80
timing, 537
Toeplitz structure (TOEP), 516
truncated power function (TPF) basis, 32
uniform correlation (UC), 516
uniform correlation structure, 522
unstructured correlations, 516
unstructured covariance matrix (UN), 516
validation, 80
variance components (VC), 516
variance ratios, 512
loadings
PCA procedure, 724, 729
local influence
score residuals (PHSELECT), 771
log-likelihood functions
NLMOD procedure, 692
Logit confidence limits
proportions (FREQTAB), 220
LOGSELECT procedure, 548, 570
association statistics, 597
B-spline basis, 33
candidates for addition or removal, 596
class level, 562, 595
classification table, 560, 597
complete separation, 583
computational method, 81, 82
certainty limits, 570
convergence criterion, 44–46, 48, 559
convergence status, 595
correlation matrix, 560, 597
covariance matrix, 560, 597
dimensions, 596
displayed output, 594
existence of MLEs, 583
fit statistics, 597
function-based convergence criteria, 44, 45
global test, 596
gradient-based convergence criteria, 44, 46
infinite parameter estimates, 583
input data tables, 561
iteration history, 595
LASSO selection, 584
link function, 570
maximum likelihood estimates, 583
memory usage, 593
model information, 594
model options summary, 568
multithreading, 81, 592
natural cubic spline basis, 35
number of observations, 594
ODS graph names, 600
ODS table names, 35, 598
optimization algorithms, 593
optimization techniques, 48, 82
output data tables, 573
OutputCasTables table, 598
parameter estimates, 597
parameter-based convergence criteria, 45, 48
quasi-complete separation, 583
response level ordering, 568
response profile, 594
response variable options, 568
selected effects, 596
Subject Index

selection information, 595
selection reason, 596
selection summary, 596
separation, 583
splines and spline bases, 31
stop reason, 596
test data, 80
truncated power function (TPF) basis, 32
Type III tables, 597
validation, 80

low-rank approximation
GAMMOD procedure, 332

main effects
Shared Concepts, 55
VARREDUCE procedure, 1182

Mantel-Fleiss criterion
FREQTAB procedure, 264
Mantel-Haenszel chi-square test
FREQTAB procedure, 205
Mantel-Haenszel confidence limits
common risk difference (FREQTAB), 237
Mantel-Haenszel statistics
ANOVA (row mean scores) statistic (FREQTAB), 263
correlation statistic (FREQTAB), 263
FREQTAB procedure, 261
general association statistic (FREQTAB), 264
Mantel-Fleiss criterion (FREQTAB), 264

Matérn covariance structure
LMIXED procedure, 516
matrix
notation, theory (LMIXED), 525
maximum likelihood
LMIXED procedure, 489
maximum likelihood estimates
GENSELECT procedure, 405
LOGSELECT procedure, 583
maximum likelihood estimation
mixed model (LMIXED), 530

MBC procedure, 619
B-spline basis, 33
cluster initialization method, 627
computational method, 81, 82
convergence criterion, 44–46, 48
covariance structures summary, 635
displayed output, 638
expectation-maximization convergence criterion, 627
expectation-maximization iteration limit, 628
fit statistics, 638
fit summary, 639

function-based convergence criteria, 44, 45
gradient-based convergence criteria, 44, 45
input data tables, 627
iteration history, 628, 638
mixing estimates, 638
model covariance structure, 627
model information, 638
model selection criteria table, 637
model selection criterion, 627
multithreading, 81
natural cubic spline basis, 35
number of Gaussian components, 628
number of models in summary table, 629
number of observations, 638
number of parameters computation table, 637
ODS table names, 638
output data tables, 631
OutputCasTables table, 639
parameter estimates, 638
parameter singularity criterion, 628
parameter-based convergence criteria, 45, 48
pseudorandom number seed, 628
singularity criterion, 628
splines and spline bases, 31
test data, 80
timing, 639
truncated power function (TPF) basis, 32
validation, 80

McNemar’s test
FREQTAB procedure, 254
measures of association
exact tests (FREQTAB), 209
tests (FREQTAB), 208
median charts
central line, 962
central limit equations, 963
notation, 961
subgroup summary statistics, 962

Mehta-Patel network algorithm
exact tests (FREQTAB), 272
memory usage
LOGSELECT procedure, 593
Mid-p confidence limits
odds ratio (FREQTAB), 244
Mid-p confidence limits
proportions (FREQTAB), 220
Miettinen-Nurminen confidence limits
risk difference (FREQTAB), 229
minimum risk confidence limits
common risk difference (FREQTAB), 238
missing values
CORRELATION procedure, 102
FREQTAB procedure, 198
mixed model (LMIXED)
  descriptive statistics, 535
  estimation, 529
  formulation, 526
  maximum likelihood estimation, 530
  notation, 490
  objective function, 535
  REML computing method, 530
  theory, 525
mixed model equations
  L MIXED procedure, 498, 531
mixing estimates
  MBC procedure, 638
mixing matrix
  ICA procedure, 444
model
  information (GENSELECT), 413
model covariance structure
  MBC procedure, 627
model details
  PLSMOD procedure, 819
model evaluation criteria
  GAMMOD procedure, 335
model inference
  GAMMOD procedure, 338
model information
  GAMMOD procedure, 341
  ICA procedure, 443
  KCLUS procedure, 471
  L MIXED procedure, 499, 534
  LOGSELECT procedure, 594
  MBC procedure, 638
  MODELMA TRIX procedure, 665
  PCA procedure, 728
  PHSELECT procedure, 774
  PLSMOD procedure, 818
  T REESPLIT procedure, 1052
model selection criterion
  MBC procedure, 627
model specification
  NL MOD procedure, 685
  MODELMA TRIX procedure, 651, 663
class level, 665
column names of the design matrix, 664
computational method, 664
dimensions, 665
displayed output, 664
effect details, 665
input data tables, 658
introductory example, 654
model information, 665
model options summary, 663
multithreading, 664
number of observations, 665
ODS table names, 665
output data tables, 663
modified ridit scores
  FREQTAB procedure, 202
Monte Carlo estimation
  exact tests (FREQTAB), 126, 134, 275
mosaic plots
  FREQTAB procedure, 175
multiway tables
  FREQTAB procedure, 114, 147, 280
natural cubic spline basis
  ASSESS procedure, 35
  B I N NING procedure, 35
  CARDINALITY procedure, 35
  CORRELATION procedure, 35
  GAMMOD procedure, 35
  GENSELECT procedure, 35
  ICA procedure, 35
  KCLUS procedure, 35
  L MIXED procedure, 35
  LOGSELECT procedure, 35
  REGSELECT procedure, 35
  VARIMPUTE procedure, 81
  VARREDUCE procedure, 81
  ASSESS procedure, 81
  B I N NING procedure, 81
  CARDINALITY procedure, 81
  CORRELATION procedure, 81
  GAMMOD procedure, 81, 340
  GENSELECT procedure, 81, 412
  ICA procedure, 81
  KCLUS procedure, 81
  L MIXED procedure, 81, 534
  LOGSELECT procedure, 81, 592
  MBC procedure, 81
  MODELMA TRIX procedure, 664
  NL MOD procedure, 81, 694
  PARTITION procedure, 81
  PCA procedure, 81
  PHSELECT procedure, 81, 773
  PLSMOD procedure, 81
  QTRSELECT procedure, 81, 861
  REGSELECT procedure, 81, 910
  T REESPLIT procedure, 81
  VARIMPUTE procedure, 81
  VARREDUCE procedure, 81
  FREQTAB procedure, 202
  FREQTAB procedure, 126, 134, 275
  FREQTAB procedure, 175
  FREQTAB procedure, 114, 147, 280
  ASSESS procedure, 35
  B I N NING procedure, 35
  CARDINALITY procedure, 35
  CORRELATION procedure, 35
  GAMMOD procedure, 35
  GENSELECT procedure, 35
  ICA procedure, 35
  KCLUS procedure, 35
  L MIXED procedure, 35
  LOGSELECT procedure, 35
  REGSELECT procedure, 35
  VARIMPUTE procedure, 81
  VARREDUCE procedure, 81
  FREQTAB procedure, 202
  FREQTAB procedure, 126, 134, 275
  FREQTAB procedure, 175
Subject Index

PHSELECT procedure, 35
PLSMOD procedure, 35
QTRSELECT procedure, 35
REGSELECT procedure, 35
Shared Concepts, 35
TREESPLIT procedure, 35
VARIMPUTE procedure, 35
VARREDUCE procedure, 35

negative binomial distribution
NLMOD procedure, 685

nested effects
Shared Concepts, 56
VARREDUCE procedure, 1184

nested versus crossed effects
Shared Concepts, 56
VARREDUCE procedure, 1184

network algorithm
exact tests (FREQTAB), 272
Newcombe confidence limits
common risk difference (FREQTAB), 240
risk difference (FREQTAB), 230, 234
Newton-Raphson algorithm
LMIXED procedure, 530
NIPALS method
PCA procedure, 705, 724
NLMOD procedure, 673
additional estimates, 684, 697
additional estimates correlation, 697
additional estimates covariance, 697
ANOVA, 696
B-spline basis, 33
Bernoulli distribution, 685
binary distribution, 685
binomial distribution, 685
bounds, 682
computational method, 81, 82
convergence criterion, 44–46, 48
correlation, 697
covariance, 697
dimensions, 695
displayed output, 695
distribution functions, 692
empirical Bayes estimate, 688
fit statistics, 696
function-based convergence criteria, 44, 45
gamma distribution, 685
Gaussian distribution, 685
general distribution, 685
gradient-based convergence criteria, 44, 46
initial values, 685
input data tables, 681
iteration history, 695
lag functionality, 690
least squares, 691
least squares distribution, 685
linear constraints, 696
log-likelihood functions, 696
multithreading, 81, 694
natural cubic spline basis, 35
negative binomial distribution, 685
normal distribution, 685
number of observations, 695
ODS table names, 35, 697
optimization algorithms, 694
optimization techniques, 48, 82
output data tables, 681
parameter estimates, 697
parameter-based convergence criteria, 45, 48
parameters, 695
Poisson distribution, 685
prediction, 688
procedure task timing, 697
programming statements, 690
residual distribution, 685
restrictions, 689
segmented model example, 699
specifications, 695
splines and spline bases, 31
starting values, 685
test data, 80
truncated power function (TPF) basis, 32
validation, 80
noninferiority tests
binomial proportions, 222
relative risk (FREQTAB), 248
risk difference (FREQTAB), 232
nonsingular parameterization
Shared Concepts, 58
normal distribution
NLMOD procedure, 685
np charts
central line, 968
control limit equations, 969
control limit parameters, 969
notation, 967
subgroup summary statistics, 968
null model log likelihood
PHSELECT procedure, 774
number of Gaussian components
MBC procedure, 628
number of models in summary table
MBC procedure, 629
number of observations
GAMMOD procedure, 341
GENSELECT procedure, 413
ICA procedure, 443
KCLUS procedure, 471
Subject Index

LMIXED procedure, 499, 534
LOGSELECT procedure, 594
MBC procedure, 638
MODELMATRIX procedure, 665
PCA procedure, 728
PHSELECT procedure, 774
PLSMOD procedure, 818
QTRSELECT procedure, 862
REGSELECT procedure, 912
TREESPLIT procedure, 1052
VARREDUCE procedure, 1186

number of variables
PCA procedure, 728

objective function
mixed model (LMIXED), 535
odds ratio
Breslow-Day test (FREQTAB), 267
case-control studies (FREQTAB), 241
confidence limits (FREQTAB), 242, 245
exact confidence limits (FREQTAB), 243
likelihood ratio confidence limits (FREQTAB), 243
logit adjusted (FREQTAB), 265
Mantel-Haenszel adjusted (FREQTAB), 265
mid-\(p\) confidence limits (FREQTAB), 244
score confidence limits (FREQTAB), 242
Wald (log) confidence limits (FREQTAB), 242
Zelen’s exact test (FREQTAB), 268

ODS (Output Delivery System)
CORRELATION procedure and, 104
ODS graph names
FREQTAB procedure, 293
GENSELECT procedure, 417
LOGSELECT procedure, 600
PCA procedure, 730
PHSELECT procedure, 778
QTRSELECT procedure, 866
REGSELECT procedure, 917
TREESPLIT procedure, 1055
ODS Graphics
GAMMOD procedure, 344
ODS table names
ASSESS procedure, 35
BINNING procedure, 35
CARDINALITY procedure, 35
CORRELATION procedure, 35
GAMMOD procedure, 35, 343
GENSELECT procedure, 35
ICA procedure, 35, 444
KCLUS procedure, 35
LMIXED procedure, 35
LOGSELECT procedure, 35
MBC procedure, 35
NLMOD procedure, 35
PARTITION procedure, 35
PCA procedure, 35, 729
PHSELECT procedure, 35
PLSMOD procedure, 35, 819
QTRSELECT procedure, 35
REGSELECT procedure, 35
TREESPLIT procedure, 35, 1054
VARIMPUTE procedure, 35
VARREDUCE procedure, 35

optimization algorithms
GAMMOD procedure, 340
GENSELECT procedure, 412
LOGSELECT procedure, 593
NLMOD procedure, 694
PHSELECT procedure, 773

optimization information
LMIXED procedure, 535

optimization techniques
ASSESS procedure, 48, 82
BINNING procedure, 48, 82
CARDINALITY procedure, 48, 82
CORRELATION procedure, 48, 82
GAMMOD procedure, 48, 82
GENSELECT procedure, 48, 82
ICA procedure, 48, 82
KCLUS procedure, 48, 82
LMIXED procedure, 48, 82
LOGSELECT procedure, 48, 82
MBC procedure, 48, 82
NLMOD procedure, 48, 82
PARTITION procedure, 48, 82
PCA procedure, 48, 82
PHSELECT procedure, 48, 82
PLSMOD procedure, 48, 82
QTRSELECT procedure, 48, 82
REGSELECT procedure, 48, 82
Shared Concepts, 48
TREESPLIT procedure, 48, 82
VARIMPUTE procedure, 48, 82
VARREDUCE procedure, 48, 82

options summary
CLASS statement, 13, 1173
EFFECT statement, 22, 378, 503, 566, 661, 758, 806, 844, 897
MODEL statement (LMIXED), 504
PROC BINNING statement, 1098
PROC CARDINALITY statement, 1124
PROC GAMMOD statement, 314
PROC GENSELECT statement, 372
PROC KCLUS statement, 457
PROC LMIXED statement, 496
PROC LOGSELECT statement, 558
Subject Index

PROC MODELMATRIX statement, 658
PROC NLMOD statement, 680
PROC PHSELECT statement, 751
PROC QTRSELECT statement, 839
PROC REGSELECT statement, 893
PROC TREESPLIT statement, 1011
PROC VARREDUCE statement, 1171
PROC MBC statement, 626
RANDOM statement (LMIXED), 514
REPEATED statement (LMIXED), 523
ordering
  of class levels (Shared Concepts), 50
ordinal parameterization
  Shared Concepts, 59
ORTH EFFECT parameterization
  Shared Concepts, 60
ORTHORDINAL parameterization
  Shared Concepts, 60
ORTHOTHERM parameterization
  Shared Concepts, 60
ORTHPOLY parameterization
  Shared Concepts, 60
ORTHREF parameterization
  Shared Concepts, 61
outer iteration
  GAMMOD procedure, 336
OUTP= data table
  CORRELATION procedure, 102
output data tables
  ICA procedure, 439
  PCA procedure, 720
  PLSMOD procedure, 807
  saving correlations in, 110
output table
  TREESPLIT procedure, 1053
OutputCasTables
  KCLUS procedure, 475
OutputCasTables table
  GAMMOD procedure, 343
  GENSELECT procedure, 416
  LMIXED procedure, 537
  LOGSELECT procedure, 598
  MBC procedure, 639
  PARTITION procedure, 1145
  PHSELECT procedure, 776
  QTRSELECT procedure, 864
  REGSELECT procedure, 915
OUTSTAT= data table
  PCA procedure, 726
p charts
  central line, 970
  control limit equations, 970
  control limit parameters, 970
notation, 969
subgroup summary statistics, 969
pairwise deletion, 102
parameter constraints
  LMIXED procedure, 512
parameter estimates
  GAMMOD procedure, 343
  GENSELECT procedure, 415
  LOGSELECT procedure, 597
  MBC procedure, 638
  PHSELECT procedure, 776
  PLSMOD procedure, 819
  QTRSELECT procedure, 864
  REGSELECT procedure, 914
parameter singularity criterion
  MBC procedure, 628
parameter-based convergence criteria
  ASSESS procedure, 45, 48
  BINNING procedure, 45, 48
  CARDINALITY procedure, 45, 48
  CORRELATION procedure, 45, 48
  GAMMOD procedure, 45, 48
  GENSELECT procedure, 45, 48
  ICA procedure, 45, 48
  KCLUS procedure, 45, 48
  LMIXED procedure, 45, 48
  LOGSELECT procedure, 45, 48
  MBC procedure, 45, 48
  NLMOD procedure, 45, 48
  PARTITION procedure, 45, 48
  PCA procedure, 45, 48
  PHSELECT procedure, 45, 48
  PLSMOD procedure, 45, 48
  QTRSELECT procedure, 45, 48
  REGSELECT procedure, 45, 48
  Shared Concepts, 45, 48
  TREESPLIT procedure, 45, 48
  VARIMPUTE procedure, 45, 48
  VARREDUCE procedure, 45, 48
parameterization
  dummy (Shared Concepts), 54
  dummy (VARREDUCE procedure), 1181
  effect (Shared Concepts), 59
  GLM (Shared Concepts), 54, 1181
  nonsingular (Shared Concepts), 58
  ordinal (Shared Concepts), 59
  ORTHEFFECT (Shared Concepts), 60
  ORTHORDINAL (Shared Concepts), 60
  ORTHOTHERM (Shared Concepts), 60
  ORTHPOLY (Shared Concepts), 60
  ORTHREF (Shared Concepts), 61
  polynomial (Shared Concepts), 60
  reference (Shared Concepts), 60
  Shared Concepts, 51
thermometer (Shared Concepts), 59

partial correlation
  PCA procedure, 729
  principal components, 729

partial covariance
  PCA procedure, 729
  principal components, 729

partial least squares
  PLSMOD procedure, 788, 789, 811

partial likelihood
  PHSELECT procedure, 768

PARTITION procedure, 1137
  B-spline basis, 33
  classification level, 1145
  computational method, 81, 82
  convergence criterion, 44–46, 48
  displayed output, 1145
  folds, 1145
  frequency information table, 1145
  function-based convergence criteria, 44, 45
  gradient-based convergence criteria, 44, 46
  input data tables, 1140
  multithreading, 81
  natural cubic spline basis, 35
  ODS table names, 35, 1146
  optimization techniques, 48, 82
  output data tables, 715, 720, 721, 726, 727
  OUTSTAT= data table, 726
  parameter-based convergence criteria, 45, 48
  partial correlation, 729
  partial covariance, 729
  RANDOM method, 705, 725, 738
  regression coefficients, 729
  regression statistics, 728
  simple statistics, 728
  splines and spline bases, 31
  test data, 80
  timing, 729
  total variance, 729
  truncated power function (TPF) basis, 32
  validation, 80

Pearson chi-square test
  FREQTAB procedure, 203

Pearson correlation coefficient
  FREQTAB procedure, 208, 211

Pearson correlation statistics
  Pearson product-moment correlation, 100
  probability values, 100
  suppressing, 95

penalized likelihood estimation
  GAMMOD procedure, 334
  percentage of variation accounted for by extracted factors
  PLSMOD procedure, 819
  performance iteration
  GAMMOD procedure, 337

phi coefficient
  FREQTAB procedure, 207

PHSELECT procedure, 743, 761
  B-spline basis, 33
  candidates for addition or removal, 775
  class level, 753, 774
  computational method, 81, 82
  confidence limits, 761
  convergence criterion, 44–46, 48, 752
  convergence status, 774
correlation matrix, 752, 776
covariance matrix, 753, 776
dimensions, 776
displayed output, 773
fit statistics, 776
function-based convergence criteria, 44, 45
global influence, 771
gradient-based convergence criteria, 44, 46
Hessian, 753
input data tables, 753
iteration history, 774
LASSO selection, 768
left-truncation, 761
likelihood displacement, 771
model information, 774
model options summary, 760
multithreading, 81, 773
natural cubic spline basis, 35
null model log likelihood, 774
number of observations, 774
ODS graph names, 778
ODS table names, 35, 777
optimization algorithms, 773
optimization techniques, 48, 82
output data tables, 762
OutputCasTables table, 776
parameter estimates, 776
parameter-based convergence criteria, 45, 48
partial likelihood, 768
residuals, 771
selected effects, 775
selection information, 774
selection reason, 775
selection summary, 775
splines and spline bases, 31
stop reason, 775
test data, 80
truncated power function (TPF) basis, 32
Type III tables, 776
validation, 80
PLSMOD procedure, 788
algorithms, 802
B-spline basis, 33
centering, 817
centering and scaling information, 818
class level, 803, 818
compared with other procedures, 788
components, 788
computational method, 81, 82, 802
convergence criterion, 44–46, 48
dimensions, 818
displayed output, 818
elements, 821
factors, 788
factors, selecting the number of, 795
function-based convergence criteria, 44, 45
gradient-based convergence criteria, 44, 46
input data tables, 801
introductory example, 792
latent variables, 788
latent vectors, 788
missing values, 817
model details, 819
model information, 818
multithreading, 81
natural cubic spline basis, 35
number of observations, 818
ODS table names, 35, 819
optimization techniques, 48, 82
output data tables, 807, 808
parameter estimates, 819
parameter-based convergence criteria, 45, 48
partial least squares regression, 788, 789, 811
percentage of variation accounted for by extracted factors, 819
predicting new observations, 799
principal component regression, 789, 812
reduced rank regression, 789, 812
scaling, 817
SIMPLS method, 812
splines and spline bases, 31
test data, 80
test set validation, 815, 819, 821
timing, 819
truncted power function (TPF) basis, 32
validation, 80
Poisson distribution
NLMOD procedure, 685
polychoric correlation coefficient
FREQTAB procedure, 208, 214
polynomial effects
Shared Concepts, 55
VARREDUCE procedure, 1182
polynomial parameterization
Shared Concepts, 60
positive definiteness
LMIXED procedure, 520
predicted probability names
TREESPLIT procedure, 1054
prediction
NLMOD procedure, 688
prevalence-adjusted bias-adjusted kappa
FREQTAB procedure, 259
principal component regression (PLSMOD), 789, 812
principal components
interpreting eigenvalues, 710
partiaIfing out variables, 722
PROC BINNING, 1093
PROC BINNING binning methods
   bucket, 1094
cutpoint, 1094
quantile, 1094
tree, 1094
Winsorized, 1094
PROC BINNING features, 1095
profiling residual variance
   LMIXED procedure, 499
programming statements
   NLMOD procedure, 690
proportion difference
   FREQTAB procedure, 226
proportions, see binomial proportions
pseudorandom number seed
   MBC procedure, 628
QTRSELECT procedure, 827, 846
   B-spline basis, 33
candidates for addition or removal, 862
class level, 840, 862
   computational method, 81, 82, 861
convergence criterion, 44–46, 48
diagnostic statistics, 856
dimensions, 862
displayed output, 862
fit criteria, 853
fit statistics, 863
function-based convergence criteria, 44, 45
gradient-based convergence criteria, 44, 46
information criteria, 855
input data tables, 840
introductory example, 831
model options summary, 845
multithreading, 81, 861
natural cubic spline basis, 35
number of observations, 862
ODS graph names, 866
ODS table names, 35, 865
optimization techniques, 48, 82
output data tables, 847
OutputCasTables table, 864
parameter estimates, 864
parameter-based convergence criteria, 45, 48
quantile regression, 851
selected effects, 863
selection information, 862
selection reason, 863
selection summary, 862
significance level criteria, 855
splines and spline bases, 31
stop reason, 863
test data, 80, 858
timing, 864
time, 864
truncated power function (TPF) basis, 32
validation, 80, 858
quantile regression
   QTRSELECT procedure, 851
quasi-complete separation
   GENSELECT procedure, 405
   LOGSELECT procedure, 583
R charts
   central line, 964
   control limit equations, 964
   subgroup summary statistics, 963
R matrix
   LMIXED procedure, 490, 524, 526
random effects
   LMIXED procedure, 490, 514
RANDOM method
   PCA procedure, 705, 725
random-effects parameters
   LMIXED procedure, 526
rank scores
   VARREDUCE procedure, 202
reduce options
   VARREDUCE procedure, 1176
reduce options summary
   VARREDUCE procedure, 1176
reduced error
   TREESPLIT procedure, 1053
reduced rank regression, 789
   PLSMOD procedure, 812
reduced-error pruning
   TREESPLIT procedure, 1031, 1043
reference parameterization
   Shared Concepts, 59
regression
   partial least squares (PROC PLSMOD), 789, 811
   principal component (PROC PLSMOD), 789, 812
   reduced rank (PROC PLSMOD), 789, 812
regression coefficients
   PCA procedure, 729
regression effects
   Shared Concepts, 54, 55
   VARREDUCE procedure, 1182
regression statistics
   PCA procedure, 728
regression trees, see decision trees
REGSELECT procedure, 881, 900
   ANOVA table, 913
   B-spline basis, 33
candidates for addition or removal, 912
class level, 894, 912
   computational method, 81, 82, 910
   convergence criterion, 44–46, 48
diagnostic statistics, 907
dimensions, 912
displayed output, 911
fit criteria, 905
fit statistics, 914
function-based convergence criteria, 44, 45
gradient-based convergence criteria, 44, 46
input data tables, 894
introductory example, 885
model options summary, 899
multithreading, 81, 910
natural cubic spline basis, 35
number of observations, 912
ODS graph names, 917
ODS table names, 35, 915
optimization techniques, 48, 82
output data tables, 901
OutputCasTables table, 915
parameter estimates, 914
parameter-based convergence criteria, 45, 48
selected effects, 913
selection information, 912
selection summary, 912
splines and spline bases, 31
stop reason, 913
test data, 80, 909
timing, 915
truncated power function (TPF) basis, 32
Type III tables, 915
validation, 80, 909
relative risk
equivalence tests (FREQTAB), 249
likelihood ratio confidence limits (FREQTAB), 246
noninferiority tests (FREQTAB), 248
superiority tests (FREQTAB), 249
tests (FREQTAB), 248
Wald confidence limits (FREQTAB), 245
relative risks
cohort studies (FREQTAB), 244
exact confidence limits (FREQTAB), 247
FREQTAB procedure, 244
logit adjusted (FREQTAB), 266
Mantel-Haenszel adjusted (FREQTAB), 266
REML computing method
mixed model (LMIXED), 530
repeated effects
LMIXED procedure, 523
residual distribution
NLMOD procedure, 685
residual maximum likelihood (REML)
LMIXED procedure, 530
residual variance tolerance
LMIXED procedure, 499
residuals
and partial correlation (PCA), 726
partial correlation (PCA), 722
Schoenfeld (PHSELECT), 771
score (PHSELECT), 771
weighted Schoenfeld (PHSELECT), 771
response level ordering
GAMMOD procedure, 322
GENSELECT procedure, 381
LOGSELECT procedure, 568
response profile
GAMMOD procedure, 341
GENSELECT procedure, 413
LOGSELECT procedure, 594
response variable options
GAMMOD procedure, 322
GENSELECT procedure, 381
LOGSELECT procedure, 568
restricted maximum likelihood
LMIXED procedure, 489
restricted maximum likelihood (REML)
LMIXED procedure, 530
restrictions
NLMOD procedure, 689
reverse response level ordering
GAMMOD procedure, 322
GENSELECT procedure, 381
riding
LMIXED procedure, 530
ridit scores
FREQTAB procedure, 202
risk difference
certainty limits (FREQTAB), 228
equivalence tests (FREQTAB), 235
exact confidence limits (FREQTAB), 231
FREQTAB procedure, 226
noninferiority tests (FREQTAB), 232
superiority tests (FREQTAB), 235
tests (FREQTAB), 232
TOST (FREQTAB), 235, 250
risks, see also binomial proportions
FREQTAB procedure, 226
ROC information
ASSESS procedure, 1081, 1084
row mean scores statistic
Mantel-Haenszel (FREQTAB), 263
rules for lack of control, see Shewhart charts, tests for special causes
runs rules, see Shewhart charts, tests for special causes
runs tests, see Shewhart charts, tests for special causes
$s$ chart
central line, 965
control limit equations, 965
notation, 964
subgroup summary statistics, 965
saving correlations
example, 110
Schoenfeld residuals
PHSELECT procedure, 771
Schwarz Bayesian information criterion
LMIXED procedure, 536
score confidence limits
odds ratio (FREQTAB), 242
relative risk (FREQTAB), 246
risk difference (FREQTAB), 229
score residuals
PHSELECT procedure, 771
selected effects
GENSELECT procedure, 415
LOGSELECT procedure, 596
PHSELECT procedure, 775
QTRSELECT procedure, 863
REGSELECT procedure, 913
selected variable
VARREDUCE procedure, 1186
selection information
GENSELECT procedure, 414
LOGSELECT procedure, 595
PHSELECT procedure, 774
QTRSELECT procedure, 862
REGSELECT procedure, 912
selection plots
Shared Concepts, 71
selection reason
GENSELECT procedure, 415
LOGSELECT procedure, 596
PHSELECT procedure, 775
QTRSELECT procedure, 863
REGSELECT procedure, 913
SELECTION statement
syntax (Shared Concepts), 36
selection summary
GENSELECT procedure, 414
LOGSELECT procedure, 596
PHSELECT procedure, 775
QTRSELECT procedure, 863
REGSELECT procedure, 912
VARREDUCE procedure, 1186
sensitivity
FREQTAB procedure, 225
separation
GENSELECT procedure, 405
LOGSELECT procedure, 583
Shared Concepts
ABSCONV option, 44
ABSFCONV option, 44
ABSFTOL option, 44
ABSGCONV option, 44
ABSGTOL option, 44
ABSTOL option, 44
ABSXCONV option, 45
ABSXTOL option, 45
adaptive LASSO selection, 69
ANCOVA effects, 55
ANOVA effects, 54
at sign (@) operator, 53, 56, 1183
backward elimination, 64
bar (l) operator, 52, 55, 56, 1183
CLASS statement, 12, 49
classification variables, 49
CODE statement, 16
collection effect (EFFECT statement), 23
colon (:) operator, 53
collection effect (EFFECT statement), 23
dummy parameterization, 54
effect parameterization, 59
EFFECT statement, 21
FCONV option, 45
FCONV2 option, 45
forward selection, 62
forward swap selection, 67
FTOL option, 45
FTOL2 option, 45
function-based convergence criteria, 44, 45
GCONV option, 46
GCONV2 option, 46
general effects, 58
GLM parameterization, 54, 1181
gradient-based convergence criteria, 44, 46
group LASSO selection, 69
GTOL option, 46
GTOL2 option, 46
interaction effects, 56
intercept, 55
LAR selection, 67
LASSO selection, 68
levelization, 49
main effects, 55
MAXFUNC= option, 47
MAXITER= option, 47
MAXTIME= option, 47
MINITER= option, 47
missing values, CLASS variables, 50
multimember effect (EFFECT statement), 23
nested effects, 56
nested versus crossed effects, 56
nonsingular parameterization, 58
NORMALIZE= option, 48
optimization techniques, 48
ORDER= option, 50
ordering of class levels, 50
ordinal parameterization, 59
ORTHEFFECT parameterization, 60
ORTHORDINAL parameterization, 60
ORTHOTHERM parameterization, 60
ORTHPOLY parameterization, 60
ORTHPREF parameterization, 61
parameter-based convergence criteria, 45, 48
parameterization, 51
PARTITION statement, 36
polynomial effect (EFFECT statement), 25
polynomial effects, 55
polynomial parameterization, 60
reference parameterization, 59
regression effects, 54, 55
selection plots, 71
SELECTION statement, 36
singular parameterization, 55
sort order of class levels, 50
spline basis, B-spline, 33
spline basis, natural cubic spline, 35
spline basis, truncated power function, 32
spline effect (EFFECT statement), 28
stepwise selection, 65
TECHNIQUE= option, 48
thermometer parameterization, 59
XCONV option, 48
XTOL option, 48
Shewhart charts
capability indices, computing, 979
Shewhart charts, tests for special causes, 955
definitions, 974, 975
standard tests, 973–975, 977
standard tests, interpreting, 977
standard tests, modifying, 977
varying subgroup sample sizes, 978
significance level criteria
QTRSELECT procedure, 855
simple statistics
PCA procedure, 728
SIMPLS method
PLSMOD procedure, 812
singular parameterization
Shared Concepts, 55
VARREDUCE procedure, 1183
singularity criterion
MBC procedure, 628
Somers’ D statistics
FREQTAB procedure, 208, 211
sort order
of class levels (Shared Concepts), 50
spatial anisotropic exponential structure
LMIXED procedure, 516
SPC procedure
details, 956
examples, 993
input data tables, 938
reading preestablished control limits, 981
reading process specification limits, 982
Spearman rank correlation coefficient
FREQTAB procedure, 208, 212
specificity
FREQTAB procedure, 225
spline details
GAMMOD procedure, 341
spline basis, B-spline, 33
spline basis, natural cubic spline, 35
spline basis, truncated power function, 32
spline effect (EFFECT statement), 28
stepwise selection, 65
TECHNIQUE= option, 48
thermometer parameterization, 59
XCONV option, 48
XTOL option, 48
spatial anisotropic exponential structure
LMIXED procedure, 516
split-plot design
LMIXED procedure, 526
splitting criteria
TREESPLIT procedure, 1026, 1036
standard deviation, 95
standard linear model
LMIXED procedure, 490
standardization
KCLUS procedure, 474
standardized residuals
FREQTAB procedure, 204
starting values
NLMOD procedure, 685
stepwise selection
Subject Index
1217

Shared Concepts, 65
stop reason
GENSELECT procedure, 415
LOGSELECT procedure, 596
PHSELECT procedure, 775
QTRSELECT procedure, 863
REGSELECT procedure, 913
stratified analysis
FREQTAB procedure, 114, 147
Stuart's tau-c statistic
FREQTAB procedure, 208, 210
subject effect
LMIXED procedure, 515, 524
summary of commands
LMIXED procedure, 495
summary score confidence limits
common risk difference (FREQTAB), 240
sums of squares and crossproducts, 95
superiority tests
binomial proportions (FREQTAB), 223
relative risk (FREQTAB), 249
risk difference (FREQTAB), 235
supervised variable selection
VARREDUCE procedure, 1178
supplementary rules, see Shewhart charts, tests for special causes
surrogate rules
TREESPLIT procedure, 1015, 1038
symmetric decorrelation method
ICA procedure, 428
table names
LMIXED procedure, 537
table scores
FREQTAB procedure, 201
tables
contingency (FREQTAB), 114
crosstabulation (FREQTAB), 114, 281
multiway (FREQTAB), 114, 280
one-way frequency (FREQTAB), 114, 279
target variables
ASSESS procedure, 1080
Tarone's adjustment
Breslow-Day test (FREQTAB), 267
test data
ASSESS procedure, 80
BINNING procedure, 80
CARDINALITY procedure, 80
CORRELATION procedure, 80
GAMMOD procedure, 80
GENSELECT procedure, 80
ICA procedure, 80
KCLUS procedure, 80
LMIXED procedure, 80
LOGSELECT procedure, 80
MBC procedure, 80
NLMOD procedure, 80
PARTITION procedure, 80
PCA procedure, 80
PHSELECT procedure, 80
PLSMOD procedure, 80
QTRSELECT procedure, 80, 858
REGSELECT procedure, 80, 909
TREESPLIT procedure, 80
VARIMPUTE procedure, 80
VARREDUCE procedure, 80
test set validation
PLSMOD procedure, 815, 819
tests for smoothing components
GAMMOD procedure, 339, 343
tetrachoric correlation coefficient
FREQTAB procedure, 214
thermometer parameterization
Shared Concepts, 59
thin-plate regression splines
GAMMOD procedure, 331
thin-plate smoothing splines
GAMMOD procedure, 331
timing
LMIXED procedure, 516
TOST
equivalence tests (FREQTAB), 224, 235, 250
total variance
PCA procedure, 729
TPF basis, see truncated power function (TPF) basis training, validation, and test data
PARTITION procedure, 1145
tree performance
TREESPLIT procedure, 1053
tree-based binning
BINNING procedure, 1107
TREESPLIT procedure, 1001
B-spline basis, 33
best configuration, 1054
C4.5 pruning, 1030, 1042
computational method, 81, 82
convergence criterion, 44–46, 48
cost complexity, 1053
cost-complexity pruning, 1030, 1040
cross validation, 1041
cv statistics, 1052
decision trees, 1033

displayed output, 1052

evaluation history, 1054

function-based convergence criteria, 44, 45

gradient-based convergence criteria, 44, 46

hyperparameter tuning, 1049

input data tables, 1014

model information, 1052

multithreading, 81

natural cubic spline basis, 35

number of observations, 1052

ODS graph names, 1055

ODS table names, 35, 1054

optimization techniques, 48, 82

output data tables, 1029

output table, 1053

parameter-based convergence criteria, 45, 48

predicted probability names, 1054

reduced error, 1053

reduced-error pruning, 1031, 1043

splines and spline bases, 31

splitting criteria, 1026, 1036

surrogate rules, 1015, 1038

test data, 80

tree performance, 1053

truncted power function (TPF) basis, 32

tuner information, 1053

tuner results, 1054

tuner summary, 1053

tuner timing, 1054

validation, 80

variable importance, 1053

trend test

FREQTAB procedure, 251

trimmed statistics

BINNING procedure, 1108

truncted power function (TPF) basis

ASSESS procedure, 32

BINNING procedure, 32

CARDINALITY procedure, 32

CORRELATION procedure, 32

GAMMOD procedure, 32

GENSELECT procedure, 32

ICA procedure, 32

KCLUS procedure, 32

LMIXED procedure, 32

LOGSELECT procedure, 32

MBC procedure, 32

NLMOD procedure, 32

PARTITION procedure, 32

PCA procedure, 32

PHSELECT procedure, 32

PLS­MOD procedure, 32

QTR­SE­LECT procedure, 32

REGSELECT procedure, 32

Shared Concepts, 32

TREESPLIT procedure, 32

VARIMPUTE procedure, 32

VAR­REDUCE procedure, 32

tuner information

TREESPLIT procedure, 1053

tuner results

TREESPLIT procedure, 1054

tuner summary

TREESPLIT procedure, 1053

tuner timing

TREESPLIT procedure, 1054

Type III tables

LOGSELECT procedure, 597

PHSELECT procedure, 776

REGSELECT procedure, 915

u charts

central line, 972

compared with c charts, 972

c­ontrol limit equations, 972

control limit parameters, 972

notation, 971

subgroup summary statistics, 971

uncertainty coefficients

FREQTAB procedure, 208, 216, 217

uniform correlation (UC)

LMIXED procedure, 516

uniform correlation structure

LMIXED procedure, 522

un­structured correlations

LMIXED procedure, 516

unstructured covariance matrix (UN)

LMIXED procedure, 516

unsupervised variable selection

VAR­REDUCE procedure, 1178

validation

ASSESS procedure, 80

BINNING procedure, 80

CARDINALITY procedure, 80

CORRELATION procedure, 80

GAMMOD procedure, 80

GENSELECT procedure, 80

ICA procedure, 80

KCLUS procedure, 80

LMIXED procedure, 80

LOGSELECT procedure, 80

MBC procedure, 80

NLMOD procedure, 80

PARTITION procedure, 80

PCA procedure, 80

PHSELECT procedure, 80
PLSMOD procedure, 80
QTRSELECT procedure, 80, 858
REGSELECT procedure, 80, 909
TREESPLIT procedure, 80
VARIMPUTE procedure, 80
VARREDUCE procedure, 80
variable importance
  TREESPLIT procedure, 1053
variable information value (IV) table
  BINNING procedure, 1108
variable transformation information table
  BINNING procedure, 1109
variance components (VC)
  LMMIXED procedure, 516
variance ratios
  LMMIXED procedure, 512
variances, 95
VARIMPUTE procedure, 1153
  B-spline basis, 33
  computational method, 81, 82
  convergence criterion, 44–46, 48
  displayed output, 1159
  function-based convergence criteria, 44, 45
  gradient-based convergence criteria, 44, 46
  imputation information table, 1160
  imputation requests table, 1159
  input data tables, 1156
  multithreading, 81
  natural cubic spline basis, 35
  ODS table names, 35, 1160
  optimization techniques, 48, 82
  output data tables, 1158
  parameter-based convergence criteria, 45, 48
  splines and spline bases, 31
  test data, 80
  truncated power function (TPF) basis, 32
  validation, 80
VARREDUCE procedure, 1164
  ANCOVA effects, 1182
  ANOVA effects, 1182
  B-spline basis, 33
  bar (l) operator, 1182
  computational method, 81, 82, 1180
  continuous-by-class effects, 1185
  continuous-nesting-class effects, 1184
  convergence criterion, 44–46, 48
  correlations, 1172
  crossed effects, 1183
  displayed output, 1186
  dummy parameterization, 1181
  fit criteria, 1179
  function-based convergence criteria, 44, 45
  general effects, 1185
  gradient-based convergence criteria, 44, 46
input data tables, 1171
interaction effects, 1183
intercept, 1182
main effects, 1182
maximum effects, 1177
maximum steps, 1177
minimal explained variance increment, 1177
missing values, 1177
multithreading, 81
natural cubic spline basis, 35
nested effects, 1184
nested versus crossed effects, 1184
number of observations, 1186
ODS table names, 35, 1186
optimization techniques, 48, 82
parameter-based convergence criteria, 45, 48
polynomial effects, 1182
regression effects, 1182
selected variable, 1186
selection summary, 1186
singular parameterization, 1183
splines and spline bases, 31
supervised variable selection, 1178
test data, 80
truncated power function (TPF) basis, 32
unsupervised variable selection, 1178
validation, 80
variance explained, 1177

Wald confidence limits
  risk difference (FREQTAB), 230
weighted kappa coefficient
  FREQTAB procedure, 254, 257
weighted Schoenfeld residuals
  PHSELECT procedure, 771
weighting
  GAMMOD procedure, 330
Western Electric rules, see Shewhart charts, tests for special causes
whitening
  ICA procedure, 442
whitening transformation matrix
  ICA procedure, 444
Wilson confidence limits
  proportions (FREQTAB), 220
Winsorized statistics
  BINNING procedure, 1108
within-cluster statistics
  KCLUS procedure, 473

$\hat{X}$ charts
  central line, 961
  control limit equations, 961
  subgroup summary statistics, 961
Yule’s \( Q \) statistic
   FREQTAB procedure, 209

Zelen’s test
   equal odds ratios (FREQTAB), 268
   zeros, structural and random
      agreement statistics (FREQTAB), 261
## Syntax Index

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ABSCONV option</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPTIMIZATION statement</td>
<td>506</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>ABSFTOL option</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPTIMIZATION statement</td>
<td>506</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>ABSFCONV option</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPTIMIZATION statement</td>
<td>506</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>ABSGCONV option</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPTIMIZATION statement</td>
<td>506</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>ABSGTOL option</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPTIMIZATION statement</td>
<td>506</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>ABSGTOL option</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
PROC BINNING statement, 44
PROC CARDINALITY statement, 44
PROC CORRELATION statement, 44
PROC GAMMOD statement, 44
PROC GENSELECT statement, 44
PROC ICA statement, 44
PROC KCLUS statement, 44
PROC LMIXED statement, 44
PROC LOGSELECT statement, 44
PROC MBC statement, 44
PROC NLMOD statement, 44
PROC PARTITION statement, 44
PROC PCA statement, 44
PROC PHSELECT statement, 44
PROC PLSMOD statement, 44
PROC QTRSELECT statement, 44
PROC REGSELECT statement, 44
PROC TREESPLIT statement, 44
PROC VARIMPUTE statement, 44
PROC VARREDUCE statement, 44

PROC ASSESS statement, 45
PROC BINNING statement, 45
PROC CARDINALITY statement, 45
PROC CORRELATION statement, 45
PROC GAMMOD statement, 45
PROC GENSELECT statement, 45
PROC ICA statement, 45
PROC KCLUS statement, 45
PROC LMIXED statement, 45
PROC LOGSELECT statement, 45
PROC MBC statement, 45
PROC NLMOD statement, 45
PROC PARTITION statement, 45
PROC PCA statement, 45
PROC PHSELECT statement, 45
PROC PLSMOD statement, 45
PROC QTRSELECT statement, 45
PROC REGSELECT statement, 45
PROC TREESPLIT statement, 45
PROC VARIMPUTE statement, 45
PROC VARREDUCE statement, 45

PROC ASSESS statement, 44
PROC BINNING statement, 44
PROC CARDINALITY statement, 44
PROC CORRELATION statement, 44
PROC GAMMOD statement, 44
PROC GENSELECT statement, 44
PROC ICA statement, 44
PROC KCLUS statement, 44
PROC LMIXED statement, 44
PROC LOGSELECT statement, 44
PROC MBC statement, 44
PROC NLMOD statement, 44
PROC PARTITION statement, 44
PROC PCA statement, 44
PROC PHSELECT statement, 44
PROC PLSMOD statement, 44
PROC QTRSELECT statement, 44
PROC REGSELECT statement, 44
PROC TREESPLIT statement, 44
PROC VARIMPUTE statement, 44
PROC VARREDUCE statement, 44

PROC ASSESS statement, 45
PROC BINNING statement, 45
PROC CARDINALITY statement, 45
PROC CORRELATION statement, 45
PROC GAMMOD statement, 45
PROC GENSELECT statement, 45
PROC ICA statement, 45
PROC KCLUS statement, 45
PROC LMIXED statement, 45
PROC LOGSELECT statement, 45
PROC MBC statement, 45
PROC NLMOD statement, 45
PROC PARTITION statement, 45
PROC PCA statement, 45
PROC PHSELECT statement, 45
PROC PLSMOD statement, 45
PROC QTRSELECT statement, 45
PROC REGSELECT statement, 45
PROC TREESPLIT statement, 45
PROC VARIMPUTE statement, 45
PROC VARREDUCE statement, 45

PROC ASSESS statement, 44
PROC BINNING statement, 44
PROC CARDINALITY statement, 44
PROC CORRELATION statement, 44
PROC GAMMOD statement, 44
PROC GENSELECT statement, 44
PROC ICA statement, 44
PROC KCLUS statement, 44
PROC LMIXED statement, 44
PROC LOGSELECT statement, 44
PROC MBC statement, 44
PROC NLMOD statement, 44
PROC PARTITION statement, 44
PROC PCA statement, 44
PROC PHSELECT statement, 44
PROC PLSMOD statement, 44
PROC QTRSELECT statement, 44
PROC REGSELECT statement, 44
PROC TREESPLIT statement, 44
PROC VARIMPUTE statement, 44
PROC VARREDUCE statement, 44

PROC ASSESS statement, 45
PROC BINNING statement, 45
PROC CARDINALITY statement, 45
PROC CORRELATION statement, 45
PROC GAMMOD statement, 45
PROC GENSELECT statement, 45
PROC ICA statement, 45
PROC KCLUS statement, 45
PROC LMIXED statement, 45
PROC LOGSELECT statement, 45
PROC MBC statement, 45
PROC NLMOD statement, 45
PROC PARTITION statement, 45
PROC PCA statement, 45
PROC PHSELECT statement, 45
PROC PLSMOD statement, 45
PROC QTRSELECT statement, 45
PROC REGSELECT statement, 45
PROC TREESPLIT statement, 45
PROC VARIMPUTE statement, 45
PROC VARREDUCE statement, 45

PROC ASSESS statement, 44
PROC BINNING statement, 44
PROC CARDINALITY statement, 44
PROC CORRELATION statement, 44
PROC GAMMOD statement, 44
PROC GENSELECT statement, 44
PROC ICA statement, 44
PROC KCLUS statement, 44
PROC LMIXED statement, 44
PROC LOGSELECT statement, 44
PROC MBC statement, 44
PROC NLMOD statement, 44
PROC PARTITION statement, 44
PROC PCA statement, 44
PROC PHSELECT statement, 44
PROC PLSMOD statement, 44
PROC QTRSELECT statement, 44
PROC REGSELECT statement, 44
PROC TREESPLIT statement, 44
PROC VARIMPUTE statement, 44
PROC VARREDUCE statement, 44

PROC ASSESS statement, 45
PROC BINNING statement, 45
PROC CARDINALITY statement, 45
PROC CORRELATION statement, 45
PROC GAMMOD statement, 45
PROC GENSELECT statement, 45
PROC ICA statement, 45
PROC KCLUS statement, 45
PROC LMIXED statement, 45
PROC LOGSELECT statement, 45
PROC MBC statement, 45
PROC NLMOD statement, 45
PROC PARTITION statement, 45
PROC PCA statement, 45
PROC PHSELECT statement, 45
PROC PLSMOD statement, 45
PROC QTRSELECT statement, 45
PROC REGSELECT statement, 45
PROC TREESPLIT statement, 45
PROC VARIMPUTE statement, 45
PROC VARREDUCE statement, 45

ADAPTIVE option

SELECTION statement (ASSESS), 38
SELECTION statement (BINNING), 38
SELECTION statement (CARDINALITY), 38
SELECTION statement (CORRELATION), 38
SELECTION statement (GAMMOD), 38
SELECTION statement (GENSELECT), 38
SELECTION statement (ICA), 38
SELECTION statement (KCLUS), 38
SELECTION statement (LMIXED), 38
SELECTION statement (LOGSELECT), 38
SELECTION statement (MBC), 38
SELECTION statement (NLMOD), 38
SELECTION statement (PARTITION), 38
SELECTION statement (PCA), 38
SELECTION statement (PHSELECT), 38
SELECTION statement (PLSMOD), 38
SELECTION statement (QTRSELECT), 38
SELECTION statement (REGSELECT), 38
SELECTION statement (TREESPLIT), 38
SELECTION statement (VARIMPUTE), 38
SELECTION statement (VARREDUCE), 38
ADDITIONAL OPTIONS

ADD option
   VIICODE statement (TREESPLIT), 1032

ADJUST option (PLCORR)
   TABLES statement (FREQTAB), 170

AGREE option
   EXACT statement (FREQTAB), 128
   OUTPUT statement (FREQTAB), 139
   TABLES statement (FREQTAB), 150
   TEST statement (FREQTAB), 194

AIC option
   REDUCE statement, 1177

AICC option
   REDUCE statement, 1177

AJCHI option
   OUTPUT statement (FREQTAB), 139

ALGORITHM= option
   PROC PLSMOD statement, METHOD=PLS option, 802

ALL option
   OUTPUT statement (FREQTAB), 140
   OUTPUT statement (GENSELECT), 387
   OUTPUT statement (LOGSELECT), 573
   OUTPUT statement (MBC), 632
   TABLES statement (FREQTAB), 152

ALLN option
   chart statements (SPC), 949

ALLOBS option
   MODEL statement (GAMMOD), 325

ALLSTATS option
   OUTPUT statement (LMIXED), 510

ALPHA option
   PROC CORRELATION statement, 95

ALPHA= option
   ESTIMATE statement (NLMOD), 684
   EXACT statement (FREQTAB), 134
   GROW statement (TREESPLIT), 1027
   OUTPUT statement (GAMMOD), 329
   OUTPUT statement (GENSELECT), 387
   OUTPUT statement (LMIXED), 510
   OUTPUT statement (LOGSELECT), 573
   PREDICT statement (NLMOD), 688
   PROC GAMMOD statement, 315
   PROC GENSELECT statement, 373
   PROC LOGSELECT statement, 559
   PROC NLMOD statement, 681
   PROC PHSELECT statement, 752
   PROC QTRSELECT statement, 840
   PROC REGSELECT statement, 894
   PRUNE statement (TREESPLIT), 894
   RANDOM statement (LMIXED), 515
   TABLES statement (FREQTAB), 152

ASSESS procedure, BY statement, 1078

ASSESS procedure, CLASS statement
   DESCENDING option, 13, 1173
   MISSING option, 13, 1173
   ORDER= option, 13
   PARAM= option, 14
   REF= option, 15
   SPLIT option, 15

ASSESS procedure, CODE statement
   COMMENT option, 17
   FILE= option, 17
   INDENTSWIDTH= option, 17
   LABELID= option, 18
   LINESIZE= option, 18
   NOTRIM option, 18
   OUT= option, 18

ASSESS procedure, EFFECT statement
   BASIS option (spline), 28
   DATABOUNDARY option (spline), 28
   DEGREE option (polynomial), 26
   DEGREE option (spline), 29
   DETAILS option (multimember), 25
   DETAILS option (polynomial), 26
   DETAILS option (spline), 29
   KNOTMAX= option (spline), 29
   KNOTMETHOD option (spline), 29
   KNOTMIN= option (spline), 30
   MDEGREE option (polynomial), 26
   NATURALCUBIC option (spline), 30
   NOEFFECT option (multimember), 25
   NOSEPARATE option (polynomial), 26
   SEPARATE option (spline), 30
   SPLIT option (spline), 31
   STANDARDIZE option (polynomial), 26

ASSESS procedure, FITSTAT statement, 1079
   DELIMITER= option, 1079
   DLM= option, 1079
   PEVENT= option, 1079
   PVAR= option, 1079

ASSESS procedure, INPUT statement, 1080

ASSESS procedure, PROC ASSESS statement, 1077
   ABSCONV option, 44
   ABSFCONV option, 44
   ABSFTOL option, 44
   ABSGCONV option, 44
   ABSGTOL option, 44
   ABSTOL option, 44
   ABSXCONV option, 45
   ABSXTOL option, 45
   DATA= option, 1077
   FCNV option, 45
   FCNV2 option, 45
   FITSTATOUT= option, 1078
   FTOL option, 45
   FTOL2 option, 45
   GCONV option, 46
GCONV2 option, 46
GTOL option, 46
GTOL2 option, 46
LIFTOUT= option, 1078
MAXFUNC= option, 47, 1078
MAXITER= option, 47, 1078
MAXTIME= option, 47
MINITER= option, 47
NBINS= option, 1078
NCUTS= option, 1078
NORMALIZE= option, 48
NTHREADS= option, 1078
ROCOUT= option, 1078
TECHNIQUE= option, 48
XCONV option, 48
XTOL option, 48

ASSESS procedure, SELECTION statement
ADAPTIVE option, 38
CHOOSE= option, 38
COMPETITIVE option, 38
CRITERION= option, 38
DETAILS= option, 40, 41
FAST option, 39
HIERARCHY= option, 41
LSŒEPS option, 39
MAXEFFECTS= option, 39
MAXSTEPS= option, 39
METHOD= option, 37
MINEFFECTS= option, 39
ORDERSELECT option, 41
SELECT= option, 39
SELECTION= option, 43
SLE= option, 39
SLENTRY= option, 39
SLS= option, 39
SLSTAY= option, 39
STOP= option, 40
STOPHORIZON= option, 43

ASSESS procedure, syntax, 1077
ASSESS procedure, TARGET statement, 1080
EVENT= option, 1080
LEVEL= option, 1080
ASSESS procedure, VAR statement, 1080
ASSIGNMISSING= option
PROC TREESPLIT statement, 1012
ASSOCIATION option
PROC LOGSELECT statement, 559
AUTOTUNE statement
TREESPLIT procedure, 1018

BARNARD option
EXACT statement (FREQTAB), 129
BASIS option
EFFECT statement, spline (ASSESS), 28
EFFECT statement, spline (BINNING), 28
EFFECT statement, spline (CARDINALITY), 28
EFFECT statement, spline (CORRELATION), 28
EFFECT statement, spline (GAMMOD), 28
EFFECT statement, spline (GENSELECT), 28
EFFECT statement, spline (ICA), 28
EFFECT statement, spline (KCLUS), 28
EFFECT statement, spline (LMIXED), 28
EFFECT statement, spline (LOGSELECT), 28
EFFECT statement, spline (MBC), 28
EFFECT statement, spline (NLMOD), 28
EFFECT statement, spline (PARTITION), 28
EFFECT statement, spline (PCA), 28
EFFECT statement, spline (PHSELECT), 28
EFFECT statement, spline (PLSMOD), 28
EFFECT statement, spline (QTRSELECT), 28
EFFECT statement, spline (REGSELECT), 28
EFFECT statement, spline (TREESPLIT), 28
EFFECT statement, spline (VARIMPUTE), 28
EFFECT statement, spline (VARREDUCE), 28

BDCHI option
OUTPUT statement (FREQTAB), 140
BDT option (CMH)
TABLES statement (FREQTAB), 160
BEST= option
PROC CORRELATION statement, 95
BIC option
REDUCE statement, 1177
BINEPS= option
PROC LOGSELECT statement, 559
BINMETHOD = option
PROC TREESPLIT statement, 1013
BINMISSING= option
PROC BINNING statement, 1101
BINNING procedure, CLASS statement
DESCENDING option, 13, 1173
MISSING option, 13, 1173
ORDER= option, 13
PARAM= option, 14
REF= option, 15
SPLIT option, 15

BINNING procedure, CODE statement, 1102
COMMENT option, 17
FILE= option, 17, 1102
FORMATWIDTH= option, 17
INDENTSIZE= option, 17
LABELID= option, 18
LINESIZE= option, 18
NOTRIM option, 18
OUT= option, 18

BINNING procedure, EFFECT statement
BASIS option (spline), 28
DATABOUNDARY option (spline), 28
DEGREE option (polynomial), 26
DEGREE option (spline), 29
DETAILS option (multimember), 25
DETAILS option (polynomial), 26
DETAILS option (spline), 29
KNOTMAX= option (spline), 29
KNOTMETHOD option (spline), 29
KNOTMIN= option (spline), 30
MDEGREE option (polynomial), 26
NATURALCUBIC option (spline), 30
NOEFFECT option (multimember), 25
NOSEPARATE option (polynomial), 26
SEPARATE option (spline), 30
SPLIT option (spline), 31
STANDARDIZE option (polynomial), 26
BINNING procedure, INPUT statement, 1102
   LEVEL= option, 1103
   NUMBIN= option, 1103
BINNING procedure, OUTPUT statement, 1103
   COPYVARS= option, 1104
   OUT= option, 1104
   OUTLEVELBINMAP= option, 1104
BINNING procedure, PROC BINNING statement, 1098
   ABSCONV option, 44
   ABSFCCONV option, 44
   ABSFTOL option, 44
   ABSGCONV option, 44
   ABSGTOL option, 44
   ABSTOL option, 44
   ABSXCONV option, 45
   ABSXTOL option, 45
   BINMISSING= option, 1101
   DATA= option, 1099
   DISTINCTCOUNTLIMIT= option, 1101
   FCONV option, 45
   FCONV2 option, 45
   FTOL option, 45
   FTOL2 option, 45
   GCONV option, 46
   GCONV2 option, 46
   GTOL option, 46
   GTOL2 option, 46
   MAXFUNC= option, 47
   MAXITER= option, 47
   MAXTIME= option, 47
   METHOD= option, 1099
   MINITER= option, 47
   MISSINGBINSTATS= option, 1102
   MISSINGEVLENONEVENT= option, 1102
   NORMALIZE= option, 48
   NUMBIN= option, 1101
   TECHNIQUE= option, 48
   WOE(WOEADJUST=) option, 1101
   XCONV option, 48
   XTOL option, 48
BINNING procedure, SELECTION statement
   ADAPTIVE option, 38
   CHOOSE= option, 38
   COMPETITIVE option, 38
   CRITERION= option, 38
   DETAILS= option, 40, 41
   FAST option, 39
   HIERARCHY= option, 41
   LSCEFFS option, 39
   MAXEFFECTS= option, 39
   MAXSTEPS= option, 39
   METHOD= option, 37
   MINEFFECTS= option, 39
   ORDERSELECT option, 41
   SELECT= option, 39
   SELECTION= option, 43
   SLE= option, 39
   SLEENTRY= option, 39
   SLS= option, 39
   SLSTAY= option, 39
   STOP= option, 40
   STOPHORIZON= option, 43
BINNING procedure, syntax, 1098
BINNING procedure, TARGET statement, 1104
   EVENT= option, 1104
   LEVEL= option, 1104
BINOMIAL option
   EXACT statement (FREQTAB), 129
   OUTPUT statement (FREQTAB), 140
   TABLES statement (FREQTAB), 153
BLUP statement
   LMIXED procedure, 499
BONFERRONI option
   GROW statement (TREESPLIT), 1027
BOUNDS statement
   NLMOD procedure, 682
BOWKER option
   OUTPUT statement (FREQTAB), 147
BOXCHART statement
   BOXCHART procedure, 940
BY statement
   ASSESS procedure, 1078
   CORRELATION procedure, 97
   FREQTAB procedure, 126
   GAMMOD procedure, 317
   GENSELECT procedure, 375
   ICA procedure, 438
   LMIXED procedure, 500
   LOGSELECT procedure, 563
   MBC procedure, 629
   MODELMATRIX procedure, 658
   NLMOD procedure, 682
   PARTITION procedure, 1142
PCA procedure, 718
PHSELECT procedure, 754
PLSMOD procedure, 803
QTRSELECT procedure, 840
REGSELECT procedure, 894
SPC procedure, 939

C45 option
PRUNE statement (TREESPLIT), 1030

CARDINALITY procedure
VAR statement, 1126
CARDINALITY procedure, CLASS statement
DESCENDING option, 13, 1173
MISSING option, 13, 1173
ORDER= option, 13
PARAM= option, 14
REF= option, 15
SPLIT option, 15
CARDINALITY procedure, CODE statement
COMMENT option, 17
FILE= option, 17
FORMATWIDTH= option, 17
INDENTSIZE= option, 17
LABELID= option, 18
LINESIZE= option, 18
NOTRIM option, 18
OUT= option, 18
CARDINALITY procedure, EFFECT statement
BASIS option (spline), 28
DATABOUNDARY option (spline), 28
DEGREE option (polynomial), 26
DEGREE option (spline), 29
DETAILS option (multimember), 25
DETAILS option (polynomial), 26
DETAILS option (spline), 29
KNOTMAX= option (spline), 29
KNOTMETHOD option (spline), 29
KNOTMIN= option (spline), 30
MDEGREE option (polynomial), 26
NATURALCUBIC option (spline), 30
NOEFFECT option (multimember), 25
NOSEPARATE option (polynomial), 26
SEPARATE option (spline), 30
SPLIT option (spline), 31
STANDARDIZE option (polynomial), 26
CARDINALITY procedure, PROC CARDINALITY statement, 1124
ABS_CONV option, 44
ABSFC_CONV option, 44
ABSF_TOL option, 44
ABSG_CONV option, 44
ABSGTOL option, 44
ABSTOL option, 44
ABSEX_CONV option, 45
ABSXTOL option, 45
DATA= option, 1124
FCONV option, 45
FC_CONV2 option, 45
FTOL option, 45
FT_CONV2 option, 45
G_CONV option, 46
G_CONV2 option, 46
GTOL option, 46
GT_CONV2 option, 46
MAX_FUNC= option, 47
MAXITER= option, 47
MAXLEVELS= option, 1125
MAXTIME= option, 47
MINITER= option, 47
NORMALIZE= option, 48
ORDER= option, 1125
OUTCARD= option, 1124
OUTDETAILS= option, 1125
TECHNIQUE= option, 48
X_CONV option, 48
XTOL option, 48
CARDINALITY procedure, SELECTION statement
ADAPTIVE option, 38
CHOOSE= option, 38
COMPETITIVE option, 38
CRITERION= option, 38
DETAILS = option, 40, 41
FAST option, 39
HIERARCHY= option, 40, 41
LS_COEFFS option, 39
MAXEFFECTS= option, 39
MAXSTEPS= option, 39
METHOD= option, 37
MINEFFECTS= option, 39
ORDERSELECT option, 41
SELECT= option, 39
SELECTION= option, 43
SLE= option, 39
SLENTRY = option, 39
SLS= option, 39
SLSTAY= option, 39
STOP= option, 40
STOPHORIZON= option, 43
CARDINALITY procedure, syntax, 1124
CARDINALITY procedure, VAR statement
ORDER= option, 1126
CASESENSITIVE option
DISPLAY statement (CORRELATION), 20, 97
DISPLAY statement (GAMMOD), 20, 319
DISPLAY statement (GENSELECT), 20, 377
DISPLAY statement (ICA), 20, 438
DISPLAY statement (KCLUS), 20, 464
DISPLAY statement (LMIXED), 20, 502
DISPLAY statement (LOGSELECT), 20, 565
DISPLAY statement (MBC), 20, 630
DISPLAY statement (MODELMATRIX), 660
DISPLAY statement (NLMOD), 20, 683
DISPLAY statement (PARTITION), 20, 1142
DISPLAY statement (PCA), 20, 719
DISPLAY statement (PHSELECT), 20, 757
DISPLAY statement (PLSMOD), 20, 804
DISPLAY statement (QTRSELECT), 20, 842
DISPLAY statement (REGSELECT), 20, 896
DISPLAY statement (VARREDUCE), 20, 1174

CBAR= option
OUTPUT statement (GENSELECT), 388
OUTPUT statement (LOGSELECT), 574

CCHART statement
CCHART procedure, 940

CELLCHI2 option
TABLES statement (FREQTAB), 157

CENSCALE option
PROC PLSMOD statement, 801

CENTER option
MODEL statement (GENSELECT), 382
MODEL statement (LOGSELECT), 570

CENTERLASSO option
MODEL statement (GENSELECT), 382
MODEL statement (LOGSELECT), 570

CHAID option
GROW statement (TREESPLIT), 1027

chart statements
SPC procedure, 949

CHISQ option
EXACT statement (FREQTAB), 129
OUTPUT statement (FREQTAB), 140
TABLES statement (FREQTAB), 157

CHISQUARE option
GROW statement (TREESPLIT), 1027

CHOOSE= option
SELECTION statement (ASSESS), 38
SELECTION statement (BINNING), 38
SELECTION statement (CARDINALITY), 38
SELECTION statement (CORRELATION), 38
SELECTION statement (GAMMOD), 38
SELECTION statement (GENSELECT), 38
SELECTION statement (ICA), 38
SELECTION statement (KCLUS), 38
SELECTION statement (LMIXED), 38
SELECTION statement (LOGSELECT), 38
SELECTION statement (MBC), 38
SELECTION statement (NLMOD), 38
SELECTION statement (PARTITION), 38
SELECTION statement (PCA), 38
SELECTION statement (PHSELECT), 38
SELECTION statement (PLSMOD), 38
SELECTION statement (QTRSELECT), 38

SELECTION statement (REGSELECT), 38
SELECTION statement (TREESPLIT), 38
SELECTION statement (VARIMPUTE), 38
SELECTION statement (VARREDUCE), 38

CIINDICES= option
chart statements (SPC), 950

CL option
MODEL statement (LMIXED), 505
RANDOM statement (LMIXED), 515
TABLES statement (FREQTAB), 159

CL= option (BINOMIAL)
TABLES statement (FREQTAB), 154

CL= option (COMMONRISKDIFF)
TABLES statement (FREQTAB), 161

CL= option (RELRISK)
TABLES statement (FREQTAB), 185

CL= option (RISKDIFF)
TABLES statement (FREQTAB), 188

CL=AGRESTICAFFO option (RISKDIFF)
TABLES statement (FREQTAB), 189

CL=AGRESTICOULL option (BINOMIAL)
TABLES statement (FREQTAB), 154

CL=BLAKER option (BINOMIAL)
TABLES statement (FREQTAB), 154

CL=CLOPPERPEARSON option (BINOMIAL)
TABLES statement (FREQTAB), 154

CL=EXACT option (BINOMIAL)
TABLES statement (FREQTAB), 154

CL=EXACT option (RISKDIFF)
TABLES statement (FREQTAB), 189

CL=HA option (RISKDIFF)
TABLES statement (FREQTAB), 189

CL=JEFFREYS option (BINOMIAL)
TABLES statement (FREQTAB), 155

CL=K option (COMMONRISKDIFF)
TABLES statement (FREQTAB), 162

CL=LIKELIHOODRATIO option (BINOMIAL)
TABLES statement (FREQTAB), 155

CL=LOGIT option (BINOMIAL)
TABLES statement (FREQTAB), 155

CL=MH option (COMMONRISKDIFF)
TABLES statement (FREQTAB), 162

CL=MIDP option (BINOMIAL)
TABLES statement (FREQTAB), 155

CL=MR option (COMMONRISKDIFF)
TABLES statement (FREQTAB), 162

CL=NEWCOMBE option (COMMONRISKDIFF)
TABLES statement (FREQTAB), 162

CL=NEWCOMBEMR option (COMMONRISKDIFF)
TABLES statement (FREQTAB), 162

CL=SCORE option (COMMONRISKDIFF)
TABLES statement (FREQTAB), 162
CL=WALD option (BINOMIAL)  
   TABLES statement (FREQTAB), 155
CL=WALD option (RISKDIFF)  
   TABLES statement (FREQTAB), 190
CL=WILSON option (BINOMIAL)  
   TABLES statement (FREQTAB), 155
CLASS statement  
   ASSESS procedure, 12, 49  
   BINNING procedure, 12, 49  
   CARDINALITY procedure, 12, 49  
   CORRELATION procedure, 12, 49  
   GAMMOD procedure, 12, 49, 318  
   GENSELECT procedure, 12, 49, 375  
   ICA procedure, 12, 49  
   KCLUS procedure, 12, 49  
   LMIXED procedure, 12, 49, 501, 534  
   LOGSELECT procedure, 12, 49, 563  
   MBC procedure, 12, 49  
   NLMOD procedure, 12, 49  
   PARTITION procedure, 12, 49  
   PCA procedure, 12, 49  
   PHSELECT procedure, 12, 49, 754  
   PLSMOD procedure, 12, 49, 803  
   QTRSELECT procedure, 12, 49, 840  
   REGSELECT procedure, 12, 49, 894  
   TREESPLIT procedure, 12, 49, 1025  
   VARIMPUTE procedure, 12, 49  
   VARREDUCE procedure, 12, 49, 1173
CLB option  
   MODEL statement (GENSELECT), 382  
   MODEL statement (LOGSELECT), 570  
   MODEL statement (PHSELECT), 761  
   MODEL statement (QTRSELECT), 846  
   MODEL statement (REGSELECT), 900
CLUSTERSPLIT option  
   PROC TREESPLIT statement, 1014
CMH option  
   OUTPUT statement (FREQTAB), 140  
   TABLES statement (FREQTAB), 160
CMH1 option  
   OUTPUT statement (FREQTAB), 140  
   TABLES statement (FREQTAB), 161
CMH2 option  
   OUTPUT statement (FREQTAB), 141  
   TABLES statement (FREQTAB), 161
CMHCOR option  
   OUTPUT statement (FREQTAB), 141
CMHGA option  
   OUTPUT statement (FREQTAB), 141
CMHRMS option  
   OUTPUT statement (FREQTAB), 141
COCHQ option  
   OUTPUT statement (FREQTAB), 141
CODE statement  
   ASSESS procedure, 16  
   BINNING procedure, 16, 1102  
   CARDINALITY procedure, 16  
   CORRELATION procedure, 16  
   GAMMOD procedure, 16  
   GENSELECT procedure, 16, 376  
   ICA procedure, 16  
   KCLUS procedure, 16, 463  
   LMIXED procedure, 16  
   LOGSELECT procedure, 16, 563  
   MBC procedure, 16  
   NLMOD procedure, 16  
   PARTITION procedure, 16  
   PCA procedure, 16, 718  
   PHSELECT procedure, 16, 754  
   PLSMOD procedure, 16  
   QTRSELECT procedure, 16, 841  
   REGSELECT procedure, 16, 895  
   TREESPLIT procedure, 16, 1026  
   VARIMPUTE procedure, 16, 1157  
   VARREDUCE procedure, 16
COLUMN= option (COMMONRISKDIFF)  
   TABLES statement (FREQTAB), 162
COLUMN= option (RELRRISK)  
   EXACT statement (FREQTAB), 132  
   TABLES statement (FREQTAB), 186
COLUMN= option (RISKDIFF)  
   EXACT statement (FREQTAB), 132  
   TABLES statement (FREQTAB), 190
COMMENT option  
   CODE statement (ASSESS), 17  
   CODE statement (BINNING), 17  
   CODE statement (CARDINALITY), 17  
   CODE statement (CORRELATION), 17  
   CODE statement (GAMMOD), 17  
   CODE statement (GENSELECT), 17  
   CODE statement (ICA), 17  
   CODE statement (KCLUS), 17  
   CODE statement (LMIXED), 17  
   CODE statement (LOGSELECT), 17  
   CODE statement (MBC), 17  
   CODE statement (NLMOD), 17  
   CODE statement (PARTITION), 17  
   CODE statement (PCA), 17  
   CODE statement (PHSELECT), 17  
   CODE statement (PLSMOD), 17  
   CODE statement (QTRSELECT), 17  
   CODE statement (REGSELECT), 17  
   CODE statement (TREESPLIT), 17  
   CODE statement (VARIMPUTE), 17  
   CODE statement (VARREDUCE), 17
COMMON option (RISKDIFF)  
   TABLES statement (FREQTAB), 190
COMMONRISKDIFF option
<table>
<thead>
<tr>
<th>TABLES statement (FREQTAB), 161</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMOR option</td>
</tr>
<tr>
<td>EXACT statement (FREQTAB), 129</td>
</tr>
<tr>
<td>OUTPUT statement (FREQTAB), 143</td>
</tr>
<tr>
<td>COMPETITIVE option</td>
</tr>
<tr>
<td>SELECTION statement (ASSESS), 38</td>
</tr>
<tr>
<td>SELECTION statement (BINNING), 38</td>
</tr>
<tr>
<td>SELECTION statement (CARDINALITY), 38</td>
</tr>
<tr>
<td>SELECTION statement (CORRELATION), 38</td>
</tr>
<tr>
<td>SELECTION statement (GAMMOD), 38</td>
</tr>
<tr>
<td>SELECTION statement (GENSELECT), 38</td>
</tr>
<tr>
<td>SELECTION statement (ICA), 38</td>
</tr>
<tr>
<td>SELECTION statement (KCLUS), 38</td>
</tr>
<tr>
<td>SELECTION statement (LMIXED), 38</td>
</tr>
<tr>
<td>SELECTION statement (MBC), 38</td>
</tr>
<tr>
<td>SELECTION statement (NLMODE), 38</td>
</tr>
<tr>
<td>SELECTION statement (PARTITION), 38</td>
</tr>
<tr>
<td>SELECTION statement (PCA), 38</td>
</tr>
<tr>
<td>SELECTION statement (PHSELECT), 38</td>
</tr>
<tr>
<td>SELECTION statement (PLSMOD), 38</td>
</tr>
<tr>
<td>SELECTION statement (QTRSEXSELECT), 38</td>
</tr>
<tr>
<td>SELECTION statement (REGSELECT), 38</td>
</tr>
<tr>
<td>SELECTION statement (TREESPLIT), 38</td>
</tr>
<tr>
<td>SELECTION statement (VARIMPUTE), 38</td>
</tr>
<tr>
<td>SELECTION statement (VARREDUCE), 38</td>
</tr>
<tr>
<td>COMPONENT option</td>
</tr>
<tr>
<td>OUTPUT statement (GAMMOD), 329</td>
</tr>
<tr>
<td>COMPRESS option</td>
</tr>
<tr>
<td>PROC FREQTAB statement, 124</td>
</tr>
<tr>
<td>CONFIDENCE= option</td>
</tr>
<tr>
<td>PRUNE= option</td>
</tr>
<tr>
<td>CONTENTS= option</td>
</tr>
<tr>
<td>TABLES statement (TREESPLIT), 1030</td>
</tr>
<tr>
<td>CONTGY option</td>
</tr>
<tr>
<td>OUTPUT statement (FREQTAB), 141</td>
</tr>
<tr>
<td>CONTROLSTAT= option</td>
</tr>
<tr>
<td>chart statements (SPC), 950</td>
</tr>
<tr>
<td>CONVERGE= option (PLCORR)</td>
</tr>
<tr>
<td>TABLES statement (FREQTAB), 170</td>
</tr>
<tr>
<td>COPYVAR= option</td>
</tr>
<tr>
<td>OUTPUT statement (GAMMOD), 329</td>
</tr>
<tr>
<td>OUTPUT statement (GENSELECT), 387</td>
</tr>
<tr>
<td>OUTPUT statement (LMIXED), 510</td>
</tr>
<tr>
<td>OUTPUT statement (LOGSELECT), 573</td>
</tr>
<tr>
<td>OUTPUT statement (MBC), 632</td>
</tr>
<tr>
<td>OUTPUT statement (PHSELECT), 762</td>
</tr>
<tr>
<td>OUTPUT statement (QTRSEXSELECT), 848</td>
</tr>
<tr>
<td>OUTPUT statement (REGSELECT), 901</td>
</tr>
<tr>
<td>COPYVARS= option</td>
</tr>
<tr>
<td>OUTPUT statement, 466, 1029, 1104, 1144, 1159</td>
</tr>
<tr>
<td>OUTPUT statement (ICA), 440</td>
</tr>
<tr>
<td>OUTPUT statement (OUTDESIGN), 664</td>
</tr>
<tr>
<td>OUTPUT statement (PCA), 721</td>
</tr>
<tr>
<td>OUTPUT statement (PLSMOD), 808</td>
</tr>
<tr>
<td>CORR option</td>
</tr>
<tr>
<td>PROC NLMODE statement, 681</td>
</tr>
<tr>
<td>CORRB option</td>
</tr>
<tr>
<td>PROC GENSELECT statement, 373</td>
</tr>
<tr>
<td>PROC LOGSELECT statement, 560</td>
</tr>
<tr>
<td>PROC PHSELECT statement, 752</td>
</tr>
<tr>
<td>CORRECT option (BINOMIAL)</td>
</tr>
<tr>
<td>TABLES statement (FREQTAB), 155</td>
</tr>
<tr>
<td>CORRECT option (RISKDIFF)</td>
</tr>
<tr>
<td>TABLES statement (FREQTAB), 190</td>
</tr>
<tr>
<td>CORRECT=NO option (COMMONRISKDIFF)</td>
</tr>
<tr>
<td>TABLES statement (FREQTAB), 162</td>
</tr>
<tr>
<td>CORRELATION procedure</td>
</tr>
<tr>
<td>syntax, 94</td>
</tr>
<tr>
<td>CORRELATION procedure, CLASS statement</td>
</tr>
<tr>
<td>DESCENDING option, 13, 1173</td>
</tr>
<tr>
<td>MISSING option, 13, 1173</td>
</tr>
<tr>
<td>ORDER= option, 13</td>
</tr>
<tr>
<td>PARAM= option, 14</td>
</tr>
<tr>
<td>REF= option, 15</td>
</tr>
<tr>
<td>SPLIT option, 15</td>
</tr>
<tr>
<td>CORRELATION procedure, CODE statement</td>
</tr>
<tr>
<td>COMMENT option, 17</td>
</tr>
<tr>
<td>FILE= option, 17</td>
</tr>
<tr>
<td>FORMATWIDTH= option, 17</td>
</tr>
<tr>
<td>INDENTSIZE= option, 17</td>
</tr>
<tr>
<td>LABELID= option, 18</td>
</tr>
<tr>
<td>LINESIZE= option, 18</td>
</tr>
<tr>
<td>NOTRIM option, 18</td>
</tr>
<tr>
<td>OUT= option, 18</td>
</tr>
<tr>
<td>CORRELATION procedure, DISPLAY statement</td>
</tr>
<tr>
<td>CASESENSITIVE option, 20, 97</td>
</tr>
<tr>
<td>EXCLUDE option, 20, 98</td>
</tr>
<tr>
<td>EXCLUDEALL option, 20, 98</td>
</tr>
<tr>
<td>TRACE option, 20, 98</td>
</tr>
<tr>
<td>CORRELATION procedure, DISPLAYOUT statement</td>
</tr>
<tr>
<td>INCLUDEALL option, 21, 98</td>
</tr>
<tr>
<td>NOPLACE option, 21, 98</td>
</tr>
<tr>
<td>REPEATED option, 21, 98</td>
</tr>
<tr>
<td>CORRELATION procedure, EFFECT statement</td>
</tr>
<tr>
<td>BASIS option (spline), 28</td>
</tr>
<tr>
<td>DATABOUNDARY option (spline), 28</td>
</tr>
<tr>
<td>DEGREE option (polynomial), 26</td>
</tr>
<tr>
<td>DEGREE option (spline), 29</td>
</tr>
<tr>
<td>DETAILS option (multimember), 25</td>
</tr>
<tr>
<td>DETAILS option (polynomial), 26</td>
</tr>
<tr>
<td>DETAILS option (spline), 29</td>
</tr>
<tr>
<td>KNOTMAX= option (spline), 29</td>
</tr>
<tr>
<td>KNOTMETHOD option (spline), 29</td>
</tr>
<tr>
<td>KNOTMIN= option (spline), 30</td>
</tr>
<tr>
<td>MDEGREE option (polynomial), 26</td>
</tr>
<tr>
<td>NATURALCUBIC option (spline), 30</td>
</tr>
<tr>
<td>NOEFFECT option (multimember), 25</td>
</tr>
</tbody>
</table>
NOSEPARATE option (polynomial), 26
SEPARATE option (spline), 30
SPLIT option (spline), 31
STANDARDIZE option (polynomial), 26

CORRELATION procedure, PROC CORRELATION
statement, 94
ABSCONV option, 44
ABSFCONV option, 44
ABSFCTOL option, 44
ABSFCOV option, 44
ABSGCTOL option, 44
ABSTOL option, 44
ABSXCONV option, 45
ABSXTOL option, 45
ALPHA option, 95
BEST= option, 95
COV option, 95
CSSCP option, 95
DATA= option, 95
EXCLNPWGT option, 95
FCONV option, 45
FCONV2 option, 45
FTOL option, 45
FTOL2 option, 45
GCONV option, 46
GCONV2 option, 46
GTOL option, 46
GTOL2 option, 46
MAXFUNC= option, 47
MAXITER= option, 47
MAXTIME= option, 47
MINTER= option, 47
NOCORR option, 95
NOMISS option, 95
NOPROB option, 96
NORMALIZE= option, 48
NOSIMPLE option, 96
OUT= option, 96
OUTP= option, 96
RANK option, 96
SCCP option, 96
TECHNIQUE= option, 48
VARDEF= option, 96
XCONV option, 48
XTOL option, 48

CORRELATION procedure, SELECTION statement
ADAPTIVE option, 38
CHOOS= option, 38
COMPETITIVE option, 38
CRITERION= option, 38
DETAILS= option, 40, 41
FAST option, 39
HIERARCHY= option, 41
LSCOEFFS option, 39
MAXEFFECTS= option, 39
MAXSTEPS= option, 39
METHOD= option, 37
MINEFFECTS= option, 39
ORDERSELECT option, 41
SELECT= option, 39
SELECT= option, 43
SLE= option, 39
SLENTRY= option, 39
SLS= option, 39
SLSTAY= option, 39
STOP= option, 40
STOPHORIZON= option, 43

CORRELATION procedure, VAR statement, 99
CORRELATION procedure, WEIGHT statement, 99
CORRELATION procedure, WITH statement, 99
COSTCOMPLEXITY option

PRUNE statement (TREESPLIT), 1030

COV option
PROC CORRELATION statement, 95
PROC NLMOD statement, 681
PROC PCA statement, 713

COV=SPARSITY option
PROC QTRSELECT statement, 840

COVARIANCE option
PROC PCA statement, 713

COVB option
PROC GENSELECT statement, 373
PROC LOGSELECT statement, 560
PROC PHSELECT statement, 753

COVSTUCT= option
PROC MBC statement, 627

CRAMV option

OUTPUT statement (FREQTAB), 141

CRITERION= option
MODEL statement (GAMMOD), 325
PROC MBC statement, 627
SELECTION statement (ASSESS), 38
SELECTION statement (BINNING), 38
SELECTION statement (CARDINALITY), 38
SELECTION statement (CORRELATION), 38
SELECTION statement (GAMMOD), 38
SELECTION statement (GENSELECT), 38
SELECTION statement (ICA), 38
SELECTION statement (KCLUS), 38
SELECTION statement (LMixed), 38
SELECTION statement (LOGSELECT), 38
SELECTION statement (MBC), 38
SELECTION statement (NLMod), 38
SELECTION statement (PARTITION), 38
SELECTION statement (PCA), 38
SELECTION statement (PHSELECT), 38
SELECTION statement (PLSMOD), 38
SELECTION statement (QTRSELECT), 38
Syntax Index

SELECTION statement (REGSELECT), 38
SELECTION statement (TREESPLIT), 38
SELECTION statement (VARIMPUTE), 38
SELECTION statement (VARREDUCE), 38

CROSSLIST option
TABLES statement (FREQTAB), 164

CSSCP option
PROC CORRELATION statement, 95

CTABLE option
PROC LOGSELECT statement, 560

CUMCOL option
TABLES statement (FREQTAB), 165

CUMHAZ option
CODE statement (PHSELECT), 17, 755
OUTPUT statement (PHSELECT), 763

CURRCLUS option
OUTPUT statement (MBC), 632

CVCC option
PROC TREESPLIT statement, 1014

CVTEST option
PROC PLSMOD statement, 801

DATA= option
PROC ASSESS statement, 1077
PROC BINNING statement, 1099
PROC CARDINALITY statement, 1124
PROC CORRELATION statement, 95
PROC FREQTAB statement, 124
PROC GAMMOD statement, 315
PROC GENSELECT statement, 373
PROC ICA statement, 436
PROC KCLUS statement, 458
PROC LMXED statement, 497
PROC LOGSELECT statement, 561
PROC MBC statement, 627
PROC MODELMATRIX statement, 658
PROC NLMOD statement, 681
PROC PARTITION statement, 1140
PROC PCA statement, 713
PROC PHSELECT statement, 753
PROC PLSMOD statement, 801
PROC QTRSELECT statement, 840
PROC REGSELECT statement, 894
PROC SPC statement, 938, 980
PROC TREESPLIT statement, 1014
PROC VARIMPUTE statement, 1156
PROC VARREDUCE statement, 1171

DATABOUNDARY option
EFFECT statement, spline (ASSESS), 28
EFFECT statement, spline (BINNING), 28
EFFECT statement, spline (CARDINALITY), 28
EFFECT statement, spline (CORRELATION), 28
EFFECT statement, spline (GAMMOD), 28
EFFECT statement, spline (GENSELECT), 28

EFFECT statement, spline (ICA), 28
EFFECT statement, spline (KCLUS), 28
EFFECT statement, spline (LMIXED), 28
EFFECT statement, spline (LOGSELECT), 28
EFFECT statement, spline (MBC), 28
EFFECT statement, spline (NLMOD), 28
EFFECT statement, spline (PARTITION), 28
EFFECT statement, spline (PCA), 28
EFFECT statement, spline (PHSELECT), 28
EFFECT statement, spline (PLSMOD), 28
EFFECT statement, spline (QTRSELECT), 28
EFFECT statement, spline (REGSELECT), 28
EFFECT statement, spline (TREESPLIT), 28
EFFECT statement, spline (VARIMPUTE), 28
EFFECT statement, spline (VARREDUCE), 28

DDFM= option
MODEL statement (LMIXED), 505

DEGREE option
EFFECT statement, polynomial (ASSESS), 26
EFFECT statement, polynomial (BINNING), 26
EFFECT statement, polynomial (CARDINALITY), 26
EFFECT statement, polynomial (CORRELATION), 26
EFFECT statement, polynomial (GAMMOD), 26
EFFECT statement, polynomial (GENSELECT), 26
EFFECT statement, polynomial (ICA), 26
EFFECT statement, polynomial (KCLUS), 26
EFFECT statement, polynomial (LMIXED), 26
EFFECT statement, polynomial (PARTITION), 26
EFFECT statement, polynomial (PCA), 26
EFFECT statement, polynomial (PHSELECT), 26
EFFECT statement, polynomial (PLSMOD), 26
EFFECT statement, polynomial (QTRSELECT), 26
EFFECT statement, polynomial (REGSELECT), 26
EFFECT statement, polynomial (TREESPLIT), 26
EFFECT statement, polynomial (VARIMPUTE), 26
EFFECT statement, polynomial (VARREDUCE), 26

EFFECT statement, spline (ASSESS), 29
EFFECT statement, spline (BINNING), 29
EFFECT statement, spline (CARDINALITY), 29
EFFECT statement, spline (CORRELATION), 29
EFFECT statement, spline (GAMMOD), 29
EFFECT statement, spline (GENSELECT), 29
EFFECT statement, spline (ICA), 29
EFFECT statement, spline (KCLUS), 29
EFFECT statement, spline (LMIXED), 29
EFFECT statement, spline (LOGSELECT), 29
EFFECT statement, spline (MBC), 29
EFFECT statement, spline (NLMOD), 29
EFFECT statement, spline (PARTITION), 29
EFFECT statement, spline (PCA), 29
EFFECT statement, spline (PHSELECT), 29
EFFECT statement, spline (PLSMOD), 29
EFFECT statement, spline (QTRSELECT), 29
EFFECT statement, spline (REGSELECT), 29
EFFECT statement, spline (TREESPLIT), 29
EFFECT statement, spline (VARIMPUTE), 29
EFFECT statement, spline (VARREDUCE), 29
EFFECT statement, multimember (GENSELECT), 25
EFFECT statement, multimember (ICA), 25
EFFECT statement, multimember (KCLUS), 25
EFFECT statement, multimember (LMIXED), 25
EFFECT statement, multimember (LOGSELECT), 25
EFFECT statement, multimember (MBC), 25
EFFECT statement, multimember (NLMOD), 25
EFFECT statement, multimember (PARTITION), 25
EFFECT statement, multimember (PCA), 25
EFFECT statement, multimember (PHSELECT), 25
EFFECT statement, multimember (PLSMOD), 25
EFFECT statement, multimember (QTRSELECT), 25
EFFECT statement, multimember (REGSELECT), 25
EFFECT statement, multimember (TREESPLIT), 25
EFFECT statement, multimember (VARIMPUTE), 25
EFFECT statement, polynomial (ASSESS), 26
EFFECT statement, polynomial (BINNING), 26
EFFECT statement, polynomial (CARDINALITY), 26
EFFECT statement, polynomial (CORRELATION), 26
EFFECT statement, polynomial (GENSELECT), 26
EFFECT statement, polynomial (ICA), 26
EFFECT statement, polynomial (KCLUS), 26
EFFECT statement, polynomial (LMIXED), 26
EFFECT statement, polynomial (LOGSELECT), 26
EFFECT statement, polynomial (MBC), 26
EFFECT statement, polynomial (NLMOD), 26
EFFECT statement, polynomial (PARTITION), 26
EFFECT statement, polynomial (PCA), 26
EFFECT statement, polynomial (PHSELECT), 26
EFFECT statement, polynomial (PLSMOD), 26
EFFECT statement, polynomial (QTRSELECT), 26
EFFECT statement, polynomial (REGSELECT), 26
EFFECT statement, polynomial (TREESPLIT), 26
EFFECT statement, polynomial (VARIMPUTE), 26

DELIMITER= option
FITSTAT statement, 1079
DESCENDING option
CLASS statement (ASSESS), 13, 1173
CLASS statement (BINNING), 13, 1173
CLASS statement (CARDINALITY), 13, 1173
CLASS statement (CORRELATION), 13, 1173
CLASS statement (GAMMOD), 13, 1173
CLASS statement (GENSELECT), 13, 1173
CLASS statement (ICA), 13, 1173
CLASS statement (KCLUS), 13, 1173
CLASS statement (LMIXED), 13, 1173
CLASS statement (LOGSELECT), 13, 1173
CLASS statement (MBC), 13, 1173
CLASS statement (NLMOD), 13, 1173
CLASS statement (PARTITION), 13, 1173
CLASS statement (PCA), 13, 1173
CLASS statement (PHSELECT), 13, 1173
CLASS statement (PLSMOD), 13, 1173
CLASS statement (QTRSELECT), 13, 1173
CLASS statement (REGSELECT), 13, 1173
CLASS statement (TREESPLIT), 13, 1173
CLASS statement (VARIMPUTE), 13, 1173
CLASS statement (VARREDUCE), 13, 1173
MODEL statement (GAMMOD), 322
MODEL statement (GENSELECT), 381
MODEL statement (LOGSELECT), 568
DETAILS option
EFFECT statement, multimember (ASSESS), 25
EFFECT statement, multimember (BINNING), 25
EFFECT statement, multimember (CARDINALITY), 25
EFFECT statement, multimember (CORRELATION), 25
EFFECT statement, multimember (GENSELECT), 25
EFFECT statement, polynomial (GAMMOD), 26
EFFECT statement, polynomial (GENSELECT), 26
EFFECT statement, polynomial (ICA), 26
EFFECT statement, polynomial (KCLUS), 26
EFFECT statement, polynomial (LMIXED), 26
EFFECT statement, polynomial (LOGSELECT), 26
EFFECT statement, polynomial (MBC), 26
EFFECT statement, polynomial (NLMOD), 26
EFFECT statement, polynomial (PARTITION), 26
EFFECT statement, polynomial (PCA), 26
EFFECT statement, polynomial (PHSELECT), 26
EFFECT statement, polynomial (PLSMOD), 26
EFFECT statement, polynomial (QTRSELECT), 26
EFFECT statement, polynomial (REGSELECT), 26
EFFECT statement, polynomial (TREESPLIT), 26
EFFECT statement, polynomial (VARIMPUTE), 26
| EFFECT statement, polynomial (V ARREDUCE) | PREDICT statement (NLMOD), 688 |
| EFFECT statement, spline (ASSESS) | PROCESS NLMOD statement, 681 |
| EFFECT statement, spline (BINNING) | DF= option (CHISQ) |
| EFFECT statement, spline (CARDINALITY) | TABLES statement (FREQQTAB), 158 |
| EFFECT statement, spline (CORRELATION) | DFBETA option |
| EFFECT statement, spline (GAMMOD) | OUTPUT statement (PHSELECT), 763 |
| EFFECT statement, spline (GENSELECT) | DIFCHISQ= option |
| EFFECT statement, spline (ICA) | OUTPUT statement (GENSELECT), 388 |
| EFFECT statement, spline (KCLUS) | OUTPUT statement (LOGSELECT), 574 |
| EFFECT statement, spline (LOGSELECT) | DIFDEV= option |
| EFFECT statement, spline (MBC) | OUTPUT statement (GENSELECT), 388 |
| EFFECT statement, spline (NLMOD) | OUTPUT statement (LOGSELECT), 574 |
| EFFECT statement, spline (PARTITION) | DISPERSION= option |
| EFFECT statement, spline (PCA) | MODEL statement (GAMMOD), 326 |
| EFFECT statement, spline (PHSELECT) | DISPLAY statement |
| EFFECT statement, spline (PLSMOD) | CORRELATION procedure, 19, 97 |
| EFFECT statement, spline (QTRSELECT) | GAMMOD procedure, 19, 318 |
| EFFECT statement, spline (REGSELECT) | GENSELECT procedure, 19, 376 |
| EFFECT statement, spline (TREESPLIT) | ICA procedure, 19, 438 |
| EFFECT statement, spline (VARIMPUTE) | KCLUS procedure, 19, 463 |
| EFFECT statement, spline (VARREDUCE) | LMIXED procedure, 19, 501 |
| MODEL statement (GAMMOD) | LOGSELECT procedure, 19, 564 |
| PROC PLSMOD statement | MBC procedure, 19, 629 |
| DETAILS= option | MODELMATRIX procedure, 659 |
| SELECTION statement (ASSESS), 40, 41 | NLMOD procedure, 19, 682 |
| SELECTION statement (BINNING), 40, 41 | PARTITION procedure, 19, 1142 |
| SELECTION statement (CARDINALITY), 40, 41 | PCA procedure, 19, 718 |
| SELECTION statement (CORRELATION), 40, 41 | PHSELECT procedure, 19, 756 |
| SELECTION statement (GAMMOD), 40, 41 | PLSMOD procedure, 19, 804 |
| SELECTION statement (GENSELECT), 40, 41 | QTRSELECT procedure, 19, 842 |
| SELECTION statement (ICA), 40, 41 | REGSELECT procedure, 19, 895 |
| SELECTION statement (KCLUS), 40, 41 | VARREDUCE procedure, 19, 1174 |
| SELECTION statement (LMIXED), 40, 41 | DISPLAYOUT statement |
| SELECTION statement (LOGSELECT), 40, 41 | CORRELATION procedure, 20, 98 |
| SELECTION statement (MBC), 40, 41 | GAMMOD procedure, 20, 319 |
| SELECTION statement (NLMOD), 40, 41 | GENSELECT procedure, 20, 377 |
| SELECTION statement (PARTITION), 40, 41 | ICA procedure, 20, 439 |
| SELECTION statement (PCA), 40, 41 | KCLUS procedure, 20, 464 |
| SELECTION statement (PHSELECT), 40, 41 | LMIXED procedure, 20, 502 |
| SELECTION statement (PLSMOD), 40, 41 | LOGSELECT procedure, 20, 565 |
| SELECTION statement (QTRSELECT), 40, 41 | MBC procedure, 20, 630 |
| SELECTION statement (REGSELECT), 40, 41 | MODELMATRIX procedure, 660 |
| SELECTION statement (TREESPLIT), 40, 41 | NLMOD procedure, 20, 683 |
| SELECTION statement (VARIMPUTE), 40, 41 | PARTITION procedure, 20, 1143 |
| SELECTION statement (VARREDUCE), 40, 41 | PCA procedure, 20, 719 |
| DEVIATION option | PHSELECT procedure, 20, 757 |
| TABLES statement (FREQQTAB), 165 | PLSMOD procedure, 20, 805 |
| DF= option | QTRSELECT procedure, 20, 843 |
| ESTIMATE statement (NLMOD), 684 | REGSELECT procedure, 20, 896 |
| MODEL statement (GAMMOD), 324 | VARREDUCE procedure, 20, 1175 |
| DISTANCE= option | PROC KCLUS statement, 458 |
| DISTANCENOM= option | PROC KCLUS statement, 459 |
DISTINCTCOUNTLIMIT= option
PROC BINNING statement, 1101
DISTRIBUTION= option
MODEL statement (GAMMOD), 326
MODEL statement (GENSELECT), 382
DLM= option
FITSTAT statement, 1079
DMMETHOD= option
PROC LMIXED statement, 497
ECORR option
PROC NLMOD statement, 681
ECOV option
PROC NLMOD statement, 681
EFFECT statement
ASSESS procedure, 21
BINNING procedure, 21
collection effect, 23
CORRELATION procedure, 21
GAMMOD procedure, 21
GENSELECT procedure, 21, 378
ICA procedure, 21
KCLUS procedure, 21
LMIXED procedure, 21
LOGSELECT procedure, 21, 566
MBC procedure, 21
MODELMATRIX procedure, 661
multimember effect, 23
NLMOD procedure, 21
PARTITION procedure, 21
PCA procedure, 21
PHSELECT procedure, 21, 758
PLSMOD procedure, 21, 805
polynomial effect, 25
QTRSELECT procedure, 21, 843
REGSELECT procedure, 21, 897
spline effect, 28
TREESPLIT procedure, 21
VARIMPUTE procedure, 21
VARREDUCE procedure, 21
EIGENTHRESHOLD= option
PROC ICA statement, 436
EIGHTHRESH= option
PROC ICA statement, 436
EMCRIETERION= option
PROC MBC statement, 627
ENTROPY option
GROW statement (TREESPLIT), 1027
ENTRY= option
MODEL statement (PHSELECT), 761
EPSILON= option
PROC PCA statement, METHOD=ITERGS option, 713
PROC PCA statement, METHOD=NIPALS option, 714
PROC PLSMOD statement, METHOD=PLS option, 802
EQCONS= option
PARMS statement (LMIXED), 512
EQKAP option
OUTPUT statement (FREQTAB), 141
EQOR option
EXACT statement (FREQTAB), 129
OUTPUT statement (FREQTAB), 141
EQUAL option (RELRISK)
TABLES statement (FREQTAB), 186
EQUAL option (RISKDIFF)
TABLES statement (FREQTAB), 190
EQUIVALENCE option (BINOMIAL)
TABLES statement (FREQTAB), 155
EQUIVALENCE option (RELRISK)
TABLES statement (FREQTAB), 186
EQUIVALENCE option (RISKDIFF)
TABLES statement (FREQTAB), 190
EQWKP option
OUTPUT statement (FREQTAB), 141
ESTIMATE statement
NLMOD procedure, 684
EVALHISTORY= option
AUTOTUNE statement, 1019
EVENT= option
PROC PARTITION statement, 1141
TARGET statement, 1080, 1104
EVENTPROP= option
PROC PARTITION statement, 1141
EXACT option
OUTPUT statement (FREQTAB), 142
TABLES statement (FREQTAB), 165
EXACT statement
FREQTAB procedure, 126
EXCHART option
chart statements (SPC), 950
EXCLNPWGRT option
PROC CORRELATION statement, 95
EXCLUDE option
DISPLAY statement (CORRELATION), 20, 98
DISPLAY statement (GAMMOD), 20, 319
DISPLAY statement (GENSELECT), 20, 377
DISPLAY statement (ICA), 20, 439
DISPLAY statement (KCLUS), 20, 464
DISPLAY statement (LMIXED), 20, 502
DISPLAY statement (LOGSELECT), 20, 565
DISPLAY statement (MBC), 20, 630
DISPLAY statement (MODELMATRIX), 660
DISPLAY statement (NLMOD), 20, 683
DISPLAY statement (PARTITION), 20, 1143
DISPLAY statement (PCA), 20, 719
DISPLAY statement (PHSELECT), 20, 757
DISPLAY statement (PLSMOD), 20, 805
DISPLAY statement (QTRSELECT), 20, 842
DISPLAY statement (REGSELECT), 20, 896
DISPLAY statement (VARREduce), 20, 1174
EXCLUDEALL option
  DISPLAY statement (CORRELATION), 20, 98
  DISPLAY statement (GAMMOD), 20, 319
  DISPLAY statement (GENSELECT), 20, 377
  DISPLAY statement (ICA), 20, 439
  DISPLAY statement (KCLUS), 20, 464
  DISPLAY statement (LMixed), 20, 502
  DISPLAY statement (LOGSELECT), 20, 565
  DISPLAY statement (MBC), 20, 630
  DISPLAY statement (MODELMATRIX), 660
  DISPLAY statement (NLMod), 20, 683
  DISPLAY statement (PARTITION), 20, 1143
  DISPLAY statement (PCA), 20, 719
  DISPLAY statement (PHSELECT), 20, 757
  DISPLAY statement (PLSMOD), 20, 805
  DISPLAY statement (QTRSELECT), 20, 842
  DISPLAY statement (REGSELECT), 20, 896
  DISPLAY statement (VARREduce), 20, 1174
EXPECTED option
  TABLES statement (FREQTAB), 165

FAST option
  SELECTION statement (ASSESS), 39
  SELECTION statement (BINNING), 39
  SELECTION statement (CARDINALITY), 39
  SELECTION statement (CORRELATION), 39
  SELECTION statement (GAMMOD), 39
  SELECTION statement (GENSELECT), 39
  SELECTION statement (ICA), 39
  SELECTION statement (KCLUS), 39
  SELECTION statement (LMixed), 39
  SELECTION statement (LOGSELECT), 39
  SELECTION statement (MBC), 39
  SELECTION statement (NLMod), 39
  SELECTION statement (PARTITION), 39
  SELECTION statement (PCA), 39
  SELECTION statement (PHSELECT), 39
  SELECTION statement (PLSMOD), 39
  SELECTION statement (QTRSELECT), 39
  SELECTION statement (REGSELECT), 39
  SELECTION statement (TREESPLIT), 39
  SELECTION statement (VARIMPUTE), 39
  SELECTION statement (VARREduce), 39

FCONV option
  PROC ASSESS statement, 45
  PROC BINNING statement, 45
  PROC CARDINALITY statement, 45
  PROC CORRELATION statement, 45
  PROC GAMMOD statement, 45
  PROC GENSELECT statement, 45
  PROC ICA statement, 45
  PROC KLCLUS statement, 45
  PROC LMixed statement, 45
  PROC LOGSELECT statement, 45
  PROC MBC statement, 45
  PROC NLMod statement, 45
  PROC PARTITION statement, 45
  PROC PCA statement, 45
  PROC PHSELECT statement, 45
  PROC PLSMOD statement, 45
  PROC QTRSELECT statement, 45
  PROC REGSELECT statement, 45
  PROC TREESPLIT statement, 45
  PROC VARIMPUTE statement, 45
  PROC VARREduce statement, 45

FCONV2 option
  PROC ASSESS statement, 45
  PROC BINNING statement, 45
  PROC CARDINALITY statement, 45
  PROC CORRELATION statement, 45
  PROC GAMMOD statement, 45
  PROC GENSELECT statement, 45
  PROC ICA statement, 45
  PROC KLCLUS statement, 45
  PROC LMixed statement, 45
  PROC LOGSELECT statement, 45
  PROC MBC statement, 45
  PROC NLMod statement, 45
  PROC PARTITION statement, 45
  PROC PCA statement, 45
  PROC PHSELECT statement, 45
  PROC PLSMOD statement, 45
  PROC QTRSELECT statement, 45
  PROC REGSELECT statement, 45
  PROC TREESPLIT statement, 45
  PROC VARIMPUTE statement, 45
  PROC VARREduce statement, 45

FCONV2= option
  OPTIMIZATION statement, 507

FCONV= option
  OPTIMIZATION statement, 506

FDHESSIAN= option
  MODEL statement (GAMMOD), 327

FILE= option
  CODE statement, 463, 1102, 1157
  CODE statement (ASSESS), 17
  CODE statement (BINNING), 17
  CODE statement (CARDINALITY), 17
  CODE statement (CORRELATION), 17
  CODE statement (GAMMOD), 17
  CODE statement (GENSELECT), 17
  CODE statement (ICA), 17
  CODE statement (KCLUS), 17
CODE statement (LMIXED), 17
CODE statement (LOGSELECT), 17
CODE statement (MBC), 17
CODE statement (NLMOD), 17
CODE statement (PARTITION), 17
CODE statement (PCA), 17
CODE statement (PHSELECT), 17, 755
CODE statement (PLSMOD), 17
CODE statement (QTRSELECT), 17
CODE statement (REGSELECT), 17
CODE statement (TREESPLIT), 17
CODE statement (VARIMPUTE), 17
CODE statement (VARREDUCE), 17

FISHER option
   EXACT statement (FREQTAB), 129
   OUTPUT statement (FREQTAB), 142
   TABLES statement (FREQTAB), 165

FITSTAT statement
   ASSESS procedure, 1079
   FITSTATOUT= option
   PROC ASSESS statement, 1078
   FIXEDOFFSET= option
   OUTPUT statement (LOGSELECT), 573
   FOLDNAME= option
   OUTPUT statement, 1144
   FORMAT= option
   TABLES statement (FREQTAB), 165
   FORMATWIDTH= option
   CODE statement (ASSESS), 17
   CODE statement (BINNING), 17
   CODE statement (CARDINALITY), 17
   CODE statement (CORRELATION), 17
   CODE statement (GAMMOD), 17
   CODE statement (GENSELECT), 17
   CODE statement (ICA), 17
   CODE statement (KCLUS), 17
   CODE statement (LMIXED), 17
   CODE statement (LOGSELECT), 17
   CODE statement (MBC), 17
   CODE statement (NLMOD), 17
   CODE statement (PARTITION), 17
   CODE statement (PCA), 17
   CODE statement (PHSELECT), 17
   CODE statement (PLSMOD), 17
   CODE statement (QTRSELECT), 17
   CODE statement (REGSELECT), 17
   CODE statement (TREESPLIT), 17
   CODE statement (VARIMPUTE), 17
   CODE statement (VARREDUCE), 17
   FORMCHAR= option
   PROC FREQTAB statement, 124
   FRACTION= option
   AUTOTUNE statement, 1020

FREQ statement
   ASSESS procedure, 1079
   BINNING procedure, 1102
   CARDINALITY procedure, 1125
   CORRELATION procedure, 98
   GAMMOD procedure, 320
   GENSELECT procedure, 379
   KCLUS procedure, 465
   LOGSELECT procedure, 567
   MODELMATRIX procedure, 662
   PCA procedure, 720
   PHSELECT procedure, 760
   REGSELECT procedure, 899
   TREESPLIT procedure, 1026
   VARIMPUTE procedure, 1157
   VARREDUCE procedure, 1175
   FREQNAME= option
   OUTPUT statement, 1144
   FREQTAB procedure
   syntax, 123
   FREQTAB procedure, EXACT statement, 126
      AGREE option, 128
      ALPHA= option, 134
      BARNARD option, 129
      BINOMIAL option, 129
      CHISQ option, 129
      COLUMN= option (RELRISK), 132
      COLUMN= option (RISKDIFF), 132
      COMOR option, 129
      EOR option, 129
      FISHER option, 129
      JT option, 130
      KAPPA option, 130
      KENTB option, 130
      LRCHI option, 130
      MAXTIME= option, 134
      MC option, 134
      MCNEM option, 130
      MEASURES option, 130
      METHOD= option (RELRISK), 132
      METHOD= option (RISKDIFF), 132
      MHCHI option, 131
      MIDP option, 135
      N= option, 135
      OR option, 131
      PCHI option, 131
      PCORR option, 131
      PARTITION statement (PHSELECT), 36, 765
      PARTITION statement (QTRSELECT), 36, 850
      PARTITION statement (REGSELECT), 36, 903
      PARTITION statement (TREESPLIT), 36, 1030
      PLSMOD procedure, PARTITION statement, 810
<table>
<thead>
<tr>
<th>Syntax Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>PFORMAT= option, 135</td>
</tr>
<tr>
<td>POINT option, 135</td>
</tr>
<tr>
<td>RELRISK option, 131</td>
</tr>
<tr>
<td>RISKDIFF option, 132</td>
</tr>
<tr>
<td>SCORR option, 133</td>
</tr>
<tr>
<td>SEED= option, 136</td>
</tr>
<tr>
<td>SMDCR option, 133</td>
</tr>
<tr>
<td>SMDRC option, 133</td>
</tr>
<tr>
<td>STUTC option, 133</td>
</tr>
<tr>
<td>SYMMETRY option, 134</td>
</tr>
<tr>
<td>TREND option, 134</td>
</tr>
<tr>
<td>WTKAPPA option, 134</td>
</tr>
<tr>
<td>ZELEN option, 129</td>
</tr>
<tr>
<td>FREQTAB procedure, OUTPUT statement, 136</td>
</tr>
<tr>
<td>AGRE option, 139</td>
</tr>
<tr>
<td>AJCHI option, 139</td>
</tr>
<tr>
<td>ALL option, 140</td>
</tr>
<tr>
<td>BDCHI option, 140</td>
</tr>
<tr>
<td>BINOMIAL option, 140</td>
</tr>
<tr>
<td>BOWKER option, 147</td>
</tr>
<tr>
<td>CHISQ option, 140</td>
</tr>
<tr>
<td>CMH option, 140</td>
</tr>
<tr>
<td>CMH1 option, 140</td>
</tr>
<tr>
<td>CMH2 option, 141</td>
</tr>
<tr>
<td>CMHCOR option, 141</td>
</tr>
<tr>
<td>CMHGA option, 141</td>
</tr>
<tr>
<td>CMHRMS option, 141</td>
</tr>
<tr>
<td>COCH option, 141</td>
</tr>
<tr>
<td>COMOR option, 143</td>
</tr>
<tr>
<td>CONTGY option, 141</td>
</tr>
<tr>
<td>CRAMV option, 141</td>
</tr>
<tr>
<td>EQKAP option, 141</td>
</tr>
<tr>
<td>EQR option, 141</td>
</tr>
<tr>
<td>EQWK option, 141</td>
</tr>
<tr>
<td>EXACT option, 142</td>
</tr>
<tr>
<td>FISHER option, 142</td>
</tr>
<tr>
<td>GAILS option, 142</td>
</tr>
<tr>
<td>GAMMA option, 142</td>
</tr>
<tr>
<td>JT option, 142</td>
</tr>
<tr>
<td>KAPPA option, 142</td>
</tr>
<tr>
<td>KENTB option, 142</td>
</tr>
<tr>
<td>LAMCR option, 142</td>
</tr>
<tr>
<td>LAMDR option, 142</td>
</tr>
<tr>
<td>LAMRC option, 142</td>
</tr>
<tr>
<td>LGOR option, 143</td>
</tr>
<tr>
<td>LGRC option, 143</td>
</tr>
<tr>
<td>LGRC option, 143</td>
</tr>
<tr>
<td>LRCHI option, 143</td>
</tr>
<tr>
<td>MCNEM option, 143</td>
</tr>
<tr>
<td>MEASURES option, 143</td>
</tr>
<tr>
<td>MHCHI option, 143</td>
</tr>
<tr>
<td>MHOR option, 143</td>
</tr>
<tr>
<td>MHRRRC1 option, 144</td>
</tr>
<tr>
<td>MHRRRC2 option, 144</td>
</tr>
<tr>
<td>N option, 144</td>
</tr>
<tr>
<td>NMISS option, 144</td>
</tr>
<tr>
<td>OR option, 144</td>
</tr>
<tr>
<td>OUT= option, 136</td>
</tr>
<tr>
<td>output-options, 136</td>
</tr>
<tr>
<td>output-options, 136</td>
</tr>
<tr>
<td>PCHI option, 144</td>
</tr>
<tr>
<td>PCORR option, 144</td>
</tr>
<tr>
<td>PLCORR option, 144</td>
</tr>
<tr>
<td>RDIF1 option, 145</td>
</tr>
<tr>
<td>RDIF2 option, 145</td>
</tr>
<tr>
<td>RELRISK option, 145</td>
</tr>
<tr>
<td>RISK1 option, 145</td>
</tr>
<tr>
<td>RISK11 option, 146</td>
</tr>
<tr>
<td>RISK12 option, 146</td>
</tr>
<tr>
<td>RISK2 option, 146</td>
</tr>
<tr>
<td>RISK21 option, 146</td>
</tr>
<tr>
<td>RISK22 option, 146</td>
</tr>
<tr>
<td>RISKDIFF option, 145</td>
</tr>
<tr>
<td>RISKDIFF1 option, 145</td>
</tr>
<tr>
<td>RISKDIFF2 option, 145</td>
</tr>
<tr>
<td>RRC1 option, 145</td>
</tr>
<tr>
<td>RRC2 option, 145</td>
</tr>
<tr>
<td>SCORR option, 146</td>
</tr>
<tr>
<td>SMDCR option, 146</td>
</tr>
<tr>
<td>SMDRC option, 146</td>
</tr>
<tr>
<td>STUTC option, 146</td>
</tr>
<tr>
<td>TAUB option, 142</td>
</tr>
<tr>
<td>TAUC option, 146</td>
</tr>
<tr>
<td>TREND option, 147</td>
</tr>
<tr>
<td>TSYMM option, 147</td>
</tr>
<tr>
<td>U option, 147</td>
</tr>
<tr>
<td>UCR option, 147</td>
</tr>
<tr>
<td>URC option, 147</td>
</tr>
<tr>
<td>WTKAPPA option, 147</td>
</tr>
<tr>
<td>ZELEN option, 141</td>
</tr>
<tr>
<td>FREQTAB procedure, PROC FREQTAB statement, 123</td>
</tr>
<tr>
<td>COMPRESS option, 124</td>
</tr>
<tr>
<td>DATA= option, 124</td>
</tr>
<tr>
<td>FORMCHAR= option, 124</td>
</tr>
<tr>
<td>MISSING option, 125</td>
</tr>
<tr>
<td>NLEVELS option, 125</td>
</tr>
<tr>
<td>NOPRINT option, 125</td>
</tr>
<tr>
<td>ORDER= option, 125</td>
</tr>
<tr>
<td>PAGE option, 126</td>
</tr>
<tr>
<td>FREQTAB procedure, TABLES statement, 147</td>
</tr>
<tr>
<td>ADJUST option (PLCORR), 170</td>
</tr>
<tr>
<td>AGREE option, 150</td>
</tr>
<tr>
<td>ALL option, 152</td>
</tr>
<tr>
<td>ALPHA= option, 152</td>
</tr>
<tr>
<td>BDT option (CMH), 160</td>
</tr>
<tr>
<td>BINOMIAL option, 153</td>
</tr>
<tr>
<td>CELLC option, 153</td>
</tr>
<tr>
<td>CHISQ option, 157</td>
</tr>
</tbody>
</table>
Syntax Index

CL option, 159
CL= option (BINOMIAL), 154
CL= option (COMMONRISKDIFF), 161
CL= option (RELRISK), 185
CL= option (RISKDIFF), 188
CL=AGRESTICAFFO option (RISKDIFF), 189
CL=AGRESTICOULL option (BINOMIAL), 154
CL=BLAKER option (BINOMIAL), 154
CL=CLOPPERPEARSON option (BINOMIAL), 154
CL=EXACT option (BINOMIAL), 154
CL=EXACT option (RISKDIFF), 189
CL=HA option (RISKDIFF), 189
CL=JEFFREYS option (BINOMIAL), 155
CL=K option (COMMONRISKDIFF), 162
CL=LIKELIHOODRATIO option (BINOMIAL), 155
CL=LOGIT option (BINOMIAL), 155
CL=MH option (COMMONRISKDIFF), 162
CL=MR option (COMMONRISKDIFF), 162
CL=NEWCOMBE option (COMMONRISKDIFF), 162
CL=NEWCOMBEMR option (COMMONRISKDIFF), 162
CL=SCORE option (COMMONRISKDIFF), 162
CL=WALD option (BINOMIAL), 155
CL=WALD option (RISKDIFF), 190
CL=WILSON option (BINOMIAL), 155
CMH option, 160
CMH1 option, 161
CMH2 option, 161
COLUMN= option (COMMONRISKDIFF), 162
COLUMN= option (RELRISK), 186
COLUMN= option (RISKDIFF), 190
COMMON option (RISKDIFF), 190
COMMONRISKDIFF option, 161
CONTENTS= option, 163
CONVERGE= option (PLCORR), 170
CORRECT option (BINOMIAL), 155
CORRECT option (RISKDIFF), 190
CORRECT=NO option (COMMONRISKDIFF), 162
CROSSLIST option, 164
CUMCOL option, 165
DEVIATION option, 165
DF= option (CHISQ), 158
EQUAL option (RELRISK), 186
EQUAL option (RISKDIFF), 190
EQUIVALENCE option (BINOMIAL), 155
EQUIVALENCE option (RELRISK), 186
EQUIVALENCE option (RISKDIFF), 190
EXACT option, 165
EXPECTED option, 165
FISHER option, 165
FORMAT= option, 165
GAILSIMON option, 166
I2 option (CMH), 160
JT option, 166
LEVEL= option (BINOMIAL), 156
LIST option, 166
LRCHI option (CHISQ), 158
MANTELFLEISS option (CMH), 160
MARGIN= option (BINOMIAL), 156
MARGIN= option (RELRISK), 186
MARGIN= option (RISKDIFF), 191
MAXITER= option (PLCORR), 170
MAXLEVELS= option, 166
MEASURES option, 166
METHOD= option (RELRISK), 186
METHOD= option (RISKDIFF), 191
METHOD=FM option (RELRISK), 186
METHOD=FM option (RISKDIFF), 191
METHOD=HA option (RISKDIFF), 191
METHOD=LR option (RELRISK), 187
METHOD=NEWCOMBE option (RISKDIFF), 191
METHOD=WALD option (RELRISK), 187
METHOD=WALD option (RISKDIFF), 191
METHOD=WALDMODIFIED option (RELRISK), 187
MISSPRINT option, 125
NOCOL option, 167
NOCUM option, 167
NOFREQ option, 167
NONINFERIORITY option (BINOMIAL), 156
NONINFERIORITY option (RELRISK), 187
NONINFERIORITY option (RISKDIFF), 192
NOPERCENT option, 167
NOPRINT option, 167
NORISKS option (RISKDIFF), 192
NOROW option, 167
NOSPARSE option, 167
NOWARN option, 168
OR option, 168
OUT= option, 169
OUTCUM option, 169
OUTEXPECT option, 169
OUTLEVEL option (BINOMIAL), 157
OUTPCT option, 169
P= option (BINOMIAL), 157
PEARSONRES option (CROSSLIST), 164
PLCORR option, 170
PLOTS= option, 170
PLOTS=AGREEPLOT option, 172
PLOTS=CUMFREQPLOT option, 172
PLOTS=DEVIATIONPLOT option, 173
Syntax Index

PLOTS=FREQPLOT option, 173
PLOTS=KAPPAPLOT option, 174
PLOTS=MOSAICPLOT option, 175
PLOTS=NONE option, 176
PLOTS=ODDSRATIOPILOT option, 176
PLOTS=RELRSKPLOT option, 176
PLOTS=RISKDIFFPLOT option, 177
PLOTS=WTKAPPAPLOT, 179
POLYCHORIC option, 170
PRINTALL option (RELRISK), 187
PRINTWTS option, 152, 184
PRINTWTS option (COMMONRISKDIFF), 162
QOR option (CMH), 161
RELRISK option, 184
RISKDIFF option, 187
SCORES= option, 192
SCOROUT option, 193
SENSPEC option, 193
STDRES option (CROSSLIST), 164
SUPERIORITY option (BINOMIAL), 157
SUPERIORITY option (RELRISK), 187
SUPERIORITY option (RISKDIFF), 192
TEST option (COMMONRISKDIFF), 163
TEST=MH option (COMMONRISKDIFF), 163
TEST=MR option (COMMONRISKDIFF), 163
TEST=SCORE option (COMMONRISKDIFF), 163
TESTF= option, 203
TESTF= option (CHISQ), 158
TESTP= option, 203
TESTP= option (CHISQ), 159
TOTPCT option, 193
TREND option, 193
VAR= option (BINOMIAL), 157
VAR= option (RISKDIFF), 192
WARN= option (CHISQ), 159
FREQTAB procedure, TEST statement, 193
AGREE option, 194
GAMMA option, 195
KAPPA option, 195
KENTB option, 195
MEASURES option, 195
PCORR option, 195
PLCORR option, 195
SCORR option, 196
SMDCR option, 196
SMDRC option, 196
STUTC option, 196
TAUB option, 195
TAUC option, 196
WTKAPPA option, 196
FREQTAB procedure, WEIGHT statement, 197
ZEROS option, 197
FTEST option

GROW statement (TREESPLIT), 1028
FTOL option
PROC ASSESS statement, 45
PROC BINNING statement, 45
PROC CARDINALITY statement, 45
PROC CORRELATION statement, 45
PROC GAMMOD statement, 45
PROC GENSELECT statement, 45
PROC ICA statement, 45
PROC KCLUS statement, 45
PROC LMIXED statement, 45
PROC LOGSELECT statement, 45
PROC MBC statement, 45
PROC NLMOD statement, 45
PROC PARTITION statement, 45
PROC PCA statement, 45
PROC PHSELECT statement, 45
PROC PLSMOD statement, 45
PROC QTRSELECT statement, 45
PROC REGSELECT statement, 45
PROC TREESPLIT statement, 45
PROC VARIMPUTE statement, 45
PROC VARREDUCE statement, 45
FTOL2 option
PROC ASSESS statement, 45
PROC BINNING statement, 45
PROC CARDINALITY statement, 45
PROC CORRELATION statement, 45
PROC GAMMOD statement, 45
PROC GENSELECT statement, 45
PROC ICA statement, 45
PROC KCLUS statement, 45
PROC LMIXED statement, 45
PROC LOGSELECT statement, 45
PROC MBC statement, 45
PROC NLMOD statement, 45
PROC PARTITION statement, 45
PROC PCA statement, 45
PROC PHSELECT statement, 45
PROC PLSMOD statement, 45
PROC QTRSELECT statement, 45
PROC REGSELECT statement, 45
PROC TREESPLIT statement, 45
PROC VARIMPUTE statement, 45
PROC VARREDUCE statement, 45
GAILSIMON option
OUTPUT statement (FREQTAB), 142
TABLES statement (FREQTAB), 166
GAMMA option
OUTPUT statement (FREQTAB), 142
TEST statement (FREQTAB), 195
GAMMOD procedure, 314
MODEL statement, 320
OUTPUT statement, 329
PROC GAMMOD statement, 314
syntax, 314
WEIGHT statement, 330
GAMMOD procedure, CLASS statement, 318
DESCENDING option, 13, 1173
MISSING option, 13, 1173
ORDER= option, 13
PARAM= option, 14
REF= option, 15
SPLIT option, 15
GAMMOD procedure, CODE statement
COMMENT option, 17
FILE= option, 17
FORMATWIDTH= option, 17
INDENTSIZE= option, 17
LABELID= option, 18
LINESIZE= option, 18
NOTRIM option, 18
OUT= option, 18
GAMMOD procedure, DISPLAY statement
CASESENSITIVE option, 20, 319
EXCLUDE option, 20, 319
EXCLUDEALL option, 20, 319
TRACE option, 20, 319
GAMMOD procedure, DISPLAYOUT statement
INCLUDEALL option, 21, 319
NOREPLACE option, 21, 320
REPEATED option, 21, 320
GAMMOD procedure, EFFECT statement
BASIS option (spline), 28
DATABOUNDARY option (spline), 28
DEGREE option (polynomial), 26
DEGREE option (spline), 29
DETAILS option (multimember), 25
DETAILS option (polynomial), 26
DETAILS option (spline), 29
KNOTMAX= option (spline), 29
KNOTMETHOD option (spline), 29
KNOTMIN= option (spline), 30
MDEGREE option (polynomial), 26
NATURALCUBIC option (spline), 30
NOEFFECT option (multimember), 25
NOSEPARATE option (polynomial), 26
SEPARATE option (spline), 30
SPLIT option (spline), 31
STANDARDIZE option (polynomial), 26
GAMMOD procedure, MODEL statement, 320
ALLOBS option, 325
CRITERION= option, 325
DESCENDING option, 322
DETAILS option, 323
DF= option, 324
DISPERSION= option, 326
DISTRIBUTION= option, 326
FDHESSIAN option, 327
INITIALPHI= option, 327
INITSMOOTH= option, 324
KNOTS= option, 324
LINK= option, 327
M= option, 325
MAXDF= option, 325
MAXKNOTS= option, 325
MAXPHI= option, 327
MAXSMOOTH= option, 325
METHOD= option, 328
MINPHI= option, 328
MINSMOOTH= option, 325
NORMALIZE option, 328
OFFSET= option, 328
ORDER= option, 323
RIDGE= option, 328
SCALE= option, 328
SMOOTH= option, 325
GAMMOD procedure, OUTPUT statement, 329
ALPHA= option, 329
COMPONENT option, 329
COPYV AR= option, 329
keyword= option, 329
OUT= option, 329
GAMMOD procedure, PROC GAMMOD statement, 314
ABSCONV option, 44
ABSFCNV option, 44
ABSFTOL option, 44
ABSGCONV option, 44
ABSGTOL option, 44
ABSTOL option, 44
ABSXCONV option, 45
ABSTOL option, 45
ALPHA= option, 315
DATA= option, 315
FCONV option, 45
FCONV2 option, 45
FTOL option, 45
FTOL2 option, 45
GCONV option, 46
GCONV2 option, 46
GTOL option, 46
GTOL2 option, 46
ITDETAILS option, 315
MAXFUNC= option, 47
MAXITER= option, 47
MAXTIME= option, 47
MINITER= option, 47
NOCLPRINT option, 315
NOPRINT option, 315
NORMALIZE= option, 48
PLIKEOPTIONS option, 315
plots(unpack) option, 316
PLOTS= option, 316
SEED= option, 316
SINGCHOL= option, 316
SINGULAR= option, 317
SMOOTHOPTIONS option, 317
TECHNIQUE= option, 48
XCONV option, 48
XTOL option, 48
GAMMOD procedure, SELECTION statement
ADAPTIVE option, 38
CHOOSE= option, 38
COMPETITIVE option, 38
CRITERION= option, 38
DETAILS= option, 40, 41
FAST option, 39
HIERARCHY= option, 41
LSCOEFFS option, 39
MAXEFFECTS= option, 39
MAXSTEPS= option, 39
METHOD= option, 37
MINEFFECTS= option, 39
ORDERSELECT option, 41
SELECT= option, 39
SELECTION= option, 40, 41
SLE= option, 39
SLENTRY= option, 39
SLS= option, 39
SLSTAY= option, 39
STOP= option, 40
STOPHORIZON= option, 43
GAMMOD procedure, WEIGHT statement, 330
GCONV option
PROC ASSESS statement, 46
PROC BINNING statement, 46
PROC CARDINALITY statement, 46
PROC CORRELATION statement, 46
PROC GAMMOD statement, 46
PROC GENSELECT statement, 46
PROC ICA statement, 46
PROC KCLUS statement, 46
PROC LMIXED statement, 46
PROC LOGSELECT statement, 46
PROC MBC statement, 46
PROC NLMODE statement, 46
PROC PARTITION statement, 46
PROC PCA statement, 46
PROC PHSELECT statement, 46
PROC PLSMOD statement, 46
PROC QTRSELECT statement, 46
PROC REGSELECT statement, 46
PROC TREESPLIT statement, 46
PROC VARIMPUTE statement, 46
PROC VARREDUCE statement, 46
GCONV2 option
PROC ASSESS statement, 46
PROC BINNING statement, 46
PROC CARDINALITY statement, 46
PROC CORRELATION statement, 46
PROC GAMMOD statement, 46
PROC GENSELECT statement, 46
PROC ICA statement, 46
PROC KCLUS statement, 46
PROC LMIXED statement, 46
PROC LOGSELECT statement, 46
PROC MBC statement, 46
PROC NLMODE statement, 46
PROC PARTITION statement, 46
PROC PCA statement, 46
PROC PHSELECT statement, 46
PROC PLSMOD statement, 46
PROC QTRSELECT statement, 46
PROC REGSELECT statement, 46
PROC TREESPLIT statement, 46
PROC VARIMPUTE statement, 46
PROC VARREDUCE statement, 46
GCONV2= option
OPTIMIZATION statement, 508
GCONV= option
OPTIMIZATION statement, 507
GENSELECT procedure, 372
MODEL statement, 380
PROC GENSELECT statement, 372
SELECTION statement, 392
syntax, 372
GENSELECT procedure, CLASS statement, 375
DESCENDING option, 13, 1173
MISSING option, 13, 1173
ORDER= option, 13
PARAM= option, 14
REF= option, 15
SPLIT option, 15
GENSELECT procedure, CODE statement, 376
COMMENT option, 17
FILE= option, 17
FORMATWIDTH= option, 17
INDENTSIZE= option, 17
IPROB option, 18
LABELID= option, 18
LINESIZE= option, 18
NOTRIM option, 18
OUT= option, 18
PCATALL option, 18
GENSELECT procedure, DISPLAY statement
CASESENSITIVE option, 20, 377
EXCLUDE option, 20, 377
EXCLUDEALL option, 20, 377
TRACE option, 20, 377
GENSELECT procedure, DISPLAYOUT statement
   INCLUDEALL option, 21, 378
   NOPREPLACE option, 21, 378
   REPEATED option, 21, 378
GENSELECT procedure, EFFECT statement
   BASIS option (spline), 28
   DATABOUNDARY option (spline), 28
   DEGREE option (polynomial), 26
   DEGREE option (spline), 29
   DETAILS option (multimember), 25
   DETAILS option (polynomial), 26
   DETAILS option (spline), 29
   KNOTMAX= option (spline), 29
   KNOTMETHOD option (spline), 29
   KNOTMIN= option (spline), 30
   MDEGREE option (polynomial), 26
   NATURALCUBIC option (spline), 30
   NOEFFECT option (multimember), 25
   NOSEPARATE option (polynomial), 26
   SEPARATE option (spline), 30
   SPLIT option (spline), 31
   STANDARDIZE option (polynomial), 26
GENSELECT procedure, MODEL statement, 380
   CENTER option, 382
   CENTERLASSO option, 382
   CLB option, 382
   DESCENDING option, 381
   DISTRIBUTION= option, 382
   INCLUDE option, 384
   INFORMATIVE option, 384
   LINK= option, 384
   NOCHECK option, 374
   NOINT option, 386
   OFFSET= option, 386
   ORDER= option, 381
   PHI= option, 386
   START option, 386
   TYPE3 option, 386
GENSELECT procedure, OUTPUT statement, 386
   ALL option, 387
   ALPHA= option, 387
   CBAR= option, 388
   COPYV AR= option, 387
   DIFCHISQ= option, 388
   DIFDEV= option, 388
   H= option, 388
   INDIVIDUAL= option, 389
   INTO= option, 389
   IPRED= option, 389
   keyword= option, 387
   LCL= option, 389
   LCLM= option, 389
   UCLM= option, 389
   LKL= option, 389
   LKL M= option, 389
   LKL = option, 389
   LCL M= option, 389
   LEVEL= option, 389
   LINP= option, 391
   LOWER= option, 389
   LOWERXBETA= option, 389
   OBSCAT option, 387
   OUT= option, 387
   PREDICTED= option, 389
   RESCHI= option, 389
   RESDEV= option, 390
   RESIDUAL= option, 390
   RESLIK= option, 390
   RESRAW= option, 390
   RESWORK= option, 390
   ROLE= option, 390
   STDRESCHI= option, 390
   STDRESDE V= option, 390
   STDERR= option, 390
   UCL= option, 391
   UCLM= option, 391
   UPPER= option, 391
   UPPERXBETA= option, 391
   XBETA= option, 391
GENSELECT procedure, PARTITION statement
   FRACTION option, 36, 391
   ROLEVAR= option, 36, 391
GENSELECT procedure, PROC GENSELECT statement, 372
   ABSCONV option, 44
   ABSFCONV option, 44
   ABSFTOL option, 44
   ABSGCONV option, 44
   ABSGTOL option, 44
   ABSTOL option, 44
   ABSTOL option, 44
   ABSTOL option, 44
   ABSXCONV option, 45
   ABSXTOL option, 45
   ALPHAS option, 373
   CORRB option, 373
   COVB option, 373
   DATA= option, 373
   FCONV option, 45
   FCONV2 option, 45
   FTOL option, 45
   FTOL2 option, 45
   GCONV option, 46
   GCONV2 option, 46
   GTOL option, 46
   GTOL2 option, 46
   ITHIST option, 374
   LASSORHO= option, 374
   LASSOSTEPS= option, 374
   LASSOTOL= option, 374
   MAXFUNC= option, 47
   MAXITER= option, 47
   MAXOPTBATCH= option, 374
   MAXTIME= option, 47
MINITER= option, 47
NOCLPRINT option, 374
NORMALIZE= option, 48
NOSTDERR option, 374
PAGEOBS= option, 374
PARTFIT option, 375
STB option, 375
TECHNIQUE= option, 48
XCONV option, 48
XTOL option, 48

GENSELECT procedure, SELECTION statement, 392
ADAPTIVE option, 38
CHOOSE= option, 38
COMPETITIVE option, 38
CRITERION= option, 38
DETAILS= option, 40, 41
FAST option, 39
HIERARCHY= option, 41
LSCOEFFS option, 39
MAXEFFECTS= option, 39
MAXSTEPS= option, 39
METHOD= option, 37
MINEFFECTS= option, 39
ORDERSELECT option, 41
SELECT= option, 39
SELECTION= option, 43
SLE= option, 39
SLENTRY= option, 39
SLS= option, 39
SLSTAY= option, 39
STOP= option, 40
STOPHORIZON= option, 43

GFUNC= option
PROC ICA statement, 436
GFUNCTION= option
PROC ICA statement, 436

GINI option
GROW statement (TREESPLIT), 1028

GROUP= option
PROC ASSESS statement, 46
PROC BINNING statement, 46
PROC CARDINALITY statement, 46
PROC CORRELATION statement, 46
PROC GAMMOD statement, 46
PROC GENSELECT statement, 46
PROC ICA statement, 46
PROC KCLUS statement, 46
PROC LMIXED statement, 46
PROC LOGLINEAR statement, 46
PROC MBC statement, 46
PROC NLMOD statement, 46
PROC PARTITION statement, 46
PROC PCA statement, 46
PROC PHSELECT statement, 46
PROC PLISMOD statement, 46
PROC QTRSELECT statement, 46
PROC REGSELECT statement, 46
PROC TREESPLIT statement, 46
PROC VARIMPUTE statement, 46
PROC VARREDUCE statement, 46

GTOL2 option
PROC ASSESS statement, 46
PROC BINNING statement, 46
PROC CARDINALITY statement, 46
PROC CORRELATION statement, 46
PROC GAMMOD statement, 46
PROC GENSELECT statement, 46
PROC ICA statement, 46
PROC KCLUS statement, 46
PROC LMIXED statement, 46
PROC LOGLINEAR statement, 46
PROC MBC statement, 46
PROC NLMOD statement, 46
PROC PARTITION statement, 46
PROC PCA statement, 46
PROC PHSELECT statement, 46
PROC PLISMOD statement, 46
PROC QTRSELECT statement, 46
PROC REGSELECT statement, 46
PROC TREESPLIT statement, 46
PROC VARIMPUTE statement, 46
PROC VARREDUCE statement, 46

H= option
PROC ASSESS statement, 46
PROC BINNING statement, 46
PROC CARDINALITY statement, 46
PROC CORRELATION statement, 46
PROC GAMMOD statement, 46
PROC GENSELECT statement, 46
PROC ICA statement, 46
PROC KCLUS statement, 46
PROC LMIXED statement, 46
PROC LOGLINEAR statement, 46
PROC MBC statement, 46
PROC NLMOD statement, 46
PROC PARTITION statement, 46
PROC PCA statement, 46
PROC PHSELECT statement, 46
PROC PLISMOD statement, 46
PROC QTRSELECT statement, 46
PROC REGSELECT statement, 46
PROC TREESPLIT statement, 46
PROC VARIMPUTE statement, 46
PROC VARREDUCE statement, 46

HESSIAN option
PROC PHSELECT statement, 753

HIERARCHY= option
PROC ASSESS statement, 46
PROC BINNING statement, 46
PROC CARDINALITY statement, 46
PROC CORRELATION statement, 46
PROC GAMMOD statement, 46
PROC GENSELECT statement, 46
PROC ICA statement, 46
PROC KCLUS statement, 46
PROC LMIXED statement, 46
PROC LOGLINEAR statement, 46
PROC MBC statement, 46
PROC NLMOD statement, 46
PROC PARTITION statement, 46
PROC PCA statement, 46
PROC PHSELECT statement, 46
PROC PLISMOD statement, 46
PROC QTRSELECT statement, 46
PROC REGSELECT statement, 46
PROC TREESPLIT statement, 46
PROC VARIMPUTE statement, 46
PROC VARREDUCE statement, 46

GTOL option
PROC ASSESS statement, 46
PROC BINNING statement, 46
PROC CARDINALITY statement, 46
PROC CORRELATION statement, 46
PROC GAMMOD statement, 46
PROC GENSELECT statement, 46
PROC ICA statement, 46
PROC KCLUS statement, 46
PROC LMIXED statement, 46
PROC LOGLINEAR statement, 46
PROC MBC statement, 46
PROC NLMOD statement, 46
PROC PARTITION statement, 46
PROC PCA statement, 46
PROC PHSELECT statement, 46
PROC PLISMOD statement, 46
PROC QTRSELECT statement, 46
PROC REGSELECT statement, 46
PROC TREESPLIT statement, 46
PROC VARIMPUTE statement, 46
PROC VARREDUCE statement, 46

GTOL2 option
PROC ASSESS statement, 46
PROC BINNING statement, 46
PROC CARDINALITY statement, 46
PROC CORRELATION statement, 46
PROC GAMMOD statement, 46
PROC GENSELECT statement, 46
PROC ICA statement, 46
PROC KCLUS statement, 46
PROC LMIXED statement, 46
PROC LOGLINEAR statement, 46
PROC MBC statement, 46
PROC NLMOD statement, 46
PROC PARTITION statement, 46
PROC PCA statement, 46
PROC PHSELECT statement, 46
PROC PLISMOD statement, 46
PROC QTRSELECT statement, 46
PROC REGSELECT statement, 46
PROC TREESPLIT statement, 46
PROC VARIMPUTE statement, 46
PROC VARREDUCE statement, 46
SELECTION statement (PLSMOD), 41
SELECTION statement (QTRSELECT), 41
SELECTION statement (REGSELECT), 41
SELECTION statement (TREE_SPLIT), 41
SELECTION statement (VARIMPUTE), 41
SELECTION statement (VARIABLE), 41
hold= option
PARMS statement (LMixed), 512
I2 option (CMH)
TABLES statement (FREQTAB), 160
ICA procedure
PROC ICA statement, 435
syntax, 435
ICA procedure, CLASS statement
DESCENDING option, 13, 1173
MISSING option, 13, 1173
ORDER= option, 13
PARAM= option, 14
REF= option, 15
SPLIT option, 15
ICA procedure, CODE statement
COMMENT option, 17
FILE= option, 17
FORMATWIDTH= option, 17
INDENTSIZE= option, 17
LABELID= option, 18
LINESIZE= option, 18
NOTRIM option, 18
OUT= option, 18
ICA procedure, DISPLAY statement
CASESENSITIVE option, 20, 438
EXCLUDE option, 20, 439
EXCLUDEALL option, 20, 439
TRACE option, 20, 439
ICA procedure, DISPLAYOUT statement
INCLUDEALL option, 21, 439
NOREPLACE option, 21, 439
REPEATED option, 21, 439
ICA procedure, EFFECT statement
BASIS option (spline), 28
DATABASEBOUNDARY option (spline), 28
DEGREE option (polynomial), 26
DEGREE option (spline), 29
DETAILS option (multimember), 25
DETAILS option (polynomial), 26
DETAILS option (spline), 29
KNOTMAX= option (spline), 29
KNOTMETHOD option (spline), 29
KNOTMIN= option (spline), 30
MDEGREE option (polynomial), 26
NORMALBCUBIC option (spline), 30
NOEFFECT option (multimember), 25
NOSEPARATE option (polynomial), 26
SEPARATE option (spline), 30
SPLIT option (spline), 31
STANDARDIZE option (polynomial), 26
ICA procedure, OUTPUT statement, 439
keyword option, 440
COPYVARS= option, 440
OUT= option, 440
ICA procedure, PROC ICA statement, 435
ABSCONV option, 44
ABSFCONV option, 44
ABSFTOL option, 44
ABSGCONV option, 44
ABSGTOL option, 44
ABSTOL option, 44
ABSXCONV option, 45
ABSXTOL option, 45
DATA= option, 436
EIGENTHRESH= option, 436
EIGTHRESH= option, 436
FCONV option, 45
FCONV2 option, 45
FTOL option, 45
FTOL2 option, 45
GCONV option, 46
GCONV2 option, 46
GFUNC= option, 436
GFUNCT= option, 436
GTOL option, 46
GTOL2 option, 46
MAXFUNC= option, 47
MAXITER= option, 47
MAXTIME= option, 47
METHOD= option, 436
MINITER= option, 47
N= option, 437
NOCENTER option, 437
NORMALIZE= option, 48
NOSCALE option, 437
PREFIX= option, 437
SEED= option, 437
TECHNIQUE= option, 48
WHITEPREFIX= option, 437
WHITEPREFIX= option, 437
XCONV option, 48
XTOL option, 48
ICA procedure, PROC ICA statement,
METHOD=DEFIATION option
MAXITER= option, 436
TOL= option, 436
TOLERANCE= option, 436
ICA procedure, PROC ICA statement,
METHOD=SYMMEIC option
MAXITER= option, 437
TOL= option, 437
<table>
<thead>
<tr>
<th>Option</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>TOLERANCE= option</td>
<td>437</td>
</tr>
<tr>
<td>ICA procedure, SELECTION statement</td>
<td></td>
</tr>
<tr>
<td>ADAPTIVE option</td>
<td>38</td>
</tr>
<tr>
<td>CHOOSE= option</td>
<td>38</td>
</tr>
<tr>
<td>COMPETITIVE option</td>
<td>38</td>
</tr>
<tr>
<td>CRITERION= option</td>
<td>38</td>
</tr>
<tr>
<td>DETAILS= option</td>
<td>40, 41</td>
</tr>
<tr>
<td>FAST option</td>
<td>39</td>
</tr>
<tr>
<td>HIERARCHY= option</td>
<td>41</td>
</tr>
<tr>
<td>LSCEOFFS option</td>
<td>39</td>
</tr>
<tr>
<td>MAXEFFECTS= option</td>
<td>39</td>
</tr>
<tr>
<td>MAXSTEPS= option</td>
<td>39</td>
</tr>
<tr>
<td>METHOD= option</td>
<td>37</td>
</tr>
<tr>
<td>MINEFFECTS= option</td>
<td>39</td>
</tr>
<tr>
<td>ORDERSELECT option</td>
<td>41</td>
</tr>
<tr>
<td>SELECT= option</td>
<td>39</td>
</tr>
<tr>
<td>SELECTION= option</td>
<td>43</td>
</tr>
<tr>
<td>SLE= option</td>
<td>39</td>
</tr>
<tr>
<td>SLETRY= option</td>
<td>39</td>
</tr>
<tr>
<td>SLS= option</td>
<td>39</td>
</tr>
<tr>
<td>SLSTAY= option</td>
<td>39</td>
</tr>
<tr>
<td>STOP= option</td>
<td>40</td>
</tr>
<tr>
<td>STOPHORIZON= option</td>
<td>43</td>
</tr>
<tr>
<td>ICA procedure, VAR statement</td>
<td>441</td>
</tr>
<tr>
<td>ID statement</td>
<td></td>
</tr>
<tr>
<td>NLMOD procedure</td>
<td>685</td>
</tr>
<tr>
<td>IGR option</td>
<td></td>
</tr>
<tr>
<td>GROW statement (TREESPLIT)</td>
<td>1028</td>
</tr>
<tr>
<td>IMPUTE= option</td>
<td>459</td>
</tr>
<tr>
<td>IMPUTENOM= option</td>
<td>459</td>
</tr>
<tr>
<td>INCLUDE option</td>
<td></td>
</tr>
<tr>
<td>MODEL statement (GENSELECT)</td>
<td>384</td>
</tr>
<tr>
<td>MODEL statement (LOGSELECT)</td>
<td>570</td>
</tr>
<tr>
<td>MODEL statement (QTRSELECT)</td>
<td>846</td>
</tr>
<tr>
<td>MODEL statement (REGSELECT)</td>
<td>900</td>
</tr>
<tr>
<td>INCLUDE= option</td>
<td></td>
</tr>
<tr>
<td>MODEL statement (PHSELECT)</td>
<td>761</td>
</tr>
<tr>
<td>INCLUDEALL option</td>
<td></td>
</tr>
<tr>
<td>DISPLAYOUT statement (CORRELATION)</td>
<td>21, 98</td>
</tr>
<tr>
<td>DISPLAYOUT statement (GAMMOD)</td>
<td>21, 319</td>
</tr>
<tr>
<td>DISPLAYOUT statement (GENSELECT)</td>
<td>21, 378</td>
</tr>
<tr>
<td>DISPLAYOUT statement (ICA)</td>
<td>21, 439</td>
</tr>
<tr>
<td>DISPLAYOUT statement (KCLUS)</td>
<td>21, 464</td>
</tr>
<tr>
<td>DISPLAYOUT statement (LMIXED)</td>
<td>21, 503</td>
</tr>
<tr>
<td>DISPLAYOUT statement (LOGSELECT)</td>
<td>21, 565</td>
</tr>
<tr>
<td>DISPLAYOUT statement (MBC)</td>
<td>21, 631</td>
</tr>
<tr>
<td>DISPLAYOUT statement (MODELMATRIX)</td>
<td>660</td>
</tr>
<tr>
<td>DISPLAYOUT statement (NLMOD)</td>
<td>21, 684</td>
</tr>
<tr>
<td>DISPLAYOUT statement (PARTITION)</td>
<td>21, 1143</td>
</tr>
<tr>
<td>DISPLAYOUT statement (PCA)</td>
<td>21, 720</td>
</tr>
<tr>
<td>DISPLAYOUT statement (PHSELECT)</td>
<td>21, 758</td>
</tr>
<tr>
<td>DISPLAYOUT statement (PLSMOD)</td>
<td>21, 805</td>
</tr>
<tr>
<td>DISPLAYOUT statement (QTRSELECT)</td>
<td>21, 843</td>
</tr>
<tr>
<td>DISPLAYOUT statement (REGSELECT)</td>
<td>21, 897</td>
</tr>
<tr>
<td>DISPLAYOUT statement (VARREDUCE)</td>
<td>21, 1175</td>
</tr>
<tr>
<td>INDENTSIZE= option</td>
<td></td>
</tr>
<tr>
<td>CODE statement (ASSESS)</td>
<td>17</td>
</tr>
<tr>
<td>CODE statement (BINNING)</td>
<td>17</td>
</tr>
<tr>
<td>CODE statement (CARDINALITY)</td>
<td>17</td>
</tr>
<tr>
<td>CODE statement (CORRELATION)</td>
<td>17</td>
</tr>
<tr>
<td>CODE statement (GAMMOD)</td>
<td>17</td>
</tr>
<tr>
<td>CODE statement (GENSELECT)</td>
<td>17</td>
</tr>
<tr>
<td>CODE statement (ICA)</td>
<td>17</td>
</tr>
<tr>
<td>CODE statement (KCLUS)</td>
<td>17</td>
</tr>
<tr>
<td>CODE statement (LMIXED)</td>
<td>17</td>
</tr>
<tr>
<td>CODE statement (LOGSELECT)</td>
<td>17</td>
</tr>
<tr>
<td>CODE statement (MBC)</td>
<td>17</td>
</tr>
<tr>
<td>CODE statement (NLMOD)</td>
<td>17</td>
</tr>
<tr>
<td>CODE statement (PARTITION)</td>
<td>17</td>
</tr>
<tr>
<td>CODE statement (PCA)</td>
<td>17</td>
</tr>
<tr>
<td>CODE statement (PHSELECT)</td>
<td>17</td>
</tr>
<tr>
<td>CODE statement (PLSMOD)</td>
<td>17</td>
</tr>
<tr>
<td>CODE statement (QTRSELECT)</td>
<td>17</td>
</tr>
<tr>
<td>CODE statement (REGSELECT)</td>
<td>17</td>
</tr>
<tr>
<td>CODE statement (TREESPLIT)</td>
<td>17</td>
</tr>
<tr>
<td>CODE statement (VARIMPUTE)</td>
<td>17</td>
</tr>
<tr>
<td>CODE statement (VARREDUCE)</td>
<td>17</td>
</tr>
<tr>
<td>INDIVIDUAL= option</td>
<td></td>
</tr>
<tr>
<td>OUTPUT statement (GENSELECT)</td>
<td>389</td>
</tr>
<tr>
<td>OUTPUT statement (LOGSELECT)</td>
<td>575</td>
</tr>
<tr>
<td>INFORMATIVE option</td>
<td></td>
</tr>
<tr>
<td>MODEL statement (GENSELECT)</td>
<td>384</td>
</tr>
<tr>
<td>MODEL statement (LOGSELECT)</td>
<td>570</td>
</tr>
<tr>
<td>MODEL statement (MODELMATRIX)</td>
<td>663</td>
</tr>
<tr>
<td>MODEL statement (PHSELECT)</td>
<td>761</td>
</tr>
<tr>
<td>MODEL statement (QTRSELECT)</td>
<td>846</td>
</tr>
<tr>
<td>MODEL statement (REGSELECT)</td>
<td>900</td>
</tr>
<tr>
<td>INIT= option</td>
<td></td>
</tr>
<tr>
<td>PROC KCLUS statement</td>
<td>459</td>
</tr>
<tr>
<td>PROC MBC statement</td>
<td>627</td>
</tr>
<tr>
<td>INITIALPHI= option</td>
<td></td>
</tr>
<tr>
<td>MODEL statement (GAMMOD)</td>
<td>327</td>
</tr>
<tr>
<td>INITSMOOTH= option</td>
<td></td>
</tr>
<tr>
<td>MODEL statement (GAMMOD)</td>
<td>324</td>
</tr>
<tr>
<td>INMODEL= option</td>
<td></td>
</tr>
<tr>
<td>PROC TREESPLIT statement</td>
<td>1014</td>
</tr>
<tr>
<td>INPUT statement</td>
<td></td>
</tr>
<tr>
<td>ASSESS procedure</td>
<td>1080</td>
</tr>
</tbody>
</table>
BINNING procedure, 1102
KCLUS procedure, 465
TREESPLIT procedure, 1056
VARIMPUTE procedure, 1157

INTERCEPT option
MODEL statement (PLSMD), 807

INTO= option
OUTPUT statement (GENSELECT), 389
OUTPUT statement (LOGSELECT), 575

IPRED= option
OUTPUT statement (GENSELECT), 389
OUTPUT statement (LOGSELECT), 575

IPROB option
CODE statement (GENSELECT), 18
CODE statement (LOGSELECT), 18

IRCHART statement
IRCHART procedure, 941

ITDETAILS option
PROC GAMMOD statement, 315
PROC LMIXED statement, 498

ITHIST option
PROC GENSELECT statement, 374
PROC LOGSELECT statement, 561
PROC PHSELECT statement, 753
PROC MBC statement, 628

JT option
EXACT statement (FREQTAB), 130
OUTPUT statement (FREQTAB), 142
TABLES statement (FREQTAB), 166

KAPPA option
EXACT statement (FREQTAB), 130
OUTPUT statement (FREQTAB), 142
TEST statement (FREQTAB), 195

KCLUS procedure, CLASS statement
DESCENDING option, 13, 1173
MISSING option, 13, 1173
ORDER= option, 13
PARAM= option, 14
REF= option, 15
SPLIT option, 15

KCLUS procedure, CODE statement, 463
COMMENT option, 17
FILE= option, 17, 463
FORMATWIDTH= option, 17
INDENTSIZE= option, 17
LABELID= option, 18, 463
LINESIZE= option, 18
NOTRIM option, 18
OUT= option, 18

KCLUS procedure, DISPLAY statement
CASESENSITIVE option, 20, 464
EXCLUDE option, 20, 464

EXCLUDEALL option, 20, 464
TRACE option, 20, 464

KCLUS procedure, DISPLAYOUT statement
INCLUDEALL option, 21, 464
NOREPLACE option, 21, 464
REPEATED option, 21, 464

KCLUS procedure, EFFECT statement
BASIS option (spline), 28
DISTANCEBOUNDARY option (spline), 28

KCLUS procedure, DISPLAY statement
BASIS option (polynomial), 26
KNOTMAX= option (spline), 29
KNOTMIN= option (spline), 29
MDEGREE option (polynomial), 26
NOEFFECT option (multimember), 25
SEPARATE option (spline), 30
SPLIT option (spline), 31
STANDARDIZE option (polynomial), 26

KCLUS procedure, INPUT statement, 465
LEVEL= option, 465

KCLUS procedure, OUTPUT statement
COPYVARS= option, 466
OUT= option, 466

KCLUS procedure, PROC KCLUS statement, 457
ABSCONV option, 44
ABSFCONV option, 44
ABSFTOL option, 44
ABSGCONV option, 44
ABSGTOL option, 44
ABSTOL option, 44
ABSTOL option, 44
ABSSXCONV option, 45
ABSTOL option, 45
DATA= option, 458
DISTANCE= option, 458
DISTANCENOM= option, 459
FCONV option, 45
FCONV2 option, 45
FTOL option, 45
FTOL option, 45
GCONV option, 46
GCONV2 option, 46
GTOL option, 46
GTOL option, 46
IMPUTE= option, 459
IMPUTE= option, 459
INIT= option, 459
KPRODUCTTYPEPARAMS= option, 462
MAXCLUSTERS= option, 462
MAXFUNC= option, 47
MAXITER= option, 47, 460
MAXTIME= option, 47
MINITER= option, 47
NORMALIZE= option, 48
NTHREADS, 461
OUTSTAT= option, 461
PRINTALLDISTANCES option, 462
SEED= option, 461
STANDARDIZE= option, 461
STOPCRITERION= option, 460, 462
TECHNIQUE= option, 48
XCONV option, 48
XTOL option, 48
KCLUS procedure, SCORE statement, 465
KCLUS procedure, SELECTION statement
  ADAPTIVE option, 38
  CHOOSE= option, 38
  COMPETITIVE option, 38
  CRITERION= option, 38
  DETAILS= option, 40, 41
  FAST option, 39
  HIERARCHY= option, 41
  LS COEFFS option, 39
  MAX EFFECTS= option, 39
  MAX STEPS= option, 39
  METHOD= option, 37
  MIN EFFECTS= option, 39
  ORDERSELECT option, 41
  SELECT= option, 39
  SELECTION= option, 43
  SLE= option, 39
  SLENTRY= option, 39
  SLS= option, 39
  SLSTAY= option, 39
  STOP= option, 40
  STOPHORIZON= option, 43
KCLUS procedure, syntax, 457
KENTB option
  EXACT statement (FREQTAB), 130
  OUTPUT statement (FREQTAB), 142
  TEST statement (FREQTAB), 195
keyword= option
  OUTPUT statement (PLSMOD), 808
keyword= option
  OUTPUT statement (GAMMOD), 329
  OUTPUT statement (GENSELECT), 387
  OUTPUT statement (LMIXED), 510
  OUTPUT statement (QTRSELECT), 848
  OUTPUT statement (REGSELECT), 901
KFOLD= option
  AUTOTUNE statement, 1020
  PROC PARTITION statement, 1141
  PRUNE statement (TREESPLIT), 1031
KNOTMAX= option
  EFFECT statement, spline (ASSESS), 29
  EFFECT statement, spline (BINNING), 29
  EFFECT statement, spline (CARDINALITY), 29
  EFFECT statement, spline (CORRELATION), 29
  EFFECT statement, spline (GAMMOD), 29
  EFFECT statement, spline (GENSELECT), 29
  EFFECT statement, spline (ICA), 29
  EFFECT statement, spline (KCLUS), 29
  EFFECT statement, spline (LMIXED), 29
  EFFECT statement, spline (LOGSELECT), 29
  EFFECT statement, spline (MBC), 29
  EFFECT statement, spline (NLMOD), 29
  EFFECT statement, spline (PARTITION), 29
  EFFECT statement, spline (PCA), 29
  EFFECT statement, spline (PHSELECT), 29
  EFFECT statement, spline (PLSMOD), 29
  EFFECT statement, spline (QTRSELECT), 29
  EFFECT statement, spline (REGSELECT), 29
  EFFECT statement, spline (TREESPLIT), 29
  EFFECT statement, spline (VARIMPUTE), 29
  EFFECT statement, spline (VARREDUCE), 29
KNOTMETHOD option
  EFFECT statement, spline (ASSESS), 29
  EFFECT statement, spline (BINNING), 29
  EFFECT statement, spline (CARDINALITY), 29
  EFFECT statement, spline (CORRELATION), 29
  EFFECT statement, spline (GAMMOD), 29
  EFFECT statement, spline (GENSELECT), 29
  EFFECT statement, spline (ICA), 29
  EFFECT statement, spline (KCLUS), 29
  EFFECT statement, spline (LMIXED), 29
  EFFECT statement, spline (LOGSELECT), 29
  EFFECT statement, spline (MBC), 29
  EFFECT statement, spline (NLMOD), 29
  EFFECT statement, spline (PARTITION), 29
  EFFECT statement, spline (PCA), 29
  EFFECT statement, spline (PHSELECT), 29
  EFFECT statement, spline (PLSMOD), 29
  EFFECT statement, spline (QTRSELECT), 29
  EFFECT statement, spline (REGSELECT), 29
  EFFECT statement, spline (TREESPLIT), 29
  EFFECT statement, spline (VARIMPUTE), 29
  EFFECT statement, spline (VARREDUCE), 29
KNOTMIN= option
  EFFECT statement, spline (ASSESS), 30
  EFFECT statement, spline (BINNING), 30
  EFFECT statement, spline (CARDINALITY), 30
  EFFECT statement, spline (CORRELATION), 30
  EFFECT statement, spline (GAMMOD), 30
  EFFECT statement, spline (GENSELECT), 30
  EFFECT statement, spline (ICA), 30
  EFFECT statement, spline (KCLUS), 30
  EFFECT statement, spline (LMIXED), 30
EFFECT statement, spline (LOGSELECT), 30
EFFECT statement, spline (MBC), 30
EFFECT statement, spline (NLMOD), 30
EFFECT statement, spline (PARTITION), 30
EFFECT statement, spline (PCA), 30
EFFECT statement, spline (PHSELECT), 30
EFFECT statement, spline (PLSMOD), 30
EFFECT statement, spline (QTRSELECT), 30
EFFECT statement, spline (REGSELECT), 30
EFFECT statement, spline (TREESPLIT), 30
EFFECT statement, spline (VARIMPUTE), 30
EFFECT statement, spline (VARREDUCE), 30
KNOTS= option
MODEL statement (GAMMOD), 324
KPROTOTYPEPARAMS= option
PROC KCLUS statement, 462
LABELID= option
CODE statement, 463
CODE statement (ASSESS), 18
CODE statement (BINNING), 18
CODE statement (CARDINALITY), 18
CODE statement (CORRELATION), 18
CODE statement (GAMMOD), 18
CODE statement (GENSELECT), 18
CODE statement (ICA), 18
CODE statement (KCLUS), 18
CODE statement (LMIXED), 18
CODE statement (LOGSELECT), 18
CODE statement (MBC), 18
CODE statement (NLMOD), 18
CODE statement (PARTITION), 18
CODE statement (PCA), 18
CODE statement (PHSELECT), 18
CODE statement (PLSMOD), 18
CODE statement (QTRSELECT), 18
CODE statement (REGSELECT), 18
CODE statement (TREESPLIT), 18
CODE statement (VARIMPUTE), 18
CODE statement (VARREDUCE), 18
LAMCR option
OUTPUT statement (FREQTAB), 142
LAMDAS option
OUTPUT statement (FREQTAB), 142
LAMRC option
OUTPUT statement (FREQTAB), 142
LASSORHO= option
PROC GENSELECT statement, 374
PROC LOGSELECT statement, 561
PROC PHSELECT statement, 753
LASSOSTEPS= option
PROC GENSELECT statement, 374
PROC LOGSELECT statement, 561
PROC PHSELECT statement, 753
LASSOTOL= option
PROC GENSELECT statement, 374
PROC LOGSELECT statement, 562
PROC PHSELECT statement, 753
LCL= option
OUTPUT statement (GENSELECT), 389
OUTPUT statement (LOGSELECT), 575
LCLM= option
OUTPUT statement (GENSELECT), 389
OUTPUT statement (LOGSELECT), 575
LEAVES= option
PRUNE statement (TREESPLIT), 1031
LEVEL= option
INPUT statement, 465, 1103
OUTPUT statement (GENSELECT), 389
OUTPUT statement (LOGSELECT), 575
TARGET statement, 1080, 1104
LEVEL= option (BINOMIAL)
TABLES statement (FREQTAB), 156
LGOR option
OUTPUT statement (FREQTAB), 143
LGRRC1 option
OUTPUT statement (FREQTAB), 143
LGRRC2 option
OUTPUT statement (FREQTAB), 143
LIFTOUT= option
PROC ASSESS statement, 1078
LIMIT= option
VIICODE statement (TREESPLIT), 1032
LIMITN= option
chart statements (SPC), 950
LINESIZE= option
CODE statement (ASSESS), 18
CODE statement (BINNING), 18
CODE statement (CARDINALITY), 18
CODE statement (CORRELATION), 18
CODE statement (GAMMOD), 18
CODE statement (GENSELECT), 18
CODE statement (ICA), 18
CODE statement (KCLUS), 18
CODE statement (LMIXED), 18
CODE statement (LOGSELECT), 18
CODE statement (MBC), 18
CODE statement (NLMOD), 18
CODE statement (PARTITION), 18
CODE statement (PCA), 18
CODE statement (PHSELECT), 18
CODE statement (PLSMOD), 18
CODE statement (QTRSELECT), 18
CODE statement (REGSELECT), 18
CODE statement (TREESPLIT), 18
CODE statement (VARIMPUTE), 18
CODE statement (VARREDUCE), 18
LINK= option
MODEL statement (GAMMOD), 327
MODEL statement (GENSELECT), 384
MODEL statement (LOGSELECT), 570
LINP= option
OUTPUT statement (GENSELECT), 391
OUTPUT statement (LOGSELECT), 577
LIST option
TABLES statement (FREQTAB), 166
LMIXED procedure, 495
ALLSTATS option, 510
OPTIMIZATION statement, 506
OUTPUT statement, 509
PROC LMIXED statement, 496
REPEATED statement, 523
syntax, 495
LMIXED procedure, BLUP statement, 499
LMIXED procedure, CLASS statement, 501, 534
DESCENDING option, 13, 1173
MISSING option, 13, 1173
ORDER= option, 13
PARAM= option, 14
REF= option, 15
SPLIT option, 15
LMIXED procedure, CODE statement
COMMENT option, 17
FILE= option, 17
FORMATWIDTH= option, 17
LABELID= option, 18
LINESIZE= option, 18
NOTRIM option, 18
OUT= option, 18
LMIXED procedure, DISPLAY statement
CASESENSITIVE option, 20, 502
EXCLUDE option, 20, 502
EXCLUDEALL option, 20, 502
TRACE option, 20, 502
LMIXED procedure, DISPLAYOUT statement
INCLUDEALL option, 21, 503
NOREPLACE option, 21, 503
REPEATED option, 21, 503
LMIXED procedure, EFFECT statement
BASIS option (spline), 28
DATABOUNDARY option (spline), 28
DEGREE option (polynomial), 26
DEGREE option (spline), 29
details option (multimember), 25
details option (polynomial), 26
details option (spline), 29
KNOTMAX= option (spline), 29
KNOTMETHOD option (spline), 29
KNOTMIN= option (spline), 30
MDEGREE option (polynomial), 26
NATURALCUBIC option (spline), 30
NOEFFECT option (multimember), 25
NOSEPARATE option (polynomial), 26
SEPARATE option (spline), 30
SPLIT option (spline), 31
STANDARDIZE option (polynomial), 26
LMIXED procedure, MODEL statement, 504
CL option, 505
DDFM= option, 505
NOINT option, 505
SOLUTION option, 505
LMIXED procedure, OPTIMIZATION statement, 506
ABSCONV= option, 506
ABSFCOV= option, 506
ABSGCONV= option, 506
FCOV= option, 507
FCOV2= option, 507
GCONV= option, 507
GCONV2= option, 508
MAXFUNC= option, 508
MAXITER= option, 508
MAXTIME= option, 508
MINITER= option, 508
TECHNIQUE= option, 508
XCONV= option, 509
LMIXED procedure, OUTPUT statement, 509
ALPHA= option, 510
COPYV AR= option, 510
keyword= option, 510
OUT= option, 500, 510
LMIXED procedure, PARMS statement, 511
EQCONS= option, 512
HOLD= option, 512
LOWERB= option, 512
NOITER option, 513
PARMSDATA= option, 513
PDATA= option, 513
UPPERB= option, 514
LMIXED procedure, PROC LMIXED statement, 496
ABSCONV option, 44
ABSFCOV option, 44
ABSGCONV option, 44
ABSGTOL option, 44
ABSTOL option, 44
ABSTOL option, 44
ABSTOL option, 44
ABSTOL option, 44
ABSTOL option, 44
DATA= option, 497
DMMETHOD= option, 497
FCOV option, 45
FCOV2 option, 45
FTOL option, 45
FTOL2 option, 45
GCONV option, 46
GCONV2 option, 46
GTOL option, 46
GTOL2 option, 46
ITDETAILS option, 498
MAXCLPRINT= option, 498
MAXFUNC= option, 47
MAXITER= option, 47
MAXTIME= option, 47
METHOD= option, 498
MINITER= option, 47
MMEQ option, 498
NAMELEN= option, 498
NOCLPRINT option, 498
NOINFO option, 499
NOITPRINT option, 499
NOPRINT option, 499
NOPROFILE option, 499
NORMALIZE= option, 48
RANKS option, 499
SIMPLE option, 499
SINGCHOL= option, 499
SINGRES= option, 499
SINGULAR= option, 499
TECHNIQUE= option, 48
TIMING option, 499
XCONV option, 48
XTOL option, 48
LMIXED procedure, RANDOM statement, 514
ALPHA= option, 515
CL option, 515
SOLUTION option, 515
SUBJECT= option, 515
TYPE= option, 515
LMIXED procedure, REPEATED statement, 523
GROUP= option, 524
R option, 524
RC option, 524
RCI option, 524
RCORR option, 524
RI option, 524
SUBJECT= option, 524
TYPE= option, 524
LMIXED procedure, SELECTION statement
ADAPTIVE option, 38
CHOOSE= option, 38
COMPETITIVE option, 38
CRITERION= option, 38
DETAILS= option, 40, 41
FAST option, 39
HIERARCHY= option, 41
LSCOEFFS option, 39
MAXEFFECTS= option, 39
MAXSTEPS= option, 39
METHOD= option, 37
MINEFFECTS= option, 39
ORDERSELECT option, 41
SELECT= option, 39
SELECTION= option, 43
SLE= option, 39
SLENTRY= option, 39
SLS= option, 39
SLSTAY= option, 39
STOP= option, 40
STOPHORIZON= option, 43
LOGLIK option
OUTPUT statement (MBC), 632
LOGLIKENEWALL option
PROC PHSELECT statement, 753
LOGSELECT procedure, CLASS statement, 563
DESCENDING option, 13, 1173
MISSING option, 13, 1173
ORDER= option, 13
PARAM= option, 14
REF= option, 15
SPLIT option, 15
LOGSELECT procedure, CODE statement, 563
COMMENT option, 17
FILE= option, 17
FORMATWIDTH= option, 17
INDENTSIZE= option, 17
IPROB option, 18
LABELD= option, 18
LINESIZE= option, 18
NOTRIM option, 18
OUT= option, 18
PCATALL option, 18
LOGSELECT procedure, DISPLAY statement
CASESENSITIVE option, 20, 565
EXCLUDE option, 20, 565
EXCLUDEALL option, 20, 565
TRACE option, 20, 565
LOGSELECT procedure, DISPLAYOUT statement
INCLUDEALL option, 21, 565
NOREPLACE option, 21, 565
REPEATED option, 21, 566
LOGSELECT procedure, EFFECT statement
BASIS option (spline), 28
DATABASEBOUNDARY option (spline), 28
DEGREE option (polynomial), 26
DEGREE option (spline), 29
DETAILS option (multimember), 25
DETAILS option (polynomial), 26
DETAILS option (spline), 29
KNOTMAX= option (spline), 29
KNOTMETHOD option (spline), 29
KNOTMIN= option (spline), 30
MDEGREE option (polynomial), 26
NATURALCUBIC option (spline), 30
NOEFFECT option (multimember), 25
NOSEPARATE option (polynomial), 26
SEPARATE option (spline), 30
SPLIT option (spline), 31
STANDARDIZE option (polynomial), 26
LOGSELECT procedure, MODEL statement, 567
CENTER option, 570
CENTERLASSO option, 570
CLB option, 570
DESCENDING option, 568
INCLUDE option, 570
INFORMATIVE option, 570
LINK= option, 570
NOCHECK option, 562
NOINT option, 571
OFFSET= option, 571
ORDER= option, 569
PRIOR= option, 571
START option, 572
TYPE3 option, 572
LOGSELECT procedure, OUTPUT statement, 572
ALL option, 573
ALPHA= option, 573
CBAR= option, 574
COPYVAR= option, 573
DIFCHISQ= option, 574
DIFDEV= option, 574
FIXEDOFFSET= option, 573
H= option, 575
INDIVIDUAL= option, 575
INTO= option, 575
IPRED= option, 575
LCL= option, 575
LCLM= option, 575
LEVEL= option, 575
LINP= option, 577
LOWER= option, 575
LOWERXBETA= option, 575
OBSCAT option, 573
OUT= option, 573
POST= option, 575
PREDICTED= option, 575
PREDPROBS option, 573
RESCHI= option, 576
RESDEV= option, 576
RESIDUAL= option, 576
RESLIK= option, 576
RESRAW= option, 576
RESWORK= option, 576
ROLE= option, 576
STDRRESCHI= option, 576
STDRRESDEV= option, 576
STDXBETA= option, 577
UCL= option, 577
UCLM= option, 577
LOGSELECT procedure, PARTITION statement
FRACTION option, 36, 577
ROLEVAR= option, 36, 577
LOGSELECT procedure, PROC LOGSELECT statement, 558
ABS_CONV option, 44
ABSFC_CONV option, 44
ABSF_TOL option, 44
ABSG_CONV option, 44
ABSG_TOL option, 44
ABSTOL option, 44
ABS_CONV2 option, 45
ABXTOL option, 45
ALPHA option, 559
ASSOCIATION option, 559
BINEPS option, 559
CORRB option, 560
COVB option, 560
CTABLE option, 560
DATA option, 561
FC_CONV option, 45
FC_CONV2 option, 45
FTOL option, 45
FTOL2 option, 45
G_CONV option, 46
G_CONV2 option, 46
GTOL option, 46
GTOL2 option, 46
ITHIST option, 561
LASSO_RHO= option, 561
LASSO_STEPS= option, 561
LASSO_TOL= option, 562
MAX_FUNC= option, 47
MAX_ITER= option, 47
MAXOPTBATCH= option, 562
MAX_TIME= option, 47
MINITER= option, 47
NO_CLPRINT option, 562
NORMALIZE= option, 48
HOSTDERR option, 562
PAGE_OBS option, 562
PARTFIT option, 562
STB option, 562
TECHNIQUE= option, 48
X_CONV option, 48
XTOL option, 48
LOGSELECT procedure, SELECTION statement, 578
ADAPTIVE option, 38
CHOOSE= option, 38
COMPETITIVE option, 38
CRITERION= option, 38
DETAILS= option, 40, 41
FAST option, 39
HIERARCHY= option, 41
LSCOEFFS option, 39
MAXEFFECTS= option, 39
MAXSTEPS= option, 39
METHOD= option, 37
MINEFFECTS= option, 39
ORDERSELECT option, 41
SELECT= option, 39
SELECTION= option, 43
SLE= option, 39
SLENSTRY= option, 39
SLS= option, 39
SLSTAY= option, 39
STOP= option, 40
STOPHORIZON= option, 43
LOGSELECT procedure, syntax, 558
LOWER= option
  OUTPUT statement (GENSELECT), 389
  OUTPUT statement (LOGSELECT), 575
  PREDICT statement (NLMOD), 689
LOWERB= option
  PARMS statement (LMMixed), 512
LOWERXBETA= option
  OUTPUT statement (GENSELECT), 389
  OUTPUT statement (LOGSELECT), 575
LRCHI option
  EXACT statement (FREQTAB), 130
  OUTPUT statement (FREQTAB), 143
LRCHI option (CHISQ)
  TABLES statement (FREQTAB), 158
LSCOEFFS option
  SELECTION statement (ASSESS), 39
  SELECTION statement (BINNING), 39
  SELECTION statement (CARDINALITY), 39
  SELECTION statement (CORRELATION), 39
  SELECTION statement (GAMMOD), 39
  SELECTION statement (GENSELECT), 39
  SELECTION statement (ICA), 39
  SELECTION statement (KCLUS), 39
  SELECTION statement (KCLUS), 39
  SELECTION statement (LMixed), 39
  SELECTION statement (LOGSELECT), 39
  SELECTION statement (MBC), 39
  SELECTION statement (NLMod), 39
  SELECTION statement (PARTITION), 39
  SELECTION statement (PCA), 39
  SELECTION statement (PHSELECT), 39
  SELECTION statement (PLSmod), 39
  SELECTION statement (QTRSELECT), 39
  SELECTION statement (REGSELECT), 39
  SELECTION statement (TREESPLIT), 39
  SELECTION statement (V ARIMPUTE), 39
  SELECTION statement (V ARREDUCE), 39
M= option
  MODEL statement (GAMMOD), 325
MANTELFLIESS option (CMH)
  TABLES statement (FREQTAB), 156
MARGIN= option (BINOMIAL)
  TABLES statement (FREQTAB), 160
MARGIN= option (RELRISK)
  TABLES statement (FREQTAB), 186
MARGIN= option (RISKDIFF)
  TABLES statement (FREQTAB), 191
MATRIX= option
  PROC VARREDUCE statement, 1172
MAXBAYES= option
  AUTOTUNE statement, 1020
MAXBRANCH= option
  PROC TREESPLIT statement, 1014
MAXCLPRINT= option
  PROC LMixed statement, 498
MAXCLUSTERS= option
  PROC KCLUS statement, 460
MAXDEPTH= option
  PROC TREESPLIT statement, 1015
MAXDF= option
  MODEL statement (GAMMOD), 325
MAXEFFECTS= option
  REDUCE statement, 1177
  SELECTION statement (ASSESS), 39
  SELECTION statement (BINNING), 39
  SELECTION statement (CARDINALITY), 39
  SELECTION statement (CORRELATION), 39
  SELECTION statement (GAMMOD), 39
  SELECTION statement (GENSELECT), 39
  SELECTION statement (ICA), 39
  SELECTION statement (KCLUS), 39
  SELECTION statement (LMixed), 39
  SELECTION statement (LOGSELECT), 39
  SELECTION statement (MBC), 39
  SELECTION statement (NLMod), 39
  SELECTION statement (PARTITION), 39
  SELECTION statement (PCA), 39
  SELECTION statement (PHSELECT), 39
  SELECTION statement (PLSmod), 39
  SELECTION statement (QTRSELECT), 39
  SELECTION statement (REGSELECT), 39
  SELECTION statement (TREESPLIT), 39
  SELECTION statement (V ARIMPUTE), 39
  SELECTION statement (V ARREDUCE), 39
MAXEV ALS= option
  AUTOTUNE statement, 1020
MAXFUNC= option
  OPTIMIZATION statement, 508
  PROC ASSESS statement, 47
  PROC BINNING statement, 47
  PROC CARDINALITY statement, 47
PROC CORRELATION statement, 47
PROC GAMMOD statement, 47
PROC GENSELECT statement, 47
PROC ICA statement, 47
PROC KCLUS statement, 47
PROC LMMIXED statement, 47
PROC LOGSELECT statement, 47
PROC MBC statement, 47
PROC NLMOD statement, 47
PROC PARTITION statement, 47
PROC PCA statement, 47
PROC PHSELECT statement, 47
PROC PLSMOD statement, 47
PROC QTRSELECT statement, 47
PROC REGSELECT statement, 47
PROC TREESPLIT statement, 47
PROC VARIMPUTE statement, 47
PROC VAREDUCE statement, 47
MAXITER= option
AUTOTUNE statement, 1020
OPTIMIZATION statement, 508
PROC ASSESS statement, 47, 1078
PROC BINNING statement, 47
PROC CARDINALITY statement, 47
PROC CORRELATION statement, 47
PROC GAMMOD statement, 47
PROC GENSELECT statement, 47
PROC ICA statement, METHOD=DEFLATION option, 436
PROC ICA statement, METHOD=SYMmetric option, 437
PROC KCLUS statement, 47, 460
PROC LMMIXED statement, 47
PROC LOGSELECT statement, 47
PROC MBC statement, 47
PROC NLMOD statement, 47
PROC PARTITION statement, 47
PROC PCA statement, 47
PROC PCA statement, METHOD=ITERSGS option, 713
PROC PCA statement, METHOD=NIPALS option, 714
PROC PHSELECT statement, 47
PROC PLSMOD statement, 47
PROC PLSMOD statement, METHOD=PLS option, 802
PROC QTRSELECT statement, 47
PROC REGSELECT statement, 47
PROC TREESPLIT statement, 47
PROC VARIMPUTE statement, 47
PROC VAREDUCE statement, 47
PROC MBC statement, 628
REDUCE statement, 1177
MAXITER= option (PLCORR)
TABLES statement (FREQTAB), 170
MAXKNOTS= option
MODEL statement (GAMMOD), 325
MAXLEVELS= option
PROC CARDINALITY statement, 1125
TABLES statement (FREQTAB), 166
MAXOPTBATCH= option
PROC GENSELECT statement, 374
PROC LOGSELECT statement, 562
MAXPHI= option
MODEL statement (GAMMOD), 327
MAXPOST option
OUTPUT statement (MBC), 632
MAXSMOOTH= option
MODEL statement (GAMMOD), 325
MAXSTEPS= option
SELECTION statement (ASSESS), 39
SELECTION statement (BINNING), 39
SELECTION statement (CARDINALITY), 39
SELECTION statement (CORRELATION), 39
SELECTION statement (GAMMOD), 39
SELECTION statement (GENSELECT), 39
SELECTION statement (ICA), 39
SELECTION statement (KCLUS), 39
SELECTION statement (LMMIXED), 39
SELECTION statement (LOGSELECT), 39
SELECTION statement (MBC), 39
SELECTION statement (NLMOD), 39
SELECTION statement (PARTITION), 39
SELECTION statement (PCA), 39
SELECTION statement (PHSELECT), 39
SELECTION statement (PLSMOD), 39
SELECTION statement (QTRSELECT), 39
SELECTION statement (REGSELECT), 39
SELECTION statement (TREESPLIT), 39
SELECTION statement (VARIMPUTE), 39
SELECTION statement (VAREDUCE), 39
MAXTIME= option
AUTOTUNE statement, 1021
EXACT statement (FREQTAB), 134
OPTIMIZATION statement, 508
PROC ASSESS statement, 47
PROC BINNING statement, 47
PROC CARDINALITY statement, 47
PROC CORRELATION statement, 47
PROC GAMMOD statement, 47
PROC GENSELECT statement, 47
PROC ICA statement, 47
PROC KCLUS statement, 47
PROC LMMIXED statement, 47
PROC LOGSELECT statement, 47
PROC MBC statement, 47
PROC NLMOD statement, 47
PROC PARTITION statement, 47
PROC PCA statement, 47
PROC PHSELECT statement, 47
PROC PLSMOD statement, 47
PROC QTRSELECT statement, 47
PROC REGSELECT statement, 47
PROC TREESPLIT statement, 47
PROC VARIMPUTE statement, 47
PROC VARREDUCE statement, 47
MBC procedure, 626
PROC MBC statement, 626
syntax, 626
MBC procedure, CLASS statement
DESCENDING option, 13, 1173
MISSING option, 13, 1173
ORDER= option, 13
PARAM= option, 14
REF= option, 15
SPLIT option, 15
MBC procedure, CODE statement
COMMENT option, 17
FILE= option, 17
FORMATWIDTH= option, 17
INDENTSIZE= option, 17
LABELID= option, 18
LINESIZE= option, 18
NOTRIM option, 18
OUT= option, 18
MBC procedure, DISPLAY statement
CASESENSITIVE option, 20, 630
EXCLUDE option, 20, 630
EXCLUDEALL option, 20, 630
TRACE option, 20, 630
MBC procedure, DISPLAYOUT statement
INCLUDEALL option, 21, 631
NOREPLACE option, 21, 631
REPEATED option, 21, 631
MBC procedure, EFFECT statement
BASIS option (spline), 28
DATABASEBOUNDARY option (spline), 28
DEGREE option (polynomial), 26
DEGREE option (spline), 29
DETAILS option (multimember), 25
DETAILS option (polynomial), 26
DETAILS option (spline), 29
KNOTMAX= option (spline), 29
KNOTMETHOD option (spline), 29
KNOTMIN= option (spline), 30
MDEGREE option (polynomial), 26
NATURALCUBIC option (spline), 30
NOEFFECT option (multimember), 25
NOSEPARATE option (polynomial), 26
SEPARATE option (spline), 30
SPLIT option (spline), 31
STANDARDIZE option (polynomial), 26
MBC procedure, OUTPUT statement, 631
ALL option, 632
COPYYVAR= option, 632
CURRCLUS option, 632
LOGLIK option, 632
MAXPOST option, 632
NEXTCLUS option, 632
OUT= option, 631
MBC procedure, PROC MBC statement
ABSCONV option, 44
ABSFCONV option, 44
ABSFTOL option, 44
ABSGCONV option, 44
ABSGTOL option, 44
ABSTOL option, 44
ABSTOL option, 44
ABSTOL option, 45
ABSXCONV option, 45
ABSXTOL option, 45
DATA= option, 627
FCONV option, 45
FCONV2 option, 45
FTOL option, 45
FTOL2 option, 45
GCONV option, 46
GCONV2 option, 46
GTOL option, 46
GTOL2 option, 46
MAXFUNC= option, 47
MAXITER= option, 47
MAXTIME= option, 47
MINITER= option, 47
NORMALIZE= option, 48
TECHNIQUE= option, 48
XCONV option, 48
XTOL option, 48
MBC procedure, PROC MBC statement
COVSTRUCT= option, 627
CRITERION= option, 627
EMCRITERION= option, 627
INIT= option, 627
ITHIST option, 628
MAXITER= option, 628
NCCLUSTERS= option, 628
NOISE= option, 628
SEED= option, 628
SINGPARM= option, 628
SINGULAR= option, 628
TECHNIQUE= option, 629
TOPMODELS= option, 629
MBC procedure, SELECTION statement
ADAPTIVE option, 38
CHOOSE= option, 38
COMPETITIVE option, 38
CRITERION= option, 38
DETAILS= option, 40, 41
FAST option, 39
HIERARCHY= option, 41
LSCOEFFS option, 39
MAXEFFECTS= option, 39
MAXSTEPS= option, 39
METHOD= option, 37
MINEFFECTS= option, 39
ORDERSELECT option, 41
SELECT= option, 39
SELECTION= option, 43
SLE= option, 39
SLENTRY= option, 39
SLS= option, 39
SLSTAY= option, 39
STOP= option, 40
STOPHORIZON= option, 43
MBC procedure, STORE statement, 633
MC option
   EXACT statement (FREQTAB), 134
MCHART statement
   MCHART procedure, 942
MCNEM option
   EXACT statement (FREQTAB), 130
   OUTPUT statement (FREQTAB), 143
MDEGREE option
   EFFECT statement, polynomial (ASSESS), 26
   EFFECT statement, polynomial (BINNING), 26
   EFFECT statement, polynomial (CARDINALITY), 26
   EFFECT statement, polynomial (CORRELATION), 26
   EFFECT statement, polynomial (GAMMOD), 26
   EFFECT statement, polynomial (GENSELECT), 26
   EFFECT statement, polynomial (ICA), 26
   EFFECT statement, polynomial (KCLUS), 26
   EFFECT statement, polynomial (LMIXED), 26
   EFFECT statement, polynomial (LOGSELECT), 26
   EFFECT statement, polynomial (MBC), 26
   EFFECT statement, polynomial (NLMOD), 26
   EFFECT statement, polynomial (PARTITION), 26
   EFFECT statement, polynomial (PCA), 26
   EFFECT statement, polynomial (PHSELECT), 26
   EFFECT statement, polynomial (PLSMOD), 26
   EFFECT statement, polynomial (QTRSELECT), 26
   EFFECT statement, polynomial (REGSELECT), 26
   EFFECT statement, polynomial (TREESPLIT), 26
   EFFECT statement, polynomial (VARIMPUTE), 26
   EFFECT statement, polynomial (VARREDUCE), 26
MEASURES option
   EXACT statement (FREQTAB), 130
   OUTPUT statement (FREQTAB), 143
   TABLES statement (FREQTAB), 166
   TEST statement (FREQTAB), 195
MEDCENTRAL= option
   chart statements (SPC), 951
METHOD= option
   MODEL statement (GAMMOD), 328
   PROC BINNING statement, 1099
   PROC ICA statement, 436
   PROC LMIXED statement, 498
   PROC PCA statement, 713
   PROC PLSMOD statement, 802
   SELECTION statement (ASSESS), 37
   SELECTION statement (BINNING), 37
   SELECTION statement (CARDINALITY), 37
   SELECTION statement (CORRELATION), 37
   SELECTION statement (GAMMOD), 37
   SELECTION statement (GENSELECT), 37
   SELECTION statement (ICA), 37
   SELECTION statement (KCLUS), 37
   SELECTION statement (LMIXED), 37
   SELECTION statement (LOGSELECT), 37
   SELECTION statement (MBC), 37
   SELECTION statement (NLMOD), 37
   SELECTION statement (PARTITION), 37
   SELECTION statement (PCA), 37
   SELECTION statement (PHSELECT), 37
   SELECTION statement (PLSMOD), 37
   SELECTION statement (QTRSELECT), 37
   SELECTION statement (REGSELECT), 37
   SELECTION statement (TREESPLIT), 37
   SELECTION statement (VARIMPUTE), 37
   SELECTION statement (VARREDUCE), 37
METHOD= option (RELRISK)
   EXACT statement (FREQTAB), 132
   TABLES statement (FREQTAB), 186
METHOD= option (RISKDIFF)
   EXACT statement (FREQTAB), 133
   TABLES statement (FREQTAB), 191
METHOD=FM option (RELRISK)
   TABLES statement (FREQTAB), 191
METHOD=HA option (RISKDIFF)
   TABLES statement (FREQTAB), 191
METHOD=LR option (RELRISK)
TABLES statement (FREQTAB), 187
METHOD=NEWCOMBE option (RISKDIFF)
TABLES statement (FREQTAB), 191
METHOD=WALD option (RELRISK)
TABLES statement (FREQTAB), 187
METHOD=WALD option (RISKDIFF)
TABLES statement (FREQTAB), 191
METHOD=WALDMODIFIED option (RELRISK)
TABLES statement (FREQTAB), 187
MHCHI option
EXACT statement (FREQTAB), 131
OUTPUT statement (FREQTAB), 143
MHOR option
OUTPUT statement (FREQTAB), 143
MHRRC1 option
OUTPUT statement (FREQTAB), 144
MHRRC2 option
OUTPUT statement (FREQTAB), 144
MIDP option
EXACT statement (NPRIWAY), 135
MINEFFECTS= option
SELECTION statement (ASSESS), 39
SELECTION statement (BINNING), 39
SELECTION statement (CARDINALITY), 39
SELECTION statement (CORRELATION), 39
SELECTION statement (GAMMOD), 39
SELECTION statement (GENSELECT), 39
SELECTION statement (ICA), 39
SELECTION statement (KCLUS), 39
SELECTION statement (LMIXED), 39
SELECTION statement (LOGSELECT), 39
SELECTION statement (MBC), 39
SELECTION statement (NLMOD), 39
SELECTION statement (PARTITION), 39
SELECTION statement (PCA), 39
SELECTION statement (PHSELECT), 39
SELECTION statement (PLSMD), 39
SELECTION statement (QTRSELECT), 39
SELECTION statement (REGSELECT), 39
SELECTION statement (TREESPLIT), 39
SELECTION statement (VARIMPUTE), 39
SELECTION statement (VARREDUCE), 39
MINITER= option
OPTIMIZATION statement, 508
PROC ASSESS statement, 47
PROC BINNING statement, 47
PROC CARDINALITY statement, 47
PROC CORRELATION statement, 47
PROC GAMMOD statement, 47
PROC GENSELECT statement, 47
PROC ICA statement, 47
PROC KCLUS statement, 47
PROC LMIXED statement, 47
PROC LOGSELECT statement, 47
PROC MBC statement, 47
PROC NLMOD statement, 47
PROC PARTITION statement, 47
PROC PCA statement, 47
PROC PHSELECT statement, 47
PROC PLSMD statement, 47
PROC QTRSELECT statement, 47
PROC REGSELECT statement, 47
PROC TREESPLIT statement, 47
PROC VARIMPUTE statement, 47
PROC VARREDUCE statement, 47
MINLEAFSIZE= option
PROC TREESPLIT statement, 1015
MINPHI= option
MODEL statement (GAMMOD), 328
MINSMOOTH= option
MODEL statement (GAMMOD), 325
MINUSEINSEARCH= option
PROC TREESPLIT statement, 1015
MINVARIANCEINCREMENT= option
REDUCE statement, 1177
MISS option
VIICODE statement (TREESPLIT), 1032
MISSING option
CLASS statement (ASSESS), 13, 1173
CLASS statement (BINNING), 13, 1173
CLASS statement (CARDINALITY), 13, 1173
CLASS statement (CORRELATION), 13, 1173
CLASS statement (GAMMOD), 13, 1173
CLASS statement (GENSELECT), 13, 1173
CLASS statement (ICA), 13, 1173
CLASS statement (KCLUS), 13, 1173
CLASS statement (LMIXED), 13, 1173
CLASS statement (LOGSELECT), 13, 1173
CLASS statement (MBC), 13, 1173
CLASS statement (NLMOD), 13, 1173
CLASS statement (PARTITION), 13, 1173
CLASS statement (PCA), 13, 1173
CLASS statement (PHSELECT), 13, 1173
CLASS statement (PLSMD), 13, 1173
CLASS statement (QTRSELECT), 13, 1173
CLASS statement (REGSELECT), 13, 1173
CLASS statement (TREESPLIT), 13, 1173
CLASS statement (VARIMPUTE), 13, 1173
CLASS statement (VARREDUCE), 13, 1173
PROC FREQTAB statement, 125
MISSINGBINSTATS= option
PROC BINNING statement, 1102
MISSINGEVALNONEVENT= option
PROC BINNING statement, 1102
MISSPRINT option
TABLES statement (FREQTAB), 125
MMEQ option
PROC LMIXED statement, 498
MODEL statement
  GAMMOD procedure, 320
  GENSELECT procedure, 380
  LMIXED procedure, 504
  LOGSELECT procedure, 567
  MODELMATRIX procedure, 662
  NLMOD procedure, 685
  PHSELECT procedure, 760
  PLSMOD procedure, 807
  QTRSELECT procedure, 845
  REGSELECT procedure, 899
  TREESPLIT procedure, 1028

MODELMATRIX procedure
  MODEL statement, 662
  OUTPUT statement, 663
  PROC MODELMATRIX statement, 658

MODELMATRIX procedure, DISPLAY statement
  CASESENSITIVE option, 660
  EXCLUDE option, 660
  EXCLUDEALL option, 660
  TRACE option, 660

MODELMATRIX procedure, DISPLAYOUT statement
  INCLUDEALL option, 660
  NOREPLACE option, 660
  REPEATED option, 660

MODELMATRIX procedure, MODEL statement, 662
  INFORMATIVE option, 663
  NOINT option, 663

MODELMATRIX procedure, OUTPUT statement, 663
  COPYVARS= option, 664
  OUT= option, 663
  PREFIX= option, 664

MODELMATRIX procedure, PROC MODELMATRIX statement, 658

MRCHART statement
  MRCHART procedure, 943
  MULTIPLY option
    VIICODE statement (TREESPLIT), 1032

N option
  OUTPUT statement (FREQTAB), 144

N= option
  EXACT statement (FREQTAB), 135
  PROC ICA statement, 437
  PROC PCA statement, 715

NAMELEN= option
  PROC LMIXED statement, 498

NATURALCUBIC option
  EFFECT statement, spline (ASSESS), 30
  EFFECT statement, spline (Binning), 30
  EFFECT statement, spline (CARDINALITY), 30
  EFFECT statement, spline (CORRELATION), 30
  EFFECT statement, spline (GAMMOD), 30
  EFFECT statement, spline (GENSELECT), 30
  EFFECT statement, spline (ICA), 30
  EFFECT statement, spline (KCLUS), 30
  EFFECT statement, spline (LMIXED), 30
  EFFECT statement, spline (LOGSELECT), 30
  EFFECT statement, spline (MBC), 30
  EFFECT statement, spline (PARTITION), 30
  EFFECT statement, spline (PCA), 30
  EFFECT statement, spline (PHSELECT), 30
  EFFECT statement, spline (PLSMOD), 30
  EFFECT statement, spline (QTRSELECT), 30
  EFFECT statement, spline (REGSELECT), 30
  EFFECT statement, spline (TREESPLIT), 30
  EFFECT statement, spline (VARIIMPUTE), 30
  EFFECT statement, spline (VARREDUCE), 30

NBINS= option
  PROC ASSESS statement, 1078

NCONVITER= option
  PROC MBC statement, 628

NCONVITER= option
  PROC ASSESS statement, 1078

NCONVITER= option
  PROC ASSESS statement, 1078

NEXTCLUS option
  OUTPUT statement (MBC), 632

NFAC= option
  PROC PLSMOD statement, 802

NITER= option
  PROC PCA statement, METHOD=RANDOM option, 714

NLEVELS option
  PROC FREQTAB statement, 125

NLEVELS option
  PROC FREQTAB statement, 125

NLMOD procedure, 679
  PROC NLMOD statement, 680
  syntax, 679

NLMOD procedure, BOUNDS statement, 682

NLMOD procedure, CLASS statement
  DESCENDING option, 13, 1173
  MISSING option, 13, 1173
  ORDER= option, 13
  PARAM= option, 14
  REF= option, 15
  SPLIT option, 15

NLMOD procedure, CODE statement
  COMMENT option, 17
  FILE= option, 17
  FORMATWIDTH= option, 17
  INDENTSIZE= option, 17
  LABELID= option, 18
  LINESIZE= option, 18
  NOTRIM option, 18
  OUT= option, 18
NLMOD procedure, DISPLAY statement
  CASESENSITIVE option, 20, 683
  EXCLUDE option, 20, 683
  EXCLUDEALL option, 20, 683
  TRACE option, 20, 683
NLMOD procedure, DISPLAYOUT statement
  INCLUDEALL option, 21, 684
  NOREPLACE option, 21, 684
  REPEATED option, 21, 684
NLMOD procedure, EFFECT statement
  BASIS option (spline), 28
  DATABOUNDARY option (spline), 28
  DEGREE option (polynomial), 26
  DEGREE option (spline), 29
  DETAILS option (multimember), 25
  DETAILS option (polynomial), 26
  DETAILS option (spline), 29
  KNOTMAX= option (spline), 29
  KNOTMETHOD option (spline), 29
  KNOTMIN= option (spline), 30
  MDEGREE option (polynomial), 26
  NATURALCUBIC option (spline), 30
  NOEFFECT option (multimember), 25
  NOSEPARATE option (polynomial), 26
  SEPARATE option (spline), 30
  SPLIT option (spline), 31
  STANDARDIZE option (polynomial), 26
NLMOD procedure, ESTIMATE statement, 684
  ALPHA= option, 684
  DF= option, 684
NLMOD procedure, MODEL statement, 685
NLMOD procedure, PARAMETERS statement, 685
NLMOD procedure, PREDICT statement, 688
  ALPHA= option, 688
  DF= option, 688
  LOWER= option, 689
  PRED= option, 689
  PROBT= option, 689
  STDERR= option, 689
  TVALE= option, 689
  UPPER= option, 689
NLMOD procedure, PROC NLMOD statement, 680
  ABSCONV option, 44
  ABSFCNV option, 44
  ABSFTOL option, 44
  ABSGCONV option, 44
  ABSGTOL option, 44
  ABSTOL option, 44
  ABSXCONV option, 45
  ABSXTOFL option, 45
  ALPHA= option, 681
  CORR option, 681
  COV option, 681
  DATA= option, 681
  DF= option, 681
  ECOV option, 681
  ECORR option, 681
  FCONV option, 45
  FCONV2 option, 45
  FTOL option, 45
  FTOL2 option, 45
  GCONV option, 46
  GCONV2 option, 46
  GTOL option, 46
  GTOL2 option, 46
  MAXFUNCTION= option, 47
  MAXITER= option, 47
  MAXTIME= option, 47
  MINITER= option, 47
  NOTPRINT option, 681
  NOPRINT option, 681
  NORMALIZE= option, 48
  OUT= option, 681
  SINGULAR= option, 682
  TECHNIQUE= option, 48
  XCONV option, 48
  XTOL option, 48
NLMOD procedure, RESTRICT statement, 689
NLMOD procedure, SELECTION statement
  ADAPTIVE option, 38
  CHOOSE= option, 38
  COMPETITIVE option, 38
  CRITERION= option, 38
  DETAILS= option, 40, 41
  FAST option, 39
  HIERARCHY= option, 41
  LSCOEFFS option, 39
  MAXEFFECTS= option, 39
  MAXSTEPS= option, 39
  METHOD= option, 37
  MINEFFECTS= option, 39
  ORDERSELECT option, 41
  SELECT= option, 39
  SELECTION= option, 43
  SLE= option, 39
  SLENTRY= option, 39
  SLS= option, 39
  SLSTAY= option, 39
  STOP= option, 40
  STOPHORIZON= option, 43
NMISS option
OUTPUT statement (FREQTAB), 144
NOSIGMACHECK option
  chart statements (SPC), 951
NOCENTER option
  PROC ICA statement, 437
  PROC PCA statement, METHOD=ITERGS option, 714
PROC PCA statement, METHOD=NIPALS option, 714
PROC PCA statement, METHOD=RANDOM option, 714
PROC PLSMOD statement, 803
NOCLPRINT option
PROC GAMMOD statement, 315
PROC GENSELECT statement, 374
PROC LMIXED statement, 498
PROC LOGSELECT statement, 562
PROC PHSELECT statement, 753
PROC PLSMOD statement, 803
PROC QTRSELECT statement, 840
PROC REGSELECT statement, 894
NOCOL option
   TABLES statement (FREQTAB), 167
NOCORR option
PROC CORRELSATION statement, 95
NOCUM option
   TABLES statement (FREQTAB), 167
NOCVSTIDIZE option
PROC PLSMOD statement, 803
NOEFFECT option
   EFFECT statement, multimember (ASSESS), 25
   EFFECT statement, multimember (BINNING), 25
   EFFECT statement, multimember (CARDINALITY), 25
   EFFECT statement, multimember (CORRELATION), 25
   EFFECT statement, multimember (GAMMOD), 25
   EFFECT statement, multimember (GENSELECT), 25
   EFFECT statement, multimember (ICA), 25
   EFFECT statement, multimember (KCLUS), 25
   EFFECT statement, multimember (LMIXED), 25
   EFFECT statement, multimember (LOGSELECT), 25
   EFFECT statement, multimember (MBC), 25
   EFFECT statement, multimember (NLMODE), 25
   EFFECT statement, multimember (PARTITION), 25
   EFFECT statement, multimember (PCA), 25
   EFFECT statement, multimember (PHSELECT), 25
   EFFECT statement, multimember (PLSMOD), 25
   EFFECT statement, multimember (QTRSELECT), 25
   EFFECT statement, multimember (REGSELECT), 25
   EFFECT statement, multimember (TREESPLIT), 25
   EFFECT statement, multimember (VARIMPUTE), 25
   EFFECT statement, multimember (VARREDUCE), 25
NOFREQ option
   TABLES statement (FREQTAB), 167
NOGRIDSHUFFLE option
PROC AUTOTUNE statement, 1021
NOINFO option
PROC LMIXED statement, 499
NOINT option
   MODEL statement (GENSELECT), 374, 386
   MODEL statement (LMIXED), 505
   MODEL statement (LOGSELECT), 562, 571
   MODEL statement (MODELMATRIX), 663
   MODEL statement (QTRSELECT), 846
   MODEL statement (REGSELECT), 900
   PROC PCA statement, 715
NOISE= option
PROC MBC statement, 628
NOIT option
PROC PARMS statement (LMIXED), 513
NOITPRINT option
PROC LMIXED statement, 499
PROC NLMOD statement, 681
NOMISS option
PROC CORRELATION statement, 95
NONINFERIORITY option (BINOMIAL)
   TABLES statement (FREQTAB), 156
NONINFERIORITY option (RELISIRC)
   TABLES statement (FREQTAB), 187
NONINFERIORITY option (RISKDIFF)
   TABLES statement (FREQTAB), 192
NOPERCENT option
PROC CORRELATION statement, 96
NOPRINT option
chart statements (SPC), 951
PROC FREQUENT statement, 125
PROC GAMMOD statement, 315
PROC LMIXED statement, 499
PROC NLMOD statement, 681
PROC TREESPLIT statement, 1015
PROC VARREDUCE statement, 1172
TABLES statement (FREQTAB), 167
NOPROB option
PROC CORRELATION statement, 96
NOPROFILE option
PROC LMIXED statement, 499
NOREPLACE option
   DISPLAYOUT statement (CORRELATION), 21, 98
   DISPLAYOUT statement (GAMMOD), 21, 320
   DISPLAYOUT statement (GENSELECT), 21, 378
DISPLAYOUT statement (ICA), 21, 439
DISPLAYOUT statement (KCLUS), 21, 464
DISPLAYOUT statement (LMIXED), 21, 503
DISPLAYOUT statement (LOGSELECT), 21, 565
DISPLAYOUT statement (MBC), 21, 631
DISPLAYOUT statement (MODELMATRIX), 660
DISPLAYOUT statement (NLMOD), 21, 684
DISPLAYOUT statement (PARTITION), 21, 1143
DISPLAYOUT statement (PCA), 21, 720
DISPLAYOUT statement (PHSELECT), 21, 758
DISPLAYOUT statement (PLSMOD), 21, 805
DISPLAYOUT statement (QTRSELECT), 21, 843
DISPLAYOUT statement (REGSELECT), 21, 897
DISPLAYOUT statement (VARREDUCE), 21, 1175

NORISKS option (RISKDIFF)
TABLES statement (FREQTAB), 192
NORMALIZE option
MODEL statement (GAMMOD), 328
NORMALIZE= option
PROC ASSESS statement, 48
PROC BINNING statement, 48
PROC CARDINALITY statement, 48
PROC CORRELATION statement, 48
PROC GAMMOD statement, 48
PROC GENSELECT statement, 48
PROC ICA statement, 48
PROC KCLUS statement, 48
PROC LMMIXED statement, 48
PROC LOGSELECT statement, 48
PROC MBC statement, 48
PROC NLMOD statement, 48
PROC PARTITION statement, 48
PROC PCA statement, 48
PROC PHSELECT statement, 48
PROC PLSMOD statement, 48
PROC QTRSELECT statement, 48
PROC REGSELECT statement, 48
PROC TREESPLIT statement, 48
PROC VARIMPUTE statement, 48
PROC VARREDUCE statement, 48

PROC PCA statement, METHOD=RANDOM option, 714
PROC PLSMOD statement, 803

NOSEPARATE option
EFFECT statement, polynomial (ASSESS), 26
EFFECT statement, polynomial (BINNING), 26
EFFECT statement, polynomial (CARDINALITY), 26
EFFECT statement, polynomial (CORRELATION), 26
EFFECT statement, polynomial (GAMMOD), 26
EFFECT statement, polynomial (GENSELECT), 26
EFFECT statement, polynomial (ICA), 26
EFFECT statement, polynomial (KCLUS), 26
EFFECT statement, polynomial (LMIXED), 26
EFFECT statement, polynomial (LOGSELECT), 26
EFFECT statement, polynomial (MBC), 26
EFFECT statement, polynomial (NLMOD), 26
EFFECT statement, polynomial (PARTITION), 26
EFFECT statement, polynomial (PCA), 26
EFFECT statement, polynomial (PHSELECT), 26
EFFECT statement, polynomial (PLSMOD), 26
EFFECT statement, polynomial (QTRSELECT), 26
EFFECT statement, polynomial (REGSELECT), 26
EFFECT statement, polynomial (TREESPLIT), 26
EFFECT statement, polynomial (VARIMPUTE), 26
EFFECT statement, polynomial (VARREDUCE), 26

NOSIMPLE option
PROC CORRELATION statement, 96

NOSPARSE option
TABLES statement (FREQTAB), 167

NOSTDERR option
PROC GENSELECT statement, 374
PROC LOGSELECT statement, 562
PROC PHSELECT statement, 754

NOSURVIVAL option
CODE statement (PHSELECT), 18, 755

NOTRIM option
CODE statement (ASSESS), 18
CODE statement (BINNING), 18
CODE statement (CARDINALITY), 18
CODE statement (CORRELATION), 18
CODE statement (GAMMOD), 18
CODE statement (GENSELECT), 18
CODE statement (ICA), 18
CODE statement (KCLUS), 18
CODE statement (LMIXED), 18
CODE statement (LOGSELECT), 18
CODE statement (MBC), 18
CODE statement (NLMOD), 18
CODE statement (PARTITION), 18
CODE statement (PCA), 18
CODE statement (PHSELECT), 18
CODE statement (PLSMOD), 18
CODE statement (QTRSELECT), 18
CODE statement (REGSELECT), 18
CODE statement (TREESPLIT), 18
CODE statement (V ARIMPUTE), 18
CODE statement (V ARREDUCE), 18
NOWARN option
TABLES statement (FREQTAB), 168
NPCHART statement
NPCHART procedure, 943
NSAMP= option
PROC PLSMOD statement, CVTEST option, 801
NSUBSESSIONWORKERS= option
AUTOTUNE statement, 1021
NSURROGATES= option
PROC TREESPLIT statement, 1015
NTAU option
QUANTILES option (QTRSELECT), 846
NTHREADS
PROC KCLUS statement, 461
NTHREADS= option
PROC ASSESS statement, 1078
PROC PARTITION statement, 1140
PROC SPC statement, 939
PROC V ARIMPUTE statement, 1157
NUMBIN= option
INPUT statement, 1103
PROC BINNING statement, 1101
PROC TREESPLIT statement, 1015
OBJECTIVE= option
AUTOTUNE statement, 1021
OBSCAT option
OUTPUT statement (GENSELECT), 387
OUTPUT statement (LOGSELECT), 573
OFF option
PRUNE statement (TREESPLIT), 1031
OFFSET= option
MODEL statement (GAMMOD), 328
MODEL statement (GENSELECT), 386
MODEL statement (LOGSELECT), 571
MODEL statement (PHSELECT), 761
OPTIMIZATION statement
LMIXED procedure, 506
OR option
EXACT statement (FREQTAB), 131
OUTPUT statement (FREQTAB), 144
ORDER= option
CLASS statement (ASSESS), 13
CLASS statement (BINNING), 13
CLASS statement (CARDINALITY), 13
CLASS statement (CORRELATION), 13
CLASS statement (GAMMOD), 13
CLASS statement (GENSELECT), 13
CLASS statement (ICA), 13
CLASS statement (KCLUS), 13
CLASS statement (LMIXED), 13
CLASS statement (LOGSELECT), 13
CLASS statement (MBC), 13
CLASS statement (NLMOD), 13
CLASS statement (PARTITION), 13
CLASS statement (PCA), 13
CLASS statement (PHSELECT), 13
CLASS statement (PLSMOD), 13
CLASS statement (QTRSELECT), 13
CLASS statement (REGSELECT), 13
CLASS statement (TREESPLIT), 13
CLASS statement (V ARIMPUTE), 13
CLASS statement (V ARREDUCE), 13
MODEL statement (GAMMOD), 323
MODEL statement (GENSELECT), 381
MODEL statement (LOGSELECT), 569
PROC CARDINALITY statement, 1125
PROC FREQTAB statement, 125
VAR statement, 1126
ORDERSELECT option
SELECTION statement (ASSESS), 41
SELECTION statement (BINNING), 41
SELECTION statement (CARDINALITY), 41
SELECTION statement (CORRELATION), 41
SELECTION statement (GAMMOD), 41
SELECTION statement (GENSELECT), 41
SELECTION statement (ICA), 41
SELECTION statement (KCLUS), 41
SELECTION statement (LMIXED), 41
SELECTION statement (LOGSELECT), 41
SELECTION statement (MBC), 41
SELECTION statement (NLMOD), 41
SELECTION statement (PARTITION), 41
SELECTION statement (PCA), 41
SELECTION statement (PHSELECT), 41
SELECTION statement (PLSMOD), 41
SELECTION statement (QTRSELECT), 41
SELECTION statement (REGSELECT), 41
SELECTION statement (TREESPLIT), 41
SELECTION statement (V ARIMPUTE), 41
SELECTION statement (V ARREDUCE), 41
OUT= option
CODE statement (ASSESS), 18
CODE statement (BINNING), 18
CODE statement (CARDINALITY), 18
CODE statement (CORRELATION), 18
CODE statement (GAMMOD), 18
CODE statement (GENSELECT), 18
CODE statement (ICA), 18
CODE statement (KCLUS), 18
CODE statement (LMIXED), 18
CODE statement (LOGSELECT), 18
CODE statement (MBC), 18
CODE statement (NLMODE), 18
CODE statement (PARTITION), 18
CODE statement (PCA), 18
CODE statement (PHSELECT), 18
CODE statement (PLSMOD), 18
CODE statement (QTRSELECT), 18
CODE statement (REGSELECT), 18
CODE statement (TREESPLIT), 18
CODE statement (VARIMPUTE), 18
CODE statement (VARREDUCE), 18
OUTPUT statement, 1104
OUTPUT statement (FREQTAB), 136
OUTPUT statement (GAMMOD), 329
OUTPUT statement (GENSELECT), 387
OUTPUT statement (ICA), 440
OUTPUT statement (KCLUS), 466
OUTPUT statement (LMIXED), 500, 510
OUTPUT statement (LOGSELECT), 573
OUTPUT statement (MBC), 631
OUTPUT statement (MODELMATRIX), 663
OUTPUT statement (PARTITION), 1144
OUTPUT statement (PCA), 721
OUTPUT statement (PHSELECT), 762
OUTPUT statement (PLSMOD), 808
OUTPUT statement (QTRSELECT), 847
OUTPUT statement (REGSELECT), 901
OUTPUT statement (TREESPLIT), 1029
OUTPUT statement (VARIMPUTE), 1158
PROC CORRELATION statement, 96
PROC NLMODEL statement, 681
OUTLIMITS= option
chart statements (SPC), 951
OUTMODEL option
PROC TREESPLIT statement, 1015
OUTP= option
PROC CORRELATION statement, 96
OUTPCT option
TABLES statement (FREQTAB), 169
OUTPUT statement
BINNING procedure, 1103
FREQTAB procedure, 136
GAMMOD procedure, 329
GENSELECT procedure, 386
ICA procedure, 439
LMIXED procedure, 509
LOGSELECT procedure, 572
MBC procedure, 631
MODELMATRIX procedure, 663
PARTITION procedure, 1143
PCA procedure, 720
PHSELECT procedure, 761
PLSMOD procedure, 807
QTRSELECT procedure, 847
REGSELECT procedure, 901
TREESPLIT procedure, 1028
VARIMPUTE procedure, 1158
OUTSTAT= option
PROC KCLUS statement, 461
PROC PCA statement, 715
OUTTABLE= option
chart statements (SPC), 951
P= option (BINOMIAL)
TABLES statement (FREQTAB), 157
PAGE option
PROC FREQTAB statement, 126
PAGEOBS= option
PROC GENSELECT statement, 374
PROC LOGSELECT statement, 562
PARAM= option
CLASS statement (ASSESS), 14
CLASS statement (BINNING), 14
CLASS statement (CARDINALITY), 14
CLASS statement (CORRELATION), 14
CLASS statement (GAMMOD), 14
CLASS statement (GENSELECT), 14
CLASS statement (ICA), 14
CLASS statement (KCLUS), 14
CLASS statement (LMIXED), 14
CLASS statement (LOGSELECT), 14
CLASS statement (MBC), 14
CLASS statement (NLMODE), 14
CLASS statement (PARTITION), 14
CLASS statement (PCA), 14
OUTCARD= option
PROC CARDINALITY statement, 1124
OUTCP= option
PROC VARREDUCE statement, 1172
OUTCUM option
TABLES statement (FREQTAB), 169
OUTDETAILS= option
PROC CARDINALITY statement, 1125
OUTEXPECT option
TABLES statement (FREQTAB), 169
OUTLEVEL option (BINOMIAL)
TABLES statement (FREQTAB), 157
OUTLEVELBINMAP= option
OUTPUT statement, 1104
CLASS statement (PHSELECT), 14
CLASS statement (PLSMOD), 14
CLASS statement (QTRSELECT), 14
CLASS statement (REGSELECT), 14
CLASS statement (TREESPLIT), 14
CLASS statement (VARIMPUTE), 14
CLASS statement (VARREDUCE), 14
PARAMETERS statement
NLMOD procedure, 685
PARMS statement
LMIXED procedure, 511
PARMSDATA= option
PARMS statement (LMIXED), 513
PARPREFIX= option
PROC PCA statement, 717
PARTFIT option
PROC GENSELECT statement, 375
PROC LOGSELECT statement, 562
PARTIAL statement
PCA procedure, 722
PARTIND option
PROC PARTITION statement, 1140
PARTINDNAME= option
OUTPUT statement, 1143
PARTITION procedure, OUTPUT statement, 1143
COPYVARS= option, 1144
FOLDNAME= option, 1144
FREQNAME= option, 1144
OUT= option, 1144
PARTINDNAME= option, 1144
PARTITION procedure, PROC PARTITION statement, 1140
ABSCONV option, 44
ABSFCONV option, 44
ABSFRTOL option, 44
ABSIGCONV option, 44
ABSIGTOL option, 44
ABSTOL option, 44
ABSTOL option, 44
ABSXCONV option, 45
ABSXTOL option, 45
DATA= option, 1140
EVENT= option, 1141
EVENTPROP= option, 1141
FCONV option, 45
FCONV2 option, 45
FTOL option, 45
FTOL2 option, 45
GCONV option, 46
GCONV2 option, 46
GTOL option, 46
GTOL2 option, 46
KFOLD= option, 1141
MAXFUNC= option, 47
MAXITER= option, 47
MAXTIME= option, 47
MINITER= option, 47
NORMALIZE= option, 48
NTHREADS= option, 1140
PARTIND option, 1140
SAMPPCT2= option, 1141
SAMPPCT= option, 1141
SAMPPCTEVT= option, 1141
SEED= option, 1140
TECHNIQUE= option, 48
DEGREE option (spline), 29
DETAILS option (multimember), 25
DETAILS option (polynomial), 26
DETAILS option (spline), 29
KNOTMAX= option (spline), 29
KNOTMETHOD option (spline), 29
KNOTMIN= option (spline), 30
MDEGREE option (polynomial), 26
NATURALCUBIC option (spline), 30
NOEFFECT option (multimember), 25
NOSEPARATE option (polynomial), 26
SEPARATE option (spline), 30
SPLIT option (spline), 31
STANDARDIZE option (polynomial), 26
<table>
<thead>
<tr>
<th>Option</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>XCONV option</td>
<td>48</td>
</tr>
<tr>
<td>XTOL option</td>
<td>48</td>
</tr>
<tr>
<td>PARTITION procedure, SELECTION statement</td>
<td></td>
</tr>
<tr>
<td>ADAPTIVE option</td>
<td>38</td>
</tr>
<tr>
<td>CHOOSE= option</td>
<td>38</td>
</tr>
<tr>
<td>COMPETITIVE option</td>
<td>38</td>
</tr>
<tr>
<td>CRITERION= option</td>
<td>38</td>
</tr>
<tr>
<td>DETAILS= option</td>
<td>40, 41</td>
</tr>
<tr>
<td>FAST option</td>
<td>39</td>
</tr>
<tr>
<td>HIERARCHY= option</td>
<td>41</td>
</tr>
<tr>
<td>LSCOEFFS option</td>
<td>39</td>
</tr>
<tr>
<td>MAXEFFECTS= option</td>
<td>39</td>
</tr>
<tr>
<td>MAXSTEPS= option</td>
<td>39</td>
</tr>
<tr>
<td>METHOD= option</td>
<td>37</td>
</tr>
<tr>
<td>MINEFFECTS= option</td>
<td>39</td>
</tr>
<tr>
<td>ORDERSELECT option</td>
<td>41</td>
</tr>
<tr>
<td>SELECT= option</td>
<td>39</td>
</tr>
<tr>
<td>SELECTION= option</td>
<td>43</td>
</tr>
<tr>
<td>SLE= option</td>
<td>39</td>
</tr>
<tr>
<td>SENTRY= option</td>
<td>39</td>
</tr>
<tr>
<td>SLS= option</td>
<td>39</td>
</tr>
<tr>
<td>SLSTAY= option</td>
<td>39</td>
</tr>
<tr>
<td>STOP= option</td>
<td>40</td>
</tr>
<tr>
<td>STOPHORIZON= option</td>
<td>43</td>
</tr>
<tr>
<td>PARTITION procedure, syntax</td>
<td>1140</td>
</tr>
<tr>
<td>PARTITION statement</td>
<td></td>
</tr>
<tr>
<td>GENSELECT procedure</td>
<td>36, 391</td>
</tr>
<tr>
<td>LOGSELECT procedure</td>
<td>36, 577</td>
</tr>
<tr>
<td>PHSELECT procedure</td>
<td>36, 764</td>
</tr>
<tr>
<td>PLSMOD procedure</td>
<td>810</td>
</tr>
<tr>
<td>QTRSELECT procedure</td>
<td>36, 849</td>
</tr>
<tr>
<td>REGSELECT procedure</td>
<td>36, 903</td>
</tr>
<tr>
<td>TREESPLIT procedure</td>
<td>36, 1029</td>
</tr>
<tr>
<td>PCA procedure</td>
<td>712</td>
</tr>
<tr>
<td>PROC PCA statement</td>
<td>712</td>
</tr>
<tr>
<td>PCA procedure, CLASS statement</td>
<td></td>
</tr>
<tr>
<td>DESCENDING option</td>
<td>13, 1173</td>
</tr>
<tr>
<td>MISSING option</td>
<td>13, 1173</td>
</tr>
<tr>
<td>ORDER= option</td>
<td>13</td>
</tr>
<tr>
<td>PARAM= option</td>
<td>14</td>
</tr>
<tr>
<td>REF= option</td>
<td>15</td>
</tr>
<tr>
<td>SPLIT option</td>
<td>15</td>
</tr>
<tr>
<td>PCA procedure, CODE statement</td>
<td>718</td>
</tr>
<tr>
<td>COMMENT option</td>
<td>17</td>
</tr>
<tr>
<td>FILE= option</td>
<td>17</td>
</tr>
<tr>
<td>FORMATWIDTH= option</td>
<td>17</td>
</tr>
<tr>
<td>INDENTSIZE= option</td>
<td>17</td>
</tr>
<tr>
<td>LABELID= option</td>
<td>18</td>
</tr>
<tr>
<td>LINESIZE= option</td>
<td>18</td>
</tr>
<tr>
<td>NOTRIM option</td>
<td>18</td>
</tr>
<tr>
<td>OUT= option</td>
<td>18</td>
</tr>
<tr>
<td>PCA procedure, DISPLAY statement</td>
<td></td>
</tr>
<tr>
<td>CASESENSITIVE option</td>
<td>20, 719</td>
</tr>
</tbody>
</table>
**Syntax Index**

- **MINITER= option**, 47
- **N= option**, 715
- **NOINT option**, 715
- **NORMALIZE= option**, 48
- **OUTSTAT= option**, 715
- **PARPREFIX= option**, 717
- **PLOTS= option**, 715
- **PREFIX= option**, 717
- **RPREFIX= option**, 717
- **SING= option**, 717
- **SINGULAR= option**, 717
- **STANDARD option**, 717
- **STD option**, 717
- **TECHNIQUE= option**, 48
- **VARDEF= option**, 717
- **XCONV option**, 48
- **XTOL option**, 48

**PCA procedure, PROC PCA statement**
- **METHOD=ITERGS option**
  - **EPSILON= option**, 713
  - **MAXITER= option**, 713
  - **NOCENTER option**, 714
  - **NOSCALE option**, 714

**PCA procedure, PROC PCA statement**
- **METHOD=NIPALS option**
  - **EPSILON= option**, 714
  - **MAXITER= option**, 714
  - **NOCENTER option**, 714
  - **NOSCALE option**, 714

**PCA procedure, PROC PCA statement**
- **METHOD=RANDOM option**
  - **NITER= option**, 714
  - **NOCENTER option**, 714
  - **NOSCALE option**, 714

**PCA procedure, SELECTION statement**
- **ADAPTIVE option**, 38
- **CHOOSE= option**, 38
- **COMPETITIVE option**, 38
- **CRITERION= option**, 38
- **DETAILS= option**, 40, 41
- **FAST option**, 39
- **HIERARCHY= option**, 41
- **LSCOEFFS option**, 39
- **MAXEFFECTS= option**, 39
- **MAXSTEPS= option**, 39
- **METHOD= option**, 37
- **MINEFFECTS= option**, 39
- **ORDERSELECT option**, 41
- **SELECT= option**, 39
- **SELECTION= option**, 43
- **SLE= option**, 39
- **SLENTRY= option**, 39

**SLS= option**, 39
**SLSTAY= option**, 39
**STOP= option**, 40
**STOPHORIZON= option**, 43

**PCA procedure, VAR statement**, 722

**PCATALL option**
- **CODE statement (GENSELECT)**, 18
- **CODE statement (LOGSELECT)**, 18

**PCHART statement**
- **PCHART procedure**, 944

**PCHI option**
- **EXACT statement (FREQTAB)**, 131
- **OUTPUT statement (FREQTAB)**, 144

**PCORR option**
- **EXACT statement (FREQTAB)**, 131
- **OUTPUT statement (FREQTAB)**, 144
- **TEST statement (FREQTAB)**, 195

**PCTLDEF= option**
- chart statements (SPC), 952

**PDATA= option**
- **PARMS statement (LMIXED)**, 513

**PEARSONRES option (CROSSLIST)**
- **TABLES statement (FREQTAB)**, 164

**PEVENT= option**
- **FITSTAT statement**, 1079

**PFFORMAT= option**
- **EXACT statement (FREQTAB)**, 135

**PHI= option**
- **MODEL statement (GENSELECT)**, 386

**PHSELECT procedure, CLASS statement**, 754
- **DESCENDING option**, 13, 1173
- **MISSING option**, 13, 1173
- **ORDER= option**, 13
- **PARAM= option**, 14
- **REF= option**, 15
- **SPLIT option**, 15

**PHSELECT procedure, CODE statement**, 754
- **COMMENT option**, 17
- **CUMHAZ option**, 17, 755
- **FILE= option**, 17, 755
- **FORMATWIDTH= option**, 17
- **INDENTSIZE= option**, 17
- **LABELID= option**, 18
- **LINESIZE= option**, 18
- **NOSURVIVAL option**, 18, 755
- **NOTRIM option**, 18
- **OUT= option**, 18
- **SHOWTIME option**, 18, 755
- **TIME= option**, 755
- **TIMEPOINT= option**, 18, 755

**PHSELECT procedure, DISPLAY statement**
- **CASESENSITIVE option**, 20, 757
- **EXCLUDE option**, 20, 757
- **EXCLUDEALL option**, 20, 757
TRACE option, 20, 757

PHSELECT procedure, DISPLAYOUT statement
  INCLUDEALL option, 21, 758
  NOREPLACE option, 21, 758
  REPEATED option, 21, 758

PHSELECT procedure, EFFECT statement
  BASIS option (spline), 28
  DATABOUNDARY option (spline), 28
  DEGREE option (polynomial), 26
  DEGREE option (spline), 29
  DETAILS option (multimember), 25
  DETAILS option (polynomial), 26
  DETAILS option (spline), 29
  KNOTMAX= option (spline), 29
  KNOTMETHOD option (spline), 29
  KNOTMIN= option (spline), 30
  MDEGREE option (polynomial), 26
  NATURALCUBIC option (spline), 30
  NOEFFECT option (multimember), 25
  NOEFFECT option (polynomial), 26
  SEPARATE option (polynomial), 30
  SPLIT option (spline), 31
  STANDARDIZE option (polynomial), 26

PHSELECT procedure, MODEL statement, 760
  CLB option, 761
  ENTRY= option, 761
  INCLUDE= option, 761
  INFORMATIVE option, 761
  OFFSET= option, 761
  START= option, 761
  TYPE3 option, 761

PHSELECT procedure, OUTPUT statement, 761
  COPYVAR= option, 762
  CUMHAZ option, 763
  DFBETA option, 763
  LD option, 763
  OUT= option, 762
  RESDEV option, 763
  RESMART option, 763
  RESSCH option, 763
  RESSCO option, 764
  ROLE option, 764
  STDERR option, 764
  RESRCH option, 764
  SURVIVAL option, 764
  WTRESSCH option, 764
  XBETA option, 764

PHSELECT procedure, PARTITION statement
  FRACTION option, 36, 765
  ROLEVAR= option, 36, 765

PHSELECT procedure, PROC PHSELECT statement, 751
  ABSCONV option, 44
  ABSFCONV option, 44
  ABSFTOL option, 44
PLS procedure, DISPLAY statement
CASESENSITIVE option, 20, 804
EXCLUDE option, 20, 805
EXCLUDEALL option, 20, 805
TRACE option, 20, 805
PLS procedure, DISPLAYOUT statement
INCLUDEALL option, 21, 805
NOREPLACE option, 21, 805
REPEATED option, 21, 805
PLS procedure, EFFECT statement
BASIS option (spline), 28
DATABOUNDARY option (spline), 28
DEGREE option (polynomial), 26
DEGREE option (spline), 29
DETAILS option (multimember), 25
DETAILS option (polynomial), 26
DETAILS option (spline), 29
KNOTMAX= option (spline), 29
KNOTMETHOD option (spline), 29
KNOTMIN= option (spline), 30
MDEGREE option (polynomial), 26
NATURALCUBIC option (spline), 30
NOEFFECT option (multimember), 25
NOSEPARATE option (polynomial), 26
SEPARATE option (spline), 30
SPLIT option (spline), 31
STANDARDIZE option (polynomial), 26
PLS procedure, MODEL statement, 807
INTERCEPT option, 807
SOLUTION option, 807
PLS procedure, OUTPUT statement, 807
COPYVARS= option, 808
keyword option, 808
OUT= option, 808
PLS procedure, PARTITION statement, 810
FRACTION option, 810
ROLEVAR= option, 811
PLS procedure, PROC PLS statement, 800
ABSCONV option, 44
ABSFCONV option, 44
ABSFRTOL option, 44
ABSGCONV option, 44
ABSGTOL option, 44
ABSTOL option, 44
ABSTOL option, 44
ABSTOL option, 44
ABSXCONV option, 45
ABSXTOL option, 45
CENSCALE option, 801
CVTEST option, 801
DATA= option, 801
DETAILS option, 802
FCONV option, 45
FCONV2 option, 45
FTOL option, 45
FTOL2 option, 45
Syntax Index

GCONV option, 46
GCONV2 option, 46
GTOL option, 46
GTOL2 option, 46
MAXFUNC= option, 47
MAXITER= option, 47
MAXTIME= option, 47
METHOD= option, 802
MINITER= option, 47
NFAC= option, 802
NOCENTER option, 803
NOCOMP= option, 803
NORMALIZE= option, 48
NOSCALE option, 803
TECHNIQUE= option, 48
VARSS option, 803
XTOL option, 48

PLSMOD procedure, PROC PLSMOD statement,
CVTEST option
NSAMP= option, 801
PVALUE= option, 801
SEED= option, 801
STAT= option, 801

PLSMOD procedure, PROC PLSMOD statement,
METHOD=PLS option
ALGORITHM= option, 802
EPSILON= option, 802
MAXITER= option, 802

PLSMOD procedure, SELECTION statement
ADAPTIVE option, 38
CHOOSE= option, 38
COMPETITIVE option, 38
CRITERION= option, 38
DETAILS= option, 40, 41
FAST option, 39
HIERARCHY= option, 41
LSQOEFF option, 39
MAXEFFECTS= option, 39
MAXSTEPS= option, 39
METHOD= option, 37
MINEFFECTS= option, 39
ORDERSELECT option, 41
SELECT= option, 39
SELECTION= option, 43
SLE= option, 39
SLENTRY= option, 39
SLS= option, 39
SLSTAY= option, 39
STOP= option, 40
STOPHORIZON= option, 43

POINT option
EXACT statement (FREQTAB), 135

POLYCHORIC option
TABLES statement (FREQTAB), 170

POPSIZE= option
AUTOTUNE statement, 1021, 1022

POST= option
OUTPUT statement (LOGSELECT), 575

PPREFIX= option
PROC PCA statement, 717

PRED= option
PREDICT statement (NLMODE), 689
PREDICTED= option
OUTPUT statement (GENSELECT), 389
OUTPUT statement (LOGSELECT), 575

PREDEPROBS option
OUTPUT statement (LOGSELECT), 573

PREFIX= option
OUTPUT statement (OUTDESIGN), 664
PROC ICA statement, 437
PROC PCA statement, 717

PRINTALL option (RELRISK)
TABLES statement (FREQTAB), 187

PRINTALLDISTANCES option
PROC KCLUS statement, 462

PRINTKWTS option
TABLES statement (FREQTAB), 152, 184

PRINTTARGET option
PROC TREESPLIT statement, 1018

PRINTWTS option (COMMONRISKDIFF)
TABLES statement (FREQTAB), 162

PRIOR= option
MODEL statement (LOGSELECT), 571

PROBT= option
PREDICT statement (NLMODE), 689

PROC ASSESS statement, see ASSESS procedure
PROC ASSESS statement, see ASSESS procedure
PROC BINNING statement, see BINNING procedure
PROC CARDINALITY statement, see CARDINALITY procedure
PROC CORRELATION statement, 94, see CORRELATION procedure
PROC CORRELATION statement, 94
PROC FREQTAB statement, see FREQTAB procedure
PROC GAMMOD statement, see GAMMOD procedure
PROC GENSELECT statement, see GENSELECT procedure
PROC ICA statement, see ICA procedure
ICA procedure, 435
PROC KCLUS statement, see KCLUS procedure
PROC LMIXED statement
LMIXED procedure, 496
PROC LOGSELECT statement, see LOGSELECT procedure
PROC MODELMATRIX statement, see MODELMATRIX procedure
MODELMATRIX procedure, 658
PROC NLMOD statement, see NLMOD procedure
NLMOD procedure, 680
PROC PARTITION statement, see PARTITION procedure
PROC PCA statement, see PCA procedure
PCA procedure, 712
PROC PHSELECT statement, see PHSELECT procedure
PROC PLSMOD statement, see PLSMOD procedure
PLSMOD procedure, 800
PROC QTRSELECT statement, see QTRSELECT procedure
QTRSELECT procedure, 839
PROC REGSELECT statement, see REGSELECT procedure
REGSELECT procedure, 893
PROC SPC statement, 938, see SPC procedure
PROC TREESPLIT statement, see TREESPLIT procedure
TREESPLIT procedure, 1011
PROC VARIMPUTE statement, see VARIMPUTE procedure
PROC VARRDUCED statement, see VARRDUCED procedure
PROCESSNAME= option
PROC SPC statement, 939
PROCESVALUE= option
PROC SPC statement, 939
PROC MBC statement, see MBC procedure
PRUNE statement
TREESPLIT procedure, 1030
PRUNINGTABLE option
PROC TREESPLIT statement, 1018
PVAL= option
PROC PLSMOD statement, CVTEST option, 801
PVAR= option
FITSTAT statement, 1079
QOR option (CMH)
TABLES statement (FREQTAB), 161
QTRSELECT procedure
MODEL statement, 845
OUTPUT statement, 847
PROC QTRSELECT statement, 839
QTRSELECT procedure, CLASS statement, 840
DESCENDING option, 13, 1173
MISSING option, 13, 1173
ORDER= option, 13
PARAM= option, 14
REF= option, 15
SPLIT option, 15
QTRSELECT procedure, CODE statement, 841
COMMENT option, 17
FILE= option, 17
FORMATWIDTH= option, 17
INDENTSIZE= option, 17
LABELID= option, 18
LINESIZE= option, 18
NOTRIM option, 18
OUT= option, 18
QTRSELECT procedure, DISPLAY statement
CASESENSITIVE option, 20, 842
EXCLUDE option, 20, 842
EXCLUDEALL option, 20, 842
TRACE option, 20, 842
QTRSELECT procedure, DISPLAYOUT statement
INCLUDEALL option, 21, 843
NOREPLACE option, 21, 843
REPEATED option, 21, 843
QTRSELECT procedure, EFFECT statement
BASIS option (spline), 28
DATABOUNDARY option (spline), 28
DEGREE option (polynomial), 26
DEGREE option (spline), 29
DETAILS option (multimember), 25
DETAILS option (polynomial), 26
DETAILS option (spline), 29
KNOTMAX= option (spline), 29
KNOTMETHOD option (spline), 29
KNOTMIN= option (spline), 30
MDEGREE option (polynomial), 26
NATURALCUBIC option (spline), 30
NOEFFECT option (multimember), 25
NOSEPARATE option (polynomial), 26
SEPARATE option (spline), 30
SPLIT option (spline), 31
STANDARDIZE option (polynomial), 26
QTRSELECT procedure, MODEL statement, 845
CLB option, 846
INCLUDE option, 846
INFORMATIVE option, 846
NOINT option, 846
QUANTILES option, 846
START option, 847
STB option, 847
QTRSELECT procedure, MODEL statement,
QUANTILES option
NTAU option, 846
SORT option, 846
QTRSELECT procedure, OUTPUT statement, 847
COPYVARIABLE option, 848
keyword= option, 848
OUT= option, 847
QTRSELECT procedure, PARTITION statement
FRACTION option, 36, 850
ROLEVAR= option, 36, 850
QTRSELECT procedure, PROC QTRSELECT
  statement, 839
  ABSCONV option, 44
  ABSFCONV option, 44
  ABSFTOL option, 44
  ABSGCONV option, 44
  ABSGTOL option, 44
  ABSTOL option, 44
  ABSXCONV option, 45
  ABSXTOL option, 45
  ALPHA= option, 840
  COV=SPARSITY option, 840
  DATA= option, 840
  FCONV option, 45
  FCONV2 option, 45
  FTOL option, 45
  FTOL2 option, 45
  GCONV option, 46
  GCONV2 option, 46
  GTOL option, 46
  GTOL2 option, 46
  MAXFUNC= option, 47
  MAXITER= option, 47
  MAXTIME= option, 47
  MINITER= option, 47
  NOCLPRINT option, 840
  NORMALIZE= option, 48
  TECHNIQUE= option, 48
  XCONV option, 48
  XTOL option, 48
QTRSELECT procedure, SELECTION statement, 850
  ADAPTIVE option, 38
  CHOOSE= option, 38
  COMPETITIVE option, 38
  CRITERION= option, 38
  DETAILS= option, 40, 41
  FAST option, 39
  HIERARCHY= option, 41
  LS COEFFS option, 39
  MAX EFFECTS= option, 39
  MAX STEPS= option, 39
  METHOD= option, 37
  MINEFFECTS= option, 39
  ORDERSELECT option, 41
  SELECT= option, 39
  SELECTION= option, 43
  SLE= option, 39
  SLENTRY= option, 39
  SLS= option, 39
  SLSTAY= option, 39
  STOP= option, 40
  STOPHORIZON= option, 43
  QUANTILES option
  MODEL statement (QTRSELECT), 846
  R option
    REPEATED statement (LMIXED), 524
  RANDOM statement
    L MIXED procedure, 514
  RANK option
    PROC CORRELATION statement, 96
  RANKS option
    PROC L MIXED statement, 499
  RBAIMP option
    PROC TREESPLIT statement, 1018
  RC option
    REPEATED statement (LMIXED), 524
  RCHART statement
    RCHART procedure, 945
  RCI option
    REPEATED statement (LMIXED), 524
  RCORR option
    REPEATED statement (LMIXED), 524
  RDIF1 option
    OUTPUT statement (FREQTAB), 145
  RDIF2 option
    OUTPUT statement (FREQTAB), 145
  REDUCE statement
    VARREDUCE procedure, 1176
  REDUCEDERROR option
    PRUNE statement (TREESPLIT), 1031
  REF= option
    CLASS statement (ASSESS), 15
    CLASS statement (BINNING), 15
    CLASS statement (CARDINALITY), 15
    CLASS statement (CORRELATION), 15
    CLASS statement (GAMMOD), 15
    CLASS statement (GENSELECT), 15
    CLASS statement (ICA), 15
    CLASS statement (KCLUS), 15
    CLASS statement (L MIXED), 15
    CLASS statement (LOGSELECT), 15
    CLASS statement (MBC), 15
    CLASS statement (NLMOD), 15
    CLASS statement (PARTITION), 15
    CLASS statement (PCA), 15
    CLASS statement (PHSELECT), 15
    CLASS statement (PLS MOD), 15
    CLASS statement (QTRSELECT), 15
    CLASS statement (REGSELECT), 15
    CLASS statement (TREESPLIT), 15
    CLASS statement (V ARIMPUTE), 15
    CLASS statement (V ARREDUCE), 15
REGSELECT procedure
  MODEL statement, 899
  OUTPUT statement, 901
  PROC REGSELECT statement, 893
REGSELECT procedure, CLASS statement, 894
DESCENDING option, 13, 1173
MISSING option, 13, 1173
ORDER= option, 13
PARAM= option, 14
REF= option, 15
SPLIT option, 15
REGSELECT procedure, CODE statement, 895
COMMENT option, 17
FILE= option, 17
FORMATWIDTH= option, 17
INDENTSIZE= option, 18
LABELID= option, 18
LINESIZE= option, 18
NOTRIM option, 18
OUT= option, 18
REGSELECT procedure, DISPLAY statement
CASESENSITIVE option, 20, 896
EXCLUDE option, 20, 896
EXCLUDEALL option, 20, 896
TRACE option, 20, 896
REGSELECT procedure, DISPLAYOUT statement
INCLUDEALL option, 21, 897
NOREPLACE option, 21, 897
REPEATED option, 21, 897
REGSELECT procedure, EFFECT statement
BASIS option (spline), 28
DATABOUNDARY option (spline), 28
DEGREE option (polynomial), 26
DEGREE option (spline), 29
DETAILS option (multimember), 25
DETAILS option (polynomial), 26
DETAILS option (spline), 29
KNOTMAX= option (spline), 29
KNOTMETHOD option (spline), 29
KNOTMIN= option (spline), 30
MDEGREE option (polynomial), 26
NATURALCUBIC option (spline), 30
NOEFFECT option (multimember), 25
NOSEPARATE option (polynomial), 26
SEPARATE option (spline), 30
SPLIT option (spline), 31
STANDARDIZE option (polynomial), 26
REGSELECT procedure, MODEL statement, 899
CLB option, 900
INCLUDE option, 900
INFORMATIVE option, 900
NOINT option, 900
SS3 option, 900
START option, 900
STB option, 900
TOL option, 901
VIF option, 901
REGSELECT procedure, OUTPUT statement, 901
COPYVAR= option, 901
keyword= option, 901
OUT= option, 901
REGSELECT procedure, PARTITION statement
FRACTION option, 36, 903
ROLEVAR= option, 36, 904
REGSELECT procedure, PROC REGSELECT statement, 893
ABSCONV option, 44
ABSFCONV option, 44
ABSFTOL option, 44
ABSGCONV option, 44
ABSGTOL option, 44
ABSTOL option, 44
ABSXCONV option, 44
ABSXTOL option, 44
ALPHA= option, 894
DATA= option, 894
FCONV option, 45
FCONV2 option, 45
FTOL option, 45
FTOL2 option, 45
GCONV option, 46
GCONV2 option, 46
GTOL option, 46
GTOL2 option, 46
MAXFUNC= option, 47
MAXITER= option, 47
MAXTIME= option, 47
MINITER= option, 47
NOCLPRINT option, 894
NORMALIZE= option, 48
TECHNIQUE= option, 48
XCONV option, 48
XTOL option, 48
REGSELECT procedure, SELECTION statement, 904
ADAPTIVE option, 38
CHOOSE= option, 38
COMPETITIVE option, 38
CRITERION= option, 38
DETAILS= option, 40, 41
FAST option, 39
HIERARCHY= option, 41
LSCOEFFS option, 39
MAXEFFECTS= option, 39
MAXSTEPS= option, 39
METHOD= option, 37
MINEFFECTS= option, 39
ORDERSELECT option, 41
SELECT= option, 39
SELECTION= option, 43
SLE= option, 39
SLENTRY= option, 39
SLS= option, 39
SLSTAY= option, 39
STOP= option, 40
STOPHORIZON= option, 43
RELRISK option
  EXACT statement (FREQTAB), 131
  OUTPUT statement (FREQTAB), 145
  TABLES statement (FREQTAB), 184
REPEATED option
  DISPLAYOUT statement (CORRELATION), 21, 98
  DISPLAYOUT statement (GAMMOD), 21, 320
  DISPLAYOUT statement (GENSELECT), 21, 378
  DISPLAYOUT statement (ICA), 21, 439
  DISPLAYOUT statement (KCLUS), 21, 464
  DISPLAYOUT statement (LMIXED), 21, 503
  DISPLAYOUT statement (LOGSELECT), 21, 566
  DISPLAYOUT statement (MBC), 21, 631
  DISPLAYOUT statement (MODELMATRIX), 660
  DISPLAYOUT statement (NLMOD), 21, 684
  DISPLAYOUT statement (PARTITION), 21, 1143
  DISPLAYOUT statement (PCA), 21, 720
  DISPLAYOUT statement (PHSELECT), 21, 758
  DISPLAYOUT statement (PLSMOD), 21, 805
  DISPLAYOUT statement (QTRSELECT), 21, 843
  DISPLAYOUT statement (REGSELECT), 21, 897
  DISPLAYOUT statement (VARREDUCE), 21, 1175
REPEATED statement
  LMIXED procedure, 523
RESCHI= option
  OUTPUT statement (GENSELECT), 389
  OUTPUT statement (LOGSELECT), 576
RESDEV option
  OUTPUT statement (PHSELECT), 763
RESDEV= option
  OUTPUT statement (GENSELECT), 390
  OUTPUT statement (LOGSELECT), 576
RESIDUAL= option
  OUTPUT statement (GENSELECT), 390
  OUTPUT statement (LOGSELECT), 576
RESLIK= option
  OUTPUT statement (GENSELECT), 390
  OUTPUT statement (LOGSELECT), 576
RESMART option
  OUTPUT statement (PHSELECT), 763
RESRAW= option
  OUTPUT statement (GENSELECT), 390
  OUTPUT statement (LOGSELECT), 576
RESSCH option
  OUTPUT statement (PHSELECT), 763
RESSCO option
  OUTPUT statement (PHSELECT), 764
RESTRICT statement
  NLMOD procedure, 689
RESWORK= option
  OUTPUT statement (GENSELECT), 390
  OUTPUT statement (LOGSELECT), 576
RI option
  REPEATED statement (LMIXED), 524
RIDGE= option
  MODEL statement (GAMMOD), 328
RISK1 option
  OUTPUT statement (FREQTAB), 145
RISK11 option
  OUTPUT statement (FREQTAB), 146
RISK12 option
  OUTPUT statement (FREQTAB), 146
RISK2 option
  OUTPUT statement (FREQTAB), 146
RISK21 option
  OUTPUT statement (FREQTAB), 146
RISK22 option
  OUTPUT statement (FREQTAB), 146
RISKDIFF option
  EXACT statement (FREQTAB), 132
  OUTPUT statement (FREQTAB), 145
  TABLES statement (FREQTAB), 187
RISKDIFF1 option
  OUTPUT statement (FREQTAB), 145
RISKDIFF2 option
  OUTPUT statement (FREQTAB), 145
ROCOUT= option
  PROC ASSESS statement, 1078
ROLE option
  OUTPUT statement, 1029
  OUTPUT statement (PHSELECT), 764
ROLE= option
  LD statement (PHSELECT), 763
  OUTPUT statement (GENSELECT), 390
  OUTPUT statement (LOGSELECT), 576
ROLEVAR= option
  PARTITION statement (GENSELECT), 36, 391
  PARTITION statement (LOGSELECT), 36, 577
  PARTITION statement (PHSELECT), 36, 765
  PARTITION statement (QTRSELECT), 36, 850
  PARTITION statement (REGSELECT), 36, 904
  PARTITION statement (TREESPLIT), 36, 1030
  PLSMOD procedure, PARTITION statement, 811
RPREFIX= option
  PROC PCA statement, 717
RRC1 option
  OUTPUT statement (FREQTAB), 145
<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RRC2 option</td>
<td>OUTPUT statement (FREQTAB), 145</td>
</tr>
<tr>
<td>RSS option</td>
<td>GROW statement (TREESPLIT), 1028</td>
</tr>
<tr>
<td>SAMPLESIZE= option</td>
<td>AUTOTUNE statement, 1022</td>
</tr>
<tr>
<td>SAMPPCT2= option</td>
<td>PROC PARTITION statement, 1141</td>
</tr>
<tr>
<td>SAMPPCT= option</td>
<td>PROC PARTITION statement, 1141</td>
</tr>
<tr>
<td>SAMPPCTEVT= option</td>
<td>PROC PARTITION statement, 1141</td>
</tr>
<tr>
<td>SCALE= option</td>
<td>MODEL statement (GAMMOD), 328</td>
</tr>
<tr>
<td>SCHART statement</td>
<td>SCHART procedure, 946</td>
</tr>
<tr>
<td>SCORE statement</td>
<td>KCLUS procedure, 465</td>
</tr>
<tr>
<td>SCORES= option</td>
<td>TABLES statement (FREQTAB), 192</td>
</tr>
<tr>
<td>SCOROUT option</td>
<td>TABLES statement (FREQTAB), 193</td>
</tr>
<tr>
<td>SCORR option</td>
<td>EXACT statement (FREQTAB), 133</td>
</tr>
<tr>
<td>SEARCHMETHOD= option</td>
<td>AUTOTUNE statement, 1022</td>
</tr>
<tr>
<td>SECONDOBJECTIVE= option</td>
<td>AUTOTUNE statement, 1023</td>
</tr>
<tr>
<td>SEED= option</td>
<td>EXACT statement (FREQTAB), 136</td>
</tr>
<tr>
<td>PROC GAMMOD statement</td>
<td>PROC GAMMOD statement, 316</td>
</tr>
<tr>
<td>PROC ICA statement</td>
<td>PROC ICA statement, 437</td>
</tr>
<tr>
<td>PROC KCLUS statement</td>
<td>PROC KCLUS statement, 461</td>
</tr>
<tr>
<td>PROC PARTITION statement</td>
<td>PROC PARTITION statement, 1140</td>
</tr>
<tr>
<td>PROC PCA statement</td>
<td>PROC PCA statement, METHOD=RANDOM option, 714</td>
</tr>
<tr>
<td>PROC PLSMOD statement</td>
<td>PROC PLSMOD statement, CVTEST option, 801</td>
</tr>
<tr>
<td>PROC TREESPLIT statement</td>
<td>PROC TREESPLIT statement, 1018</td>
</tr>
<tr>
<td>PROC VARIMPUTE statement</td>
<td>PROC VARIMPUTE statement, 1157</td>
</tr>
<tr>
<td>PROC MBC statement</td>
<td>PROC MBC statement, 628</td>
</tr>
<tr>
<td>SELECT= option</td>
<td>SELECTION statement (ASSESS), 39</td>
</tr>
<tr>
<td>SELECTION statement (ASSESS)</td>
<td>PROC GAMMOD statement, 316</td>
</tr>
<tr>
<td>SELECTION statement (BINNING)</td>
<td>PROC ICA statement, 437</td>
</tr>
<tr>
<td>SELECTION statement (CARDINALITY)</td>
<td>PROC KCLUS statement, 461</td>
</tr>
<tr>
<td>SELECTION statement (CORRELATION)</td>
<td>PROC PARTITION statement, 1140</td>
</tr>
<tr>
<td>SELECTION statement (GAMMOD)</td>
<td>PROC PCA statement, METHOD=RANDOM option, 714</td>
</tr>
<tr>
<td>SELECTION statement (GENSELECT)</td>
<td>PROC PLSMOD statement, CVTEST option, 801</td>
</tr>
<tr>
<td>SELECTION statement (ICA)</td>
<td>PROC TREESPLIT statement, 1018</td>
</tr>
<tr>
<td>SELECTION statement (KCLUS)</td>
<td>PROC VARIMPUTE statement, 1157</td>
</tr>
<tr>
<td>SELECTION statement (LMIXED)</td>
<td>PROC MBC statement, 628</td>
</tr>
<tr>
<td>SELECTION statement (LOGSELECT)</td>
<td>SELECTION statement (ASSESS), 39</td>
</tr>
<tr>
<td>SELECTION statement (MBC)</td>
<td>PROC GAMMOD statement, 316</td>
</tr>
<tr>
<td>SELECTION statement (NLMOD)</td>
<td>PROC ICA statement, 437</td>
</tr>
<tr>
<td>SELECTION statement (PARTITION)</td>
<td>PROC KCLUS statement, 461</td>
</tr>
<tr>
<td>SELECTION statement (PCA)</td>
<td>PROC PCA statement, METHOD=RANDOM option, 714</td>
</tr>
<tr>
<td>SELECTION statement (PHSELECT)</td>
<td>PROC PLSMOD statement, CVTEST option, 801</td>
</tr>
<tr>
<td>SELECTION statement (PLSMOD)</td>
<td>PROC TREESPLIT statement, 1018</td>
</tr>
<tr>
<td>SELECTION statement (REGSELECT)</td>
<td>SELECTION statement (ASSESS), 39</td>
</tr>
<tr>
<td>SELECTION statement (TREESPLIT)</td>
<td>SELECTION statement (BINNING), 39</td>
</tr>
<tr>
<td>SELECTION statement (VARIMPUTE)</td>
<td>SELECTION statement (ASSESS), 39</td>
</tr>
</tbody>
</table>
Syntax Index

SELECTION statement (VARIABLE), 43
SENSPEC option
   TABLES statement (FREQTAB), 193
SEPARATE option
   EFFECT statement, spline (ASSESS), 30
   EFFECT statement, spline (BINNING), 30
   EFFECT statement, spline (CARDINALITY), 30
   EFFECT statement, spline (CORRELATION), 30
   EFFECT statement, spline (GAMMOD), 30
   EFFECT statement, spline (GENSELECT), 30
   EFFECT statement, spline (ICA), 30
   EFFECT statement, spline (KCLUS), 30
   EFFECT statement, spline (LMIXED), 30
   EFFECT statement, spline (LOGSELECT), 30
   EFFECT statement, spline (MBC), 30
   EFFECT statement, spline (NLMOD), 30
   EFFECT statement, spline (PARTITION), 30
   EFFECT statement, spline (PCA), 30
   EFFECT statement, spline (PHSELECT), 30
   EFFECT statement, spline (PLSMOD), 30
   EFFECT statement, spline (QTRSELECT), 30
   EFFECT statement, spline (REGSELECT), 30
   EFFECT statement, spline (REGSPLITE), 30
   EFFECT statement, spline (VARIMPUTE), 30
   EFFECT statement, spline (VARIABLE), 30
SHOWTIME option
   CODE statement (PHSELECT), 18, 755
SIGMAS= option
   chart statements (SPC), 952
SIMPLE option
   PROC LMIXED statement, 499
SING= option
   PROC PCA statement, 717
SINGCHOL= option
   PROC GAMMOD statement, 316
   PROC LMIXED statement, 499
SINGPARM= option
   PROC MBC statement, 628
SINGRES= option
   PROC LMIXED statement, 499
SINGULAR= option
   PROC GAMMOD statement, 317
   PROC LMIXED statement, 499
   PROC NLMOD statement, 682
   PROC PCA statement, 717
   PROC MBC statement, 628
SLE= option
   SELECTION statement (ASSESS), 39
   SELECTION statement (BINNING), 39
   SELECTION statement (CARDINALITY), 39
   SELECTION statement (CORRELATION), 39
   SELECTION statement (GAMMOD), 39
   SELECTION statement (GENSELECT), 39
   SELECTION statement (ICA), 39
   SELECTION statement (KCLUS), 39
   SELECTION statement (LMIXED), 39
   SELECTION statement (LOGSELECT), 39
   SELECTION statement (MBC), 39
   SELECTION statement (NLMMOD), 39
   SELECTION statement (PARTITION), 39
   SELECTION statement (PCA), 39
   SELECTION statement (PHSELECT), 39
   SELECTION statement (PLSMOD), 39
   SELECTION statement (QTRSELECT), 39
   SELECTION statement (REGSELECT), 39
   SELECTION statement (REGSPLITE), 39
   SELECTION statement (VARIMPUTE), 39
   SELECTION statement (VARIABLE), 39
SLENTRY= option
   SELECTION statement (ASSESS), 39
   SELECTION statement (BINNING), 39
   SELECTION statement (CARDINALITY), 39
   SELECTION statement (CORRELATION), 39
   SELECTION statement (GAMMOD), 39
   SELECTION statement (GENSELECT), 39
   SELECTION statement (ICA), 39
   SELECTION statement (KCLUS), 39
   SELECTION statement (LMIXED), 39
   SELECTION statement (LOGSELECT), 39
   SELECTION statement (MBC), 39
   SELECTION statement (NLMMOD), 39
   SELECTION statement (PARTITION), 39
   SELECTION statement (PCA), 39
   SELECTION statement (PHSELECT), 39
   SELECTION statement (PLSMOD), 39
   SELECTION statement (QTRSELECT), 39
   SELECTION statement (REGSELECT), 39
   SELECTION statement (REGSPLITE), 39
   SELECTION statement (VARIMPUTE), 39
   SELECTION statement (VARIABLE), 39
SLS= option
   SELECTION statement (ASSESS), 39
   SELECTION statement (BINNING), 39
   SELECTION statement (CARDINALITY), 39
   SELECTION statement (CORRELATION), 39
   SELECTION statement (GAMMOD), 39
   SELECTION statement (GENSELECT), 39
   SELECTION statement (ICA), 39
   SELECTION statement (KCLUS), 39
   SELECTION statement (LMIXED), 39
   SELECTION statement (LOGSELECT), 39
   SELECTION statement (MBC), 39
   SELECTION statement (NLMMOD), 39
   SELECTION statement (PARTITION), 39
   SELECTION statement (PCA), 39
   SELECTION statement (PHSELECT), 39
   SELECTION statement (PLSMOD), 39
   SELECTION statement (QTRSELECT), 39
   SELECTION statement (REGSELECT), 39
   SELECTION statement (REGSPLITE), 39
   SELECTION statement (VARIMPUTE), 39
   SELECTION statement (VARIABLE), 39
**Syntax Index**

**SELECTION statement (REGSELECT), 39**
**SELECTION statement (TREESPLIT), 39**
**SELECTION statement (VARIABLEMPUTE), 39**
**SELECTION statement (VARREDUCE), 39**

**SLSTAY= option**
**SELECTION statement (ASSESS), 39**
**SELECTION statement (BINNING), 39**
**SELECTION statement (CARDINALITY), 39**
**SELECTION statement (CORRELATION), 39**
**SELECTION statement (GAMMOD), 39**
**SELECTION statement (GENSELECT), 39**
**SELECTION statement (ICLUS), 39**
**SELECTION statement (KCLUS), 39**
**SELECTION statement (LMIXED), 39**
**SELECTION statement (LOGSELECT), 39**
**SELECTION statement (PARTITION), 39**
**SELECTION statement (PCA), 39**
**SELECTION statement (PHSELECT), 39**
**SELECTION statement (PLSMOD), 39**
**SELECTION statement (QTRSELECT), 39**
**SELECTION statement (REGSELECT), 39**
**SELECTION statement (TREESPLIT), 39**
**SELECTION statement (VARIABLEMPUTE), 39**
**SELECTION statement (VARREDUCE), 39**

**SMDCR option**
**EXACT statement (FREQTAB), 133**
**OUTPUT statement (FREQTAB), 146**
**TEST statement (FREQTAB), 196**

**SMDRC option**
**EXACT statement (FREQTAB), 133**
**OUTPUT statement (FREQTAB), 146**
**TEST statement (FREQTAB), 196**

**SMETHOD= option**
**chart statements (SPC), 952**

**SMOOTH= option**
**MODEL statement (GAMMOD), 325**

**SMOOCHOPTIONS option**
**PROC GAMMOD statement, 317**

**SOLUTION option**
**MODEL statement (LMIXED), 505**
**MODEL statement (PLSMOD), 807**
**RANDOM statement (LMIXED), 515**

**SORT option**
**QUANTILES option (QTRSELECT), 846**

**SPC procedure**
**LIMITN= data table, 981**
**SPECS= data table, 982**
**syntax, 937**

**SPC procedure, BOXCHART statement, 940**
**SPC procedure, CCHART statement, 940**
**LIMITN= option, 967**
**SIGMAS= option, 967**

**SPC procedure, chart statements, 949**
**LIMITN= option, 969**
**SIGMAS= option, 969**

**SPC procedure, IRCHART statement, 941**
**LIMITN= option, 960**
**SIGMAS= option, 960**

**SPC procedure, MCHART statement, 942**
**LIMITN= option, 963**
**MEDCENTRAL= option, 963**
**SIGMAS= option, 963**

**SPC procedure, MRCHART statement, 943**
**SPC procedure, NCHART statement, 943**
**LIMITN= option, 969**
**SIGMAS= option, 969**

**SPC procedure, PCHART statement, 944**
**LIMITN= option, 970**
**SIGMAS= option, 970**

**SPC procedure, PROC SPC statement, 938**
**DATA= option, 938, 980**
**NTHREADS= option, 939**
**PROCESSNAME= option, 939**
**PROCESSVALUE= option, 939**
**SUBGROUPNAME= option, 939**
**SUBGROUPVALUE= option, 939**

**SPC procedure, RCHART statement, 945**
**LIMITN= option, 964**
**SIGMAS= option, 964**

**SPC procedure, SCHART statement, 946**
**LIMITN= option, 965**
**SIGMAS= option, 965**

**SPC procedure, UCHART statement, 946**
**LIMITN= option, 972**
**SIGMAS= option, 972**
SPC procedure, XCHART statement, 947
   LIMITN= option, 961
   SIGMAS= option, 961
SPC procedure, XRCHART statement, 948
SPC procedure, XSCHART statement, 949
SPECS= option
   chart statements (SPC), 953
SPLIT option
   CLASS statement (ASSESS), 15
   CLASS statement (BINNING), 15
   CLASS statement (CARDINALITY), 15
   CLASS statement (CORRELATION), 15
   CLASS statement (GAMMOD), 15
   CLASS statement (GENSELECT), 15
   CLASS statement (ICA), 15
   CLASS statement (KCLUS), 15
   CLASS statement (LMixed), 15
   CLASS statement (LOGSELECT), 15
   CLASS statement (MBC), 15
   CLASS statement (NLMod), 15
   CLASS statement (PARTITION), 15
   CLASS statement (PCA), 15
   CLASS statement (PHSELECT), 15
   CLASS statement (PLSMOD), 15
   CLASS statement (QTRSELECT), 15
   CLASS statement (REGSELECT), 15
   CLASS statement (TREESPLIT), 15
   CLASS statement (V ARIMPUTE), 15
   CLASS statement (V ARREDUCE), 15
   EFFECT statement, spline (ASSESS), 31
   EFFECT statement, spline (BINNING), 31
   EFFECT statement, spline (CARDINALITY), 31
   EFFECT statement, spline (CORRELATION), 31
   EFFECT statement, spline (GAMMOD), 31
   EFFECT statement, spline (GENSELECT), 31
   EFFECT statement, spline (ICA), 31
   EFFECT statement, spline (KCLUS), 31
   EFFECT statement, spline (LMixed), 31
   EFFECT statement, spline (LOGSELECT), 31
   EFFECT statement, spline (MBC), 31
   EFFECT statement, spline (NLMod), 31
   EFFECT statement, spline (PARTITION), 31
   EFFECT statement, spline (PCA), 31
   EFFECT statement, spline (PHSELECT), 31
   EFFECT statement, spline (PLSMOD), 31
   EFFECT statement, spline (QTRSELECT), 31
   EFFECT statement, spline (REGSELECT), 31
   EFFECT statement, spline (TREESPLIT), 31
   EFFECT statement, spline (V ARIMPUTE), 31
   EFFECT statement, spline (V ARREDUCE), 31
SPLITONCE option
   PROC TREESPLIT statement, 1018
SS3 option
   MODEL statement (REGSELECT), 900
SSCP option
   PROC CORRELATION statement, 96
STANDARD option
   PROC PCA statement, 717
STANDARDIZE option
   EFFECT statement, polynomial (ASSESS), 26
   EFFECT statement, polynomial (BINNING), 26
   EFFECT statement, polynomial (CARDINALITY), 26
   EFFECT statement, polynomial (CORRELATION), 26
   EFFECT statement, polynomial (GAMMOD), 26
   EFFECT statement, polynomial (GENSELECT), 26
   EFFECT statement, polynomial (ICA), 26
   EFFECT statement, polynomial (KCLUS), 26
   EFFECT statement, polynomial (LMixed), 26
   EFFECT statement, polynomial (LOGSELECT), 26
   EFFECT statement, polynomial (MBC), 26
   EFFECT statement, polynomial (PLSMOD), 26
   EFFECT statement, polynomial (QTRSELECT), 26
   EFFECT statement, polynomial (REGSELECT), 26
   EFFECT statement, polynomial (TREESPLIT), 26
   EFFECT statement, polynomial (V ARIMPUTE), 26
   EFFECT statement, polynomial (V ARREDUCE), 26
STANDARDIZE= option
   PROC KCLUS statement, 461
START option
   MODEL statement (GENSELECT), 386
   MODEL statement (LOGSELECT), 572
   MODEL statement (QTRSELECT), 847
   MODEL statement (REGSELECT), 900
START= option
   MODEL statement (PHSELECT), 761
STAT= option
   PROC PLSMOD statement, CVTEST option, 801
STB option
   MODEL statement (QTRSELECT), 847
   MODEL statement (REGSELECT), 900
   PROC GENSELECT statement, 375
   PROC LOGSELECT statement, 562
STD option
   PROC PCA statement, 717
STDERR= option
   PREDICT statement (NLMOD), 689
STDRES option (CROSSLIST)
   TABLES statement (FREQTAB), 164
STDRESCHI= option
   OUTPUT statement (GENSELECT), 390
   OUTPUT statement (LOGSELECT), 576
STDRESDEV= option
   OUTPUT statement (GENSELECT), 390
   OUTPUT statement (LOGSELECT), 576
STDXBETA option
   OUTPUT statement (PHSELECT), 764
STDXBETA= option
   OUTPUT statement (GENSELECT), 390
   OUTPUT statement (LOGSELECT), 577
STOP= option
   SELECTION statement (ASSESS), 40
   SELECTION statement (BINNING), 40
   SELECTION statement (CARDINALITY), 40
   SELECTION statement (CORRELATION), 40
   SELECTION statement (GAMMOD), 40
   SELECTION statement (GENSELECT), 40
   SELECTION statement (ICA), 40
   SELECTION statement (KCLUS), 40
   SELECTION statement (LMIXED), 40
   SELECTION statement (LOGSELECT), 40
   SELECTION statement (MBC), 40
   SELECTION statement (PARTITION), 40
   SELECTION statement (PCA), 40
   SELECTION statement (PHSELECT), 40
   SELECTION statement (PLS MOD), 40
   SELECTION statement (QTRSELECT), 40
   SELECTION statement (REGSELECT), 40
   SELECTION statement (TREESPLIT), 40
   SELECTION statement (VARIMPUTE), 40
   SELECTION statement (VARREDUCE), 40
STORE statement
   MBC procedure, 633
STRATA statement
   PHSELECT procedure, 767
STUTC option
   EXACT statement (FREQTAB), 133
   OUTPUT statement (FREQTAB), 146
   TEST statement (FREQTAB), 196
SUBGROUPN= option
   chart statements (SPC), 953
SUBGROUPNAME= option
   PROC SPC statement, 939
SUBGROUPVALUE= option
   PROC SPC statement, 939
SUBJECT= option
   RANDOM statement (LMIXED), 515
   REPEATED statement (LMIXED), 524
SUBTRACT option
   VIICODE statement (TREESPLIT), 1032
SUPERIORITY option (BINOMIAL)
   TABLES statement (FREQTAB), 157
SUPERIORITY option (RELRISK)
   TABLES statement (FREQTAB), 187
SUPERIORITY option (RISKDIFF)
   TABLES statement (FREQTAB), 192
SURVIVAL option
   OUTPUT statement (PHSELECT), 764
SVMACHINE procedure, OUTPUT statement
   COPYV ARS= option, 1159
SYM METRY option
   EXACT statement (FREQTAB), 134
syntax
   GAMMOD procedure, 314
   GENSELECT procedure, 372
   MBC procedure, 626
   NLMOD procedure, 679
TABLES statement
   FREQTAB procedure, 147
TARGET statement
   ASSESS procedure, 1080
   BINNING procedure, 1104
   TREESPLIT procedure, 1057
TARGETEVENT= option
   AUTOTUNE statement, 1023
TAUB option
   OUTPUT statement (FREQTAB), 142
Syntax Index

TEST statement (FREQTAB), 195

TAUC option
  OUTPUT statement (FREQTAB), 146
  TEST statement (FREQTAB), 196

TECHNIQUE= option
  OPTIMIZATION statement, 508
  PROC ASSESS statement, 48
  PROC BINNING statement, 48
  PROC CARDINALITY statement, 48
  PROC CORRELATION statement, 48
  PROC GAMMOD statement, 48
  PROC GENESELECT statement, 48
  PROC ICA statement, 48
  PROC KCLUS statement, 48
  PROC LOGSELECT statement, 48
  PROC MBC statement, 48
  PROC PARTITION statement, 48
  PROC PCA statement, 48
  PROC PHSELECT statement, 48
  PROC PLSMART statement, 48
  PROC QTRSELECT statement, 48
  PROC REGSELECT statement, 48
  PROC TREESPLIT statement, 48
  PROC VARPRED statement, 48
  PROC VARREDUCE statement, 48, 1173
  PROC MBC statement, 629

TEST statement
  FREQTAB procedure, 193
  TEST2RUN= option
    chart statements (SPC), 953
    SHEWHART procedure, 978
  TEST3RUN= option
    chart statements (SPC), 953
    SHEWHART procedure, 978
  TEST= option (COMMONRISKDIFF)
    TABLES statement (FREQTAB), 163
  TEST=M option (COMMONRISKDIFF)
    TABLES statement (FREQTAB), 163
  TEST=MR option (COMMONRISKDIFF)
    TABLES statement (FREQTAB), 163
  TEST=S option (COMMONRISKDIFF)
    TABLES statement (FREQTAB), 163
  TESTF= option
    TABLES statement (FREQTAB), 203
  TESTP= option (CHISQ)
    TABLES statement (FREQTAB), 203
  TESTP= option (CHISQ)
    TABLES statement (FREQTAB), 203
  TESTS2= option
    chart statements (SPC), 955
  TESTS= option
    chart statements (SPC), 954
  THRESHOLD= option
    VIICODE statement (TREESPLIT), 1032
  TIME= option
    CODE statement (PHSELECT), 755
  TIMEPOINT= option
    CODE statement (PHSELECT), 18, 755
  TIMING option
    PROC LMIXED statement, 499
  TOL option
    MODEL statement (REGSELECT), 901
  TOL= option
    PROC ICA statement, METHOD=DEFLATION option, 436
    PROC ICA statement, METHOD=SYMMETRIC option, 437
  TOLERANCE= option
    PROC ICA statement, METHOD=DEFLATION option, 436
    PROC ICA statement, METHOD=SYMMETRIC option, 437
  TOPMODELS= option
    PROC MBC statement, 629
  TOTPCT option
    TABLES statement (FREQTAB), 193
  TRACE option
    DISPLAY statement (CORRELATION), 20, 98
    DISPLAY statement (GAMMOD), 20, 319
    DISPLAY statement (GENSELECT), 20, 377
    DISPLAY statement (ICA), 20, 439
    DISPLAY statement (KCLUS), 20, 464
    DISPLAY statement (LMIXED), 20, 502
    DISPLAY statement (LOGSELECT), 20, 565
    DISPLAY statement (MBC), 20, 630
    DISPLAY statement (MODELMATRIX), 660
    DISPLAY statement (NLMOD), 20, 683
    DISPLAY statement (OPTIMIZATION), 20, 1143
    DISPLAY statement (PCA), 20, 719
    DISPLAY statement (PHSELECT), 20, 757
    DISPLAY statement (PLSMART), 20, 805
    DISPLAY statement (QTRSELECT), 20, 842
    DISPLAY statement (REGSELECT), 20, 896
    DISPLAY statement (VARREDUCE), 20, 1175
  TRAINFRACTION= option
    AUTOTUNE statement, 1023
  TREESPLIT procedure
    AUTOTUNE statement, 1018
  INPUT statement, 1056
MODEL statement, 1028
PROC TREESPLIT statement, 1011
TARGET statement, 1057
TREESPLIT procedure, AUTOTUNE statement, 1018
TUNINGPARAMETERS= option, 1023
TREESPLIT procedure, CLASS statement, 1025
DESCENDING option, 13, 1173
MISSING option, 13, 1173
ORDER= option, 13
PARAM= option, 14
REF= option, 15
SPLIT option, 15
TREESPLIT procedure, CODE statement, 1026
COMMENT option, 17
FILE= option, 17
FORMATWIDTH= option, 17
INDENTSIZE= option, 18
LABELID= option, 18
LINESIZE= option, 18
NOTRIM option, 18
OUT= option, 18
TREESPLIT procedure, EFFECT statement
BASIS option (spline), 28
DATABASEBOUNDARY option (spline), 28
DEGREE option (polynomial), 26
DEGREE option (spline), 29
DETAILS option (multimember), 25
DETAILS option (polynomial), 26
DETAILS option (spline), 29
KNOTMAX= option (spline), 29
KNOTMETHOD option (spline), 29
KNOTMIN= option (spline), 30
MDEGREE option (polynomial), 26
NATURALCUBIC option (spline), 30
NOEFFECT option (multimember), 25
NOSEPARATE option (polynomial), 26
SEPARATE option (spline), 30
SPLIT option (spline), 31
STANDARDIZE option (polynomial), 26
TREESPLIT procedure, GROW statement, 1026
ALPHA= option, 1027
BONFERRONI option, 1027
CHAID option, 1027
CHISQUARE option, 1027
ENTROPY option, 1027
FTEST option, 1028
GINI option, 1028
IGR option, 1028
RSS option, 1028
TREESPLIT procedure, INPUT statement, 1056
TREESPLIT procedure, MODEL statement, 1028
TREESPLIT procedure, OUTPUT statement, 1028
COPYV ARS= option, 1029
OUT= option, 1029
ROLE option, 1029
TREESPLIT procedure, PARTITION statement
FRACTION option, 36, 1030
ROLEVAR= option, 36, 1030
TREESPLIT procedure, PROC TREESPLIT statement, 1011
ABSCONV option, 44
ABSFCONV option, 44
ABSFTOL option, 44
ABSGCONV option, 44
ABSGTOL option, 44
ABSTOL option, 44
ABSXCONV option, 45
ABSXTOL option, 45
ASSIGNMISSING= option, 1012
BINMETHOD = option, 1013
CLUSTERSPLIT option, 1014
CVCC option, 1014
DATA= option, 1014
FCONV option, 45
FCONV2 option, 45
FTOL option, 45
FTOL2 option, 45
GCONV option, 46
GCONV2 option, 46
GTOL option, 46
GTOL2 option, 46
INMODEL= option, 1014
MAXBRANCH= option, 1014
MAXDEPTH= option, 1015
MAXFUNC= option, 47
MAXITER= option, 47
MAXTIME= option, 47
MINITER= option, 47
MINLEAFSIZE= option, 1015
MINUSEINSEARCH= option, 1015
NOR your answer here
CONFIDENCE= option, 1030
COSTCOMPLEXITY option, 1030
KFOLD= option, 1031
LEAVES= option, 1031
OFF option, 1031
REDUCEDERROR option, 1031
TREESPLIT procedure, SELECTION statement
ADAPTIVE option, 38
CHOOSE= option, 38
COMPETITIVE option, 38
CRITERION= option, 38
DETAILS= option, 40, 41
FAST option, 39
HIERARCHY= option, 41
LSCOEFFS option, 39
MAXEFFECTS= option, 39
MAXSTEPS= option, 39
METHOD= option, 37
MINEFFECTS= option, 39
ORDERSELECT option, 41
SELECT= option, 39
SELECTION= option, 43
SLE= option, 39
SLENTRY= option, 39
SLS= option, 39
SLSTAY= option, 39
STOP= option, 40
STOPHORIZON= option, 43
TREESPLIT procedure, syntax, 1011
TREESPLIT procedure, TARGET statement, 1057
TREESPLIT procedure, VIICODE statement, 1032
ADD option, 1032
LIMIT= option, 1032
MISS option, 1032
MULTIPLY option, 1032
SUBTRACT option, 1032
THRESHOLD= option, 1032
TREND option
EXACT statement (FREQTAB), 134
OUTPUT statement (FREQTAB), 147
TABLES statement (FREQTAB), 193
TSYMM option
OUTPUT statement (FREQTAB), 147
TUNINGPARAMETERS= option
AUTOTUNE statement, 1023
TVALUE= option
PREDICT statement (NLMOD), 689
TYPE3 option
MODEL statement (GENSELECT), 386
MODEL statement (LOGSELECT), 572
MODEL statement (PHSELECT), 761
TYPE= option
RANDOM statement (LMIXED), 515
REPEATED statement (LMIXED), 524
U option
OUTPUT statement (FREQTAB), 147
UCHART statement
UCHART procedure, 946
UCL= option
OUTPUT statement (GENSELECT), 391
OUTPUT statement (LOGSELECT), 577
UCLM= option
OUTPUT statement (GENSELECT), 391
OUTPUT statement (LOGSELECT), 577
URC option
OUTPUT statement (FREQTAB), 147
unpack option
PROC GAMMOD statement, 316
UPPER= option
OUTPUT statement (GENSELECT), 391
OUTPUT statement (LOGSELECT), 577
PREDICT statement (NLMMOD), 689
UPPERB= option
PARMS statement (LMIXED), 514
UPPERXBETA= option
OUTPUT statement (GENSELECT), 391
OUTPUT statement (LOGSELECT), 577
UCR option
OUTPUT statement (FREQTAB), 147
USEPARAMETERS= option
AUTOTUNE statement, 1025
VAR statement
ASSESS procedure, 1080
CARDINALITY procedure, 1126
CORRELATION procedure, 99
ICA procedure, 441
PCA procedure, 722
VAR= option (BINOMIAL)
TABLES statement (FREQTAB), 157
VAR= option (RISKDIFF)
TABLES statement (FREQTAB), 192
VARDEF= option
PROC CORRELATION statement, 96
PROC PCA statement, 717
VAREXP= option
REDUCE statement, 1177
VARIANCEEXPLAINED= option
REDUCE statement, 1177
VARIIMPUTE procedure, CLASS statement
DESCENDING option, 13, 1173
MISSING option, 13, 1173
ORDER= option, 13
PARAM= option, 14
REF= option, 15
SPLIT option, 15
VARIIMPUTE procedure, CODE statement, 1157
COMMENT option, 17
FILE= option, 17, 1157
FORMATWIDTH= option, 17
INDENTSIZE= option, 17
LABELID= option, 18
LINESIZE= option, 18
NOTRIM option, 18
OUT= option, 18
VARIMPUTE procedure, EFFECT statement
BASES option (spline), 28
DATABASENARY option (spline), 28
DEGREE option (polynomial), 26
DEGREE option (spline), 29
DETAILS option (multimember), 25
DETAILS option (polynomial), 26
DETAILS option (spline), 29
KNOTMAX= option (spline), 29
KNOTMETHOD option (spline), 29
KNOTMIN= option (spline), 30
MDEGREE option (polynomial), 26
NATURALCUBIC option (spline), 30
NOEFFECT option (multimember), 25
NOSEPARATE option (polynomial), 26
SEPARATE option (spline), 30
SPLIT option (spline), 31
STANDARDIZE option (polynomial), 26
VARIMPUTE procedure, INPUT statement, 1157
VARIMPUTE procedure, OUTPUT statement, 1158
OUT= option, 1158
VARIMPUTE procedure, PROC VARIMPUTE V ARIMPUTE statement, 1156
ABSCONV option, 44
ABSFTOL option, 44
ABSGCONV option, 44
ABSGTOL option, 44
ABSTOL option, 44
ABSXCONV option, 45
ABSXTOL option, 45
DATA= option, 1156
FCONV option, 45
FCONV2 option, 45
FTOL option, 45
FTOL2 option, 45
GCONV option, 46
GCONV2 option, 46
GTOL option, 46
GTOL2 option, 46
MAXFUNCTION= option, 47
MAXITER= option, 47
MAXTIME= option, 47
MINITER= option, 47
NORMALIZE= option, 48
NTHREADS= option, 1157
SEED= option, 1157
VARIMPUTE procedure, SELECTION statement
ADAPTIVE option, 38
CHOOSE= option, 38
COMPETITIVE option, 38
CRITERION= option, 38
DETAILS= option, 40, 41
FAST option, 39
HIERARCHY= option, 41
LSCOEFFS option, 39
MDEFFECTS= option, 39
MAXSTEPS= option, 39
METHOD= option, 37
MINFUNCTIONS= option, 39
ORDERSELECT option, 41
SELECT= option, 39
SELECTION= option, 43
SLE= option, 39
SLENTRY= option, 39
SLS= option, 39
SLSTAY= option, 39
STOP= option, 40
STOPHORIZON= option, 43
VARIMPUTE procedure, syntax, 1156
VARINC= option
REDUCE statement, 1177
VARREDUCE procedure, CLASS statement, 1173
DESCENDING option, 13, 1173
MISSING option, 13, 1173
ORDER= option, 13
PARAM= option, 14
REF= option, 15
SPLIT option, 15
VARREDUCE procedure, CODE statement
COMMENT option, 17
FILE= option, 17
FORMATWIDTH= option, 17
INDENTSIZE= option, 17
LABELID= option, 18
LINESIZE= option, 18
NOTRIM option, 18
OUT= option, 18
VARREDUCE procedure, DISPLAY statement
CASESENSITIVE option, 20, 1174
EXCLUDE option, 20, 1174
EXCLUDEALL option, 20, 1174
TRACE option, 20, 1175
VARREDUCE procedure, DISPLAYOUT statement
INCLUDEALL option, 21, 1175
NOREPLACE option, 21, 1175
REPEATED option, 21, 1175
VARREDUCE procedure, EFFECT statement
BASIS option (spline), 28
DATABOUNDARY option (spline), 28
DEGREE option (polynomial), 26
DEGREE option (spline), 29
DETAILS option (multimember), 25
DETAILS option (polynomial), 26
DETAILS option (spline), 29
KNOTMAX= option (spline), 29
KNOTMETHOD option (spline), 29
KNOTMIN= option (spline), 30
MDEGREE option (polynomial), 26
NATURALCUBIC option (spline), 30
NOEFFECT option (multimember), 25
NOSEPARATE option (polynomial), 26
SEPARATE option (spline), 30
SPLIT option (spline), 31
STANDARDIZE option (polynomial), 26
VARREDUCE procedure, PROC VARREDUCE
    statement, 1171
    ABSCONV option, 44
    ABSCFCONV option, 44
    ABSFTOL option, 44
    ABSSGCONV option, 44
    ABSGTOL option, 44
    ABIOTOL option, 44
    ABXCONV option, 45
    ABSXTOL option, 45
    DATA= option, 1171
    FCONV option, 45
    FCONV2 option, 45
    FTOL option, 45
    FTOL2 option, 45
    GCONV option, 46
    GCONV2 option, 46
    GTOL option, 46
    GTOL2 option, 46
    MATRIX= option, 1172
    MAXFUNC= option, 47
    MAXITER= option, 47
    MAXTIME= option, 47
    MINITER= option, 47
    NOPRINT option, 1172
    NORMALIZE= option, 48
    OUTCP= option, 1172
    TECHNIQUE= option, 48, 1173
    XCONV option, 48
    XTL= option, 48
    VARREDUCE procedure, REDUCE statement, 1176
    AIC option, 1177
    AICC option, 1177
    BIC option, 1177
    MAXEFFECTS= option, 1177
    MAXITER= option, 1177
    MINVARIANCEINCREMENT= option, 1177
    VAREXP= option, 1177
    VARIANCEEXPLAINED= option, 1177
    VARINC= option, 1177
    VAREXP option
    VARSS option
    PROC PLSMOD statement, 803
    VIF option
    MODEL statement (REGSELECT), 901
    VII= option
    PROC TREESPLIT statement, 1018
    VIICODE statement
    TREESPLIT procedure, 1032
    WARN= option (CHISQ)
    TABLES statement (FREQTAB), 159
    WEIGHT statement
    CORRELATION statement (FREQTAB), 99
    FREQTAB procedure, 197
    GAMMOD procedure, 330
    GENSELECT procedure, 393
    LMMIXED procedure, 525
    LOGSELECT procedure, 579
    MODELMATRIX procedure, 664
    PCA procedure, 723
    PHSELECT procedure, 767
    QTRSELECT procedure, 851
    REGSELECT procedure, 905
    TREESPLIT procedure, 1032
    WHITEPREFIX= option
    PROC ICA statement, 437
    WITH statement
    CORRELATION procedure, 99
WOE(WOEADJUST=) option
   PROC BINNING statement, 1101
WPREFIX= option
   PROC ICA statement, 437
WTKAPPA option
   EXACT statement (FREQTAB), 134
   OUTPUT statement (FREQTAB), 147
   TEST statement (FREQTAB), 196
WTRESSCH option
   OUTPUT statement (PHSELECT), 764
XBETA option
   OUTPUT statement (PHSELECT), 764
XBETA= option
   OUTPUT statement (GENSELECT), 391
   OUTPUT statement (LOGSELECT), 577
XCHART statement
   XCHART procedure, 947
XCONV option
   PROC ASSESS statement, 48
   PROC BINNING statement, 48
   PROC CARDINALITY statement, 48
   PROC CORRELATION statement, 48
   PROC GAMMOD statement, 48
   PROC GENSELECT statement, 48
   PROC ICA statement, 48
   PROC KCLUS statement, 48
   PROC LMMIXED statement, 48
   PROC LOGSELECT statement, 48
   PROC MBC statement, 48
   PROC NLMOD statement, 48
   PROC PARTITION statement, 48
   PROC PCA statement, 48
   PROC PHSELECT statement, 48
   PROC PLSMIXED statement, 48
   PROC QTRSELECT statement, 48
   PROC REGSELECT statement, 48
   PROC TREESPLIT statement, 48
   PROC VARIMPUTE statement, 48
   PROC VARREDUCE statement, 48
XCONV= option
   OPTIMIZATION statement, 509
XRCHART statement
   XRCHART procedure, 948
XSCHART statement
   XSCHART procedure, 949
XTOL option
   PROC ASSESS statement, 48
   PROC BINNING statement, 48
   PROC CARDINALITY statement, 48
   PROC CORRELATION statement, 48
   PROC GAMMOD statement, 48
   PROC GENSELECT statement, 48
   PROC ICA statement, 48
   PROC KCLUS statement, 48
   PROC LMMIXED statement, 48
   PROC LOGSELECT statement, 48
   PROC MBC statement, 48
   PROC NLMOD statement, 48
   PROC PARTITION statement, 48
   PROC PCA statement, 48
   PROC PHSELECT statement, 48
   PROC PLSMIXED statement, 48
   PROC QTRSELECT statement, 48
   PROC REGSELECT statement, 48
   PROC TREESPLIT statement, 48
   PROC VARIMPUTE statement, 48
   PROC VARREDUCE statement, 48
ZELEN option
   EXACT statement (FREQTAB), 129
   OUTPUT statement (FREQTAB), 141
ZEROS option
   WEIGHT statement (FREQTAB), 197
Gain Greater Insight into Your SAS® Software with SAS Books.

Discover all that you need on your journey to knowledge and empowerment.

support.sas.com/bookstore for additional books and resources.