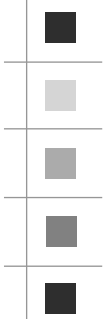
SAS Publishing

TECHNICAL REPORT

SAS/STAT® Software: Changes and Enhancements, Release 8.2



The Power to Know™

58309

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS/STAT® Software: Changes and Enhancements, Release 8.2*, Cary, NC: SAS Institute Inc., 2001

ISBN 1-58025-850-6

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

1st printing, January 2001

Table of Contents

Chapter 1
The BOXPLOT Procedure

Chapter Table of Contents

Chapter 1
The BOXPLOT Procedure

Overview

You can now specify variables from the input data set to be plotted as overlays on the box plot of an analysis variable. You can control the appearance of the symbols used to plot the overlay variables and the lines connecting the plotted points.

You can also specify uniform resource locators (URLs) to be associated with box-and-whisker plots and overlay points. When graphics output is directed to HTML, these URLs can provide links to other Web pages.

Syntax

PLOT Statement

The PLOT statement supports the following new options.

CCOVERLAY=(*color-list***)**
specifies colors for the line segments connecting points on overlays. Colors in the CCOVERLAY= list are matched with variables in the corresponding positions in the OVERLAY= list. By default, points are connected by line segments of the same color as the plotted points. You can specify the value NONE to suppress the line segments connecting points on an overlay.

COVERLAY=(*color-list***)**
specifies the colors used to plot overlay variables. Colors in the COVERLAY= list are matched with variables in the corresponding positions in the OVERLAY= list.

HTML=(*variable***)**
specifies uniform resource locators (URLs) as values of the specified character variable, for formatted values of a numeric variable. These URLs are associated with box-and-whisker plots when graphics output is directed into HTML. The value of the HTML= variable should be the same for each observation with a given value of the group variable.

LOVERLAY=(*linetypes***)**
specifies line types for the line segments connecting points on overlays. Line types in the LOVERLAY= list are matched with variables in the corresponding positions in the OVERLAY= list.

NOOVERLAYLEGEND
suppresses the legend for overlay variables that is displayed by default when the OVERLAY= option is specified.

OVERLAY=(*variable-list***)**
specifies variables to be plotted as overlays on a box plot. A point is plotted for each overlay variable at each group for which it has a non-missing value. The value of a particular overlay variable should be the same for each observation in the input data set with a given value of the group variable.

OVERLAYHTML=(*variable-list***)**
specifies variables whose values are URLs to be associated with points on overlays. These URLs are associated with points on an overlay plot when graphics output is directed into HTML. Variables in the OVERLAYHTML= list are matched with variables in the corresponding positions in the OVERLAY= list. The value of the OVERLAYHTML= variable should be the same for each observation with a given value of the group variable.

OVERLAYLEGLAB='*label***'**
specifies the label displayed to the left of the overlay legend. The label can be up to 16 characters long and must be enclosed in quotes.

OVERLAYSYM=(*symbol-list***)**
specifies symbols used to plot points on overlays. Symbols in the OVERLAYSYM= list are matched with variables in the corresponding positions in the OVERLAY= list.

OVERLAYSYMHT=(*value-list***)**
specifies the heights of symbols used to plot points on overlays. Heights in the OVERLAYSYMHT= list are matched with variables in the corresponding positions in the OVERLAY= list.

WOVERLAY=(*value-list***)**
specifies the widths in pixels of the line segments connecting points on overlays. Widths in the WOVERLAY= list are matched with variables in the corresponding positions in the OVERLAY= list.

Details

Values specified with the COVERLAY=, OVERLAYSYM=, and OVERLAYSYMHT= options are matched with overlay variables by position in the option value lists. When one of these options is not specified, the associated values are taken from the SYMBOL definitions currently in effect. If an option is specified but lists fewer values than the number of overlay variables, the overlays for the remaining unmatched variables use values from the SYMBOL definitions. SYMBOL1 is used in the box plot, so the SYMBOL definitions associated by default with the overlay plots start with SYMBOL2.

The vertical axis of a box plot is scaled to accommodate each box-and-whisker plot and all the points on any overlaid plots. If the range of overlay variable values is much greater than the range of analysis variable values, the box-and-whisker plots may be compressed and difficult to read.

Examples

Example 1.1. Putting Overlays on Box Plots

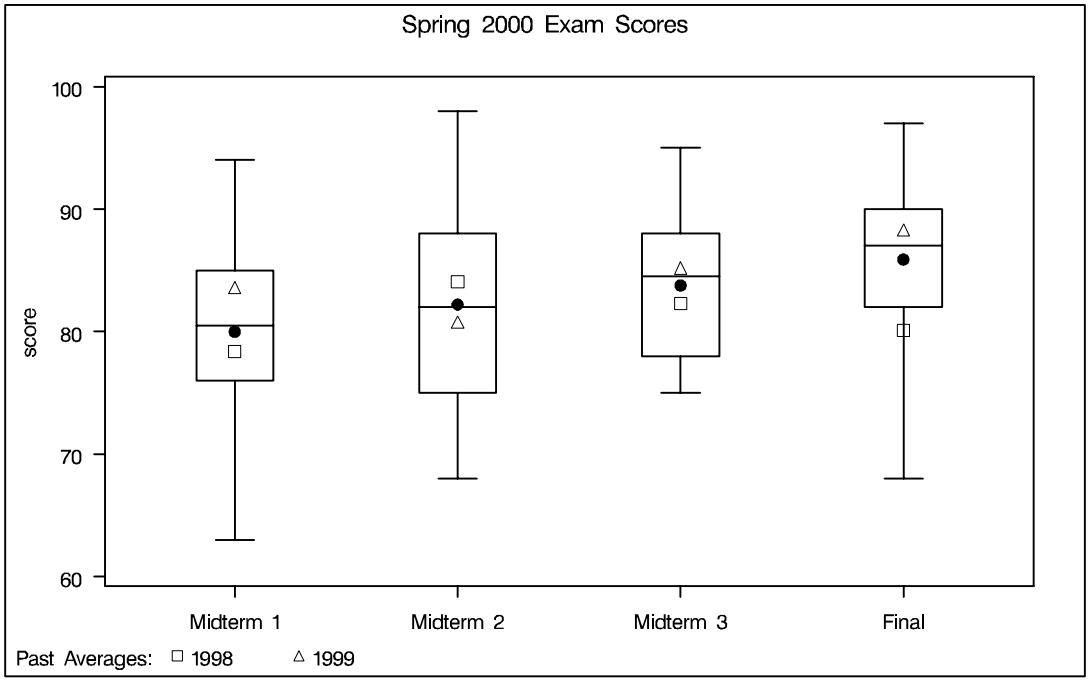
The following statements create two data sets. The first, called Scores, contains student scores for each of four exams given during the spring 2000 semester. The History data set contains average scores made on the exams by classes in the two previous spring semesters. The History data set is merged with Scores to produce an input data set for the BOXPLOT procedure.

```
proc format;
   value examfmt
   1='Midterm 1'
   2='Midterm 2'
   3='Midterm 3'
   4='Final'
   ;

data Scores;
   input exam @;
   do i=1 to 18;
      input score @;
      output;
      end;
   drop i;
   datalines;
1 78 92 83 81 65 90 72 80 63 88 85 79 94 71 76 84 77 82
2 82 88 94 90 73 79 75 84 68 82 86 73 89 75 77 98 81 86
3 88 85 95 91 76 77 78 88 80 76 83 84 91 75 83 88 85 85
4 85 90 91 93 68 82 82 97 77 83 85 89 95 77 84 90 89 89
;
run;

data History;
   label avg98 = '1998';
   label avg99 = '1999';
   input exam avg98 avg99;
   datalines;
1  78.4 83.6
2  84.1 80.8
3  82.3 85.2
4  80.1 88.3
;
run;

data Scores;
  merge scores history;
  by exam;
run;
```

You can create a box plot of scores on the four exams from 2000, overlaid with averages from the two previous years. The following statements produce the box plot with overlays shown in Output 1.1.1.

```
symbol1 v=dot c=salmon;
title 'Spring 2000 Exam Scores';
proc boxplot data=Scores;
   plot score * exam /
      cframe        = vligb
      cboxes        = dagr
      cboxfill      = ywh
      overlay       = (avg98 avg99)
      coverlay      = (vigb vig)
      overlaysym    = (square circle)
      ccoverlay     = (none none)
      overlayleglab = 'Past Averages:'
      nohlabel
      ;
   format exam examfmt.;
run;
```

Output 1.1.1. Box Plot of Exam Scores



The OVERLAY= option specifies the variables to plot as overlays on the box plot. The COVERLAY= and OVERLAYSYM= options select the colors and shapes of the symbols used to plot the overlays. CCOVERLAY= specifies the colors of the line segments used to connect the overlay points. In this case the keyword NONE is specified for each overlay, suppressing the connecting lines.

The order in which variables appear in the OVERLAY= list determines the values from related option lists that are used to plot the overlays. The variable avg98 appears first in the OVERLAY= list, so it is plotted using the first values from the related lists: COVERLAY= VIGB, OVERLAYSYM= SQUARE, and CCOVERLAY= NONE. The values associated with the second overlay variable, avg99, are VIG, CIRCLE, and NONE.

Chapter 2
The CATMOD Procedure

Chapter Table of Contents

Chapter 2
The CATMOD Procedure

Overview

You can now produce Wald confidence limits for the parameter estimates with the CLPARM option in the MODEL statement; the ALPHA= option enables you to set the significance level.

Syntax

MODEL Statement

The following options have been added to the MODEL statement.

ALPHA=*value*
sets the significance level for the Wald confidence intervals for parameter estimates. The value must be between 0 and 1. The default value of 0.05 results in the calculation of a 95% confidence interval. This option has no effect unless the CLPARM option is also specified.

CLPARM
produces Wald confidence limits for the parameter estimates. The confidence coefficient can be specified with the ALPHA= option.

The $100(1-\alpha)\%$ Wald confidence interval for a parameter β is defined as

$$\hat{\beta} \pm z_{1-\alpha/2}\hat{\sigma}$$

where z_p is the $100p$th percentile of the standard normal distribution, $\hat{\beta}$ is the parameter estimate, and $\hat{\sigma}$ is the estimate of its standard error.

Chapter 3
The FACTOR Procedure

Chapter Table of Contents

Chapter 3
The FACTOR Procedure

Syntax

PROC FACTOR Statement

The following options have been restored.

FLAG=*p*
flags absolute values larger than *p* with an asterisk when used with the ROUND option. The default value of the FLAG= option is the root mean square of all the values in the matrix being printed. Negative values are not allowed for the FLAG= option. The FLAG= option is not effective when standard errors or confidence intervals are also printed.

FUZZ=*p*
prints correlations and factor loadings with absolute values less than *p* as missing. For partial correlations, the FUZZ= value is divided by 2. For residual correlations, the FUZZ= value is divided by 4. The exact values in any matrix can be obtained from the OUTSTAT= and ODS output data sets. Negative values are not allowed for the FUZZ= option. The FUZZ= option is not effective when standard errors or confidence intervals are also printed.

ROUND
prints correlation and loading matrices with entries multiplied by 100 and rounded to the nearest integer. The exact values can be obtained from the OUTSTAT= and ODS output data sets. The ROUND option is not effective when standard errors or confidence intervals are also printed. See also the FLAG= option.

Chapter 4
The FREQ Procedure

Chapter Table of Contents

Chapter 4
The FREQ Procedure

Overview

The new POINT option in the EXACT statement requests exact point probabilities for the test statistics.

The BINOMIAL option in the TABLES statement has a new LEVEL= suboption to specify which variable level PROC FREQ uses to compute the binomial proportion for one-way tables. By default, PROC FREQ computes the binomial proportion for the first variable level that appears in the output.

The new BINOMIALC option in the TABLES statement computes the BINOMIAL option statistics, but includes a continuity correction in the asymptotic confidence limits and asymptotic test. Similarly, the new RISKDIFFC option in the TABLES statement computes the RISKDIFF statistics for 2×2 tables, but includes a continuity correction in the asymptotic confidence limits.

Syntax

EXACT Statement

EXACT *statistic-options* < / *computation-options* > **;**

The EXACT statement requests exact tests or confidence limits for the specified statistics. Optionally, PROC FREQ computes Monte Carlo estimates of the exact *p*-values. The *statistic-options* specify the statistics for which to provide exact tests or confidence limits. The *computation-options* specify options for the computation of exact statistics.

The following new *computation-option* is now available in the EXACT statement after the slash (/):

POINT
requests exact point probabilities for the test statistics.

The POINT option is available for all the EXACT statement *statistic-options* except the OR option, which provides exact confidence limits as opposed to an exact test. The POINT option is not available with the MC option, which provides Monte Carlo estimation of exact *p*-values instead of direct exact *p*-value computation.

TABLES Statement

The BINOMIAL option has a new LEVEL= suboption. The BINOMIALC and RISKDIFFC options have been added to the TABLES statement.

BINOMIAL < **(P=** *value***) | (LEVEL=** *level-number* **|** *level-value***)** >
requests the binomial proportion for one-way tables. The BINOMIAL option also provides the asymptotic standard error, asymptotic and exact confidence intervals, and the asymptotic test for the binomial proportion. To request an exact test for the binomial proportion, use the BINOMIAL option in the EXACT statement.

To specify the null hypothesis proportion for the test, use P=. If you omit P=*value*, PROC FREQ uses 0.5 as the default for the test. By default, the BINOMIAL option computes the proportion of observations for the first variable level that appears in the output. To specify a different level, use LEVEL=*level-number* or LEVEL=*level-value*, where *level-number* is the variable level's number or order in the output, and *level-value* is the formatted value of the variable level.

To include a continuity correction in the asymptotic confidence interval and test, use the BINOMIALC option instead of the BINOMIAL option.

BINOMIALC < **(P=** *value***) | (LEVEL=** *level-number* **|** *level-value***)** >
requests the BINOMIAL option statistics for one-way tables, but includes a continuity correction in the asymptotic confidence interval and the asymptotic test. The BINOMIAL option statistics include the binomial proportion, the asymptotic standard error, asymptotic and exact confidence intervals, and the asymptotic test for the binomial proportion. To request an exact test for the binomial proportion, use the BINOMIAL option in the EXACT statement.

To specify the null hypothesis proportion for the test, use P=. If you omit P=*value*, PROC FREQ uses 0.5 as the default for the test. By default, the BINOMIALC option computes the proportion of observations for the first variable level that appears in the output. To specify a different level, use LEVEL=*level-number* or LEVEL=*level-value*, where *level-number* is the variable level's number or order in the output, and *level-value* is the formatted value of the variable level.

RISKDIFFC
requests the RISKDIFF option statistics for 2×2 tables, but includes a continuity correction in the asymptotic confidence limits. The RISKDIFF option statistics include the column 1 and 2 risks (or binomial proportions), risk differences, and their confidence limits.

Chapter 5
The GAM Procedure

Chapter Table of Contents

Chapter 5
The GAM Procedure

Overview

The GAM procedure is a new procedure that fits generalized additive models as those models are defined by Hastie and Tibshirani (1990). This procedure provides an array of powerful tools for data analysis, based on nonparametric regression and smoothing techniques.

Nonparametric regression relaxes the usual assumption of linearity and enables you to uncover structure in the relationship between the independent variables and the dependent variable that might otherwise be missed. The SAS System provides many procedures for nonparametric regression, such as the LOESS procedure for local regression and the TPSPLINE procedure for thin-plate smoothing splines. The generalized additive models fit by the GAM procedure combine

- an additive assumption (Stone 1985) that enables relatively many nonparametric relationships to be explored simultaneously with
- the distributional flexibility of generalized linear models (Nelder 1972)

Thus, you can use the GAM procedure when you have multiple independent variables whose effect you want to model nonparametrically, or when the dependent variable is not normally distributed. See the "Nonparametric Regression" section on page 33 for more details on the form of generalized additive models.

The GAM procedure

- provides nonparametric estimates for additive models
- supports the use of multidimensional data
- supports multiple SCORE statements
- fits both generalized semiparametric additive models and generalized additive models
- enables you to choose a particular model by specifying the model degrees of freedom or smoothing parameter

Getting Started

The following example illustrates the use of the GAM procedure to explore in a nonparametric way how two factors affect a response. The data come from a study (Sockett et al. 1987) of the factors affecting patterns of insulin-dependent diabetes mellitus

in children. The objective is to investigate the dependence of the level of serum C-peptide on various other factors in order to understand the patterns of residual insulin secretion. The response measurement is the logarithm of C-peptide concentration (pmol/ml) at diagnosis, and the predictor measurements are age and base deficit (a measure of acidity):

```
title 'Patterns of Diabetes';
data diabetes;
   input Age BaseDeficit CPeptide @@;
   logCP = log(CPeptide);
   datalines;
 5.2   -8.1  4.8   8.8  -16.1  4.1  10.5   -0.9  5.2
10.6   -7.8  5.5  10.4  -29.0  5.0   1.8  -19.2  3.4
12.7  -18.9  3.4  15.6  -10.6  4.9   5.8   -2.8  5.6
 1.9  -25.0  3.7   2.2   -3.1  3.9   4.8   -7.8  4.5
 7.9  -13.9  4.8   5.2   -4.5  4.9   0.9  -11.6  3.0
11.8   -2.1  4.6   7.9   -2.0  4.8  11.5   -9.0  5.5
10.6  -11.2  4.5   8.5   -0.2  5.3  11.1   -6.1  4.7
12.8   -1.0  6.6  11.3   -3.6  5.1   1.0   -8.2  3.9
14.5   -0.5  5.7  11.9   -2.0  5.1   8.1   -1.6  5.2
13.8  -11.9  3.7  15.5   -0.7  4.9   9.8   -1.2  4.8
11.0  -14.3  4.4  12.4   -0.8  5.2  11.1  -16.8  5.1
 5.1   -5.1  4.6   4.8   -9.5  3.9   4.2  -17.0  5.1
 6.9   -3.3  5.1  13.2   -0.7  6.0   9.9   -3.3  4.9
12.5  -13.6  4.1  13.2   -1.9  4.6   8.9  -10.0  4.9
10.8  -13.5  5.1
;
run;
```

The following statements perform the desired analysis. The PROC GAM statement invokes the procedure and specifies the diabetes data set as input. The MODEL statement specifies logCP as the response variable and requests that univariate B-splines with the default of 4 degrees of freedom be used to model the effect of Age and BaseDeficit. The OUTPUT statement specifies that partial prediction curves are to be saved in the data set estimates:

```
title 'Patterns of Diabetes';
proc gam data=diabetes;
   model logCP = spline(age) spline(BaseDeficit);
   output out=estimates p;
run;
```

The results are shown in Figure 5.1 and Figure 5.2.

```
                          Patterns of Diabetes

                            The GAM Procedure
                        Dependent Variable: logCP
          Smoothing Model Component(s): spline(Age) spline(BaseDeficit)

                        Summary of Input Data Set

                  Number of Observations                 43
                  Number of Missing Observations          0
                  Distribution                     Gaussian
                  Link Function                    Identity


                    Iteration Summary and Fit Statistics

           Final Number of Backfitting Iterations                   5
           Final Backfitting Criterion                 5.542745E-10
           The Deviance of the Final Estimate          0.4180791724
```

Figure 5.1. Summary Statistics

Figure 5.1 shows two tables. The first table summarizes the input data set and the distributional family used for the model, and the second one summarizes the convergence criterion for backfitting.

```
                          Patterns of Diabetes

                            The GAM Procedure
                        Dependent Variable: logCP
          Smoothing Model Component(s): spline(Age) spline(BaseDeficit)

                        Regression Model Analysis
                           Parameter Estimates

                          Parameter       Standard
    Parameter              Estimate          Error    t Value    Pr > |t|

    Intercept               1.48141        0.05120      28.93      <.0001
    Linear(Age)             0.01437        0.00437       3.28      0.0024
    Linear(BaseDeficit)     0.00807        0.00247       3.27      0.0025


                        Smoothing Model Analysis
                    Fit Summary for Smoothing Components

                                                                       Num
                         Smoothing                                  Unique
Component                Parameter             DF            GCV       Obs

Spline(Age)               0.995582       4.000000       0.011675        37
Spline(BaseDeficit)       0.995299       4.000000       0.012437        39


                        Smoothing Model Analysis
                          Analysis of Deviance

                                       Sum of
 Source                      DF        Squares    Chi-Square    Pr > ChiSq

 Spline(Age)                4.00000    0.150761       12.2605        0.0155
 Spline(BaseDeficit)        4.00000    0.081273        6.6095        0.1580
```

Figure 5.2. Analysis of Model

Figure 5.2 displays summary statistics for the model. It consists of three tables. The first is the "Parameter Estimates" table for the parametric part of the model. It indicates that the linear trends for both Age and BaseDeficit are highly significant with *p*-values of 0.0024 and 0.0025. The second table is the summary of smoothing components of the nonparametric part of the model. Since the GAM fit used the default DF = 4, the main point of this table is to present the smoothing parameter values that yield this DF for each component. Finally, the third table shows the "Analysis of Deviance" table for the nonparametric component of the model.

The P option in the OUTPUT statement puts the partial predictions for Age and BaseDeficit in the output data set. You can compute the entire partial prediction effect for each factor by adding the estimated linear terms to the respective partial predictions, as in the following statement:

```
data estimates; set estimates;
   P2_age         = P_age         + 0.01437*age;
   P2_BaseDeficit = P_BaseDeficit + 0.00807*BaseDeficit;
run;
```

Plotting the partial predictions is one way to explore the overall shape of the relationship between each factor and the response. First of all, the following statements set up the graphics options:

```
legend1 frame cframe=ligr cborder=black label=none position=center;
axis1 label=(angle=90 rotate=0) minor=none;
axis2 minor=none order=(0 to 16 by 4);
symbol1 color=red  interpol=join value=none line=1;
symbol2 color=blue interpol=join value=none line=2;
```

Then the following statements plot the partial prediction curves for Age and BaseDeficit:

```
proc sort data=estimates;
   by age;

title;
proc gplot data=estimates;
   plot P_age *age = 1
        P2_Age*Age = 2 /overlay legend=legend1 frame
            cframe=ligr name='gam1' vaxis=axis1 haxis=axis2;
run;

proc sort data=estimates;
   by BaseDeficit;

axis2 minor=none;
proc gplot data=estimates;
   plot P_BaseDeficit *BaseDeficit = 1
        P2_BaseDeficit*BaseDeficit = 2 /
           overlay legend=legend1 frame cframe=ligr name='gam2'
           vaxis=axis1 haxis=axis2;
run;
```

Finally, the following statements redisplay the curves side by side for easy comparison:

```
goptions display;
proc greplay  tc=tempcat nofs;
   igout gseg;
tdef newtwo des='two plots of equal size'
1/llx=0   lly=0
  ulx=0   uly=100
  urx=50  ury=100
  lrx=50  lry=0
2/llx=50  lly=0
  ulx=50  uly=100
  urx=100 ury=100
  lrx=100 lry=0
;
template newtwo;
   treplay 1:gam1
           2:gam2;
run; quit;
```

The resulting plots for each predictor with (dotted lines) and without (solid lines) the linear term are shown in Figure 5.3.

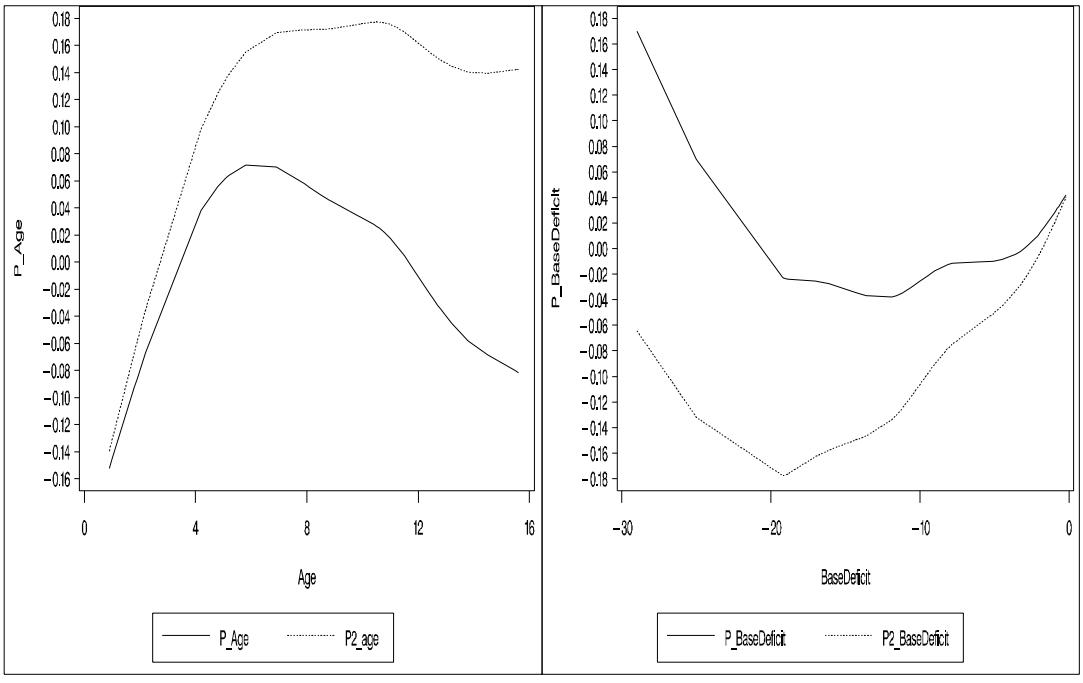


Figure 5.3. Partial Prediction for Each Predictor

Both plots show a strong quadratic pattern, with a possible indication of higher-order behavior. Further investigation is required to determine whether these patterns are real or not.

Syntax

PROC GAM < *option* > **;**
CLASS *variables* **;**
MODEL *dependent* = < *PARAM(effects)* >
smoothing effects < */options* > **;**
SCORE *data=SAS-data-set out=SAS-data-set* **;**
OUTPUT <*out=SAS-data-set*> *keyword* < · · ·*keyword*> <*/option* > **;**
BY *variables* **;**
ID *variables* **;**
FREQ *variable* **;**

The syntax of the GAM procedure is similar to that of other regression procedures in the SAS System. The PROC GAM and MODEL statements are required. The SCORE statement can appear multiple times; all other statements appear only once.

The syntax for PROC GAM is described in the following sections in alphabetical order after the description of the PROC GAM statement.

PROC GAM Statement

PROC GAM< *option* > **;**

The PROC GAM statement invokes the procedure. You can specify the following option.

DATA=*SAS-data-set*
specifies the SAS data set to be read by PROC GAM. The default value is the most recently created data set.

BY Statement

BY *variables* **;**

You can specify a BY statement with PROC GAM to obtain separate analyses on observations in groups defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in the order of the BY variables.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data using the SORT procedure with a similar BY statement.
- Specify the BY statement option NOTSORTED or DESCENDING in the BY statement for the GAM procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (ac-

cording to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.

- Create an index for the BY variables using the DATASETS procedure.

For more information on the BY statement, refer to the discussion in *SAS Language Reference: Concepts, Version 8*. For more information on the DATASETS procedure, refer to the discussion in the *SAS Procedures Guide, Version 8*.

CLASS Statement

CLASS *variables* **;**

The CLASS statement names the classification variables to be used in the analysis. Typical class variables are TREATMENT, SEX, RACE, GROUP, and REPLICATION. If the CLASS statement is used, it must appear before the MODEL statement.

Classification variables can be either character or numeric. Class levels are determined from the formatted values of the CLASS variables. Thus, you can use formats to group values into levels. Refer to the discussion of the FORMAT procedure in the *SAS Procedures Guide, Version 8*, and the discussions for the FORMAT statement and SAS formats in *SAS Language Reference: Dictionary, Version 8*.

FREQ Statement

FREQ *variable* **;**

The FREQ statement names a variable that provides frequencies for each observation in the DATA= data set. Specifically, if *n* is the value of the FREQ variable for a given observation, then that observation is used *n* times.

The analysis produced using a FREQ statement reflects the expanded number of observations. You can produce the same analysis (without the FREQ statement) by first creating a new data set that contains the expanded number of observations. For example, if the value of the FREQ variable is 5 for the first observation, the first five observations in the new data set are identical. Each observation in the old data set is replicated n_i times in the new data set, where n_i is the value of the FREQ variable for that observation.

If the value of the FREQ variable is missing or is less than 1, the observation is not used in the analysis. If the value is not an integer, only the integer portion is used.

The FREQ statement is not available when a loess smoother is included in the model.

ID Statement

ID *variables* **;**

The variables in the ID statement are copied from the input data set to the OUT= data set. If you omit the ID statement, only the variables used in the MODEL statement and requested statistics are included in the output data set.

MODEL Statement

MODEL *dependent=<PARAM(effects)><smoothing effects> </options>***;**

MODEL *event/trials=<PARAM(effects)> <smoothing effects> </options>* **;**

The MODEL statement specifies the dependent variable and the independent effects you want to use to model its values. Specify the independent parametric variables inside the parentheses of PARAM( ). The parametric variables can be either CLASS variables or continuous variables. Class variables must be declared with a CLASS statement. Interactions between variables can also be included as parametric effects. The syntax for the specification of effects is the same as for the GLM procedure.

Any number of smoothing effects can be specified, as follows:

Smoothing Effect	Meaning
SPLINE(variable <, df=number>)	fit smoothing spline with the variable and with DF=number
LOESS(variable <, df=number>)	fit local regression with the variable and with DF=number
SPLINE2(variable, variable <,df=number>)	fit bivariate thin-plate smoothing spline with DF=number

If you do not specify the DF=number option with a smoothing effect, DF=4 is used by default, unless you specify the METHOD=GCV model option.

Both parametric effects and smoothing effects are optional, but at least one of them must be present.

If only parametric variables are present, PROC GAM fits a parametric linear model using the terms inside the parentheses of PARAM( ). If only smoothing effects are present, PROC GAM fits a nonparametric additive model. If both types of effect are present, PROC GAM fits a semiparametric model using the parametric effects as the linear part of the model.

The following table shows how to specify various models for a dependent variable y and independent variables x, x1, and x2.

Table 5.1. Syntax for Common GAM Models

Type of Model	Syntax	Mathematical Form
Parametric	model y = param(x);	$E(y) = \beta_0 + \beta_1 x$
Nonparametric	model y = spline(x);	$E(y) = \beta_0 + s(x)$
Nonparametric	model y = loess(x);	$E(y) = \beta_0 + s(x)$
Semiparametric	model y = param(x1) spline(x2);	$E(y) = \beta_0 + \beta_1 x_1 + s(x_2)$
Additive	model y = spline(x1) spline(x2);	$E(y) = \beta_0 + s_1(x_1) + s_2(x_2)$
Thin-plate spline	model y = spline(x1,x2);	$E(y) = \beta_0 + s(x_1, x_2)$

You can specify the following options in the MODEL statement.

ALPHA=*number*

specifies the significance level α of the confidence limits on the final nonparametric component estimates when you request confidence limits to be included in the output data set. Specify *number* as a value between 0 and 1. The default value is 0.05. See the "OUTPUT Statement" section on page 32 for more information on the OUTPUT statement.

DIST=*distribution-id*

specifies the distribution family used in the model. The *distribution-id* can be either GAUSSIAN, BINOMIAL, BINARY, GAMMA, or POISSON. The canonical link is used with those distributions. Although theoretically, alternative links are possible, with nonparametric models the final fit is relatively insensitive to the precise choice of link function. Therefore, only the canonical link for each distribution family is implemented in PROC GAM. The loess smoother is not available for the Binomial distribution when the number of trials is greater than 1.

EPSILON=*number*

specifies the convergence criterion for the backfitting algorithm. The default value is $1.0e-8$.

EPSSCORE=*number*

specifies the convergence criterion for the local score algorithm. The default value is $1.0e-8$.

ITPRINT

produces an iteration summary table for the smoothing effects.

MAXITER=*number*

specifies the maximum number of iterations for the backfitting algorithm. The default value is 50.

MAXITSCORE=*number*

specifies the maximum number of iterations for the local score algorithm. The default value is 100.

METHOD=GCV
specifies that the value of the smoothing parameter should be selected by generalized cross validation. If you specify both METHOD=GCV and the DF= option for the smoothing effects, the user-specified DF= is used, and the METHOD=GCV option is ignored. See the "Selection of Smoothing Parameters" section on page 39 for more details on the GCV method.

NOTEST
requests that the procedure not produce the "Analysis of Deviance" table. This option reduces the running time of the procedure.

OUTPUT Statement

OUTPUT *OUT=SAS-data-set* < *keyword* · · · *keyword* > **;**

The OUTPUT statement creates a new SAS data set containing diagnostic measures calculated after fitting the model.

You can request a variety of diagnostic measures that are calculated for each observation in the data set. The new data set contains the variables specified in the MODEL statement in addition to the requested variables. If no *keyword* is present, the data set contains only the predicted values.

Details on the specifications in the OUTPUT statement are as follows.

OUT=*SAS-data-set*
specifies the name of the new data set to contain the diagnostic measures. This specification is required.

keyword
specifies the statistics to include in the output data set. The keywords and the statistics they represent are as follows:

PRED	predicted values for each smoothing component and overall predicted values at design points
UCLM	upper confidence limits for each predicted smoothing component
LCLM	lower confidence limits for each predicted smoothing component
ADIAG	diagonal element of the hat matrix associated with the observation for each smoothing spline component
RESID	residual standardized by its weights
STD	standard deviation of the prediction for each smoothing component
ALL	implies all preceding keywords

The names of the new variables that contain the statistics are formed by using a prefix of one or more characters that identify the statistic, followed by an underscore (_), followed by the variable name.

The prefixes of those new variables are as follows:

Keywords	Prefix
PRED	P_
UCLM	UCLM_
LCLM	LCLM_
ADIAG	ADIAG_
RESID	R_
STD	STD_ for spline
	STDP_ for loess

For example, suppose that you have a dependent variable y and an independent smoothing variable x, and you specify the keywords PRED and ADIAG. In this case, the output SAS data set will contain the variables P_y, P_x, and ADIAG_x.

SCORE Statement

SCORE *DATA=SAS-data-set OUT=SAS-data-set* **;**

The SCORE statement calculates predicted values for a new data set. The variables generated by the SCORE statement use the same naming conventions with prefixes as the OUTPUT statement. If you have multiple data sets to predict, you can specify multiple SCORE statements. You must use a SCORE statement for each data set.

The following options must be specified in the SCORE statement.

DATA=*SAS-data-set*
specifies an input SAS data set containing all the variables included in independent effects in the MODEL statement. The predicted response is computed for each observation in the SCORE DATA= data set.

OUT=*SAS-data-set*
specifies the name of the SAS data set to contain the predictions.

Details

Nonparametric Regression

Nonparametric regression relaxes the usual assumption of linearity and enables you to explore the data more flexibly, uncovering structure in the data that might otherwise be missed.

However, many forms of nonparametric regression do not perform well when the number of independent variables in the model is large. The sparseness of data in this setting causes the variances of the estimates to be unacceptably large. The problem of rapidly increasing variance for increasing dimensionality is sometimes referred to as the "curse of dimensionality." Interpretability is another problem with nonparametric regression based on kernel and smoothing spline estimates. The information

these estimates contain about the relationship between the dependent and independent variables is often difficult to comprehend.

To overcome these difficulties, Stone (1985) proposed additive models. These models estimate an additive approximation to the multivariate regression function. The benefits of an additive approximation are at least twofold. First, since each of the individual additive terms is estimated using a univariate smoother, the curse of dimensionality is avoided, at the cost of not being able to approximate universally. Second, estimates of the individual terms explain how the dependent variable changes with the corresponding independent variables.

To extend the additive model to a wide range of distribution families, Hastie and Tibshirani (1990) proposed generalized additive models. These models enable the mean of the dependent variable to depend on an additive predictor through a nonlinear link function. The models permit the response probability distribution to be any member of the exponential family of distributions. Many widely used statistical models belong to this general class; they include additive models for Gaussian data, nonparametric logistic models for binary data, and nonparametric log-linear models for Poisson data.

Additive Models and Generalized Additive Models

This section describes the methodology and the fitting procedure behind generalized additive models.

Let Y be a response random variable and $X_1, X_2, \cdots, X_p$ be a set of predictor variables. A regression procedure can be viewed as a method for estimating the expected value of Y given the values of $X_1, X_2, \cdots, X_p$. The standard linear regression model assumes a linear form for the conditional expectation

$$E(Y|X_1, X_2, \cdots, X_p) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p$$

Given a sample, estimates of $\beta_0, \beta_1, \cdots, \beta_p$ are usually obtained by the least squares method.

The additive model generalizes the linear model by modeling the conditional expectation as

$$E(Y|X_1, X_2, \cdots, X_p) = s_0 + s_1(X_1) + s_2(X_2) + \cdots + s_p(X_p)$$

where $s_i(X), i = 1, 2, \ldots, p$ are smooth functions.

In order to be estimable, the smooth functions s_i have to satisfy standardized conditions such as $Es_j(X_j) = 0$. These functions are not given a parametric form but instead are estimated in a nonparametric fashion.

While traditional linear models and additive models can be used in most statistical data analysis, there are types of problems for which they are not appropriate. For example, the normal distribution may not be adequate for modeling discrete responses such as counts or bounded responses such as proportions.

Generalized additive models address these difficulties, extending additive models to many other distributions besides just the normal. Thus, generalized additive models can be applied to a much wider range of data analysis problems.

Similar to generalized linear models, generalized additive models consist of a random component, an additive component, and a link function relating the two components. The response Y, the random component, is assumed to have exponential family density

$$f_Y(y;\theta;\phi) = \exp\left\{\frac{y\theta - b(\theta)}{a(\phi)} + c(y,\phi)\right\}$$

where θ is called the natural parameter and ϕ is the scale parameter. The mean of the response variable μ is related to the set of covariates $X_1, X_2, \cdots, X_p$ by a link function g. The quantity

$$\eta = s_0 + \sum_{i=1}^{p} s_i(X_i)$$

where $s_1(\cdot), \cdots, s_p(\cdot)$ are smooth functions defines the additive component, and the relationship between μ and η is defined by $g(\mu) = \eta$. The most commonly used link function is the canonical link, for which $\eta = \theta$.

Generalized additive models and generalized linear models can be applied in similar situations, but they serve different analytic purposes. Generalized linear models emphasize estimation and inference for the parameters of the model, while generalized additive models focus on exploring data nonparametrically. Generalized additive models are more suitable for exploring the data set and visualizing the relationship between the dependent variable and the independent variables.

Backfitting and Local Scoring Algorithms

Much of the development and notation in this section follows Hastie and Tibshirani (1986). Consider the estimation of the smoothing terms $s_0, s_1(\cdot), \cdots, s_p(\cdot)$ in the additive model

$$\eta(X) = s_0 + \sum_{i=1}^{p} s_j(X_j)$$

where $E\left[s_j(X_j)\right] = 0$ for every j. Since the algorithm for additive models is the basis for fitting generalized additive models, the algorithm for additive models is discussed first.

Many ways are available to approach the formulation and estimation of additive models. The backfitting algorithm is a general algorithm that can fit an additive model using any regression-type fitting mechanisms.

Define the jth set of partial residuals as

$$R_j = Y - s_0 - \sum_{k \neq j} s_k(X_k)$$

then $E(R_j|X_j) = s_j(X_j)$. This observation provides a way for estimating each smoothing function $s_j(\cdot)$ given estimates $\{\hat{s}_i(\cdot), i \neq j\}$ for all the others. The resulting iterative procedure is known as the backfitting algorithm (Friedman and Stuetzle 1981). The following formulation is taken from Hastie and Tibshirani (1986).

The Backfitting Algorithm

1. **Initialization:**
   $s_0 = E(Y), s_1^1 = s_2^1 = \cdots = s_p^1 = 0, m = 0.$
2. **Iterate:**
   $m = m + 1$
   for $j = 1$ to p do:
   $R_j = Y - s_0 - \sum_{k=1}^{j-1} s_k^m(X_k) - \sum_{k=j+1}^{p} s_k^{m-1}(X_k)$
   $S_j^m = E(R_j|X_j).$
3. **Until:**
   RSS $= \text{Avg}(Y - s_0 - \sum_{j=1}^{p} s_j^m(X_j))^2$ fails to decrease, or satisfies the convergence criterion.

In the preceding notation, $s_j^m(\cdot)$ denotes the estimate of $s_j(\cdot)$ at the mth iteration. It can be shown that RSS never increases at any step, which implies that the algorithm always converges.

The GAM procedure uses the following condition as the convergence criterion for the backfitting algorithm:

$$\frac{\sum_{j=1}^{n}\sum_{i=1}^{k} w(x_i)(s_i^{m-1}(x_j) - s_i^m(x_j))^2}{\sum_{j=1}^{n}\sum_{i=1}^{k} w(x_i)(s_i^{m-1}(x_j))^2} \leq \epsilon$$

where $\epsilon = 10^{-8}$ by default; you can change this with EPSILON= option on the MODEL statement. A weighted backfitting algorithm has the same form as for the unweighted case, except that the smoothers are weighted. The weights might represent the relative precision of each observation or might arise as part of another iterative procedure. For example, weights are used in the local scoring procedure described later in this section.

The algorithm so far described fits just additive models. The algorithm for generalized additive models is a little more complicated. Generalized additive models extend generalized linear models in the same manner that additive models extend linear regression models, that is, by replacing form $\alpha + \sum_j X_j\beta_j$ with the additive form $\alpha + \sum_j f_j(X_j)$. Thus, it is helpful to review the iteratively reweighted least-square procedure for computing the maximum likelihood estimates in a generalized linear model.

For generalized linear models, the maximum likelihood estimate of β is defined by the score equations

$$\sum_{i=1}^{n} x_{ij} \left(\frac{\partial \mu_i}{\partial \eta_i} \right) V_i^{-1}(y_i - \mu_i),\ j = 0, 1, \cdots, p$$

where V_i is the variance matrix for Y_i. The Fisher scoring procedure is the standard method for solving these equations. It involves a Newton-Raphson algorithm using the expected information matrix. An equivalent procedure that is convenient for this problem is called dependent variable regression and is a form of iteratively reweighted least squares. Given a current coefficient vector β^0, with corresponding linear predictor η^0 and fitted values μ^0, construct the adjusted dependent variable

$$z_i = \eta_i^0 + (y_i - \mu_i^0) \left(\frac{\partial \eta_i}{\partial \mu_i} \right)_0$$

Define weights w_i by

$$w_i^{-1} = \left(\frac{\partial \eta_i}{\partial \mu_i} \right)_0^2 V_i^0$$

The algorithm proceeds by regressing z_i on x with weight w_i to obtain a revised estimate β. Then a new μ^0 and η^0 are computed, new z_is are computed, and the process is repeated until the change in the deviance

$$D(y; \hat{\mu}) = 2(l(\mu_{max}; y) - l(\hat{\mu}' y))$$

is sufficiently small.

Some adjusted dependent variables and weights for commonly used models are listed in the following table.

Distribution	Link	Adjusted Dependent(Z)	Weights(w)
Normal	identity	y	1
Bin(n, μ)	logit	$\eta + (y - \mu)/n\mu(1 - \mu)$	$n\mu(1 - \mu)$
Gamma	-reciprocal	$\eta + (y - \mu)/\mu^2$	$1/\mu^2$
Poisson	log	$\eta + (y - \mu)/\mu$	μ

Generalized additive models differ from generalized linear models in that an additive predictor replaces the linear predictor. Estimation of the additive terms is accomplished by replacing the weighted linear regression in the adjusted dependent variable regression by the weighted backfitting algorithm for fitting a weighted additive mode. This results in the following algorithm described as the *local scoring algorithm*. The name "local scoring" derives from the fact that local averaging is used to generalize

the Fisher scoring procedure. The following formulation is again taken from Hastie and Tibshirani (1986).

The General Local Scoring Algorithm

1. **Initialization:**
   $s_i = g(E(y)), s_1^0 = s_2^0 = \cdots = s_p^0 = 0, m = 0.$
2. **Iterate:**
   $m = m + 1$
   Form the adjusted dependent variable, predictor, and mean based on the previous iteration using following formulas:
   $Z = \eta^{m-1} + (Y - \mu^{m-1})(\partial\eta/\partial\mu^{m-1})$
   $\eta^{m-1} = s_0 + \sum_{j=1}^{p} s_j^{m-1}(X_j)$
   $\mu^{m-1} = g^{-1}(\eta^{m-1}).$

   Form the weights
   $w_i = (\partial\mu^{m-1}/\partial\eta^{m-1})^2 V_i^{-1}.$
   Fit an additive model to Z using the backfitting algorithm with weights W to obtain estimated functions $s_j^m(\cdot)$.
3. **Until:**
   Avg($D(Y, \mu^m)$) fails to decrease, where Avg($D(Y, \mu^m)$) is an average of the deviance of estimate μ^m; or satisfies the convergence criterion.

The GAM procedure uses the following condition as the convergence criterion for local scoring:

$$\frac{\sum_{j=1}^{n}\sum_{i=1}^{k} w(x_i)(s_i^{m-1}(x_j) - s_i^m(x_j))^2}{\sum_{j=1}^{n}\sum_{i=1}^{k} w(x_i)(s_i^{m-1}(x_j))^2} \leq \epsilon^s$$

where $\epsilon^s = 10^{-8}$ by default; you can change this with the EPSSCORE= option on the MODEL statement.

The estimating procedure for generalized additive models consists of two loops. Inside each step of the local scoring algorithm (outer loop), a weighted backfitting algorithm (inner loop) is used until convergence, that is, until the RSS fails to decrease. Then, based on the estimates from this weighted backfitting algorithm, a new set of weights is calculated and the next iteration of the scoring algorithm starts. The scoring algorithm stops when the deviance of the estimates ceases to decrease.

Smoothers

A smoother is a tool for summarizing the trend of a response measurement Y as a function of one or more predictor measurements $X_1, \cdots, X_p$. It produces an estimate of the trend that is less variable than Y itself. An important property of a smoother is its nonparametric nature. It doesn't assume a rigid form for the dependence of Y on $X_1, \cdots, X_p$. This section gives a brief overview of the smoothers that can be used with the GAM procedure.

Cubic Smoothing Spline

A smoothing spline is the solution to the following optimization problem: among all functions $\eta(x)$ with two continuous derivatives, find one that minimizes the penalized least square

$$\sum_{i=1}^{n}(y_i - \eta(x_i))^2 + \lambda \int_a^b (\eta^{''}(t))^2 dt$$

where λ is a fixed constant, and $a \leq x_1 \leq \cdots \leq x_n \leq b$. The first term measures closeness to the data while the second term penalizes curvature in the function. It can be shown that there exists an explicit, unique minimizer, and that minimizer is a natural cubic spline with knots at the unique values of x_i.

The parameter λ is the smoothing parameter. Large values of λ produce smoother curves while smaller values produce wiggly curves.

Local Regression

Local regression is proposed by Cleveland, Devlin, and Grosse (1988). The idea of local regression is that at a predictor x, the regression function $\eta(x)$ can be locally approximated by the value of a function in some specified parametric class. Such a local approximation is obtained by fitting a regression surface to the data points within a chosen neighborhood of the point x. A weighted least squares algorithm is used to fit linear or quadratic functions of the predictors at the centers of neighborhoods. The radius of each neighborhood is chosen so that the neighborhood contains a specified percentage of the data points. The smoothing parameter for the local regression procedure, which controls the smoothness of the estimated curve, is the fraction of the data in each local neighborhood. Data points in a given local neighborhood are weighted by a smooth decreasing function of their distance from the center of the neighborhood. Refer to "The LOESS Procedure" in *SAS/STAT User's Guide, Version 8* for more details.

Thin-Plate Smoothing Spline

The thin-plate smoothing spline is a multivariate version of the cubic smoothing spline. The theoretical foundations for the thin-plate smoothing spline are described in Duchon (1976, 1977) and Meinguet (1979). Further results and applications are given in Wahba and Wendelberger (1980). Refer to "The TPSPLINE Procedure" in *SAS/STAT User's Guide, Version 8* for more details.

Selection of Smoothing Parameters

CV and GCV

The smoothers discussed here have a single smoothing parameter. In choosing the smoothing parameter, cross validation can be used. Cross validation works by leaving points (x_i, y_i) out one at a time, estimating the squared residual for smooth function at x_i based on the remaining $n-1$ data points, and choosing the smoother to minimize the sum of those squared residuals. This mimics the use of training and test samples for prediction. The cross validation function is defined as

$$CV(\lambda) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{\eta}_{\lambda}^{-i}(x_i))^2$$

where $\hat{\eta}_{\lambda}^{-i}(x_i)$ indicates the fit at x_i, computed by leaving out the ith data point. The quantity $nCV(\lambda)$ is sometimes called the prediction sum of squares or $PRESS$ (Allen 1974).

All of the smoothers fit by the GAM procedure can be formulated as a linear combination of the sample responses

$$\hat{\eta}(x) = A(\lambda)Y$$

for some matrix $A(\lambda)$, which depends on λ. (The matrix $A(\lambda)$ depends on x and the sample data, as well, but this dependence is suppressed in the preceding equation.) Let a_{ii} be the diagonal elements of the $A(\lambda)$. Then the CV function can be expressed as

$$CV(\lambda) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{(y_i - \hat{\eta}_{\lambda}(x_i))}{1 - a_{ii}}\right)^2$$

In most cases, it is very time consuming to compute the quantity a_{ii}. To solve this computational problem, Wahba (1990) has proposed the generalized cross validation function (GCV) that can be used to solve a wide variety of problems involving selection of a parameter to minimize the prediction risk.

The GCV function is defined as

$$GCV(\lambda) = \frac{\sum_{i=1}^{n}(y_i - \hat{\eta}_{\lambda}(x_i))^2}{(n - \mathrm{tr}(A(\lambda)))^2}$$

The GCV formula simply replaces the a_{ii} with $\mathrm{tr}(A(\lambda))/n$. Therefore, it can be viewed as a weighted version of CV. In most of the cases of interest, GCV is closely related to CV but much easier to compute. The GAM procedure uses the GCV function as the criterion for choosing the smoothing parameters.

Degrees of Freedom

The estimated GAM model can be expressed as

$$\hat{\eta}(X) = \hat{s}_0 + \sum_{i=1}^{p} A_i(y, \lambda)Y$$

Because the weights are calculated based on previous iteration during the local scoring iteration, the matrices A_i may depend on Y for non-Gaussian data. However, for the final iteration, the A_i matrix has the same role as the projection matrix in linear regression; therefore, nonparametric degrees of freedom (DF) for the ith smoother can be defined as

$$DF(smoother) = \mathrm{tr}(A_i(y, \lambda))$$

The GAM procedure gives you the option of specifying the degrees of freedom for each individual smoothing component. If you choose a particular value for the degrees of freedom, then during every local scoring algorithm the corresponding smoothing parameter λ will be set to satisfying

$$\text{tr}(A_i(y, \lambda)) = df$$

The final estimate for the smoother during this local scoring iteration will be based on this λ.

Confidence Intervals for Smoothers

In the GAM procedure, curvewise confidence intervals for smoothing splines and pointwise confidence intervals for loess are provided in the output dataset.

Curvewise Confidence Interval for Smoothing Spline

Viewing the spline model as a Bayesian model, Wahba (1983) proposed Bayesian confidence intervals for smoothing spline estimates as follows:

$$\hat{s}_\lambda(x_i) \pm z_{\alpha/2}\sqrt{\hat{\sigma}^2 a_{ii}(\lambda)}$$

where $a_{ii}(\lambda)$ is the ith diagonal element of the $A(\lambda)$ matrix and $z_{\alpha/2}$ is the $\alpha/2$ point of the normal distribution. The confidence intervals are interpreted as intervals "across the function" as opposed to point-wise intervals.

Suppose that you fit a spline estimate to experimental data that consists of a true function f and a random error term, ϵ_i. In repeated experiments, it is likely that about $100(1-\alpha)\%$ of the confidence intervals cover the corresponding true values, although some values are covered every time and other values are not covered by the confidence intervals most of the time. This effect is more pronounced when the true response curve or surface has small regions of particularly rapid change.

Pointwise Confidence Interval for Loess

As defined in Cleveland et al. (1988), a standardized residual for a loess smoother follows a t distribution with ρ degrees of freedom, where ρ is called the "lookup degrees of freedom", defined as

$$\rho = \delta_1^2/\delta_2$$

where $\delta_1 = Trace(I-A(\lambda))^T(I-A(\lambda))$ and $\delta_2 = Trace((I-A(\lambda))^T(I-A(\lambda)))^2$. Therefore an approximate pointwise confidence interval at x_i is

$$\hat{s}_\lambda(x_i) \pm t_{\alpha/2;\rho}\hat{\sigma}(x_i)$$

where $\hat{\sigma}(x_i)$ is the estimate of the standard deviation.

Distribution Family and Canonical Link

For each distribution, more than one link can exist. Different link functions may result in slight differences in estimates for parametric models. However, the difference will be less pronounced for nonparametric models because of the flexibility of nonparametric model forms. To simplify the calculation, the GAM procedure uses the canonical link.

The GAM procedure can fit the data from the Gaussian and binomial distributions.

The Gaussian Model

With this model, the link function is the identity function, and the generalized additive model is the additive model.

The Binomial and Logistic Models

A binomial response model assumes that the proportion of successes Y is such that Y has a $Bin(n, p(x))$ distribution. The $Bin(n, p(x))$ refers to the binomial distribution with parameters n and $p(x)$. Often the data are binary, in which case $n = 1$. The canonical link is

$$g(p) = log\frac{p}{n-p} = \eta$$

The Poisson Model

The link function for the Poisson model is the log function. Assume the mean of the Poisson distribution is $\mu(x)$, the dependence of $\mu(x)$ and independent variable $x_1, \cdots, x_k$ is

$$g(\mu) = log(\mu) = \eta$$

The Gamma Model

Let the mean of the Gamma distribution be $\mu(x)$. The canonical link function for the Gamma distribution is -1/$\mu(x)$. Therefore, the relationship between $\mu(x)$ and the independent variable $x_1, \cdots, x_k$ is

$$g(\mu) = -\frac{1}{\mu} = \eta$$

Forms of Additive Models

Suppose that y is a continuous variable, and x1 and x2 are two explanatory variables of interest. To fit an additive model, you can use a MODEL statement similar to that used in many regression procedures in the SAS system:

```
model y = spline(x1) spline(x2);
```

This model statement requires the procedure to fit the following model:

$$\eta(x1, x2) = \textit{Intercept} + s_1(x1) + s_2(x2)$$

where the $s_i()$ terms denote nonparametric spline functions of the respective explanatory variables.

The GAM procedure can fit semiparametric models. The following MODEL statement assumes a linear relation with x1 and an unknown functional relation with x2:

```
model y = param(x1) spline(x2);
```

If you want to fit a model containing a functional two-way interaction between x1 and x2, you can use the following MODEL statement:

```
model y = spline2(x1,x2);
```

In this case, the GAM procedure fits a model equivalent to that of PROC TPSPLINE.

ODS Table Names

PROC GAM assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in the following table. For more information on ODS, refer to the chapter "Using the Output Delivery System" in *SAS/STAT User's Guide, Version 8*.

Table 5.2. ODS Tables Produced by PROC GAM

ODS Table Name	Description	Statement	Option
ANODEV	Analysis of Deviance table for smoothing variables	PROC	default
ClassSummary	Summary of class variables	PROC	default
ConvergenceStatus	Convergence status of the local score algorithm	PROC	default
InputSummary	Input data summary	PROC	default
IterHistory	Iteration history table	MODEL	ITPRINT
IterSummary	Iteration summary	PROC	default
FitSummary	Fit parameters and fit summary	PROC	default
ParameterEstimates	Parameter estimation for regression variables	PROC	default

By referring to the names of such tables, you can use the ODS OUTPUT statement to place one or more of these tables in output data sets.

Examples

Example 5.1. Generalized Additive Model with Binary Data

The following example illustrates the capabilities of the GAM procedure and compares it to the GENMOD procedure.

The data used in this example are based on a study by Bell et al. (1989). Bell and his associates studied the result of multiple-level thoracic and lumbar laminectomy,

a corrective spinal surgery commonly performed on children. The data in the study consist of retrospective measurements on 83 patients. The specific outcome of interest is the presence (1) or absence (0) of kyphosis, defined as a forward flexion of the spine of at least 40 degrees from vertical. The available predictor variables are Age in months at time of the operation, the starting of vertebrae levels involved in the operation (StartVert), and the number of levels involved (NumVert). The goal of this analysis is to identify risk factors for kyphosis. PROC GENMOD can be used to investigate the relationship among kyphosis and the predictors. The following DATA step creates the data kyphosis:

```
title 'Comparing PROC GAM with PROC GENMOD';
data kyphosis;
   input Age StartVert NumVert Kyphosis @@;
   datalines;
71  5  3  0     158 14 3 0    128 5   4 1
2   1  5  0     1   15 4 0    1   16  2 0
61  17 2  0     37  16 3 0    113 16  2 0
59  12 6  1     82  14 5 1    148 16  3 0
18  2  5  0     1   12 4 0    243 8   8 0
168 18 3  0     1   16 3 0    78  15  6 0
175 13 5  0     80  16 5 0    27  9   4 0
22  16 2  0     105 5  6 1    96  12  3 1
131 3  2  0     15  2  7 1    9   13  5 0
12  2 14  1     8   6  3 0    100 14  3 0
4   16 3  0     151 16 2 0    31  16  3 0
125 11 2  0     130 13 5 0    112 16  3 0
140 11 5  0     93  16 3 0    1   9   3 0
52  6  5  1     20  9  6 0    91  12  5 1
73  1  5  1     35  13 3 0    143 3   9 0
61  1  4  0     97  16 3 0    139 10  3 1
136 15 4  0     131 13 5 0    121 3   3 1
177 14 2  0     68  10 5 0    9   17  2 0
139 6  10 1     2   17 2 0    140 15  4 0
72  15 5  0     2   13 3 0    120 8   5 1
51  9  7  0     102 13 3 0    130 1   4 1
114 8  7  1     81  1  4 0    118 16  3 0
118 16 4  0     17  10 4 0    195 17  2 0
159 13 4  0     18  11 4 0    15  16  5 0
158 15 4  0     127 12 4 0    87  16  4 0
206 10 4  0     11  15 3 0    178 15  4 0
157 13 3  1     26  13 7 0    120 13  2 0
42  6  7  1     36  13 4 0
;

proc genmod;
   model Kyphosis = Age StartVert NumVert
                    / link=logit dist=binomial;
run;
```

Output 5.1.1. GENMOD Analysis: Partial Output

Comparing PROC GAM with PROC GENMOD

The GENMOD Procedure

PROC GENMOD is modeling the probability that Kyphosis='0'. One way to change this to model the probability that Kyphosis='1' is to specify the DESCENDING option in the PROC statement.

Analysis Of Parameter Estimates

Parameter	DF	Estimate	Standard Error	Wald 95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	1.2497	1.2424	-1.1853	3.6848	1.01	0.3145
Age	1	-0.0061	0.0055	-0.0170	0.0048	1.21	0.2713
StartVert	1	0.1972	0.0657	0.0684	0.3260	9.01	0.0027
NumVert	1	-0.3031	0.1790	-0.6540	0.0477	2.87	0.0904
Scale	0	1.0000	0.0000	1.0000	1.0000		

NOTE: The scale parameter was held fixed.

The GENMOD analysis of the independent variable effects is shown in Output 5.1.1. Based on these results, the only significant factor is StartVert with a log odds ratio of -0.1972. The variable NumVert has a p-value of 0.0904 with a log odds ratio of 0.3031.

The GENMOD procedure assumes a strict linear relationship between the response and the predictors. The following SAS statements use PROC GAM to investigate a less restrictive model, with moderately flexible spline terms for each of the predictors:

```
title 'Comparing PROC GAM with PROC GENMOD';
proc gam data=kyphosis;
   model Kyphosis=spline(Age,df=3) spline(StartVert,df=3)
                  spline(NumVert,df=3) /dist = binomial;
   output out=estimate p;
run;
```

The MODEL statement requests an additive model using a univariate B-spline for each term. The option dist=binomial with binary responses specifies a logistic model. Each term is fitted using a smoothing spline with three degrees of freedom. Although this might seem to be an unduly modest amount of flexibility, it is better to be conservative with a data set this small. An output data set estimate containing predicted values is requested by the OUTPUT statement.

Output 5.1.2 and Output 5.1.3 list the output from PROC GAM.

Output 5.1.2. Summary Statistics

Comparing PROC GAM with PROC GENMOD

The GAM Procedure
Dependent Variable: Kyphosis
Smoothing Model Component(s): spline(Age) spline(StartVert) spline(NumVert)

Summary of Input Data Set

Number of Observations	83
Number of Missing Observations	0
Distribution	Binomial
Link Function	Logit

Iteration Summary and Fit Statistics

Number of local score iterations	9
Local score convergence criterion	2.6635758E-9
Final Number of Backfitting Iterations	1
Final Backfitting Criterion	5.2326788E-9
The Deviance of the Final Estimate	46.610922317

Output 5.1.3. Model Fit Statistics

Comparing PROC GAM with PROC GENMOD

The GAM Procedure
Dependent Variable: Kyphosis
Smoothing Model Component(s): spline(Age) spline(StartVert) spline(NumVert)

Regression Model Analysis
Parameter Estimates

Parameter	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	-2.01533	1.45620	-1.38	0.1706
Linear(Age)	0.01213	0.00794	1.53	0.1308
Linear(StartVert)	-0.18615	0.07628	-2.44	0.0171
Linear(NumVert)	0.38347	0.19102	2.01	0.0484

Smoothing Model Analysis
Fit Summary for Smoothing Components

Component	Smoothing Parameter	DF	GCV	Num Unique Obs
Spline(Age)	0.999996	3.000000	328.512864	66
Spline(StartVert)	0.999551	3.000000	317.646703	16
Spline(NumVert)	0.921758	3.000000	20.144058	10

Smoothing Model Analysis
Analysis of Deviance

Source	DF	Sum of Squares	Chi-Square	Pr > ChiSq
Spline(Age)	3.00000	10.494369	16.4358	0.0009
Spline(StartVert)	3.00000	5.494968	8.6060	0.0350
Spline(NumVert)	3.00000	2.184518	3.4213	0.3311

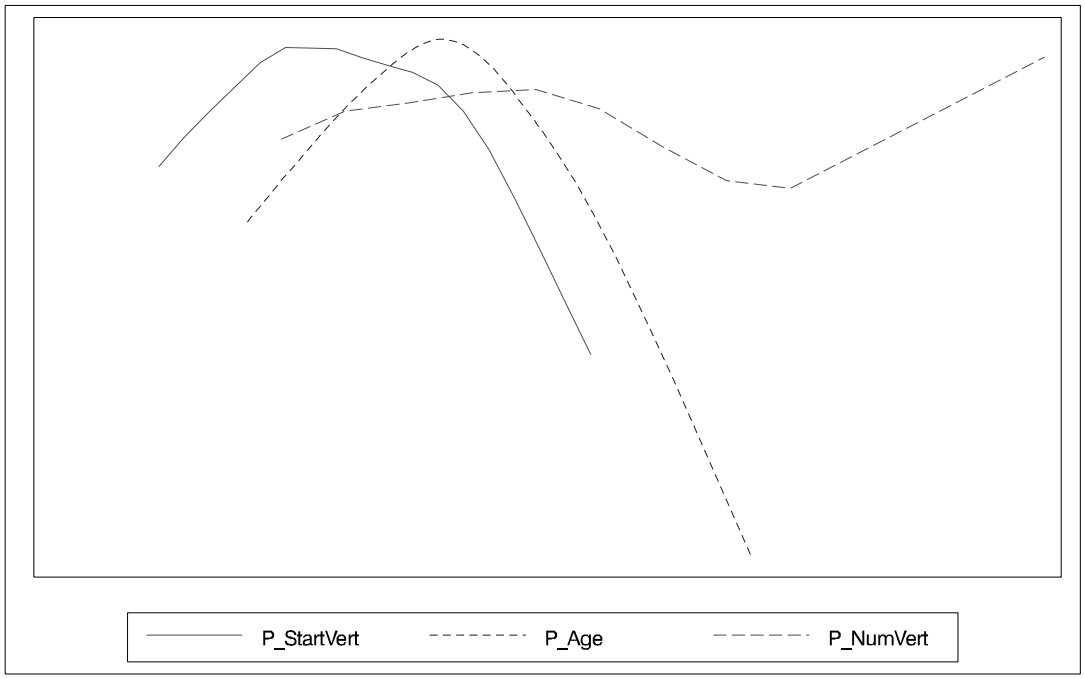
The critical part of the GAM results is the "Analysis of Deviance" table, shown in Output 5.1.3. For each smoothing effect in the model, this table gives an χ^2-test comparing the deviance between the full model and the model without this variable. In this case, the analysis of deviance results indicates that the effect of Age and StartVert are significant, while the effect of NumVert is insignificant. Plots of predictions against predictor can be used to investigate why PROC GAM and PROC GENMOD produce different results.

The GAM statement requests an output data set of predicted values to be created. Since the estimate of the generalized additive model is the sum of functional estimates of individual predictors, plus a constant, the output data set will contain a column of partial prediction for each predictor. If requested, a Bayesian confidence interval or a point-wise standard-error band, as defined in Hastie and Tibshirani (1990), can be produced in the output data set.

Using the following statements, the data set estimate is plotted in Output 5.1.4:

```
proc sort data=estimate(keep=StartVert P_StartVert)
          out =StartVert;
   by StartVert;
proc sort data=estimate(keep=Age       P_Age      )
          out =Age;
   by Age;
proc sort data=estimate(keep=NumVert   P_NumVert  )
          out =NumVert;
   by NumVert;
data Plot; merge StartVert Age NumVert;
proc standard m=0 s=1 data=Plot out=Plot;
   var StartVert Age NumVert;
run;

legend1 frame cframe=ligr cborder=black label=none
        position=center;
axis1 label=none minor=none major=none value=none;
axis2 label=none minor=none major=none value=none;
symbol1 color=red   interpol=join value=none line=1;
symbol2 color=blue  interpol=join value=none line=2;
symbol3 color=green interpol=join value=none line=3;
proc gplot data=Plot;
   title;
   plot P_StartVert*StartVert=1
        P_Age      *Age      =2
        P_NumVert  *NumVert  =3 / overlay legend=legend1
        frame cframe=ligr vaxis=axis1 haxis=axis2;
run;
```

Output 5.1.4. Partial Prediction for Each Predictor



The plot shows that the partial predictions corresponding to both Age and StartVert have a strong quadratic pattern, while NumVert has a more complicated but weaker pattern. However, in the plot for NumVert, notice that about half the vertical range of the function is determined by the point at the upper extreme. It would be a good idea, therefore, to re-run the analysis without this point, to see how much it affects the conclusions. You can do this by simply including a WHERE clause when specifying the data set for the GAM procedure, as in the following code:

```
title 'Comparing PROC GAM with PROC GENMOD';
proc gam data=kyphosis(where=(NumVert^=14));
   model Kyphosis=spline(Age,df=3) spline(StartVert,df=3)
                  spline(NumVert, df=3) /dist = binomial;
   output out=estimate p;
run;
```

Output 5.1.5. Analysis After Removing NumVert=14

Comparing PROC GAM with PROC GENMOD

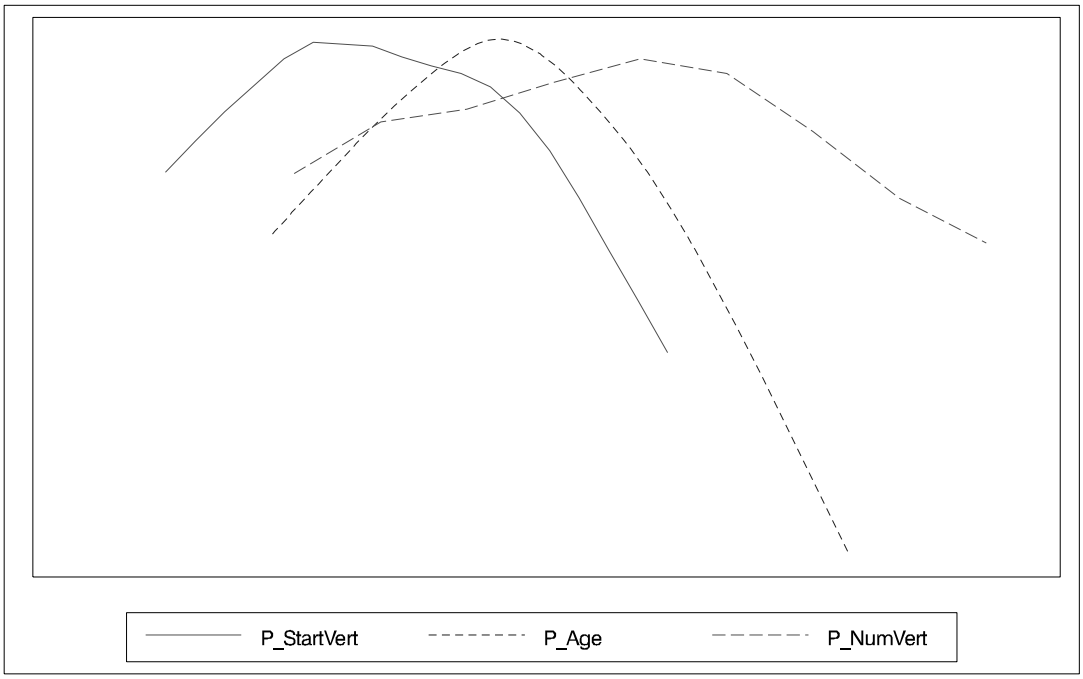
The GAM Procedure
Dependent Variable: Kyphosis
Smoothing Model Component(s): spline(Age) spline(StartVert) spline(NumVert)

Smoothing Model Analysis
Analysis of Deviance

Source	DF	Sum of Squares	Chi-Square	Pr > ChiSq
Spline(Age)	3.00000	10.587557	16.9032	0.0007
Spline(StartVert)	3.00000	5.477093	8.7443	0.0329
Spline(NumVert)	3.00000	3.209089	5.1234	0.1630

The analysis of deviance table from this re-analysis is shown in Output 5.1.5, and Output 5.1.6 shows the re-computed partial predictor plots.

Output 5.1.6. Partial Prediction After Removing NumVert=14



After removing data point NumVert=14, the predictors age and StartVert are still significant and the variable NumVert still insignificant. Based on the plot in Output 5.1.6, the removed point has almost no effect on estimates of curve shape for variables Age and StartVert. But the removal has a dramatic effect on the variable NumVert: this curve for this variable NumVert now also seems quadratic, though it is much less pronounced than for the other two variables.

Having used the GAM procedure to discover an appropriate form of the dependence of Kyphosis on each of the three independent variables, you can use the GENMOD procedure to fit and assess the corresponding parametric model. The following code

fits a GENMOD model with quadratic terms for all three variables, including tests for the joint linear and quadratic effects of each variable. The resulting contrast tests are shown in Output 5.1.7.

```
title 'Comparing PROC GAM with PROC GENMOD';
proc genmod data=kyphosis(where=(NumVert^=14));
   model kyphosis = Age       Age      *Age
                    StartVert StartVert*StartVert
                    NumVert   NumVert  *NumVert
                    /link=logit  dist=binomial;
   contrast 'Age'       Age       1, Age*Age             1;
   contrast 'StartVert' StartVert 1, StartVert*StartVert 1;
   contrast 'NumVert'   NumVert   1, NumVert*NumVert     1;
run;
```

Output 5.1.7. Joint Linear and Quadratic Tests

```
                    Comparing PROC GAM with PROC GENMOD

                              The GENMOD Procedure

PROC GENMOD is modeling the probability that Kyphosis='0'. One way to change
this to model the probability that Kyphosis='1' is to specify the DESCENDING
option in the PROC statement.


                                Contrast Results

                                      Chi-
            Contrast         DF     Square    Pr > ChiSq    Type

            Age               2      13.63        0.0011    LR
            StartVert         2      15.41        0.0005    LR
            NumVert           2       3.56        0.1684    LR
```

The results for the quadratic GENMOD model are now quite consistent with the GAM results.

From this example, you can see that PROC GAM is very useful in visualizing the data and detecting the nonlinearity among the variables.

Example 5.2. Comparing PROC GAM with PROC TPSPLINE

This example compares the GAM procedure with the TPSPLINE procedure, another nonparametric procedure that fits a smooth surface to multivariate data. It does not assume additivity of the model and uses very general basis functions for model fitting, making the TPSPLINE procedure much slower than the GAM procedure. For more details about the TPSPLINE procedure, refer to "The TPSPLINE Procedure" in *SAS/STAT User's Guide, Version 8*.

The data used here is also analyzed in "The TPSPLINE Procedure" in *SAS/STAT User's Guide, Version 8*. It presents age-adjusted melanoma incidences for 37 years from the Connecticut Tumor Registry (Houghton, Flannery, and Viola 1980):

```
title 'Comparing PROC GAM with PROC TPSPLINE';
data melanoma;
   input  year incidences @@;
   datalines;
1936    0.9   1937   0.8  1938   0.8  1939   1.3
1940    1.4   1941   1.2  1942   1.7  1943   1.8
1944    1.6   1945   1.5  1946   1.5  1947   2.0
1948    2.5   1949   2.7  1950   2.9  1951   2.5
1952    3.1   1953   2.4  1954   2.2  1955   2.9
1956    2.5   1957   2.6  1958   3.2  1959   3.8
1960    4.2   1961   3.9  1962   3.7  1963   3.3
1964    3.7   1965   3.9  1966   4.1  1967   3.8
1968    4.7   1969   4.4  1970   4.8  1971   4.8
1972    4.8
;
run;
```

The variable incidences records the number of melanoma cases per 100,000 people for the years 1936 to 1972.

Four to five degrees of freedom for each nonparametric term in a generalized additive model fits most data well. However, to select DF more objectively you can use the GCV option to minimize the generalized cross validation function, as shown in the following PROC GAM code:

```
title 'Comparing PROC GAM with PROC TPSPLINE';
proc gam data=melanoma;
   model incidences = spline(year) /method = GCV;
   output out=gam p;
run;
```

The results are listed in Output 5.2.1 and Output 5.2.2.

Output 5.2.1. Summary Statistics

```
            Comparing PROC GAM with PROC TPSPLINE

                    The GAM Procedure
                Dependent Variable: incidences
          Smoothing Model Component(s): spline(year)

                  Summary of Input Data Set

           Number of Observations                  37
           Number of Missing Observations           0
           Distribution                      Gaussian
           Link Function                     Identity


             Iteration Summary and Fit Statistics

Final Number of Backfitting Iterations                    2
Final Backfitting Criterion                               0
The Deviance of the Final Estimate             1.2242517494
```

Output 5.2.2. Model Fit Statistics

Comparing PROC GAM with PROC TPSPLINE

The GAM Procedure
Dependent Variable: incidences
Smoothing Model Component(s): spline(year)

Regression Model Analysis
Parameter Estimates

Parameter	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	-212.69706	7.00491	-30.36	<.0001
Linear(year)	0.11029	0.00358	30.77	<.0001

Smoothing Model Analysis
Fit Summary for Smoothing Components

Component	Smoothing Parameter	DF	GCV	Num Unique Obs
Spline(year)	0.634903	13.414936	0.088803	37

Smoothing Model Analysis
Analysis of Deviance

Source	DF	Sum of Squares	Chi-Square	Pr > ChiSq
Spline(year)	13.41494	2.736763	50.4880	<.0001

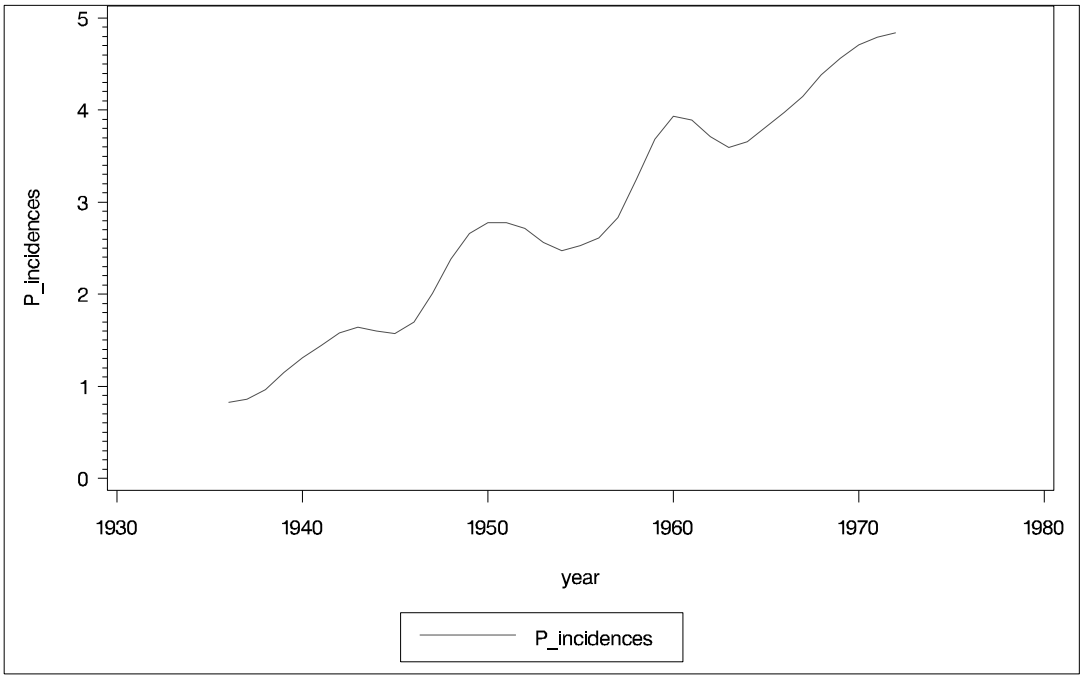
Based on the summary of the model, the final model has a DF = 13.414936 and the nonparametric trend is highly significant. Note that this DF is much greater than the default value of 4, indicating that there is a great deal of structure in the yearly incidence rates of melanoma. A prediction plot should reveal the nature of this structure:

```
legend1 frame cframe=ligr cborder=black label=none
        position=center;
axis1   label=(angle=90 rotate=0);
axis2   minor=none;
symbol1 color=red interpol=join value=none line=1;

proc sort data=gam; by year;
proc gplot data=gam;
   title;
   plot p_incidences*year = 1 /overlay legend=legend1
   frame cframe=ligr vaxis=axis1 haxis=axis2;
run;
```

Output 5.2.3 shows the predicted melanoma rate over time. Two features stand out on this plot:

- Melanoma incidence on the whole increases over the period of the study.
- A strong periodic effect is evident, with a period of a little more than a decade. This has been attributed to the 11-year sunspot cycle: the more sunspots there are, the more melanoma cases there are likely to be.

Output 5.2.3. Predicted Melanoma Incidence Rates By Year



Since PROC TPSPLINE also fits a nonparametric model, PROC TPSPLINE and PROC GAM fits should be very similar for this univariate case. The following code produces the TPSPLINE analysis shown in Output 5.2.4:

```
title 'Comparing PROC GAM with PROC TPSPLINE';
proc tpspline data=melanoma;
   model incidences = (year);
   output out=tpspline p;
run;
```

Output 5.2.4. Analysis from PROC TPSPLINE

```
Comparing PROC GAM with PROC TPSPLINE

        The TPSPLINE Procedure
  Dependent Variable:  incidences

      Summary of Input Data Set

Number of Non-Missing Observations    37
Number of Missing Observations         0
Unique Smoothing Design Points        37


       Summary of Final Model

Number of Regression Variables        0
Number of Smoothing Variables         1
Order of Derivative in the Penalty    2
Dimension of Polynomial Space         2


        Summary Statistics
       of Final Estimation

   log10(n*Lambda)            -0.0607
   Smoothing Penalty           0.5171
   Residual SS                 1.2243
   Tr(I-A)                    22.5852
   Model DF                   14.4148
   Standard Deviation          0.2328
```

The TPSPLINE model analysis shows that the DF for the model is 14.4148. This is consistent with the GAM model because the DF value in GAM excludes the degree of freedom of the linear Linear(year) term. The OUTPUT statements in the PROC GAM and PROC TPSPLINE code create gam and tpspline data sets containing the predicted values for the respective procedures. You can use the following code to look at the values for the two procedures side by side:

```
title 'Comparing PROC GAM with PROC TPSPLINE';
data both; merge gam     (rename=(p_incidences=gam     ))
                 tpspline(rename=(p_incidences=tpspline));
proc print data=both;
   var year gam tpspline;
run;
```

The results, the first ten of which are displayed in Output 5.2.5, show that PROC GAM and PROC TPSPLINE give essentially the same predictions for this problem.

Output 5.2.5. Melanoma Predictions for First Ten Years

Comparing PROC GAM with PROC TPSPLINE

Obs	year	gam	tpspline
1	1936	0.82425	0.82424
2	1937	0.85580	0.85580
3	1938	0.96379	0.96379
4	1939	1.15046	1.15046
5	1940	1.31044	1.31044
6	1941	1.43881	1.43881
7	1942	1.58218	1.58218
8	1943	1.64382	1.64382
9	1944	1.60148	1.60148
10	1945	1.57498	1.57499

Example 5.3. Poisson Regression Analysis of Component Reliability

In this example, the number of maintenance repairs on a complex system are modeled as realizations of Poisson random variables. The system under investigation has a large number of components, which occasionally break down and are replaced or repaired. During a four-year period, the system was observed to be in a state of steady operation, meaning that the rate of operation remained approximately constant. A monthly maintenance record is available for that period, which tracks the number of components removed for maintenance each month. The data are listed in the following statements that create a SAS data set.

```
title 'Analysis of Component Reliability';
data equip;
   input year month removals @@;
   datalines;
1987   1  2 1987   2  4 1987   3  3
1987   4  3 1987   5  3 1987   6  8
1987   7  2 1987   8  6 1987   9  3
1987  10  9 1987  11  4 1987  12  10
1988   1  4 1988   2  6 1988   3  4
1988   4  4 1988   5  3 1988   6  5
1988   7  3 1988   8  4 1988   9  5
1988  10  3 1988  11  6 1988  12  3
1989   1  2 1989   2  6 1989   3  1
1989   4  5 1989   5  5 1989   6  4
1989   7  2 1989   8  2 1989   9  2
1989  10  5 1989  11  1 1989  12  10
1990   1  3 1990   2  8 1990   3  12
1990   4  7 1990   5  3 1990   6  2
1990   7  4 1990   8  3 1990   9  0
1990  10  6 1990  11  6 1990  12  6
;
run;
```

For planning purposes, it is of interest to understand the long- and short-term trends in the maintenance needs of the system. Over the long term, it is suspected that

the quality of new components and repair work improves over time, so the number of component removals would tend to decrease from year to year. It is not known whether the robustness of the system is affected by seasonal variations in the operating environment, but this possibility is also of interest.

Because the maintenance record is in the form of counts, the number of removals are modeled as realizations of Poisson random variables. Denote by λ_{ij} the unobserved component removal rate for year i and month j. Since the data were recorded at regular intervals (from a system operating at a constant rate), each λ_{ij} is assumed to be a function of year and month only.

A preliminary two-way analysis is performed using PROC GENMOD to make broad inferences on repair trends. A log-link is specified for the model

$$\log \lambda_{ij} = \mu + \mathit{Year}_i + \mathit{MONTH}_j,$$

where μ is a grand mean, YEAR_i is the effect of the ith year, and MONTH_j is the effect of the jth month. A CLASS statement declares the variables year and month as categorical. Type III sum of squares are requested to test whether there is an overall effect of year and/or month.

```
title2 'Two-way model';
proc genmod data=equip;
   class year month;
   model removals=year month
         / dist=Poisson link=log type3;
run;
```

Output 5.3.1. PROC GENMOD Listing for Type III Analysis

```
        Analysis of Component Reliability
                  Two-way model

              The GENMOD Procedure

        LR Statistics For Type 3 Analysis

                                 Chi-
Source              DF       Square    Pr > ChiSq

year                 3         2.63        0.4527
month               11        21.12        0.0321
```

Output 5.3.1 displays the listed Type III statistics for the fitted model. With the test for year effects yielding a *p*-value of 0.4527, there is no evidence of a long-term trend in maintenance rates. Apparently, the quality of new or repaired components did not change between 1987 and 1990. However, the test for monthly trends does yield a small *p*-value of 0.0321, indicating that seasonal trends are just barely significant at the $\alpha = 0.05$ level.

The Type III tests indicate that the $YEAR_i$ term may be dropped from the model. The focus of the analysis is now on identifying the form of the underlying seasonal trend, which is a task that PROC GAM is especially suited for. PROC GAM will be used to fit both a reduced categorical model, with $YEAR_i$ eliminated, and a nonparametric spline model. Although PROC GENMOD also has the capability to fit categorical models, as demonstrated above, PROC GAM will be used to fit both models for a better comparison.

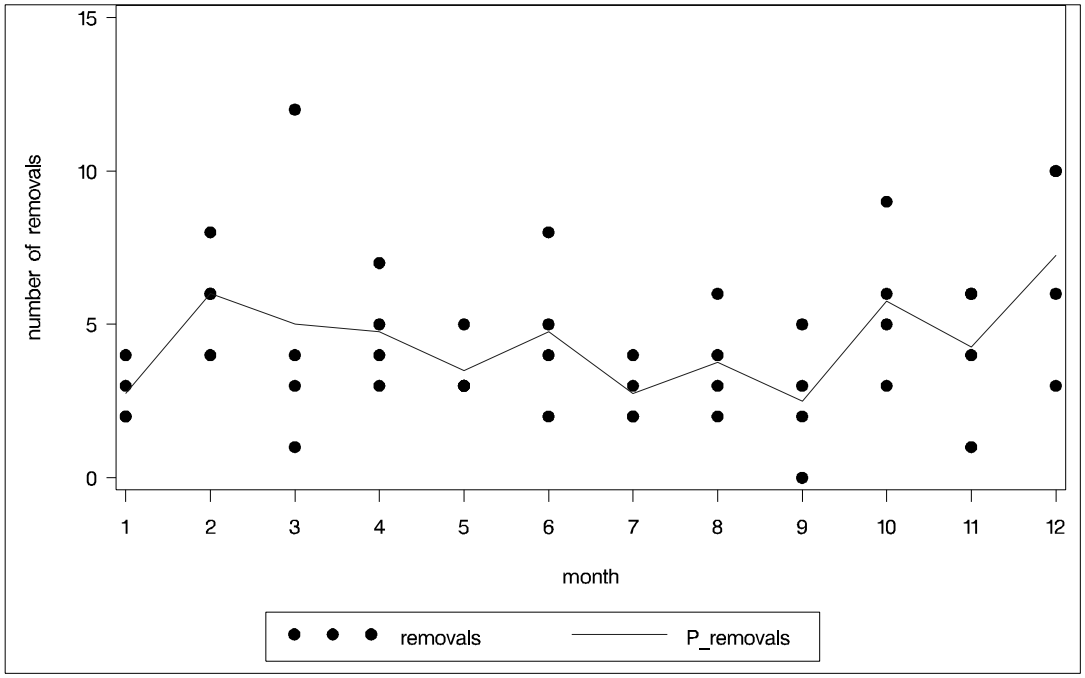
The following PROC GAM statements specify the reduced categorical model. For this part of the analysis, a CLASS statement is again used to specify that month is a categorical variable. In the follow-up, the seasonal effect will be treated as a nonparametric function of month.

```
title2 'One-way model';
proc gam data=equip;
   class month;
   model removals=param(month)
                  / dist=Poisson;
   output out=est p;
run;
```

The following statements generate a plot of the estimated seasonal trend. Note that the predicted values in the output data set correspond to the *logarithms* of the λ_{ij}, and so the exponential function is applied to put them on the scale of the original data. The plot is displayed in Output 5.3.2.

```
proc sort data=est out=plot;
   by month;
run;

data plot;
   set plot;
   P_removals = exp(P_removals);
run;

legend1 frame cframe=ligr cborder=black label=none
        position=center;
axis1   minor=none order=(0 to 15 by 5)
        label=(angle=90 rotate=0 "number of removals");
axis2   minor=none label=("month");
symbol1 color=black interpol=none value=dot;
symbol2 color=blue  interpol=join value=none line=1;

title;
proc gplot data=plot;
   plot removals*month=1 P_removals*month=2
          / overlay cframe=ligr legend=legend1 frame
            vaxis=axis1 haxis=axis2;
run; quit;
```

Output 5.3.2. Predicted Seasonal Trend from a Parametric Model Fit Using a CLASS Statement



The predicted repair rates shown in Output 5.3.2 form a jagged seasonal pattern. Ignoring the month-to-month fluctuations, which are difficult to explain and may be artifacts of random noise, the general removal rate trend starts by increasing at the beginning of the year; the trend flattens out in February and then decreases through August; it flattens out again in September and begins an increasing trend which continues throughout the rest of the year.

One advantage of nonparametric regression is its ability to highlight general trends in the data, such as those described above, and attribute local fluctuations to unexplained random noise. The nonparametric regression model used by PROC GAM specifies that the underlying removal rates λ_j are of the form

$$\log \lambda_j = \beta_0 + \beta_1 x_j + s(x_j)$$

where x_j is the value of month at month j, β_1 is a linear coefficient, and s is a nonparametric regression function. β_1 and s define the linear and nonparametric parts, respectively, of the seasonal trend.

The following statements request that PROC GAM fit a cubic spline model to the monthly repair data. The output listing is displayed in Output 5.3.3.

```
title 'Analysis of Component Reliability';
title2 'Spline model';
proc gam data=equip;
   model removals=spline(month)
                / dist=Poisson method=gcv;
   output out=est p lclm uclm;
run;
```

The METHOD=GCV option is used to determine an appropriate level of smoothing. The keywords LCLM and UCLM in the OUTPUT statement requests that lower and upper 95% confidence bounds on each $s(x_{ij})$ be included in the output data set.

Output 5.3.3. PROC GAM Listing for Cubic Spline Regression Using the METHOD=GCV Option

```
                          Spline model

                       The GAM Procedure
                Dependent Variable: removals
          Smoothing Model Component(s): spline(month)

                   Summary of Input Data Set

            Number of Observations                 48
            Number of Missing Observations          0
            Distribution                      Poisson
            Link Function                         Log


              Iteration Summary and Fit Statistics

  Number of local score iterations                     5
  Local score convergence criterion          2.546171E-11
  Final Number of Backfitting Iterations               1
  Final Backfitting Criterion                5.894286E-11
  The Deviance of the Final Estimate         56.901544117
```

Output 5.3.4. Model Fit Statistics

Spline model

The GAM Procedure
Dependent Variable: removals
Smoothing Model Component(s): spline(month)

Regression Model Analysis
Parameter Estimates

Parameter	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1.34594	0.14509	9.28	<.0001
Linear(month)	0.02274	0.01893	1.20	0.2362

Smoothing Model Analysis
Fit Summary for Smoothing Components

Component	Smoothing Parameter	DF	GCV	Num Unique Obs
Spline(month)	0.901512	2.879980	0.115848	12

Smoothing Model Analysis
Analysis of Deviance

Source	DF	Sum of Squares	Chi-Square	Pr > ChiSq
Spline(month)	2.87998	8.877763	6.8836	0.0693

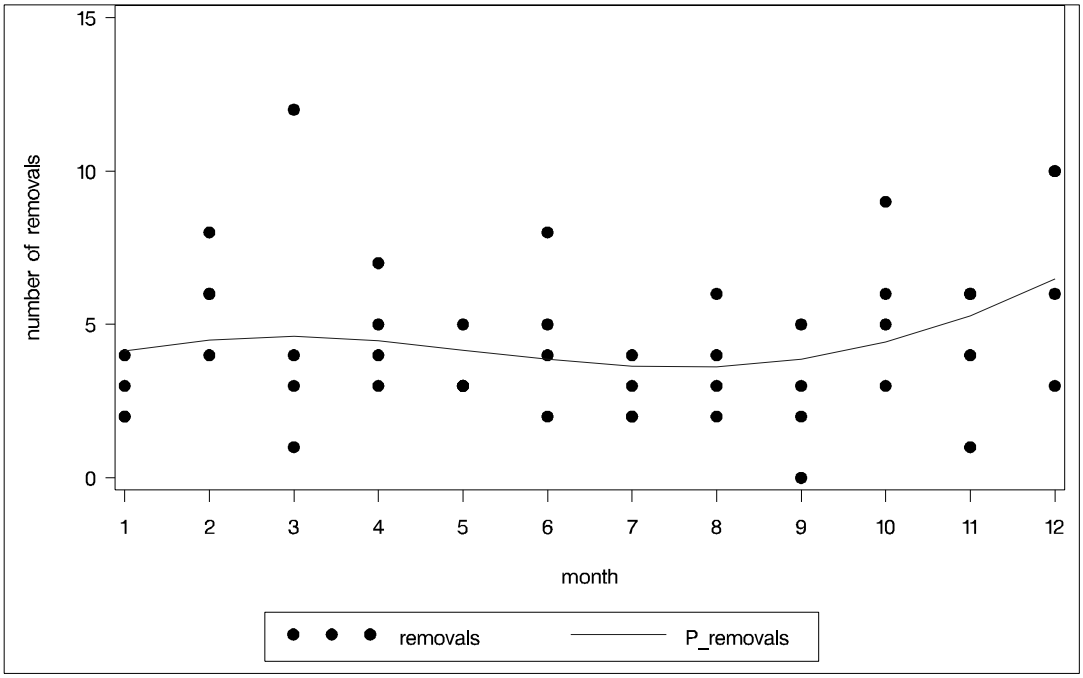
Notice in the listing of Output 5.3.4 that the DF value chosen by minimizing GCV is about 2.88, which is smaller than the default value of 4. This indicates that the spline model of the seasonal trend is relatively simple. As indicated by the "Analysis of Deviance" table, it is a non-significant feature of the data: the table lists a *p*-value of 0.0693 for the hypothesis of no seasonal trend. Note also that the "Parameter Estimates" table lists a non-significant *p*-value of 0.2362 for the hypothesis of no linear factor in the seasonal trend.

Using the following code, you can plot the predicted repair rates as displayed in Output 5.3.5.

```
proc sort data=est out=plot;
  by month;
run;

data plot;
  set plot;
  P_removals = exp(P_removals);
run;

legend1 frame cframe=ligr cborder=black label=none
        position=center;
axis1 minor=none order=(0 to 15 by 5)
```

```
      label=(angle=90 rotate=0 "number of removals");
axis2 minor=none label=("month");
symbol1 color=black interpol=none value=dot;
symbol2 color=blue  interpol=join value=none line=1;

title;
proc gplot data=plot;
   plot removals*month=1 P_removals*month=2
       / overlay cframe=ligr legend=legend1 frame
         vaxis=axis1 haxis=axis2;
run; quit;
```

Output 5.3.5. Predicted Seasonal Trend from a Cubic Spline Model



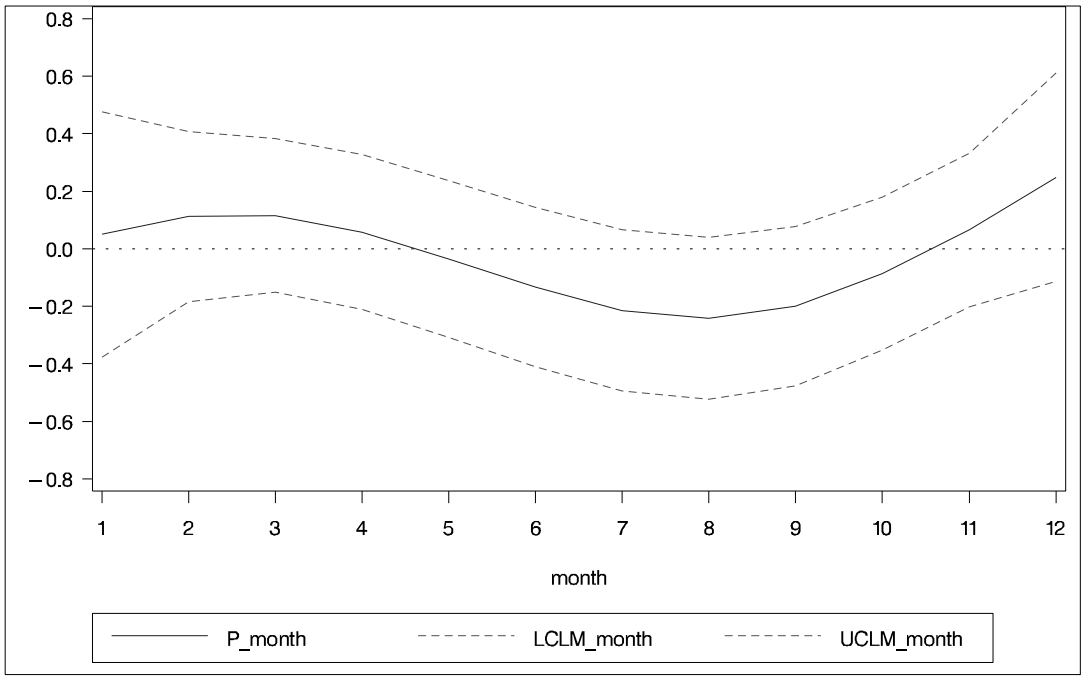
In Output 5.3.5, it is apparent that the pattern of repair rates follows the general pattern observed in Output 5.3.2. However, the plot of Output 5.3.5 is much cleaner as the month-to-month fluctuations are smoothed out to reveal the broader seasonal trend.

Both Output 5.3.2 and Output 5.3.5 show plots of the predicted dependent variable removals (after applying the exponential function) given in the output data set. You can also investigate trends in the data by examining the fit for an independent variable. For the variable month, the output statistics P_month, LCLM_month, and UCLM_month are the fitted values and confidence limits for the nonlinear component of the effect of month on removals. This component is the function s in the nonparametric regression model.

The following statements produce a plot of each $s(x_{ij})$ along with 95% confidence bounds. The plot is displayed in Output 5.3.6.

```
legend1 frame cframe=ligr cborder=black
        label=none position=center;
axis1 minor=none  order=(-0.8 to 0.8 by 0.2) label=none;
axis2 minor=none label=("month");
symbol1 color=black interpol=none value=dot;
symbol2 color=blue  interpol=join value=none line=1;
symbol3 color=red   interpol=join value=none line=4;

proc gplot data=plot;
   plot P_month*month=2
        LCLM_month*month=3 UCLM_month*month=3
        / overlay cframe=ligr legend=legend1 frame
          vref=0 lv=34 vaxis=axis1 haxis=axis2;
run;
```

Output 5.3.6. Estimated Nonparametric Factor of Seasonal Trend (Solid Line), Along with 95% Confidence Bounds (Dashed Lines)



The small *p*-value in Output 5.3.2 of 0.0321 for the hypothesis of no seasonal trend indicates that the data exhibit significant seasonal structure. However, Output 5.3.6 is a graphical illustration of a degree of indistinctness in that structure. For instance, the horizontal reference line at zero is entirely within the 95% confidence band, that is, the estimated nonlinear part of the trend is relatively flat. Thus, despite evidence of seasonality based on the parametric model, it is difficult to narrow down its significant effects to a specific part of the year.

Example 5.4. Comparing PROC GAM with PROC LOESS

In an analysis of simulated data from a hypothetical chemistry experiment, additive nonparametric regression performed by PROC GAM is compared to the unrestricted multidimensional procedure of PROC LOESS.

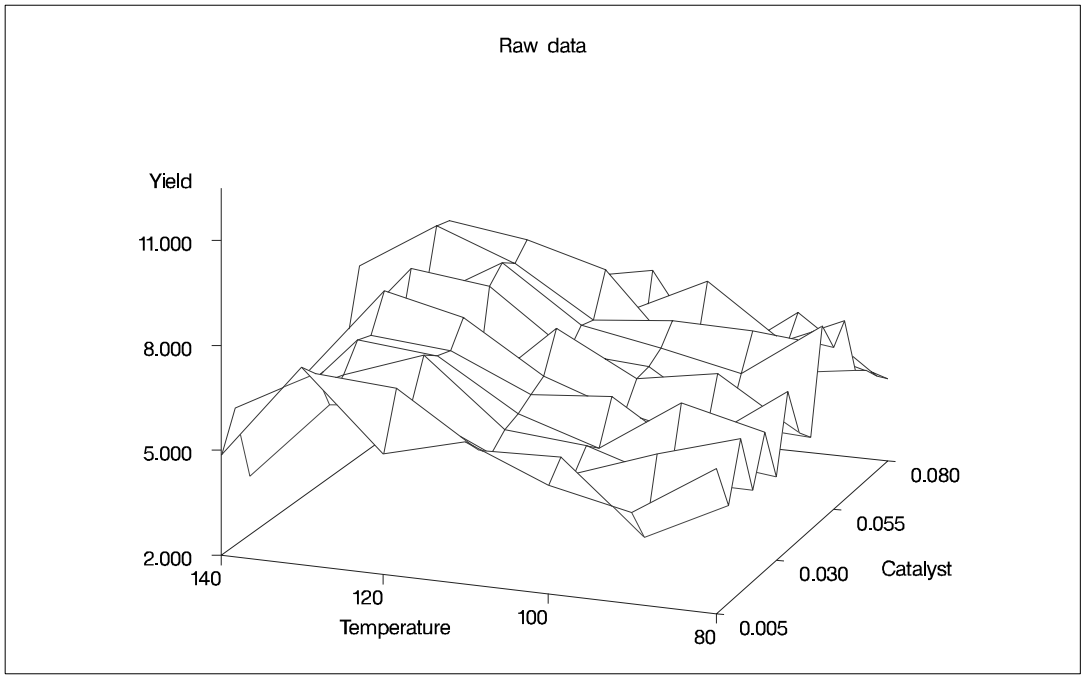
In each repetition of the experiment, a catalyst is added to a chemical solution, thereby inducing synthesis of a new material. The data are measurements of the temperature of the solution, the amount of catalyst added, and the yield of the chemical reaction. The following code reads and plots the raw data.

```
data ExperimentA;
   format Temperature f4.0 Catalyst f6.3 Yield f8.3;
   input Temperature Catalyst Yield @@;
   datalines;
 80 0.005 6.039  80 0.010 4.719  80 0.015 6.301
 80 0.020 4.558  80 0.025 5.917  80 0.030 4.365
 80 0.035 6.540  80 0.040 5.063  80 0.045 4.668
 80 0.050 7.641  80 0.055 6.736  80 0.060 7.255
 80 0.065 5.515  80 0.070 5.260  80 0.075 4.813
 80 0.080 4.465  90 0.005 4.540  90 0.010 3.553
 90 0.015 5.611  90 0.020 4.586  90 0.025 6.503
 90 0.030 4.671  90 0.035 4.919  90 0.040 6.536
 90 0.045 4.799  90 0.050 6.002  90 0.055 6.988
 90 0.060 6.206  90 0.065 5.193  90 0.070 5.783
 90 0.075 6.482  90 0.080 5.222 100 0.005 5.042
100 0.010 5.551 100 0.015 4.804 100 0.020 5.313
100 0.025 4.957 100 0.030 6.177 100 0.035 5.433
100 0.040 6.139 100 0.045 6.217 100 0.050 6.498
100 0.055 7.037 100 0.060 5.589 100 0.065 5.593
100 0.070 7.438 100 0.075 4.794 100 0.080 3.692
110 0.005 6.005 110 0.010 5.493 110 0.015 5.107
110 0.020 5.511 110 0.025 5.692 110 0.030 5.969
110 0.035 6.244 110 0.040 7.364 110 0.045 6.412
110 0.050 6.928 110 0.055 6.814 110 0.060 8.071
110 0.065 6.038 110 0.070 6.295 110 0.075 4.308
110 0.080 7.020 120 0.005 5.409 120 0.010 7.009
120 0.015 6.160 120 0.020 7.408 120 0.025 7.123
120 0.030 7.009 120 0.035 7.708 120 0.040 5.278
120 0.045 8.111 120 0.050 8.547 120 0.055 8.279
120 0.060 8.736 120 0.065 6.988 120 0.070 6.283
120 0.075 7.367 120 0.080 6.579 130 0.005 7.629
130 0.010 7.171 130 0.015 5.997 130 0.020 6.587
130 0.025 7.335 130 0.030 7.209 130 0.035 8.259
130 0.040 6.530 130 0.045 8.400 130 0.050 7.218
130 0.055 9.167 130 0.060 9.082 130 0.065 7.680
130 0.070 7.139 130 0.075 7.275 130 0.080 7.544
140 0.005 4.860 140 0.010 5.932 140 0.015 3.685
140 0.020 5.581 140 0.025 4.935 140 0.030 5.197
140 0.035 5.559 140 0.040 4.836 140 0.045 5.795
140 0.050 5.524 140 0.055 7.736 140 0.060 5.628
140 0.065 6.644 140 0.070 3.785 140 0.075 4.853
140 0.080 6.006
;
```

```
title2 'Raw data';
proc g3d data=ExperimentA;
   plot Temperature*Catalyst=Yield
        / zmin=2 zmax=11;
run;
```

The plot is displayed in Output 5.4.1.

Output 5.4.1. Surface Plot of Yield by Temperature and Amount of Catalyst



A surface fitted to the plot of Output 5.4.1 by PROC LOESS will be of a very general (and flexible) type, since the procedure requires only weak assumptions about the structure of the dependencies among the data. PROC GAM, on the other hand, makes stronger structural assumptions by restricting the fitted surface to an additive form. These differences will be demonstrated in this example.

The following code requests that both PROC LOESS and PROC GAM fit surfaces to the data.

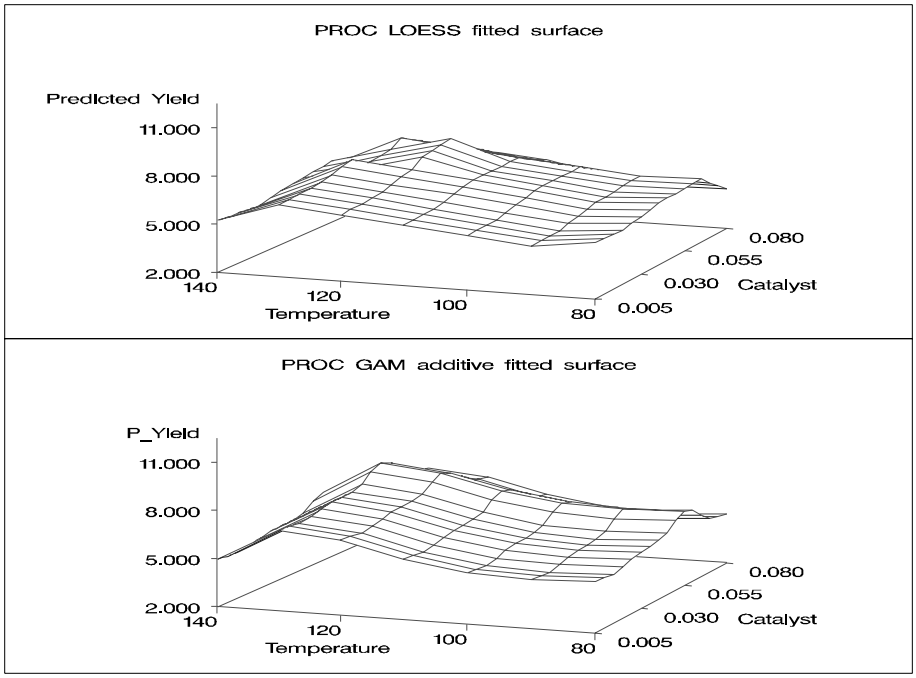
```
ods output OutputStatistics=PredLOESS;
proc loess data=ExperimentA;
   model Yield = Temperature Catalyst
         / scale=sd degree=2 select=gcv;
run;
ods output close;

proc gam data=ExperimentA;
   model Yield = loess(Temperature) loess(Catalyst)
                 / method=gcv;
   output out=PredGAM;
run;
```

In both cases the smoothing parameter was chosen as the value that minimizes GCV. This is performed automatically by PROC LOESS and PROC GAM.

The following code generates plots of the predicted yields, which are displayed in Output 5.4.2.

```
title2 'PROC LOESS fitted surface';
proc g3d data=PredLOESS;
   format pred f6.3;
   plot Temperature*Catalyst=pred
        / name='LOESSA' zmin=2 zmax=11;
run;

title2 'PROC GAM additive fitted surface';
proc g3d data=PredGAM;
   format P_Yield f6.3;
   plot Temperature*Catalyst=P_Yield
        / name='GAMA' zmin=2 zmax=11;
run;

goptions display;
proc greplay nofs tc=sashelp.templt template=v2;
   igout=gseg;
   treplay 1:loessa 2:gama;
run; quit;
```

Output 5.4.2. Fitted Regression Surfaces



Though both PROC LOESS and PROC GAM use the statistical technique loess, it is apparent from Output 5.4.2 that the manner in which it is applied is very different.

By smoothing out the data in local neighborhoods, PROC LOESS essentially fits a surface to the data in pieces, one neighborhood at a time. The local regions are treated independently, so separate areas of the fitted surface are only weakly related. PROC GAM imposes structure by requiring that cross-sections of the fitted surface always have the same shape, thereby relating regions that have a common value of the same individual regressor variable. Under that restriction, the loess technique need not be applied to the entire multidimensional scatter plot, but only to one-dimensional cross-sections of the data.

The advantage of using additive model fitting is that its statistical power is directed toward univariate smoothing, and so it is able to discern the finer details of any underlying structure in the data. Regression data may be very sparse when viewed in the context of multidimensional space, even when every individual set of regressor values densely covers its range. This is the familiar curse of dimensionality. Sparse data greatly restricts the effectiveness of nonparametric procedures, but additive model fitting, when appropriate, is one way to overcome this limitation.

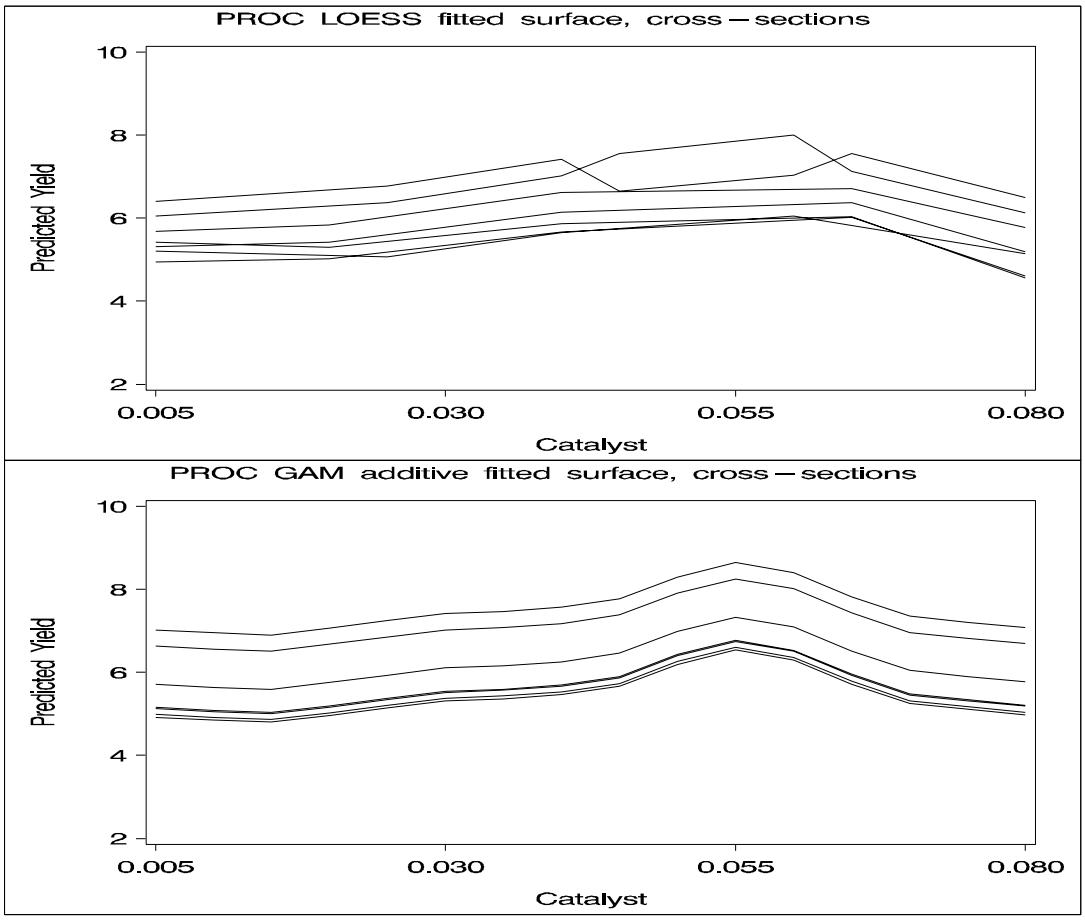
To examine these properties, plots of cross-sections of unrestricted (PROC LOESS) and additive (PROC GAM) fitted surfaces for the variable Catalyst are generated by the following code. The code for creating the cross-section plots and overlaying them is somewhat complicated, so a macro %XPlot is employed to make it easy to create this plot for the results of each procedure.

```
axis1 minor=none order=(2 to 11 by 2)
      label=(angle=90 rotate=0 "Predicted Yield");
axis2 minor=none order=(0.005 to 0.080 by 0.025)
      label=("Catalyst");
symbol1 color=blue interpol=join value=none
        line=1 width=1;

%macro XPLOT(proc=,name=);

proc sort data=Pred&proc;
   by Catalyst Temperature;
run;

data PredX&proc;
   keep Pred80 Pred90 Pred100 Pred110 Pred120 Pred130
        Pred140  Catalyst;
   array xPred{8:14} Pred80 Pred90 Pred100 Pred110
         Pred120 Pred130 Pred140;
   retain Pred80 Pred90 Pred100 Pred110 Pred120
          Pred130 Pred140;
   set Pred&proc;
   %if &proc=LOESS %then %do;
       xPred{Temperature/10} = pred;
   %end;
   %else %if &proc=GAM %then %do;
         xPred{Temperature/10} = P_Yield;
   %end;
   if abs(Temperature-140)<1 then output;
run;
```

```
proc gplot data=PredX&proc;
   plot Pred140*Catalyst=1 Pred130*Catalyst=1
        Pred120*Catalyst=1 Pred110*Catalyst=1
        Pred100*Catalyst=1 Pred90*Catalyst=1
        Pred80*Catalyst=1
        / overlay cframe=ligr name=&name
          vaxis=axis1 haxis=axis2;
run; quit;

%mend;

title;
title2 'PROC LOESS fitted surface, cross-sections';
%XPLOT(proc=LOESS,name='XLOESSA');
title2 'PROC GAM additive fitted surface, cross-sections';
%XPLOT(proc=GAM,name='XGAMA');

goptions display;
proc greplay nofs tc=sashelp.templt template=v2;
     igout=gseg;
     treplay 1:xloessa 2:xgama;
run; quit;
```

The plots are displayed in Output 5.4.3.

Output 5.4.3. Cross-sections of Fitted Regression Surfaces



Notice that the graphs in the top panel (PROC LOESS) of Output 5.4.3 have varying shapes, while every graph in the bottom panel (PROC GAM) is the same curve shifted vertically. This illustrates precisely the kind of structural differences that distinguish additive models. A second important comparison to make in Output 5.4.2 and Output 5.4.3 is the level of detail in the fitted regression surfaces. Cross-sections of the PROC LOESS surface are rather flat, but those of the additive surface have a clear shape. In particular, the ridge near Catalyst=0.055 is only vaguely evident in the PROC LOESS surface, but it is plainly revealed by the additive procedure.

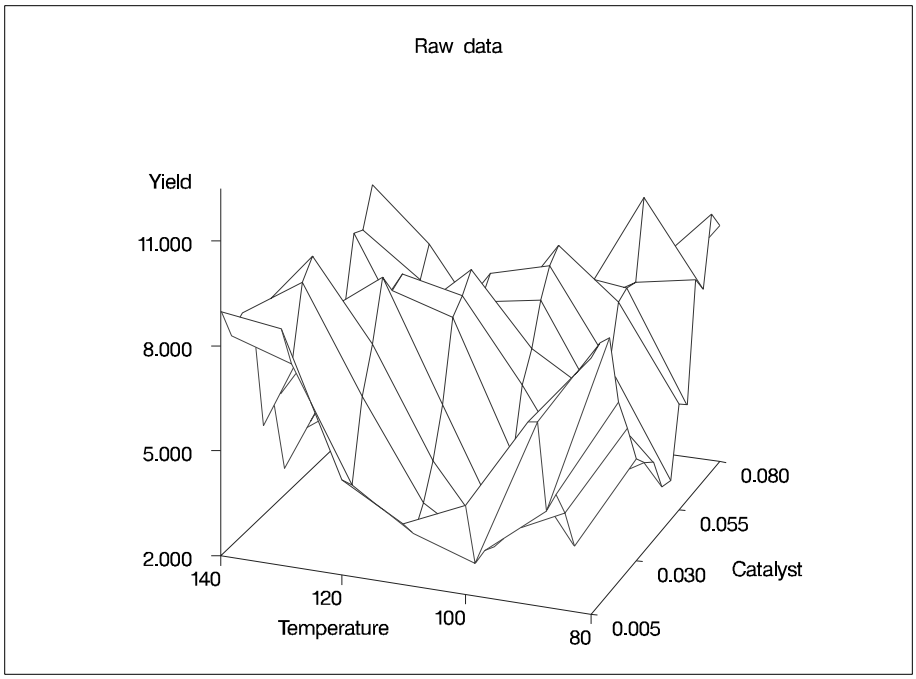
For an example of a situation where unrestricted multidimensional fitting is preferred over additive regression, consider the following simulated data from a similar experiment. The following code creates another SAS data set and plot.

```
data ExperimentB;
   format Temperature f4.0 Catalyst f6.3 Yield f8.3;
   input Temperature Catalyst Yield @@;
   datalines;
 80 0.005  9.115  80 0.010  9.275  80 0.015  9.160
 80 0.020  7.065  80 0.025  6.054  80 0.030  4.899
 80 0.035  4.504  80 0.040  4.238  80 0.045  3.232
 80 0.050  3.135  80 0.055  5.100  80 0.060  4.802
```

```
 80 0.065  8.218  80 0.070  7.679  80 0.075  9.669
 80 0.080  9.071  90 0.005  7.085  90 0.010  6.814
 90 0.015  4.009  90 0.020  4.199  90 0.025  3.377
 90 0.030  2.141  90 0.035  3.500  90 0.040  5.967
 90 0.045  5.268  90 0.050  6.238  90 0.055  7.847
 90 0.060  7.992  90 0.065  7.904  90 0.070 10.184
 90 0.075  7.914  90 0.080  6.842 100 0.005  4.497
100 0.010  2.565 100 0.015  2.637 100 0.020  2.436
100 0.025  2.525 100 0.030  4.474 100 0.035  6.238
100 0.040  7.029 100 0.045  8.183 100 0.050  8.939
100 0.055  9.283 100 0.060  8.246 100 0.065  6.927
100 0.070  7.062 100 0.075  5.615 100 0.080  4.687
110 0.005  3.706 110 0.010  3.154 110 0.015  3.726
110 0.020  4.634 110 0.025  5.970 110 0.030  8.219
110 0.035  8.590 110 0.040  9.097 110 0.045  7.887
110 0.050  8.480 110 0.055  6.818 110 0.060  7.666
110 0.065  4.375 110 0.070  3.994 110 0.075  3.630
110 0.080  2.685 120 0.005  4.697 120 0.010  4.268
120 0.015  6.507 120 0.020  7.747 120 0.025  9.412
120 0.030  8.761 120 0.035  8.997 120 0.040  7.538
120 0.045  7.003 120 0.050  6.010 120 0.055  3.886
120 0.060  4.897 120 0.065  2.562 120 0.070  2.714
120 0.075  3.141 120 0.080  5.081 130 0.005  8.729
130 0.010  7.460 130 0.015  9.549 130 0.020 10.049
130 0.025  8.131 130 0.030  7.553 130 0.035  6.191
130 0.040  6.272 130 0.045  4.649 130 0.050  3.884
130 0.055  2.522 130 0.060  4.366 130 0.065  3.272
130 0.070  4.906 130 0.075  6.538 130 0.080  7.380
140 0.005  8.991 140 0.010  8.029 140 0.015  8.417
140 0.020  8.049 140 0.025  4.608 140 0.030  5.025
140 0.035  2.795 140 0.040  3.123 140 0.045  3.407
140 0.050  4.183 140 0.055  3.750 140 0.060  6.316
140 0.065  5.799 140 0.070  7.992 140 0.075  7.835
140 0.080  8.985
;
run;

title2 'Raw data';
proc g3d data=ExperimentB;
   plot Temperature*Catalyst=Yield
        / zmin=2 zmax=11;
run;
```

A plot of the raw data is displayed in Output 5.4.4.

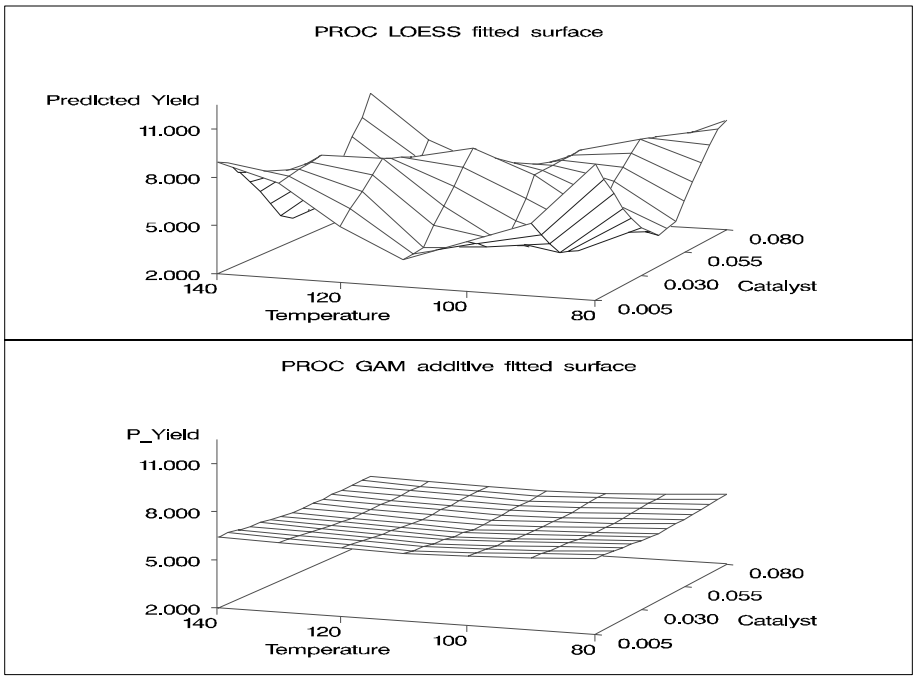
Output 5.4.4. Raw Data from Experiment B



Though the surface displayed in Output 5.4.4 is quite jagged, a distinct feature of the plot is a large ridge that runs diagonally across its surface. One would expect that the ridge would appear in the fitted regression surface of an appropriate nonparametric procedure. Nevertheless, between PROC LOESS and PROC GAM, only PROC LOESS is able to capture this significant feature.

The SAS code for fitting the new data is essentially the same as that for the data set from the first experiment. Similar to Output 5.4.2, multivariate and additive fitted surfaces for these data are displayed in Output 5.4.5.

Output 5.4.5. Fitted Regression Surfaces

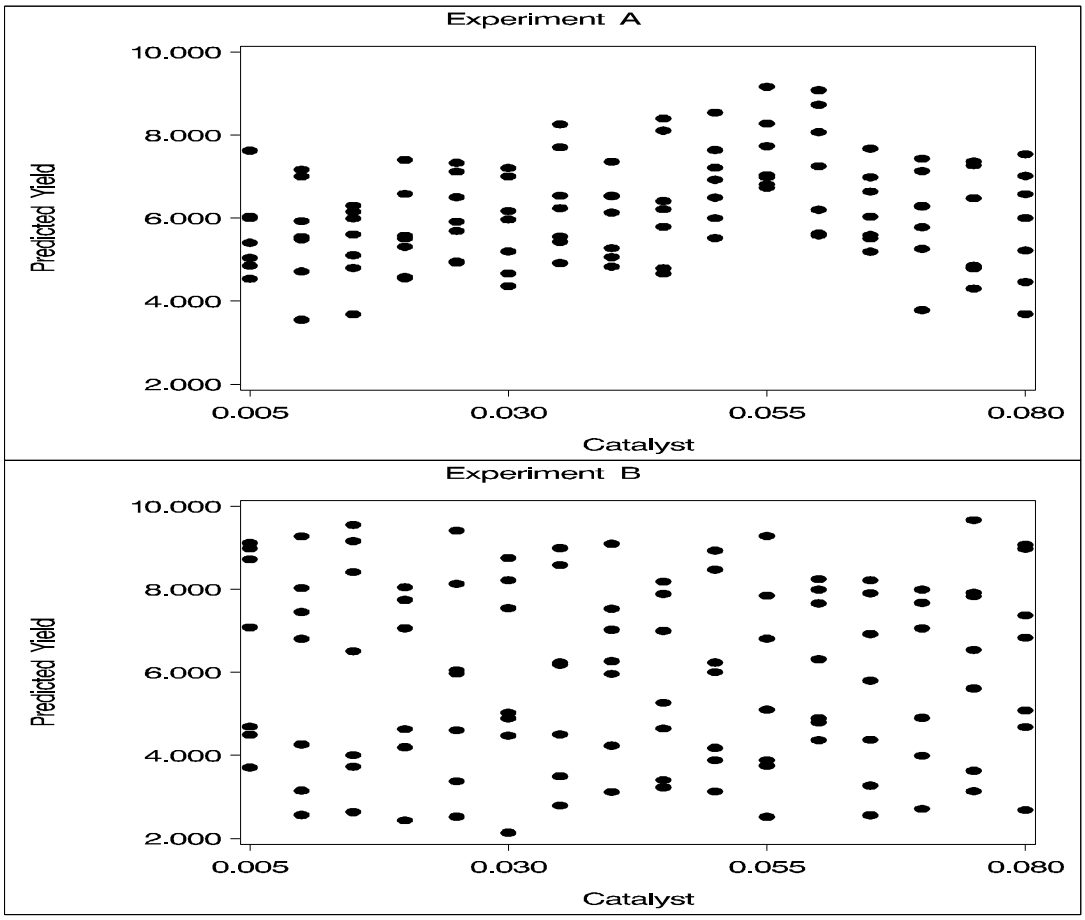


It is clear from Output 5.4.5 that the results of PROC LOESS and PROC GAM are completely different. While the plot in the top panel resembles the raw data plot in Output 5.4.4, the plot in the bottom panel is essentially featureless.

To understand what is happening, compare the scatter plots of Yield by Catalyst for the two data sets in this example. These are generated by the following code and displayed in Output 5.4.6.

```
axis1 minor=none order=(2 to 11 by 2)
      label=(angle=90 rotate=0 "Predicted Yield");
axis2 minor=none order=(0.005 to 0.080 by 0.025) label=("Catalyst");
symbol2 c=yellow v=dot i=none;

title2 'Experiment A';
proc gplot data=ExperimentA;
   plot Yield*Catalyst=2
        / cframe=ligr name='ExptA'
          vaxis=axis1 haxis=axis2;
run; quit;

title2 'Experiment B';
proc gplot data=ExperimentB;
     plot Yield*Catalyst=2
          / cframe=ligr name='ExptB'
            vaxis=axis1 haxis=axis2;
run; quit;

proc greplay nofs tc=sashelp.templt template=v2;
     igout=gseg;
     treplay 1:expta 2:exptb;
run; quit;
```

Output 5.4.6. Scatterplots of Yield by Catalyst



The top panel of Output 5.4.6 hints at the same kind of structure exhibited in the fitted cross-sections of Output 5.4.3. In PROC GAM, the additive model component corresponding to Catalyst is fit to a similar scatter plot, with the partial residuals computed in the backfitting algorithm, so it is able to capture the trend seen here. In contrast, when the second data set is viewed from the perspective of Output 5.4.6, the diagonal ridge apparent in Output 5.4.4 is washed out, and no clear structure shows up in the scatter plot. As a result, the additive model fit produced by PROC GAM is relatively featureless.

References

Allen, D.M. (1974), "The Relationship Between Variable Selection and Data Augmentation and a Method of Prediction," *Technometrics*, 16, 125–127.

Bell, D., Walker, J., O'Connor, G., Orrel, J. and Tibshirani, R. (1989), "Spinal Deformation Following Multi-Level Thoracic and Lumbar Laminectomy in Children." Submitted for publication.

Cleveland, W.S., Devlin, S.J., and Grosse, E. (1988), "Regression by Local Fitting," *Journal of Econometrics*, 37, 87–114.

Duchon, J. (1976), "Fonctions-Spline et Esperances Conditionnelles de Champs Gaussiens," *Ann. Sci. Univ. Clermont Ferrand II Math.*, 14, 19–27.

Duchon, J. (1977), "Splines Minimizing Rotation-Invariant Semi-Norms in Sovolev Spaces," in *Constructive Theory of Functions of Several Variables*, ed. W. Schempp and K. Zeller, New York: Springer-Verlag, 85–100.

Friedman, J.H. and Stuetzle, W. (1981), "Projection Pursuit Regression," *Journal of the American Statistical Association*, 76, 817–823.

Hastie, T.J. and Tibshirani, R.J. (1986), "Generalized Additive Models (with discussion)," *Statistical Science*, 1, 297–318.

Hastie, T.J. and Tibshirani, R.J. (1990), *Generalized Additive Models*, New York: Chapman and Hall.

Houghton, A.N., Flannery, J., and Viola, M.V. (1980), "Malignant Melanoma in Connecticut and Denmark," *International Journal of Cancer*, 25, 95–104.

Meinguet, J. (1979), "Multivariate Interpolation at Arbitrary Points Made Simple," *Journal of Applied Mathematics and Physics (ZAMP)*, 30, 292–304.

Nelder, J.A. and Wedderburn, R.W.M. (1972), "Generalized Linear Models," *Journal of the Royal Statistical Society, Series A*, 135, 370–384.

SAS Institute Inc. (1999), *SAS Language Reference: Concepts, Version 8*, Cary, NC: SAS Institute Inc.

SAS Institute Inc. (1999), *SAS Language Reference: Dictionary, Version 8*, Cary, NC: SAS Institute Inc.

SAS Institute Inc. (1999), *SAS Procedures Guide, Version 8*, Cary, NC: SAS Institute Inc.

SAS Institute Inc. (1999), *SAS/STAT User's Guide, Version 8*, Cary, NC: SAS Institute Inc.

Sockett, E.B., Daneman, D., Clarson, C., and Ehrich, R.M. (1987), "Factors Affecting and Patterns of Residual Insulin Secretion During the First Year of Type I (Insulin Dependent) Diabetes Mellitus in Children," *Diabet*, 30, 453–459.

Stone, C.J. (1985), "Additive Regression and Other Nonparametric Models," *Annals of Statistics*, 13, 689–705.

Wahba, G. (1983), "Bayesian 'Confidence Intervals' for the Cross Validated Smoothing Spline," *Journal of the Royal Statistical Society, Series B*, 45, 133–150.

Wahba, G. (1990), *Spline Models for Observational Data*, Philadelphia: Society for Industrial and Applied Mathematics.

Wahba, G. and Wendelberger, J. (1980), "Some New Mathematical Methods for Variational Objective Analysis Using Splines and Cross Validation," *Monthly Weather Review*, 108, 1122–1145.

Chapter 6
The LIFEREG Procedure

Chapter Table of Contents

Chapter 6
The LIFEREG Procedure

Overview

You can specify interaction terms among any of the specified explanatory variables in the MODEL statement in the same way as in the GLM procedure. Because of the more complicated models you can specify, only a single MODEL statement is now allowed for each PROC LIFEREG statement. If more than one MODEL statement is specified, only the last one is used.

You can use the new PROBPLOT statement to construct probability plots for complete or censored data. You can display the fitted distribution and confidence intervals along with nonparametric estimates of the CDF on the probability plots. In addition, you can control graphical features such as plot layout, colors, plotting symbols, line styles, and the text fonts used in the probability plot.

You can use the new XDATA= data set to provide the values for effects when the predicted probability and quantile and their confidence limits are computed. By default, the LIFEREG procedure uses the overall mean for effects containing continuous a variable (or variables) and the highest level of a single classification variable as the reference level.

You can use the new INEST= data set to provide the initial values for parameters in the model. The structure of this data set is similar to that of the OUTEST= data set. This data set will overwrite the original INITIAL= option in the MODEL statement.

The OUTEST= data set is extended to all models whether there is a CLASS statement or not. The OUTEST= data set does not include the fixed scale nor the fixed shape parameter.

You can now obtain standardized and Cox-Snell residuals in the OUPTUT= data set.

Syntax

PROC LIFEREG Statement

PROC LIFEREG < *options* > **;**

GOUT=*graphics-catalog*
specifies a graphics catalog in which to save graphics output.

INEST= *SAS-data-set*
specifies an input SAS data set that contains initial estimates for all the parameters in

the model. See the section "INEST= Data Set" on page 92 for a detailed description of the contents of the INEST= data set.

NAMELEN=*n*
specifies the length of effect names in tables and output data sets to be n characters, where n is a value between 20 and 200. The default length is 20 characters.

XDATA= *SAS-data-set*
specifies an input SAS data set that contains values for all the independent variables in the MODEL statement and variables in the CLASS statement for probability plotting. If there are covariates specified in a MODEL statement and a probability plot is requested with a PROBPLOT statement, you specify fixed values for the effects in the MODEL statement with the XDATA= data set. See the section "XDATA= Data Set" on page 93 for a detailed description of the contents of the XDATA= data set.

INSET Statement

INSET < *keyword-list* > < / *options*>**;**

The box or table of summary information produced on plots made with the PROBPLOT statement is called an *inset*. You can use the INSET statement to customize the information that is printed in the inset box as well as to customize the appearance of the inset box. To supply the information that is displayed in the inset box, you specify *keywords* corresponding to the information that you want shown. For example, the following statements produce a probability plot with the number of observations, the number of right-censored observations, the name of the distribution, and the estimated Weibull shape parameter in the inset.

```
proc lifereg data=epidemic;
   model life = dose / dist = Weibull;
   probplot ;
   inset nobs right dist shape;
run;
```

By default, inset entries are identified with appropriate labels. However, you can provide a customized label by specifying the *keyword* for that entry followed by the equal sign (=) and the label in quotes. For example, the following INSET statement produces an inset containing the number of observations and the name of the distribution, labeled "Sample Size" and "Distribution" in the inset.

```
inset nobs='Sample Size' dist='Distribution';
```

If you specify a keyword that does not apply to the plot you are creating, then the keyword is ignored.

If you specify more than one INSET statement, only the first one is used.

The following table lists keywords available in the INSET statement to display summary statistics, distribution parameters, and distribution fitting information.

Table 6.1. INSET Statement Keywords

CONFIDENCE	confidence coefficient for all confidence intervals
DIST	name of the distribution
INTERVAL	number of interval-censored observations
LEFT	number of left-censored observations
NOBS	number of observations
NMISS	number of observations with missing values
RIGHT	number of right-censored observations
SCALE	value of the scale parameter
SHAPE	value of the shape parameter
UNCENSORED	number of uncensored observations

The following *options* control the appearance of the box. All *options* are specified after the slash (/) in the INSET statement.

CFILL=*color*
specifies the color for the filling box.

CFILLH=*color*
specifies the color for the filling box header.

CFRAME=*color*
specifies the color for the frame.

CHEADER=*color*
specifies the color for text in the header.

CTEXT=*color*
specifies the color for the text.

FONT=*font*
specifies the software font for the text.

HEIGHT=*value*
specifies the height of the text.

HEADER=*'quoted string'*
specifies text for the header or box title.

NOFRAME
omits the frame around the box.

POS= *value* <**DATA | PERCENT**>
determines the position of the inset. The *value* can be a compass point (N, NE, E, SE, S, SW, W, NW) or a pair of coordinates (x, y) enclosed in parentheses. The coordinates can be specified in screen percent units or axis data units. The default is screen percent units.

REFPOINT= *name*
specifies the reference point for an inset that is positioned by a pair of coordinates

with the POS= option. You use the REFPOINT= option in conjunction with the POS= coordinates. The REFPOINT= option specifies which corner of the inset frame you have specified with coordinates (x, y), and it can take the value of BR (bottom right), BL (bottom left), TR (top right), or TL (top left). The default is REFPOINT=BL. If the inset position is specified as a compass point, then the REFPOINT= option is ignored.

MODEL Statement

<label:> **MODEL** *response<***censor(list)>=effects* *< / options >* **;**

<label:> **MODEL** *(lower,upper)=effects* *< / options >* **;**

<label:> **MODEL** *events/trials=effects* *< / options >* **;**

Only a single MODEL statement can be used with one invocation of the LIFEREG procedure. If multiple MODEL statements are present, only the last is used. The optional *label* is used to label the model estimates in the output SAS data set and OUTEST= data set.

ALPHA=*value*
sets the significance level for the confidence intervals for regression parameters and estimated survival probabilities. The value must be between 0 and 1. By default, ALPHA = 0.05.

CONVERGE=*value*
The default is changed to CONVERGE=1E−8.

OUTPUT Statement

OUTPUT *<***OUT=***SAS-data-set>* *keyword***=***name* *<. . .keyword***=***name>* **;**

Two new keywords are available.

CRESIDUAL | CRES specifies a variable to contain the Cox-Snell residuals

$$-\log(S(w_i))$$

where S is the standard survival function and

$$w_i = \frac{t_i - \mathbf{x}_i'\mathbf{b}}{\sigma}$$

If the response variable in the corresponding model statement is binomial, then the residuals are not computed, and this variable contains missing values.

SRESIDUAL | SRES specifies a variable to contain the standardized residuals

$$\frac{t_i - \mathbf{x}_i'\mathbf{b}}{\sigma}$$

If the response variable in the corresponding model statement is binomial, then the residuals are not computed, and this variable contains missing values.

PROBPLOT Statement

PROBPLOT | PPLOT < */ options* > **;**

You can use the PROBPLOT statement to create a probability plot from lifetime data. The data can be uncensored, right-censored, or arbitrarily censored. You can specify any number of PROBPLOT statements after a MODEL statement. The syntax used for the response in the MODEL statement determines the type of censoring assumed in creating the probability plot. The model fit with the MODEL statement is plotted along with the data. If there are covariates in the model, they are set to constant values specified in the XDATA= data set when creating the probability plot. If no XDATA= data set is specified, continuous variables are set to their overall mean values and categorical variables specified in the CLASS statement are set to their highest levels.

You can specify the following options to control the content, layout, and appearance of a probability plot.

ANNOTATE=*SAS-data-set*
ANNO=*SAS-data-set*
specifies an ANNOTATE data set, as described in *SAS/GRAPH Software: Reference*, that enables you to add features to the probability plot. The ANNOTATE= data set you specify in the PROBPLOT statement is used for all plots created by the statement.

CAXIS=*color*
CAXES=*color*
specifies the color used for the axes and tick marks. This option overrides any COLOR= specifications in an AXIS statement. The default is the first color in the device color list.

CCENSOE=*color*
specifies the color for filling the censor plot area. The default is the first color in the device color list.

CENBIN
plots censored data as frequency counts (rounding-off for non-integer frequency) rather than as individual points.

CENCOLOR=*color*
specifies the color for the censor symbol. The default is the first color in the device color list.

CENSYMBOL=*symbol* **| (***symbol list***)**
specifies symbols for censored values. The *symbol* is one of the symbol names (plus, star, square, diamond, triangle, hash, paw, point, dot, and circle) or a letter (A–Z). If you do not specify the CENSYMBOL= option, the symbol used for censored values is the same as for failures.

CFIT=*color*
specifies the color for the fitted probability line and confidence curves. The default is the first color in the device color list.

CFRAME=*color*
CFR=*color*
specifies the color for the area enclosed by the axes and frame. This area is not shaded by default.

CGRID=*color*
specifies the color for grid lines. The default is the first color in the device color list.

CHREF=*color*
CH=*color*
specifies the color for lines requested by the HREF= option. The default is the first color in the device color list.

CTEXT=*color*
specifies the color for tick mark values and axis labels. The default is the color specified for the CTEXT= option in the most recent GOPTIONS statement.

CVREF=*color*
CV=*color*
specifies the color for lines requested by the VREF= option. The default is the first color in the device color list.

DESCRIPTION='*string***'**
DES='*string***'**
specifies a description, up to 40 characters, that appears in the PROC GREPLAY master menu. The default is the variable name.

FONT=*font*
specifies a software font for reference line and axis labels. You can also specify fonts for axis labels in an AXIS statement. The FONT= font takes precedence over the FTEXT= font specified in the most recent GOPTIONS statement. Hardware characters are used by default.

HCL
computes and draws confidence limits for the predicted probabilities in the horizontal direction.

HEIGHT=*value*
specifies the height of text used outside framed areas. The default value is 3.846 (in percentage).

HLOWER=*value*
specifies the lower limit on the lifetime axis scale. The HLOWER= option specifies

value as the lower lifetime axis tick mark. The tick mark interval and the upper axis limit are determined automatically.

HOFFSET=*value*
specifies the offset for the horizontal axis. The default value is 1.

HUPPER=*value*
specifies *value* as the upper lifetime axis tick mark. The tick mark interval and the lower axis limit are determined automatically.

HREF < (INTERSECT) > =*value-list*
requests reference lines perpendicular to the horizontal axis. If (INTERSECT) is specified, a second reference line perpendicular to the vertical axis is drawn that intersects the fit line at the same point as the horizontal axis reference line. If a horizontal axis reference line label is specified, the intersecting vertical axis reference line is labeled with the vertical axis value. See also the CHREF=, HREFLABELS=, and LHREF= options.

HREFLABELS=*'label1' . . . 'labeln'*
HREFLABEL=*'label1' . . . 'labeln'*
HREFLAB=*'label1' . . . 'labeln'*
specifies labels for the lines requested by the HREF= option. The number of labels must equal the number of lines. Enclose each label in quotes. Labels can be up to 16 characters.

HREFLABPOS=*n*
specifies the vertical position of labels for HREF= lines. The following table shows the valid values for *n* and the corresponding label placements.

n	**label placement**
1	top
2	staggered from top
3	bottom
4	staggered from bottom
5	alternating from top
6	alternating from bottom

INBORDER
requests a border around probability plots.

INTERTILE=*value*
specifies the distance between tiles.

ITPRINTEM
displays the iteration history for the Turnbull algorithm.

JITTER=*value*
specifies the amount to jitter overlaying plot symbols, in units of symbol width.

LFIT=*linetype*
specifies a line style for fitted curves and confidence limits. By default, fitted curves are drawn by connecting solid lines (*linetype = 1*), and confidence limits are drawn by connecting dashed lines (*linetype = 3*).

LGRID=*linetype*
specifies a line style for all grid lines. *linetype* is between 1 and 46. The default is 35.

LHREF=*linetype*
LH=*linetype*
specifies the line type for lines requested by the HREF= option. The default is 2, which produces a dashed line.

LVREF=*linetype*
LV=*linetype*
specifies the line type for lines requested by the VREF= option. The default is 2, which produces a dashed line.

MAXITEM=*n1 <,n2> n1*
specifies the maximum number of iterations allowed for the Turnbull algorithm. Iteration history will be printed in increments of *n2* if requested with the ITPRINTEM option. See the section "Arbitrarily Censored Data" on page 90 for details.

NAME=*'string'*
specifies a name for the plot, up to eight characters, that appears in the PROC GREPLAY master menu. The default is 'LIFEREG'.

NOCENPLOT
suppresses the plotting of censored data points.

NOCONF
suppresses the default percentile confidence bands on the probability plot.

NODATA
suppresses plotting of the estimated empirical probability plot.

NOFIT
suppresses the fitted probability (percentile) line and confidence bands.

NOFRAME
suppresses the frame around plotting areas.

NOGRID
suppresses grid lines.

NOHLABEL
suppresses horizontal labels.

NOHTICK
suppresses horizontal tick marks.

NOPOLISH
suppresses setting small interval probabilities to zero in the Turnbull algorithm.

NOVLABEL
suppresses vertical labels.

NOVTICK
suppresses vertical tick marks.

NPINTERVALS=*interval type*
specifies one of the two kinds of confidence limits for the estimated cumulative probabilities, pointwise (NPINTERVALS=POINT) or simultaneous (NPINTERVALS=SIMUL), requested by the PPOUT option to be printed in the tabular output.

PCTLIST=*value-list*
specifies the list of percentages for which to compute percentile estimates. *value-list* must be a list of values separated by blanks or commas. Each value in the list must be between 0 and 100.

PLOWER=*value*
specifies the lower limit on the probability axis scale. The PLOWER= option specifies *value* as the lower probability axis tick mark. The tick mark interval and the upper axis limit are determined automatically.

PRINTPROBS
prints intervals and associated probabilities for the Turnbull algorithm.

PUPPER=*value*
specifies the upper limit on the probability axis scale. The PUPPER= option specifies *value* as the upper probability axis tick mark. The tick mark interval and the lower axis limit are determined automatically.

PPOS=*character-list*
specifies the plotting position type. See the section "Probability Plotting" on page 88 for details.

PPOS	Method
EXPRANK	expected ranks
MEDRANK	median ranks
MEDRANK1	median ranks (exact formula)
KM	Kaplan-Meier
MKM	modified Kaplan-Meier (default)

PPOUT
specifies that a table of the cumulative probabilities plotted on the probability plot be printed. Kaplan-Meier estimates of the cumulative probabilities are also printed, along with standard errors and confidence limits. The confidence limits can be pointwise or simultaneous, as specified by the NPINTERVALS= option.

PROBLIST=*value-list*
specifies the list of initial values for the Turnbull algorithm.

ROTATE
requests probability plots with probability scale on the horizontal axis.

SQUARE
makes the layout of the probability plots square.

TOLLIKE=*value*
specifies the criterion for convergence in the Turnbull algorithm.

TOLPROB=*value*
specifies the criterion for setting the interval probability to zero in the Turnbull algorithm.

VAXISLABEL='*string*'
specifies a label for the vertical axis.

VREF=*value-list*
requests reference lines perpendicular to the vertical axis. If (INTERSECT) is specified, a second reference line perpendicular to the horizontal axis is drawn that intersects the fit line at the same point as the vertical axis reference line. If a vertical axis reference line label is specified, the intersecting horizontal axis reference line is labeled with the horizontal axis value. See also the entries for the CVREF=, LVREF=, and VREFLABELS= options.

VREFLABELS=*'label1' . . . 'labeln'*
VREFLABEL=*'label1' . . . 'labeln'*
VREFLAB=*'label1' . . . 'labeln'*
specifies labels for the lines requested by the VREF= option. The number of labels must equal the number of lines. Enclose each label in quotes. Labels can be up to 16 characters.

VREFLABPOS=*n*
specifies the horizontal position of labels for VREF= lines. The valid values for *n* and the corresponding label placements are shown in the following table.

n	**label placement**
1	left
2	right

WAXIS=*n*
specifies line thickness for axes and frame. The default value is 1.

WFIT=*n*
specifies line thickness for fitted curves. The default value is 1.

WGRID=*n*
specifies line thickness for grids. The default value is 1.

WREFL=*n*
specifies line thickness for reference lines. The default value is 1.

Details

Confidence Intervals

Confidence intervals are computed for all model parameters and are reported in the "Analysis of Parameter Estimates" table. The confidence coefficient can be specified with the ALPHA=α MODEL statement option, resulting in a $(1-\alpha)\times 100\%$ two-sided confidence coefficient. The default confidence coefficient is 95%, corresponding to $\alpha = .05$.

Regression Parameters

A two-sided $(1-\alpha)\times 100\%$ confidence interval $[\beta_{iL},\ \ \beta_{iU}]$ for the regression parameter β_i is based on the asymptotic normality of the maximum likelihood estimator $\hat{\beta}_i$ and is computed by

$$\beta_{iL} = \hat{\beta}_i - z_{1-\alpha/2}(\mathrm{SE}_{\hat{\beta}_i})$$

$$\beta_{iU} = \hat{\beta}_i + z_{1-\alpha/2}(\mathrm{SE}_{\hat{\beta}_i})$$

where $\mathrm{SE}_{\hat{\beta}_i}$ is the estimated standard error of $\hat{\beta}_i$, and z_p is the $p\times 100\%$ percentile of the standard normal distribution.

Scale Parameter

A two-sided $(1-\alpha)\times 100\%$ confidence interval $[\sigma_L,\ \ \sigma_U]$ for the scale parameter σ in the location-scale model is based on the asymptotic normality of the logarithm of the maximum-likelihood estimator $\log(\hat{\sigma})$, and is computed by

$$\sigma_L = \hat{\sigma}/\exp[z_{1-\alpha/2}(\mathrm{SE}_{\hat{\sigma}})/\hat{\sigma}]$$

$$\sigma_U = \hat{\sigma}\exp[z_{1-\alpha/2}(\mathrm{SE}_{\hat{\sigma}})/\hat{\sigma}]$$

Refer to Meeker and Escobar (1998) for more information.

Weibull Scale and Shape Parameters

The Weibull distribution scale parameter η and shape parameter β are obtained by transforming the extreme value location parameter μ and scale parameter σ:

$$\eta = \exp(\mu)$$

$$\beta = 1/\sigma$$

Consequently, two-sided $(1-\alpha)\times 100\%$ confidence intervals for the Weibull scale and shape parameters are computed as

$$[\eta_L,\ \ \eta_U] = [\exp(\mu_L),\ \ \exp(\mu_U)]$$

$$[\beta_L,\ \ \beta_U] = [1/\sigma_U,\ \ 1/\sigma_L]$$

Gamma Shape Parameter

A two-sided $(1-\alpha)\times 100\%$ confidence interval for the 3-parameter gamma shape parameter δ is computed by

$$[\delta_L,\ \ \delta_U] = [\hat{\delta} - z_{1-\alpha/2}(\mathrm{SE}_{\hat{\delta}}),\ \ \hat{\delta} + z_{1-\alpha/2}(\mathrm{SE}_{\hat{\delta}})]$$

Probability Plotting

Probability plots are useful tools for the display and analysis of lifetime data. Probability plots use an inverse distribution scale so that a cumulative distribution function (CDF) plots as a straight line. A nonparametric estimate of the CDF of the lifetime data will plot approximately as a straight line, thus providing a visual assessment of goodness-of-fit.

You can use the PROBPLOT statement in LIFEREG to create probability plots of data that are complete, right-censored, interval-censored, or a combination of censoring types (arbitrarily censored). A line representing the maximum likelihood fit from the MODEL statement and pointwise parametric confidence bands for the cumulative probabilities are also included on the plot.

A random variable Y belongs to a *location-scale* family of distributions if its CDF F is of the form

$$Pr\{Y \le y\} = F(y) = G\left(\frac{y-\mu}{\sigma}\right)$$

where μ is the location parameter and σ is the scale parameter. Here, G is a CDF that cannot depend on any unknown parameters, and G is the CDF of Y if $\mu = 0$ and $\sigma = 1$. For example, if Y is a normal random variable with mean μ and standard deviation σ,

$$G(u) = \Phi(u) = \int_{-\infty}^{u} \frac{1}{\sqrt{2\pi}} \exp(-\frac{u^2}{2})\, du$$

and

$$F(y) = \Phi\left(\frac{y-\mu}{\sigma}\right)$$

The normal, extreme value, and logistic distributions are location-scale models. The 3-parameter gamma distribution is a location-scale model if the shape parameter δ is fixed. If T has a lognormal, Weibull, or log-logistic distribution, then $\log(T)$ has a distribution that is a location-scale model. Probability plots are constructed for lognormal, Weibull, and log-logistic distributions by using $\log(T)$ instead of T in the plots.

Let $y_{(1)} \le y_{(2)} \le \ldots \le y_{(n)}$ be ordered observations of a random sample with distribution function $F(y)$. A probability plot is a plot of the points $y_{(i)}$ against $m_i = G^{-1}(a_i)$, where $a_i = \hat{F}(y_i)$ is an estimate of the CDF $F(y_{(i)}) = G\left(\frac{y_{(i)}-\mu}{\sigma}\right)$.

The nonparametric CDF estimates a_i are sometimes called *plotting positions*. The axis on which the points m_i are plotted is usually labeled with a probability scale (the scale of a_i).

If F is one of the location-scale distributions, then y is the lifetime; otherwise, the log of the lifetime is used to transform the distribution to a location-scale model.

If the data actually have the stated distribution, then $\hat{F} \approx F$,

$$m_i = G^{-1}(\hat{F}(y_i)) \approx G^{-1}(G\left(\frac{y_{(i)} - \mu}{\sigma}\right)) = \frac{y_{(i)} - \mu}{\sigma}$$

and points $(y_{(i)}, m_i)$ should fall approximately on a straight line.

There are several ways to compute the nonparametric CDF estimates used in probability plots from lifetime data. These are discussed in the next two sections.

Complete and Right-Censored Data

The censoring times must be taken into account when you compute plotting positions for right-censored data. The modified Kaplan-Meier method described in the following section is the default method for computing nonparametric CDF estimates for display on probability plots. Refer to Abernethy (1996), Meeker and Escobar (1998), and Nelson (1982) for discussions of the methods described in the following sections.

Expected Ranks, Kaplan-Meier, and Modified Kaplan-Meier Methods

Let $y_{(1)} \leq y_{(2)} \leq \ldots \leq y_{(n)}$ be ordered observations of a random sample including failure times and censor times. Order the data in increasing order. Label all the data with reverse ranks r_i, with $r_1 = n, \ldots, r_n = 1$. For the lifetime (not censoring time) corresponding to reverse rank r_i, compute the survival function estimate

$$S_i = \left[\frac{r_i}{r_i + 1}\right] S_{i-1}$$

with $S_0 = 1$. The expected rank plotting position is computed as $a_i = 1 - S_i$. The option PPOS=EXPRANK specifies the expected rank plotting position.

For the Kaplan-Meier method,

$$S_i = \left[\frac{r_i - 1}{r_i}\right] S_{i-1}$$

The Kaplan-Meier plotting position is then computed as $a'_i = 1 - S_i$. The option PPOS=KM specifies the Kaplan-Meier plotting position.

For the modified Kaplan-Meier method, use

$$S'_i = \frac{S_i + S_{i-1}}{2}$$

where S_i is computed from the Kaplan-Meier formula with $S_0 = 1$. The plotting position is then computed as $a''_i = 1 - S'_i$. The option PPOS=MKM specifies the

modified Kaplan-Meier plotting position. If the PPOS option is not specified, the modified Kaplan-Meier plotting position is used as the default method.

For complete samples, $a_i = i/(n+1)$ for the expected rank method, $a_i' = i/n$ for the Kaplan-Meier method, and $a_i'' = (i - .5)/n$ for the modified Kaplan-Meier method. If the largest observation is a failure for the Kaplan-Meier estimator, then $F_n = 1$ and the point is not plotted.

Median Ranks

Let $y_{(1)} \leq y_{(2)} \leq \ldots \leq y_{(n)}$ be ordered observations of a random sample including failure times and censor times. A failure order number j_i is assigned to the *i*th failure: $j_i = j_{i-1} + \Delta$, where $j_0 = 0$. The increment Δ is initially 1 and is modified when a censoring time is encountered in the ordered sample. The new increment is computed as

$$\Delta = \frac{(n+1) - \text{ previous failure order number}}{1 + \text{ number of items beyond previous censored item}}$$

The plotting position is computed for the ith failure time as

$$a_i = \frac{j_i - .3}{n + .4}$$

For complete samples, the failure order number j_i is equal to i, the order of the failure in the sample. In this case, the preceding equation for a_i is an approximation to the median plotting position computed as the median of the *i*th-order statistic from the uniform distribution on (0, 1). In the censored case, j_i is not necessarily an integer, but the preceding equation still provides an approximation to the median plotting position. The PPOS=MEDRANK option specifies the median rank plotting position.

Arbitrarily Censored Data

The LIFEREG procedure can create probability plots for data that consists of combinations of exact, left-censored, right-censored, and interval-censored lifetimes, that is, arbitrarily censored data. The LIFEREG procedure uses an iterative algorithm developed by Turnbull (1976) to compute a nonparametric maximum likelihood estimate of the cumulative distribution function for the data. Since the technique is maximum likelihood, standard errors of the cumulative probability estimates are computed from the inverse of the associated Fisher information matrix. This algorithm is an example of the expectation-maximization (EM) algorithm. The default initial estimate assigns equal probabilities to each interval. You can specify different initial values with the PROBLIST= option. Convergence is determined if the change in the log likelihood between two successive iterations is less than delta, where the default value of delta is 10^{-8}. You can specify a different value for delta with the TOLLIKE= option. Iterations will be terminated if the algorithm does not converge after a fixed number of iterations. The default maximum number of iterations is 1000. Some data may require more iterations for convergence. You can specify the maximum allowed number of iterations with the MAXITEM= option in the PROBPLOT statement. The iteration history of the log likelihood is displayed if you specify the ITPRINTEM option. The iteration history of the estimated interval probabilities are also displayed if you specify both options ITPRINTEM and PRINTPROBS.

If an interval probability is smaller than a tolerance (10^{-6} by default) after convergence, the probability is set to zero, the interval probabilities are renormalized so that they add to one, and iterations are restarted. Usually the algorithm converges in just a few more iterations. You can change the default value of the tolerance with the TOLPROB= option. You can specify the NOPOLISH option to avoid setting small probabilities to zero and restarting the algorithm.

If you specify the ITPRINTEM option, a table summarizing the Turnbull estimate of the interval probabilities is displayed. The columns labeled "Reduced Gradient" and "Lagrange Multiplier" are used in checking final convergence of the maximum likelihood estimate. The Lagrange multipliers must all be greater than or equal to zero, or the solution is not maximum likelihood. Refer to Gentleman and Geyer (1994) for more details of the convergence checking. Also refer to Meeker and Escobar (1998, chap. 3) for more information.

See Example 6.3 on page 100 for an illustration.

Nonparametric Confidence Intervals

You can use the PPOUT option in the PROBPLOT statement to create a table containing the nonparametric CDF estimates computed by the selected method, Kaplan-Meier CDF estimates, standard errors of the Kaplan-Meier estimator, and nonparametric confidence limits for the CDF. The confidence limits are either pointwise or simultaneous, depending on the value of the NPINTERVALS= option in the PROBPLOT statement. The method used in the LIFEREG procedure for computation of approximate pointwise and simultaneous confidence intervals for cumulative failure probabilities relies on the Kaplan-Meier estimator of the cumulative distribution function of failure time and approximate standard deviation of the Kaplan-Meier estimator. For the case of arbitrarily censored data, the Turnbull algorithm, discussed previously, provides an extension of the Kaplan-Meier estimator. Both the Kaplan-Meier and the Turnbull estimators provide an estimate of the standard error of the CDF estimator, $\mathrm{se}_{\hat{F}}$, that is used in computing confidence intervals.

Pointwise Confidence Intervals

Approximate $(1-\alpha)100\%$ pointwise confidence intervals are computed as in Meeker and Escobar (1998, section 3.6) as

$$[F_L, \quad F_U] = \left[\frac{\hat{F}}{\hat{F} + (1-\hat{F})w}, \quad \frac{\hat{F}}{\hat{F} + (1-\hat{F})/w} \right]$$

where

$$w = \exp\left[\frac{z_{1-\alpha/2}\mathrm{se}_{\hat{F}}}{(\hat{F}(1-\hat{F}))} \right]$$

where z_p is the pth quantile of the standard normal distribution.

Simultaneous Confidence Intervals

Approximate $(1-\alpha)100\%$ simultaneous confidence bands valid over the lifetime interval (t_a, t_b) are computed as the "Equal Precision" case of Nair (1984) and Meeker

and Escobar (1998, section 3.8) as

$$[F_L,\ \ F_U] = \left[\frac{\hat{F}}{\hat{F} + (1-\hat{F})w},\ \ \frac{\hat{F}}{\hat{F} + (1-\hat{F})/w}\right]$$

where

$$w = \exp\left[\frac{e_{a,b,1-\alpha/2}\mathrm{se}_{\hat{F}}}{(\hat{F}(1-\hat{F}))}\right]$$

where the factor $x = e_{a,b,1-\alpha/2}$ is the solution of

$$x\exp(-x^2/2)\log\left[\frac{(1-a)b}{(1-b)a}\right]/\sqrt{8\pi} = \alpha/2$$

The time interval (t_a, t_b) over which the bands are valid depends in a complicated way on the constants a and b defined in Nair (1984), $0 < a < b < 1$. a and b are chosen by default so that the confidence bands are valid between the lowest and highest times corresponding to failures in the case of multiply censored data, or, to the lowest and highest intervals for which probabilities are computed for arbitrarily censored data. You can optionally specify a and b directly with the NPINTERVALS=SIMULTANEOUS(a, b) option in the PROBPLOT statement.

INEST= Data Set

If specified, the INEST= data set specifies initial estimates for all the parameters in the model. The INEST= data set must contain the intercept variable (named Intercept) and all independent variables in the MODEL statement.

If BY processing is used, the INEST= data set should also include the BY variables, and there must be at least one observation for each BY group. If there is more than one observation in one BY group, the first one read is used for that BY group.

If the INEST= data set also contains the _TYPE_ variable, only observations with _TYPE_ value 'PARMS' are used as starting values. Combining the INEST= data set and the option MAXITER= in the MODEL statement, partial scoring can be done, such as predicting on a validation data set by using the model built from a training data set.

You can specify starting values for the iterative algorithm in the INEST= data set. This data set overwrites the INITIAL= option in the MODEL statement, which is a little difficult to use for models including multilevel interaction effects. The INEST= data set has the same structure as the OUTEST= data set but is not required to have all the variables or observations that appear in the OUTEST= data set. One simple use of the INEST= option is passing the previous OUTEST= data set directly to the next model as an INEST= data set, assuming that the two models have the same parameterization.

XDATA= Data Set

The XDATA= data set is used for plotting the predicted probability when there are covariates specified in a MODEL statement and a probability plot is specified with a PROBPLOT statement. See Example 6.1 on page 94 for an illustration.

The XDATA= data set is an input SAS data set that contains values for all the independent variables in the MODEL statement and variables in the CLASS statement.The XDATA= data set has the same structure as the DATA= data set but is not required to have all the variables or observations that appear in the DATA= data set.

The XDATA= data set must contain all the independent variables in the MODEL statement and variables in the CLASS statement. Even though variables in the CLASS statement may not be used, valid values are required for these variables in the XDATA= data set. Missing values are not allowed. Missing values are not allowed in the XDATA= data set for any of the independent variables either. Missing values are allowed for the dependent variables and other variables if they are included in the XDATA= data set.

If BY processing is used, the XDATA= data set should also include the BY variables, and there must be at least one valid observation for each BY group. If there is more than one valid observation in a BY group, the last one read is used for that BY group.

If there is no XDATA= data set in the PROC LIFEREG statement, by default, the LIFEREG procedure will use the overall mean for effects containing a continuous variable (or variables) and the highest level of a single classification variable as reference level. The rules are summarized as follows:

- If the effect contains a continuous variable (or variables), the overall mean of this effect (not the variables) is used.
- If the effect is a single classification variable, the highest level of the variable is used.

Examples

Example 6.1. Analysis of Arbitrarily Censored Data with Interaction Effects

The following artificial data are for a study of the natural recovery time of mice after injection of a certain toxin. 20 mice were grouped by sex (sex: 1 = Male, 2 = Female) with equal sizes. Their ages (in days) were recorded at the injection. Their recovery times (in minutes) were also recorded. Toxin density in blood was used to decide whether a mouse recovered. Mice were checked at two times for recovery. If a mouse had recovered at the first time, the observation is left-censored, and no further measurement is made. The variable time1 is set to missing and time2 is set to the measurement time to indicate left-censoring. If a mouse had not recovered at the first time, it was checked later at a second time. If it had recovered by the second measurement time, the observation is interval-censored and the variable time1 is set to the first measurement time and time2 is set to the second measurement time. If there was no recovery at the second measurement, the observation is right-censored, and time1 is set to the second measurement time and time2 is set to missing to indicate right-censoring.

The following statements create a SAS data set containing the data from the experiment and fit a Weibull model with age, sex, and age and sex interaction as covariates.

```
title 'Natural Recovery Time';
data mice;
   input sex age time1 time2 ;
   datalines;
1  57  631   631
1  45  .     170
1  54  227   227
1  43  143   143
1  64  916   .
1  67  691   705
1  44  100   100
1  59  730   .
1  47  365   365
1  74  1916  1916
2  79  1326  .
2  75  837   837
2  84  1200  1235
2  54  .     365
2  74  1255  1255
2  71  1823  .
2  65  537   637
2  33  583   683
2  77  955   .
2  46  577   577
;

data xrow1;
   input sex age time1 time2 ;
```

```
   datalines;
1  50  .  .
;

data xrow2;
   input sex age time1 time2 ;
   datalines;
 2  60.6  .  .
 ;

proc lifereg data=mice xdata=xrow1;
   class sex  ;
   model (time1, time2) = age sex age*sex / dist=Weibull;

   probplot  / nodata
       font = swiss
       plower=.5
       vref(intersect) = 75
       vreflab = '75 Percent'
       vreflabpos = 2
       cfit=blue
       cframe=ligr
       ;
   inset  / cfill = white
       ctext = blue;
run; quit;
```

Standard output is shown in Output 6.1.1. Tables containing general model information, Type III tests for the main effects and interaction terms, and parameter estimates are created.

Output 6.1.1. Parameter Estimates for the Interaction Model

Natural Recovery Time

The LIFEREG Procedure

Model Information

Data Set	WORK.MICE
Dependent Variable	Log(time1)
Dependent Variable	Log(time2)
Number of Observations	20
Noncensored Values	9
Right Censored Values	5
Left Censored Values	2
Interval Censored Values	4
Name of Distribution	Weibull
Log Likelihood	-25.91033295

Type III Analysis of Effects

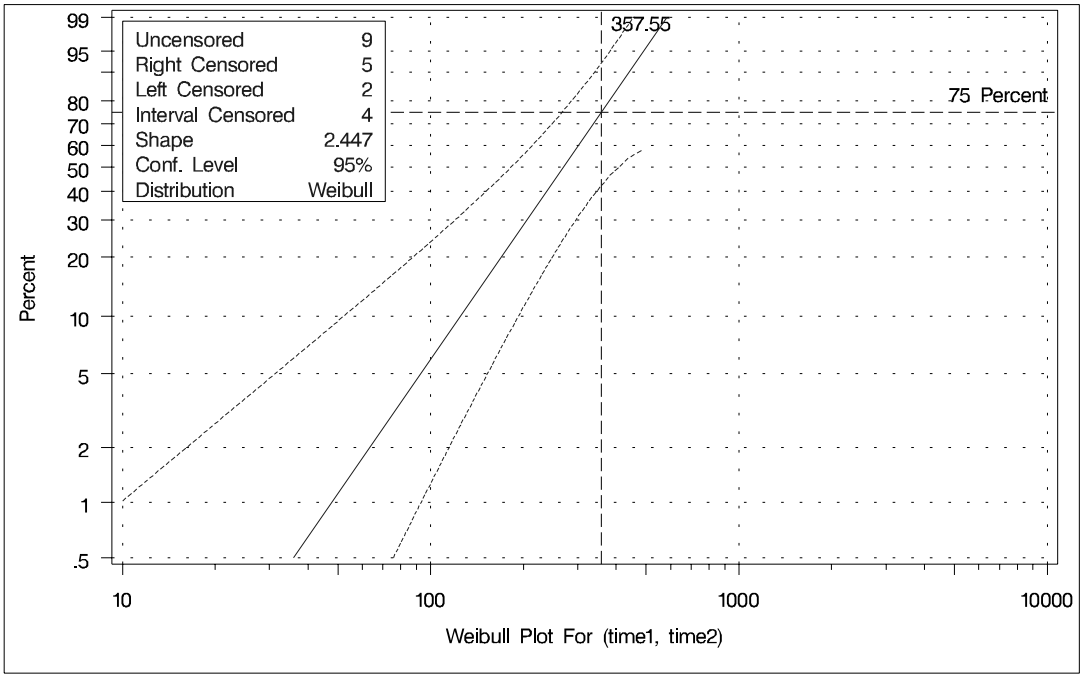
Effect	DF	Wald Chi-Square	Pr > ChiSq
age	1	33.8496	<.0001
sex	1	14.0245	0.0002
age*sex	1	10.7196	0.0011

Analysis of Parameter Estimates

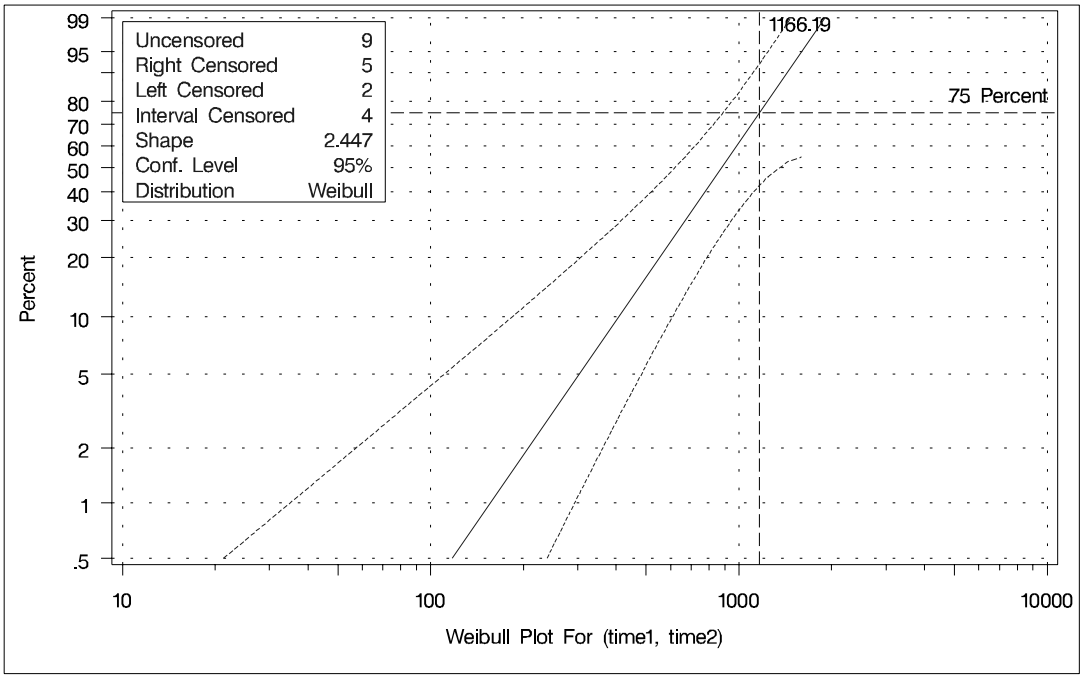
Parameter		DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept		1	5.4110	0.5549	4.3234	6.4986	95.08	<.0001
age		1	0.0250	0.0086	0.0081	0.0419	8.42	0.0037
sex	1	1	-3.9808	1.0630	-6.0643	-1.8974	14.02	0.0002
sex	2	0	0.0000	0.0000	0.0000	0.0000	.	.
age*sex	1	1	0.0613	0.0187	0.0246	0.0980	10.72	0.0011
age*sex	2	0	0.0000	0.0000	0.0000	0.0000	.	.
Scale		1	0.4087	0.0900	0.2654	0.6294		
Weibull Shape		1	2.4468	0.5391	1.5887	3.7682		

The following two plots display the predicted probability against the recovery time for two different populations. Output 6.1.2 is created with the PROBPLOT statement with the option XDATA= xrow1, which specifies the population with sex = 1, age = 50. Although the SAS statements are not shown, Output 6.1.3 is created with the PROBPLOT statement with the option XDATA= xrow2, which specifies the population with sex = 2, age = 60.6. These are the default values that the LIFEREG procedure would use for the probability plot if the XDATA= option had not been specified. Reference lines are used to display specified predicted probability points and their relative locations on the plot.

Output 6.1.2. Probability Plot for Recovery Time with sex = 1, age = 50



Output 6.1.3. Probability Plot for Recovery Time with sex = 2, age = 60.6



Example 6.2. Probability Plotting–Right Censoring

The following statements create a SAS data set containing observed and right-censored lifetimes of 70 diesel engine fans (Nelson 1982, p. 318).

```
title 'Engine Fan Lifetime Study';
data fan;
   input lifetime censor@@;
   lifetime = lifetime / 1000;
   label lifetime = Lifetime;
   datalines;
 450 0    460 1   1150 0   1150 0   1560 1
1600 0   1660 1   1850 1   1850 1   1850 1
1850 1   1850 1   2030 1   2030 1   2030 1
2070 0   2070 0   2080 0   2200 1   3000 1
3000 1   3000 1   3000 1   3100 0   3200 1
3450 0   3750 1   3750 1   4150 1   4150 1
4150 1   4150 1   4300 1   4300 1   4300 1
4300 1   4600 0   4850 1   4850 1   4850 1
4850 1   5000 1   5000 1   5000 1   6100 1
6100 0   6100 1   6100 1   6300 1   6450 1
6450 1   6700 1   7450 1   7800 1   7800 1
8100 1   8100 1   8200 1   8500 1   8500 1
8500 1   8750 1   8750 0   8750 1   9400 1
9900 1  10100 1  10100 1  10100 1  11500 1
;
run;
```

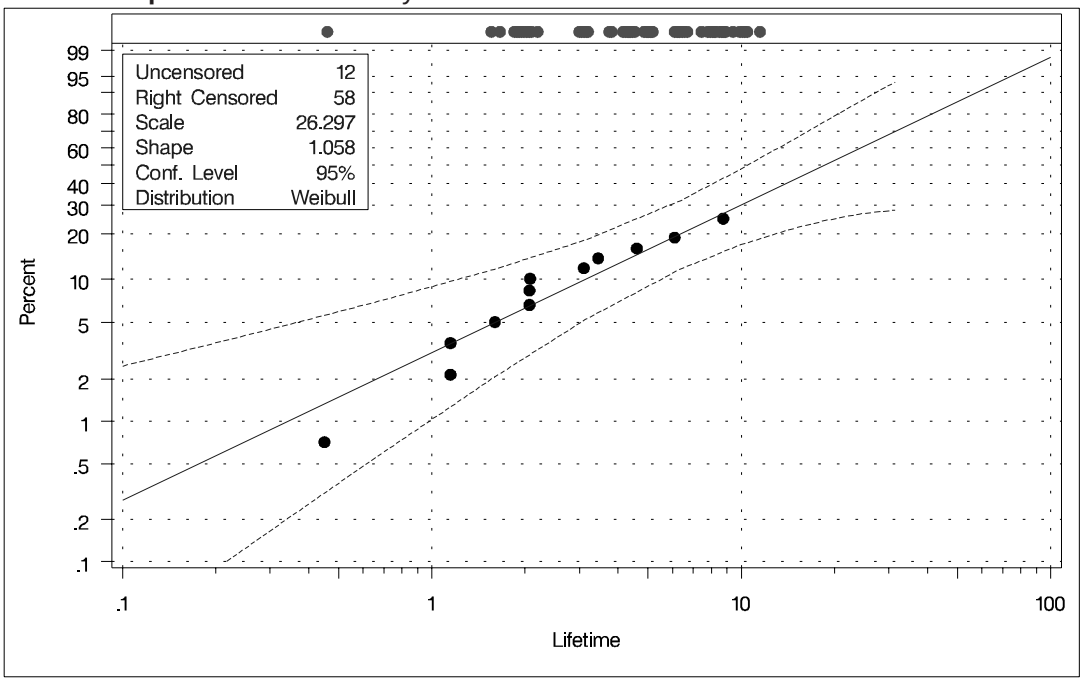
Some of the fans had not failed at the time the data were collected, and the unfailed units have right-censored lifetimes. The variable LIFETIME represents either a failure time or a censoring time in thousands of hours. The variable CENSOR is equal to 0 if the value of LIFETIME is a failure time, and it is equal to 1 if the value is a censoring time. The following statements use the LIFEREG procedure to produce the probability plot with an inset for the engine lifetimes.

```
symbol v=dot c=white;
proc lifereg;
   model lifetime*censor( 1 ) = / d = weibull;
   probplot
       cencolor = red
       cframe   = ligr
       cfit     = blue
       ppout
       npintervals=simul
       ;
   inset /
       cfill = white
       ctext = blue;
run; quit;
```

The resulting graphical output is shown in Output 6.2.1. The estimated CDF, a line representing the maximum-likelihood fit, and pointwise parametric confidence bands

are plotted in the body of Output 6.2.1. The values of right-censored observations are plotted along the top of the graph. The "Cumulative Probability Estimates" table is also shown in Output 6.2.2.

Output 6.2.1. Probability Plot for the Fan Data



Output 6.2.2. CDF Estimates

Engine Fan Lifetime Study

The LIFEREG Procedure

Cumulative Probability Estimates

Lifetime	Cumulative Probability	Simultaneous 95% Confidence Limits Lower	Simultaneous 95% Confidence Limits Upper	Kaplan-Meier Estimate	Kaplan-Meier Standard Error
0.45	0.0071	0.0007	0.2114	0.0143	0.0142
1.15	0.0215	0.0033	0.2114	0.0288	0.0201
1.15	0.0360	0.0073	0.2168	0.0433	0.0244
1.6	0.0506	0.0125	0.2304	0.0580	0.0282
2.07	0.0666	0.0190	0.2539	0.0751	0.0324
2.07	0.0837	0.0264	0.2760	0.0923	0.0361
2.08	0.1008	0.0344	0.2972	0.1094	0.0392
3.1	0.1189	0.0436	0.3223	0.1283	0.0427
3.45	0.1380	0.0535	0.3471	0.1477	0.0460
4.6	0.1602	0.0653	0.3844	0.1728	0.0510
6.1	0.1887	0.0791	0.4349	0.2046	0.0581
8.75	0.2488	0.0884	0.6391	0.2930	0.0980

Example 6.3. Probability Plotting–Arbitrarily Censoring

Table 6.2 contains microprocessor failure data from Nelson (1990). Units were inspected at predetermined time intervals. The data consist of inspection interval endpoints (in hours) and the number of units failing in each interval. A missing (.) lower endpoint indicates left censoring, and a missing upper endpoint indicates right censoring. These can be thought of as semi-infinite intervals with a lower (upper) endpoint of negative (positive) infinity for left (right) censoring.

Table 6.2. Interval-Censored Data

Lower Endpoint	Upper Endpoint	Number Failed
.	6	6
6	12	2
24	48	2
24	.	1
48	168	1
48	.	839
168	500	1
168	.	150
500	1000	2
500	.	149
1000	2000	1
1000	.	147
2000	.	122

The following SAS program will compute the Turnbull estimate and create a lognormal probability plot.

```
data micro;
   input t1 t2 f ;
   datalines;
. 6 6
6 12 2
12 24 0
24 48 2
24 .  1
48 168 1
48 .   839
168 500 1
168 .   150
500 1000 2
500 .    149
1000 2000 1
1000 . 147
2000 . 122
;
run;
```

```
title;
symbol v=dot c=white;
proc lifereg data=micro;
   model ( t1 t2 ) = / d=lognormal intercept=25 scale=5;
   weight f;
   probplot
      cframe = ligr
      cfit   = blue
      pupper = 10
      itprintem
      printprobs
      maxitem = (1000,25)
      ppout;
   inset / cfill = white;
run; quit;
```

The two initial values INTERCEPT= 25 and SCALE= 5 in the MODEL statement are used to aid convergence in the model-fitting algorithm.

The following tables are created by the PROBPLOT statement in addition to the standard tabular output from the MODEL statement. Output 6.3.1 shows the iteration history for the Turnbull estimate of the CDF for the microprocessor data. With both options ITPRINTEM and PRINTPROBS specified in the PROBPLOT statement, this table contains the log likelihoods and interval probabilities for every 25th iteration and the last iteration. It would only contain the log likelihoods if the option PRINTPROBS were not specified.

Output 6.3.1. Iteration History for the Turnbull Estimate

The LIFEREG Procedure

Iteration History for the Turnbull Estimate of the CDF

Iteration	Loglikelihood	(., 6)	(6, 12)	(24, 48)	(48, 168)
		(168, 500)	(500, 1000)	(1000, 2000)	(2000, .)
0	-1133.4051	0.125	0.125	0.125	0.125
		0.125	0.125	0.125	0.125
25	-104.16622	0.00421644	0.00140548	0.00140648	0.00173338
		0.00237846	0.00846094	0.04565407	0.93474475
50	-101.15151	0.00421644	0.00140548	0.00140648	0.00173293
		0.00234891	0.00727679	0.01174486	0.96986811
75	-101.06641	0.00421644	0.00140548	0.00140648	0.00173293
		0.00234891	0.00727127	0.00835638	0.9732621
100	-101.06534	0.00421644	0.00140548	0.00140648	0.00173293
		0.00234891	0.00727125	0.00801814	0.97360037
125	-101.06533	0.00421644	0.00140548	0.00140648	0.00173293
		0.00234891	0.00727125	0.00798438	0.97363413
130	-101.06533	0.00421644	0.00140548	0.00140648	0.00173293
		0.00234891	0.00727125	0.007983	0.97363551

Output 6.3.2. Summary for the Turnbull Algorithm

The LIFEREG Procedure

Lower Lifetime	Upper Lifetime	Probability	Reduced Gradient	Lagrange Multiplier
.	6	0.0042	0	0
6	12	0.0014	0	0
24	48	0.0014	0	0
48	168	0.0017	0	0
168	500	0.0023	0	0
500	1000	0.0073	-7.219342E-9	0
1000	2000	0.0080	-0.037063236	0
2000	.	0.9736	0.0003038877	0

The table in Output 6.3.2 summarizes the Turnbull estimates of the interval probabilities, the reduced gradients, and Lagrange multipliers as described in the section "Arbitrarily Censored Data" on page 90.

Output 6.3.3. Final CDF Estimates for Turnbull Algorithm

The LIFEREG Procedure

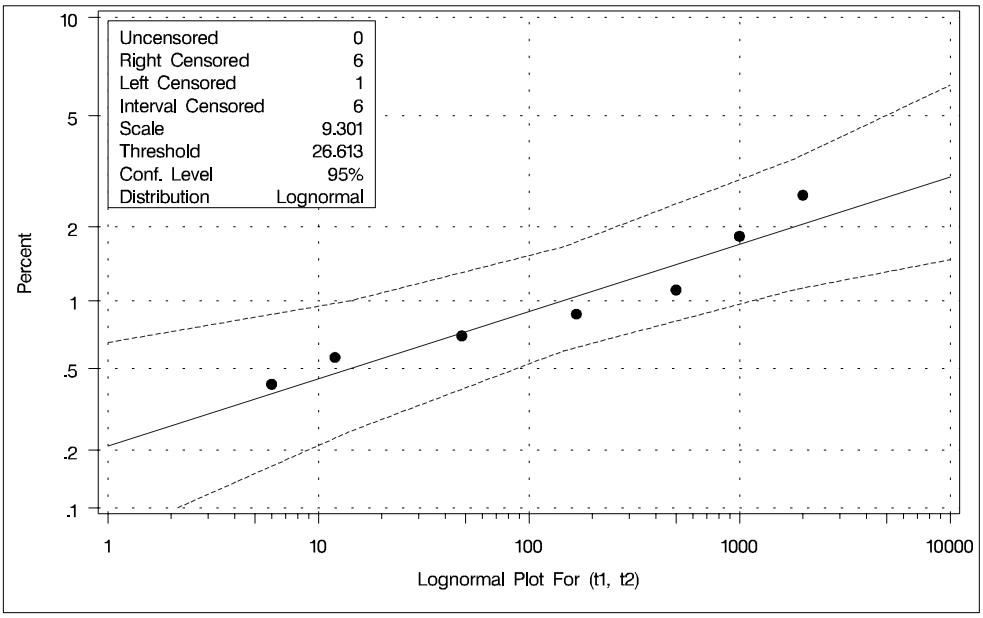
Cumulative Probability Estimates

Lower Lifetime	Upper Lifetime	Cumulative Probability	Pointwise 95% Confidence Limits Lower	Pointwise 95% Confidence Limits Upper	Standard Error
6	6	0.0042	0.0019	0.0094	0.0017
12	24	0.0056	0.0028	0.0112	0.0020
48	48	0.0070	0.0038	0.0130	0.0022
168	168	0.0088	0.0047	0.0164	0.0028
500	500	0.0111	0.0058	0.0211	0.0037
1000	1000	0.0184	0.0094	0.0357	0.0063
2000	2000	0.0264	0.0124	0.0553	0.0101

Output 6.3.3 shows the final estimate of the CDF, along with standard errors and nonparametric confidence limits. Two kinds of nonparametric confidence limits, pointwise or simultaneous, are available. The default is the pointwise nonparametric confidence limits. You can specify the simultaneous nonparametric confidence limits by the NPINTERVALS= SIMUL option.

Output 6.3.4 shows the CDF estimates, the maximum likelihood fit, and the pointwise parametric confidence limits plotted on a lognormal probability plot.

Output 6.3.4. Lognormal Probability Plot for the Microprocessor Data



References

Abernethy, R.B. (1996), *The New Weibull Handbook*, Second Edition, North Palm Beach, FL: Robert B. Abernethy.

Allison, P.D. (1995), *Survival Analysis Using the SAS System: A Practical Guide*, Cary, NC: SAS Institute.

Cox, D.R. (1972), "Regression Models and Life Tables (with discussion)," *Journal of the Royal Statistical Society, Series B*, 34, 187–220.

Cox, D.R. and Oakes, D. (1984), *Analysis of Survival Data*, London: Chapman and Hall.

Elandt-Johnson, R.C. and Johnson, N.L. (1980), *Survival Models and Data Analysis*, New York: John Wiley & Sons, Inc.

Gentleman, R. and Geyer, C.J. (1994), "Maximum Likelihood for Interval Censored Data: Consistency and Computation," *Biometrika*, 81 (3), 618–623.

Green, W.H. (1993), *Econometric Analysis*, Second Edition, New York: Cambridge University Press.

Gross, A.J. and Clark, V.A. (1975), *Survival Distributions: Reliability Applications in the Biomedical Sciences*, New York: John Wiley & Sons, Inc.

Kalbfleisch, J.D. and Prentice, R.L. (1980), *The Statistical Analysis of Failure Time Data*, New York: John Wiley & Sons, Inc.

Klein, J.P. and Moeschberger, M.L. (1997), *Survival Analysis: Techniques for Censored and Truncated Data*, Berlin: Springer–Verlag.

Lawless, J.E. (1982), *Statistical Models and Methods for Lifetime Data*, New York: John Wiley & Sons, Inc.

Lee, E.T. (1980), *Statistical Methods for Survival Data Analysis*, Belmont, CA: Lifetime Learning Publications.

Maddala, G.S. (1983), *Limited-Dependent and Qualitative Variables in Econometrics*, New York: Cambridge University Press.

Meeker, W.Q. and Escobar, L.A.(1998), *Statistical Methods for Reliability Data*, New York: John Wiley & Sons.

Mroz, T.A. (1987), "The Sensitivity of an Empirical Model of Married Women's Work to Economic and Statistical Assumptions," *Econometrica* 55, 765–799.

Nair, V.N. (1984), "Confidence Bands for Survival Functions With Censored Data: A Comparative Study," *Technometrics*, 26 (3), 265–275.

Nelson, W. (1982), *Applied Life Data Analysis*, New York: John Wiley & Sons.

Nelson, W. (1990), *Accelerated Testing: Statistical Models, Test Plans, and Data Analyses*, New York: John Wiley & Sons.

Rao, C.R. (1973), *Linear Statistical Inference and Its Applications*, New York: John Wiley & Sons, Inc.

Tobin, J. (1958), "Estimation of Relationships for Limited Dependent Variables," *Econometrica*, 26, 24–36.

Turnbull, B.W. (1976), "The Empirical Distribution Function with Arbitrarily Grouped, Censored and Truncated Data," *Journal of the Royal Statistical Society, Series B*, 38, 290–295.

Chapter 7
The LOESS Procedure

Chapter Table of Contents

Chapter 7
The LOESS Procedure

Overview

The SELECT= option in the MODEL statement now enables you to select the smoothing parameter so that an appropriate measure of the model degrees of freedom matches a given target value.

Syntax

MODEL Statement

The following updated option is available in the MODEL statement after a slash (/).

SELECT=*criterion* **< (<GLOBAL> <STEPS> <RANGE(***lower***,***upper***)>)>**
SELECT=*DFType* **(** *target* **<GLOBAL> <STEPS> <RANGE(***lower***,***upper***)>)**
specifies that automatic smoothing parameter selection be done using the named *criterion* or *DFType*. Valid values for the *criterion* are

AICC specifies the AIC_C criterion (Hurvich, Simonoff, and Tsai 1998).

AICC1 specifies the AIC_{C_1} criterion (Hurvich, Simonoff, and Tsai 1998).

GCV specifies the generalized cross-validation criterion (Craven and Wahba 1979).

The *DFType* specifies the measure to be used to estimate the model degrees of freedom. The measures implemented in PROC LOESS all depend on the prediction matrix L relating the observed and predicted values of the dependent variable. Valid values are

DF1 specifies Trace(L).

DF2 specifies Trace$(L^T L)$.

DF3 specifies 2Trace(L) $-$ Trace$(L^T L)$.

For both types of selection, the smoothing parameter value is selected to yield a minimum of an optimization criterion. If you specify *criterion* as one of AICC, AICC1, or GCV, the optimization criterion is the specified *criterion*. If you specify *DFType* as one of DF1, DF2, or DF3, the optimization criterion is $|$ *DFType* $-$ *target* $|$, where *target* is a specified target degree of freedom value. Note that if you specify a *DFType*, then you must also subsequently specify a target value in parentheses. See the section "Automatic Smoothing Parameter Selection Based on Degrees of Freedom" on page 109 for definitions and properties of the selection criteria.

The selection is done as follows:

- If you specify the SMOOTH=*value-list* option, then PROC LOESS selects the value in this list that yields the global minimum of the specified optimization criterion, with ties going to the larger value.
- If you do not specify the SMOOTH= option, then PROC LOESS finds a local minimum of the specified optimization criterion using a golden section search.

You can specify the following modifiers in parentheses after the specified criterion to alter the behavior of the SELECT= option:

GLOBAL	specifies that a global minimum be found within the range of smoothing parameter values examined. This modifier has no effect if you also specify the SMOOTH= option in the MODEL statement.
STEPS	specifies that all models evaluated in the selection process be displayed.
RANGE(*lower*,*upper*)	specifies that only smoothing parameter values greater than or equal to *lower* and less than or equal to *upper* be examined.

For models with one dependent variable, if you specify neither the SELECT= nor the SMOOTH= options in the MODEL statement, then PROC LOESS uses SELECT=AICC.

The following table summarizes how the smoothing parameter values are chosen for various combinations of the SMOOTH= option, the SELECT= option, and the SELECT= option modifiers.

Table 7.1. Smoothing Parameter Value(s) Used for Combinations of SMOOTH= and SELECT= OPTIONS for Models with One Dependent Variable

Syntax	Search Method	Search Domain
default	golden section using AICC	$(0, 1]$
SMOOTH=*list*	no selection	values in *list*
SMOOTH=*list* SELECT=*criterion*	global	values in *list*
SMOOTH=*list* SELECT=*criterion* (RANGE(l, u))	global	values in *list* within $[l, u]$
SELECT=*criterion*	golden section	$(0, 1]$
SELECT=*criterion* (RANGE(*l*,*u*))	golden section	$[l, u]$
SELECT=*criterion* (GLOBAL)	global	$(0, 1]$
SELECT=*criterion* (GLOBAL RANGE(l, u))	global	$[l, u]$

Some examples of using the SELECT= option are

SELECT=GCV	specifies selection using the GCV *criterion*.
SELECT=DF1(6.3)	specifies selection using the DF1 *DFType* with target value 6.3.

SELECT=AICC(STEPS) specifies selection using the AICC *criterion* showing all step details.

SELECT=DF2(7 GLOBAL) specifies selection using the DF2 *DFType* with target value 7 using a global search algorithm.

Note: The SELECT= option cannot be used for models with more than one dependent variable.

Details

Automatic Smoothing Parameter Selection Based on Degrees of Freedom

PROC LOESS can now perform automatic smoothing parameter selection by setting (as near as possible) an approximate measure of model degrees of freedom to a specified target value. These methods are useful for making meaningful comparisons between loess fits and other parametric and nonparametric fits. There are three commonly used measures of model degrees of freedom in nonparametric models. All of them are functions of the prediction matrix L such that

$$\hat{y} = Ly$$

where y is the vector of observed values and $\hat{y}$ is the corresponding vector of predicted values of the dependent variable. The measures are:

$$\begin{aligned} \text{DF1} &= \text{Trace}(L) \\ \text{DF2} &= \text{Trace}(L^T L) \\ \text{DF3} &= 2\text{Trace}(L) - \text{Trace}(L^T L) \end{aligned}$$

A discussion of their properties can be found in Hastie and Tibshirani (1990). You invoke these methods by specifying the SELECT=*DFType(target)* option in the MODEL statement, where *DFType* is one of DF1, DF2, or DF3. The criterion that is minimized is $|\,\textit{DFType} - \textit{target}\,|$.

Example

Example 7.1. Melanoma Incidences

The following data from the Connecticut Tumor Registry presents age-adjusted numbers of melanoma incidences per 100,000 people for 37 years from 1936 to 1972 (Houghton, Flannery, and Viola 1980).

```
data Melanoma;
   input  Year Incidences @@;
   format Year d4.0;
datalines;
1936    0.9   1937   0.8  1938   0.8  1939   1.3
```

```
1940    1.4   1941   1.2  1942   1.7  1943   1.8
1944    1.6   1945   1.5  1946   1.5  1947   2.0
1948    2.5   1949   2.7  1950   2.9  1951   2.5
1952    3.1   1953   2.4  1954   2.2  1955   2.9
1956    2.5   1957   2.6  1958   3.2  1959   3.8
1960    4.2   1961   3.9  1962   3.7  1963   3.3
1964    3.7   1965   3.9  1966   4.1  1967   3.8
1968    4.7   1969   4.4  1970   4.8  1971   4.8
1972    4.8
;
```

The following statements fit a model with a target value of 2 for the DF2 criterion.

```
proc loess data=Melanoma;
   model Incidences=Year / select=DF2(2);
   ods output outputStatistics=out2;
run;
```

The optimal smoothing parameter selection table is shown in Output 7.1.1.

Output 7.1.1. Optimal Smoothing Criterion Table from PROC LOESS

```
                The LOESS Procedure
        Dependent Variable: Incidences

            Optimal Smoothing
                Criterion

                          Smoothing
              DF2         Parameter

            2.30235         1.00000
```

Note that the actual value of DF2 achieved is not exactly the target value of 2. This occurs because even though the smoothing parameter is a continuous variable, smoothing parameter values in the range $(0, 1]$ correspond to a finite set of local models. Among this finite set of models, the model with a smoothing parameter value of 1 yields the DF2 measure that is closest to the specified target value of 2.

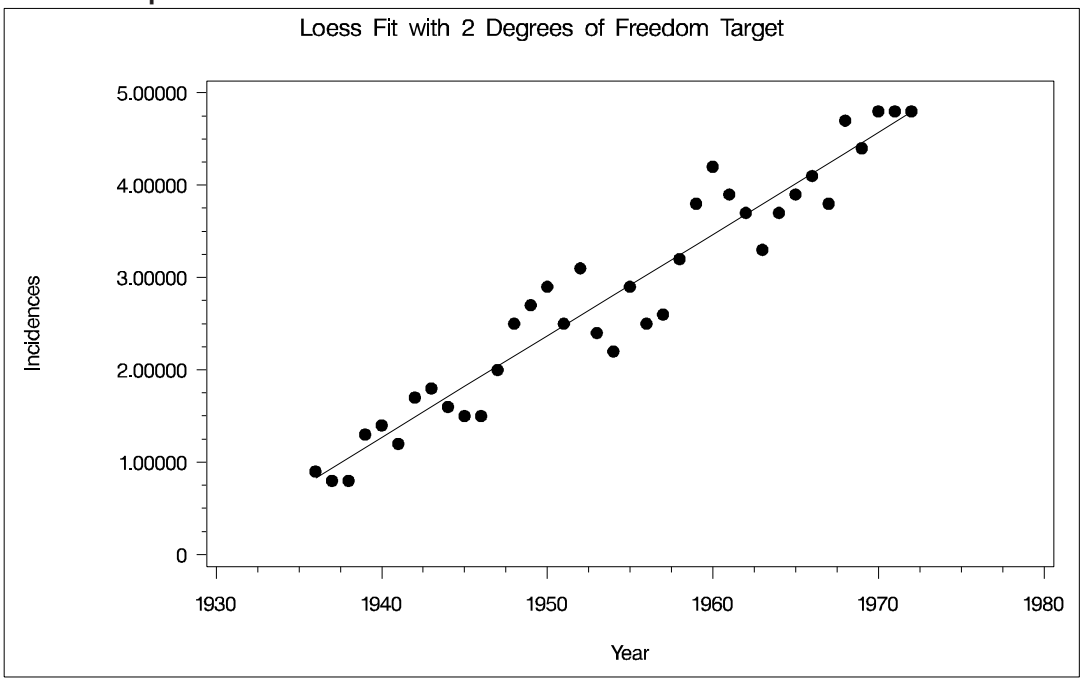
Recall that the smoothing parameter value specifies the fraction of the data to be used in each local model. Thus, the selected smoothing parameter value 1 corresponds to using all the data points in each local linear model. This is to be expected, as a LOESS model with about two degrees of freedom has enough latitude to determine one common slope and intercept for all local models. Hence, in this case the LOESS model is really just a global linear model. You can confirm this observation by plotting the fit obtained as follows:

```
symbol1 color=blue value=dot ;
symbol2 color=blue interpol=join value=none;
axis1 label=(angle=90 rotate=0);
%let opts = vaxis=axis1 frame cframe=ligr hm=3 vm=3 overlay;

proc gplot data=out2;
```

```
  title1 'Melanoma Data with Default Loess Fit';
  plot  DepVar*Year Pred*Year/ &opts;
run;
```

The graph produced is shown in Output 7.1.2.

Output 7.1.2. Scatter Plot of Incidences versus Year for the Melanoma Data



The loess fit in the previous plot captures the increasing trend in the data. In order to capture the oscillations in the data that are caused by the eleven-year sunspot cycle, it is necessary to use a loess model with more degrees of freedom. The following statements fit such a model with a target value of 7 degrees of freedom. Loosely speaking, this corresponds to a global fit with a polynomial of degree 6.

```
proc loess data=Melanoma;
   model Incidences=Year / select=DF2(7);
   ods output outputStatistics=out7;
run;
```

The optimal smoothing parameter selection table is shown in Output 7.1.3.

Output 7.1.3. Optimal Smoothing Criterion Table from PROC LOESS

The LOESS Procedure
Dependent Variable: Incidences

Optimal Smoothing Criterion

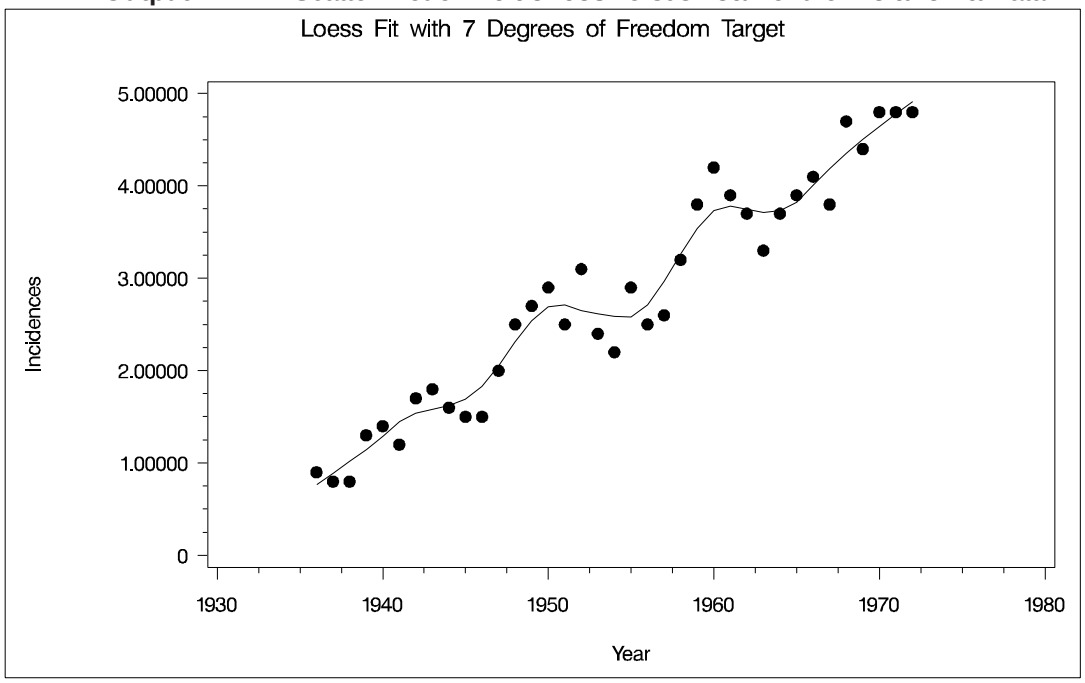
DF2	Smoothing Parameter
7.31083	0.25676

You can plot the fit with the following statements:

```
symbol1 color=blue value=dot ;
symbol2 color=blue interpol=join value=none;
axis1 label=(angle=90 rotate=0);
%let opts = vaxis=axis1 frame cframe=ligr hm=3 vm=3 overlay;

proc gplot data=out7;
  title1 'Melanoma Data with Default Loess Fit';
  plot  DepVar*Year Pred*Year/ &opts;
run;
```

The result is shown in Output 7.1.4

Output 7.1.4. Scatter Plot of Incidences versus Year for the Melanoma Data



References

Akaike, H. (1973), "Information Theory and an Extension of the Maximum Likelihood Principle," in *Proceedings of the Second International Symposium on Information Theory*, eds. Petrov and Csaki, 267-281.

Craven, P. and Wahba, G. (1979), "Smoothing Noisy Data with Spline Functions," *Numerical Mathematics*, 31, 377–403.

Hastie, T.J. and Tibshirani, R.J. (1990), *Generalized Additive Models*, Monographs on Statistics and Applied Probability 43, London: Chapman and Hall.

Houghton, A.N., Flannery, J., and Viola, M.V. (1980), "Malignant Melanoma in Connecticut and Denmark," *International Journal of Cancer*, 25, 95–104.

Hurvich, C.M., Simonoff, J.S., and Tsai, C.L. (1998), "Smoothing Parameter Selection in Nonparametric Regression Using an Improved Akaike Information Criterion," *Journal of the Royal Statistical Society B*, 60, 271–293.

Chapter 8
The LOGISTIC Procedure

Chapter Table of Contents

Chapter 8
The LOGISTIC Procedure

Overview

You can now fit a generalized logit model when you have nominal response data. In this multiple-logit model, each nonreference response category is contrasted against the reference category. This model reduces to the binary logit model when there are only two response categories. Unlike the cumulative logit model (for ordinal response data) that contains one set of slope parameters for the entire model, the generalized logit model contains a set of slope parameters for each response category other than the reference category. The generalized logit model is specified by the MODEL statement option LINK=GLOGIT. You can choose the response category to use as the reference in forming the various logit functions. The default is to use the last response category as the reference.

Options specific to the response variable can be specified immediately after the response variable in the MODEL statement by enclosing them in a pair of parentheses. These response variable options include

- the EVENT= option for specifying the event category for binary response models
- the REFERENCE= option for specifying the reference category
- the ORDER= option for specifying the sorting order of the response categories
- the DESCENDING option for reversing the order of the response categories

For ordinal models, names of the intercept parameters in the OUTEST= data are changed to reflect the corresponding response levels.

Syntax

MODEL Statement

There are two sets of options in the MODEL statement. Options specific to the response variable are referred to as "Response Variable options." They are specified after the response variable in a pair of parentheses. The other set of options, specified after a slash (/) following the model effects, are referred to as "MODEL options."

MODEL *variable* < *(variable_options)* > = < *effects* >< */options* >**;**

The Response Variable options include the DESCENDING, EVENT=, ORDER=, and REF= options. The MODEL option LINK= has an additional value for specifying the generalized logit model.

DESCENDING | DESC

reverses the order of the response categories (levels). If both the DESCENDING and ORDER= options are specified, PROC LOGISTIC orders the response categories according to the ORDER= option and then reverses that order.

EVENT=*'category'* **|** *keyword*

specifies the event category (level) for the binary response model. PROC LOGISTIC models the probability of the event category. The EVENT= option has no effect when there are more than two response categories. You can specify the event category (formatted if a format is applied) in quotes or you can specify one of the following keywords. The default is EVENT=FIRST.

FIRST designates the first ordered category as the event

LAST designates the last ordered category as the event

Consider the example where the response variable Y takes the values 1 and 0 for event and nonevent, respectively, and Exposure is the explanatory variable. To model the event probability, you simply specify

```
proc logistic;
   model Y(event='1') = Exposure;
run;
```

ORDER= DATA | FORMATTED | FREQ | INTERNAL

specifies the sorting order of the categories (levels) of the response variable. The following table shows the interpretation of the ORDER= values.

Value of ORDER=	Levels Sorted By
DATA	order of appearance in the input data set
FORMATTED	external formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	descending frequency count; levels with the most observations come first in the order
INTERNAL	unformatted value

By default, ORDER=FORMATTED. For the FORMATTED and INTERNAL specifications, the sort order may be machine dependent. For numeric response variables with no explicit format (that is, no corresponding FORMAT statement in the current PROC LOGISTIC run or in the DATA step that created the data set), ORDER=FORMATTED orders the response levels by their numeric values.*

*This represents a change from earlier releases for how response levels are ordered. In releases previous to Version 8, numeric levels with no explicit format were ordered by their BEST12. formatted values. If you want to continue to use this ordering, you would specify this format explicitly. The change was implemented because the former default behavior for ORDER=FORMATTED often resulted in levels not being ordered numerically and usually required the user to intervene with an explicit format or ORDER=INTERNAL to obtain the more natural ordering.

For more information on sorting order, see the chapter on the SORT procedure in the *SAS Procedures Guide*.

REFERENCE=*'category'* **|** *keyword*
REF=*'category'* **|** *keyword*

specifies the reference category for the generalized logit model and the binary response model. For the generalized logit model, each nonreference category is contrasted with the reference category. For the binary response model, specifying one response category as the reference is equivalent to specifying the other response category as the event category. You can specify the reference category (formatted if a format is applied) in quotes or you can specify one of the following keywords. The default is REF=LAST.

FIRST	designates the first ordered category as the reference
LAST	designates the last ordered category as the reference

LINK=*keyword*
L=*keyword*

specifies the function linking the response probabilities to the linear predictors. You can specify one of the following keywords. The default is LINK=LOGIT.

CLOGLOG	the complementary log-log function. PROC LOGISTIC fits the binary CLOGLOG model when there are two response categories and fits the cumulative CLOGLOG model when there are more than two response categories.
GLOGIT	the generalized logit function. PROC LOGISTIC fits the generalized logit model where each nonreference category is contrasted with the reference category. You can use the response variable option REF= to specify the reference category.
LOGIT	the log odds function. PROC LOGISTIC fits the binary logit model when there are two response categories and fits the cumulative logit model when there are more than two response categories.
PROBIT	the inverse standard normal distribution function. PROC LOGISTIC fits the binary probit model when there are two response categories and fits the cumulative probit model when there are more than two response categories.

Details

Generalized Logit Model

Formulation of the generalized logit models for nominal response variables can be found in Agresti (1990). Let Y be the response variable with categories $1, \ldots, r$. Let $\mathbf{x} = (x_0, x_1, \ldots, x_p)'$ be a $(p+1)$ vector of covariates, with $x_0 \equiv 1$. By choosing k as the reference category, the jth logit is given by

$$\log\left[\frac{\Pr(\mathrm{Y}=j|\mathbf{x})}{\Pr(\mathrm{Y}=k|\mathbf{x})}\right] = \mathbf{x}'\boldsymbol{\beta}_j \qquad 0 \le j \le r, j \ne k$$

where $\boldsymbol{\beta}_j$ is a $(p+1)$ vector of the regression coefficients for the jth logit.

For a sample of n subjects $\{(y_i, \mathbf{x}_i), 1 \le i \le n\}$, the log likelihood for the generalized logit model is

$$l = l(\boldsymbol{\beta}_j, 1 \le j \le r, j \ne k) = \sum_{i=1}^{n} \log(\Pr(\mathrm{Y}=y_i|\mathbf{x}_i))$$

and the log likelihood is maximized with respect to $\boldsymbol{\beta}_j, 1 \le j \le r, j \ne k$.

Newton's Method of Parameter Estimation

For the convenience of notation, consider the last response level to be the reference level. The response probabilities $\pi_1, \ldots, \pi_r$ are given by

$$\begin{aligned}
\pi_r \equiv \Pr(\mathrm{Y}=r|\mathbf{x}) &= \frac{1}{1+\sum_{l=1}^{r-1} \mathrm{e}^{\mathbf{x}'\boldsymbol{\beta}_l}} \\
\pi_j \equiv \Pr(\mathrm{Y}=j|\mathbf{x}) &= \pi_r \mathrm{e}^{\mathbf{x}'\boldsymbol{\beta}_j} \qquad 1 \le j \le r-1
\end{aligned}$$

For a single response vector $y \equiv (y_1, \ldots, y_r)'$ with $y_j = 1$ if $Y = j$ and 0 otherwise, the log likelihood is

$$l(\pi_1, \ldots, \pi_r; \mathbf{y}) = \sum_{j=1}^{r} y_j \log(\pi_j)$$

Note that $\sum_{j=1}^{r} y_j = 1$ and $\sum_{j=1}^{r} \pi_j = 1$, so that

$$\frac{\partial l(\pi_1, \ldots, \pi_r; \mathbf{y})}{\partial \pi_j} = \frac{y_j}{\pi_j} - \frac{y_r}{\pi_r} \qquad , 1 \le j \le r$$

The first partial derivatives of $\pi_1, \ldots, \pi_r$ with respect to $\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_{r-1}$ are

$$\begin{aligned}
\frac{\partial \pi_j}{\partial \boldsymbol{\beta}_l} &= \begin{cases} -\pi_j \pi_l \mathbf{x} & 1 \le j \ne l \le r-1 \\ (\pi_j - \pi_j^2)\mathbf{x} & 1 \le j = l \le r-1 \end{cases} \\
\frac{\partial \pi_r}{\partial \boldsymbol{\beta}_l} &= -\pi_r \pi_l \mathbf{x} \qquad 1 \le l \le r-1
\end{aligned}$$

Denote $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_{r-1})'$ and $\boldsymbol{\pi}^* = (\pi_1, \ldots, \pi_r)'$. Then,

$$\mathbf{D} \equiv \frac{\partial(\pi_1, \ldots, \pi_r)}{\partial(\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_{r-1})} \equiv \begin{pmatrix} \frac{\partial' \pi_1}{\partial \boldsymbol{\beta}_1} & \cdots & \frac{\partial' \pi_1}{\partial \boldsymbol{\beta}_{r-1}} \\ \vdots & \ddots & \vdots \\ \frac{\partial' \pi_r}{\partial \boldsymbol{\beta}_1} & \cdots & \frac{\partial' \pi_r}{\partial \boldsymbol{\beta}_{r-1}} \end{pmatrix} = \begin{pmatrix} \operatorname{diag}(\boldsymbol{\pi}) - \boldsymbol{\pi}\boldsymbol{\pi}' \\ -\pi_r \boldsymbol{\pi}' \end{pmatrix} \otimes \mathbf{x}'$$

The estimating equations become

$$\mathbf{D}'\mathbf{W}(\mathbf{y} - \boldsymbol{\pi}^*) = 0 \qquad \text{where} \qquad \mathbf{W} = \operatorname{diag}(\pi_1^{-1}, \ldots, \pi_r^{-1})$$

Let

$$\mathbf{H} = \mathbf{D}'\mathbf{W}\mathbf{D} = (\operatorname{diag}(\boldsymbol{\pi}) - \boldsymbol{\pi}\boldsymbol{\pi}') \otimes \mathbf{x}\mathbf{x}'$$

$$\mathbf{g} = \mathbf{D}'\mathbf{W}(\mathbf{y} - \boldsymbol{\pi}^*) = (y_1 - \pi_1, \ldots, y_{r-1} - \pi_{r-1})' \otimes \mathbf{x}$$

Let $\boldsymbol{\beta} = (\boldsymbol{\beta}_1', \ldots, \boldsymbol{\beta}_{r-1}')'$. The estimate of $\boldsymbol{\beta}$ is obtained iteratively as follows:

$$\boldsymbol{\beta}^{l+1} = \boldsymbol{\beta}^l + (\sum_i w_i \mathbf{H}_i)^{-1} \sum_i w_i \mathbf{g}_i$$

where i indexes the observations and w_i is the product of the corresponding weight and frequency.

Confidence Limits for the Predicted Probabilities

By the delta method,

$$\sigma^2(\hat{\pi_j}) = \left(\frac{\partial \pi_j}{\partial \boldsymbol{\beta}}\right)' \mathbf{V}(\hat{\boldsymbol{\beta}}) \frac{\partial \pi_j}{\partial \boldsymbol{\beta}}$$

where

$$\left(\frac{\partial \pi_j}{\partial \boldsymbol{\beta}}\right)' = \left[\left(\frac{\partial \pi_j}{\partial \boldsymbol{\beta}_1}\right)', \ldots, \left(\frac{\partial \pi_j}{\partial \boldsymbol{\beta}_{r-1}}\right)'\right]$$

A 100(1$-\alpha$)% confidence level for π_j is given by

$$\hat{\pi}_j \pm z_{1-\alpha/2} \hat{\sigma}(\hat{\pi_j})$$

where $\hat{\sigma}(\hat{\pi_j})$ is obtained by replacing $\boldsymbol{\beta}$ by $\hat{\boldsymbol{\beta}}$ in $\sigma(\hat{\pi_j})$.

No Intercept Model

When the NOINT option is specified with the LINK=GLOGIT option, all intercepts are suppressed. This differs from the cumulative model where only the first intercept is suppressed when the NOINT option is specified.

Fit Statistics

Suppose there are r response categories and s covariates (each dummy variable being counted as a separate covariate). The number of parameters estimated is $p = (r-1)(1+s)$. Let L be the likelihood.

Akaike Information Criterion:

$$AIC = -2\log(L) + 2p$$

Schwartz Criterion:

$$SC = -2\log(L) + p\log(\sum_j f_j)$$

where f_j is the frequency of the jth observation.

Exact Conditional Analysis

When an EXACT statement and the LINK=GLOGIT option is specified, the generalized logit model is fit as described in Hirji (1992). If there are only two response levels, the binary logit model is fit instead. Hypothesis tests for each effect are computed across logit functions, but individual parameters are estimated for each logit function.

Association of Observed Values and Predicted Probabilities

When the LINK=GLOGIT option is specified, the "Association of Observed Values and Predicted Probabilities" table is suppressed unless there are only two response levels; in that case, the generalized logit model is reduced to the binary logit model.

Printing and Outputting Parameter Estimates

Each logit function has a set of parameters for the intercept and covariates. Instead of printing and outputting the parameter estimates by logit function, PROC LOGISTIC presents all parameter estimates for the intercept first, followed by all estimates of the first covariate, etc.

Since each logit function contrasts a nonreference response category with the reference category, the "Analysis of Maximum Likelihood Estimates" table includes the response variable column whose values are used to identify the corresponding logit function.

For the OUTEST= data set, names of parameters corresponding to the nonreference category 'xxx' contain _xxx as the suffix. For example, suppose the variable Net3 represents the television network viewed at a certain time, with values 'ABC', 'CBS', and 'NBC'. The following code fits a generalized logit model with Age and Gender (a CLASS variable with values Female and Male) as explanatory variables.

```
proc logistic;
   class Gender;
   model Net3 = Age Gender / link=glogit;
run;
```

Since 'NBC' is the last value in the sorted order of the response categories, it corresponds to the default reference category. There are two logit functions, one contrasting 'ABC' with 'NBC' and the other contrasting 'CBS' with 'NBC'. For each

logit, there are three parameters: an intercept parameter, a slope parameter for Age, and a slope parameter for Gender (since there are only two gender levels and the EFFECT parameterization is used by default). The names of the parameters and their descriptions are as follows.

Parameter	Description
Intercept_ABC	intercept parameter for the logit contrasting 'ABC' with 'NBC'
Intercept_CBS	intercept parameter for the logit contrasting 'CBS' with 'NBC'
Age_ABC	Age parameter for the logit contrasting 'ABC' with 'NBC'
Age_CBS	Age parameter for the logit contrasting 'CBS' with 'NBC'
GenderFemale_ABC	Gender=Female parameter for the logit contrasting 'ABC' with 'NBC'
GenderFemale_CBS	Gender=Female parameter for the logit contrasting 'CBS' with 'NBC'

Out= Output Data Set

If any of the XBETA=, STDXBETA=, PREDICTED=, LOWER=, and UPPER= options are specified in the OUTPUT statement when there are more than two response categories, each input observation generates as many output observations as the number of response categories. The predicted probabilities and their confidence limits correspond to the probabilities of individual response categories rather than the cumulative probabilities as in the case of fitting a cumulative model. Regression diagnostics are suppressed when there are more than two response categories. You can specify PREDPROB=(I C) to obtain the predicted probabilities of individual response categories as well as the predicted cumulative probabilities.

Intercept Names for Ordinal Models in OUTEST= Data set

If there are only two response categories in the entire data set, the intercept parameter is named Intercept; otherwise, the intercept parameters are named Intercept_xxx, where 'xxx' is the value (formatted if a format is applied) of the corresponding response category.

Exact Conditional Analysis: Computational Resources

PROC LOGISTIC uses a relatively fast and efficient algorithm for the exact analyses (Hirji, Mehta, and Patel 1987; Hirji 1992). This recently developed algorithm, together with improvements in computer power, makes it feasible to perform exact computations for data sets where previously only asymptotic methods could be applied, and for data sets where maximum likelihood methods fail to converge. Nevertheless, many problems require a prohibitive amount of time and memory for exact computations, depending on the speed and memory available on your computer. For such problems, consider whether exact methods are really necessary. Stokes, Davis,

and Koch (2000) suggest looking at exact p-values when the sample size is small and the approximate p-values from the unconditional analysis are less than 0.10.

A formula does not exist that can predict the amount of time and memory necessary to generate the exact conditional distributions for a particular problem. The time and memory required depends on several factors, including the total sample size, the number of parameters of interest, the number of nuisance parameters, and the order in which the parameters are processed. At any time while PROC LOGISTIC is deriving the distributions, you can terminate the computations by pressing the system interrupt key sequence (refer to the SAS Companion for your system) and choosing to stop computations. If you run out of memory, refer to the SAS Companion for your system to see how to allocate more.

You can use the MAXTIME= option in the EXACTOPTIONS option to limit the total amount of time PROC LOGISTIC uses to derive all of the exact distributions. If PROC LOGISTIC does not finish within that time, the procedure terminates. If you need to derive several distributions, it may be more feasible to request one distribution at a time.

You can monitor the progress of the procedure by submitting your program with the STATUSTIME= option in the EXACTOPTIONS option. If the procedure is too slow, you can try reordering the variables in the MODEL statement or reparameterizing your classification variables (Hirji, Mehta, and Patel 1987). This may help since processing large covariate values first enables the infeasibility criteria to reject paths earlier in the algorithm; for example, a continuous variable clustered around a few values may reject paths earlier than a more uniformly distributed covariate (refer to Derr (2000) for an illustration of how the algorithm works for a simple data set). PROC LOGISTIC attempts to speed up computations by first combining observations with the same covariate values, then sorting the dummy variables for the class variables you are conditioning on followed by the continuous variables being conditioned on, in order of their appearance in the MODEL statement. Each variable X_i is sorted based on the weights $x_{ij}w_j$, where i indexes the variable and j indexes the observation; for example, if $\sum_j 1_{\{x_{ij}>0\}}x_{ij}w_j > \sum_j 1_{\{x_{ij}<0\}}|x_{ij}|w_j$, then the positive values of x_{ij} are sorted in descending order followed by the negative values in ascending order followed by the zeros.

Example

Example 8.1. Generalized Logit Model

Halloween trick-or-treaters are given a choice of candy in three bowls: one bowl contains small chocolate candy bars, one contains lollipops, and the last contains sugar candies. The children are classified by gender and by apparent age (child and teenager), and the following candy preferences are observed:

		Candy		
Gender	Age	chocolate	lollipop	sugar
boy	child	2	13	3
boy	teenager	10	9	3
girl	child	3	9	1
girl	teenager	8	0	1

Interest centers on whether age or gender affects the choice of type of candy. A generalized logit model can be fit to relate these three factors. Since the data set may be too small for the asymptotic analysis to be valid, an exact analysis is also performed. The following statements perform this analysis.

```
data halloween;
   format Candy $9.;
   input Gender $ Age $ Candy $ count @@;
   datalines;
boy  child chocolate  2   boy  teenager chocolate 10
boy  child lollipop  13   boy  teenager lollipop   9
boy  child sugar      3   boy  teenager sugar      3
girl child chocolate  3   girl teenager chocolate  8
girl child lollipop   9   girl teenager lollipop   0
girl child sugar      1   girl teenager sugar      1
;
proc logistic data=halloween;
   freq count;
   class Gender(ref='girl') Age(ref='child') / param=ref;
   model Candy(ref='chocolate') = Gender Age / link=glogit;
   exact Gender Age / joint estimate=both;
run;
```

Reference levels for both Gender and Age are declared in the CLASS statement, while the reference level for Candy is specified in the MODEL statement. Since the response is nominal, a generalized logit model is fit by specifying the LINK=GLOGIT. For the exact analysis, a joint test for the parameters Gender and Age is requested, conditional on the intercepts. Output 8.1.1 through Output 8.1.6 display the results of the analyses.

Output 8.1.1.

```
                        The LOGISTIC Procedure

                          Model Information

              Data Set                      WORK.HALLOWEEN
              Response Variable             Candy
              Number of Response Levels     3
              Number of Observations        11
              Frequency Variable            count
              Sum of Frequencies            62
              Model                         generalized logit
              Optimization Technique        Fisher's scoring

NOTE: 1 observation having zero frequency or weight was excluded since it does
      not contribute to the analysis.
```

Output 8.1.2.

Response Profile

Ordered Value	Candy	Total Frequency
1	chocolate	23
2	lollipop	31
3	sugar	8

Logits modeled use Candy='chocolate' as the reference category.

The "Response Profile" table (Output 8.1.2) indicates that 'chocolate' is the reference category for the Candy variable, so the logits being modeled are

$$\log\left(\frac{\Pr(\text{Candy} = \text{lollipop})}{\Pr(\text{Candy} = \text{chocolate})}\right) \quad \text{and} \quad \log\left(\frac{\Pr(\text{Candy} = \text{sugar})}{\Pr(\text{Candy} = \text{chocolate})}\right)$$

Output 8.1.3.

Class Level Information

Class	Value	Design Variables
Gender	boy	1
	girl	0
Age	child	0
	teenager	1

The "Class Level Information" table (Output 8.1.3) shows that 'girl' and 'child' are the reference levels for Gender and Age, respectively.

Output 8.1.4. Asymptotic Results

Model Convergence Status

Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics

Criterion	Intercept Only	Intercept and Covariates
AIC	125.354	114.448
SC	129.608	127.210
-2 Log L	121.354	102.448

Output 8.1.4. (continued)

Testing Global Null Hypothesis: BETA=0

Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	18.9061	4	0.0008
Score	16.9631	4	0.0020
Wald	12.8115	4	0.0122

Type III Analysis of Effects

Effect	DF	Wald Chi-Square	Pr > ChiSq
Gender	2	4.7168	0.0946
Age	2	12.2325	0.0022

All of the hypothesis tests in Output 8.1.4 show that the model fits, although the Type III tests indicate that Gender has marginal influence.

Output 8.1.5. Asymptotic Results (continued)

Analysis of Maximum Likelihood Estimates

Parameter		Candy	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept		lollipop	1	0.7698	0.5782	1.7722	0.1831
Intercept		sugar	1	-0.9033	0.8664	1.0869	0.2972
Gender	boy	lollipop	1	1.5758	0.7569	4.3347	0.0373
Gender	boy	sugar	1	1.5261	1.0158	2.2570	0.1330
Age	teenager	lollipop	1	-2.6472	0.7572	12.2212	0.0005
Age	teenager	sugar	1	-1.7416	0.9623	3.2754	0.0703

Odds Ratio Estimates

Effect	Candy	Point Estimate	95% Wald Confidence Limits	
Gender boy vs girl	lollipop	4.835	1.097	21.313
Gender boy vs girl	sugar	4.600	0.628	33.686
Age teenager vs child	lollipop	0.071	0.016	0.313
Age teenager vs child	sugar	0.175	0.027	1.155

The parameter estimates and odds ratios for the asymptotic analysis are displayed in Output 8.1.5, and show that the odds of choosing a lollipop over a chocolate bar are five ($4.835 \approx 5$) times higher for boys versus girls, and a child is 14 ($1/0.071 \approx 14$) times more likely than a teenager to choose a lollipop over a chocolate bar.

Note in the "Analysis of Maximum Likelihood Estimates" table that the dummy parameters for the class variables are labeled by their nonreference level, and that the "Candy" column indicates the nonreference response category for the logit.

Output 8.1.6. Exact Results

Exact Conditional Analysis

Conditional Exact Tests

Effect	Test	Statistic	--- p-Value --- Exact	Mid
Joint	Score	16.6895	0.0013	0.0013
	Probability	7.115E-7	0.0009	0.0009
Gender	Score	5.0830	0.0870	0.0835
	Probability	0.00697	0.0988	0.0953
Age	Score	14.5093	0.0003	0.0003
	Probability	0.000032	0.0003	0.0003

Exact Parameter Estimates

Parameter		Candy	Estimate	95% Confidence Limits		p-Value
Gender	boy	lollipop	1.5017	-0.0692	3.4081	0.0641
Gender	boy	sugar	1.4114	-0.7079	4.0869	0.2715
Age	teenager	lollipop	-2.5231	-4.4303	-0.9979	0.0002
Age	teenager	sugar	-1.6244	-3.9734	0.5146	0.1673

Exact Odds Ratios

Parameter		Candy	Estimate	95% Confidence Limits		p-Value
Gender	boy	lollipop	4.489	0.933	30.209	0.0641
Gender	boy	sugar	4.102	0.493	59.553	0.2715
Age	teenager	lollipop	0.080	0.012	0.369	0.0002
Age	teenager	sugar	0.197	0.019	1.673	0.1673

The exact analysis (Output 8.1.6) produces results similar to the asymptotic analysis. The exact score statistic for the joint test is very close to the asymptotic global test, and the parameter estimates and odds ratios are quite similar. However, the contrast between boys and girls is only marginally significant.

References

Agresti, A. (1990), *Categorical Data Analysis*, New York: John Wiley & Sons, Inc.

Derr, R.E. (2000), "Performing Exact Logistic Regression with the SAS System," *Proceedings of the Twenty-Fifth Annual SAS Users Group International Conference*, Cary, NC: SAS Institute Inc.

Hirji, K.F. (1992), "Computing Exact Distributions for Polytomous Response Data," *JASA*, 487–492.

Hirji, K.F., Mehta, C.R., and Patel, N.R. (1987), "Computing Distributions for Exact Logistic Regression," *JASA*, 82, 1110–1117.

Stokes, M.E., Davis, C.S., and Koch, G.G. (2000), *Categorical Data Analysis Using the SAS System*, Second Edition, Cary, NC: SAS Institute Inc.

Chapter 9
The MI Procedure

Chapter Table of Contents

Chapter 9
The MI Procedure

Overview

The experimental MI procedure performs multiple imputation of missing data. Missing values are an issue in a substantial number of statistical analyses. Most SAS statistical procedures exclude observations with any missing variable values from the analysis. These observations are called incomplete cases. While analyzing only complete cases has its simplicity, the information contained in the incomplete cases is lost. This approach also ignores possible systematic differences between the complete cases and the incomplete cases, and the resulting inference may not be applicable to the population of all cases, especially with a smaller number of complete cases.

Some SAS procedures use all the available cases in an analysis, that is, cases with available information. For example, the CORR procedure estimates a variable mean by using all cases with nonmissing values for this variable, ignoring the possible missing values in other variables. PROC CORR also estimates a correlation by using all cases with nonmissing values for this pair of variables. This makes better use of the available data, but the resulting correlation matrix may not be positive definite.

Another strategy for handling missing data is simple imputation, which substitutes a value for each missing value. Standard statistical procedures for complete data analysis can then be used with the filled-in data set. For example, each missing value can be imputed with the variable mean of the complete cases, or it can be imputed with the mean conditional on observed values of other variables. This approach treats missing values as if they were known in the complete-data analysis. However, single imputation does not reflect the uncertainty about the predictions of the unknown missing values, and the resulting estimated variances of the parameter estimates will be biased toward zero (Rubin 1987, p. 13).

Instead of filling in a single value for each missing value, multiple imputation (Rubin 1976; 1987) replaces each missing value with a set of plausible values that represent the uncertainty about the right value to impute. The multiply imputed data sets are then analyzed by using standard procedures for complete data and combining the results from these analyses. No matter which complete-data analysis is used, the process of combining results from different data sets is essentially the same.

Multiple imputation does not attempt to estimate each missing value through simulated values but rather to represent a random sample of the missing values. This process results in valid statistical inferences that properly reflect the uncertainty due to missing values; for example, confidence intervals with the correct probability coverage.

Multiple imputation inference involves three distinct phases:

1. The missing data are filled in *m* times to generate *m* complete data sets.
2. The *m* complete data sets are analyzed using standard statistical analyses.
3. The results from the *m* complete data sets are combined to produce inferential results.

The new MI procedure creates multiply imputed data sets for incomplete multivariate data. It uses methods that incorporate appropriate variability across the *m* imputations. The method of choice depends on the patterns of missingness. A data set with variables Y_1, Y_2, ..., Y_p (in that order) is said to have a monotone missing pattern when the event that a variable Y_j is missing for a particular individual implies that all subsequent variables Y_k, $k > j$, are missing for that individual.

For data sets with monotone missing patterns, either a parametric regression method (Rubin 1987) that assumes multivariate normality or a nonparametric method that uses propensity scores (Rubin 1987; Lavori, Dawson, and Shera 1995) is appropriate. For data sets with arbitrary missing patterns, a Markov Chain Monte Carlo (MCMC) method (Schafer 1997) that assumes multivariate normality is used to impute all missing values or just enough missing values to make the imputed data sets have monotone missing patterns.

Once the *m* complete data sets are analyzed using standard SAS procedures, the new MIANALYZE procedure can be used to generate valid statistical inferences about these parameters by combining results from the *m* analyses. These two procedures are available in experimental form in Release 8.2 of the SAS System.

Often, as few as three to five imputations are adequate in multiple imputation (Rubin 1996, p. 480). The relative efficiency of the small m imputation estimator is high for cases with little missing information (Rubin 1987, p. 114). Also see the "Multiple Imputation Efficiency" section on page 174.

Multiple imputation inference assumes that the model (variables) you used to analyze the multiply imputed data (the analyst's model) is the same as the model used to impute missing values in multiple imputation (the imputer's model). But in practice, the two models may not be the same. The consequence for different scenarios (Schafer 1997, pp. 139–143) is discussed in the "Imputer's Model Versus Analyst's Model" section on page 174.

In addition to the multiple imputation method, a simulation-based method of parameter simulation can also be used to analyze the data for many incomplete-data problems. Although the MI procedure does not offer a simulation-based method of parameter simulation, the choice between the two methods (Schafer 1997, pp. 89–90, 135–136) is examined in the "Parameter Simulation Versus Multiple Imputation" section on page 175.

Getting Started

Consider the following Fitness data set that has been altered to contain an arbitrary pattern of missingness:

```
*----------------- Data on Physical Fitness -----------------*
| These measurements were made on men involved in a physical |
| fitness course at N.C. State University.                   |
| Only selected variables of                                 |
| Oxygen (oxygen intake, ml per kg body weight per minute),  |
| Runtime (time to run 1.5 miles in minutes), and            |
| RunPulse (heart rate while running) are used.              |
| Certain values were changed to missing for the analysis.   |
*------------------------------------------------------------*;
```

```
data FitMiss;
   input Oxygen RunTime RunPulse @@;
   datalines;
44.609  11.37  178     45.313  10.07  185
54.297   8.65  156     59.571    .      .
49.874   9.22    .     44.811  11.63  176
  .     11.95  176          .  10.85    .
39.442  13.08  174     60.055   8.63  170
50.541    .      .     37.388  14.03  186
44.754  11.12  176     47.273    .      .
51.855  10.33  166     49.156   8.95  180
40.836  10.95  168     46.672  10.00    .
46.774  10.25    .     50.388  10.08  168
39.407  12.63  174     46.080  11.17  156
45.441   9.63  164       .      8.92    .
45.118  11.08    .     39.203  12.88  168
45.790  10.47  186     50.545   9.93  148
48.673   9.40  186     47.920  11.50  170
47.467  10.50  170
;
```

Suppose that the data are multivariate normally distributed and the missing data are missing at random (MAR). That is, the probability that an observation is missing can depend on the observed variable values of the individual, but not on the missing variable values of the individual. See the "Statistical Assumptions for Multiple Imputation" section on page 154 for a detailed description of the MAR assumption.

The following statements invoke the MI procedure and impute missing values for the FitMiss data set.

```
proc mi data=FitMiss seed=37851 mu0=50 10 180 out=outmi;
   var Oxygen RunTime RunPulse;
run;
```

The MI Procedure

Model Information

Data Set	WORK.FITMISS
Method	MCMC
Multiple Imputation Chain	Single Chain
Initial Estimates for MCMC	EM Posterior Mode
Start	Starting Value
Prior	Jeffreys
Number of Imputations	5
Number of Burn-in Iterations	200
Number of Iterations	100
Seed for random number generator	37851

Figure 9.1. Model Information

The "Model Information" table describes the method used in the multiple imputation process. By default, the procedure uses the Markov Chain Monte Carlo (MCMC) method with a single chain to create five imputations. The posterior mode, the highest observed-data posterior density, with a noninformative prior, is computed from the EM algorithm and is used as the starting value for the chain.

The MI procedure takes 200 burn-in iterations before the first imputation and 100 iterations between imputations. In a Markov chain, the information in the current iteration has influence on the state of the next iteration. The burn-in iterations are iterations in the beginning of each chain that are used both to eliminate the series of dependence on the starting value of the chain and to achieve the stationary distribution. The between-imputation iterations in a single chain are used to eliminate the series of dependence between the two imputations.

The MI Procedure

Missing Data Patterns

Group	Oxygen	Run Time	Run Pulse	Freq	Percent
1	X	X	X	21	67.74
2	X	X	.	4	12.90
3	X	.	.	3	9.68
4	.	X	X	1	3.23
5	.	X	.	2	6.45

Missing Data Patterns

	Group Means		
Group	Oxygen	RunTime	RunPulse
1	46.353810	10.809524	171.666667
2	47.109500	10.137500	.
3	52.461667	.	.
4	.	11.950000	176.000000
5	.	9.885000	.

Figure 9.2. Missing Data Patterns

The "Missing Data Patterns" table lists distinct missing data patterns with corresponding frequencies and percents. Here, an "X" means that the variable is observed in the corresponding group and a "." means that the variable is missing. The table also displays group-specific variable means. The MI procedure sorts the data into groups based on whether an individual's value is observed or missing for each variable to be analyzed. For a detailed description of missing data patterns, see the "Missing Data Patterns" section on page 155.

The MI Procedure

Multiple Imputation Variance Information

Variable	Between	Within	Total	DF
		Variance		
Oxygen	0.045321	0.937239	0.991624	26.113
RunTime	0.005853	0.072217	0.079241	24.45
RunPulse	0.611864	3.247163	3.981400	19.227

Multiple Imputation Variance Information

Variable	Relative Increase in Variance	Fraction Missing Information
Oxygen	0.058027	0.056263
RunTime	0.097265	0.092202
RunPulse	0.226116	0.197941

Figure 9.3. Variance Information

After the completion of *m* imputations, the "Multiple Imputation Variance Information" table displays the between-imputation variance, within-imputation variance, and total variance for combining complete-data inferences. It also displays the degrees of freedom for the total variance. The relative increase in variance due to missing values and the fraction of missing information for each variable are also displayed. A detailed description of these statistics is provided in the "Combining Inferences from Multiply Imputed Data Sets" section on page 173.

The following "Multiple Imputation Parameter Estimates" table displays the estimated mean and standard error of the mean for each variable. The inferences are based on the *t* distribution. The table also displays a 95% confidence interval for the mean and a *t* statistic with the associated *p*-value for the hypothesis that the population mean is equal to the value specified with the MU0= option. A detailed description of these statistics is provided in the "Combining Inferences from Multiply Imputed Data Sets" section on page 173.

The MI Procedure

Multiple Imputation Parameter Estimates

Variable	Mean	Std Error	95% Confidence Limits		DF
Oxygen	47.126919	0.995803	45.0804	49.1734	26.113
RunTime	10.546494	0.281498	9.9661	11.1269	24.45
RunPulse	171.621676	1.995344	167.4487	175.7946	19.227

Multiple Imputation Parameter Estimates

Variable	Minimum	Maximum	Mu0	t for H0: Mean=Mu0	Pr > |t|
Oxygen	46.849494	47.318758	50.000000	-2.89	0.0077
RunTime	10.464123	10.669193	10.000000	1.94	0.0638
RunPulse	170.623678	172.680679	180.000000	-4.20	0.0005

Figure 9.4. Parameter Estimates

In addition to the output tables, the procedure also creates a data set with imputed values. The imputed data sets are stored in the outmi data set, with the index variable _Imputation_ indicating the imputation numbers. The data set can now be analyzed using standard statistical procedures with _Imputation_ as a BY variable.

The following statements list the first ten observations of data set outmi.

```
proc print data=outmi (obs=10);
  title 'First 10 Observations of the Imputed Data Set';
run;
```

First 10 Observations of the Imputed Data Set

Obs	_Imputation_	Oxygen	RunTime	Run Pulse
1	1	44.6090	11.3700	178.000
2	1	45.3130	10.0700	185.000
3	1	54.2970	8.6500	156.000
4	1	59.5710	6.1569	138.583
5	1	49.8740	9.2200	164.163
6	1	44.8110	11.6300	176.000
7	1	46.0264	11.9500	176.000
8	1	42.3040	10.8500	182.486
9	1	39.4420	13.0800	174.000
10	1	60.0550	8.6300	170.000

Figure 9.5. Imputed Data Set

The table shows that the precision of the imputed values differs from the precision of the observed values. You can use the ROUND= option to make the imputed values consistent with the observed values.

Syntax

The following statements are available in PROC MI.

PROC MI < *options* > **;**

BY *variables* **;**
EM < *options* > **;**
FREQ *variable* **;**
MCMC < *options* > **;**
MONOTONE < *options* > **;**
TRANSFORM *transform (variables* < */ options* >*)*
< . . . *transform (variables* < */ options* >*)* > **;**
VAR *variables* **;**

The BY statement specifies groups in which separate multiple imputation analyses are performed.

The EM statement uses the EM algorithm to compute the maximum likelihood estimate (MLE) of the data with missing values, assuming a multivariate normal distribution for the data.

The FREQ statement specifies the variable that represents the frequency of occurrence for other values in the observation.

The MCMC statement uses a Markov chain Monte Carlo method to impute values for a data set with an arbitrary missing pattern. The MONOTONE statement uses either a parametric regression method or a nonparametric method based on propensity scores to impute values for a data set with a monotone missing pattern. Note that you can use either an MCMC statement or a MONOTONE statement, but not both. When neither of these two statements is specified, the MCMC method with its default options is used.

The TRANSFORM statement lists the variables to be transformed before the imputation process. The imputed values of these transformed variables will be reverse-transformed to the original forms before the imputation.

The VAR statement lists the numeric variables to be analyzed. If you omit the VAR statement, all numeric variables not listed in other statements are used.

The PROC MI statement is the only required statement for the MI procedure. The rest of this section provides detailed syntax information for each of these statements, beginning with the PROC MI statement. The remaining statements are in alphabetical order.

PROC MI Statement

PROC MI < *options* > **;**

The following table summarizes the options available in the PROC MI statement.

Table 9.1. Summary of PROC MI Options

Tasks	Options
Specify data sets	
input data set	DATA=
output data set with imputed values	OUT=
Specify imputation details	
number of imputations	NIMPUTE=
seed to begin random number generator	SEED=
units to round imputed variable values	ROUND=
maximum values for imputed variable values	MAXIMUM=
minimum values for imputed variable values	MINIMUM=
singularity tolerance	SINGULAR=
Specify statistical analysis	
level for the confidence interval, $(1 - \alpha)$	ALPHA=
means under the null hypothesis	MU0=
Control printed output	
suppress all displayed output	NOPRINT
displays univariate statistics and correlations	SIMPLE

The following options can be used in the PROC MI statement (in alphabetical order):

ALPHA=α

specifies that confidence limits be constructed for the mean estimates with confidence level $100(1 - \alpha)\%$, where $0 < \alpha < 1$. The default is ALPHA=0.05.

DATA=*SAS-data-set*

names the SAS data set to be analyzed by PROC MI. By default, the procedure uses the most recently created SAS data set.

MAXIMUM=*numbers*

specifies maximum values for imputed variables. When an intended imputed value is greater than the maximum, PROC MI redraws another value for imputation. If only one number is specified, that number is used for all variables. If more than one number is specified, you must use a VAR statement, and the specified numbers must correspond to variables in the VAR statement. A missing value indicates no restriction on the maximum for the corresponding variable. The default is MAXIMUM=., no restriction on the maximum.

The MAXIMUM= option is related to the MINIMUM= and ROUND= options, which are used to make the imputed values more consistent with the observed variable values. These options are not applicable if you specify the METHOD=PROPENSITY option in the MONOTONE statement.

When specifying a maximum for the first variable only, you must also specify a missing value after the maximum. Otherwise, the maximum is used for all variables. For example, the MAXIMUM= 100 . option sets a maximum of 100 for the first analysis variable only and no maximum for the remaining variables. The MAXIMUM= . 100 option sets a maximum of 100 for the second analysis variable only and no maximum for the other variables.

MINIMUM=*numbers*

specifies the minimum values for imputed variables. When an intended imputed value is less than the minimum, PROC MI redraws another value for imputation. If only one number is specified, that number is used for all variables. If more than one number is specified, you must use a VAR statement, and the specified numbers must correspond to variables in the VAR statement. A missing value indicates no restriction on the minimum for the corresponding variable. The default is MINIMUM=., no restriction on the minimum.

MU0=*numbers*

THETA0=*numbers*

specifies the parameter values $\boldsymbol{\mu}_0$ under the null hypothesis $\boldsymbol{\mu} = \boldsymbol{\mu}_0$ for the population means corresponding to the analysis variables. Each hypothesis is tested with a t test. If only one number is specified, that number is used for all variables. If more than one number is specified, you must use a VAR statement, and the specified numbers must correspond to variables in the VAR statement. The default is MU0=0.

If a variable is transformed as specified in a TRANSFORM statement, then the same transformation for that variable is also applied to its corresponding specified MU0= value in the t test. If the parameter values $\boldsymbol{\mu}_0$ for a transformed variable is not specified, then $\boldsymbol{\mu}_0 = 0$ is used for that transformed variable.

NIMPUTE=*number*

specifies the number of imputations. The default is NIMPUTE=5. You can specify NIMPUTE=0 to skip the imputation. In this case, only tables of model information, missing data patterns, descriptive statistics (SIMPLE option), and MLE from the EM algorithm (EM statement) are displayed.

NOPRINT

suppresses the display of all output. Note that this option temporarily disables the Output Delivery System (ODS). For more information, refer to the chapter "Using the Output Delivery System" in the *SAS/STAT User's Guide, Version 8*.

OUT=*SAS-data-set*

creates an output SAS data set containing imputation results. The data set includes an index variable, _Imputation_, to identify the imputation number. For each imputation, the data set contains all variables in the input data set with missing values replaced by the imputed values. See the "Output Data Sets" section on page 171 for a description of this data set.

If you want to create a permanent SAS data set, you must specify a two-level name. For more information on permanent SAS data sets, refer to the section "SAS Files" in *SAS Language Reference: Concepts, Version 8*.

ROUND=*numbers*

specifies the units to round variables in the imputation. If only one number is specified, that number is used for all variables. If more than one number is specified, you must use a VAR statement, and the specified numbers must correspond to variables in the VAR statement. The default number is a missing value, which indicates no rounding for imputed variables.

When specifying a roundoff unit for the first variable only, you must also specify a missing value after the roundoff unit. Otherwise, the roundoff unit is used for all variables. For example, the option "ROUND= 10 ." sets a roundoff unit of 10 for the first analysis variable only and no rounding for the remaining variables. The option "ROUND= . 10" sets a roundoff unit of 10 for the second analysis variable only and no rounding for other variables.

You can use the ROUND= option to set the precision of imputed values. For example, with a roundoff unit of 0.001, each value is rounded to the nearest multiple of 0.001. That is, each value has three significant digits after the decimal point. See Example 9.3 for a usage of this option.

SEED=*number*

specifies a positive integer. PROC MI uses the value of the SEED= option to start the pseudo-random number generator. The default is a value generated from reading the time of day from the computer's clock. However, in order to duplicate the results under identical situations, you must control the value of the seed explicitly rather than rely on the clock reading.

The seed information is displayed in the "Model Information" table so that the results can be reproduced by specifying this seed with the SEED= option. You need to specify the same seed number in the future to reproduce the results.

SIMPLE

displays simple descriptive univariate statistics and pairwise correlations from available cases. For a detailed description of these statistics, see the "Descriptive Statistics" section on page 152.

SINGULAR=*p*

specifies the criterion for determining the singularity of a covariance matrix, where $0 < p < 1$. The default is SINGULAR=1E$-$8.

Suppose that $\mathbf{S}$ is a covariance matrix and v is the number of variables in $\mathbf{S}$. Based on the spectral decomposition $\mathbf{S} = \mathbf{\Gamma}\mathbf{\Lambda}\mathbf{\Gamma}'$, where $\mathbf{\Lambda}$ is a diagonal matrix of eigenvalues λ_j, $j = 1, \ldots, v$, where $\lambda_i \geq \lambda_j$ when $i < j$, and $\mathbf{\Gamma}$ is a matrix with the corresponding orthonormal eigenvectors of $\mathbf{S}$ as columns, $\mathbf{S}$ is considered singular when an eigenvalue λ_j is less than $p\bar{\lambda}$, where the average $\bar{\lambda} = \sum_{k=1}^{v} \lambda_k / v$.

BY Statement

BY *variables* **;**

You can specify a BY statement with PROC MI to obtain separate analyses on observations in groups defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data using the SORT procedure with a similar BY statement.
- Specify the BY statement option NOTSORTED or DESCENDING in the BY statement for the MI procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables using the DATASETS procedure.

For more information on the BY statement, refer to the discussion in *SAS Language Reference: Concepts, Version 8*. For more information on the DATASETS procedure, refer to the discussion in the *SAS Procedures Guide, Version 8*.

EM Statement

EM < *options* > **;**

The expectation-maximization (EM) algorithm is a technique for maximum likelihood estimation in parametric models for incomplete data. The EM statement uses the EM algorithm to compute the MLE for $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, the means and covariance matrix, of a multivariate normal distribution from the input data set with missing values. PROC MI uses the means and standard deviations from available cases as the initial estimates for the EM algorithm. The correlations are set to zero.

You can also use the EM statement with the NIMPUTE=0 option in the PROC statement to compute the EM estimates without multiple imputation, as shown in Example 9.1 in the “Examples” section on page 177.

The following five options are available with the EM statement.

CONVERGE=*p*
sets the convergence criterion. The value must be between 0 and 1. The iterations are considered to have converged when the maximum change in the parameter estimates between iteration steps is less than the value specified. The change is a relative change if the parameter is greater than 0.01 in absolute value; otherwise, it is an absolute change. By default, CONVERGE=1E-4.

ITPRINT
prints the iteration history in the EM algorithm.

MAXITER=*number*
specifies the maximum number of iterations used in the EM algorithm. The default is MAXITER=200.

OUTEM=*SAS-data-set*
creates an output SAS data set of TYPE=COV containing the MLE of the parameter vector (μ, Σ). These estimates are computed with the EM algorithm. See the "Output Data Sets" section on page 171 for a description of this output data set.

OUTITER $<$ **(options)** $>$ **=***SAS-data-set*
creates an output SAS data set of TYPE=COV containing parameters for each iteration. The data set includes a variable named _Iteration_ to identify the iteration number.

The parameters in the output data set depend on the options specified. You can specify the MEAN and COV options to output the mean and covariance parameters. When no options are specified, the output data set contains the mean parameters for each iteration. See the "Output Data Sets" section on page 171 for a description of this data set.

FREQ Statement

FREQ *variable* **;**

If one variable in your input data set represents the frequency of occurrence for other values in the observation, specify the variable name in a FREQ statement. PROC MI then treats the data set as if each observation appears n times, where n is the value of the FREQ variable for the observation. If the value of the FREQ variable is less than one, the observation is not used in the analysis. Only the integer portion of the value is used. The total number of observations is considered to be equal to the sum of the FREQ variable when PROC MI calculates significance probabilities.

MCMC Statement

MCMC < *options* > **;**

The MCMC statement specifies the details of the MCMC method for imputation. The following table summarizes the options available for the MCMC statement.

Table 9.2. Summary of Options in MCMC

Tasks	Options
Specify data sets	
input parameter estimates for imputations	INEST=
output parameter estimates used in imputations	OUTEST=
output parameter estimates used in iterations	OUTITER=
Specify imputation details	
monotone/full imputation	IMPUTE=
single/multiple chain	CHAIN=
number of burn-in iterations for each chain	NBITER=
number of iterations between imputations in a chain	NITER=
initial parameter estimates for MCMC	INITIAL=
prior parameter information	PRIOR=
starting parameters	START=
Specify output graphics	
displays time-series plots	TIMEPLOT=
displays autocorrelation plots	ACFPLOT=
graphics catalog name for saving graphics output	GOUT=
Control printed output	
displays worst linear function	WLF
displays initial parameter values for MCMC	DISPLAYINIT

The following are the options available for the MCMC statement (in alphabetical order):

ACFPLOT < *(options < / display-options >)* >
displays the autocorrelation function plots of parameters from iterations.

The available options are:

COV < *(< variables > < variable1*variable2 > < . . . variable1*variable2 >)* >
displays plots of variances for variables in the list and covariances for pairs of variables in the list. When the option COV is specified without variables, variances for all variables and covariances for all pairs of variables are used.

MEAN < *(variables)* >
displays plots of means for variables in the list. When the option MEAN is specified without variables, all variables are used.

WLF
displays the plot for the worst linear function.

When the ACFPLOT is specified without the preceding options, the procedure displays plots of means for all variables that are used.

The display-options provide additional information for the autocorrelation function plots. The available display-options are:

CCONF=*color*
specifies the color of the displayed confidence limits. The default is CCONF=BLACK.

CFRAME=*color*
specifies the color for filling the area enclosed by the axes and the frame. By default, this area is not filled.

CNEEDLES=*color*
specifies the color of the vertical line segments (needles) that connect autocorrelations to the reference line. The default is CNEEDLES=BLACK.

CREF=*color*
specifies the color of the displayed reference line. The default is CREF=BLACK.

CSYMBOL=*color*
specifies the color of the displayed data points. The default is CSYMBOL=BLACK.

HSYMBOL=*number*
specifies the height for data points in percentage screen units. The default is HSYMBOL=1.

LCONF=*linetype*
specifies the line type for the displayed confidence limits. The default is LREF=1, a solid line.

LOG
requests that the logarithmic transformations of parameters be used to compute the autocorrelations. It's generally used for the variances of variables. When a parameter has values less than or equal to zero, the corresponding plot is not created.

LREF=*linetype*
specifies the line type for the displayed reference line. The default is LREF=3, a dashed line.

NLAG=*number*
specifies the maximum lag of the series. The default is NLAG=20. The autocorrelations at each lag are displayed in the graph.

SYMBOL=*value*

specifies the symbol for data points in percentage screen units. The default is SYMBOL=STAR.

TITLE=*'string'*

specifies the title to be displayed in the autocorrelation function plots. The default is TITLE='Autocorrelation Plot'.

WCONF=*number*

specifies the width for the displayed confidence limits in percentage screen units. If you specify the WCONF=0 option, the confidence limits are not displayed. The default is WCONF=1.

WNEEDLES=*number*

specifies the width for the displayed needles that connect autocorrelations to the reference line in percentage screen units. If you specify the WNEEDLES=0 option, the needles are not displayed. The default is WNEEDLES=1.

WREF=*number*

specifies the width for the displayed reference line in percentage screen units. If you specify the WREF=0 option, the reference line is not displayed. The default is WREF=1.

For example, the statement

```
acfplot( mean( y1) cov(y1) /log);
```

requests autocorrelation function plots for the means and variances of the variable `y1`, respectively. Logarithmic transformations of both the means and variances are used in the plots. For a detailed description of the autocorrelation function plot, see the "Autocorrelation Function Plot" section on page 169; refer also to Schafer (1997, pp. 120-126) and the *SAS/ETS User's Guide, Version 8*.

CHAIN=SINGLE | MULTIPLE

specifies whether a single chain is used for all imputations or a separate chain is used for each imputation. The default is CHAIN=SINGLE.

DISPLAYINIT

displays initial parameter values in the MCMC process for each imputation.

GOUT=*graphics-catalog*

specifies the graphics catalog for saving graphics output from PROC MI. The default is WORK.GSEG. For more information, refer to the chapter "The GREPLAY Procedure" in *SAS/GRAPH Software: Reference, Version 8*.

IMPUTE=FULL | MONOTONE

specifies whether a full-data imputation is used for all missing values or a monotone-data imputation is used for a subset of missing values to make the imputed data sets have a monotone missing pattern. The default is IMPUTE=FULL. When IMPUTE=MONOTONE is specified, the order in the VAR statement is used to complete the monotone pattern.

INEST=*SAS-data-set*
names a SAS data set of TYPE=EST containing parameter estimates for imputations. These estimates are used to impute values for observations in the DATA= data set. A detailed description of the data set is provided in the "Input Data Sets" section on page 170.

INITIAL=EM < (*options*) >
INITIAL=INPUT=*SAS-data-set*
specifies the initial mean and covariance estimates for the MCMC process. The default is INITIAL=EM.

You can specify INITIAL=INPUT=*SAS-data-set* to read the initial estimates of the mean and covariance matrix for each imputation from a SAS data set. See the "Input Data Sets" section on page 170 for a description of this data set.

With INITIAL=EM, PROC MI derives parameter estimates for a posterior mode, the highest observed-data posterior density, from the EM algorithm. The MLE from EM is used to start the EM algorithm for the posterior mode, and the resulting EM estimates are used to begin the MCMC process.

The following four options are available with INITIAL=EM.

BOOTSTRAP < =*number* >
requests bootstrap resampling, which uses a simple random sample with replacement from the input data set for the initial estimate. You can explicitly specify the number of observations in the random sample. Alternatively, you can implicitly specify the number of observations in the random sample by specifying the proportion $p, 0 < p <= 1$, to request $[np]$ observations in the random sample, where *n* is the number of observations in the data set and $[np]$ is the integer part of np. This produces an overdispersed initial estimate that provides different starting values for the MCMC process. If you specify the BOOTSTRAP option without the number, *p*=0.75 is used by default.

CONVERGE=*p*
sets the convergence criterion. The value must be between 0 and 1. The iterations are considered to have converged when the maximum change in the parameter estimates between iteration steps is less than the value specified. The change is a relative change if the parameter is greater than 0.01 in absolute value; otherwise, it is an absolute change. By default, CONVERGE=1E-4.

ITPRINT
prints the iteration history in the EM algorithm for the posterior mode.

MAXITER=*number*
specifies the maximum number of iterations used in the EM algorithm. The default is MAXITER=200.

NBITER=*number*
specifies the number of burn-in iterations before the first imputation in each chain. The default is NBITER=200.

NITER=*number*
specifies the number of iterations between imputations in a single chain. The default is NITER=100.

OUTEST=*SAS-data-set*
creates an output SAS data set of TYPE=EST. The data set contains parameter estimates used in each imputation. The data set also includes a variable named _Imputation_ to identify the imputation number. See the "Output Data Sets" section on page 171 for a description of this data set.

OUTITER < **(options)** > **=***SAS-data-set*
creates an output SAS data set of TYPE=COV containing parameters used in the imputation step for each iteration. The data set includes variables named _Imputation_ and _Iteration_ to identify the imputation number and iteration number.

The parameters in the output data set depend on the options specified. You can specify options MEAN, STD, COV, LR, LR_POST, and WLF to output parameters of means, standard deviations, covariances, -2 log LR statistic, -2 log LR statistic of the posterior mode, and the worst linear function. When no options are specified, the output data set contains the mean parameters used in the imputation step for each iteration. See the "Output Data Sets" section on page 171 for a description of this data set.

PRIOR=*name*
specifies the prior information for the means and covariances. Valid values for *name* are as follows:

JEFFREYS	specifies a noninformative prior.
RIDGE=*number*	specifies a ridge prior.
INPUT=*SAS-data-set*	specifies a data set containing prior information.

For a detailed description of the prior information, see the "Bayesian Estimation of the Mean Vector and Covariance Matrix" section on page 161 and the "Posterior Step" section on page 162. If you do not specify the PRIOR= option, the default is PRIOR=JEFFREYS.

The PRIOR=INPUT= option specifies a TYPE=COV data set from which the prior information of the mean vector and the covariance matrix is read. See the "Input Data Sets" section on page 170 for a description of this data set.

START=VALUE | DIST
specifies that the initial parameter estimates are used as either the starting value (START=VALUE) or as the starting distribution (START=DIST) in the first imputation step of each chain. The default is START=VALUE.

TIMEPLOT < *(options < / display-options >)* >
displays the time-series plots of parameters from iterations.

The available options are:

COV < *(< variables > < variable1*variable2 > < . . . variable1*variable2 >)* >
displays plots of variances for variables in the list and covariances for pairs of variables in the list. When the option COV is specified without variables, variances for all variables and covariances for all pairs of variables are used.

MEAN < *(variables)* >
displays plots of means for variables in the list. When the option MEAN is specified without variables, all variables are used.

WLF
displays the plot for the worst linear function.

When the TIMEPLOT is specified without the preceding options, the procedure displays plots of means for all variables are used.

The display-options provide additional information for the time-series plots. The available display-options are:

CFRAME=*color*
specifies the color for filling the area enclosed by the axes and the frame. By default, this area is not filled.

CSYMBOL=*color*
specifies the color of the data points to be displayed in the time-series plots. The default is CSYMBOL=BLACK.

HSYMBOL=*number*
specifies the height for data points in percentage screen units. The default is HSYMBOL=1.

LOG
requests that the logarithmic transformations of parameters be used. It's generally used for the variances of variables. When a parameter value is less than or equal to zero, the value is not displayed in the corresponding plot.

SYMBOL=*value*
specifies the symbol for data points in percentage screen units. The default is SYMBOL=PLUS.

TITLE=*'string'*
specifies the title to be displayed in the time-series plots. The default is TITLE='Time-series Plot for Iterations'.

For a detailed description of the time-series plot, see the "Time-Series Plot" section on page 168 and Schafer (1997, pp. 120–126).

WLF
displays the worst linear function of parameters. This scalar function of parameters μ and Σ is "worst" in the sense that its values from iterations converge most slowly among parameters. For a detailed description of this statistic, see the "Worst Linear Function of Parameters" section on page 168.

MONOTONE Statement

MONOTONE < *options* > **;**

The MONOTONE statement specifies an imputation method for data sets with monotone missingness. You must also specify a VAR statement and the data set must have a monotone missing pattern with variables ordered in the VAR list. When both MONOTONE and MCMC statements are specified, the MONOTONE statement is not used.. You can specify the following options in a MONOTONE statement.

METHOD=REG | REGRESSION
METHOD=PROPENSITY < / **NGROUPS =** *number*>
specifies the imputation method for a data set with a monotone missing pattern. You can specify either METHOD=REG, a parametric regression method, or METHOD=PROPENSITY, a nonparametric method based on propensity scores. The default is METHOD=REG.

When METHOD=PROPENSITY is specified, the MAXIMUM=, MINIMUM=, and ROUND= options, which make the imputed values more consistent with the observed variable values, are not applicable.

NGROUPS=*number*
specifies the number of groups based on propensity scores for METHOD=PROPENSITY. The default is NGROUPS=5.

See the "Regression Method for Monotone Missing Data" section on page 157 for a detailed description of the regression method, and the "Propensity Score Method for Monotone Missing Data" section on page 158 for the propensity score method.

TRANSFORM Statement

TRANSFORM *transform (variables* < */ options* >*)*
< . . . *transform (variables* < */ options* >*)* > **;**

The TRANSFORM statement lists the transformations and their associated variables to be transformed. The options are transformation options that provide additional information for the transformation.

The MI procedure assumes that the data are from a multivariate normal distribution when either the regression method or the MCMC method is used. When some variables in a data set are clearly non-normal, it is useful to transform these variables to conform to the multivariate normality assumption. With a TRANSFORM statement, variables are transformed before the imputation process and these transformed variable values are displayed in all of the results. When you specify an OUT= option, the variable values are reverse-transformed to create the imputed data set.

The following transformations can be used as the *transform* in the TRANSFORM statement.

BOXCOX
specifies the Box-Cox transformation of variables. The variable Y is transformed to $\frac{(Y+c)^{\lambda}-1}{\lambda}$, where c is a constant such that each value of $Y + c$ must be positive and the constant $\lambda > 0$.

EXP
specifies the exponential transformation of variables. The variable Y is transformed to $e^{(Y+c)}$, where c is a constant.

LOG
specifies the logarithmic transformation of variables. The variable Y is transformed to $\log(Y + c)$, where c is a constant such that each value of Y+c must be positive.

LOGIT
specifies the logit transformation of variables. The variable Y is transformed to $\log(\frac{Y/c}{1-Y/c})$, where the constant $c > 0$ and the values of Y/c must be between 0 and 1.

POWER
specifies the power transformation of variables. The variable Y is transformed to $(Y + c)^{\lambda}$, where c is a constant such that each value of $Y + c$ must be positive and the constant $\lambda \neq 0$.

The following options provide the constant c and λ values in the transformations.

C=*number*
specifies the c value in the transformation. The default is $c = 1$ for logit transformation and $c = 0$ for other transformations.

LAMBDA=*number*
specifies the λ value in the power and Box-Cox transformations. You must specify the λ value for these two transformations.

For example, the statement

```
transform log(y1) power(y2/c=1 lambda=.5);
```

requests that variables $\log(\mathrm{y1})$, a logarithmic transformation for the variable y1, and $\sqrt{\mathrm{y2}+1}$, a power transformation for the variable y2, be used in the imputation.

If the MU0= option is used to specify a parameter value $\boldsymbol{\mu}_0$ for a transformed variable, the same transformation for the variable is also applied to its corresponding MU0= value in the t test. Otherwise, $\boldsymbol{\mu}_0 = 0$ is used for the transformed variable. See Example 9.7 for a usage of the TRANSFORM statement.

VAR Statement

VAR *variables* **;**

The VAR statement lists the variables to be analyzed. The variables must be numeric. If you omit the VAR statement, all numeric variables not mentioned in other statements are used. The VAR statement is required if you specify a MONOTONE statement, an IMPUTE=MONOTONE option in the MCMC statement, or more than one number in the MU0=, MAXIMUM=, MINIMUM=, or ROUND= option.

Details

Descriptive Statistics

Suppose $\mathbf{Y}$ is the $n \times p$ matrix of complete data, which may not be fully observed, n_0 is the number of observations fully observed, and n_j is the number of observations with observed values for variable Y_j.

With complete cases, the sample mean vector is

$$\overline{\mathbf{y}} = \frac{1}{n_0} \sum \mathbf{y}_i$$

and the CSSCP matrix is

$$\sum (\mathbf{y}_i - \overline{\mathbf{y}})(\mathbf{y}_i - \overline{\mathbf{y}})'$$

where each summation is over the fully observed observations.

The sample covariance matrix is

$$\mathbf{S} = \frac{1}{n_0 - 1} \sum (\mathbf{y}_i - \overline{\mathbf{y}})(\mathbf{y}_i - \overline{\mathbf{y}})'$$

and is an unbiased estimate of the covariance matrix.

The correlation matrix $\mathbf{R}$ containing the Pearson product-moment correlations of the variables is derived by scaling the corresponding covariance matrix:

$$\mathbf{R} = \mathbf{D}^{-1} \mathbf{S}\, \mathbf{D}^{-1}$$

where $\mathbf{D}$ is a diagonal matrix whose diagonal elements are the square roots of the diagonal elements of $\mathbf{S}$.

With available cases, the corrected sum of squares for variable Y_j is

$$\sum (y_{ji} - \overline{y}_j)^2$$

where $\overline{y}_j = \frac{1}{n_j} \sum y_{ji}$ is the sample mean and each summation is over observations with observed values for variable Y_j.

The variance is

$$s_{jj}^2 = \frac{1}{n_j - 1} \sum (y_{ji} - \overline{y}_j)^2$$

The correlations for available cases contain pairwise correlations for each pair of variables. Each correlation is computed from all observations that have nonmissing values for the corresponding pair of variables.

EM Algorithm for Data with Missing Values

The EM algorithm (Dempster, Laird, and Rubin 1977) is a technique that finds maximum likelihood estimates in parametric models for incomplete data. The books by Little and Rubin (1987), Schafer (1997), and McLachlan and Krishnan (1997) provide detailed description and applications of the EM algorithm.

The EM algorithm is an iterative procedure that finds the MLE of the parameter vector by repeating the following steps:

1. The expectation E-step:
Given a set of parameter estimates, such as a mean vector and covariance matrix for a multivariate normal distribution, the E-step calculates the conditional expectation of the complete-data log likelihood given the observed data and the parameter estimates.

2. The maximization M-step:
Given a complete-data log likelihood, the M-step finds the parameter estimates to maximize the complete-data log likelihood from the E-step.

The two steps are iterated until the iterations converge.

In the EM process, the observed-data log likelihood is non-decreasing at each iteration. For multivariate normal data, suppose there are G groups with distinct missing patterns. Then the observed-data log likelihood being maximized can be expressed as

$$\ln L(\boldsymbol{\theta}|Y_{obs}) = \sum_{g=1}^{G} \ln L_g(\boldsymbol{\theta}|Y_{obs})$$

where $\ln L_g(\boldsymbol{\theta}|Y_{obs})$ is the observed-data log likelihood from the g_{th} group, and

$$\ln L_g(\boldsymbol{\theta}|Y_{obs}) = -\frac{n_g}{2} \ln |\Sigma_g| - \frac{1}{2} \sum_{ig} (\mathbf{y}_{ig} - \boldsymbol{\mu}_g)' {\Sigma_g}^{-1} (\mathbf{y}_{ig} - \boldsymbol{\mu}_g)$$

where n_g is the number of observations in the g_{th} group, the summation is over observations in the g_{th} group, $\mathbf{y}_{ig}$ is a vector of observed values corresponding to observed variables, $\boldsymbol{\mu}_g$ is the corresponding mean vector, and $\boldsymbol{\Sigma}_g$ is the associated covariance matrix.

Refer to Schafer (1997, pp. 163–181) for a detailed description of the EM algorithm for multivariate normal data.

PROC MI uses the means and standard deviations from available cases as the initial estimates for the EM algorithm. The correlations are set to zero. For a discussion of suggested starting values for the algorithm, see Schafer (1997, p. 169).

You can specify the convergence criterion with the CONVERGE= option in the EM statement. The iterations are considered to have converged when the maximum change in the parameter estimates between iteration steps is less than the value specified. You can also specify the maximum number of iterations used in the EM algorithm with the MAXITER= option.

The MI procedure displays tables of the initial parameter estimates used to begin the EM process and the MLE parameter estimates derived from EM. You can also display the EM iteration history with the option ITPRINT. PROC MI lists the iteration number, the likelihood -2 Log L, and parameter values $\boldsymbol{\mu}$ at each iteration. You can also save the MLE derived from the EM algorithm in a SAS data set specified with the OUTEM= option.

Statistical Assumptions for Multiple Imputation

The MI procedure assumes that the data are from a continuous multivariate distribution and contain missing values that can occur on any of the variables. It also assumes that the data are from a multivariate normal distribution when either the regression method or the MCMC method is used.

Suppose $\mathbf{Y}$ is the $n \times p$ matrix of complete data, which is not fully observed, and denote the observed part of $\mathbf{Y}$ by $\mathbf{Y}_{obs}$ and the missing part by $\mathbf{Y}_{mis}$. The SAS MI and MIANALYZE procedures assume that the missing data are missing at random (MAR), that is, the probability that an observation is missing can depend on $\mathbf{Y}_{obs}$, but not on $\mathbf{Y}_{mis}$ (Rubin 1976; 1987, p. 53).

To be more precise, suppose that $\mathbf{R}$ is the $n \times p$ matrix of response indicators whose elements are zero or one depending on whether the corresponding elements of Y are missing or observed. Then the MAR assumption is that the distribution of $\mathbf{R}$ can depend on Y_{obs} but not on Y_{mis}.

$$p(\mathbf{R}|Y_{obs}, Y_{mis}) = p(\mathbf{R}|Y_{obs})$$

For example, consider a trivariate data set with variables Y_1 and Y_2 fully observed, and a variable Y_3 that has missing values. MAR assumes that the probability that Y_3 is missing for an individual can be related to the individual's values of variables Y_1 and Y_2, but not to its value of Y_3. On the other hand, if a complete case and an incomplete case for Y_3 with exactly the same values for variables Y_1 and Y_2 have systematically different values, then there exists a response bias for Y_3, and MAR is violated.

The MAR assumption is not the same as missing completely at random (MCAR), which is a special case of MAR. Under the MCAR assumption, the missing data values are a simple random sample of all data values; the missingness does not depend on the values of any variables in the data set.

Furthermore, the MI and MIANALYZE procedures assume that the parameters $\boldsymbol{\theta}$ of the data model and the parameters ϕ of the model for the missing data indicators are distinct. That is, knowing the values of $\boldsymbol{\theta}$ does not provide any additional information about ϕ, and vice versa. If both the MAR and distinctness assumptions are satisfied, the missing-data mechanism is said to be ignorable (Rubin 1987, pp. 50–54; Schafer 1997, pp. 10–11) .

Missing Data Patterns

The MI procedure sorts the data into groups based on whether an individual's value is observed or missing for each variable to be analyzed. The input data set does not need to be sorted in any order.

For example, with variables Y_1, Y_2, and Y_3 (in that order) in a data set, up to eight groups of observations can be formed from the data set. The following figure displays the eight groups of observations and an unique missing pattern for each group:

Missing Data Patterns

Group	Y1	Y2	Y3
1	X	X	X
2	X	X	.
3	X	.	X
4	X	.	.
5	.	X	X
6	.	X	.
7	.	.	X
8	.	.	.

Figure 9.6. Missing Data Patterns

Here, an "X" means that the variable is observed in the corresponding group and a "." means that the variable is missing.

The variable order is used to derive the order of the groups from the data set, and thus determines the order of missing values in the data to be imputed. If you specify a different order of variables in the VAR statement, then the results are different even if the other specifications remain the same.

A data set with variables Y_1, Y_2, ..., Y_p (in that order) is said to have a monotone missing pattern when the event that a variable Y_j is missing for a particular individual implies that all subsequent variables Y_k, $k > j$, are missing for that individual. Alternatively, when a variable Y_j is observed for a particular individual, it is assumed that all previous variables Y_k, $k < j$, are also observed for that individual.

For example, the following figure displays a data set of three variables with a monotone missing pattern. Note that this data set does not have any observations with missing patterns such as in Groups 3, 5, 6, 7, or 8 in the previous example.

Monotone Missing Data Patterns

Group	Y1	Y2	Y3
1	X	X	X
2	X	X	.
3	X	.	.

Figure 9.7. Monotone Missing Patterns

Imputation Mechanisms

This section describes the three methods for multiple imputation that are available in the MI procedure. The method of choice depends on the patterns of missingness in the data.

- For data sets with monotone missing patterns, either a parametric regression method (Rubin 1987) that assumes multivariate normality or a nonparametric method that uses propensity scores (Rubin 1987; Lavori, Dawson, and Shera 1995) is appropriate.
- For data sets with arbitrary missing patterns, a Markov Chain Monte Carlo (MCMC) method (Schafer 1997) that assumes multivariate normality is used to impute either all missing values or just enough missing values to make the imputed data sets have monotone missing patterns.

With a monotone missing data pattern, you have greater flexibility in your choice of strategies. For example, in addition to the MCMC method, you can also implement other methods, such as a regression method, that do not use Markov chains.

With an arbitrary missing data pattern, you can often use the MCMC method, which creates multiple imputations by drawing simulations from a Bayesian predictive distribution for normal data. Another way to handle a data set with an arbitrary missing data pattern is to use the MCMC approach to impute enough values to make the missing data pattern monotone. Then, you can use a more flexible imputation method. This approach is described in the "Producing Monotone Missingness with the MCMC Method" section on page 164.

Although the regression and MCMC methods assume multivariate normality, inferences based on multiple imputation can be robust to departures from the multivariate normality if the amount of missing information is not large. It often makes sense to use a normal model to create multiple imputations even when the observed data are somewhat non-normal, as supported by simulation studies described in Schafer (1997) and the original references therein.

You can also use a TRANSFORM statement to transform variables to conform to the multivariate normality assumption. With a TRANSFORM statement, variables are transformed before the imputation process and then are reverse-transformed to create the imputed data set.

Li (1988) presented an argument for convergence of the MCMC method in the continuous case in theory and used it to create imputations for incomplete multivariate continuous data. But in practice, it is not easy to check the convergence of a Markov chain, especially for parameters from a large number of variables. PROC MI generates statistics and plots that you can use to check for convergence of the MCMC process. The details are described in the "Convergence in MCMC" section on page 167.

Regression Method for Monotone Missing Data

A data set with variables Y_1, Y_2, ..., Y_p (in that order) is said to have a monotone missing pattern when the event that a variable Y_j is observed for a particular individual implies that all previous variables Y_k, $k < j$, are also observed for that individual.

In the regression method, a regression model is fitted for each variable with missing values, with the previous variables as covariates. Based on the fitted regression coefficients, a new regression model is simulated from the posterior predictive distribution of the parameters and is used to impute the missing values for each variable (Rubin 1987, pp. 166–167). The process is repeated sequentially for variables with missing values. That is, for a variable Y_j with missing values, a model

$$Y_j = \beta_0 + \beta_1 Y_1 + \beta_2 Y_2 + \ldots + \beta_{j-1} Y_{j-1}$$

is fitted using observations with observed values for variables Y_1, Y_2, ..., Y_j.

The fitted model includes the regression parameter estimates $\hat{\beta} = (\hat{\beta}_0, \hat{\beta}_1, ..., \hat{\beta}_{j-1})$ and the associated covariance matrix $\hat{\sigma}_j^2 \mathbf{V}_j$, where $\mathbf{V}_j$ is the usual $\mathbf{X}'\mathbf{X}$ inverse matrix derived from the intercept and variables $Y_1, Y_2, ..., Y_{j-1}$.

For each imputation, new parameters $\beta_* = (\beta_{*0}, \beta_{*1}, ..., \beta_{*(j-1)})$ and σ_{*j}^2 are drawn from the posterior predictive distribution of the parameters. That is, they are simulated from $(\hat{\beta}_0, \hat{\beta}_1, ..., \hat{\beta}_{j-1})$, σ_j^2, and $\mathbf{V}_j$. The variance is drawn as

$$\sigma_{*j}^2 = \hat{\sigma}_j^2 (n_j - j)/g$$

where g is a $\chi^2_{n_j - j}$ random variate and n_j is the number of nonmissing observations for Y_j. The regression coefficients are drawn as

$$\beta_* = \hat{\beta} + \sigma_{*j} \mathbf{V}'_{hj} \mathbf{Z}$$

where $\mathbf{V}'_{hj}$ is the upper triangular matrix in the Cholesky decomposition, $\mathbf{V}_j = \mathbf{V}'_{hj} \mathbf{V}_{hj}$, and $\mathbf{Z}$ is a vector of j independent random normal variates.

The missing values are then replaced by

$$\beta_{*0} + \beta_{*1}\, y_1 + \beta_{*2}\, y_2 + \ldots + \beta_{*(j-1)}\, y_{j-1} + z_i\, \sigma_{*j}$$

where $y_1, y_2, ..., y_{j-1}$ are the covariate values of the first $j - 1$ variables and z_i is a simulated normal deviate.

Propensity Score Method for Monotone Missing Data

A propensity score is generally defined as the conditional probability of assignment to a particular treatment given a vector of observed covariates (Rosenbaum and Rubin 1983). In the propensity score method, for each variable with missing values, a propensity score is generated for each observation to estimate the probability that the observation is missing. The observations are then grouped based on these propensity scores, and an approximate Bayesian bootstrap imputation (Rubin 1987, p. 124) is applied to each group (Lavori, Dawson, and Shera 1995).

A data set with variables Y_1, Y_2, ..., Y_p (in that order) is said to have a monotone missing pattern when the event that a variable Y_j is observed for a particular individual implies that all previous variables Y_k, $k < j$, are also observed for that individual. The propensity score method uses the following steps to impute values for each variable Y_j with missing values:

1. Create an indicator variable R_j with the value 0 for observations with missing Y_j and 1 otherwise.

2. Fit a logistic regression model

$$\text{logit}(p_j) = \beta_0 + \beta_1 Y_1 + \beta_2 Y_2 + \ldots + \beta_{j-1} Y_{j-1}$$

where $p_j = Pr(R_j = 0|Y_1, Y_2, ..., Y_{j-1})$ and $\text{logit}(p) = \log(p/(1-p))$.

3. Create a propensity score for each observation to estimate the probability that it is missing.

4. Divide the observations into a fixed number of groups (typically assumed to be five) based on these propensity scores.

5. Apply an approximate Bayesian bootstrap imputation to each group. In group k, suppose that Y_{obs} denotes the n_1 observations with nonmissing Y_j values and Y_{mis} denotes the n_0 observations with missing Y_j. The approximate Bayesian bootstrap imputation first draws n_1 observations randomly with replacement from Y_{obs} to create a new data set Y^*_{obs}. This is a nonparametric analogue of drawing parameters from the posterior predictive distribution of the parameters. The process then draws the n_0 values for Y_{mis} randomly with replacement from Y^*_{obs}.

Steps 1 through 5 are repeated sequentially for each variable with missing values.

Note that the propensity score method was originally designed for a randomized experiment with repeated measures on the response variables. The goal was to impute the missing values on the response variables. The method uses only the covariate information that is associated with whether the imputed variable values are missing. It does not use correlations among variables. It is effective for inferences about the distributions of individual imputed variables, such as an univariate analysis, but it is not appropriate for analyses involving relationship among variables, such as a regression analysis. It can also produce badly biased estimates of regression coefficients when data on predictor variables are missing (Allison 2000).

MCMC Method for Arbitrary Missing Data

The Markov Chain Monte Carlo (MCMC) method originated in physics as a tool for exploring equilibrium distributions of interacting molecules. In statistical applications, it is used to generate pseudo-random draws from multidimensional and otherwise intractable probability distributions via Markov chains. A Markov chain is a sequence of random variables in which the distribution of each element depends only on the value of the previous one.

In MCMC simulation, one constructs a Markov chain long enough for the distribution of the elements to stabilize to a stationary distribution, which is the distribution of interest. By repeatedly simulating steps of the chain, the method simulates draws from the distribution of interest. Refer to Schafer (1997) for a detailed discussion of this method.

In Bayesian inference, information about unknown parameters is expressed in the form of a posterior probability distribution. This posterior distribution is computed using Bayes' theorem

$$p(\boldsymbol{\theta}|y) = \frac{p(y|\boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(y|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}}$$

MCMC has been applied as a method for exploring posterior distributions in Bayesian inference. That is, through MCMC, one can simulate the entire joint posterior distribution of the unknown quantities and obtain simulation-based estimates of posterior parameters that are of interest.

In many incomplete data problems, the observed-data posterior $p(\boldsymbol{\theta}|Y_{obs})$ is intractable and cannot easily be simulated. However, when Y_{obs} is augmented by an estimated/simulated value of the missing data Y_{mis}, the complete-data posterior $p(\boldsymbol{\theta}|Y_{obs}, Y_{mis})$ is much easier to simulate. Assuming that the data are from a multivariate normal distribution, data augmentation can be applied to Bayesian inference with missing data by repeating the following steps:

1. **The imputation I-step:**
Given an estimated mean vector and covariance matrix, the I-step simulates the missing values for each observation independently. That is, if you denote the variables with missing values for observation *i* by $Y_{i(mis)}$ and the variables with observed values by $Y_{i(obs)}$, then the I-step draws values for $Y_{i(mis)}$ from a conditional distribution for $Y_{i(mis)}$ given $Y_{i(obs)}$.

2. **The posterior P-step:**
Given a complete sample, the P-step simulates the posterior population mean vector and covariance matrix. These new estimates are then used in the next I-step. Without prior information about the parameters, a noninformative prior is used. You can also use other informative priors. For example, a prior information about the covariance matrix can be helpful to stabilize the inference about the mean vector for a near singular covariance matrix.

The two steps are iterated long enough for the results to be reliable for a multiply imputed data set (Schafer 1997, p. 72). That is, with a current parameter estimate $\boldsymbol{\theta}^{(t)}$ at the tth iteration, the I-step draws $Y_{mis}^{(t+1)}$ from $p(Y_{mis}|Y_{obs}, \boldsymbol{\theta}^{(t)})$ and the P-step draws $\boldsymbol{\theta}^{(t+1)}$ from $p(\boldsymbol{\theta}|Y_{obs}, Y_{mis}^{(t+1)})$.

This creates a Markov chain

$(Y_{mis}^{(1)}, \boldsymbol{\theta}^{(1)})$, $(Y_{mis}^{(2)}, \boldsymbol{\theta}^{(2)})$, ... ,

which converges in distribution to $p(Y_{mis}, \boldsymbol{\theta}|Y_{obs})$. Assuming the iterates converge to a stationary distribution, the goal is to simulate an approximately independent draw of the missing values from this distribution.

To validate the imputation results, you should repeat the process with different random number generators and starting values based on different initial parameter estimates.

The next three sections provide details for the imputation step, Bayesian estimation of the mean vector and covariance matrix, and the posterior step.

Imputation Step

In each iteration, starting with a given mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$, the imputation step draws values for the missing data from the conditional distribution Y_{mis} given Y_{obs}.

Suppose $\boldsymbol{\mu} = [\boldsymbol{\mu}_1', \boldsymbol{\mu}_2']'$ is the partitioned mean vector of two sets of variables, Y_{obs} and Y_{mis}, where $\boldsymbol{\mu}_1$ is the mean vector for variables Y_{obs} and $\boldsymbol{\mu}_2$ is the mean vector for variables Y_{mis}.

Also suppose

$$\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{12}' & \boldsymbol{\Sigma}_{22} \end{bmatrix}$$

is the partitioned covariance matrix for these variables, where $\boldsymbol{\Sigma}_{11}$ is the covariance matrix for variables Y_{obs}, $\boldsymbol{\Sigma}_{22}$ is the covariance matrix for variables Y_{mis}, and $\boldsymbol{\Sigma}_{12}$ is the covariance matrix between variables Y_{obs} and variables Y_{mis}.

By using the sweep operator (Goodnight 1979) on the pivots of the $\boldsymbol{\Sigma}_{11}$ submatrix, the matrix becomes

$$\begin{bmatrix} \boldsymbol{\Sigma}_{11}^{-1} & \boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\Sigma}_{12} \\ -\boldsymbol{\Sigma}_{12}'\boldsymbol{\Sigma}_{11}^{-1} & \boldsymbol{\Sigma}_{22.1} \end{bmatrix}$$

where $\boldsymbol{\Sigma}_{22.1} = \boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{12}'\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\Sigma}_{12}$ can be used to compute the conditional covariance matrix of $\mathbf{Y}_{mis}$ after controlling for $\mathbf{Y}_{obs}$.

For an observation with the preceding missing pattern, the conditional distribution of Y_{mis} given $Y_{obs} = \mathbf{y}_1$ is a multivariate normal distribution with the mean vector

$$\boldsymbol{\mu}_{2.1} = \boldsymbol{\mu}_2 + \boldsymbol{\Sigma}'_{12}\boldsymbol{\Sigma}_{11}^{-1}(\mathbf{y}_1 - \boldsymbol{\mu}_1)$$

and the conditional covariance matrix

$$\boldsymbol{\Sigma}_{22.1} = \boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}'_{12}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\Sigma}_{12}$$

Bayesian Estimation of the Mean Vector and Covariance Matrix

Suppose that $\mathbf{Y} = (\mathbf{y}'_1, \mathbf{y}'_2, ..., \mathbf{y}'_n)'$ is an $(n \times p)$ matrix made up of n $(p \times 1)$ independent vectors $\mathbf{y}_i$, each of which has a multivariate normal distribution with mean zero and covariance matrix $\boldsymbol{\Lambda}$. Then the SSCP matrix

$$\mathbf{A} = \mathbf{Y}'\mathbf{Y} = \sum_i \mathbf{y}_i\mathbf{y}'_i$$

has a Wishart distribution $W(n, \boldsymbol{\Lambda})$.

When each observation $\mathbf{y}_i$ is distributed with a multivariate normal distribution with an unknown mean μ, then the CSSCP matrix

$$\mathbf{A} = \sum_i (\mathbf{y}_i - \overline{\mathbf{y}})(\mathbf{y}_i - \overline{\mathbf{y}})'$$

has a Wishart distribution $W(n-1, \boldsymbol{\Lambda})$.

If $\mathbf{A}$ has a Wishart distribution $W(n, \boldsymbol{\Lambda})$, then $\mathbf{B} = \mathbf{A}^{-1}$ has an inverted Wishart distribution $W^{-1}(n, \boldsymbol{\Psi})$, where n is the degrees of freedom and $\boldsymbol{\Psi} = \boldsymbol{\Lambda}^{-1}$ is the precision matrix (Anderson 1984).

Note that, instead of using the parameter $\boldsymbol{\Psi} = \boldsymbol{\Lambda}^{-1}$ for the inverted Wishart distribution, Schafer (1997) uses the parameter $\boldsymbol{\Lambda}$.

Suppose that each observation in the data matrix $\mathbf{Y}$ has a multivariate normal distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. Then with a prior inverted Wishart distribution for $\boldsymbol{\Sigma}$ and a prior normal distribution for $\boldsymbol{\mu}$

$$\begin{aligned} \boldsymbol{\Sigma} &\sim W^{-1}(m, \boldsymbol{\Psi}) \\ \boldsymbol{\mu}|\boldsymbol{\Sigma} &\sim N\left(\boldsymbol{\mu}_0, \frac{1}{\tau}\boldsymbol{\Sigma}\right) \end{aligned}$$

where $\tau > 0$ is a fixed number. The posterior distribution (Anderson 1984, p. 270; Schafer 1997, p. 152) is

$$\begin{aligned} \boldsymbol{\Sigma}|\mathbf{Y} &\sim W^{-1}\left(n+m, (n-1)\mathbf{S} + \boldsymbol{\Psi} + \frac{n\tau}{n+\tau}(\bar{\mathbf{y}} - \boldsymbol{\mu}_0)(\bar{\mathbf{y}} - \boldsymbol{\mu}_0)'\right) \\ \boldsymbol{\mu}|(\boldsymbol{\Sigma}, \mathbf{Y}) &\sim N\left(\frac{1}{n+\tau}(n\bar{\mathbf{y}} + \tau\boldsymbol{\mu}_0), \frac{1}{n+\tau}\boldsymbol{\Sigma}\right) \end{aligned}$$

where $(n-1)\mathbf{S}$ is the CSSCP matrix.

Posterior Step

In each iteration, the posterior step simulates the posterior population mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ from prior information for $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$, and the complete sample estimates.

You can specify the prior parameter information using one of the following methods:

- PRIOR=JEFFREYS, which uses a noninformative prior.
- PRIOR=INPUT=, which provides a prior information for $\boldsymbol{\Sigma}$ in the data set. Optionally, it also provides a prior information for μ in the data set.
- PRIOR=RIDGE=, which uses a ridge prior.

The next four subsections provide details of the posterior step for different prior distributions.

1. A Noninformative Prior

Without prior information about the mean and covariance estimates, a noninformative prior can be used by specifying the PRIOR=JEFFREYS option. The posterior distributions (Schafer 1997, p. 154) are

$$\begin{aligned} \boldsymbol{\Sigma}^{(t+1)}|\mathbf{Y} &\sim W^{-1}\left(n-1,\ (n-1)\mathbf{S}\right) \\ \boldsymbol{\mu}^{(t+1)}|(\boldsymbol{\Sigma}^{(t+1)},\mathbf{Y}) &\sim N\left(\bar{\mathbf{y}},\ \frac{1}{n}\boldsymbol{\Sigma}^{(t+1)}\right) \end{aligned}$$

2. An Informative Prior for μ and $\boldsymbol{\Sigma}$

When prior information is available for the parameters μ and $\boldsymbol{\Sigma}$, you can provide it with a SAS data set that you specify with the PRIOR=INPUT= option.

$$\begin{aligned} \boldsymbol{\Sigma} &\sim W^{-1}\left(d^*,\ d^*\mathbf{S}^*\right) \\ \boldsymbol{\mu}|\boldsymbol{\Sigma} &\sim N\left(\boldsymbol{\mu}_0,\ \frac{1}{n_0}\boldsymbol{\Sigma}\right) \end{aligned}$$

To obtain the prior distribution for $\boldsymbol{\Sigma}$, PROC MI reads the matrix $\mathbf{S}^*$ from observations in the data set with _TYPE_='COV', and it reads $n^* = d^* + 1$ from observations with _TYPE_='N'.

To obtain the prior distribution for μ, PROC MI reads the mean vector $\boldsymbol{\mu}_0$ from observations with _TYPE_='MEAN', and it reads n_0 from observations with _TYPE_='N_MEAN'. When there are no observations with _TYPE_='N_MEAN', PROC MI reads n_0 from observations with _TYPE_='N'.

The resulting posterior distribution, as described in the "Bayesian Estimation of the Mean Vector and Covariance Matrix" section on page 161, is given by

$$\begin{aligned} \boldsymbol{\Sigma}^{(t+1)}|\mathbf{Y} &\sim W^{-1}\left(n+d^*,\ (n-1)\mathbf{S}+d^*\mathbf{S}^*+\mathbf{S}_m\right) \\ \boldsymbol{\mu}^{(t+1)}\mid\left(\boldsymbol{\Sigma}^{(t+1)},\mathbf{Y}\right) &\sim N\left(\frac{1}{n+n_0}(n\bar{\mathbf{y}}+n_0\boldsymbol{\mu}_0),\ \frac{1}{n+n_0}\boldsymbol{\Sigma}^{(t+1)}\right) \end{aligned}$$

where

$$\mathbf{S}_m=\frac{nn_0}{n+n_0}(\bar{\mathbf{y}}-\boldsymbol{\mu}_0)(\bar{\mathbf{y}}-\boldsymbol{\mu}_0)'$$

3. An Informative Prior for $\boldsymbol{\Sigma}$

When the sample covariance matrix $\mathbf{S}$ is singular or near singular, prior information about $\boldsymbol{\Sigma}$ can also be used without prior information about μ to stabilize the inference about μ. You can provide it with a SAS data set that you specify with the PRIOR=INPUT= option.

To obtain the prior distribution for $\boldsymbol{\Sigma}$, PROC MI reads the matrix $\mathbf{S}^*$ from observations in the data set with _TYPE_='COV', and it reads n^* from observations with _TYPE_='N'.

Note that if the PRIOR=INPUT= data set also contains observations with _TYPE_='MEAN', then a complete informative prior for both μ and $\boldsymbol{\Sigma}$ will be used.

Corresponding to the prior for $\boldsymbol{\Sigma}$

$$\boldsymbol{\Sigma} \sim W^{-1}\left(d^*,\ d^*\mathbf{S}^*\right)$$

the posterior distribution for $\boldsymbol{\Sigma}$ (Anderson 1984, p. 269) is

$$\boldsymbol{\Sigma}^{(t+1)}|\mathbf{Y} \sim W^{-1}\left((n-1)+d^*,\ (n-1)\mathbf{S}+d^*\mathbf{S}^*\right)$$

Thus, an estimate of $\boldsymbol{\Sigma}$ is given by the weighted average

$$\frac{1}{(n-1)+d^*}\ \left((n-1)\mathbf{S}+d^*\mathbf{S}^*\right)$$

and the posterior distribution for $(\boldsymbol{\mu},\boldsymbol{\Sigma})$ becomes

$$\begin{aligned} \boldsymbol{\Sigma}^{(t+1)}|\mathbf{Y} &\sim W^{-1}\left((n-1)+d^*,\ (n-1)\mathbf{S}+d^*\mathbf{S}^*\right) \\ \boldsymbol{\mu}^{(t+1)}\mid\left(\boldsymbol{\Sigma}^{(t+1)},\mathbf{Y}\right) &\sim N\left(\bar{\mathbf{y}},\ \frac{1}{n}\boldsymbol{\Sigma}^{(t+1)}\right) \end{aligned}$$

4. A Ridge Prior

A special case of the preceding adjustment is a ridge prior with $\mathbf{S}^*$ = Diag $\mathbf{S}$ (Schafer 1997, p. 156). That is, $\mathbf{S}^*$ is a diagonal matrix with diagonal elements equal to the corresponding elements in $\mathbf{S}$.

You can request a ridge prior by using the PRIOR=RIDGE= option. You can explicitly specify the number $d^* \geq 1$ in the PRIOR=RIDGE=d^* option. Or you can implicitly specify the number by specifying the proportion p in the PRIOR=RIDGE=p option to request $d^* = (n-1)p$.

The posterior is then given by

$$\begin{aligned} \boldsymbol{\Sigma}^{(t+1)}|\mathbf{Y} &\sim W^{-1}\left((n-1)+d^*,\ (n-1)\mathbf{S}+d^*\mathbf{S}^*\right) \\ \boldsymbol{\mu}^{(t+1)}\,|\,\left(\boldsymbol{\Sigma}^{(t+1)},\mathbf{Y}\right) &\sim N\left(\bar{y},\ \frac{1}{n}\,\boldsymbol{\Sigma}^{(t+1)}\right) \end{aligned}$$

Producing Monotone Missingness with the MCMC Method

The monotone data MCMC method was first proposed by Li (1988), and Liu (1993) described the algorithm. The method is useful especially when a data set is close to having a monotone missing pattern. In this case, the method only needs to impute a few missing values to the data set to have a monotone missing pattern in the imputed data set. Compared to a full data imputation that imputes all missing values, the monotone data MCMC method imputes fewer missing values in each iteration and achieves approximate stationarity in fewer iterations (Schafer 1997, p. 227).

You can request the monotone MCMC method by specifying the option IMPUTE=MONOTONE in the MCMC statement. The "Missing Data Patterns" table now denotes the variables with missing values by "." or "O". A "." means that the variable is missing and will be imputed and an "O" means that the variable is missing and will not be imputed. The tables of "Multiple Imputation Variance Information" and "Multiple Imputation Parameter Estimates" are not created.

You must specify the variables in the VAR statement. The variable order in the list determines the monotone missing pattern in the imputed data set. With a different order in the VAR list, the results will be different because the monotone missing pattern to be constructed will be different.

Assuming that the data are from a multivariate normal distribution, then similar to the MCMC method, the monotone MCMC method repeats the following steps:

1. **The imputation I-step:**
Given an estimated mean vector and covariance matrix, the I-step simulates the missing values for each observation independently. Only a subset of missing values are simulated to achieve a monotone pattern of missingness.

2. **The posterior P-step:**
Given a new sample with a monotone pattern of missingness, the P-step simulates the posterior population mean vector and covariance matrix with a noninformative Jeffreys prior. These new estimates are then used in the next I-step.

Imputation Step

The I-step is almost identical to the I-step described in the "MCMC Method for Arbitrary Missing Data" section on page 159 except that here only a subset of missing values need to be simulated. To state this precisely, denote the variables with observed values for observation *i* by $Y_{i(obs)}$ and the variables with missing values by $Y_{i(mis)} = (Y_{i(m1)}Y_{i(m2)})$, where $Y_{i(m1)}$ is a subset of the the missing variables that will result a monotone missingness when their values are imputed. Then the I-step draws values for $Y_{i(m1)}$ from a conditional distribution for $Y_{i(m1)}$ given $Y_{i(obs)}$.

Posterior Step

The P-step is different from the P-step described in the "MCMC Method for Arbitrary Missing Data" section on page 159. Instead of simulating the $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ parameters from the full imputed data set, the P-step here simulates the $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ parameters through simulated regression coefficients from regression models based on the imputed data set with a monotone pattern of missingness. The step is similar to the process described in the "Regression Method for Monotone Missing Data" section on page 157.

That is, for the variable Y_j, a model

$$Y_j = \beta_0 + \beta_1 Y_1 + \beta_2 Y_2 + \ldots + \beta_{j-1} Y_{j-1}$$

is fitted using nonmissing observations.

The fitted model consists of the regression parameter estimates $\hat{\boldsymbol{\beta}} = (\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_{j-1})$ and the associated covariance matrix $\hat{\sigma}_j^2 \mathbf{V}_j$, where $\mathbf{V}_j$ is the usual $\mathbf{X}'\mathbf{X}$ inverse matrix from the intercept and variables $Y_1, Y_2, ..., Y_{j-1}$.

For each imputation, new parameters $\boldsymbol{\beta}_* = (\beta_{*0}, \beta_{*1}, ..., \beta_{*(j-1)})$ and σ_{*j}^2 are drawn from the posterior predictive distribution of the parameters. That is, they are simulated from $(\hat{\beta}_0, \hat{\beta}_1, ..., \hat{\beta}_{j-1})$, σ_j^2, and $\mathbf{V}_j$. The variance is drawn as

$$\sigma_{*j}^2 = \hat{\sigma}_j^2 (n_j - j)/g$$

where g is a $\chi^2_{n_j - p + j - 1}$ random variate and n_j is the number of nonmissing observations for Y_j. The regression coefficients are drawn as

$$\boldsymbol{\beta}_* = \hat{\boldsymbol{\beta}} + \sigma_{*j} \mathbf{V}'_{hj} \mathbf{Z}$$

where $\mathbf{V}'_{hj}$ is the upper triangular matrix in the Cholesky decomposition $\mathbf{V}_j = \mathbf{V}'_{hj}\mathbf{V}_{hj}$ and $\mathbf{Z}$ is a vector of j independent random normal variates.

These simulated values of $\boldsymbol{\beta}_*$ and σ_{*j}^2 are then used to re-create the parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$. For a detailed description of how to produce monotone-missingness with the MCMC method for a multivariate normal data, refer to Schafer (1997, pp. 226–235).

MCMC Method Specifications

With MCMC, you can impute either all missing values (IMPUTE=FULL) or just enough missing values to make the imputed data set have a monotone missing pattern (IMPUTE=MONOTONE). In the process, either a single chain for all imputations (CHAIN=SINGLE) or a separate chain for each imputation (CHAIN=MULTIPLE) is used. Refer to Schafer (1997, pp. 137–138) for a discussion of single versus multiple chains.

You can specify the number of initial burn-in iterations before the first imputation with the NBITER= option. This number is also used for subsequent chains for multiple chains. For a single chain, you can also specify the number of iterations between imputations with the NITER= option.

You can explicitly specify initial parameter values for the MCMC process with the INITIAL=INPUT= data set option. Alternatively, you can use the EM algorithm to derive a set of initial parameter values for MCMC with the option INITIAL=EM. These estimates are used as either the starting value (START=VALUE) or as the starting distribution (START=DIST) for the MCMC process. For multiple chains, these estimates are used again as either the starting value (START=VALUE) or as the starting distribution (START=DIST) for the subsequent chains.

You can specify the prior parameter information in the PRIOR= option. You can use a noninformative prior (PRIOR=JEFFREYS), a ridge prior (PRIOR=RIDGE), or an informative prior specified in a data set (PRIOR=INPUT).

The parameter estimates used to generate imputed values in each imputation can be saved in a data set with the OUTEST= option. Later, this data set can be read with the INEST= option to provide the reference distribution for imputing missing values for a new data set.

By default, the MCMC method uses a single chain to produce five imputations. It completes 200 burn-in iterations before the first imputation and 100 iterations between imputations. The posterior mode computed from the EM algorithm with a noninformative prior is used as the starting values for the MCMC process.

INITIAL=EM Specifications

The EM algorithm is used to find the maximum likelihood estimates for incomplete data in the EM statement. You can also use the EM algorithm to find a posterior mode, the parameter estimates that maximize the observed-data posterior density. The resulting posterior mode provides a good starting value for the MCMC process.

With INITIAL=EM, PROC MI uses the MLE of the parameter vector as the initial estimates in the EM algorithm for the posterior mode. You can use the ITPRINT option in INITIAL=EM to display the iteration history for the EM algorithm.

You can use the CONVERGE= option to specify the convergence criterion in deriving the EM posterior mode. The iterations are considered to have converged when the maximum change in the parameter estimates between iteration steps is less than the value specified. By default, CONVERGE=1E-4.

You can also use the MAXITER= option to specify the maximum number of iterations in the EM algorithm. By default, MAXITER=200.

With the BOOTSTRAP option, you can use overdispersed starting values for the MCMC process. In this case, PROC MI applies the EM algorithm to a bootstrap sample, a simple random sample with replacement from the input data set, to derive the initial estimates for each chain (Schafer 1997, p. 128).

Convergence in MCMC

The theoretical convergence of the MCMC process has been explored under various conditions, as described in Schafer (1997, p. 70). However, in practice, verification of convergence is not a simple matter and cannot be easily implemented in the MI procedure.

The parameters used in the imputation step for each iteration can be saved in an output data set with the OUTITER= option. These include the means, standard deviations, covariances, the worst linear function, and observed-data LR statistics. You can then monitor the convergence in a single chain by displaying time-series plots and autocorrelations for those parameter values (Schafer 1997, p. 120). The time-series and autocorrelation function plots for parameters such as variable means, covariances, and the worst linear function can be displayed by specifying the TIMEPLOT and ACFPLOT option.

You can apply EM to a bootstrap sample to obtain overdispersed starting values for multiple chains (Gelman and Rubin 1992). This provides a conservative estimate of the number of iterations needed before each imputation.

The next four subsections provide useful statistics and plots that can be used to check the convergence of the MCMC process.

LR Statistics

You can save the observed-data likelihood ratio (LR) statistic in each iteration with the LR option in the OUTITER= data set. The statistic is based on the observed-data likelihood with parameter values used in the iteration and the observed-data maximum likelihood derived from the EM algorithm.

In each iteration, the LR statistic is given by

$$-2\log\left(\frac{f(\hat{\boldsymbol{\theta}}_i)}{f(\hat{\boldsymbol{\theta}})}\right)$$

where $f(\hat{\boldsymbol{\theta}})$ is the observed-data maximum likelihood derived from the EM algorithm and $f(\hat{\boldsymbol{\theta}}_i)$ is the observed-data likelihood for $\hat{\boldsymbol{\theta}}_i$ used in the iteration.

Similarly, you can also save the observed-data LR posterior mode statistic for each iteration with the LR_POST option. This statistic is based on the observed-data posterior density with parameter values used in each iteration and the observed-data posterior mode derived from the EM algorithm for posterior mode.

For large samples, these LR statistics tends to be approximately χ^2 distributed with degrees of freedom equal to the dimension of $\boldsymbol{\theta}$ (Schafer 1997, p. 131). For example, with a large number of iterations, if the values of the LR statistic do not behave like a random sample from the described χ^2 distribution, then there is evidence that the MCMC process has not converged.

Worst Linear Function of Parameters

The worst linear function (WLF) of parameters (Schafer 1997, pp. 129-131) is a scalar function of parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ that is "worst" in the sense that its function values converge most slowly among parameters in the MCMC process. The convergence of this function is evidence that other parameters are likely to converge as well.

For linear functions of parameters $\boldsymbol{\theta} = (\boldsymbol{\mu}, \boldsymbol{\Sigma})$, a worst linear function of $\boldsymbol{\theta}$ has the highest asymptotic rate of missing information. The function can be derived from the iterative values of $\boldsymbol{\theta}$ near the posterior mode in the EM algorithm. That is, an estimated worst linear function of $\boldsymbol{\theta}$ is

$$w(\boldsymbol{\theta}) = \mathbf{v}' \left(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}\right)$$

where $\hat{\boldsymbol{\theta}}$ is the posterior mode and the coefficients $\mathbf{v} = \hat{\boldsymbol{\theta}}_{(-1)} - \hat{\boldsymbol{\theta}}$ is the difference between the estimated value of $\boldsymbol{\theta}$ one step prior to convergence and the converged value $\hat{\boldsymbol{\theta}}$.

You can display the coefficients of the worst linear function, $\mathbf{v}$, by specifying the WLF option in the MCMC statement. You can save the function value from each iteration in an OUTITER= data set by specifying the WLF option in the OUTITER option. You can also display the worst linear function values from iterations in an autocorrelation plot or a time-series plot by specifying WLF as an ACFPLOT or TIMEPLOT option, respectively.

Note that when the observed-data posterior is nearly normal, the WLF is one of the slowest functions to approach stationarity. When the posterior is not close to normal, other functions may take much longer than the WLF to converge, as described in Schafer (1997, p.130).

Time-Series Plot

A time-series plot for a parameter ξ is a scatter plot of successive parameter estimates ξ_i against the iteration number i. The plot provides a simple way to examine the convergence behavior of the estimation algorithm for ξ. Long-term trends in the plot indicate that successive iterations are highly correlated and that the series of iterations has not converged.

You can display time-series plots for the worst linear function, the variable means, variable variances, and covariances of variables. You can also request logarithmic transformations for positive parameters in the plots with the LOG option. When a parameter value is less than or equal to zero, the value is not displayed in the corresponding plot.

By default, the MI procedure uses the plus sign (+) as the plot symbol to display the points with a height of one (percentage screen unit) in a time-series plot. You can use the SYMBOL=, CSYMBOL=, and HSYMBOL= options to change the shape, color, and height of the plot symbol.

By default, the plot title "Time-Series Plot" is displayed in a time-series plot. You can request another title by using the TITLE= option in TIMEPLOT. When another title is also specified in a TITLE statement, this title is displayed as the main title and the plot title is displayed as a subtitle in the plot.

You can use options in the GOPTIONS statement to change the color and height of the title. Refer to the chapter "The SAS/GRAPH Statements" in *SAS/GRAPH Software: Reference, Version 8* for a description of title options. See Example 9.6 for a usage of the time-series plot.

Autocorrelation Function Plot

To examine relationships of successive parameter estimates ξ, the autocorrelation function (ACF) can be used. For a stationary series, $\xi_i, i \geq 1$, in time series data, the autocorrelation function at lag k is

$$\rho_k = \frac{\mathrm{Cov}(\xi_i, \xi_{i+k})}{\mathrm{Var}(\xi_i)}$$

The sample k_{th} order autocorrelation is computed as

$$r_k = \frac{\sum_{i=1}^{n-k}(\xi_i - \overline{\xi})(\xi_{i+k} - \overline{\xi})}{\sum_{i=1}^{n}(\xi_i - \overline{\xi})^2}$$

You can display autocorrelation function plots for the worst linear function, the variable means, variable variances, and covariances of variables. You can also request logarithmic transformations for parameters in the plots with the LOG option. When a parameter has values less than or equal to zero, the corresponding plot is not created.

You specify the maximum number of lags of the series with the NLAG= option. The autocorrelations at each lag less than or equal to the specified lag are displayed in the graph. In addition, the plot also displays approximate 95% confidence limits for the autocorrelations. At lag k, the confidence limits indicate a set of approximate 95% critical values for testing the hypothesis $\rho_j = 0, j \geq k$.

By default, the MI procedure uses the star sign (*) as the plot symbol to display the points with a height of one (percentage screen unit) in the plot, a solid line to display the reference line of zero autocorrelation, vertical line segments to connect autocorrelations to the reference line, and a pair of dashed lines to display approximately 95% confidence limits for the autocorrelations.

You can use the SYMBOL=, CSYMBOL=, and HSYMBOL= options to change the shape, color, and height of the plot symbol, and the CNEEDLES= and WNEEDLES= options to change the color and width of the needles. You can also use the LREF=, CREF=, and WREF= options to change the line type, color, and width of the reference line. Similarly, you can use the LCONF=, CCONF=, and WCONF= options to change the line type, color, and width of the confidence limits.

By default, the plot title "Autocorrelation Plot" is displayed in a autocorrelation function plot. You can request another title by using the TITLE= option in ACFPLOT. When another title is also specified in a TITLE statement, this title is displayed as the main title and the plot title is displayed as a subtitle in the plot.

You can use options in the GOPTIONS statement to change the color and height of the title. Refer to the chapter "The SAS/GRAPH Statements" in *SAS/GRAPH Software: Reference, Version 8* for a description of title options. See Example 9.6 for a usage of the autocorrelation function plot.

Input Data Sets

You can specify the input data set with missing values with the DATA= option in the PROC MI statement. When a MCMC method is used, you can specify the data set containing the reference distribution information for imputation with the INEST= option, the data set containing initial parameter estimates for the MCMC process with the INITIAL=INPUT= option, and the data set containing information for the prior distribution with the PRIOR=INPUT= option in the MCMC statement.

DATA=*SAS-data-set*

The input DATA= data set is an ordinary SAS data set containing multivariate data with missing values.

INEST=*SAS-data-set*

The input INEST= data set is a TYPE=EST data set and contains a variable _Imputation_ to identify the imputation number. For each imputation, PROC MI reads the point estimate from the observations with _TYPE_='PARM' or _TYPE_='PARMS' and the associated covariances from the observations with _TYPE_='COV' or _TYPE_='COVB'. These estimates are used as the reference distribution to impute values for observations in the DATA= data set. When the input INEST= data set also contains observations with _TYPE_='SEED', PROC MI reads the seed information for the random number generator from these observations. Otherwise, the SEED= option provides the seed information. See Example 9.8 for a usage of this option.

INITIAL=INPUT=*SAS-data-set*

The input INITIAL=INPUT= data set is a TYPE=COV or CORR data set and provides initial parameter estimates for the MCMC process. The covariances derived from the TYPE=COV/CORR data set are divided by the number of observations to get the correct covariance matrix for the point estimate (sample mean).

If TYPE=COV, PROC MI reads the number of observations from the observations with _TYPE_='N', the point estimate from the observations with _TYPE_='MEAN', and the covariances from the observations with _TYPE_='COV'.

If TYPE=CORR, PROC MI reads the number of observations from the observations with _TYPE_='N', the point estimate from the observations with _TYPE_='MEAN', the correlations from the observations with _TYPE_='CORR', and the standard deviations from the observations with _TYPE_='STD'.

PRIOR=INPUT=*SAS-data-set*

The input PRIOR=INPUT= data set is a TYPE=COV data set that provides information for the prior distribution. You can use the data set to specify a prior distribution for $\mathbf{\Sigma}$ of the form

$$\mathbf{\Sigma} \sim W^{-1}\left(d^*,\, d^*\mathbf{S}^*\right)$$

where $d^* = n^* - 1$ is the degrees of freedom. PROC MI reads the matrix $\mathbf{S}^*$ from observations with _TYPE_='COV' and n^* from observations with _TYPE_='N'.

You can also use this data set to specify a prior distribution for μ of the form

$$\boldsymbol{\mu} \sim N\left(\boldsymbol{\mu}_0,\, \frac{1}{n_0}\mathbf{\Sigma}\right)$$

PROC MI reads the mean vector $\boldsymbol{\mu}_0$ from observations with _TYPE_='MEAN' and n_0 from observations with _TYPE_='N_MEAN'. When there are no observations with _TYPE_='N_MEAN', PROC MI reads n_0 from observations with _TYPE_='N'.

Output Data Sets

You can specify the output data set of imputed values with the OUT= option in the PROC MI statement. When an EM statement is used, you can specify the data set containing MLE computed with the EM algorithm with the OUTEM= option in the EM statement. When a MCMC method is used, you can specify the data set containing parameter estimates used in each imputation with the OUTEST= option and the data set containing parameters used in the imputation step for each iteration with the OUTITER option in the MCMC statement.

OUT=*SAS-data-set*

The OUT= data set contains all the variables in the original data set and a new variable named _Imputation_ that identifies the imputation. For each imputation, the data set contains all variables in the input DATA= data set with missing values replaced by imputed values.

OUTEM=*SAS-data-set*

The OUTEM= data set is a TYPE=COV data set and contains the MLE computed with the EM algorithm. The observations with _TYPE_='MEAN' contain the estimated mean and the observations with _TYPE_='COV' contain the estimated covariances.

OUTEST=*SAS-data-set*

The OUTEST= data set is a TYPE=EST data set and contains parameter estimates used in each imputation in the MCMC method. It also includes an index variable named _Imputation_, which identifies the imputation.

The observations with _TYPE_='SEED' contain the seed information for the random number generator. The observations with _TYPE_='PARM' or _TYPE_='PARMS' contain the point estimate and the observations with _TYPE_='COV' or _TYPE_='COVB' contain the associated covariances. These estimates are used as the parameters of the reference distribution to impute values for observations in the DATA= dataset.

Note that these estimates are the values used in the I-step before each imputation. These are not the parameter values simulated from the P-step in the same iteration. See Example 9.8 for a usage of this option.

OUTITER < (options) > =*SAS-data-set* in an EM statement

The OUTITER= data set in an EM statement is a TYPE=COV data set and contains parameters for each iteration. It also includes a variable _Iteration_ that provides the iteration number.

The parameters in the output data set depend on the options specified. You can specify the MEAN and COV options for OUTITER. With the MEAN option, the output data set contains the mean parameters in observations with the variable _TYPE_='MEAN'. Similarly, with the MEAN option, the output data set contains the covariance parameters in observations with the variable _TYPE_='COV'. When no options are specified, the output data set contains the mean parameters for each iteration.

OUTITER < (options) > =*SAS-data-set* in a MCMC statement

The OUTITER= data set in a MCMC statement is a TYPE=COV data set and contains parameters used in the imputation step for each iteration. It also includes variables named _Imputation_ and _Iteration_, which provide the imputation number and iteration number.

The parameters in the output data set depend on the options specified. The following table summarizes the options available for OUTITER and the corresponding values for the output variable _TYPE_.

Table 9.3. Summary of Options for OUTITER in a MCMC statement

Options	Output Parameters	_TYPE_
MEAN	mean parameters	MEAN
STD	standard deviations	STD
COV	covariances	COV
LR	-2 log LR statistic	LOG_LR
LR_POST	-2 log LR statistic of the posterior mode	LOG_POST
WLF	worst linear function	WLF

When no options are specified, the output data set contains the mean parameters used in the imputation step for each iteration. For a detailed description of the worst linear function and LR statistics, see the "Convergence in MCMC" section on page 167.

Combining Inferences from Multiply Imputed Data Sets

With m imputations, m different sets of the point and variance estimates for a parameter Q can be computed. Suppose $\hat{Q}_i$ and $\hat{U}_i$ are the point and variance estimates from the ith imputed data set, i=1, 2, ..., m. Then the combined point estimate for Q from multiple imputation is the average of the m complete-data estimates:

$$\overline{Q} = \frac{1}{m}\sum_{i=1}^{m}\hat{Q}_i$$

Suppose $\overline{U}$ is the within-imputation variance, which is the average of the m complete-data estimates:

$$\overline{U} = \frac{1}{m}\sum_{i=1}^{m}\hat{U}_i$$

and B is the between-imputation variance

$$B = \frac{1}{m-1}\sum_{i=1}^{m}(\hat{Q}_i - \overline{Q})^2$$

Then the variance estimate associated with $\overline{Q}$ is the total variance (Rubin 1987)

$$T = \overline{U} + (1 + \frac{1}{m})B$$

The statistic $(Q - \overline{Q})T^{-(1/2)}$ is approximately distributed as t with v_m degrees of freedom (Rubin 1987), where

$$v_m = (m-1)[1 + \frac{\overline{U}}{(1+m^{-1})B}]^2$$

When the complete-data degrees of freedom v_0 is small, and there is only a modest proportion of missing data, the computed degrees of freedom, v_m, can be much larger than v_0, which is inappropriate. Barnard and Rubin (1999) recommend the use of an adjusted degrees of freedom

$$v_m^* = \left[\frac{1}{v_m} + \frac{1}{\hat{v}_{obs}}\right]^{-1}$$

where $\hat{v}_{obs} = (1-\gamma)\,v_0(v_0+1)/(v_0+3)$ and $\gamma = (1+m^{-1})B/T$.

Note that the MI procedure uses the adjusted degrees of freedom, v_m^*, for inference.

The degrees of freedom v_m depends on m and the ratio

$$r = \frac{(1+m^{-1})B}{\overline{U}}$$

The ratio r is called the relative increase in variance due to nonresponse (Rubin 1987). When there is no missing information about Q, the values of r and B are both zero. With a large value of m or a small value of r, the degrees of freedom v will be large and the distribution of $(Q - \overline{Q})T^{-(1/2)}$ will be approximately normal.

Another useful statistic is the fraction of missing information about Q:

$$\hat{\lambda} = \frac{r + 2/(v+3)}{r+1}$$

Both statistics r and λ are helpful diagnostics for assessing how the missing data contribute to the uncertainty about Q.

Multiple Imputation Efficiency

The relative efficiency (RE) of using the finite m imputation estimator, rather than using an infinite number for the fully efficient imputation, in units of variance, is approximately a function of m and λ (Rubin 1987, p. 114).

$$RE = (1 + \frac{\lambda}{m})^{-1}$$

The following table shows relative efficiencies with different values of m and λ. For cases with little missing information, only a small number of imputations are necessary.

Table 9.4. Relative Efficiency

	λ				
m	**10%**	**20%**	**30%**	**50%**	**70%**
3	0.9677	0.9375	0.9091	0.8571	0.8108
5	0.9804	0.9615	0.9434	0.9091	0.8772
10	0.9901	0.9804	0.9709	0.9524	0.9346
20	0.9950	0.9901	0.9852	0.9756	0.9662

Imputer's Model Versus Analyst's Model

Schafer (1997, pp. 139-143) provides comprehensive coverage of this topic, and the following discussion is largely based on his work.

Multiple imputation inference assumes that the model you used to analyze the multiply imputed data (the analyst's model) is the same as the model used to impute missing values in multiple imputation (the imputer's model). But in practice, the two models may not be the same.

For example, consider the same trivariate data set with variables Y_1 and Y_2 fully observed, and a variable Y_3 with missing values. An imputer creates multiple imputations with the model $Y_3 = Y_1\ Y_2$. However, the analyst can later use the simpler model $Y_3 = Y_1$. In this case, the analyst assumes more than the imputer. That is, the analyst assumes there is no relationship between variables Y_3 and Y_2.

The effect of the discrepancy between the models depends on whether the analyst's additional assumption is true. If the assumption is true, the imputer's model still applies. The inferences derived from multiple imputations will still be valid, although they may be somewhat conservative because they reflect the additional uncertainty of estimating the relationship between Y_3 and Y_2.

On the other hand, suppose that the analyst models $Y_3 = Y_1$, and there is a relationship between variables Y_3 and Y_2. Then the model $Y_3 = Y_1$ will be biased and is inappropriate. Appropriate results can be generated only from appropriate analyst's models.

Another type of discrepancy occurs when the imputer assumes more than the analyst. For example, suppose that an imputer creates multiple imputations with the model $Y_3 = Y_1$, but the analyst later fits a model $Y_3 = Y_1\ Y_2$. When the assumption is true, the imputer's model is a correct model and the inferences still hold.

On the other hand, suppose there is a relationship between Y_3 and Y_2. Imputations created under the incorrect assumption that there is no relationship between Y_3 and Y_2 will make the analyst's estimate of the relationship biased toward zero. Multiple imputations created under an incorrect model can lead to incorrect conclusions.

Thus, generally you should include as many variables as you can when doing multiple imputation. The precision you lose when you include unimportant predictors is usually a relatively small price to pay for the general validity of analyses of the resultant multiply imputed data set (Rubin 1996).

Note that it is good practice to include a description of the imputer's model with the multiply imputed data set. That way, the analysts will have information about the variables involved in the imputation and which relationships among the variables have been implicitly set to zero.

Parameter Simulation Versus Multiple Imputation

For many incomplete-data problems, simulation-based methods of parameter simulation and multiple imputation can be used to analyze the data. In parameter simulation, you simulate random values of parameters from the observed-data posterior distribution and make simple inferences about these parameters (Schafer 1997, p. 89).

When a set of well-defined population parameters $\boldsymbol{\theta}$ are of interest, parameter simulation can be used to directly examine and summarize simulated values of $\boldsymbol{\theta}$. This usually requires a large number of iterations, and involves calculating appropriate summaries of the resulting dependent sample of the iterates of the $\boldsymbol{\theta}$. If only a small set of parameters are involved, parameter simulation can be suitable (Schafer 1997).

In multiple imputation, the unknown missing data are replaced by multiple sets of simulated values. Each complete data set is then analyzed by standard complete-data methods. The variability among the results from these repeated analyses provides a measure of the uncertainty due to missing data. Combining this between-imputation variation with the ordinary within-imputation sample variation provides statistical inference for the parameters of interest. Multiple imputation is suitable for analyses that are more exploratory in nature.

Multiple imputation only requires a small number of imputations. Generating and storing a few imputations can be more efficient than generating and storing a large number of iterations for parameter simulation.

When fractions of missing information are low, methods that average over simulated values of the missing data, as in multiple imputation, can be much more efficient than methods that average over simulated values of θ as in parameter simulation (Schafer 1997).

ODS Table Names

PROC MI assigns a name to each table it creates. You must use these names to reference tables when using the Output Delivery System (ODS). These names are listed in the following table. For more information on ODS, refer to the chapter "Using the Output Delivery System" in the *SAS/STAT User's Guide, Version 8.*

Table 9.5. ODS Tables Produced in PROC MI

ODS Table Name	Description	Option
ModelInfo	Model information	
MissPattern	Missing data patterns	
Transform	Variable Transformations	TRANSFORM statement
Univariate	Univariate statistics for available cases	SIMPLE
Corr	Pairwise correlations for available cases	SIMPLE
EMInitEst	Initial parameter values for EM	EM statement
EMEst	MLE of the parameter vector from EM	EM statement
EMIter	EM iteration history for MLE	ITPRINT in EM statement
EMPIter	EM iteration history for posterior mode	ITPRINT in INITIAL=EM
EMPEst	Posterior mode parameter values from EM	INITIAL=EM
EMWlf	Coefficients of the worst linear function	WLF
MCMCInitEst	Initial parameter estimates for MCMC	DISPLAYINIT in MCMC
VarianceInfo	Between-imputation, within-imputation, and total variances	
ParmEst	Parameter estimates	

Examples

The following FitMono data set has a monotone missing data pattern and is used in Example 9.2 with the propensity score method and in Example 9.3 with the regression method. The FitMiss data set created in the "Getting Started" section is used in other examples. Note that the original data set has been altered for these examples.

```
*----------------- Data on Physical Fitness -----------------*
| These measurements were made on men involved in a physical |
| fitness course at N.C. State University.                   |
| Only selected variables of                                 |
| Oxygen (oxygen intake, ml per kg body weight per minute),  |
| Runtime (time to run 1.5 miles in minutes), and            |
| RunPulse (heart rate while running) are used.              |
| Certain values were changed to missing for the analysis.   |
*------------------------------------------------------------*;
   data FitMono;
      input Oxygen RunTime RunPulse @@;
      datalines;
   44.609  11.37  178     45.313  10.07  185
   54.297   8.65  156     59.571    .      .
   49.874   9.22    .     44.811  11.63  176
   45.681  11.95  176     49.091  10.85    .
   39.442  13.08  174     60.055   8.63  170
   50.541    .      .     37.388  14.03  186
   44.754  11.12  176     47.273    .      .
   51.855  10.33  166     49.156   8.95  180
   40.836  10.95  168     46.672  10.00    .
   46.774  10.25    .     50.388  10.08  168
   39.407  12.63  174     46.080  11.17  156
   45.441   9.63  164     54.625   8.92  146
   45.118  11.08    .     39.203  12.88  168
   45.790  10.47  186     50.545   9.93  148
   48.673   9.40  186     47.920  11.50  170
   47.467  10.50  170
   ;
```

Example 9.1. EM Algorithm for MLE

This example uses the EM algorithm to compute the maximum likelihood estimates for the parameters of a multivariate normal distribution using data with missing values. The following statements invoke the MI procedure and request the EM algorithm to compute the MLE for (μ, Σ) of a multivariate normal distribution from the input data set FitMiss.

```
proc mi data=FitMiss seed=55417 simple nimpute=0;
   em itprint outem=outem;
   var Oxygen RunTime RunPulse;
run;
```

Note when you specify the option NIMPUTE=0, the missing values will not be imputed. The procedure generates the following output:

Output 9.1.1. Model Information

The MI Procedure

Model Information

Data Set	WORK.FITMISS
Method	MCMC
Multiple Imputation Chain	Single Chain
Initial Estimates for MCMC	EM Posterior Mode
Start	Starting Value
Prior	Jeffreys
Number of Imputations	0
Number of Burn-in Iterations	200
Number of Iterations	100
Seed for random number generator	55417

The "Model Information" table describes the method and options used in the procedure.

Output 9.1.2. Missing Data Patterns

The MI Procedure

Missing Data Patterns

Group	Oxygen	Run Time	Run Pulse	Freq	Percent
1	X	X	X	21	67.74
2	X	X	.	4	12.90
3	X	.	.	3	9.68
4	.	X	X	1	3.23
5	.	X	.	2	6.45

Missing Data Patterns

	Group Means		
Group	Oxygen	RunTime	RunPulse
1	46.353810	10.809524	171.666667
2	47.109500	10.137500	.
3	52.461667	.	.
4	.	11.950000	176.000000
5	.	9.885000	.

The "Missing Data Patterns" table lists distinct missing data patterns with corresponding frequencies and percents. Here, "X" means that the variable is observed in the corresponding group and "." means that the variable is missing. The table also displays group-specific variable means.

With the SIMPLE option, the procedure displays simple descriptive univariate statistics for available cases in the "Univariate Statistics" table and correlations from pairwise available cases in the "Pairwise Correlations" table.

Output 9.1.3. Univariate Statistics

The MI Procedure

Univariate Statistics

Variable	N	Mean	Std Dev	Minimum	Maximum
Oxygen	28	47.11618	5.41305	37.38800	60.05500
RunTime	28	10.68821	1.37988	8.63000	14.03000
RunPulse	22	171.86364	10.14324	148.00000	186.00000

Output 9.1.4. Pairwise Correlations

The MI Procedure

Pairwise Correlations

	Oxygen	RunTime	RunPulse
Oxygen	1.000000000	-0.849118562	-0.343961742
RunTime	-0.849118562	1.000000000	0.247258191
RunPulse	-0.343961742	0.247258191	1.000000000

With the EM statement, the procedure displays the initial parameter estimates for EM.

Output 9.1.5. Initial Parameter Estimates for EM

The MI Procedure

Initial Parameter Estimates for EM

TYPE	_NAME_	Oxygen	RunTime	RunPulse
MEAN		47.116179	10.688214	171.863636
COV	Oxygen	29.301078	0	0
COV	RunTime	0	1.904067	0
COV	RunPulse	0	0	102.885281

With the ITPRINT option, the "EM (MLE) Iteration History" table displays the iteration history for the EM algorithm.

Output 9.1.6. EM (MLE) Iteration History

The MI Procedure

EM (MLE) Iteration History

Iteration	-2 Log L	Oxygen	RunTime	RunPulse
0	289.544782	47.116179	10.688214	171.863636
1	263.549489	47.116179	10.688214	171.863636
2	255.851312	47.139089	10.603506	171.538203
3	254.616428	47.122353	10.571685	171.426790
4	254.494971	47.111080	10.560585	171.398296
5	254.483973	47.106523	10.556768	171.389208
6	254.482920	47.104899	10.555485	171.385257
7	254.482813	47.104348	10.555062	171.383345
8	254.482801	47.104165	10.554923	171.382424
9	254.482800	47.104105	10.554878	171.381992
10	254.482800	47.104086	10.554864	171.381796

The procedure then displays the EM (MLE) parameter estimates, the maximum likelihood estimates for μ and Σ of a multivariate normal distribution from the data set FitMiss.

Output 9.1.7. EM (MLE) Parameter Estimates

The MI Procedure

EM (MLE) Parameter Estimates

TYPE	_NAME_	Oxygen	RunTime	RunPulse
MEAN		47.104086	10.554864	171.381796
COV	Oxygen	27.798014	-6.457929	-18.030790
COV	RunTime	-6.457929	2.015491	3.516092
COV	RunPulse	-18.030790	3.516092	97.766559

You can also output the EM (MLE) parameter estimates into an output data set with the OUTEM= option. The following statements list the observations in the output data set outem.

```
proc print data=outem;
   title 'EM Estimates';
run;
```

Output 9.1.8. EM Estimates

EM Estimates

Obs	_TYPE_	_NAME_	Oxygen	RunTime	RunPulse
1	MEAN		47.1041	10.5549	171.382
2	COV	Oxygen	27.7980	-6.4579	-18.031
3	COV	RunTime	-6.4579	2.0155	3.516
4	COV	RunPulse	-18.0308	3.5161	97.767

The output data set outem is a TYPE=COV data set. The observation with _TYPE_='MEAN' contains the MLE for the parameter μ and the observations with _TYPE_='COV' contain the MLE for the parameter Σ of a multivariate normal distribution from the data set FitMiss.

Example 9.2. Propensity Score Method

This example uses the propensity score method to impute missing values in a data set with a monotone missing pattern. The following statements invoke the MI procedure and request the propensity score method. The resulting data set is named outpscore.

```
proc mi data=FitMono seed=55417 simple out=outpscore;
   monotone method=propensity;
   var Oxygen RunTime RunPulse;
run;
```

Note that the VAR statement is required and the data set must have a monotone missing pattern with variables as ordered in the VAR statement. The procedure generates the following output:

Output 9.2.1. Model Information

The MI Procedure

Model Information

Data Set	WORK.FITMONO
Method	Propensity
Number of Imputations	5
Number of Groups on Propensity	5
Seed for random number generator	55417

The "Model Information" table describes the method and options used in the multiple imputation process. By default, the observations are sorted into five groups based on the propensity scores, and five imputations are created for the missing data.

Output 9.2.2. Missing Data Patterns

The MI Procedure

Missing Data Patterns

Group	Oxygen	Run Time	Run Pulse	Freq	Percent
1	X	X	X	23	74.19
2	X	X	.	5	16.13
3	X	.	.	3	9.68

Missing Data Patterns

	Group Means		
Group	Oxygen	RunTime	RunPulse
1	46.684174	10.776957	170.739130
2	47.505800	10.280000	.
3	52.461667	.	.

The "Missing Data Patterns" table lists distinct missing data patterns with corresponding frequencies and percents. Here, "X" means that the variable is observed in the corresponding group and "." means that the variable is missing. The table also displays group-specific variable means.

Output 9.2.3. Variance Information

The MI Procedure

Multiple Imputation Variance Information

	Variance			
Variable	Between	Within	Total	DF
RunTime	0.001068	0.059100	0.060382	27.498
RunPulse	1.147555	4.686646	6.063711	17.006

Multiple Imputation Variance Information

Variable	Relative Increase in Variance	Fraction Missing Information
RunTime	0.021688	0.021448
RunPulse	0.293828	0.246288

After the completion of *m* imputations, the "Multiple Imputation Variance Information" table displays the between-imputation variance, within-imputation variance, and total variance for combining complete-data inferences. It also displays the degrees of freedom for the total variance. The relative increase in variance due to miss-

ingness and the fraction of missing information for each variable are also displayed. A detailed description of these statistics is provided in the "Combining Inferences from Multiply Imputed Data Sets" section on page 173.

The "Multiple Imputation Parameter Estimates" table displays the estimated mean and standard error of the mean for each variable. The inferences are based on the *t*-distributions. For each variable, the table also displays a 95% mean confidence interval and a *t*-statistic with the associated *p*-value for the hypothesis that the population mean is equal to the value specified in the MU0= option, which is zero by default.

Output 9.2.4. Parameter Estimates

The MI Procedure

Multiple Imputation Parameter Estimates

Variable	Mean	Std Error	95% Confidence Limits		DF
RunTime	10.603677	0.245727	10.0999	11.1074	27.498
RunPulse	170.400000	2.462460	165.2048	175.5952	17.006

Multiple Imputation Parameter Estimates

Variable	Minimum	Maximum	Mu0	t for H0: Mean=Mu0	Pr > |t|
RunTime	10.558065	10.648387	0	43.15	<.0001
RunPulse	168.967742	171.838710	0	69.20	<.0001

The following statements list the first ten observations of the data set outpscore.

```
proc print data=outpscore(obs=10);
   title 'First 10 Observations of the Imputed Data Set';
run;
```

Output 9.2.5. Imputed Data Set

First 10 Observations of the Imputed Data Set

Obs	_Imputation_	Oxygen	Run Time	Run Pulse
1	1	44.609	11.37	178
2	1	45.313	10.07	185
3	1	54.297	8.65	156
4	1	59.571	8.63	146
5	1	49.874	9.22	156
6	1	44.811	11.63	176
7	1	45.681	11.95	176
8	1	49.091	10.85	156
9	1	39.442	13.08	174
10	1	60.055	8.63	170

Example 9.3. Regression Method

This example uses the regression method to impute missing values in a data set with a monotone missing pattern. The following statements invoke the MI procedure and request the regression method. The resulting data set is named outreg.

```
proc mi data=FitMono round=.001 .01 1  mu0= 50 10 150
        seed=55417 out=outreg;
   monotone method=reg;
   var Oxygen RunTime RunPulse;
run;
```

The ROUND= option is used to round the imputed values to the same precision as observed values. The values specified with the ROUND= option are matched with the variables Oxygen, RunTime, and RunPulse in the order listed with the VAR statement. The MU0= option requests *t* tests for the hypotheses that the population means corresponding to the variables in the VAR statement are Oxygen=50, RunTime=10, and RunPulse=150.

The "Missing Data Patterns" table lists distinct missing data patterns with corresponding frequencies and percents. It is identical to the table in the previous example.

After the completion of five imputations by default, the "Multiple Imputation Variance Information" table displays the between-imputation variance, within-imputation variance, and total variance for combining complete-data inferences. The relative increase in variance due to missingness and the fraction of missing information for each variable are also displayed. These statistics are described in the "Combining Inferences from Multiply Imputed Data Sets" section on page 173.

Output 9.3.1. Variance Information

The MI Procedure

Multiple Imputation Variance Information

Variable	Variance: Between	Within	Total	DF
RunTime	0.004443	0.068684	0.074016	25.294
RunPulse	1.790531	4.045134	6.193770	11.846

Multiple Imputation Variance Information

Variable	Relative Increase in Variance	Fraction Missing Information
RunTime	0.077629	0.074435
RunPulse	0.531166	0.382947

The "Multiple Imputation Parameter Estimates" table displays a 95% mean confidence interval and a *t*-statistic with its associated *p*-value for each of the hypotheses requested with the MU0= option.

Output 9.3.2. Parameter Estimates

The MI Procedure

Multiple Imputation Parameter Estimates

Variable	Mean	Std Error	95% Confidence Limits		DF
RunTime	10.575871	0.272059	10.0159	11.1359	25.294
RunPulse	170.425806	2.488729	164.9955	175.8561	11.846

Multiple Imputation Parameter Estimates

Variable	Minimum	Maximum	Mu0	t for H0: Mean=Mu0	Pr > |t|
RunTime	10.506452	10.680968	10.000000	2.12	0.0443
RunPulse	169.290323	171.935484	150.000000	8.21	<.0001

The following statements list the first ten observations of the data set outreg. Note that the imputed values rounded to the same precision as the observed values.

```
proc print data=outreg(obs=10);
   title 'First 10 Observations of the Imputed Data Set';
run;
```

Output 9.3.3. Imputed Data Set

First 10 Observations of the Imputed Data Set

Obs	_Imputation_	Oxygen	Run Time	Run Pulse
1	1	44.609	11.37	178
2	1	45.313	10.07	185
3	1	54.297	8.65	156
4	1	59.571	7.18	156
5	1	49.874	9.22	192
6	1	44.811	11.63	176
7	1	45.681	11.95	176
8	1	49.091	10.85	174
9	1	39.442	13.08	174
10	1	60.055	8.63	170

Example 9.4. MCMC Method

This example uses the MCMC method to impute missing values for a data set with an arbitrary missing pattern. The following statements invoke the MI procedure and specify the MCMC method with three imputations.

```
proc mi data=FitMiss seed=55417 nimpute=3 mu0=50 10 180;
   mcmc chain=multiple displayinit initial=em(itprint);
   var Oxygen RunTime RunPulse;
run;
```

Output 9.4.1. Model Information

The MI Procedure

Model Information

Data Set	WORK.FITMISS
Method	MCMC
Multiple Imputation Chain	Multiple Chains
Initial Estimates for MCMC	EM Posterior Mode
Start	Starting Value
Prior	Jeffreys
Number of Imputations	3
Number of Burn-in Iterations	200
Seed for random number generator	55417

With CHAIN=MULTIPLE, the procedure uses multiple chains and completes the default 200 burn-in iterations before each imputation. The 200 burn-in iterations are used to make the iterations converge to the stationary distribution before the imputation.

By default, the procedure uses a noninformative Jeffreys prior to derive the posterior mode from the EM algorithm as the starting values for the MCMC process.

The following "Missing Data Patterns" table lists distinct missing data patterns with corresponding statistics.

Output 9.4.2. Missing Data Patterns

The MI Procedure

Missing Data Patterns

Group	Oxygen	Run Time	Run Pulse	Freq	Percent
1	X	X	X	21	67.74
2	X	X	.	4	12.90
3	X	.	.	3	9.68
4	.	X	X	1	3.23
5	.	X	.	2	6.45

Missing Data Patterns

	Group Means		
Group	Oxygen	RunTime	RunPulse
1	46.353810	10.809524	171.666667
2	47.109500	10.137500	.
3	52.461667	.	.
4	.	11.950000	176.000000
5	.	9.885000	.

With the ITPRINT option in INITIAL=EM, the procedure also displays the "EM (Posterior Mode) Iteration History" table.

Output 9.4.3. EM (Posterior Mode) Iteration History

The MI Procedure

EM (Posterior Mode) Iteration History

Iteration	-2 Log L	-2 Log Posterior	Oxygen	RunTime
0	254.482800	282.909590	47.104086	10.554864
1	255.081159	282.051588	47.104079	10.554859
2	255.271405	282.017488	47.104077	10.554858
3	255.318621	282.015372	47.104002	10.554524
4	255.330259	282.015232	47.103861	10.554388
5	255.333160	282.015222	47.103797	10.554341
6	255.333896	282.015222	47.103774	10.554325
7	255.334085	282.015222	47.103766	10.554320

EM (Posterior Mode) Iteration History

Iteration	RunPulse
0	171.381796
1	171.381708
2	171.381669
3	171.381853
4	171.382058
5	171.382152
6	171.382186
7	171.382197

With the DISPLAYINIT option in the MCMC statement, the following "Initial Parameter Estimates for MCMC" table displays the starting mean and covariance estimates used in MCMC. The same starting estimates are used for the MCMC process for multiple chains because the EM algorithm is applied to the same data set in each chain. You can explicitly specify different initial estimates for different imputations, or you can use the bootstrap to generate different parameter estimates from the EM algorithm for the MCMC process.

Output 9.4.4. Initial Parameter Estimates

The MI Procedure

Initial Parameter Estimates for MCMC

TYPE	_NAME_	Oxygen	RunTime	RunPulse
MEAN		47.103766	10.554320	171.382197
COV	Oxygen	24.549968	-5.726112	-15.926034
COV	RunTime	-5.726112	1.781407	3.124798
COV	RunPulse	-15.926034	3.124798	83.164044

The following two tables display variance information and parameter estimates from the multiple imputation.

Output 9.4.5. Variance Information

The MI Procedure

Multiple Imputation Variance Information

Variable	Variance: Between	Within	Total	DF
Oxygen	0.009200	0.987880	1.000148	27.778
RunTime	0.002255	0.069112	0.072119	26.388
RunPulse	0.043126	3.650388	3.707889	27.653

Multiple Imputation Variance Information

Variable	Relative Increase in Variance	Fraction Missing Information
Oxygen	0.012418	0.012414
RunTime	0.043503	0.043351
RunPulse	0.015752	0.015744

Output 9.4.6. Parameter Estimates

The MI Procedure

Multiple Imputation Parameter Estimates

Variable	Mean	Std Error	95% Confidence Limits		DF
Oxygen	47.198228	1.000074	45.1489	49.2475	27.778
RunTime	10.510911	0.268549	9.9593	11.0625	26.388
RunPulse	172.113649	1.925588	168.1670	176.0603	27.653

Multiple Imputation Parameter Estimates

Variable	Minimum	Maximum	Mu0	t for H0: Mean=Mu0	Pr > |t|
Oxygen	47.132351	47.308274	50.000000	-2.80	0.0092
RunTime	10.456079	10.538446	10.000000	1.90	0.0681
RunPulse	171.943144	172.344920	180.000000	-4.10	0.0003

Example 9.5. Producing Monotone Missingness with MCMC

This example uses the MCMC method to impute just enough missing values for a data set with an arbitrary missing pattern so that each imputed data set has a monotone missing pattern based on the order of variables in the VAR statement.

The following statements invoke the MI procedure and specify the the IMPUTE=MONOTONE option to create the imputed data set with a monotone missing pattern. You must specify a VAR list to provide the order of variables for the imputed data to achieve a monotone missing pattern.

```
proc mi data=FitMiss seed=55417 out=outmono;
   mcmc impute=monotone;
   var Oxygen RunTime RunPulse;
run;
```

Output 9.5.1. Model Information

The MI Procedure

Model Information

Data Set	WORK.FITMISS
Method	Monotone-data MCMC
Multiple Imputation Chain	Single Chain
Initial Estimates for MCMC	EM Posterior Mode
Start	Starting Value
Prior	Jeffreys
Number of Imputations	5
Number of Burn-in Iterations	200
Number of Iterations	100
Seed for random number generator	55417

The following "Missing Data Patterns" table lists distinct missing data patterns with corresponding statistics. Here, an "X" means that the variable is observed in the corresponding group, a "." means that the variable is missing and will be imputed to achieve the monotone missingness for the imputed data set, and an "O" means that the variable is missing and will not be imputed. The table also displays group-specific variable means.

Output 9.5.2. Missing Data Pattern

The MI Procedure

Missing Data Patterns

Group	Oxygen	Run Time	Run Pulse	Freq	Percent
1	X	X	X	21	67.74
2	X	X	O	4	12.90
3	X	O	O	3	9.68
4	.	X	X	1	3.23
5	.	X	O	2	6.45

Missing Data Patterns

	Group Means		
Group	Oxygen	RunTime	RunPulse
1	46.353810	10.809524	171.666667
2	47.109500	10.137500	.
3	52.461667	.	.
4	.	11.950000	176.000000
5	.	9.885000	.

As shown in the table, the MI procedure only needs to impute three missing values from Group 4 and Group 5 to achieve a monotone missing pattern for the imputed data set.

When using the MCMC method to produce an imputed data set with a monotone missing pattern, tables of variance information and parameter estimates are not created.

The following statements are used just to show the monotone missingness of the output data set outmono.

```
proc mi data=outmono ( where= (_Imputation_=1) )
   nimpute=0;
   var Oxygen RunTime RunPulse;
run;
```

Output 9.5.3. Monotone Missing Data Pattern

The MI Procedure

Missing Data Patterns

Group	Oxygen	Run Time	Run Pulse	Freq	Percent
1	X	X	X	22	70.97
2	X	X	.	6	19.35
3	X	.	.	3	9.68

Missing Data Patterns

Group	Group Means		
	Oxygen	RunTime	RunPulse
1	46.307744	10.861364	171.863636
2	46.372151	10.053333	.
3	52.461667	.	.

The following statements impute one value for each missing value in the monotone missingness data set outmono. The variable _Imputation_ is renamed to Impute so that it will not be overwritten by the the new variable _Imputation_ being created in the MI procedure.

```
proc mi data=outmono( rename=(_Imputation_=Impute))
        nimpute=1 seed=43672
        out=outds( rename=(Impute=_Imputation_) drop=_Imputation_);
   monotone method=reg;
   var Oxygen RunTime RunPulse;
   by Impute;
run;
```

The variable Impute is renamed to _Imputation_ in the output data outds. This makes the output data set have the same structure as output data sets generated from other imputation methods. You can then analyze these data sets by using other SAS procedures and combine these results by using the procedure MIANALYZE. Note that the VAR statement is required with a MONOTONE statement to provide the variable order for the monotone missing pattern.

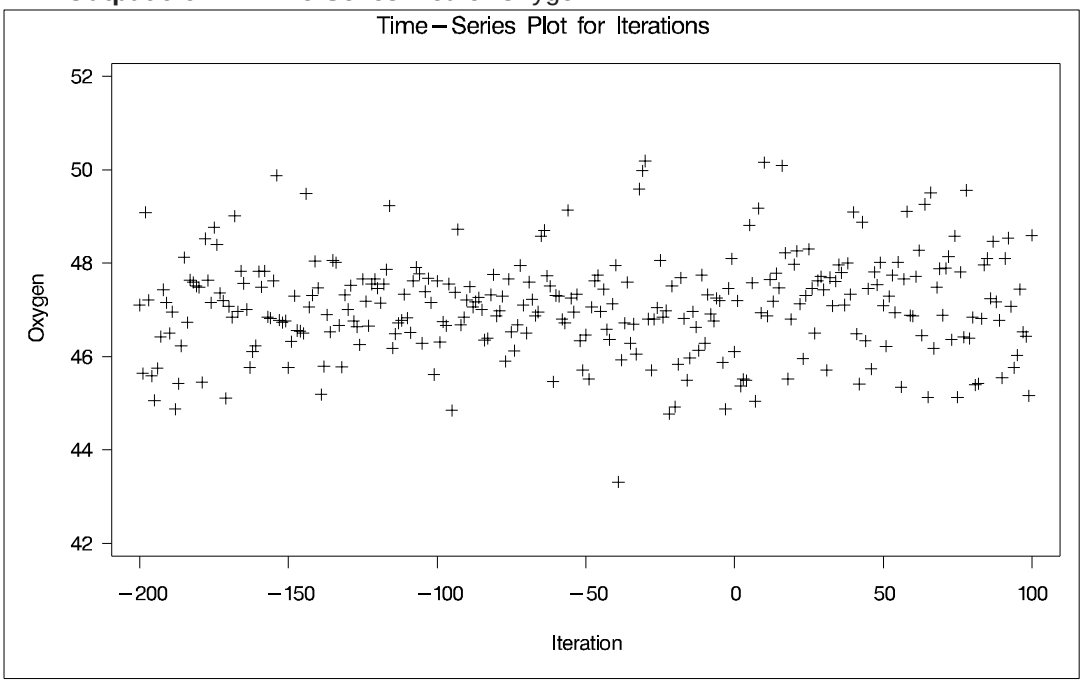
Example 9.6. Checking Convergence in MCMC

This example uses the MCMC method with a single chain. It also displays time-series and autocorrelation plots to check convergence for the single chain.

The following statements use the MCMC method to create an iteration plot for the successive estimates of the mean of Oxygen. Note that iterations during the burn-in period are indicated with negative iteration numbers. These statements also create an autocorrelation function plot for the variable Oxygen.

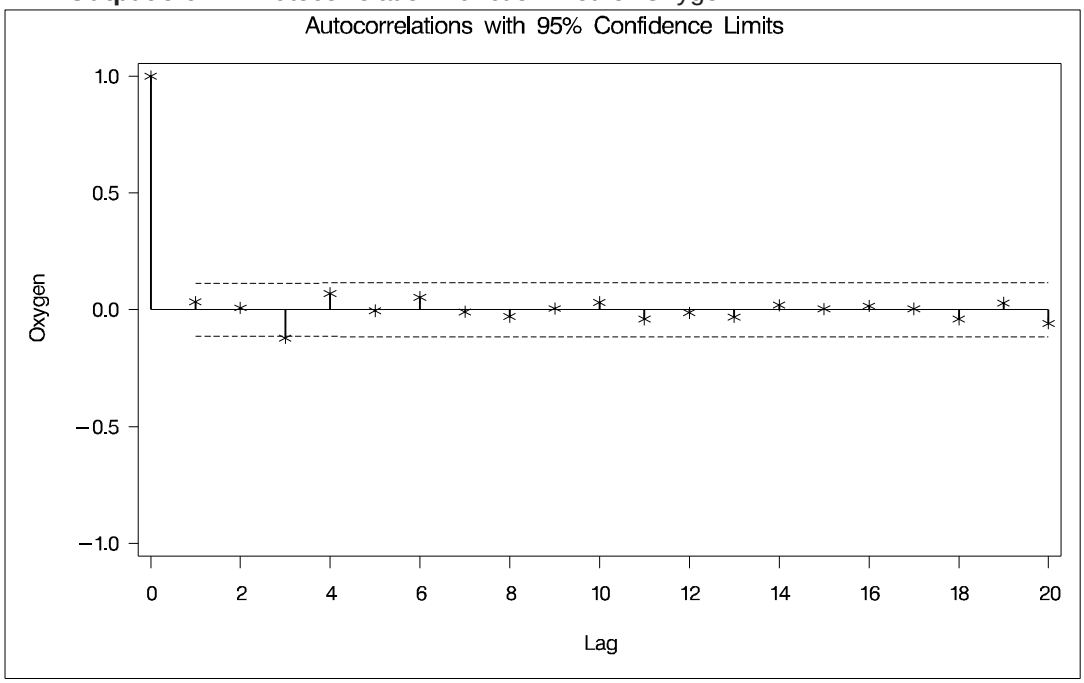
```
proc mi data=FitMiss seed=37921 noprint nimpute=2;
   mcmc timeplot(mean(Oxygen)) acfplot(mean(Oxygen));
   var Oxygen RunTime RunPulse;
run;
```

Output 9.6.1. Time-Series Plot for Oxygen



By default, the MI procedure uses the plus sign (+) as the plot symbol to display the points in the plot. The time-series plot shows no apparent trends for the variable Oxygen.

Output 9.6.2. Autocorrelation Function Plot for Oxygen

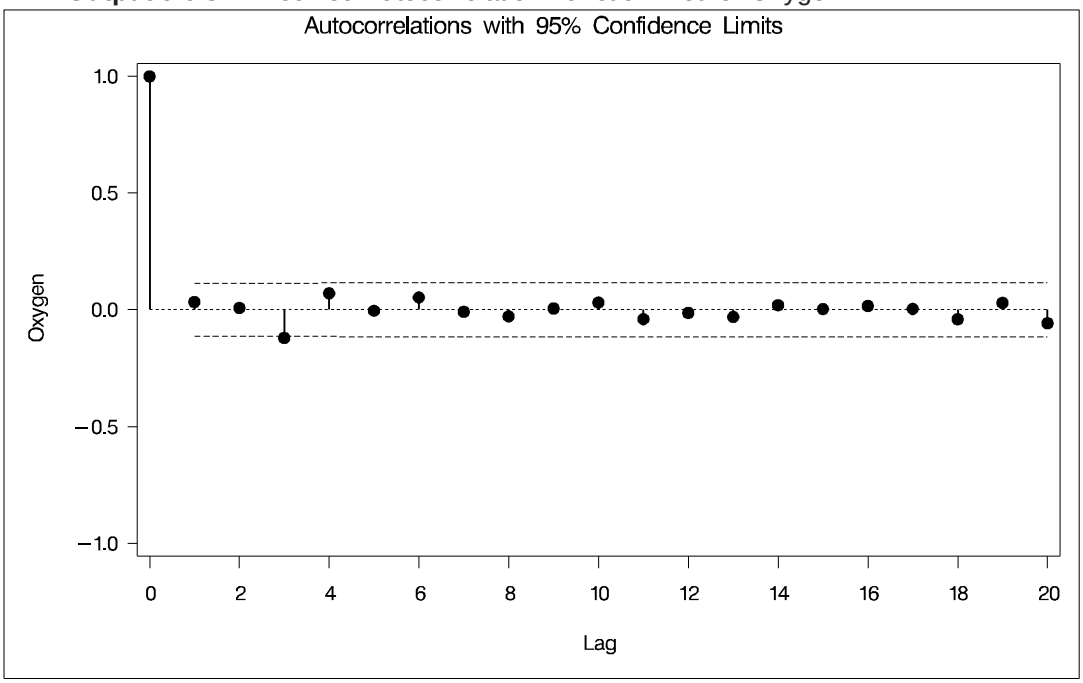


By default, the MI procedure uses the star sign (*) as the plot symbol to display the points in the plot, a solid line to display the reference line of zero autocorrelation, and a pair of dashed lines to display approximately 95% confidence limits for the autocorrelations. The autocorrelation function plot shows no significant positive or negative autocorrelation.

The following statements use display options to modify the autocorrelation function plot for Oxygen.

```
proc mi data=FitMiss seed=37921 noprint nimpute=2;
   mcmc acfplot(mean(Oxygen) / symbol=dot lref=2);
   var Oxygen RunTime RunPulse;
run;
```

Output 9.6.3. Modified Autocorrelation Function Plot for Oxygen



You can also create plots for the worst linear function, the means of other variables, the variances of variables, and covariances between variables. Alternatively, you can use the OUTITER option to save statistics such as the means, standard deviations, covariances, -2 log LR statistic, -2 log LR statistic of the posterior mode, and worst linear function from each iteration in an output data set. Then you can do a more in-depth time-series analysis of the iterations with other procedures, such as PROC AUTOREG and PROC ARIMA in the *SAS/ETS User's Guide, Version 8*.

Example 9.7. Transformation to Normality

This example applies the MCMC method to the FitMiss data set in which the variable Oxygen is transformed. Assume that Oxygen is skewed and can be transformed to normality with a logarithmic transformation. The following statements invoke the MI procedure and specify the transformation. The TRANSFORM statement specifies the log transformation for Oxygen. Note that the values displayed for Oxygen in all of the results correspond to transformed values.

```
proc mi data=FitMiss seed=37921 mu0=50 10 180 out=outmi;
   transform log(Oxygen);
   mcmc chain=multiple displayinit;
   var Oxygen RunTime RunPulse;
run;
```

The following "Missing Data Patterns" table lists distinct missing data patterns with corresponding statistics for the FitMiss data. Note that the values of Oxygen shown in the tables are transformed values.

Output 9.7.1. Missing Data Pattern

The MI Procedure

Missing Data Patterns

Group	Oxygen	Run Time	Run Pulse	Freq	Percent
1	X	X	X	21	67.74
2	X	X	.	4	12.90
3	X	.	.	3	9.68
4	.	X	X	1	3.23
5	.	X	.	2	6.45

Transformed Variables: Oxygen

Missing Data Patterns

	Group Means		
Group	Oxygen	RunTime	RunPulse
1	3.829760	10.809524	171.666667
2	3.851813	10.137500	.
3	3.955298	.	.
4	.	11.950000	176.000000
5	.	9.885000	.

Transformed Variables: Oxygen

The following "Variable Transformations" table lists the variables that have been transformed.

Output 9.7.2. Missing Data Pattern

The MI Procedure

Variable Transformations

Variable	_Transform_
Oxygen	LOG

The following "Initial Parameter Estimates for MCMC" table displays the starting mean and covariance estimates used in the MCMC process.

Output 9.7.3. Initial Parameter Estimates

The MI Procedure

Initial Parameter Estimates for MCMC

TYPE	_NAME_	Oxygen	RunTime	RunPulse
MEAN		3.846122	10.557605	171.382949
COV	Oxygen	0.010827	-0.120891	-0.328772
COV	RunTime	-0.120891	1.744580	3.011179
COV	RunPulse	-0.328772	3.011179	82.747608

Transformed Variables: Oxygen

The following table displays variance information from the multiple imputation.

Output 9.7.4. Variance Information

```
                          The MI Procedure

                Multiple Imputation Variance Information

                 -----------------Variance-----------------
   Variable          Between          Within           Total       DF

 * Oxygen       0.000004541        0.000398        0.000404   27.766
   RunTime         0.000814        0.063128        0.064105   27.708
   RunPulse        0.182700        3.498974        3.718214   25.923

                       * Transformed Variables

                Multiple Imputation Variance Information

                                   Relative       Fraction
                                   Increase        Missing
                    Variable     in Variance    Information

                  * Oxygen          0.013685       0.013590
                    RunTime         0.015478       0.015356
                    RunPulse        0.062658       0.060595

                       * Transformed Variables
```

The following table displays parameter estimates from the multiple imputation. Note that the parameter value of Mu0 has also been transformed using the logarithmic transformation.

Output 9.7.5. Parameter Estimates

```
                          The MI Procedure

                  Multiple Imputation Parameter Estimates

    Variable            Mean      Std Error    95% Confidence Limits        DF

  * Oxygen          3.845991       0.020091       3.8048       3.8872   27.766
    RunTime        10.586242       0.253190      10.0674      11.1051   27.708
    RunPulse      170.849654       1.928267     166.8855     174.8138   25.923

                         * Transformed Variables

                  Multiple Imputation Parameter Estimates

                                                          t for H0:
  Variable         Minimum         Maximum            Mu0    Mean=Mu0   Pr > |t|

* Oxygen          3.843860        3.848775       3.912023       -3.29     0.0028
  RunTime        10.547440       10.616746      10.000000        2.32     0.0282
  RunPulse      170.315955      171.324638     180.000000       -4.75     <.0001

                         * Transformed Variables
```

The following statements list the first ten observations of the data set outmi. Note that the values for Oxygen are in the original scale.

```
proc print data=outmi(obs=10);
   title 'First 10 Observations of the Imputed Data Set';
run;
```

Output 9.7.6. Imputed Data Set in Original Scale

First 10 Observations of the Imputed Data Set

Obs	_Imputation_	Oxygen	RunTime	Run Pulse
1	1	44.6090	11.3700	178.000
2	1	45.3130	10.0700	185.000
3	1	54.2970	8.6500	156.000
4	1	59.5710	8.4840	155.503
5	1	49.8740	9.2200	166.031
6	1	44.8110	11.6300	176.000
7	1	43.4130	11.9500	176.000
8	1	44.6435	10.8500	173.761
9	1	39.4420	13.0800	174.000
10	1	60.0550	8.6300	170.000

The preceding results can also be produced from the following statements without using a TRANSFORM statement.

```
data temp;
   set FitMiss;
   LogOxygen= log(Oxygen);
run;

proc mi data=temp seed=37921 mu0=3.91202 10 180 out=outtemp;
   mcmc chain=multiple displayinit;
   var LogOxygen RunTime RunPulse;
run;

data outmi;
   set outtemp;
   Oxygen= exp(LogOxygen);
run;
```

Note that a transformed value of log(50)=3.91202 is used in the MU0= option.

Example 9.8. Saving and Using Parameters for MCMC

This example uses the MCMC method with multiple chains as specified in Example 9.4. It saves the parameter values used for each imputation in an output data set of type EST. This output data set can then be used to impute missing values in other similar input data sets. The following statements invoke the MI procedure and specify the MCMC method with multiple chains to create three imputations.

```
proc mi data=FitMiss seed=55417 nimpute=3 mu0=50 10 180 noprint;
   mcmc chain=multiple outest=miest;
   var Oxygen RunTime RunPulse;
run;
```

The following statements list the parameters used for the imputations. Note that the data set includes observations with _TYPE_='SEED' containing the seed to start the next random number generator.

```
proc print data=miest;
   title 'Parameters for the Imputations';
run;
```

Output 9.8.1. OUTEST Data Set

Parameters for the Imputations

Obs	_Imputation_	_TYPE_	_NAME_	Oxygen	RunTime	RunPulse
1	1	SEED		2099769086.00	2099769086.00	2099769086.00
2	1	PARM		49.31	10.00	172.19
3	1	COV	Oxygen	32.05	-7.47	-28.32
4	1	COV	RunTime	-7.47	2.41	6.75
5	1	COV	RunPulse	-28.32	6.75	128.61
6	2	SEED		419117425.00	419117425.00	419117425.00
7	2	PARM		47.49	10.43	171.58
8	2	COV	Oxygen	41.02	-8.60	-34.29
9	2	COV	RunTime	-8.60	2.25	7.61
10	2	COV	RunPulse	-34.29	7.61	142.94
11	3	SEED		535522494.00	535522494.00	535522494.00
12	3	PARM		45.98	10.82	172.45
13	3	COV	Oxygen	43.24	-9.90	8.14
14	3	COV	RunTime	-9.90	2.75	-2.72
15	3	COV	RunPulse	8.14	-2.72	218.32

The following statements invoke the MI procedure and use the INEST= option in the MCMC statement.

```
proc mi data=FitMiss;
   mcmc inest=miest;
   var Oxygen RunTime RunPulse;
run;
```

Output 9.8.2. Model Information

The MI Procedure

Model Information

Data Set	WORK.FITMISS
Method	MCMC
INEST Data Set	WORK.MIEST
Number of Imputations	3

The remaining tables for the example are identical to the tables in Example 9.4.

References

Anderson, T.W. (1984), *An Introduction to Multivariate Statistical Analysis*, Second Edition, New York: John Wiley & Sons, Inc.

Allison, P.D. (2000), "Multiple Imputation for Missing Data: A Cautionary Tale," *Sociological Methods and Research*, 28, 301–309.

Barnard, J. and Rubin, D.B. (1999), "Small-Sample Degrees of Freedom with Multiple Imputation," *Biometrika*, 86, 948–955.

Dempster, A.P., Laird, N.M., and Rubin, D.B. (1977), "Maximum Likelihood from Incomplete Data via the EM Algorithm," *Journal of the Royal Statistical Society*, Ser. B., 39, 1–38.

Gelman, A. and Rubin, D.B. (1992), "Inference from Iterative Simulation Using Multiple Sequences," *Statistical Science*, 7, 457–472.

Goodnight, J.H. (1979), "A Tutorial on the Sweep Operator," *American Statistician*, 33, 149–158.

Lavori, P.W., Dawson, R., and Shera, D. (1995), "A Multiple Imputation Strategy for Clinical Trials with Truncation of Patient Data," *Statistics in Medicine*, 14, 1913–1925.

Li, K.H. (1988), "Imputation Using Markov Chains," *Journal of Statistical Computation and Simulation*, 30, 57–79.

Li, K.H., Raghunathan, T.E., and Rubin, D.B. (1991), "Large-Sample Significance Levels from Multiply Imputed Data Using Moment-Based Statistics and an F Reference Distribution," *Journal of the American Statistical Association*, 86, 1065–1073.

Little, R.J.A. and Rubin, D.B. (1987), *Statistical Analysis with Missing Data*, New York: John Wiley & Sons, Inc.

Liu, C. (1993), "Bartlett's Decomposition of the Posterior Distribution of the Covariance for Normal Monotone Ignorable Missing Data," *Journal of Multivariate Analysis*, 46, 198–206.

McLachlan, G.J. and Krishnan, T. (1997), *The EM Algorithm and Extensions*, New York: John Wiley & Sons, Inc.

Rosenbaum, P.R. and Rubin, D.B. (1983), "The Central Role of the Propensity Score in Observational Studies for Causal Effects," *Biometrika*, 70, 41–55.

Rubin, D.B. (1976), "Inference and Missing Data," *Biometrika*, 63, 581–592.

Rubin, D.B. (1987), *Multiple Imputation for Nonresponse in Surveys*, New York: John Wiley & Sons, Inc.

Rubin, D.B. (1996), "Multiple Imputation After 18+ Years," *Journal of the American Statistical Association*, 91, 473–489.

SAS Institute Inc. (1999), *SAS/ETS User's Guide, Version 8*, Cary, NC: SAS Institute Inc.

SAS Institute Inc. (1999), *SAS/GRAPH Software: Reference, Version 8*, Cary, NC: SAS Institute Inc.

SAS Institute Inc. (1999), *SAS Language Reference: Concepts, Version 8*, Cary, NC: SAS Institute Inc.

SAS Institute Inc. (1999), *SAS Procedures Guide, Version 8*, Cary, NC: SAS Institute Inc.

SAS Institute Inc. (1999), *SAS/STAT User's Guide, Version 8*, Cary, NC: SAS Institute Inc.

Schafer, J.L. (1997), *Analysis of Incomplete Multivariate Data*, New York: Chapman and Hall.

Tanner, M.A. and Wong, W.H. (1987), "The Calculation of Posterior Distributions by Data Augmentation," *Journal of the American Statistical Association*, 82, 528–540.

Chapter 10
The MIANALYZE Procedure

Chapter Table of Contents

Chapter 10
The MIANALYZE Procedure

Overview

The experimental MIANALYZE procedure combines the results of the analyses of imputations and generates valid statistical inferences. Multiple imputation provides a useful strategy for analyzing data sets with missing values. Instead of filling in a single value for each missing value, Rubin's (1976; 1987) multiple imputation strategy replaces each missing value with a set of plausible values that represent the uncertainty about the right value to impute. You can implement the strategy with two SAS procedures: PROC MI, which generates imputed data sets, and PROC MIANALYZE, which combines the results of analyses carried out on the data sets. These two procedures are available in experimental form in Release 8.2 of the SAS System.

These analyses of imputations are obtained by using standard SAS procedures (such as PROC REG) for complete data. No matter which complete-data analysis is used, the process of combining results from different imputed data sets is essentially the same. This results in valid statistical inferences that properly reflect the uncertainty due to missing values.

The MIANALYZE procedure reads the parameter estimates and associated covariance matrix that are computed by the standard statistical procedure for each imputed data set. The MIANALYZE procedure then derives valid univariate and multivariate inferences for these parameters.

For some parameters of interest, it is not straightforward to compute estimates and associated covariance matrices with standard statistical SAS procedures. Examples include correlation coefficients between two variables and ratios of variable means. Special cases such as these are described in the "Examples of the Complete-Data Inferences" section on page 214.

Getting Started

The Fitness data set has been altered to contain an arbitrary missing pattern:

```
*----------------- Data on Physical Fitness -----------------*
| These measurements were made on men involved in a physical |
| fitness course at N.C. State University.                   |
| Only selected variables of                                 |
| Oxygen (oxygen intake, ml per kg body weight per minute),  |
| Runtime (time to run 1.5 miles in minutes), and            |
| RunPulse (heart rate while running) are used.              |
| Certain values were changed to missing for the analysis.   |
*------------------------------------------------------------*;
```

```
data FitMiss;
   input Oxygen RunTime RunPulse @@;
   datalines;
44.609  11.37  178     45.313  10.07  185
54.297   8.65  156     59.571    .      .
49.874   9.22    .     44.811  11.63  176
  .     11.95  176     49.091  10.85    .
39.442  13.08  174     60.055   8.63  170
50.541    .      .     37.388  14.03  186
44.754  11.12  176     47.273    .      .
51.855  10.33  166     49.156   8.95  180
40.836  10.95  168     46.672  10.00    .
  .     10.25    .     50.388  10.08  168
39.407  12.63  174     46.080  11.17  156
45.441   9.63  164       .      8.92  146
45.118  11.08    .     39.203  12.88  168
45.790  10.47  186     50.545   9.93  148
48.673   9.40  186     47.920  11.50  170
47.467  10.50  170
;
```

Assume that the data are multivariate normally distributed and that the missing data are missing at random (see the "Statistical Assumptions for Multiple Imputation" section in "The MI Procedure" chapter for a description of these assumptions). The following statements use the MI procedure to impute missing values for the FitMiss data set.

```
proc mi data=FitMiss noprint out=outmi seed=37851;
   var Oxygen RunTime RunPulse;
run;
```

The MI procedure creates imputed data sets, which are stored in the outmi data set. A variable named _Imputation_ indicates the imputation numbers. Based on m imputations, m different sets of the point and variance estimates for a parameter can be computed. In this example, $m = 5$ is the default.

The following statements generate regression coefficients for each of the five imputed data sets:

```
proc reg data=outmi outest=outreg covout noprint;
   model Oxygen= RunTime RunPulse;
   by _Imputation_;
run;

proc print data=outreg(obs=8);
   var _Imputation_ _Type_ _Name_
      Intercept RunTime RunPulse;
   title 'Parameter Estimates from Imputed Data Sets';
run;
```

Parameter Estimates from Imputed Data Sets

Obs	_Imputation_	_TYPE_	_NAME_	Intercept	RunTime	RunPulse
1	1	PARMS		97.2874	-2.98892	-0.10684
2	1	COV	Intercept	55.7516	-0.73348	-0.27870
3	1	COV	RunTime	-0.7335	0.15167	-0.00509
4	1	COV	RunPulse	-0.2787	-0.00509	0.00194
5	2	PARMS		90.9324	-2.93338	-0.07391
6	2	COV	Intercept	37.5576	-0.25970	-0.20442
7	2	COV	RunTime	-0.2597	0.13978	-0.00722
8	2	COV	RunPulse	-0.2044	-0.00722	0.00166

Figure 10.1. Parameter Estimates

The following statements combine the five sets of regression coefficients:

```
proc mianalyze data=outreg;
   var Intercept RunTime RunPulse;
run;
```

The MIANALYZE Procedure

Model Information

Data Set	WORK.OUTREG
Number of Imputations	5

Multiple Imputation Variance Information

	Variance			
Parameter	Between	Within	Total	DF
Intercept	74.179857	57.287519	146.303348	10.805
RunTime	0.034202	0.142151	0.183193	79.694
RunPulse	0.001533	0.002304	0.004144	20.292

Multiple Imputation Variance Information

Parameter	Relative Increase in Variance	Fraction Missing Information
Intercept	1.553843	0.665161
RunTime	0.288719	0.242803
RunPulse	0.798522	0.491731

Figure 10.2. Model Information and Variance Information Tables

The "Model Information" table lists the input data set(s) and the number of imputations. The "Multiple Imputation Variance Information" table displays the between-imputation, within-imputation, and total variances for combining complete-data inferences. It also displays the degrees of freedom for the total variance, the relative increase in variance due to missing values, and the fraction of missing information for each parameter estimate.

```
                        The MIANALYZE Procedure

                  Multiple Imputation Parameter Estimates

Parameter        Estimate      Std Error    95% Confidence Limits        DF

Intercept       92.156840      12.095592     65.47596     118.8377   10.805
RunTime         -2.955317       0.428011     -3.80714      -2.1035   79.694
RunPulse        -0.079851       0.064376     -0.21401       0.0543   20.292

                  Multiple Imputation Parameter Estimates

                  Parameter         Minimum        Maximum

                  Intercept       77.939497      99.920480
                  RunTime         -3.159663      -2.660085
                  RunPulse        -0.112277      -0.015111

                  Multiple Imputation Parameter Estimates

                                          t for H0:
          Parameter          Theta0   Parameter=Theta0   Pr > |t|

          Intercept               0              7.62     <.0001
          RunTime                 0             -6.90     <.0001
          RunPulse                0             -1.24     0.2290
```

Figure 10.3. Multiple Imputation Parameter Estimates

The "Multiple Imputation Parameter Estimates" table displays a combined estimate and standard error for each regression coefficient (parameter). Inferences are based on *t* distributions. The table displays a 95% confidence interval and a *t*-test with the associated *p*-value for the hypothesis that the parameter is equal to the value specified with the THETA0= option (in this case, zero by default). The minimum and maximum parameter estimates from the imputed data sets are also displayed.

Syntax

The following statements are available in PROC MIANALYZE.

PROC MIANALYZE < *options* > **;**

BY *variables* **;**
VAR *variables* **;**

The BY statement specifies groups in which separate analyses are performed.

The VAR statement lists the numeric variables to be analyzed. The statement is required.

The rest of this section gives detailed syntax information for each of these statements. The PROC MIANALYZE statement and the VAR statement are required for the MIANALYZE procedure.

PROC MIANALYZE Statement

PROC MIANALYZE < *options* > **;**

The following table summarizes the options in the PROC MIANALYZE statement.

Table 10.1. Summary of PROC MIANALYZE Options

Tasks	Options
Specify input data sets	
a COV, CORR, or EST type data set	DATA=
parameter estimates and covariance matrices	PARMS=, COVB=
parameter estimates and $(X'X)^{-1}$ matrices	PARMS=, XPXI=
Specify statistical analysis	
parameters under the null hypothesis	THETA0=
level for the confidence interval	ALPHA=
complete-data degrees of freedom	EDF=
multivariate inferences	MULT

The following are explanations of the options that can be used in the PROC MIANALYZE statement (in alphabetical order):

ALPHA=*p*

specifies that confidence limits are to be constructed for the parameter estimates with confidence level $100(1-p)\%$, where $0 < p < 1$. The default is p=0.05.

COVB=*SAS-data-set*

names an input SAS data set that contains covariance matrices of the parameter estimates from imputed data sets. If you provide a COVB= data set, you must also provide a PARMS= data set.

DATA=*SAS-data-set*

names a specially structured input SAS data set that contains estimates from imputed data sets. This data set must have a TYPE of EST, COV, or CORR:

- if TYPE=EST, the data set contains the parameter estimates and associated covariance matrices.
- if TYPE=COV, the data set contains the sample means, sample sizes, and covariance matrices. Each covariance matrix for variables is divided by the sample size n to create the covariance matrix for parameter estimates.
- if TYPE=CORR, the data set contains the sample means, sample sizes, standard errors, and correlation matrices. The covariance matrices are computed from the correlation matrices and associated standard errors. Each covariance matrix for variables is divided by the sample size n to create the covariance matrix for parameter estimates.

If you do not specify an input data set with the DATA=, COVB=, or XPXI= option, then the most recently created SAS data set is used as an input DATA= data set.

EDF=*number*

specifies the complete-data degrees of freedom for the parameter estimates. This is used to compute an adjusted degrees of freedom for each parameter estimate. By default, EDF=∞ and the degrees of freedom for each parameter estimate is not adjusted.

MULT
MULTIVARIATE

requests multivariate inference for the variables.

PARMS=*SAS-data-set*

names an input SAS data set that contains parameter estimates computed from imputed data sets. If you provide a PARMS= data set, you must also provide a COVB= or XPXI= data set.

THETA0=*numbers*
MU0=*numbers*

specifies the parameter values $\boldsymbol{\theta}_0$ under the null hypothesis $\boldsymbol{\theta} = \boldsymbol{\theta}_0$ in the *t* tests for location for the variables. If only one number $\boldsymbol{\theta}_0$ is specified, that number is used for all variables. If more than one number is specified, the specified numbers correspond to variables in the VAR statement in the order in which they appear in the VAR statement.

XPXI=*SAS-data-set*

names an input SAS data set that contains the $(X'X)^{-1}$ matrices associated with the parameter estimates computed from imputed data sets. If you provide an XPXI= data set, you must also provide a PARMS= data set. In this case, PROC MIANALYZE reads the standard errors of the estimates from the PARMS= data. The standard errors and $(X'X)^{-1}$ matrices are used to derive the covariance matrices.

BY Statement

BY *variables* **;**

You can specify a BY statement with PROC MIANALYZE to obtain separate analyses on observations in groups defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data using the SORT procedure with a similar BY statement.
- Specify the BY statement option NOTSORTED or DESCENDING in the BY statement for the MI procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables using the DATASETS procedure.

For more information on the BY statement, refer to the discussion in *SAS Language Reference: Concepts, Version 8*. For more information on the DATASETS procedure, refer to the discussion in the *SAS Procedures Guide, Version 8*.

VAR Statement

VAR *variables* **;**

The VAR statement lists the variables to be analyzed. The statement is required, and the variables must be numeric.

Details

Input Data Sets

You can specify input data sets using one of the following option combinations:

- DATA=, which provides both parameter estimates and the associated covariance matrix in a single input data set.
- PARMS= and COVB=, which provide parameter estimates and the associated covariance matrix in separate data sets, respectively.
- PARMS= and XPXI=, which provide parameter estimates and the associated standard errors in a PARMS= data set and the associated $(X'X)^{-1}$ matrix in an XPXI= data set.

The appropriate combination depends on the SAS procedure you used to create the parameter estimates and associated covariance matrix. For instance, if you used PROC REG to create an OUTEST= data set containing the parameter estimates and covariance matrix, you would use the DATA= option to read the OUTEST= data set. Each input data set contains the variable _Imputation_ to identify the imputation by number.

If you do not specify an input data set with the DATA=, COVB=, or XPXI= option, then the most recently created SAS data set is used as an input DATA= data set.

DATA= data set

The input DATA= data set is a specially structured SAS data set created by statistical procedures available with SAS software. The data set must have a TYPE of EST, COV, or CORR.

With TYPE=EST, the MIANALYZE procedure reads parameter estimates from observations with _TYPE_='PARM' or _TYPE_='PARMS', and covariance matrices for parameter estimates from observations with _TYPE_='COV' or _TYPE_='COVB'.

With TYPE=COV, the procedure reads sample means from observations with _TYPE_='MEAN', sample size n from observations with _TYPE_='N', and covariance matrices for variables from observations with _TYPE_='COV'.

With TYPE=CORR, the procedure reads sample means from observations with _TYPE_='MEAN', sample size n from observations with _TYPE_='N', correlation matrices for variables from observations with _TYPE_='CORR', and standard errors for variables from observations with _TYPE_='STD'. The standard errors and correlation matrix are used to generate a covariance matrix for the variables.

Note that with TYPE=COV or CORR, each covariance matrix for the variables is divided by n to create the covariance matrix for the sample means.

PARMS= and COVB= data sets

The input PARMS= data set contains parameter estimates, and the input COVB= data set contains associated covariance matrices computed from imputed data sets. Such data sets are typically created with an ODS OUTPUT statement using procedures such as PROC MIXED and PROC GENMOD.

The MIANALYZE procedure uses a PARMS= data set to read parameter names from the variable Parameter or Effect, and it reads parameter estimates from the variable Estimate.

The MIANALYZE procedure uses a COVB= data set to read parameter names from the variable Parameter, Effect, or RowName, and it reads covariance matrices from the subsequent variables Col1, Col2, ... or Prm1, Prm2, ... in the data set.

PARMS= and XPXI= data sets

The input PARMS= data set contains parameter estimates, and the input XPXI= data set contains associated $(X'X)^{-1}$ matrices computed from imputed data sets. Such data sets are typically created with an ODS OUTPUT statement using a procedure such as PROC GLM.

The MIANALYZE procedure uses a PARMS= data set to read parameter names from the variable Parameter; it reads parameter estimates from the variable Estimate, and it reads standard errors for parameter estimates from the variable StdErr.

The MIANALYZE procedure uses an XPXI= data set to read parameter names from the variable Parameter, and it reads $(X'X)^{-1}$ matrices from the subsequent variables in the data set.

Combining Inferences from Imputed Data Sets

With m imputations, m different sets of the point and variance estimates for a parameter Q can be computed. Let $\hat{Q}_i$ and $\hat{U}_i$ be the point and variance estimates from the ith imputed data set, i=1, 2, ..., m. Then the combined point estimate for Q from multiple imputation is the average of the m complete-data estimates:

$$\overline{Q} = \frac{1}{m}\sum_{i=1}^{m}\hat{Q}_i$$

Let $\overline{U}$ be the within-imputation variance, which is the average of the m complete-data estimates:

$$\overline{U} = \frac{1}{m}\sum_{i=1}^{m}\hat{U}_i$$

and B be the between-imputation variance

$$B = \frac{1}{m-1}\sum_{i=1}^{m}(\hat{Q}_i - \overline{Q})^2$$

Then the variance estimate associated with $\overline{Q}$ is the total variance (Rubin 1987)

$$T = \overline{U} + (1 + \frac{1}{m})B$$

The statistic $(Q - \overline{Q})T^{-(1/2)}$ is approximately distributed as t with v_m degrees of freedom (Rubin 1987), where

$$v_m = (m-1)[1 + \frac{\overline{U}}{(1+m^{-1})B}]^2$$

When the complete-data degrees of freedom v_0 is small, and there is only a modest proportion of missing data, the computed degrees of freedom, v_m, can be much larger than v_0, which is inappropriate. Barnard and Rubin (1999) recommend the use of an adjusted degrees of freedom

$$v_m^* = \left[\frac{1}{v_m} + \frac{1}{\hat{v}_{obs}}\right]^{-1}$$

where $\hat{v}_{obs} = (1-\gamma)\,v_0(v_0+1)/(v_0+3)$ and $\gamma = (1+m^{-1})B/T$.

If you specify the complete-data degrees of freedom v_0 with the EDF= option, the MIANALYZE procedure uses the adjusted degrees of freedom, v_m^*, for inference. Otherwise, the degrees of freedom v_m is used.

The degrees of freedom v_m depends on m and the ratio

$$r = \frac{(1 + m^{-1})B}{\overline{U}}$$

The ratio r is called the relative increase in variance due to nonresponse (Rubin 1987). When there is no missing information about Q, the values of r and B are both zero. With a large value of m or a small value of r, the degrees of freedom v will be large and the distribution of $(Q - \overline{Q})T^{-(1/2)}$ will be approximately normal.

Another useful statistic is the fraction of missing information about Q:

$$\hat{\lambda} = \frac{r + 2/(v + 3)}{r + 1}$$

Both statistics r and λ are helpful diagnostics for assessing how the missing data contribute to the uncertainty about Q.

Multiple Imputation Efficiency

The relative efficiency (RE) of using the finite m imputation estimator, rather than using an infinite number for the fully efficient imputation, in units of variance, is approximately a function of m and λ (Rubin 1987, p. 114).

$$RE = (1 + \frac{\lambda}{m})^{-1}$$

The following table shows relative efficiencies with different values of m and λ. For cases with little missing information, only a small number of imputations are necessary.

Table 10.2. Relative Efficiency

	λ				
m	**10%**	**20%**	**30%**	**50%**	**70%**
3	0.9677	0.9375	0.9091	0.8571	0.8108
5	0.9804	0.9615	0.9434	0.9091	0.8772
10	0.9901	0.9804	0.9709	0.9524	0.9346
20	0.9950	0.9901	0.9852	0.9756	0.9662

Multivariate Inferences

Multivariate inference based on Wald tests can be done with m imputed data sets. The approach is a generalization of the approach taken in the univariate case (Rubin 1987, p. 137; Schafer 1997, p. 113). Suppose that $\hat{\mathbf{Q}}_i$ and $\hat{\mathbf{U}}_i$ are the point and covariance matrix estimates for a vector-valued parameter $\mathbf{Q}$ (such as a multivariate mean) from the ith imputed data set, i=1, 2, ..., m. Then the combined point estimate for $\mathbf{Q}$ from the multiple imputation is the average of the m complete-data estimates:

$$\overline{\mathbf{Q}} = \frac{1}{m}\sum_{i=1}^{m}\hat{\mathbf{Q}}_i$$

Suppose that $\overline{\mathbf{U}}$ is the within-imputation covariance matrix, which is the average of the m complete-data estimates

$$\overline{\mathbf{U}} = \frac{1}{m}\sum_{i=1}^{m}\hat{\mathbf{U}}_i$$

and suppose that $\mathbf{B}$ is the between-imputation covariance matrix

$$\mathbf{B} = \frac{1}{m-1}\sum_{i=1}^{m}(\hat{\mathbf{Q}}_i - \overline{\mathbf{Q}})(\hat{\mathbf{Q}}_i - \overline{\mathbf{Q}})'$$

Then the covariance matrix associated with $\overline{\mathbf{Q}}$ is the total covariance matrix

$$\mathbf{T}_0 = \overline{\mathbf{U}} + (1 + \frac{1}{m})\mathbf{B}$$

The natural multivariate extension of the t statistic used in the univariate case is the F statistic

$$F_0 = (\mathbf{Q} - \overline{\mathbf{Q}})'\mathbf{T}_0^{-1}(\mathbf{Q} - \overline{\mathbf{Q}})$$

with degrees of freedom p and

$$v = (m-1)(1 + 1/r)^2$$

where

$$r = (1 + \frac{1}{m})trace(\mathbf{B}\overline{\mathbf{U}}^{-1})/p$$

is an average relative increase in variance due to nonresponse (Rubin 1987, p. 137; Schafer 1997, p. 114).

However, the reference distribution of the statistic F_0 is not easily derived. Especially for small m, the between-imputation covariance matrix $\mathbf{B}$ is unstable and does not have full rank for $m \leq p$ (Schafer 1997, p. 113).

One solution is to make an additional assumption that the population between-imputation and within-imputation covariance matrices are proportional to each other (Schafer 1997, p. 113). This assumption implies that the fractions of missing information for all components of $\mathbf{Q}$ are equal. Under this assumption, a more stable estimate of the total covariance matrix is

$$\mathbf{T} = (1 + r)\overline{\mathbf{U}}$$

With the total covariance matrix $\mathbf{T}$, the F statistic (Rubin 1987, p. 137)

$$F = (\mathbf{Q} - \overline{\mathbf{Q}})'\mathbf{T}^{-1}(\mathbf{Q} - \overline{\mathbf{Q}})/p$$

has an F distribution with degrees of freedom p and v_1, where

$$v_1 = \frac{1}{2}(p + 1)(m - 1)(1 + \frac{1}{r})^2$$

For $t = p(m - 1) \leq 4$, PROC MIANALYZE uses the degrees of freedom v_1 in the analysis. For $t = p(m-1) > 4$, PROC MIANALYZE uses v_2, a better approximation of the degrees of freedom given by Li, Raghunathan, and Rubin (1991).

$$v_2 = 4 + (t - 4)(1 + \frac{1}{r}(1 - \frac{2}{t}))^2$$

Examples of the Complete-Data Inferences

For a given parameter of interest, it is not always possible to compute the estimate and associated covariance matrix directly from a SAS procedure. This section gives examples of parameters with their estimates and associated covariance matrices, which provide the input to the MIANALYZE procedure. Some are straightforward, and others require special techniques.

Means

For a population mean vector $\boldsymbol{\mu}$, the usual estimate is the sample mean vector

$$\overline{\mathbf{y}} = \frac{1}{n}\sum \mathbf{y}_i$$

A variance estimate for $\overline{\mathbf{y}}$ is $\frac{1}{n}\mathbf{S}$, where $\mathbf{S}$ is the sample covariance matrix

$$\mathbf{S} = \frac{1}{n-1}\sum (\mathbf{y}_i - \overline{\mathbf{y}})(\mathbf{y}_i - \overline{\mathbf{y}})'$$

These statistics can be computed from a procedure such as CORR. This approach is illustrated in Example 10.1.

Regression Coefficients

Many SAS procedures are available for regression analysis. Among them, REG provides the most general analysis capabilities, and others like LOGISTIC and MIXED provide more specialized analyses.

Some regression procedures, such as REG and LOGISTIC, create an EST type data set that contains both the parameter estimates for the regression coefficients and their associated covariance matrix. You can read an EST type data set in the MIANALYZE procedure with the DATA= option. This approach is illustrated in Example 10.2.

Other procedures, such as GLM, MIXED, and GENMOD, do not generate EST type data sets for regression coefficients. For MIXED and GENMOD, you can use ODS OUTPUT statement to save parameter estimates in a data set and the associated covariance matrix in a separate data set. These data sets are then read in the MIANALYZE procedure with the PARMS= and COVB= options, respectively. This approach is illustrated in Example 10.3 for PROC MIXED and in Example 10.4 for PROC GENMOD.

PROC GLM does not display tables for covariance matrices. However, you can use the ODS OUTPUT statement to save parameter estimates and associated standard errors in a data set and the associated $(X'X)^{-1}$ matrix in a separate data set. These data sets are then read in the MIANALYZE procedure with the PARMS= and XPXI= options, respectively. This approach is illustrated in Example 10.5.

Correlation Coefficients

For the population correlation coefficient ρ, a point estimate is the sample correlation coefficient r. However, for nonzero ρ, the distribution of r is skewed.

The distribution of r can be normalized through Fisher's Z transformation

$$z(r) = \frac{1}{2}\, log\left(\frac{1+r}{1-r}\right)$$

$z(r)$ is approximately normally distributed with mean $z(\rho)$ and variance $1/(n-3)$.

With a point estimate $\hat{z}$ and an approximate 95% confidence interval (z_1, z_2) for $z(\rho)$, a point estimate $\hat{r}$ and a 95% confidence interval (r_1, r_2) for ρ can be obtained by applying the inverse transformation

$$r = \frac{e^{2z}-1}{e^{2z}+1}$$

to $z = \hat{z}, z_1$, and z_2.

This approach is illustrated in Example 10.3.

Ratios of Variable Means

For the ratio μ_1/μ_2 of means for variables Y_1 and Y_2, the point estimate is $\overline{y}_1/\overline{y}_2$, the ratio of the sample means. The Taylor expansion and delta method can be applied to the function y_1/y_2 to obtain the variance estimate (Schafer 1997, p. 196)

$$\frac{1}{n}\left[\left(\frac{\overline{y}_1}{\overline{y}_2^2}\right)^2 s_{22} - 2\left(\frac{\overline{y}_1}{y_2^2}\right)\left(\frac{1}{\overline{y}_2}\right)s_{12} + \left(\frac{1}{\overline{y}_2}\right)^2 s_{11}\right]$$

where s_{11} and s_{22} are the sample variances of Y_1 and Y_2, respectively, and s_{12} is the sample covariance between Y_1 and Y_2.

A ratio of sample means will be approximately unbiased and normally distributed if the coefficient of variation of the denominator (the standard error for the mean divided by the estimated mean) is 10% or less (Cochran 1977, p. 166; Schafer 1997, p. 196). This approach is illustrated in Example 10.7.

ODS Table Names

PROC MIANALYZE assigns a name to each table it creates. You must use these names to reference tables when using the Output Delivery System (ODS). These names are listed in the following table. For more information on ODS, refer to the chapter "Using the Output Delivery System" in the *SAS/STAT User's Guide, Version 8.*

Table 10.3. ODS Tables Produced in PROC MIANALYZE

ODS Table Name	Description	Option
ModelInfo	Model information	
VarianceInfo	Variance information	
ParmEst	Parameter estimates	
BetweenCov	Between-imputation covariance matrix	MULT
WithinCov	Within-imputation covariance matrix	MULT
TotalCov	Total covariance matrix	MULT
MultInf	Multivariate inference	MULT

Examples

The following statements generate five imputed data sets to be used in this section. The data set FitMiss was created in the section "Getting Started" on page 203. See "The MI Procedure" chapter for details concerning the MI procedure.

```
proc mi data=FitMiss seed=37851 noprint out=outmi;
   var Oxygen RunTime RunPulse;
run;
```

Example 10.1. Reading Means and Covariance Matrices from a DATA= COV Data Set

This example creates a COV type data set that contains sample means and covariance matrices computed from imputed data sets. These estimates are then combined to generate valid statistical inferences about the population means.

The following statements use the CORR procedure to generate sample means and a covariance matrix for the variables in each imputed data set.

```
proc corr data=outmi cov out=outcov(type=cov) nocorr noprint;
   var Oxygen RunTime RunPulse;
   by _Imputation_;
run;

proc print data=outcov(obs=12);
   title 'CORR Means and Covariance Matrices'
         ' (First Two Imputations)';
run;
```

Output 10.1.1. COV Data Set

CORR Means and Covariance Matrices (First Two Imputations)

Obs	_Imputation_	_TYPE_	_NAME_	Oxygen	RunTime	RunPulse
1	1	COV	Oxygen	28.2139	-5.9997	-29.700
2	1	COV	RunTime	-5.9997	1.8353	4.812
3	1	COV	RunPulse	-29.7002	4.8121	143.368
4	1	MEAN		47.3458	10.5859	171.301
5	1	STD		5.3117	1.3547	11.974
6	1	N		31.0000	31.0000	31.000
7	2	COV	Oxygen	28.7419	-6.6895	-39.006
8	2	COV	RunTime	-6.6895	2.0553	8.938
9	2	COV	RunPulse	-39.0060	8.9385	172.998
10	2	MEAN		47.3316	10.6004	169.204
11	2	STD		5.3612	1.4336	13.153
12	2	N		31.0000	31.0000	31.000

Note that the covariance matrices in the data set outcov are estimated covariance matrices of variables, $V(\mathbf{y})$. The estimated covariance matrix of the sample means is $V(\overline{\mathbf{y}}) = V(\mathbf{y})/n$, where n is the sample size, and is not the same as an estimated covariance matrix for variables.

The following statements combine the results for the imputed data sets, and derive both univariate and multivariate inferences about the means. The EDF= option is specified to request that the adjusted degrees of freedom be used in the analysis. For sample means based on 31 observations, the complete-data error degrees of freedom is 30.

```
proc mianalyze data=outcov edf=30 mult;
   var Oxygen RunTime RunPulse;
run;
```

Output 10.1.2. Multiple Imputation Variance Information

The MIANALYZE Procedure

Multiple Imputation Variance Information

Parameter	Variance: Between	Within	Total	DF
Oxygen	0.007552	0.969527	0.978590	27.904
RunTime	0.001577	0.070505	0.072397	27.317
RunPulse	1.363982	4.469865	6.106642	15.052

Multiple Imputation Variance Information

Parameter	Relative Increase in Variance	Fraction Missing Information
Oxygen	0.009348	0.009304
RunTime	0.026834	0.026466
RunPulse	0.366181	0.292981

The "Multiple Imputation Variance Information" table displays the between-imputation variance, within-imputation variance, and total variance for each univariate inference. It also displays the degrees of freedom for the total variance. The relative increase in variance due to missing values and the fraction of missing information for each variable are also displayed. A detailed description of these statistics is provided in the "Combining Inferences from Imputed Data Sets" section on page 211 and the "Multiple Imputation Efficiency" section on page 212.

Output 10.1.3. Multiple Imputation Parameter Estimates

The MIANALYZE Procedure

Multiple Imputation Parameter Estimates

Parameter	Estimate	Std Error	95% Confidence Limits		DF
Oxygen	47.360514	0.989237	45.3338	49.3872	27.904
RunTime	10.557687	0.269067	10.0059	11.1095	27.317
RunPulse	169.970152	2.471162	164.7046	175.2357	15.052

Multiple Imputation Parameter Estimates

Parameter	Minimum	Maximum
Oxygen	47.273244	47.506505
RunTime	10.505186	10.600445
RunPulse	169.053679	171.301061

Multiple Imputation Parameter Estimates

Parameter	Theta0	t for H0: Parameter=Theta0	Pr > |t|
Oxygen	0	47.88	<.0001
RunTime	0	39.24	<.0001
RunPulse	0	68.78	<.0001

The "Multiple Imputation Parameter Estimates" table displays the estimated mean and corresponding standard error for each variable. The table also displays a 95% confidence interval for the mean and a *t* statistic with the associated *p*-value for testing the hypothesis that the mean is equal to the value specified. You can use the THETA0= option to specify the value for the null hypothesis, which is zero by default. The table also displays the minimum and maximum parameter estimates from the imputed data sets.

With the MULT option, the procedure displays the between-imputation covariance matrix, within-imputation covariance matrix, and total covariance matrix for multivariate inference.

Output 10.1.4. Within-Imputation Covariance Matrices

The MIANALYZE Procedure

Within-Imputation Covariance Matrix

	Oxygen	RunTime	RunPulse
Oxygen	0.969526960	-0.222851446	-0.997345212
RunTime	-0.222851446	0.070505098	0.222928579
RunPulse	-0.997345212	0.222928579	4.469864543

Between-Imputation Covariance Matrix

	Oxygen	RunTime	RunPulse
Oxygen	0.007552454	-0.002444417	0.068595335
RunTime	-0.002444417	0.001576638	-0.010973127
RunPulse	0.068595335	-0.010973127	1.363981626

Total Covariance Matrix

	Oxygen	RunTime	RunPulse
Oxygen	1.160483133	-0.266743840	-1.193780414
RunTime	-0.266743840	0.084391647	0.266836165
RunPulse	-1.193780414	0.266836165	5.350240500

Assuming that the between-imputation covariance matrix is proportional to the within-imputation covariance matrix, the procedure also displays a multivariate inference for all the parameters taken jointly.

Output 10.1.5. Multiple Imputation Multivariate Inference

The MIANALYZE Procedure

Multiple Imputation Multivariate Inference
Assuming Proportionality of Between/Within Covariance Matrices

Avg Relative Increase in Variance	Num DF	Den DF	F for H0: Parameter=Theta0	Pr > F
0.196958	3	222.91	11408.0	<.0001

Assuming that the within-imputation covariance matrix is proportional to the between-imputation covariance matrix, the table shows a significant p-value for the null hypothesis that the population means are all equal to zero.

With the exception of the multivariate inference, the preceding results could also have been obtained with the MI procedure.

Example 10.2. Reading Regression Results from a DATA= EST Data Set

This example creates an EST type data set that contains regression coefficients and their corresponding covariance matrices computed from imputed data sets. These estimates are then combined to generate valid statistical inferences about the regression model.

The following statements use the REG procedure to generate regression coefficients:

```
proc reg data=outmi outest=outreg covout noprint;
   model Oxygen= RunTime RunPulse;
   by _Imputation_;
run;

proc print data=outreg(obs=8);
   var _Imputation_ _Type_ _Name_
      Intercept RunTime RunPulse;
   title 'REG Model Coefficients and Covariance matrices'
         ' (First Two Imputations)';
run;
```

Output 10.2.1. EST Type Data Set

REG Model Coefficients and Covariance matrices (First Two Imputations)

Obs	_Imputation_	_TYPE_	_NAME_	Intercept	RunTime	RunPulse
1	1	PARMS		97.2874	-2.98892	-0.10684
2	1	COV	Intercept	55.7516	-0.73348	-0.27870
3	1	COV	RunTime	-0.7335	0.15167	-0.00509
4	1	COV	RunPulse	-0.2787	-0.00509	0.00194
5	2	PARMS		90.9324	-2.93338	-0.07391
6	2	COV	Intercept	37.5576	-0.25970	-0.20442
7	2	COV	RunTime	-0.2597	0.13978	-0.00722
8	2	COV	RunPulse	-0.2044	-0.00722	0.00166

The following statements combine the results for the imputed data sets. The EDF= option is specified to request that the adjusted degrees of freedom be used in the analysis. For a regression model with three independent variables (including the intercept) and 31 observations, the complete-data error degrees of freedom is 28.

```
proc mianalyze data=outreg edf=28;
   var Intercept RunTime RunPulse;
run;
```

Output 10.2.2. Multiple Imputation Variance Information

The MIANALYZE Procedure

Multiple Imputation Variance Information

Parameter	Variance: Between	Variance: Within	Variance: Total	DF
Intercept	74.179857	57.287519	146.303348	5.2619
RunTime	0.034202	0.142151	0.183193	16.195
RunPulse	0.001533	0.002304	0.004144	8.4786

Multiple Imputation Variance Information

Parameter	Relative Increase in Variance	Fraction Missing Information
Intercept	1.553843	0.665161
RunTime	0.288719	0.242803
RunPulse	0.798522	0.491731

The "Multiple Imputation Variance Information" table displays the between-imputation, within-imputation, and total variances for combining complete-data inferences.

Output 10.2.3. Multiple Imputation Parameter Estimates

The MIANALYZE Procedure

Multiple Imputation Parameter Estimates

Parameter	Estimate	Std Error	95% Confidence Limits		DF
Intercept	92.156840	12.095592	61.52383	122.7898	5.2619
RunTime	-2.955317	0.428011	-3.86177	-2.0489	16.195
RunPulse	-0.079851	0.064376	-0.22686	0.0672	8.4786

Multiple Imputation Parameter Estimates

Parameter	Minimum	Maximum
Intercept	77.939497	99.920480
RunTime	-3.159663	-2.660085
RunPulse	-0.112277	-0.015111

Multiple Imputation Parameter Estimates

Parameter	Theta0	t for H0: Parameter=Theta0	Pr > |t|
Intercept	0	7.62	0.0005
RunTime	0	-6.90	<.0001
RunPulse	0	-1.24	0.2481

The "Multiple Imputation Parameter Estimates" table displays the estimated mean and standard error of the regression coefficients. The inferences are based on the *t* distribution. The table also displays a 95% mean confidence interval and a *t* test with the associated *p*-value for the hypothesis that the regression coefficient is equal to zero. Since the p-value for RunPulse is 0.1597, this variable can be removed from the regression model.

Example 10.3. Reading Mixed Model Results from PARMS= and COVB= Data Sets

This example creates data sets containing parameter estimates and covariance matrices computed by a mixed model analysis for a set of imputed data sets. These estimates are then combined to generate valid statistical inferences about the parameters.

The following PROC MIXED statements generate the fixed-effect parameter estimates and covariance matrix for each imputed data set:

```
proc mixed data=outmi;
   model Oxygen= RunTime RunPulse/solution covb;
   by _Imputation_;
   ods output SolutionF=mixparms CovB=mixcovb;
run;

proc print data=mixparms (obs=6);
   var _Imputation_ Effect Estimate;
   title 'MIXED Model Coefficients (First Two Imputations)';
run;

proc print data=mixcovb (obs=6);
   var _Imputation_ Effect Col1 Col2 Col3;
   title 'MIXED Covariance Matrices (First Two Imputations)';
run;
```

Output 10.3.1. PROC MIXED Model Coefficients

MIXED Model Coefficients (First Two Imputations)

Obs	_Imputation_	Effect	Estimate
1	1	Intercept	97.2874
2	1	RunTime	-2.9889
3	1	RunPulse	-0.1068
4	2	Intercept	90.9324
5	2	RunTime	-2.9334
6	2	RunPulse	-0.07391

Output 10.3.2. PROC MIXED Covariance Matrices

Covariance Matrices (First Two Imputations)

Obs	_Imputation_	Effect	Col1	Col2	Col3
1	1	Intercept	55.7516	-0.7335	-0.2787
2	1	RunTime	-0.7335	0.1517	-0.00509
3	1	RunPulse	-0.2787	-0.00509	0.001942
4	2	Intercept	37.5576	-0.2597	-0.2044
5	2	RunTime	-0.2597	0.1398	-0.00722
6	2	RunPulse	-0.2044	-0.00722	0.001661

The following statements use the MIANALYZE procedure with PARMS= and COVB= input data sets to produce the same results as in Example 10.2:

```
proc mianalyze parms=mixparms covb=mixcovb edf=28;
   var Intercept RunTime RunPulse;
run;
```

Example 10.4. Reading Generalized Linear Model Results from PARMS= and COVB= Data Sets

This example creates data sets containing parameter estimates and corresponding covariance matrices computed by a generalized linear model analysis for a set of imputed data sets. These estimates are then combined to generate valid statistical inferences about the model parameters.

The following statements use PROC GENMOD to generate the parameter estimates and covariance matrix for each imputed data set:

```
proc genmod data=outmi;
   model Oxygen= RunTime RunPulse/covb;
   by _Imputation_;
   ods output ParameterEstimates=gmparms
              CovB=gmcovb;
run;

proc print data=gmparms (obs=8);
   var _Imputation_ Parameter Estimate;
   title 'GENMOD Model Coefficients (First Two Imputations)';
run;

proc print data=gmcovb (obs=8);
   var _Imputation_ RowName Prm1 Prm2 Prm3;
   title 'GENMOD Covariance Matrices (First Two Imputations)';
run;
```

Output 10.4.1. PROC GENMOD Model Coefficients

GENMOD Model Coefficients (First Two Imputations)

Obs	_Imputation_	Parameter	Estimate
1	1	Intercept	97.2874
2	1	RunTime	-2.9889
3	1	RunPulse	-0.1068
4	1	Scale	2.6227
5	2	Intercept	90.9324
6	2	RunTime	-2.9334
7	2	RunPulse	-0.0739
8	2	Scale	2.4567

The following table displays the covariance matrices for the first two imputed data sets. Note that the GENMOD procedure computes maximum likelihood estimates for the covariance matrix.

Output 10.4.2. PROC GENMOD Covariance Matrices

GENMOD Covariance Matrices (First Two Imputations)

Obs	_Imputation_	Row Name	Prm1	Prm2	Prm3
1	1	Prm1	50.356306	-0.662498	-0.251728
2	1	Prm2	-0.662498	0.1369885	-0.004598
3	1	Prm3	-0.251728	-0.004598	0.0017536
4	1	Scale	-4.14E-16	3.635E-16	-1.53E-17
5	2	Prm1	33.923008	-0.234572	-0.184639
6	2	Prm2	-0.234572	0.1262507	-0.006523
7	2	Prm3	-0.184639	-0.006523	0.0014999
8	2	Scale	1.858E-14	8.886E-16	-1.58E-16

The following statements use the MIANALYZE procedure with PARMS= and COVB= input data sets:

```
proc mianalyze parms=gmparms covb=gmcovb;
   var Intercept RunTime RunPulse;
run;
```

Since the GENMOD procedure computes maximum likelihood estimates for the covariance matrix, the EDF= option is not used. The resulting model coefficients are identical to the estimates from Example 10.2, but the standard errors are slightly different because in this example, maximum likelihood estimates for the standard errors are combined without the EDF= option, whereas in Example 10.2, unbiased estimates for the standard errors are combined with the EDF= option.

Example 10.5. Reading GLM Results from PARMS= and XPXI= Data Sets

This example creates data sets containing parameter estimates and corresponding $(X'X)^{-1}$ matrices computed by a general linear model analysis for a set of imputed data sets. These estimates are then combined to generate valid statistical inferences about the model parameters.

The following statements use PROC GLM to generate the parameter estimates and $(X'X)^{-1}$ matrix for each imputed data set:

```
proc glm data=outmi;
   model Oxygen= RunTime RunPulse/inverse;
   by _Imputation_;
   ods output ParameterEstimates=glmparms
              InvXPX=glmxpxi;
run;

proc print data=glmparms (obs=6);
   var _Imputation_ Parameter Estimate;
   title 'GLM Model Coefficients (First Two Imputations)';
run;

proc print data=glmxpxi (obs=8);
   var _Imputation_ Parameter Intercept RunTime RunPulse;
   title 'GLM X''X Inverse Matrices (First Two Imputations)';
run;
```

Output 10.5.1. PROC GLM Model Coefficients

GLM Model Coefficients (First Two Imputations)

Obs	_Imputation_	Parameter	Estimate
1	1	Intercept	97.28741708
2	1	RunTime	-2.98892274
3	1	RunPulse	-0.10683710
4	2	Intercept	90.93235575
5	2	RunTime	-2.93337517
6	2	RunPulse	-0.07390872

Output 10.5.2. PROC GLM $(X'X)^{-1}$ Matrices

GLM X'X Inverse Matrices (First Two Imputations)

Obs	_Imputation_	Parameter	Intercept	RunTime	RunPulse
1	1	Intercept	7.3205589063	-0.096310799	-0.036595038
2	1	RunTime	-0.096310799	0.0199147383	-0.000668435
3	1	RunPulse	-0.036595038	-0.000668435	0.0002549371
4	1	Oxygen	97.28741708	-2.988922744	-0.106837096
5	2	Intercept	5.620924071	-0.038867743	-0.030594091
6	2	RunTime	-0.038867743	0.0209193054	-0.00108086
7	2	RunPulse	-0.030594091	-0.00108086	0.0002485262
8	2	Oxygen	90.932355745	-2.933375175	-0.073908724

The following statements use the MIANALYZE procedure with PARMS= and XPXI= input data sets to produce the same results as in Example 10.2:

```
proc mianalyze parms=glmparms xpxi=glmxpxi edf=28;
   var Intercept RunTime RunPulse;
run;
```

Example 10.6. Combining Correlation Coefficients

This example combines sample correlation coefficients and associated variances computed from a set of imputed data sets.

The following statements use the CORR procedure to compute the correlation r between variables Oxygen and RunTime for each imputed data set:

```
proc corr data=outmi out=outcorr;
   var Oxygen RunTime;
   by _Imputation_;
run;

proc print data=outcorr (obs=10);
   title 'Correlations (First Two Imputations)';
run;
```

Output 10.6.1. CORR Type Data Set

Correlations (First Two Imputations)

Obs	_Imputation_	_TYPE_	_NAME_	Oxygen	RunTime
1	1	MEAN		47.3458	10.5859
2	1	STD		5.3117	1.3547
3	1	N		31.0000	31.0000
4	1	CORR	Oxygen	1.0000	-0.8338
5	1	CORR	RunTime	-0.8338	1.0000
6	2	MEAN		47.3316	10.6004
7	2	STD		5.3612	1.4336
8	2	N		31.0000	31.0000
9	2	CORR	Oxygen	1.0000	-0.8704
10	2	CORR	RunTime	-0.8704	1.0000

The following statements compute Fisher's Z transformation of r

$$z = \frac{1}{2} log \left(\frac{1+r}{1-r} \right)$$

and the variance estimate corresponding to z, $1/(n-3)$.

```
data ztrans(type=EST);
   set outcorr (drop= RunTime rename= Oxygen= Z);
   if (_type_ = 'N' or _name_ = 'RunTime');

   if (_type_ = 'CORR')  then do;
      _type_= 'PARMS';
      _name_='';
      Z= 0.5 * log((1+Z)/(1-Z));
      end;


else if (_type_ = 'N')  then do;
   _type_='COVB';
   _name_='Z';
   Z= 1. / (Z-3);
   end;
run;

proc print data=ztrans;
   title 'EST Type Data Set with Fisher''s Z Transformation';
run;
```

Output 10.6.2. Fisher's Z Transformation

EST Type Data Set with Fisher's Z Transformation

Obs	_Imputation_	_TYPE_	_NAME_	Z
1	1	COVB	Z	0.03571
2	1	PARMS		-1.20037
3	2	COVB	Z	0.03571
4	2	PARMS		-1.33458
5	3	COVB	Z	0.03571
6	3	PARMS		-1.34657
7	4	COVB	Z	0.03571
8	4	PARMS		-1.20194
9	5	COVB	Z	0.03571
10	5	PARMS		-1.29004

The following statements use the MIANALYZE procedure to generate a combined parameter estimate $\hat{z}$ and variance for Fisher's z. The ODS statement is used to save the parameter estimates in an output data set.

```
proc mianalyze data=ztrans;
   ods output ParmEst=parms;
   var z;
run;
```

Output 10.6.3. Inferences Based on Fisher's Z

The MIANALYZE Procedure

Multiple Imputation Parameter Estimates

Parameter	Estimate	Std Error	95% Confidence Limits		DF
z	-1.274701	0.204097	-1.67720	-0.87220	196.63

Multiple Imputation Parameter Estimates

Parameter	Minimum	Maximum
z	-1.346574	-1.200373

Multiple Imputation Parameter Estimates

Parameter	Theta0	t for H0: Parameter=Theta0	Pr > |t|
z	0	-6.25	<.0001

In addition to the estimate for z, PROC MIANALYZE also generates 95% confidence limits for z, $\hat{z}_{.025}$ and $\hat{z}_{.975}$. An estimate of the correlation coefficient and 95% confidence limits can then be generated from the inverse transformation

$$r = \frac{e^{2z} - 1}{e^{2z} + 1}$$

for $z = \hat{z}$, $\hat{z}_{.025}$, and $\hat{z}_{.975}$.

The following statements print the estimate for z and 95% confidence limits for z:

```
proc print data=parms;
   title 'Parameter Estimates with 95% Confidence Limits';
   var Estimate LCLMean UCLMean;
run;
```

Output 10.6.4. Parameter Estimates with 95% Confidence Limits

Parameter Estimates with 95% Confidence Limits

Obs	Estimate	LCLMean	UCLMean
1	-1.274701	-1.67720	-0.87220

The following statements generate an estimate of the correlation coefficient and 95% confidence limits:

```
data corr_ci;
   set parms;
   r=     (exp(2*Estimate)-1)/(exp(2*Estimate)+1);
   r_low= (exp(2*LCLMean)-1) /(exp(2*LCLMean)+1);
   r_upp= (exp(2*UCLMean)-1) /(exp(2*UCLMean)+1);
;

proc print data=corr_ci;
   title 'Estimated Correlation Coefficient'
         ' with 95% Confidence Limits';
   var r r_low r_upp;
run;
```

Output 10.6.5. Estimated Correlation Coefficient

Estimated Correlation Coefficient with 95% Confidence Limits

Obs	r	r_low	r_upp
1	-0.85507	-0.93250	-0.70249

Example 10.7. Combining Ratios of Variable Means

This example combines ratios of variable means and associated variances computed from imputed data sets.

The following statements use the CORR procedure to compute the means and covariance matrix of variables Oxygen and RunTime for each imputed data set. Within each imputation, the data set is sorted so that observations with _TYPE_='MEAN' and _TYPE_='N' are read before observations with _TYPE_='COV'.

```
proc corr data=outmi cov nocorr noprint out=outcov;
   var RunTime RunPulse;
   by _Imputation_;
run;

proc sort data=outcov;
   by _Imputation_ _name_;
run;

proc print data=outcov (obs=10);
   title 'Means and Covariance Matrices (First Two Imputations)';
run;
```

Output 10.7.1. Means and Covariance Matrices

Means and Covariance Matrices (First Two Imputations)

Obs	_Imputation_	_TYPE_	_NAME_	RunTime	Run Pulse
1	1	MEAN		10.5859	171.301
2	1	STD		1.3547	11.974
3	1	N		31.0000	31.000
4	1	COV	RunPulse	4.8121	143.368
5	1	COV	RunTime	1.8353	4.812
6	2	MEAN		10.6004	169.204
7	2	STD		1.4336	13.153
8	2	N		31.0000	31.000
9	2	COV	RunPulse	8.9385	172.998
10	2	COV	RunTime	2.0553	8.938

For each imputation, the following statements compute a ratio estimate of the means for the variables RunPulse and RunTime, and a corresponding variance estimate:

```
data vratio (type=EST);
   set outcov;
   keep _Imputation_ _type_ _name_ ratio;
   retain TMean PMean N VarP;

   if (_type_ = 'N')  then N= RunTime;

   if (_type_ = 'MEAN')  then do;
      TMean= RunTime;
      PMean= RunPulse;
      Ratio= RunPulse / RunTime;
      _type_= 'PARMS';
      end;

   if (_type_ = 'COV' and _name_ = 'RunPulse')
      then VarP= RunPulse;

   if (_type_ = 'COV' and _name_ = 'RunTime')  then do;
      Ratio= ( PMean**2/TMean**4 * RunTime
                     - 2 * PMean/TMean**3 * RunPulse
                     + (1./ TMean**2) * VarP ) / N;
      _name_= 'Ratio';
      end;

   if ( _type_= 'STD' or _type_= 'N' or _name_= 'RunPulse')
      then delete;
run;

proc print data=vratio;
  title 'EST Type Data Set with Ratios of Variable Means';
run;
```

Output 10.7.2. Ratio of Variable Means

EST Type Data Set with Ratios of Variable Means

Obs	_Imputation_	_TYPE_	_NAME_	Ratio
1	1	PARMS		16.1821
2	1	COV	Ratio	0.1348
3	2	PARMS		15.9620
4	2	COV	Ratio	0.1181
5	3	PARMS		16.0602
6	3	COV	Ratio	0.1376
7	4	PARMS		15.9968
8	4	COV	Ratio	0.1409
9	5	PARMS		16.2961
10	5	COV	Ratio	0.1678

The following statements use the MIANALYZE procedure to generate a combined point estimate for the ratio of variable means and a variance estimate:

```
proc mianalyze data=vratio theta0=1;
   var ratio;
run;
```

Output 10.7.3. Inferences for a Ratio of Variable Means

The MIANALYZE Procedure

Multiple Imputation Parameter Estimates

Parameter	Estimate	Std Error	95% Confidence Limits		DF
ratio	16.099445	0.403440	15.30394	16.89495	201.33

Multiple Imputation Parameter Estimates

Parameter	Minimum	Maximum
ratio	15.961997	16.296124

Multiple Imputation Parameter Estimates

Parameter	Theta0	t for H0: Parameter=Theta0	Pr > |t|
ratio	1.000000	37.43	<.0001

The variable RunTime has an estimated mean of about 10.6 and a standard error of about 1.4 in each imputed data set. An estimated coefficient of variation is $(1.4/\sqrt{31})/10.6 = 2.37\%$. Since the coefficient of variation is less than 10%, the ratio of sample means is approximately unbiased and normally distributed.

References

Barnard, J. and Rubin, D.B. (1999), "Small-Sample Degrees of Freedom with Multiple Imputation," *Biometrika*, 86, 948–955.

Cochran, W.J. (1977), *Sampling Techniques*, Second Edition, New York: John Wiley & Sons, Inc.

Li, K.H., Raghunathan, T.E., and Rubin, D.B. (1991), "Large-Sample Significance Levels from Multiply Imputed Data Using Moment-Based Statistics and an F Reference Distribution," *Journal of the American Statistical Association*, 86, 1065–1073.

Rubin, D.B. (1976), "Inference and Missing Data," *Biometrika*, 63, 581–592.

Rubin, D.B. (1987), *Multiple Imputation for Nonresponse in Surveys*, New York: John Wiley & Sons, Inc.

Rubin, D.B. (1996), "Multiple Imputation After 18+ Years," *Journal of the American Statistical Association*, 91, 473–489.

SAS Institute Inc. (1999), *SAS Language Reference: Concepts, Version 8*, Cary, NC: SAS Institute Inc.

SAS Institute Inc. (1999), *SAS Procedures Guide, Version 8*, Cary, NC: SAS Institute Inc.

SAS Institute Inc. (1999), *SAS/STAT User's Guide, Version 8*, Cary, NC: SAS Institute Inc.

Schafer, J.L. (1997), *Analysis of Incomplete Multivariate Data*, New York: Chapman and Hall.

Chapter 11
The NPAR1WAY Procedure

Chapter Table of Contents

Chapter 11
The NPAR1WAY Procedure

Overview

The new KS option in the EXACT statement provides the exact Kolmogorov-Smirnov test for two-sample data. The new POINT option in the EXACT statement provides exact point probabilities for the test statistics.

Syntax

EXACT Statement

EXACT *statistic-options* < / *computation-options* > **;**

The EXACT statement requests exact tests for the specified statistics. Optionally, PROC NPAR1WAY computes Monte Carlo estimates of the exact *p*-values. The *statistic-options* specify the statistics for which to provide exact tests. The *computation-options* specify options for the computation of exact statistics.

The following new *statistic-option* is now available in the EXACT statement:

KS
requests the exact Kolmogorov-Smirnov test for two-sample data.

The following new *computation-option* is now available in the EXACT statement after the slash (/):

POINT
requests exact point probabilities for the test statistics.

The POINT option is available for all EXACT statement *statistic-options*. The POINT option is not available with the MC option, which provides Monte Carlo estimation of exact *p*-values instead of direct exact *p*-value computation.

Chapter 12
The PROBIT Procedure

Chapter Table of Contents

Chapter 12
The PROBIT Procedure

Overview

You can specify interaction terms among any of the specified explanatory variables in the MODEL statement in the same way as in the GLM procedure. Because of the more complicated models that you can specify, only a single MODEL statement is now allowed for each PROC PROBIT statement. If more than one MODEL statement is specified, only the last one is used.

You can now graphically display results of model parameter estimation. For binary data analysis, you can construct plots of predicted probabilities, inverse predicted probabilities, and cumulative probabilities. For ordinal multinomial models, you can display plots of predicted probabilitites, linear predictors, and cumulative probabilities. In addition, you can control graphical features such as plot layout, colors, plotting symbols, line styles, and the text fonts used in the plots.

You can use the new XDATA= data set to provide the values for effects other than the single continuous dosage effect when the predicted dosage and their fiducial limits are computed. These specified values can also be used for generating plots.

You can use the new INEST= data set to provide the initial values for parameters in the model. The structure of this data set is similar to that of the OUTEST= data set. This data set will overwrite the original INITIAL= option in the MODEL statement.

The OUTEST= data set is extended to both binomial and multinomial models. It also generates the OUTEST data set whether there is a CLASS statement or not. The OUTEST= data set does not include a fixed threshold.

You can use the new AGGREGATE= option in the MODEL statement to define the subpopulations for calculating the Pearson chi-square statistic and the deviance.

Syntax

PROC PROBIT Statement

PROC PROBIT < *options* > **;**

GOUT=*graphics-catalog*
specifies a graphics catalog in which to save graphics output.

INEST= *SAS-data-set*
specifies an input SAS data set that contains initial estimates for all the parameters

in the model. See the section "INEST= *SAS-data-set*" on page 277 for a detailed description of the contents of the INEST= data set.

NAMELEN=*n*
specifies the length of effect names in tables and output data sets to be n characters, where n is a value between 20 and 200. The default length is 20 characters.

XDATA= *SAS-data-set*
specifies an input SAS data set that contains values for all the independent variables in the MODEL statement and variables in the CLASS statement. If there are covariates specified in a MODEL statement, you specify fixed values for the effects in the MODEL statement by the XDATA= data set when predicted values and/or fiducial limits for a single continuous variable (dose variable) are required. These specified values for the effects in the MODEL statement are also used for generating plots. See the section "XDATA= *SAS-data-set*" on page 280 for a detailed description of the contents of the XDATA= data set.

CDFPLOT Statement

CDFPLOT <*var = variable*> <*options* >**;**

The CDFPLOT statement plots the predicted cumulative distribution function (CDF) of the multinomial response variable as a function of a single continuous independent variable (dose variable). You can only use this statement after a multinomial model statement.

VAR= *(variable)*
specifies a single continuous variable (dose variable) in the independent variable list of the MODEL statement. If a VAR= variable is not specified, the first single continuous variable in the independent variable list of the MODEL statement is used. If such a variable does not exist in the independent variable list of the MODEL statement, an error is reported.

The predicted cumulative distribution function is defined as

$$\hat{F}_j(\mathbf{x}) = C + (1 - C)F(\hat{a}_j + \mathbf{x}'\hat{\mathbf{b}})$$

where $j = 1, \ldots, k$ are the indexes of the k levels of the multinomial response variable, *F* is the CDF of the distribution used to model the cumulative probabilities, $\hat{\mathbf{b}}$ is the vector of estimated parameters, $\mathbf{x}$ is the covariate vector, $\hat{a}_j$ are estimated ordinal intercepts with $\hat{a}_1 = 0$, and C is the threshold parameter, either known or estimated from the model. Let x_1 be the covariate corresponding to the dose variable and $\mathbf{x}_{-1}$ be the vector of the rest of the covariates. Let the corresponding estimated parameters be $\hat{b}_1$ and $\hat{\mathbf{b}}_{-1}$. Then

$$\hat{F}_j(\mathbf{x}) = C + (1 - C)F(\hat{a}_j + x_1\hat{b}_1 + \mathbf{x}'_{-1}\hat{\mathbf{b}}_{-1})$$

To plot $\hat{F}_j$ as a function of x_1, $\mathbf{x}_{-1}$ must be specified. You can use the XDATA= option to provide the values of $\mathbf{x}_{-1}$ (see the XDATA= option in the PROC PROBIT statement for details), or use the default values that follow the rules:

- If the effect contains a continuous variable (or variables), the overall mean of this effect is used.
- If the effect is a single classification variable, the highest level of the variable is used.

options

specify the levels of the multinomial response variable for which the cdf curves are requested, and add features to the plot. There are $k-1$ curves for a k-level multinomial response variable (for the highest level, it is the constant line 1). You can specify any of them to be plotted by the LEVEL= option in the CDFPLOT statement. See the LEVEL= option for how to specify the levels.

An attached box on the right side of the plot is used to label these curves with the names of their levels. You can specify the color of this box using the CLABBOX= option.

You can use options in the CDFPLOT statement to

- superimpose specification limits
- specify the levels for which the cdf curves are requested
- specify graphical enhancements (such as color or text height)

Summary of Options

The following tables list all *options* by function. The "Dictionary of Options" on page 246 describes each option in detail.

CDF Options

Table 12.1. Options for CDFPLOT

LEVEL=*character-list*	specifies the names of the levels for which the cdf curves are requested
NOTHRESH	suppresses the threshold line
THRESHLABPOS=*value*	specifies the position for the label of the threshold line

General Options

Table 12.2. Color Options

CAXIS=*color*	specifies color for axis
CFIT=*color*	specifies color for fitted curves
CFRAME=*color*	specifies color for frame
CGRID=*color*	specifies color for grid lines
CHREF=*color*	specifies color for HREF= lines
CLABBOX=*color*	specifies color for label box
CTEXT=*color*	specifies color for text
CVREF=*color*	specifies color for VREF= lines

Table 12.3. Options to Enhance Plots Produced on Graphics Devices

ANNOTATE= *SAS-data-set*	specifies an ANNOTATE data set
INBORDER	requests a border around plot
LFIT=*linetype*	specifies line style for fitted curves
LGRID=*linetype*	specifies line style for grid lines
NOFRAME	suppresses the frame around plotting areas
NOGRID	suppresses grid lines
NOFIT	suppresses cdf curves
NOHLABEL	suppresses horizontal labels
NOHTICK	suppresses horizontal ticks
NOVTICK	suppresses vertical ticks
TURNVLABELS	vertically string out characters in vertical labels
WFIT=*n*	specifies thickness for fitted curves
WGRID=*n*	specifies thickness for grids
WREFL=*n*	specifies thickness for reference lines

Table 12.4. Axis Options

HAXIS=*value1 to value2 <by value3>*	specifies tick mark values for horizontal axis
HOFFSET=*value*	specifies offset for horizontal axis
HLOWER=*value*	specifies lower limit on horizontal axis scale
HUPPER=*value*	specifies upper limit on horizontal axis scale
NHTICK=*n*	specifies number of ticks for horizontal axis
NVTICK=*n*	specifies number of ticks for vertical axis
VAXIS=*value1 to value2 <by value3>*	specifies tick mark values for vertical axis
VAXISLABEL=*'label'*	specifies label for vertical axis
VOFFSET=*value*	specifies offset for vertical axis
VLOWER=*value*	specifies lower limit on vertical axis scale
VUPPER=*value*	specifies upper limit on vertical axis scale
WAXIS=*n*	specifies thickness for axis

Table 12.5. Graphics Catalog Options

DESCRIPTION=*'string'*	specifies description for graphics catalog member
NAME=*'string'*	specifies name for plot in graphics catalog

Table 12.6. Options for Text Enhancement

FONT=*font*	specifies software font for text
HEIGHT=*value*	specifies height of text used outside framed areas
INFONT=*font*	specifies software font for text inside framed areas
INHEIGHT=*value*	specifies height of text inside framed areas

Table 12.7. Options for Reference Lines

HREF< (INTERSECT)> =value-list	requests horizontal reference line
HREFLABELS= ('label1',. . .,'labeln')	specifies labels for HREF= lines
HREFLABPOS=n	specifies vertical position of labels for HREF= lines
LHREF=*linetype*	specifies line style for HREF= lines
LVREF=*linetype*	specifies line style for VREF= lines
VREF<(INTERSECT)> =value-list	requests vertical reference line
VREFLABELS= ('label1',. . .,'labeln')	specifies labels for VREF= lines
VREFLABPOS=n	specifies horizontal position of labels for VREF= lines

Dictionary of Options

The following entries provide detailed descriptions of the *options* in the CDFPLOT statement.

ANNOTATE=*SAS-data-set*
ANNO=*SAS-data-set*
specifies an ANNOTATE data set, as described in *SAS/GRAPH Software: Reference*, that enables you to add features to the cdf plot. The ANNOTATE= data set you specify in the CDFPLOT statement is used for all plots created by the statement.

CAXIS=*color*
CAXES=*color*
specifies the color used for the axes and tick marks. This option overrides any COLOR= specifications in an AXIS statement. The default is the first color in the device color list.

CFIT=*color*
specifies the color for the fitted cdf curves. The default is the first color in the device color list.

CFRAME=*color*
CFR=*color*
specifies the color for the area enclosed by the axes and frame. This area is not shaded by default.

CGRID=*color*
specifies the color for grid lines. The default is the first color in the device color list.

CLABBOX=*color*
specifies the color for the area enclosed by the label box for cdf curves. This area is not shaded by default.

CHREF=*color*
CH=*color*
specifies the color for lines requested by the HREF= option. The default is the first color in the device color list.

CTEXT=*color*
specifies the color for tick mark values and axis labels. The default is the color specified for the CTEXT= option in the most recent GOPTIONS statement.

CVREF=*color*
CV=*color*
specifies the color for lines requested by the VREF= option. The default is the first color in the device color list.

DESCRIPTION='*string***'**
DES='*string***'**
specifies a description, up to 40 characters, that appears in the PROC GREPLAY master menu. The default is the variable name.

FONT=*font*
specifies a software font for reference line and axis labels. You can also specify fonts for axis labels in an AXIS statement. The FONT= font takes precedence over the FTEXT= font specified in the most recent GOPTIONS statement. Hardware characters are used by default.

HAXIS=*value1 to value2***<***by value3***>**
specifies tick mark values for the horizontal axis. *value1*, *value2*, and *value3* must be numeric, and *value1* must be less than *value2*. The lower tick mark is *value1*. Tick marks are drawn at increments of *value3*. The last tick mark is the greatest value that does not exceed *value2*. If *value3* is omitted, a value of 1 is used.

Examples of HAXIS= lists are:

```
haxis = 0 to 10
haxis = 2 to 10 by 2
haxis = 0 to 200 by 10
```

HEIGHT=*value*
specifies the height of text used outside framed areas. The default value is 3.846 (in percentage).

HLOWER=*value*
specifies the lower limit on the horizontal axis scale. The HLOWER= option specifies *value* as the lower horizontal axis tick mark. The tick mark interval and the upper axis limit are determined automatically. This option has no effect if the HAXIS= option is used.

HOFFSET=*value*
specifies offset for horizontal axis. The default value is 1.

HUPPER=*value*
specifies *value* as the upper horizontal axis tick mark. The tick mark interval and the lower axis limit are determined automatically. This option has no effect if the HAXIS= option is used.

HREF <(INTERSECT)> =*value-list*
requests reference lines perpendicular to the horizontal axis. If (INTERSECT) is specified, a second reference line perpendicular to the vertical axis is drawn that intersects the fit line at the same point as the horizontal axis reference line. If a horizontal axis reference line label is specified, the intersecting vertical axis reference line is labeled with the vertical axis value. See also the CHREF=, HREFLABELS=, and LHREF= options.

HREFLABELS=*'label1'*,. . .,*'labeln'*
HREFLABEL=*'label1'*,. . .,*'labeln'*
HREFLAB=*'label1'*,. . .,*'labeln'*
specifies labels for the lines requested by the HREF= option. The number of labels must equal the number of lines. Enclose each label in quotes. Labels can be up to 16 characters.

HREFLABPOS=n

specifies the vertical position of labels for HREF= lines. The following table shows valid values for *n* and the corresponding label placements.

n	**label placement**
1	top
2	staggered from top
3	bottom
4	staggered from bottom
5	alternating from top
6	alternating from bottom

INBORDER

requests a border around cdf plots.

LEVEL= (*character-list***)**
ORDINAL= (*character-list***)**

specifies the names of the levels for which cdf curves are requested. Names should be quoted and separated by space. If there is no correct name provided, no cdf curve is plotted.

LFIT=*linetype*

specifies a line style for fitted curves. By default, fitted curves are drawn by connecting solid lines (*linetype = 1*).

LGRID=*linetype*

specifies a line style for all grid lines. *linetype* is between 1 and 46. The default is 35.

LHREF=*linetype*
LH=*linetype*

specifies the line type for lines requested by the HREF= option. The default is 2, which produces a dashed line.

LVREF=*linetype*
LV=*linetype*

specifies the line type for lines requested by the VREF= option. The default is 2, which produces a dashed line.

NAME=*'string'*

specifies a name for the plot, up to eight characters, that appears in the PROC GREPLAY master menu. The default is 'PROBIT'.

NOFIT

suppresses the fitted cdf curves.

NOFRAME

suppresses the frame around plotting areas.

NOGRID

suppresses grid lines.

NOHLABEL

suppresses horizontal labels.

NOHTICK
suppresses horizontal tick marks.

NOTHRESH
suppresses the threshold line.

NOVLABEL
suppresses vertical labels.

NOVTICK
suppresses vertical tick marks.

THRESHLABPOS=*n*
specifies the horizontal position of labels for the threshold line. The following table shows valid values for *n* and the corresponding label placements.

n	**label placement**
1	left
2	right

VAXIS=*value1 to value2***<***by value3***>**
specifies tick mark values for the vertical axis. *value1*, *value2*, and *value3* must be numeric, and *value1* must be less than *value2*. The lower tick mark is *value1*. Tick marks are drawn at increments of *value3*. The last tick mark is the greatest value that does not exceed *value2*. This method of specification of tick marks is not valid for logarithmic axes. If *value3* is omitted, a value of 1 is used.

Examples of VAXIS= lists are:

```
vaxis = 0 to 10
vaxis = 0 to 2 by .1
```

VAXISLABEL=*'string'*
specifies a label for the vertical axis.

VLOWER=*value*
specifies the lower limit on the vertical axis scale. The VLOWER= option specifies *value* as the lower vertical axis tick mark. The tick mark interval and the upper axis limit are determined automatically. This option has no effect if the VAXIS= option is used.

VREF=*value-list*
requests reference lines perpendicular to the vertical axis. If (INTERSECT) is specified, a second reference line perpendicular to the horizontal axis is drawn that intersects the fit line at the same point as the vertical axis reference line. If a vertical axis reference line label is specified, the intersecting horizontal axis reference line is labeled with the horizontal axis value. See also the CVREF=, LVREF=, and VREFLABELS= options.

VREFLABELS=*'label1',...,'labeln'*
VREFLABEL=*'label1',...,'labeln'*
VREFLAB=*'label1',...,'labeln'*
specifies labels for the lines requested by the VREF= option. The number of labels

must equal the number of lines. Enclose each label in quotes. Labels can be up to 16 characters.

VREFLABPOS=*n*
specifies the horizontal position of labels for VREF= lines. The following table shows valid values for *n* and the corresponding label placements.

n	**label placement**
1	left
2	right

VUPPER=*value*
specifies the upper limit on the vertical axis scale. The VUPPER= option specifies *value* as the upper vertical axis tick mark. The tick mark interval and the lower axis limit are determined automatically. This option has no effect if the VAXIS= option is used.

WAXIS=*n*
specifies line thickness for axes and frame. The default value is 1.

WFIT=*n*
specifies line thickness for fitted curves. The default value is 1.

WGRID=*n*
specifies line thickness for grids. The default value is 1.

WREFL=*n*
specifies line thickness for reference lines. The default value is 1.

INSET Statement

INSET <*keyword-list*> < / *options*>**;**

The box or table of summary information produced on plots made with the CDFPLOT, IPPPLOT, LPREDPLOT, and PREDPPLOT statement is called an *inset*. You can use the INSET statement to customize both the information that is printed in the inset box and the appearance of the inset box. To supply the information that is displayed in the inset box, you specify *keywords* corresponding to the information you want shown. For example, the following statements produce a predicted probability plot with the number of trials, the number of events, the name of the distribution, and the estimated optimum natural threshold in the inset.

```
proc probit data=epidemic;
   model r/n = dose;
   predpplot ;
   inset nobs ntrials nevents dist optc;
run;
```

By default, inset entries are identified with appropriate labels. However, you can provide a customized label by specifying the *keyword* for that entry followed by the

equal sign (=) and the label in quotes. For example, the following INSET statement produces an inset containing the number of observations and the name of the distribution, labeled "Sample Size" and "Distribution" in the inset.

```
inset nobs='Sample Size' dist='Distribution';
```

If you specify a keyword that does not apply to the plot you are creating, then the keyword is ignored.

The *options* control the appearance of the box.

If you specify more than one INSET statement, only the first one is used.

Keywords Used in the INSET Statement

The following tables list keywords available in the INSET statement to display summary statistics, distribution parameters, and distribution fitting information.

Table 12.8. Summary Statistics

NOBS	number of observations
NTRIALS	number of trials
NEVENTS	number of events
C	the user inputted threshold
OPTC	the estimated natural threshold
NRESPLEV	number of levels of the response variable

Table 12.9. General Information

CONFIDENCE	confidence coefficient for all confidence intervals or for the Weibayes fit
DIST	name of the distribution

Options Used in the INSET Statement

The following tables list the options available in the INSET statement.

Table 12.10. General Appearance Options

FONT=*font*	specifies software font for text
HEIGHT=*value*	specifies height of text
HEADER=*'quoted string'*	specifies text for header or box title
NOFRAME	omits frame around box
POS= *value* <DATA | PERCENT>	determines the position of the inset. The *value* can be a compass point (N, NE, E, SE, S, SW, W, NW) or a pair of coordinates (x, y) enclosed in parentheses. The coordinates can be specified in axis percent units or axis data units.
REFPOINT= *name*	specifies the reference point for an inset that is positioned by a pair of coordinates with the POS= option. You use the REFPOINT= option in conjunction with the POS= coordinates. The REFPOINT= option specifies which corner of the inset frame you have specified with coordinates (x, y) and it can take the value of BR (bottom right), BL (bottom left), TR (top right), or TL (top left). The default is REFPOINT=BL. If the inset position is specified as a compass point, then the REFPOINT= option is ignored.

Table 12.11. Color and Pattern Options

CFILL=*color*	color for filling box
CFILLH=*color*	color for filling box header
CFRAME=*color*	color for frame
CHEADER=*color*	color for text in header
CTEXT=*color*	color for text

IPPPLOT Statement

IPPPLOT <*var = variable*> <*options*>**;**

The IPPPLOT statement plots the inverse of the predicted probability against a single continuous variable (dose variable) in the MODEL statement for the binomial model. You can only use this statement after a binomial model statement. The confidence limits for the predicted values of the dose variable are the computed fiducial limits, not the inverse of the confidence limits of the predicted probabilities. Refer to the section "Inverse Confidence Limits" on page 278 for more detail.

VAR= *(variable)*
specifies a single continuous variable (dose variable) in the independent variable list of the MODEL statement. If a VAR= variable is not specified, the first single continuous variable in the independent variable list of the MODEL statement is used. If such a variable does not exist in the independent variable list of the MODEL statement, an error is reported.

For the binomial model, the response variable is a probability. An estimate of the dose level $\hat{x}_1$ needed for a response of p is given by

$$\hat{x}_1 = (F^{-1}(p) - \mathbf{x}_{-1}^{'}\hat{\mathbf{b}}_{-1})/\hat{b}_1$$

where F is the cumulative distribution function used to model the probability, $\mathbf{x}_{-1}$ is the vector of the rest of the covariates, $\hat{\mathbf{b}}_{-1}$ is the vector of the estimated parameters corresponding to $\mathbf{x}_{-1}$, and $\hat{b}_1$ is the estimated parameter for the dose variable of interest.

To plot $\hat{x}_1$ as a function of p, $\mathbf{x}_{-1}$ must be specified. You can use the XDATA= option to provide the values of $\mathbf{x}_{-1}$ (see the XDATA= option in the PROC PROBIT statement for details), or use the default values that follow the rules:

- If the effect contains a continuous variable (or variables), the overall mean of this effect is used.
- If the effect is a single classification variable, the highest level of the variable is used.

options

add features to the plot.

You can use options in the IPPPLOT statement to

- superimpose specification limits
- suppress or add the observed data points on the plot
- suppress or add the fiducial limits on the plot
- specify graphical enhancements (such as color or text height)

Summary of Options

The following tables list all *options* by function. The "Dictionary of Options" on page 256 describes each option in detail.

IPP Options

Table 12.12. Plot Layout Options for IPPPLOT

NOCONF	suppresses fiducial limits
NODATA	suppresses observed data points on the plot
NOTHRESH	suppresses the threshold line
THRESHLABPOS=*value*	specifies the position for the label of the threshold line

General Options

Table 12.13. Color Options

CAXIS=*color*	specifies color for axis
CFIT=*color*	specifies color for fitted curves
CFRAME=*color*	specifies color for frame
CGRID=*color*	specifies color for grid lines
CHREF=*color*	specifies color for HREF= lines
CTEXT=*color*	specifies color for text
CVREF=*color*	specifies color for VREF= lines

Table 12.14. Options to Enhance Plots Produced on Graphics Devices

ANNOTATE= *SAS-data-set*	specifies an ANNOTATE data set
INBORDER	requests a border around plot
LFIT=*linetype*	specifies line style for fitted curves and confidence limits
LGRID=*linetype*	specifies line style for grid lines
NOFRAME	suppresses the frame around plotting areas
NOGRID	suppresses grid lines
NOFIT	suppresses fitted curves
NOHLABEL	suppresses horizontal labels
NOHTICK	suppresses horizontal ticks
NOVTICK	suppresses vertical ticks
TURNVLABELS	vertically string out characters in vertical labels
WFIT=*n*	specifies thickness for fitted curves
WGRID=*n*	specifies thickness for grids
WREFL=*n*	specifies thickness for reference lines

Table 12.15. Axis Options

HAXIS=*value1 to value2 <by value3>*	specifies tick mark values for horizontal axis
HOFFSET=*value*	specifies offset for horizontal axis
HLOWER=*value*	specifies lower limit on horizontal axis scale
HUPPER=*value*	specifies upper limit on horizontal axis scale
NHTICK=*n*	specifies number of ticks for horizontal axis
NVTICK=*n*	specifies number of ticks for vertical axis
VAXIS=*value1 to value2 <by value3>*	specifies tick mark values for vertical axis
VAXISLABEL=*'label'*	specifies label for vertical axis
VOFFSET=*value*	specifies offset for vertical axis
VLOWER=*value*	specifies lower limit on vertical axis scale
VUPPER=*value*	specifies upper limit on vertical axis scale
WAXIS=*n*	specifies thickness for axis

Table 12.16. Options for Reference Lines

HREF<(INTERSECT)> =value-list	requests horizontal reference line
HREFLABELS= ('label1',. . .,'labeln')	specifies labels for HREF= lines
HREFLABPOS=n	specifies vertical position of labels for HREF= lines
LHREF=*linetype*	specifies line style for HREF= lines
LVREF=*linetype*	specifies line style for VREF= lines
VREF<(INTERSECT)> =value-list	requests vertical reference line
VREFLABELS= ('label1',. . .,'labeln')	specifies labels for VREF= lines
VREFLABPOS=n	specifies horizontal position of labels for VREF= lines

Table 12.17. Graphics Catalog Options

DESCRIPTION=*'string'*	specifies description for graphics catalog member
NAME=*'string'*	specifies name for plot in graphics catalog

Table 12.18. Options for Text Enhancement

FONT=*font*	specifies software font for text
HEIGHT=*value*	specifies height of text used outside framed areas
INFONT=*font*	specifies software font for text inside framed areas
INHEIGHT=*value*	specifies height of text inside framed areas

Dictionary of Options

The following entries provide detailed descriptions of the *options* in the IPPPLOT statement.

ANNOTATE=*SAS-data-set*
ANNO=*SAS-data-set*
specifies an ANNOTATE data set, as described in *SAS/GRAPH Software: Reference*, that enables you to add features to the ipp plot. The ANNOTATE= data set you specify in the IPPPLOT statement is used for all plots created by the statement.

CAXIS=*color*
CAXES=*color*
specifies the color used for the axes and tick marks. This option overrides any COLOR= specifications in an AXIS statement. The default is the first color in the device color list.

CFIT=*color*
specifies the color for the fitted ipp curves The default is the first color in the device color list.

CFRAME=*color*
CFR=*color*
specifies the color for the area enclosed by the axes and frame. This area is not shaded by default.

CGRID=*color*
specifies the color for grid lines. The default is the first color in the device color list.

CHREF=*color*
CH=*color*
specifies the color for lines requested by the HREF= option. The default is the first color in the device color list.

CTEXT=*color*
specifies the color for tick mark values and axis labels. The default is the color specified for the CTEXT= option in the most recent GOPTIONS statement.

CVREF=*color*
CV=*color*
specifies the color for lines requested by the VREF= option. The default is the first color in the device color list.

DESCRIPTION='*string***'**
DES='*string***'**
specifies a description, up to 40 characters, that appears in the PROC GREPLAY master menu. The default is the variable name.

FONT=*font*
specifies a software font for reference line and axis labels. You can also specify fonts for axis labels in an AXIS statement. The FONT= font takes precedence over the FTEXT= font specified in the most recent GOPTIONS statement. Hardware characters are used by default.

HAXIS=*value1 to value2***<***by value3***>**
specifies tick mark values for the horizontal axis. *value1*, *value2*, and *value3* must be numeric, and *value1* must be less than *value2*. The lower tick mark is *value1*. Tick marks are drawn at increments of *value3*. The last tick mark is the greatest value that does not exceed *value2*. If *value3* is omitted, a value of 1 is used.

Examples of HAXIS= lists are:

```
haxis = 0 to 10
haxis = 2 to 10 by 2
haxis = 0 to 200 by 10
```

HEIGHT=*value*
specifies the height of text used outside framed areas. The default value is 3.846 (in percentage).

HLOWER=*value*
specifies the lower limit on the horizontal axis scale. The HLOWER= option specifies *value* as the lower horizontal axis tick mark. The tick mark interval and the upper axis limit are determined automatically. This option has no effect if the HAXIS= option is used.

HOFFSET=*value*
specifies offset for horizontal axis. The default value is 1.

HUPPER=*value*
specifies *value* as the upper horizontal axis tick mark. The tick mark interval and the lower axis limit are determined automatically. This option has no effect if the HAXIS= option is used.

HREF < (INTERSECT) > =*value-list*
requests reference lines perpendicular to the horizontal axis. If (INTERSECT) is specified, a second reference line perpendicular to the vertical axis is drawn that intersects the fit line at the same point as the horizontal axis reference line. If a horizontal axis reference line label is specified, the intersecting vertical axis reference line is labeled with the vertical axis value. See also the CHREF=, HREFLABELS=, and LHREF= options.

HREFLABELS=*'label1'***,**. . .**,***'labeln'*
HREFLABEL=*'label1'***,**. . .**,***'labeln'*
HREFLAB=*'label1'***,**. . .**,***'labeln'*
specifies labels for the lines requested by the HREF= option. The number of labels must equal the number of lines. Enclose each label in quotes. Labels can be up to 16 characters.

HREFLABPOS=*n*
specifies the vertical position of labels for HREF= lines. The following table shows valid values for *n* and the corresponding label placements.

n	**label placement**
1	top
2	staggered from top
3	bottom
4	staggered from bottom
5	alternating from top
6	alternating from bottom

INBORDER
requests a border around ipp plots.

LFIT=*linetype*
specifies a line style for fitted curves and confidence limits. By default, fitted curves are drawn by connecting solid lines (*linetype = 1*) and confidence limits are drawn by connecting dashed lines (*linetype = 3*).

LGRID=*linetype*
specifies a line style for all grid lines. *linetype* is between 1 and 46. The default is 35.

LHREF=*linetype*
LH=*linetype*
specifies the line type for lines requested by the HREF= option. The default is 2, which produces a dashed line.

LVREF=*linetype*
LV=*linetype*
specifies the line type for lines requested by the VREF= option. The default is 2, which produces a dashed line.

NAME=*'string'*
specifies a name for the plot, up to eight characters, that appears in the PROC GREPLAY master menu. The default is 'PROBIT'.

NOCONF
suppresses fiducial limits from the plot.

NODATA
suppresses observed data points from the plot.

NOFIT
suppresses the fitted ipp curves.

NOFRAME
suppresses the frame around plotting areas.

NOGRID
suppresses grid lines.

NOHLABEL
suppresses horizontal labels.

NOHTICK
suppresses horizontal tick marks.

NOTHRESH
suppresses the threshold line.

NOVLABEL
suppresses vertical labels.

NOVTICK
suppresses vertical tick marks.

THRESHLABPOS=*n*
specifies the vertical position of labels for the threshold line. The following table shows valid values for *n* and the corresponding label placements.

n	**label placement**
1	top
2	bottom

VAXIS=*value1 to value2***<***by value3***>**
specifies tick mark values for the vertical axis. *value1*, *value2*, and *value3* must be numeric, and *value1* must be less than *value2*. The lower tick mark is *value1*. Tick marks are drawn at increments of *value3*. The last tick mark is the greatest value that does not exceed *value2*. This method of specification of tick marks is not valid for logarithmic axes. If *value3* is omitted, a value of 1 is used.

Examples of VAXIS= lists are:

```
vaxis = 0 to 10
vaxis = 0 to 2 by .1
```

VAXISLABEL=*'string'*
specifies a label for the vertical axis.

VLOWER=*value*
specifies the lower limit on the vertical axis scale. The VLOWER= option specifies *value* as the lower vertical axis tick mark. The tick mark interval and the upper axis limit are determined automatically. This option has no effect if the VAXIS= option is used.

VREF=*value-list*
requests reference lines perpendicular to the vertical axis. If (INTERSECT) is specified, a second reference line perpendicular to the horizontal axis is drawn that intersects the fit line at the same point as the vertical axis reference line. If a vertical axis reference line label is specified, the intersecting horizontal axis reference line is labeled with the horizontal axis value. See also the CVREF=, LVREF=, and VREFLABELS= options.

VREFLABELS=*'label1',. . .,'labeln'*
VREFLABEL=*'label1',. . .,'labeln'*
VREFLAB=*'label1',. . .,'labeln'*
specifies labels for the lines requested by the VREF= option. The number of labels must equal the number of lines. Enclose each label in quotes. Labels can be up to 16 characters.

VREFLABPOS=n
specifies the horizontal position of labels for VREF= lines. The following table shows valid values for *n* and the corresponding label placements.

n	**label placement**
1	left
2	right

VUPPER=*value*
specifies the upper limit on the vertical axis scale. The VUPPER= option specifies *value* as the upper vertical axis tick mark. The tick mark interval and the lower axis limit are determined automatically. This option has no effect if the VAXIS= option is used.

WAXIS=n
specifies line thickness for axes and frame. The default value is 1.

WFIT=n
specifies line thickness for fitted curves. The default value is 1.

WGRID=n
specifies line thickness for grids. The default value is 1.

WREFL=n
specifies line thickness for reference lines. The default value is 1.

LPREDPLOT Statement

LPREDPLOT <*var = variable*> <*options*>**;**

The LPREDPLOT statement plots the linear predictor $\mathbf{x}'\mathbf{b}$ against a single continuous variable (dose variable) in the MODEL statement for either the binomial model or the multinomial model. The confidence limits for the predicted values are only available for the binomial model.

VAR= *(variable)*
specifies a single continuous variable (dose variable) in the independent variable list of the MODEL statement for which the linear predictor plot is plotted. If a VAR= variable is not specified, the first single continuous variable in the independent variable list of the MODEL statement is used. If such a variable does not exist in the independent variable list of the MODEL statement, an error is reported.

Let x_1 be the covariate of the dose variable, $\mathbf{x}_{-1}$ be the vector of the rest of the covariates, $\hat{\mathbf{b}}_{-1}$ be the vector of estimated parameters corresponding to $\mathbf{x}_{-1}$, and $\hat{b}_1$ be the estimated parameter for the dose variable of interest.

To plot $\hat{\mathbf{x}}'\mathbf{b}$ as a function of x_1, $\mathbf{x}_{-1}$ must be specified. You can use the XDATA= option to provide the values of $\mathbf{x}_{-1}$ (see the XDATA= option in the PROC PROBIT statement for details), or use the default values that follow the rules:

- If the effect contains a continuous variable (or variables), the overall mean of this effect is used.

- If the effect is a single classification variable, the highest level of the variable is used.

options

add features to the plot.

For the multinomial model, you can use the LEVEL= option to specify the levels for which the linear predictor lines are plotted. The lines are labeled by the names of their levels in the middle.

You can use options in the LPREDPLOT statement to

- superimpose specification limits
- suppress or add the observed data points on the plot for the binomial model
- suppress or add the confidence limits for the binomial model
- specify the levels for which the linear predictor lines are requested for the multinomial model
- specify graphical enhancements (such as color or text height)

Summary of Options

The following tables list all *options* by function. The "Dictionary of Options" on page 263 describes each option in detail.

LPRED Options

Table 12.19. Plot Layout Options for LPREDPLOT

LEVEL=*character-list*	specifies the names of the levels for which the linear predictor lines are requested (only for the multinomial model)
NOCONF	suppresses fiducial limits (only for the binomial model)
NODATA	suppresses observed data points on the plot (only for the binomial model)
NOTHRESH	suppresses the threshold line
THRESHLABPOS=*value*	specifies the position for the label of the threshold line

General Options

Table 12.20. Color Options

CAXIS=*color*	specifies color for axis
CFIT=*color*	specifies color for fitted curves
CFRAME=*color*	specifies color for frame
CGRID=*color*	specifies color for grid lines
CHREF=*color*	specifies color for HREF= lines
CTEXT=*color*	specifies color for text
CVREF=*color*	specifies color for VREF= lines

Table 12.21. Options to Enhance Plots Produced on Graphics Devices

ANNOTATE=*SAS-data-set*	specifies an ANNOTATE data set
INBORDER	requests a border around plot
LFIT=*linetype*	specifies line style for fitted curves and confidence limits
LGRID=*linetype*	specifies line style for grid lines
NOFRAME	suppresses the frame around plotting areas
NOGRID	suppresses grid lines
NOFIT	suppresses fitted curves
NOHLABEL	suppresses horizontal labels
NOHTICK	suppresses horizontal ticks
NOVTICK	suppresses vertical ticks
TURNVLABELS	vertically strings out characters in vertical labels
WFIT=*n*	specifies thickness for fitted curves
WGRID=*n*	specifies thickness for grids
WREFL=*n*	specifies thickness for reference lines

Table 12.22. Axis Options

HAXIS=*value1 to value2 <by value3>*	specifies tick mark values for horizontal axis
HOFFSET=*value*	specifies offset for horizontal axis
HLOWER=*value*	specifies lower limit on horizontal axis scale
HUPPER=*value*	specifies upper limit on horizontal axis scale
NHTICK=*n*	specifies number of ticks for horizontal axis
NVTICK=*n*	specifies number of ticks for vertical axis
VAXIS=*value1 to value2 <by value3>*	specifies tick mark values for vertical axis
VAXISLABEL=*'label'*	specifies label for vertical axis
VOFFSET=*value*	specifies offset for vertical axis
VLOWER=*value*	specifies lower limit on vertical axis scale
VUPPER=*value*	specifies upper limit on vertical axis scale
WAXIS=*n*	specifies thickness for axis

Table 12.23. Graphics Catalog Options

DESCRIPTION=*'string'*	specifies description for graphics catalog member
NAME=*'string'*	specifies name for plot in graphics catalog

Table 12.24. Options for Text Enhancement

FONT=*font*	specifies software font for text
HEIGHT=*value*	specifies height of text used outside framed areas
INFONT=*font*	specifies software font for text inside framed areas
INHEIGHT=*value*	specifies height of text inside framed areas

Table 12.25. Options for Reference Lines

HREF<(INTERSECT)> =value-list	requests horizontal reference line
HREFLABELS= ('label1',. . .,'labeln')	specifies labels for HREF= lines
HREFLABPOS=n	specifies vertical position of labels for HREF= lines
LHREF=*linetype*	specifies line style for HREF= lines
LVREF=*linetype*	specifies line style for VREF= lines
VREF<(INTERSECT)> =value-list	requests vertical reference line
VREFLABELS= ('label1',. . .,'labeln')	specifies labels for VREF= lines
VREFLABPOS=n	specifies horizontal position of labels for VREF= lines

Dictionary of Options

The following entries provide detailed descriptions of the *options* in the LPREDPLOT statement.

ANNOTATE=*SAS-data-set*
ANNO=*SAS-data-set*

specifies an ANNOTATE data set, as described in *SAS/GRAPH Software: Reference*, that enables you to add features to the lpred plot. The ANNOTATE= data set you specify in the LPREDPLOT statement is used for all plots created by the statement.

CAXIS=*color*
CAXES=*color*

specifies the color used for the axes and tick marks. This option overrides any COLOR= specifications in an AXIS statement. The default is the first color in the device color list.

CFIT=*color*

specifies the color for the fitted lpred lines. The default is the first color in the device color list.

CFRAME=*color*
CFR=*color*

specifies the color for the area enclosed by the axes and frame. This area is not shaded by default.

CGRID=*color*
specifies the color for grid lines. The default is the first color in the device color list.

CHREF=*color*
CH=*color*
specifies the color for lines requested by the HREF= option. The default is the first color in the device color list.

CTEXT=*color*
specifies the color for tick mark values and axis labels. The default is the color specified for the CTEXT= option in the most recent GOPTIONS statement.

CVREF=*color*
CV=*color*
specifies the color for lines requested by the VREF= option. The default is the first color in the device color list.

DESCRIPTION='*string***'**
DES='*string***'**
specifies a description, up to 40 characters, that appears in the PROC GREPLAY master menu. The default is the variable name.

FONT=*font*
specifies a software font for reference line and axis labels. You can also specify fonts for axis labels in an AXIS statement. The FONT= font takes precedence over the FTEXT= font specified in the most recent GOPTIONS statement. Hardware characters are used by default.

HAXIS=*value1 to value2***<***by value3***>**
specifies tick mark values for the horizontal axis. *value1*, *value2*, and *value3* must be numeric, and *value1* must be less than *value2*. The lower tick mark is *value1*. Tick marks are drawn at increments of *value3*. The last tick mark is the greatest value that does not exceed *value2*. If *value3* is omitted, a value of 1 is used.

Examples of HAXIS= lists are:

```
haxis = 0 to 10
haxis = 2 to 10 by 2
haxis = 0 to 200 by 10
```

HEIGHT=*value*
specifies the height of text used outside framed areas. The default value is 3.846 (in percentage).

HLOWER=*value*
specifies the lower limit on the horizontal axis scale. The HLOWER= option specifies *value* as the lower horizontal axis tick mark. The tick mark interval and the upper axis limit are determined automatically. This option has no effect if the HAXIS= option is used.

HOFFSET=*value*
specifies offset for horizontal axis. The default value is 1.

HUPPER=*value*

specifies *value* as the upper horizontal axis tick mark. The tick mark interval and the lower axis limit are determined automatically. This option has no effect if the HAXIS= option is used.

HREF < (INTERSECT) > =*value-list*

requests reference lines perpendicular to the horizontal axis. If (INTERSECT) is specified, a second reference line perpendicular to the vertical axis is drawn that intersects the fit line at the same point as the horizontal axis reference line. If a horizontal axis reference line label is specified, the intersecting vertical axis reference line is labeled with the vertical axis value. See also the CHREF=, HREFLABELS=, and LHREF= options.

HREFLABELS=*'label1'***,**. . .**,***'labeln'*
HREFLABEL=*'label1'***,**. . .**,***'labeln'*
HREFLAB=*'label1'***,**. . .**,***'labeln'*

specifies labels for the lines requested by the HREF= option. The number of labels must equal the number of lines. Enclose each label in quotes. Labels can be up to 16 characters.

HREFLABPOS=*n*

specifies the vertical position of labels for HREF= lines. The following table shows valid values for *n* and the corresponding label placements.

n	**label placement**
1	top
2	staggered from top
3	bottom
4	staggered from bottom
5	alternating from top
6	alternating from bottom

INBORDER

requests a border around lpred plots.

LEVEL= (*character-list***)**
ORDINAL= (*character-list***)**

specifies the names of the levels for which linear predictor lines are requested. Names should be quoted and separated by space. If there is no correct name provided, no lpred line is plotted.

LFIT=*linetype*

specifies a line style for fitted curves and confidence limits. By default, fitted curves are drawn by connecting solid lines (*linetype = 1*) and confidence limits are drawn by connecting dashed lines (*linetype = 3*).

LGRID=*linetype*

specifies a line style for all grid lines. *linetype* is between 1 and 46. The default is 35.

LHREF=*linetype*
LH=*linetype*
specifies the line type for lines requested by the HREF= option. The default is 2, which produces a dashed line.

LVREF=*linetype*
LV=*linetype*
specifies the line type for lines requested by the VREF= option. The default is 2, which produces a dashed line.

NAME=*'string'*
specifies a name for the plot, up to eight characters, that appears in the PROC GREPLAY master menu. The default is 'PROBIT'.

NOCONF
suppresses confidence limits from the plot. This only works for the binomial model. Confidence limits are not plotted for the multinomial model.

NODATA
suppresses observed data points from the plot. This only works for the binomial model. Data points are not plotted for the multinomial model.

NOFIT
suppresses the fitted lpred lines.

NOFRAME
suppresses the frame around plotting areas.

NOGRID
suppresses grid lines.

NOHLABEL
suppresses horizontal labels.

NOHTICK
suppresses horizontal tick marks.

NOTHRESH
suppresses the threshold line.

NOVLABEL
suppresses vertical labels.

NOVTICK
suppresses vertical tick marks.

THRESHLABPOS=n
specifies the horizontal position of labels for the threshold line. The following table shows valid values for *n* and the corresponding label placements.

n	**label placement**
1	left
2	right

VAXIS=*value1 to value2***<***by value3***>**
specifies tick mark values for the vertical axis. *value1*, *value2*, and *value3* must be numeric, and *value1* must be less than *value2*. The lower tick mark is *value1*. Tick marks are drawn at increments of *value3*. The last tick mark is the greatest value that does not exceed *value2*. This method of specification of tick marks is not valid for logarithmic axes. If *value3* is omitted, a value of 1 is used.

Examples of VAXIS= lists are:

```
vaxis = 0 to 10
vaxis = 0 to 2 by .1
```

VAXISLABEL=*'string'*
specifies a label for the vertical axis.

VLOWER=*value*
specifies the lower limit on the vertical axis scale. The VLOWER= option specifies *value* as the lower vertical axis tick mark. The tick mark interval and the upper axis limit are determined automatically. This option has no effect if the VAXIS= option is used.

VREF=*value-list*
requests reference lines perpendicular to the vertical axis. If (INTERSECT) is specified, a second reference line perpendicular to the horizontal axis is drawn that intersects the fit line at the same point as the vertical axis reference line. If a vertical axis reference line label is specified, the intersecting horizontal axis reference line is labeled with the horizontal axis value. See also the CVREF=, LVREF=, and VREFLABELS= options.

VREFLABELS=*'label1',...,'labeln'*
VREFLABEL=*'label1',...,'labeln'*
VREFLAB=*'label1',...,'labeln'*
specifies labels for the lines requested by the VREF= option. The number of labels must equal the number of lines. Enclose each label in quotes. Labels can be up to 16 characters.

VREFLABPOS=*n*
specifies the horizontal position of labels for VREF= lines. The following table shows valid values for *n* and the corresponding label placements.

n	**label placement**
1	left
2	right

VUPPER=*value*
specifies the upper limit on the vertical axis scale. The VUPPER= option specifies *value* as the upper vertical axis tick mark. The tick mark interval and the lower axis limit are determined automatically. This option has no effect if the VAXIS= option is used.

WAXIS=*n*
specifies line thickness for axes and frame. The default value is 1.

WFIT=n
specifies line thickness for fitted lines. The default value is 1.

WGRID=n
specifies line thickness for grids. The default value is 1.

WREFL=n
specifies line thickness for reference lines. The default value is 1.

MODEL Statement

<*label:*> **MODEL** *response=effects* < */ options* > **;**

<*label:*> **MODEL** *events/trials=effects* < */ options* > **;**

AGGREGATE
AGGREGATE= *(variable-list)*
specifies the subpopulations on which the Pearson chi-square test statistic and the log-likelihood ratio chi-square test statistic (deviance) are calculated if the LACKFIT option is specified. See the section "Rescaling the Covariance Matrix" on page 280 for details of Pearson's chi-square and deviance calculations.

Observations with common values in the given list of variables are regarded as coming from the same subpopulation. Variables in the list can be any variables in the input data set. Specifying the AGGREGATE option is equivalent to specifying the AGGREGATE= option with a variable list that includes all independent variables in the MODEL statement. The PROBIT procedure sorts the input data set according to the variables specified in this list. Information for the sorted data set is reported in the "Response-Covariate Profile" table.

The deviance and Pearson goodness-of-fit statistics are calculated if the LACKFIT option is specified in the MODEL statement. The calculated results are reported in the "Goodness-of-Fit" table. If the Pearson chi-square test is significant with the test level specified by the HPROB= option, the fiducial limits, if required with the INVERSECL option in the MODEL statement, are modified. Also, the covariance matrix is re-scaled by the dispersion parameter when the SCALE= option is specified.

ALPHA=*value*
sets the significance level for the confidence intervals for regression parameters, fiducial limits for the predicted values, and confidence intervals for the predicted probabilities. The value must be between 0 and 1. The default value is ALPHA=0.05.

CONVERGE=*value*
The default is changed to CONVERGE=1E−8.

SCALE= *scale*
enables you to specify the method for estimating the dispersion parameter. To correct for overdispersion or underdispersion, the covariance matrix is multiplied by the estimate of the dispersion parameter. Valid values for *scale* are as follows:

D | DEVIANCE — specifies that the dispersion parameter be estimated by the deviance divided by its degrees of freedom.

P | PEARSON — specifies that the dispersion parameter be estimated by the Pearson chi-square statistic divided by its degrees of freedom. This is set as the default.

You can use the AGGREGATE= option to define the subpopulations for calculating the Pearson chi-square statistic and the deviance.

The "Goodness-of-Fit" table includes the Pearson chi-square statistic, the deviance, their degrees of freedom, the ratio of each statistic divided by its degrees of freedom, and the corresponding p-value.

PREDPPLOT Statement

PREDPPLOT <*var = variable*> <*options*>**;**

The PREDPPLOT statement plots the predicted probability against a single continuous variable (dose variable) in the MODEL statement for both the binomial model and the multinomial model. Confidence limits are only available for the binomial model. An attached box on the right side of the plot is used to label predicted probability curves with the names of their levels for the multinomial model. You can specify the color of this box using the CLABBOX= option.

VAR= *(variable)*
specifies a single continuous variable (dose variable) in the independent variable list of the MODEL statement. If a VAR= variable is not specified, the first single continuous variable in the independent variable list of the MODEL statement is used. If such a variable does not exist in the independent variable list of the MODEL statement, an error is reported.

The predicted probability is

$$\hat{p} = C + (1 - C)F(\mathbf{x}^{'}\hat{\mathbf{b}})$$

for the binomial model and

$$\begin{aligned}
\hat{p}_1 &= C + (1 - C)F(\mathbf{x}^{'}\hat{\mathbf{b}}) \\
\hat{p}_j &= (1 - C)(F(\hat{a}_j + \mathbf{x}^{'}\hat{\mathbf{b}}) - F(\hat{a}_{j-1} + \mathbf{x}^{'}\hat{\mathbf{b}})) \quad j = 2, \ldots, k - 1 \\
\hat{p}_k &= (1 - C)(1 - F(\hat{a}_{k-1} + \mathbf{x}^{'}\hat{\mathbf{b}}))
\end{aligned}$$

for the multinomial model with k response levels, where F is the cumulative distribution function used to model the probability, $\mathbf{x}^{'}$ is the vector of the covariates, $\hat{a}_j$ are the estimated ordinal intercepts with $\hat{a}_1 = 0$, C is the threshold parameter, either known or estimated from the model, and $\hat{\mathbf{b}}^{'}$ is the vector of estimated parameters.

To plot $\hat{p}$ (or $\hat{p}_j$) as a function of a continuous variable x_1, the remaining covariates $\mathbf{x}_{-1}$ must be specified. You can use the XDATA= option to provide the values of $\mathbf{x}_{-1}$ (see the XDATA= option in the PROC PROBIT statement for details), or use the default values that follow the rules:

- If the effect contains a continuous variable (or variables), the overall mean of this effect is used.
- If the effect is a single classification variable, the highest level of the variable is used.

options

enable you to plot the observed data and add features to the plot.

You can use options in the PREDPPLOT statement to

- superimpose specification limits
- suppress or add observed data points for the binomial model
- suppress or add confidence limits for the binomial model
- specify the levels for which predicted probability curves are requested for the multinomial model
- specify graphical enhancements (such as color or text height)

Summary of Options

The following tables list all *options* by function. The "Dictionary of Options" on page 273 describes each option in detail.

PREDPPLOT Options

Table 12.26. Plot Layout Options for PREDPPLOT

LEVEL=*character-list*	specifies the names of the levels for which the predicted probability curves are requested (only for the multinomial model)
NOCONF	suppresses confidence limits
NODATA	suppresses observed data points on the plot
NOTHRESH	suppresses the threshold line
THRESHLABPOS=*value*	specifies the position for the label of the threshold line

General Options

Table 12.27. Color Options

CAXIS=*color*	specifies color for the axes
CFIT=*color*	specifies color for fitted curves
CFRAME=*color*	specifies color for frame
CGRID=*color*	specifies color for grid lines
CHREF=*color*	specifies color for HREF= lines
CLABBOX=*color*	specifies color for label box
CTEXT=*color*	specifies color for text
CVREF=*color*	specifies color for VREF= lines

Table 12.28. Options to Enhance Plots Produced on Graphics Devices

ANNOTATE= *SAS-data-set*	specifies an ANNOTATE data set
INBORDER	requests a border around plot
LFIT=*linetype*	specifies line style for fitted curves and confidence limits
LGRID=*linetype*	specifies line style for grid lines
NOFRAME	suppresses the frame around plotting areas
NOGRID	suppresses grid lines
NOFIT	suppresses fitted curves
NOHLABEL	suppresses horizontal labels
NOHTICK	suppresses horizontal ticks
NOVTICK	suppresses vertical ticks
TURNVLABELS	vertically string out characters in vertical labels
WFIT=*n*	specifies thickness for fitted curves
WGRID=*n*	specifies thickness for grids
WREFL=*n*	specifies thickness for reference lines

Table 12.29. Axis Options

HAXIS=*value1 to value2 <by value3>*	specifies tick mark values for horizontal axis
HOFFSET=*value*	specifies offset for horizontal axis
HLOWER=*value*	specifies lower limit on horizontal axis scale
HUPPER=*value*	specifies upper limit on horizontal axis scale
NHTICK=*n*	specifies number of ticks for horizontal axis
NVTICK=*n*	specifies number of ticks for vertical axis
VAXIS=*value1 to value2 <by value3>*	specifies tick mark values for vertical axis
VAXISLABEL=*'label'*	specifies label for vertical axis
VOFFSET=*value*	specifies offset for vertical axis
VLOWER=*value*	specifies lower limit on vertical axis scale
VUPPER=*value*	specifies upper limit on vertical axis scale
WAXIS=*n*	specifies thickness for axis

Table 12.30. Graphics Catalog Options

DESCRIPTION=*'string'*	specifies description for graphics catalog member
NAME='*string*'	specifies name for plot in graphics catalog

Table 12.31. Options for Text Enhancement

FONT=*font*	specifies software font for text
HEIGHT=*value*	specifies height of text used outside framed areas
INFONT=*font*	specifies software font for text inside framed areas
INHEIGHT=*value*	specifies height of text inside framed areas

Table 12.32. Options for Reference Lines

HREF<(INTERSECT)> =value-list	requests horizontal reference line
HREFLABELS= ('label1',. . .,'labeln')	specifies labels for HREF= lines
HREFLABPOS=n	specifies vertical position of labels for HREF= lines
LHREF=*linetype*	specifies line style for HREF= lines
LVREF=*linetype*	specifies line style for VREF= lines
VREF<(INTERSECT)> =value-list	requests vertical reference line
VREFLABELS= ('label1',. . .,'labeln')	specifies labels for VREF= lines
VREFLABPOS=n	specifies horizontal position of labels for VREF= lines

Dictionary of Options

The following entries provide detailed descriptions of the *options* in the PREDPPLOT statement.

ANNOTATE=*SAS-data-set*
ANNO=*SAS-data-set*

specifies an ANNOTATE data set, as described in *SAS/GRAPH Software: Reference*, that enables you to add features to the predicted probability plot. The ANNOTATE= data set you specify in the PREDPPLOT statement is used for all plots created by the statement.

CAXIS=*color*
CAXES=*color*

specifies the color used for the axes and tick marks. This option overrides any COLOR= specifications in an AXIS statement. The default is the first color in the device color list.

CFIT=*color*

specifies the color for the fitted predicted probability curves. The default is the first color in the device color list.

CFRAME=*color*
CFR=*color*

specifies the color for the area enclosed by the axes and frame. This area is not shaded by default.

CGRID=*color*

specifies the color for grid lines. The default is the first color in the device color list.

CHREF=*color*
CH=*color*

specifies the color for lines requested by the HREF= option. The default is the first color in the device color list.

CTEXT=*color*

specifies the color for tick mark values and axis labels. The default is the color specified for the CTEXT= option in the most recent GOPTIONS statement.

CVREF=*color*
CV=*color*

specifies the color for lines requested by the VREF= option. The default is the first color in the device color list.

DESCRIPTION='*string***'**
DES='*string***'**

specifies a description, up to 40 characters, that appears in the PROC GREPLAY master menu. The default is the variable name.

FONT=*font*
specifies a software font for reference line and axis labels. You can also specify fonts for axis labels in an AXIS statement. The FONT= font takes precedence over the FTEXT= font specified in the most recent GOPTIONS statement. Hardware characters are used by default.

HAXIS=*value1 to value2***<***by value3***>**
specifies tick mark values for the horizontal axis. *value1*, *value2*, and *value3* must be numeric, and *value1* must be less than *value2*. The lower tick mark is *value1*. Tick marks are drawn at increments of *value3*. The last tick mark is the greatest value that does not exceed *value2*. If *value3* is omitted, a value of 1 is used.

Examples of HAXIS= lists are:

```
haxis = 0 to 10
haxis = 2 to 10 by 2
haxis = 0 to 200 by 10
```

HEIGHT=*value*
specifies the height of text used outside framed areas.

HLOWER=*value*
specifies the lower limit on the horizontal axis scale. The HLOWER= option specifies *value* as the lower horizontal axis tick mark. The tick mark interval and the upper axis limit are determined automatically. This option has no effect if the HAXIS= option is used.

HOFFSET=*value*
specifies the offset for the horizontal axis. The default value is 1.

HUPPER=*value*
specifies *value* as the upper horizontal axis tick mark. The tick mark interval and the lower axis limit are determined automatically. This option has no effect if the HAXIS= option is used.

HREF < (INTERSECT) > =*value-list*
requests reference lines perpendicular to the horizontal axis. If (INTERSECT) is specified, a second reference line perpendicular to the vertical axis is drawn that intersects the fit line at the same point as the horizontal axis reference line. If a horizontal axis reference line label is specified, the intersecting vertical axis reference line is labeled with the vertical axis value. See also the CHREF=, HREFLABELS=, and LHREF= options.

HREFLABELS=*'label1'*,. . .,*'labeln'*
HREFLABEL=*'label1'*,. . .,*'labeln'*
HREFLAB=*'label1'*,. . .,*'labeln'*
specifies labels for the lines requested by the HREF= option. The number of labels must equal the number of lines. Enclose each label in quotes. Labels can be up to 16 characters.

HREFLABPOS=*n*
specifies the vertical position of labels for HREF= lines. The following table shows valid values for *n* and the corresponding label placements.

n	**label placement**
1	top
2	staggered from top
3	bottom
4	staggered from bottom
5	alternating from top
6	alternating from bottom

INBORDER
requests a border around predicted probability plots.

LEVEL= (*character-list***)**
ORDINAL= (*character-list***)**
specifies the names of the levels for which predicted probability curves are requested. Names should be quoted and separated by space. If there is no correct name provided, no fitted probability curve is plotted.

LFIT=*linetype*
specifies a line style for fitted curves and confidence limits. By default, fitted curves are drawn by connecting solid lines (*linetype = 1*) and confidence limits are drawn by connecting dashed lines (*linetype = 3*).

LGRID=*linetype*
specifies a line style for all grid lines. *linetype* is between 1 and 46. The default is 35.

LHREF=*linetype*
LH=*linetype*
specifies the line type for lines requested by the HREF= option. The default is 2, which produces a dashed line.

LVREF=*linetype*
LV=*linetype*
specifies the line type for lines requested by the VREF= option. The default is 2, which produces a dashed line.

NAME=*'string'*
specifies a name for the plot, up to eight characters, that appears in the PROC GREPLAY master menu. The default is 'PROBIT'.

NOCONF
suppresses confidence limits from the plot. This only works for the binomial model. Confidence limits are not plotted for the multinomial model.

NODATA
suppresses observed data points from the plot. This only works for the binomial model. The data points are not plotted for the multinomial model.

NOFIT
suppresses the fitted predicted probability curves.

NOFRAME
suppresses the frame around plotting areas.

NOGRID
suppresses grid lines.

NOHLABEL
suppresses horizontal labels.

NOHTICK
suppresses horizontal tick marks.

NOTHRESH
suppresses the threshold line.

NOVLABEL
suppresses vertical labels.

NOVTICK
suppresses vertical tick marks.

THRESHLABPOS=*n*
specifies the horizontal position of labels for the threshold line. The following table shows valid values for *n* and the corresponding label placements.

n	**label placement**
1	left
2	right

VAXIS=*value1 to value2***<***by value3***>**
specifies tick mark values for the vertical axis. *value1*, *value2*, and *value3* must be numeric, and *value1* must be less than *value2*. The lower tick mark is *value1*. Tick marks are drawn at increments of *value3*. The last tick mark is the greatest value that does not exceed *value2*. This method of specification of tick marks is not valid for logarithmic axes. If *value3* is omitted, a value of 1 is used.

Examples of VAXIS= lists are:

```
vaxis = 0 to 10
vaxis = 0 to 2 by .1
```

VAXISLABEL=*'string'*
specifies a label for the vertical axis.

VLOWER=*value*
specifies the lower limit on the vertical axis scale. The VLOWER= option specifies *value* as the lower vertical axis tick mark. The tick mark interval and the upper axis limit are determined automatically. This option has no effect if the VAXIS= option is used.

VREF=*value-list*
requests reference lines perpendicular to the vertical axis. If (INTERSECT) is specified, a second reference line perpendicular to the horizontal axis is drawn that intersects the fit line at the same point as the vertical axis reference line. If a vertical axis reference line label is specified, the intersecting horizontal axis reference line is labeled with the horizontal axis value. See also the CVREF=, LVREF=, and VREFLABELS= options.

VREFLABELS=*'label1',. . .,'labeln'*
VREFLABEL=*'label1',. . .,'labeln'*
VREFLAB=*'label1',. . .,'labeln'*
specifies labels for the lines requested by the VREF= option. The number of labels must equal the number of lines. Enclose each label in quotes. Labels can be up to 16 characters.

VREFLABPOS=*n*
specifies the horizontal position of labels for VREF= lines. The following table shows valid values for *n* and the corresponding label placements.

n	**label placement**
1	left
2	right

VUPPER=*value*
specifies the upper limit on the vertical axis scale. The VUPPER= option specifies *value* as the upper vertical axis tick mark. The tick mark interval and the lower axis limit are determined automatically. This option has no effect if the VAXIS= option is used.

WAXIS=*n*
specifies line thickness for axes and frame. The default value is 1.

WFIT=*n*
specifies line thickness for fitted curves. The default value is 1.

WGRID=*n*
specifies line thickness for grids. The default value is 1.

WREFL=*n*
specifies line thickness for reference lines. The default value is 1.

Details

INEST= *SAS-data-set*

The INEST= data set names a SAS data set that specifies initial estimates for all the parameters in the model.

The INEST= data set must contain the intercept variables (named Intercept for binary response model and Intercept, Intercept2, Intercept3, and so forth, for multinomial response models) and all independent variables in the MODEL statement.

If BY processing is used, the INEST= data set should also include the BY variables, and there must be at least one observation for each BY group. If there is more than one observation in one BY group, the first one read is used for that BY group.

If the INEST= data set also contains the _TYPE_ variable, only observations with _TYPE_ value 'PARMS' are used as starting values. Combining the INEST= data set and the option MAXIT= in the MODEL statement, partial scoring can be done, such as predicting on a validation data set by using the model built from a training data set.

You can specify starting values for the iterative algorithm in the INEST= data set. This data set overwrites the INITIAL= option in the MODEL statement, which is a little difficult to use for models with multilevel interaction effects. The INEST= data set has the same structure as the OUTEST= data set, but is not required to have all the variables or observations that appear in the OUTEST= data set. One simple use of the INEST= option is passing the previous OUTEST= data set directly to the next model as an INEST= data set, assuming that the two models have the same parameterization.

Inverse Confidence Limits

In bioassay problems, estimates of the values of the independent variables that yield a desired response are often needed. For instance, the value yielding a 50% response rate (called the ED50 or LD50) is often used. The INVERSECL option requests that confidence limits be computed for the value of the independent variable that yields a specified response. These limits are computed only for the first continuous variable effect in the model. The other variables are set either at their mean values if they are continuous or at the reference (last) level if they are discrete variables. For a discussion of inverse confidence limits, refer to Hubert, Bohidar, and Peace (1988).

For the PROBIT procedure, the response variable is a probability. An estimate of the first continuous variable value needed to achieve a response of p is given by

$$\hat{x}_1 = \frac{1}{b_1}\left(F^{-1}(p) - \mathbf{x}^{*\prime}\mathbf{b}^*\right)$$

where F is the cumulative distribution function used to model the probability, $\mathbf{x}^*$ is the vector of independent variables excluding the first one, which can be specified by the XDATA= option described in the section "XDATA= *SAS-data-set*" on page 280, $\mathbf{b}^*$ is the vector of parameter estimates excluding the first one, and b_1 is the estimated regression coefficient for the independent variable of interest. Note that, for both binary and ordinal models, the INVERSECL option provides estimates of the value of x_1 yielding $\Pr(\text{first response level}) = p$, for various values of p.

This estimator is given as a ratio of random variables, for example, $r = a/b$. Confidence limits for this ratio can be computed using Fieller's theorem. A brief description of this theorem follows. Refer to Finney (1971) for a more complete description of Fieller's theorem.

If the random variables a and b are thought to be distributed as jointly normal, then for any fixed value r the following probability statement holds if z is an $\alpha/2$ quantile from the standard normal distribution and $\mathbf{V}$ is the variance-covariance matrix of a and b.

$$\Pr\left((a - rb)^2 > z^2(V_{aa} - 2rV_{ab} + r^2V_{bb})\right) = \alpha$$

Usually the inequality can be solved for r to yield a confidence interval. The PROBIT procedure uses a value of 1.96 for z, corresponding to an α value of 0.05, unless the goodness-of-fit p-value is less than the specified value of the HPROB= option. When this happens, the covariance matrix is scaled by the heterogeneity factor, and a t distribution quantile is used for z.

It is possible for the roots of the equation for r to be imaginary or for the confidence interval to be all points outside of an interval. In these cases, the limits are set to missing by the PROBIT procedure.

Although the normal and logistic distribution give comparable fitted values of p if the empirically observed proportions are not too extreme, they can give appreciably different values when extrapolated into the tails. Correspondingly, the estimates of the confidence limits and dose values can be different for the two distributions even when they agree quite well in the body of the data. Extrapolation outside of the range of the actual data are often sensitive to model assumptions, and caution is advised if extrapolation is necessary.

Lack of Fit Tests

Two goodness-of-fit tests can be requested from the PROBIT procedure: a Pearson chi-square test and a log-likelihood ratio chi-square test.

To compute the test statistics, you can use the AGGREGATE or AGGREGATE= option grouping the observations into subpopulations. If neither AGGREGATE nor AGGREGATE= is specified, PROC PROBIT assumes that each observation is from a separate subpopulation and computes the goodness-of-fit test statistics only for the *events/trials* syntax.

If the Pearson goodness-of-fit chi-square test is requested and the p-value for the test is too small, variances and covariances are adjusted by a heterogeneity factor (the goodness-of-fit chi-square divided by its degrees of freedom) and a critical value from the t distribution is used to compute the fiducial limits. The Pearson chi-square test statistic is computed as

$$\chi^2_P = \sum_{i=1}^{m} \sum_{j=1}^{k} \frac{(r_{ij} - n_i \hat{p}_{ij})^2}{n_i \hat{p}_{ij}}$$

where the sum on i is over grouping, the sum on j is over levels of response, the r_{ij} is the frequency of response level j for the ith grouping, n_i is the total frequency for the ith grouping, and $\hat{p}_{ij}$ is the fitted probability for the jth level at the ith grouping.

The likelihood ratio chi-square test statistic is computed as

$$\chi^2_D = 2 \sum_{i=1}^{m} \sum_{j=1}^{k} r_{ij} \ln \left(\frac{r_{ij}}{n_i \hat{p}_{ij}} \right)$$

This quantity is sometimes called the deviance. If the modeled probabilities fit the data, these statistics should be approximately distributed as chi-square with degrees of freedom equal to $(k-1) \times m - q$, where k is the number of levels of the multinomial or binomial response, m is the number of sets of independent variable values (covariate patterns), and q is the number of parameters fit in the model.

In order for the Pearson statistic and the deviance to be distributed as chi-square, there must be sufficient replication within the groupings. When this is not true, the data are

sparse, and the p-values for these statistics are not valid and should be ignored. Similarly, these statistics, divided by their degrees of freedom, cannot serve as indicators of overdispersion. A large difference between the Pearson statistic and the deviance provides some evidence that the data are too sparse to use either statistic.

Rescaling the Covariance Matrix

One way of correcting overdispersion is to multiply the covariance matrix by a dispersion parameter. You can supply the value of the dispersion parameter directly, or you can estimate the dispersion parameter based on either the Pearson chi-square statistic or the deviance for the fitted model.

The Pearson chi-square statistic χ^2_P and the deviance χ^2_D are defined in the section "Lack of Fit Tests" on page 279. If the SCALE= option is specified in the MODEL statement, the dispersion parameter is estimated by

$$\widehat{\sigma}^2 = \begin{cases} \chi^2_P/(m(k-1)-q) & \text{SCALE=PEARSON} \\ \chi^2_D/(m(k-1)-q) & \text{SCALE=DEVIANCE} \\ (constant)^2 & \text{SCALE=}\textit{constant} \end{cases}$$

In order for the Pearson statistic and the deviance to be distributed as chi-square, there must be sufficient replication within the subpopulations. When this is not true, the data are sparse, and the p-values for these statistics are not valid and should be ignored. Similarly, these statistics, divided by their degrees of freedom, cannot serve as indicators of overdispersion. A large difference between the Pearson statistic and the deviance provides some evidence that the data are too sparse to use either statistic.

You can use the AGGREGATE (or AGGREGATE=) option to define the subpopulation profiles. If you do not specify this option, each observation is regarded as coming from a separate subpopulation. For *events/trials* syntax, each observation represents n Bernoulli trials, where n is the value of the *trials* variable; for *single-trial* syntax, each observation represents a single trial. Without the AGGREGATE (or AGGREGATE=) option, the Pearson chi-square statistic and the deviance are calculated only for *events/trials* syntax.

Note that the parameter estimates are not changed by this method. However, their standard errors are adjusted for overdispersion, affecting their significance tests.

XDATA= *SAS-data-set*

The XDATA= data set is used for specifiying values for the effects in the MODEL statement when predicted values and/or fiducial limits for a single continuous variable (dose variable) are required. It is also used for plots specified by the CDFPLOT, IPPPLOT, LPREDPLOT, and PREDPPLOT statement.

The XDATA= data names a SAS data set that contains user input values for all the independent variables in the MODEL statement and the variables in the CLASS statement. The XDATA= data set has the same structure as the DATA= data set but is not required to have all the variables or observations that appear in the DATA= data set.

The XDATA= data set must contain all the independent variables in the MODEL statement and the variables in the CLASS statement. Even though variables in the CLASS statement may not be used in the MODEL statement, valid values are required for these variables in the XDATA= data set. Missing values are not allowed. For independent variables in the MODEL statement, although the dose variable's value is not used in the computing of predicted values and/or fiducial limits for the dose variable, missing values are not allowed in the XDATA= data set for any of the independent variables. Missing values are allowed for the dependent variables and other variables if they are included in the XDATA= data set and not listed in the CLASS statement.

If BY processing is used, the XDATA= data set should also include the BY variables, and there must be at least one valid observation for each BY group. If there is more than one valid observation in a BY group, the last one read is used for that BY group.

If there is no XDATA= data set in the PROC PROBIT statement, by default, the PROBIT procedure will use overall mean for effects containing a continuous variable (or variables) and the highest level of a single classification variable as the reference level. The rules are summarized as follows:

- If the effect contains a continuous variable (or variables), the overall mean of this effect is used.
- If the effect is a single classification variable, the highest level of the variable is used.

Examples

Example 12.1. Multilevel Response

In this example, two preparations, a standard preparation and a test preparation, are each given at several dose levels to groups of insects. The symptoms are recorded for each insect within each group, and two multilevel probit models are fit. Because the natural sort order of the three levels is not the same as the response order, the ORDER=DATA option is specified in the PROC PROBIT statement to get the desired order.

The following statements produce Output 12.1.1:

```
data multi;
   input Prep $ Dose Symptoms $ N;
   LDose=log10(Dose);
   if Prep='test' then PrepDose=LDose;
   else PrepDose=0;
   datalines;
stand     10      None       33
stand     10      Mild        7
stand     10      Severe     10
stand     20      None       17
```

```
stand     20      Mild       13
stand     20      Severe     17
stand     30      None       14
stand     30      Mild        3
stand     30      Severe     28
stand     40      None        9
stand     40      Mild        8
stand     40      Severe     32
test      10      None       44
test      10      Mild        6
test      10      Severe      0
test      20      None       32
test      20      Mild       10
test      20      Severe     12
test      30      None       23
test      30      Mild        7
test      30      Severe     21
test      40      None       16
test      40      Mild        6
test      40      Severe     19
;

proc probit order=data;
   class Prep Symptoms;
   nonpara: model Symptoms=Prep LDose PrepDose / lackfit;
   weight N;
   title 'Probit Models for Symptom Severity';
run;

proc probit order=data;
   class Prep Symptoms;
   parallel: model Symptoms=Prep LDose / lackfit;
   weight N;
   title 'Probit Models for Symptom Severity';
run;
```

The first model allows for nonparallelism between the dose response curves for the two preparations by inclusion of an interaction between Prep and LDose. The interaction term is labeled PrepDose in the "Analysis of Parameter Estimates" table. The results of this first model indicate that the parameter for the interaction term is not significant, having a Wald chi-square of 0.73. Also, since the first model is a generalization of the second, a likelihood ratio test statistic for this same parameter can be obtained by multiplying the difference in log likelihoods between the two models by 2. The value obtained, $2 \times (-345.94 - (-346.31))$, is 0.73. This is in close agreement with the Wald chi-square from the first model. The lack-of-fit test statistics for the two models do not indicate a problem with either fit.

Output 12.1.1. Multilevel Response: PROC PROBIT

Probit Models for Symptom Severity

Probit Procedure

Class Level Information

Name	Levels	Values
Prep	2	stand test
Symptoms	3	None Mild Severe

Probit Models for Symptom Severity

Probit Procedure

Model Information

Data Set	WORK.MULTI
Dependent Variable	Symptoms
Weight Variable	N
Number of Observations	23
Missing Values	1
Name of Distribution	Normal
Log Likelihood	-345.9401767

Probit Models for Symptom Severity

Probit Procedure

Analysis of Parameter Estimates

Parameter		DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept		1	3.8080	0.6252	2.5827	5.0333	37.10	<.0001
Intercept2		1	0.4684	0.0559	0.3589	0.5780	70.19	<.0001
Prep	stand	1	-1.2573	0.8190	-2.8624	0.3479	2.36	0.1247
Prep	test	0	0.0000	0.0000	0.0000	0.0000	.	.
LDose		1	-2.1512	0.3909	-2.9173	-1.3851	30.29	<.0001
PrepDose		1	-0.5072	0.5945	-1.6724	0.6580	0.73	0.3935

Probit Models for Symptom Severity

Probit Procedure

Class Level Information

Name	Levels	Values
Prep	2	stand test
Symptoms	3	None Mild Severe

Probit Models for Symptom Severity

Probit Procedure

Model Information

Data Set	WORK.MULTI
Dependent Variable	Symptoms
Weight Variable	N
Number of Observations	23
Missing Values	1
Name of Distribution	Normal
Log Likelihood	-346.306141

Probit Models for Symptom Severity

Probit Procedure

Analysis of Parameter Estimates

Parameter		DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept		1	3.4148	0.4126	2.6061	4.2235	68.50	<.0001
Intercept2		1	0.4678	0.0558	0.3584	0.5772	70.19	<.0001
Prep	stand	1	-0.5675	0.1259	-0.8142	-0.3208	20.33	<.0001
Prep	test	0	0.0000	0.0000	0.0000	0.0000	.	.
LDose		1	-2.3721	0.2949	-2.9502	-1.7940	64.68	<.0001

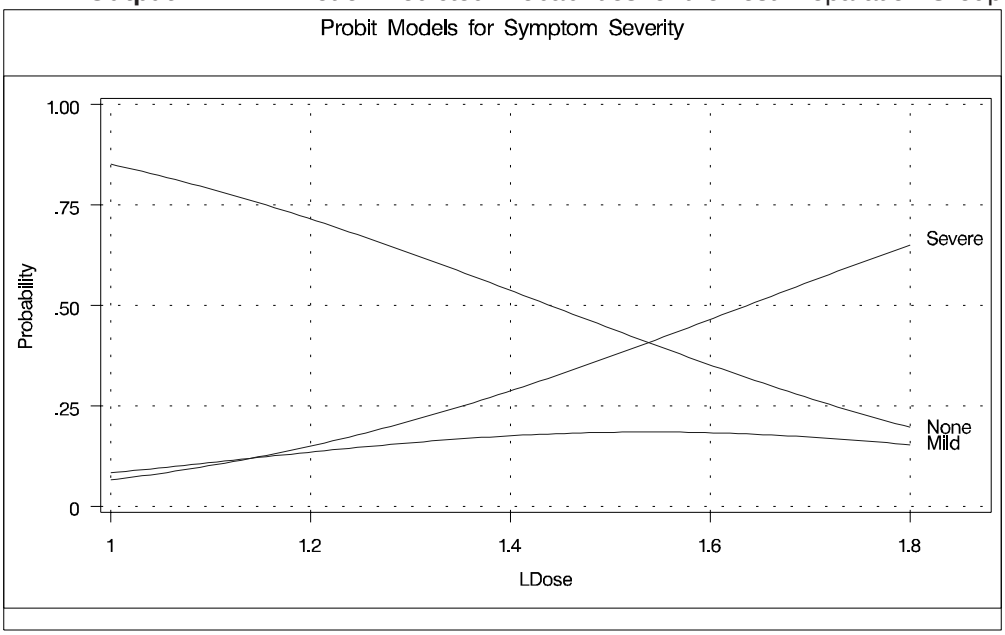
The negative coefficient associated with LDose indicates that the probability of having no symptoms (Symptoms='None') or no or mild symptoms (Symptoms='None' or Symptoms='Mild') decreases as LDose increases; that is, the probability of a severe symptom increases with LDose. This association is apparent for both treatment groups.

The negative coefficient associated with the standard treatment group (Prep = stand) indicates that the standard treatment is associated with more severe symptoms across all Ldose values.

The following statements use the PREDPPLOT statement to create the plot shown in Output 12.1.2. It plots the probabilities of the response taking on individual levels as a function of LDose. Since there are two covariates, LDose and Prep, the value of the CLASS variable Prep is fixed at the highest level, test. Although not shown here, the CDFPLOT statement creates similar plots of the cumulative response probabilities, instead of individual response level probabilities.

```
proc probit data=multi order=data;
   class Prep Symptoms;
   parallel: model Symptoms=Prep LDose / lackfit;
   predpplot var=ldose  level=("None" "Mild" "Severe")
             cfit=blue cframe=ligr inborder noconf ;
   weight N;
   title 'Probit Models for Symptom Severity';
run;
```

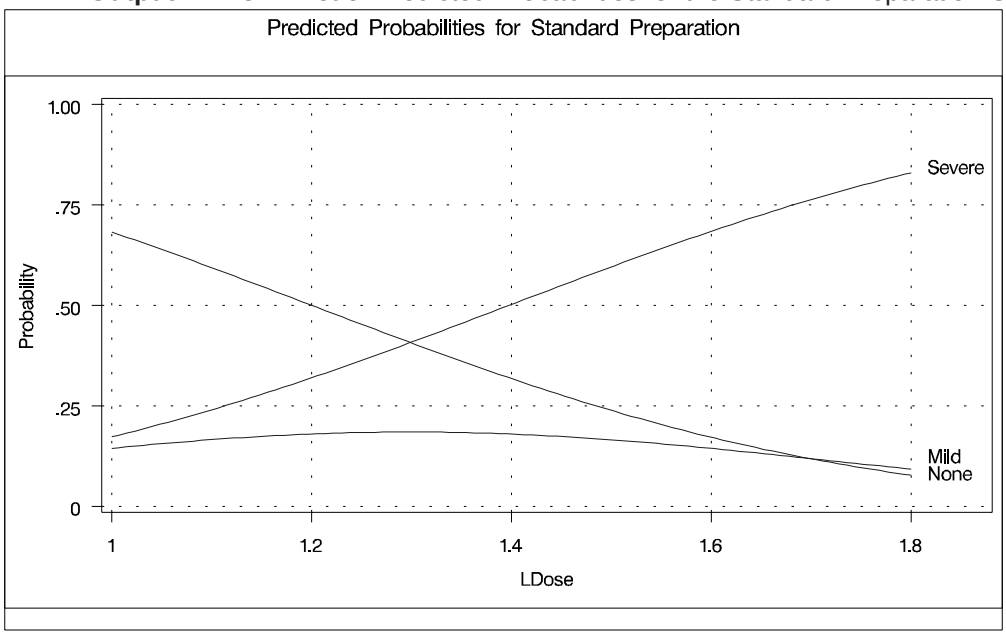
Output 12.1.2. Plot of Predicted Probabilities for the Test Preparation Group



The following statements use the XDATA= data set to create a plot of the predicted probabilities with Prep set to the stand level. The resulting plot is shown in Output 12.1.3.

```
data xrow;
   input Prep $ Dose Symptoms $ N;
   LDose=log10(Dose);
   datalines;
stand     40      Severe     32
run;

proc probit data=multi order=data xdata=xrow;
   class Prep Symptoms;
   parallel: model Symptoms=Prep LDose / lackfit;
   predpplot var=ldose  level=("None" "Mild" "Severe")
             cfit=blue cframe=ligr inborder noconf ;
   weight N;
   title 'Predicted Probabilities for Standard Preparation';
run;
```

Output 12.1.3. Plot of Predicted Probabilities for the Standard Preparation Group



Example 12.2. An Epidemiology Study

The data, which is from an epidemiology study, consists of five variables: the number, r, of individuals surviving after an epidemic, out of n treated, for combinations of medicine dosage (dose), treatment (treat = A, B), and sex (sex = 0(Female), 1(Male)).

To see if the two treatments have different effects on male and female individual survival rate, the interaction term between the two variables treat and sex is included in the model.

The following invocation of PROC PROBIT fits the binary probit model to the grouped data:

```
data epidemic;
   input treat$ dose n r sex;
   label dose = Dose;
   datalines;
A  2.17 142 142  0
A   .57 132  47  1
A  1.68 128 105  1
A  1.08 126 100  0
A  1.79 125 118  0
B  1.66 117 115  1
B  1.49 127 114  0
B  1.17  51  44  1
B  2.00 127 126  0
B   .80 129 100  1
;
```

```
data xval;
   input treat $ dose sex ;
   datalines;
B  2.  1
;

title 'Epidemiology Study';
symbol v=dot c=white;
proc probit optc lackfit covout data = epidemic
                        outest = out1 xdata = xval;
   class treat sex;
   model r/n = dose treat sex sex*treat/corrb covb inversecl;
   output out = out2 p =p;

   predpplot
            var = dose
            font = swiss
            vref(intersect) =  .6667
            vreflab = 'two thirds'
            vreflabpos = 2
            cfit=blue
            cframe=ligr
            ;
   inset    /
            cfill = white
            ctext = blue
            pos = se ;

   ippplot
            font = swiss
            href(intersect) =  .75
            hreflab = 'three quarters'
            vreflabpos = 2
            threshlabpos = 2
            cfit=blue
            cframe=ligr
            ;
   inset    /
            cfill = white
            ctext = blue;

   lpredplot
            font = swiss
            vref(intersect) =  1.
            vreflab = 'unit probit'
            vreflabpos = 2
            cfit=blue
            cframe=ligr
            ;
   inset    /
            cfill = white
            ctext = blue;
run;
```

The results of this analysis are shown in the following tables and figures.

Beginning with SAS Release 8.2, the PROBIT procedure does not support multiple MODEL statements. Only the last one is used if there is more than one MODEL statement in one invocation of the PROBIT procedure.

Output 12.2.1. Class Level Information

Epidemiology Study

Probit Procedure

Class Level Information

Name	Levels	Values
treat	2	A B
sex	2	0 1

Output 12.2.1 displays the table of level information for *all* classification variables in the CLASS statement.

Output 12.2.2. Parameter Information

Epidemiology Study

Probit Procedure

Parameter Information

Parameter	Effect	treat	sex
Intercept	Intercept		
dose	dose		
treatA	treat	A	
treatB	treat	B	
sex0	sex		0
sex1	sex		1
treatAsex0	treat*sex	A	0
treatAsex1	treat*sex	A	1
treatBsex0	treat*sex	B	0
treatBsex1	treat*sex	B	1

Output 12.2.2 displays the table of parameter information for the effects in the MODEL statement. The name of a parameter is generated from combining the variable names and level names in the effect. The maximum length of a parameter name is 32. The name of the effects are specified in the MODEL statement. The length of names of effects can be specified by the NAMELEN= option in the PROC PROBIT statement, with the default length 20.

Output 12.2.3. Model Information

```
                          Epidemiology Study

                           Probit Procedure

                          Model Information

                Data Set                    WORK.EPIDEMIC
                Events Variable                         r
                Trials Variable                         n
                Number of Observations                 10
                Number of Events                     1011
                Number of Trials                     1204
                Name of Distribution               Normal
                Log Likelihood               -387.2467391


Algorithm converged.
```

Output 12.2.3 displays background information about the model fit. Included are the name of the input data set, the response variables used, and the number of observations, events, and trials. The table also includes the status of the convergence of the model fitting algorithm and the final value of log-likelihood function.

Output 12.2.4. Goodness-of-Fit Tests and Response-Covariate Profile

```
                          Epidemiology Study

                           Probit Procedure

                       Goodness-of-Fit Tests

  Statistic                          Value       DF    Pr > ChiSq

  Pearson Chi-Square                4.9317        4        0.2944
  L.R.    Chi-Square                5.7079        4        0.2220


                    Response-Covariate Profile

                 Response Levels                2
                 Number of Covariate Values    10
```

Output 12.2.4 displays the table of goodness-of-fit tests requested with the LACKFIT option in the PROC PROBIT statement. Two goodness-of-fit statistics, the Pearson chi-square statistic and the likelihood ratio chi-square statistic, are computed. The grouping method for computing these statistics can be specified by the AGGREGATE= option. The details can be found in the AGGREGATE= option and an example can be found in the second part of this example. By default, the PROBIT procedure uses the covariates in the MODEL statement to do grouping. Observations with the same values of the covariates in the MODEL statement are grouped into cells, and the two statistics are computed according to these cells. The total number of cells and the number of levels for the response variable are reported next in the "Response-Covariate Profile."

In this example, neither the Pearson chi-square nor the log-likelihood ratio chi-square tests are significant at the 0.1 level, which is the default test level used by the PROBIT

procedure. That means that the model, which includes the interaction of treat and sex, is suitable for this epidemiology data set. (Further investigation shows that models without the interaction of treat and sex are not acceptable by either test.)

Output 12.2.5. Type III Tests

Epidemiology Study

Probit Procedure

Type III Analysis of Effects

Effect	DF	Wald Chi-Square	Pr > ChiSq
dose	1	42.1691	<.0001
treat	1	16.1421	<.0001
sex	1	1.7710	0.1833
treat*sex	1	13.9343	0.0002

Output 12.2.5 displays the Type III test results for all effects specified in the MODEL statement, which include the degrees of freedom for the effect, the Wald Chi-Square test statistic, and the p-value.

Output 12.2.6. Analysis of Parameter Estimates

Epidemiology Study

Probit Procedure

Analysis of Parameter Estimates

Parameter		DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept		1	-0.8871	0.3632	-1.5991	-0.1752	5.96	0.0146
dose		1	1.6774	0.2583	1.1711	2.1837	42.17	<.0001
treat	A	1	-1.2537	0.2616	-1.7664	-0.7410	22.97	<.0001
treat	B	0	0.0000	0.0000	0.0000	0.0000	.	.
sex	0	1	-0.4633	0.2289	-0.9119	-0.0147	4.10	0.0429
sex	1	0	0.0000	0.0000	0.0000	0.0000	.	.
treat*sex	A 0	1	1.2899	0.3456	0.6126	1.9672	13.93	0.0002
treat*sex	A 1	0	0.0000	0.0000	0.0000	0.0000	.	.
treat*sex	B 0	0	0.0000	0.0000	0.0000	0.0000	.	.
treat*sex	B 1	0	0.0000	0.0000	0.0000	0.0000	.	.
_C_		1	0.2735	0.0946	0.0881	0.4589		

Output 12.2.6 displays the table of parameter estimates for the model. The PROBIT procedure displays information for all the parameters of an effect. Degenerate parameters are indicated by 0 degree of freedom. Confidence intervals are computed for all parameters with non-zero degrees of freedom, including the natural threshold C if the OPTC option is specified in the PROC PROBIT statement. The confidence level can be specified by the ALPHA= option in the MODEL statement. The default confidence level is 95%.

From this table, you can see the following results:

- dose has significant positive effect on the survival rate.

- Individuals under treatment A have a lower survival rate.
- Male individuals have a higher survival rate.
- Female individuals under treatment A have a higher survival rate.

Output 12.2.7. Estimated Covariance Matrix

Epidemiology Study

Probit Procedure

Estimated Covariance Matrix

	Intercept	dose	treatA	sex0	treatAsex0
Intercept	0.131944	-0.087353	0.053551	0.030285	-0.067056
dose	-0.087353	0.066723	-0.047506	-0.034081	0.058620
treatA	0.053551	-0.047506	0.068425	0.036063	-0.075323
sex0	0.030285	-0.034081	0.036063	0.052383	-0.063599
treatAsex0	-0.067056	0.058620	-0.075323	-0.063599	0.119408
_C_	-0.028073	0.018196	-0.017084	-0.008088	0.019134

Estimated Covariance Matrix

	_C_
Intercept	-0.028073
dose	0.018196
treatA	-0.017084
sex0	-0.008088
treatAsex0	0.019134
_C_	0.008948

Output 12.2.8. Estimated Correlation Matrix

Epidemiology Study

Probit Procedure

Estimated Correlation Matrix

	Intercept	dose	treatA	sex0	treatAsex0
Intercept	1.000000	-0.930998	0.563595	0.364284	-0.534227
dose	-0.930998	1.000000	-0.703083	-0.576477	0.656744
treatA	0.563595	-0.703083	1.000000	0.602359	-0.833299
sex0	0.364284	-0.576477	0.602359	1.000000	-0.804154
treatAsex0	-0.534227	0.656744	-0.833299	-0.804154	1.000000
_C_	-0.817027	0.744699	-0.690420	-0.373565	0.585364

Estimated Correlation Matrix

	_C_
Intercept	-0.817027
dose	0.744699
treatA	-0.690420
sex0	-0.373565
treatAsex0	0.585364
_C_	1.000000

Output 12.2.7 and Output 12.2.8 display tables of estimated covariance matrix and estimated correlation matrix for estimated parameters with a non-zero degree of freedom, respectively. They are computed by the inverse of the Hessian matrix of the estimated parameters.

Output 12.2.9. Probit Analysis on Dose

Epidemiology Study

Probit Procedure

Probit Analysis on dose

Probability	dose	95% Fiducial Limits	
0.01	-0.85801	-1.81301	-0.33743
0.02	-0.69549	-1.58167	-0.21116
0.03	-0.59238	-1.43501	-0.13093
0.04	-0.51482	-1.32476	-0.07050
0.05	-0.45172	-1.23513	-0.02130
0.06	-0.39802	-1.15888	0.02063
0.07	-0.35093	-1.09206	0.05742
0.08	-0.30877	-1.03226	0.09039
0.09	-0.27043	-0.97790	0.12040
0.10	-0.23513	-0.92788	0.14805
0.15	-0.08900	-0.72107	0.26278
0.20	0.02714	-0.55706	0.35434
0.25	0.12678	-0.41669	0.43322
0.30	0.21625	-0.29095	0.50437
0.35	0.29917	-0.17477	0.57064
0.40	0.37785	-0.06487	0.63387
0.45	0.45397	0.04104	0.69546
0.50	0.52888	0.14481	0.75654
0.55	0.60380	0.24800	0.81819
0.60	0.67992	0.35213	0.88157
0.65	0.75860	0.45879	0.94803
0.70	0.84151	0.56985	1.01942
0.75	0.93099	0.68770	1.09847
0.80	1.03063	0.81571	1.18970
0.85	1.14677	0.95926	1.30171
0.90	1.29290	1.12867	1.45386
0.91	1.32819	1.16747	1.49273
0.92	1.36654	1.20867	1.53590
0.93	1.40870	1.25284	1.58450
0.94	1.45579	1.30084	1.64012
0.95	1.50949	1.35397	1.70515
0.96	1.57258	1.41443	1.78353
0.97	1.65015	1.48626	1.88238
0.98	1.75326	1.57833	2.01720
0.99	1.91577	1.71776	2.23537

Output 12.2.9 displays the computed values and fiducial limits for the first single continuous variable dose in the MODEL statement, given the probability levels, without the effect of the natural threshold, and when the option INVERSECL in the MODEL statement is specified. If there is no single continuous variable in the MODEL specification but the INVERSECL option is specified, an error is reported. If the XDATA= option is used to input a data set for the independent variables in the MODEL statement, the PROBIT procedure uses these values for the independent variables other than the single continuous variable. Missing values are not permitted in the XDATA= data set for the independent variables, although the value for the single continuous variable is not used in the computing of the fiducial limits. A suitable valid value

should be given. In the data set xval created by the SAS statements on page 286, Dose = 2.

See the section "XDATA= *SAS-data-set*" on page 280 for the default values for those effects other than the single continuous variable, for which the fiducial limits are computed.

In this example, there are two classification variables, treat and sex. Fiducial limits for the dose variable are computed for the highest level of the classification variables, treat = B and sex = 1, which is the default specification. Since these are the default values, you would get the same values and fiducial limits if you did not specify the XDATA= option in this example. The confidence level for the fiducial limits can be specified by the ALPHA= option in the MODEL statement. The default value is 0.05, resulting in 95% confidence limits.

If a LOG10 or LOG option is used in the PROC PROBIT statement, the values and the fiducial limits are computed for both the single continuous variable and its logarithm.

Output 12.2.10. Outest Data Set for Epidemiology Study

Obs	_MODEL_	_NAME_	_TYPE_	_DIST_	_STATUS_	_LNLIKE_	r	Intercept
1		r	PARMS	Normal	0 Converged	-387.247	-1.00000	-0.88714
2		Intercept	COV	Normal	0 Converged	-387.247	-0.88714	0.13194
3		dose	COV	Normal	0 Converged	-387.247	1.67739	-0.08735
4		treatA	COV	Normal	0 Converged	-387.247	-1.25367	0.05355
5		treatB	COV	Normal	0 Converged	-387.247	0.00000	0.00000
6		sex0	COV	Normal	0 Converged	-387.247	-0.46329	0.03029
7		sex1	COV	Normal	0 Converged	-387.247	0.00000	0.00000
8		treatAsex0	COV	Normal	0 Converged	-387.247	1.28991	-0.06706
9		treatAsex1	COV	Normal	0 Converged	-387.247	0.00000	0.00000
10		treatBsex0	COV	Normal	0 Converged	-387.247	0.00000	0.00000
11		treatBsex1	COV	Normal	0 Converged	-387.247	0.00000	0.00000
12		_C_	COV	Normal	0 Converged	-387.247	0.27347	-0.02807

Obs	dose	treatA	treat B	sex0	sex1	treat Asex0	treat Asex1	treat Bsex0	treat Bsex1	_C_
1	1.67739	-1.25367	0	-0.46329	0	1.28991	0	0	0	0.27347
2	-0.08735	0.05355	0	0.03029	0	-0.06706	0	0	0	-0.02807
3	0.06672	-0.04751	0	-0.03408	0	0.05862	0	0	0	0.01820
4	-0.04751	0.06843	0	0.03606	0	-0.07532	0	0	0	-0.01708
5	0.00000	0.00000	0	0.00000	0	0.00000	0	0	0	0.00000
6	-0.03408	0.03606	0	0.05238	0	-0.06360	0	0	0	-0.00809
7	0.00000	0.00000	0	0.00000	0	0.00000	0	0	0	0.00000
8	0.05862	-0.07532	0	-0.06360	0	0.11941	0	0	0	0.01913
9	0.00000	0.00000	0	0.00000	0	0.00000	0	0	0	0.00000
10	0.00000	0.00000	0	0.00000	0	0.00000	0	0	0	0.00000
11	0.00000	0.00000	0	0.00000	0	0.00000	0	0	0	0.00000
12	0.01820	-0.01708	0	-0.00809	0	0.01913	0	0	0	0.00895

Output 12.2.10 displays the OUTEST= data set. All parameters for an effect are included.

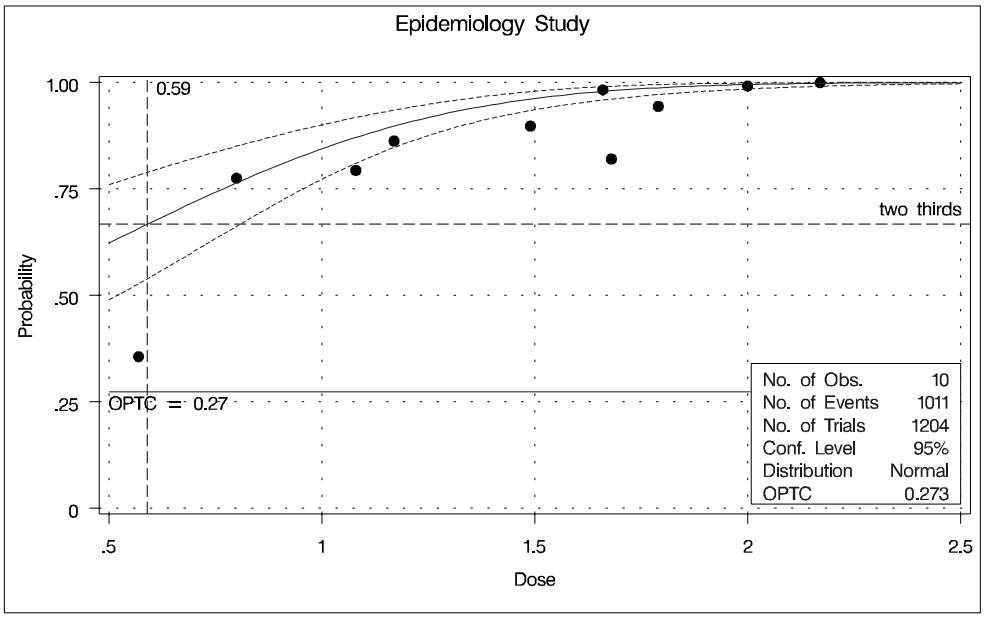
The following three outputs, Output 12.2.11, Output 12.2.12, and Output 12.2.13, are generated from the three plot statements. The first plot, specified with the PREDPPLOT statement, is the plot of the predicted probability against the single continuous variable Dose, which is specified by the VAR= option in the PREDPPLOT statement. This single continuous variable must be in the MODEL statement. If the VAR= op-

tion is not used, the first single continuous variable in the MODEL statement is used. In this example, you would get the same plot if the VAR = dose was not used in the PREDPPLOT statement. You can specify values of other independent variables in the MODEL statement using an XDATA= data set, or by using the default values.

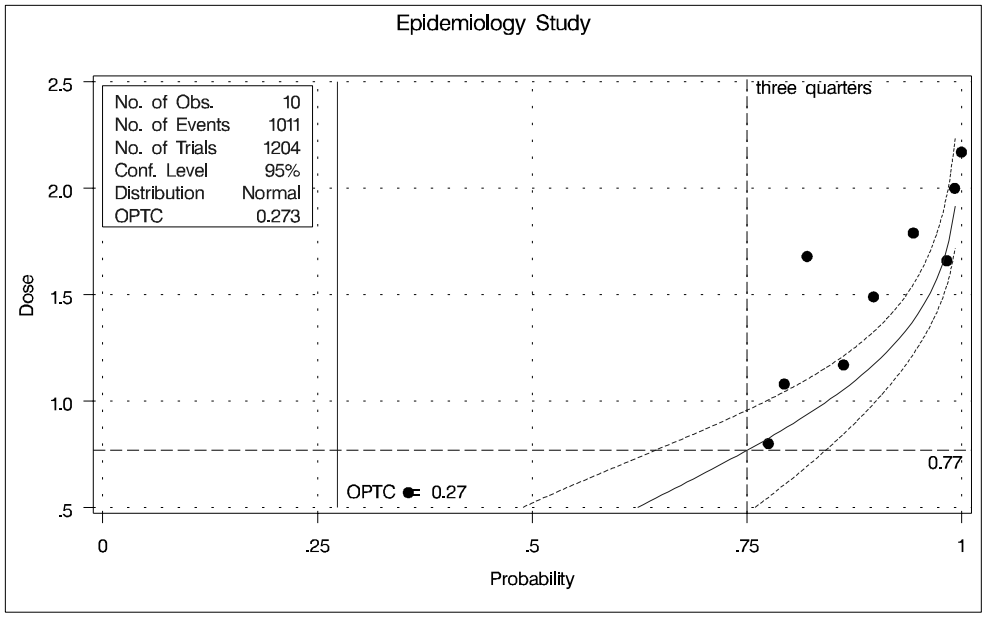
The second plot, specified with the IPPPLOT statement, is the inverse of the predicted probability plot with the fiducial limits. It should be pointed out that the fiducial limits are *not* just the inverse of the confidence limits in the predicted probability plot; see the section "Inverse Confidence Limits" on page 278 for the computation of these limits. The third plot, specified with the LPREDPLOT statement, is the plot of the linear predictor $\mathbf{x}'\beta$ against the first single continuous variable (or the single continuous variable specified by the VAR= option) with the Wald confidence intervals.

After each plot statement, an optional INSET statement is used to draw a box within the plot (inset box). In the inset box, information about the model fitting can be specified. See "INSET Statement" on page 250 for more detail.

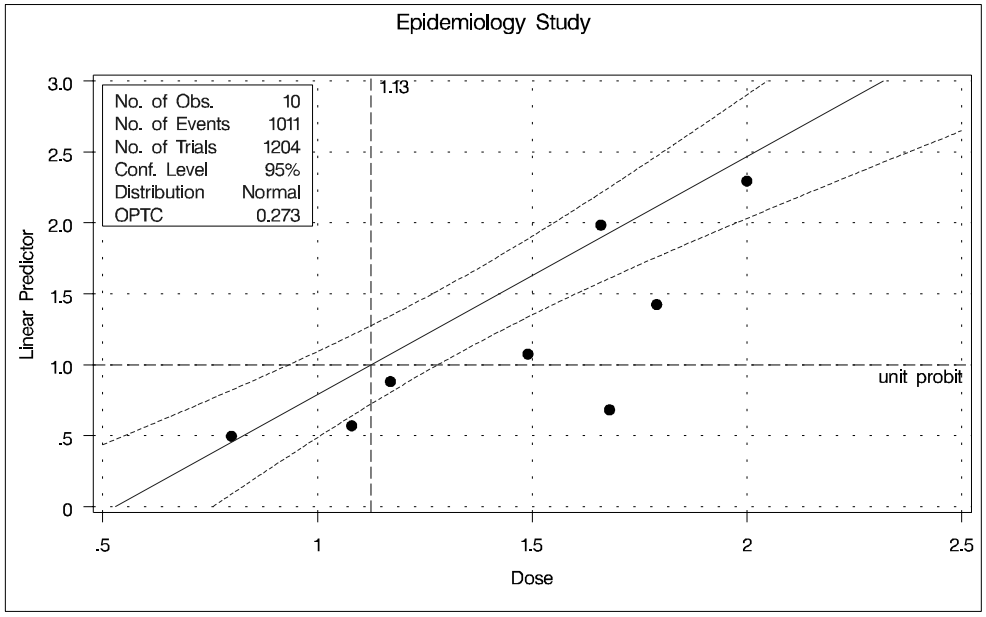
Output 12.2.11. Predicted Probability Plot



Output 12.2.12. Inverse Predicted Probability Plot



Output 12.2.13. Linear Predictor Plot



Combining INEST= data set and the MAXIT= option in the MODEL statement, the PROBIT procedure can do prediction, if the parameterizations for the models used for the training data and the validation data are exactly the same.

After the first invocation of PROC PROBIT, you have the estimated parameters and their covariance matrix in the data set OUTEST = Out1, and the fitted probabilities for the training data set epidemic in the data set OUTPUT = Out2. See Output 12.2.10 on page 293 for the data set Out1 and Output 12.2.14 on page 297 for the data set Out2.

The validation data are collected in data set validate. The second invocation of PROC PROBIT simply passes the estimated parameters from the training data set epidemic to the validation data set validate for prediction. The predicted probabilities are stored in the data set OUTPUT = Out3 (see Output 12.2.15 on page 297). The third invocation of PROC PROBIT passes the estimated parameters as initial values for a new fit of the validation data set using the same model. Predicted probabilities are stored in the data set OUTPUT = Out4 (see Output 12.2.16 on page 297). Goodness-of-Fit tests are computed based on the cells grouped by the AGGREGATE= group variable. Results are shown in Output 12.2.17 on page 298.

```
data validate;
   input treat $ dose  sex  n r group;
   datalines;
B  2.0  0  44 43  1
B  2.0  1  54 52  2
B  1.5  1  36 32  3
B  1.5  0  45 40  4
A  2.0  0  66 64  5
A  2.0  1  89 89  6
A  1.5  1  45 39  7
A  1.5  0  66 60  8
B  2.0  0  44 44  1
B  2.0  1  54 54  2
B  1.5  1  36 30  3
B  1.5  0  45 41  4
A  2.0  0  66 65  5
A  2.0  1  89 88  6
A  1.5  1  45 38  7
A  1.5  0  66 59  8
;

proc probit optc data = validate inest = out1;
   class treat sex;
   model r/n = dose treat sex sex*treat  / maxit = 0 ;
   output out = out3 p =p;
run ;

proc probit optc lackfit data = validate inest = out1;
   class treat sex;
   model r/n = dose treat sex sex*treat / aggregate = group ;
   output out = out4 p =p;
run ;
```

Output 12.2.14. Out2

Obs	treat	dose	n	r	sex	p
1	A	2.17	142	142	0	0.99272
2	A	0.57	132	47	1	0.35925
3	A	1.68	128	105	1	0.81899
4	A	1.08	126	100	0	0.77517
5	A	1.79	125	118	0	0.96682
6	B	1.66	117	115	1	0.97901
7	B	1.49	127	114	0	0.90896
8	B	1.17	51	44	1	0.89749
9	B	2.00	127	126	0	0.98364
10	B	0.80	129	100	1	0.76414

Output 12.2.15. Out3

Obs	treat	dose	sex	n	r	group	p
1	B	2.0	0	44	43	1	0.98364
2	B	2.0	1	54	52	2	0.99506
3	B	1.5	1	36	32	3	0.96247
4	B	1.5	0	45	40	4	0.91145
5	A	2.0	0	66	64	5	0.98500
6	A	2.0	1	89	89	6	0.91835
7	A	1.5	1	45	39	7	0.74300
8	A	1.5	0	66	60	8	0.91666
9	B	2.0	0	44	44	1	0.98364
10	B	2.0	1	54	54	2	0.99506
11	B	1.5	1	36	30	3	0.96247
12	B	1.5	0	45	41	4	0.91145
13	A	2.0	0	66	65	5	0.98500
14	A	2.0	1	89	88	6	0.91835
15	A	1.5	1	45	38	7	0.74300
16	A	1.5	0	66	59	8	0.91666

Output 12.2.16. Out4

Obs	treat	dose	sex	n	r	group	p
1	B	2.0	0	44	43	1	0.98954
2	B	2.0	1	54	52	2	0.98262
3	B	1.5	1	36	32	3	0.86187
4	B	1.5	0	45	40	4	0.90095
5	A	2.0	0	66	64	5	0.98768
6	A	2.0	1	89	89	6	0.98614
7	A	1.5	1	45	39	7	0.88075
8	A	1.5	0	66	60	8	0.88964
9	B	2.0	0	44	44	1	0.98954
10	B	2.0	1	54	54	2	0.98262
11	B	1.5	1	36	30	3	0.86187
12	B	1.5	0	45	41	4	0.90095
13	A	2.0	0	66	65	5	0.98768
14	A	2.0	1	89	88	6	0.98614
15	A	1.5	1	45	38	7	0.88075
16	A	1.5	0	66	59	8	0.88964

Output 12.2.17. Goodness-of-Fit Table

Probit Procedure

Goodness-of-Fit Tests

Statistic	Value	DF	Pr > ChiSq
Pearson Chi-Square	2.8101	2	0.2454
L.R. Chi-Square	2.8080	2	0.2456

References

Agresti, A. (1990), *Categorical Data Analysis*, New York: John Wiley & Sons.

Collett, D (1991), *Modelling Binary Data*, London: Chapman and Hall.

Cox, D.R. (1970), *Analysis of Binary Data*, London: Chapman and Hall.

Cox, D.R. and Oakes, D. (1984), *Analysis of Survival Data*, London: Chapman and Hall.

Finney, D.J. (1971), *Probit Analysis*, Third Edition, London: Cambridge University Press.

Hubert, J.J., Bohidar, N.R., and Peace, K.E. (1988), "Assessment of Pharmacological Activity," *Biopharmaceutical Statistics for Drug Development*, ed. K.E. Peace, New York: Marcel Dekker.

Chapter 13
The REG Procedure

Chapter Table of Contents

Chapter 13
The REG Procedure

Overview

The SCORR1 and SCORR2 options in the MODEL statement now enable you to specify TEST and SEQTEST options that request F-tests, *p*-values, and cumulative R-Square values as variables are sequentially added to a model.

Syntax

MODEL Statement

The following new and updated options are available in the MODEL statement after a slash (/).

PARTIALR2 < **(** < **TESTS** > < **SEQTESTS** > **)** >
See the SCORR1 option.

SCORR1 < **(** < **TESTS** > < **SEQTESTS** > **)** >
displays the squared semi-partial correlation coefficients using Type I sums of squares. This is calculated as SS/SST, where SST is the corrected total SS. If the NOINT option is used, the uncorrected total SS is used in the denominator. The optional arguments TESTS and SEQTESTS request F-tests, *p*-values, and cumulative R-Square values as variables are sequentially added to a model. The F-test values are computed as the Type I sum of squares for the variable in question divided by a mean square error. If you specify the TESTS option, the denominator MSE is the residual mean square for the full model specified in the MODEL statement. If you specify the SEQTESTS option, the denominator MSE is the residual mean square for the model containing all the independent variables that have been added to the model up to and including the variable in question. The TESTS and SEQTESTS options are not supported if you specify model selection methods, or if you specify the RIDGE or PCOMIT options. Note that the PARTIALR2 option is a synonym for the SCORR1 option.

SCORR2 < **(TESTS)** >
displays the squared semi-partial correlation coefficients using Type II sums of squares. These are calculated the same way as with the SCORR1 option, except that Type II SS are used instead of Type I SS. The optional TEST argument requests F-tests, *p*-values, and cumulative R-Square values as variables are sequentially added to a model. The F-test values are computed as the Type II sum of squares for the variable in question divided by the residual mean square for the full model specified in the MODEL statement. The TESTS option is not supported if you specify model selection methods, or if you specify the RIDGE or PCOMIT options.

Chapter 14
The SURVEYMEANS Procedure

Chapter Table of Contents

Chapter 14
The SURVEYMEANS Procedure

Overview

You can estimate ratios with the new RATIO statement in the SURVEYMEANS procedure. You can also obtain covariance coefficients for total estimators.

Syntax

PROC SURVEYMEANS Statement

PROC SURVEYMEANS < *options* > < *statistic-keywords* > **;**

In the PROC SURVEYMEANS statement, you can use *statistic-keywords* to specify statistics for the procedure to compute.

statistic-keywords

specifies the statistics for the procedure to compute. If you do not specify any statistic-keywords, PROC SURVEYMEANS computes the NOBS, MEAN, STDERR, and CLM statistics by default.

The statistics produced depend on the type of the analysis variable. If you name a numeric variable in the CLASS statement, then the procedure analyzes that variable as a categorical variable. The procedure always analyzes character variables as categorical.

PROC SURVEYMEANS computes MIN, MAX, and RANGE for numeric variables but not for categorical variables. For numeric variables, the keyword MEAN produces the mean, but for categorical variables it produces the proportion in each category or level. Also for categorical variables, the keyword NOBS produces the number of observations for each variable level, and the keyword NMISS produces the number of missing observations for each level. If you request the keyword NCLUSTER for a categorical variable, PROC SURVEYMEANS displays for each level the number of clusters with observations in that level. PROC SURVEYMEANS computes SUMWGT in the same way for both categorical and numeric variables, as the sum of the weights over all nonmissing observations.

PROC SURVEYMEANS performs univariate analysis, analyzing each variable separately. Thus, the number of nonmissing and missing observations may not be the same for all analysis variables.

If you use the keyword RATIO without the keyword MEAN, the keyword MEAN is implied.

Other available statistics computed for a ratio are N, NCLU, SUMWGT, RATIO, STDERR, DF, T, PROBT, and CLM, as listed below. If no statistics are requested, the procedure will compute the ratio and its standard error by default for a RATIO statement.

The valid statistic-keywords are as follows:

ALL	all statistics listed
CLM	$100(1-\alpha)$% confidence limits for the MEAN, where α is determined by the ALPHA= option, and the default is $\alpha = 0.05$
CLSUM	$100(1-\alpha)$% confidence limits for the SUM, where α is determined by the ALPHA= option, and the default is $\alpha = 0.05$
CV	coefficient of variation for MEAN
CVSUM	coefficient of variation for SUM
DF	degrees of freedom for the *t* test
MAX	maximum value
MEAN	mean for a numeric variable, or the proportion in each category for a categorical variable
MIN	minimum value
NCLUSTER	number of clusters
NMISS	number of missing observations
NOBS	number of nonmissing observations
RANGE	range, MAX−MIN
RATIO	ratio of means or proportions
STD	standard deviation of the SUM. When you request SUM, the procedure computes STD by default.
STDERR	standard error of the MEAN or RATIO. When you request MEAN or RATIO, the procedure computes STDERR by default.
SUM	weighted sum, $\sum w_i y_i$, or estimated population total when the appropriate sampling weights are used
SUMWGT	sum of the weights, $\sum w_i$
T	*t*-value and its corresponding *p*-value with DF degrees of freedom for $$H_0 : \theta = 0$$ where θ is the population mean or the population ratio
VAR	variance of the MEAN or RATIO
VARSUM	variance of the SUM

RATIO Statement

RATIO < *'label'* > *variables* / *variables* **;**

The RATIO statement requests ratio analysis for means or proportions of analysis variables. A ratio statement names the variables whose means will be used as numerators or denominators in a ratio. Variables appearing before the slash (/), called *numerator variables*, are used for numerators. Variables appearing after the slash (/), called *denominator variables*, are used for denominators. These *variables* can be any number of analysis variables, either continuous or categorical, in the input data set.

You can optionally specify a label for each RATIO statement to identify the ratios in the output. Labels must be enclosed in single quotes.

If a RATIO statement does not have any numerator variable or denominator variable specified, the RATIO statement will be ignored.

A numerator or denominator variable must be an analysis variable. That is, if there is a VAR statement, then a numerator or denominator variable must appear in the VAR statement. If there is no VAR statement, a numerator or denominator variable must be on the default analysis variable list. If a numerator or denominator variable is not an analysis variable, it is ignored.

The computation of ratios depends on whether the numerator and denominator variables are continuous or categorical.

For continuous variables, ratios are calculated with the mean of the variables. For example, for continuous variables X, Y, Z, and T, the following RATIO statement requests the procedure to analyze the ratios $\bar{x}/\bar{z}$, $\bar{x}/\bar{t}$, $\bar{y}/\bar{z}$, and $\bar{y}/\bar{t}$:

```
ratio x y / z t;
```

If a continuous variable appears as both a numerator and a denominator variable, the ratio of this variable itself is ignored.

For categorical variables, ratios are calculated with the proportions for the categories of a categorical variable. For example, if categorical variable Gender has values "Male" and "Female," with proportions p_m = Pr(Gender="Male") and p_f = Pr(Gender="Female"), and Y is a continuous variable, then the following RATIO statement requests the procedure to analyze the ratios p_m/p_f, p_f/p_m, $\bar{y}/p_m$, and $\bar{y}/p_f$:

```
ratio Gender y / Gender;
```

If a categorical variable appears as both a numerator and a denominator variable, then the ratios of the proportions for all categories are computed, except the ratio of each category with itself.

You may have more than one RATIO statement. Each RATIO statement produces ratios independently using its own numerator and denominator variables. Each RATIO statement also produces its own ratio analysis table.

Available statistics for a ratio are

- N, number of observations used to compute the ratio
- NCLU, number of clusters
- SUMWGT, sum of weights
- RATIO, ratio
- STDERR, standard error of ratio
- VAR, variance of ratio
- T, *t*-value of ratio
- PROBT, *p*-value of *t*
- DF, degrees of freedom of *t*
- CLM, confidence limits of ratio

The procedure will calculate these statistics based on the statistic-keywords described on page 305, which you specified in the PROC statement. If a statistic-keyword is not appropriate for RATIO statement, that statistic-keyword is ignored. If no valid statistics are requested for a RATIO statement, the procedure will compute the ratio and its standard error by default.

When calculating the means or proportions for the numerator and denominator variables in a ratio, an observation is excluded if it has a missing value in either the continuous numerator variable or the denominator variable. An observation with missing values is also excluded for the categorical numerator or denominator variables, unless the MISSING option is used.

Details

Statistical Computations

Definitions and Notation

For a stratified clustered sample design, together with the sampling weights, the sample can be represented by an $n \times (P+1)$ matrix

$$\begin{aligned}(\mathbf{w}, \mathbf{Y}) &= (w_{hij}, \mathbf{y}_{hij}) \\ &= \left(w_{hij}, y_{hij}^{(1)}, y_{hij}^{(2)}, \ldots, y_{hij}^{(P)}\right)\end{aligned}$$

where

- $h = 1, 2, \ldots, H$ is the stratum number, with a total of H strata

- $i = 1, 2, \ldots, n_h$ is the cluster number within stratum h, with a total of n_h clusters
- $j = 1, 2, \ldots, m_{hi}$ is the unit number within cluster i of stratum h, with a total of m_{hi} units
- $p = 1, 2, \ldots, P$ is the analysis variable number, with a total of P variables
- $n = \sum_{h=1}^{H} \sum_{i=1}^{n_h} m_{hi}$ is the total number of observations in the sample
- w_{hij} denotes the sampling weight for observation j in cluster i of stratum h
- $\mathbf{y}_{hij} = \left(y_{hij}^{(1)}, y_{hij}^{(2)}, \ldots, y_{hij}^{(P)}\right)$ are the observed values of the analysis variables for observation j in cluster i of stratum h, including both the values of numerical variables and the values of indicator variables for levels of categorical variables.

For a categorical variable C, let l denote the number of levels of C, and denote the level values as $c_1, c_2, \ldots, c_l$. Then there are l indicator variables associated with these levels. That is, for level $C = c_k$ $(k = 1, 2, \ldots, l)$, a $y^{(q)}$ $(q \in \{1, 2, \ldots, P\})$ contains the values of the indicator variable for the category $C = c_k$, with the value of observation j in cluster i of stratum h:

$$y_{hij}^{(q)} = I_{\{C=c_k\}}(h, i, j) = \begin{cases} 1 & \text{if } C_{hij} = c_k \\ 0 & \text{otherwise} \end{cases}$$

Therefore, the total number of analysis variables, P, is the total number of numerical variables plus the total number of levels of all categorical variables.

Also, f_h denotes the sampling rate for stratum h. You can use the TOTAL= option or the RATE= option to input population totals or sampling rates. If you input stratum totals, PROC SURVEYMEANS computes f_h as the ratio of the stratum sample size to the stratum total. If you input stratum sampling rates, PROC SURVEYMEANS uses these values directly for f_h. If you do not specify the TOTAL= option or the RATE= option, then the procedure assumes that the stratum sampling rates f_h are negligible, and a finite population correction is not used when computing variances.

This notation is also applicable to other sample designs. For example, for a sample design without stratification, you can let $H = 1$; for a sample design without clusters, you can let $m_{hi} = 1$ for every h and i.

Ratio

When you use a RATIO statement, the procedure produces statistics requested by the statistics-keywords in the PROC SURVEYMEANS statement.

Suppose that you want to calculate the ratio of variable Y over variable X. Let x_{hij} be the value of variable X for the jth member in cluster i in the hth stratum.

The ratio of Y over X is

$$\widehat{R} = \frac{\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} w_{hij}\, y_{hij}}{\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} w_{hij}\, x_{hij}}$$

PROC SURVEYMEANS uses the Taylor series expansion method to estimate the variance of the ratio $\widehat{R}$ as

$$\widehat{V}(\widehat{R}) = \sum_{h=1}^{H} \frac{n_h(1-f_h)}{n_h-1} \sum_{i=1}^{n_h} (g_{hi\cdot} - \bar{g}_{h\cdot\cdot})^2$$

where

$$g_{hi\cdot} = \frac{\sum_{j=1}^{m_{hi}} w_{hij}\,(y_{hij} - x_{hij}\widehat{R})}{\sum_{h=1}^{H}\sum_{i=1}^{n_h}\sum_{j=1}^{m_{hi}} w_{hij}\,x_{hij}}$$

$$\bar{g}_{h\cdot\cdot} = \left(\sum_{i=1}^{n_h} g_{hi\cdot}\right) / n_h$$

The standard error of the ratio is the square root of the estimated variance.

$$\text{StdErr}(\widehat{R}) = \sqrt{\widehat{V}(\widehat{R})}$$

Coefficient of Variation

If you specify the keyword CV, PROC SURVEYMEANS computes the coefficient of variation, which is the ratio of the standard error of the mean to the estimated mean.

$$cv(\bar{Y}) = \text{StdErr}(\hat{\bar{Y}}) \;/\; \hat{\bar{Y}}$$

If you specify the keyword CVSUM, PROC SURVEYMEANS computes the coefficient of variation for the estimated total, which is the ratio of the standard deviation of the sum to the estimated total.

$$cv(Y) = \text{Std}(\widehat{Y}) \;/\; \widehat{Y}$$

Displayed Output

If you have a RATIO statement, the procedure displays a "Ratio Analysis" table.

Ratio Analysis

The "Ratio Analysis" table displays all of the statistics that you request with statistic-keywords in the PROC statement described on page 305. If you do not specify any statistic-keywords, then by default this table displays the ratio and its standard error. The "Ratio Analysis" table may contain the following information for each ratio, depending on which statistic-keywords you request:

- Numerator, which identifies the numerator variable of the ratio
- Denominator, which identifies the denominator variable of the ratio
- N, which is the number of observations used in the ratio analysis
- Number of Clusters
- Sum of Weights

- DF, which is the degrees of freedom for the t test
- Ratio
- Std Error of Ratio, which is the standard error of the ratio
- Var of Ratio, which is the variance of the ratio
- t Value, for testing H_0 : population RATIO $= 0$
- Pr $> \mid t \mid$, which is the two-sided p-value for the t test
- Lower and Upper $100(1-\alpha)$% CL for Ratio, which are confidence limits for the Ratio

When you use the ODS output statement to create an output data set, if you use labels for your RATIO statement, these labels are saved in a variable RatioLabel in the output data set.

ODS Table Names

PROC SURVEYMEANS assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in the following table.

Table 14.1. ODS Tables Produced in PROC SURVEYMEANS

ODS Table Name	Description	Statement	Option
ClassVarInfo	Class level information	CLASS	default
Domain	Statistics in domains	DOMAIN	default
Ratio	Statistics for ratios	RATIO	default
Statistics	Statistics	PROC	default
StrataInfo	Stratum information	STRATA	LIST
Summary	Data summary	PROC	default

Example

Example 14.1. Ratio Analysis

You are studying profiles of the 800 top-performing companies to provide information on their impact on the economy. In particular, suppose you are interested in the profit per employee and the sale per employee among these companies. You select a sample of 66 companies with unequal probability across market types. The data for these 66 companies is saved in the following data set:

```
data Company;
   length Type $14;
   input Type$ Asset Sale Value Profit Employee Weight;
   datalines;
Other            2764.0  1828.0  1850.3   144.0   18.7   9.6
Energy          13246.2  4633.5  4387.7   462.9   24.3  42.6
Finance          3597.7   377.8    93.0    14.0    1.1  12.2
```

Transportation	6646.1	6414.2	2377.5	348.2	47.1	21.8
HiTech	1068.4	1689.8	1430.2	72.9	4.6	4.3
Manufacturing	1125.0	1719.4	1057.5	98.1	20.4	4.5
Other	1459.0	1241.4	452.7	24.5	20.1	5.5
Finance	2672.3	262.5	296.2	23.1	2.2	9.3
Finance	311.0	566.2	932.0	52.8	2.7	1.9
Energy	1148.6	1014.6	485.1	60.6	4.0	4.5
Finance	5327.0	572.4	372.9	25.2	4.2	17.7
Energy	1602.7	678.4	653.0	75.6	2.8	6.0
Energy	5808.8	1288.4	2007.0	318.8	5.9	19.2
Medical	268.8	204.4	820.9	45.6	3.7	1.8
Transportation	5222.6	2627.8	1910.0	245.6	22.8	17.4
Other	872.7	1419.4	939.3	69.7	12.2	3.7
Retail	4461.7	8946.8	4662.7	289.0	132.1	15.0
HiTech	6719.2	6942.0	8240.2	381.3	85.8	22.1
Retail	833.4	1538.8	1090.3	64.9	15.4	3.5
Finance	415.9	167.3	1126.8	56.8	0.7	2.2
HiTech	442.4	1139.9	1039.9	57.6	22.7	2.3
Other	801.5	1157.0	664.2	56.9	15.5	3.4
Finance	4954.8	468.8	366.4	41.7	3.0	16.5
Finance	2661.9	257.9	181.1	21.2	2.1	9.3
Finance	5345.8	530.1	337.4	36.4	4.3	17.8
Energy	3334.3	1644.7	1407.8	157.6	6.4	11.4
Manufacturing	1826.6	2671.7	483.2	71.3	25.3	6.7
Retail	618.8	2354.7	767.7	58.6	19.0	2.9
Retail	1529.1	6534.0	826.3	58.3	65.8	5.7
Manufacturing	4458.4	4824.5	3132.1	28.9	67.0	15.0
HiTech	5831.7	6611.1	9464.7	459.6	86.7	19.3
Medical	6468.3	4199.2	3170.4	270.1	59.5	21.3
Energy	1720.7	473.1	811.1	86.6	1.6	6.3
Energy	1679.7	1379.9	721.1	91.8	4.5	6.2
Retail	4018.2	16823.4	2038.3	178.1	162.0	13.6
Other	227.1	575.8	1083.8	62.6	1.9	1.6
Finance	3872.8	362.0	209.3	27.6	2.4	13.1
Retail	3359.3	4844.7	2651.4	224.1	75.6	11.5
Energy	1295.6	356.9	180.8	162.3	0.6	5.0
Energy	1658.0	626.6	688.0	126.0	3.5	6.1
Finance	12156.7	1345.5	680.7	106.6	9.4	39.2
HiTech	3982.6	4196.0	3946.8	313.9	64.3	13.5
Finance	8760.7	886.4	1006.9	90.0	7.5	28.5
Manufacturing	2362.2	3153.3	1080.0	137.0	25.2	8.4
Transportation	2499.9	3419.0	992.6	47.2	25.3	8.8
Energy	1430.4	1610.0	664.3	77.7	3.5	5.4
Energy	13666.5	15465.4	2736.7	411.4	26.6	43.9
Manufacturing	4069.3	4174.7	2907.6	289.2	38.2	13.7
Energy	2924.7	711.9	1067.8	146.7	3.4	10.1
Transportation	1262.1	1716.0	364.3	71.2	14.5	4.9
Medical	684.4	672.9	287.4	61.8	6.0	3.1
Energy	3069.3	1719.0	1439.0	196.4	4.9	10.6
Medical	246.5	318.8	924.1	43.8	3.1	1.7
Finance	11562.2	1128.5	580.4	64.2	6.7	37.3
Finance	9316.0	1059.4	816.5	95.9	8.0	30.2
Retail	1094.3	3848.0	563.3	29.4	44.7	4.4
Retail	1102.1	4878.3	932.4	65.2	47.3	4.4

```
HiTech           466.4   675.8   845.7    64.5    5.2   2.4
Manufacturing  10839.4  5468.7  1895.4   232.8   47.8  35.0
Manufacturing    733.5  2135.3    96.6    10.9    2.7   3.2
Manufacturing  10354.2 14477.4  5607.2   321.9  188.5  33.5
Energy          1902.1  2697.9   329.3    34.2    2.2   6.9
Other           2245.2  2132.2  2230.4   198.9    8.0   8.0
Transportation   949.4  1248.3   298.9    35.4   10.4   3.9
Retail          2834.4  2884.6   458.2    41.2   49.8   9.8
Retail          2621.1  6173.8  1992.7   183.7  115.1   9.2
;
```

For each company in your sample, you have the following information in the data set:

Variable	Description
Type	the type of market of the company
Asset	the company's assets in millions of dollars
Sale	the market value of the company in millions of dollars
Profit	the company's profit in millions of dollars
Employee	the number of employees in thousands
Weight	the sampling weight

The following SAS statements illustrate how you can use PROC SURVEYMEANS to estimate these ratios:

```
title1 'Ratio Analysis in Top Companies Profile Study';
proc surveymeans data=Company total=800 ratio;
   var Profit Sale Employee;
   weight Weight;
   ratio Profit Sale / Employee;
run;
```

The RATIO statement requests the ratio of the profit and the sale to the number of employees.

Output 14.1.1. Estimate Ratios

```
          Ratio Analysis in Top Companies Profile Study

                The SURVEYMEANS Procedure

                     Ratio Analysis

Numerator Denominator        Ratio         Std Err
-------------------------------------------------
Sale      Employee      114.332497       20.502742
Profit    Employee        5.119698        1.058939
-------------------------------------------------
```

Output 14.1.1 shows the estimated ratios and their standard errors. Because the profit and the sale figures are in millions of dollars, and the employee figures in thousands, the profit per employee is estimated as $5,120 with a standard error of $1,059, and the sale per employee is $114,332 with a standard error of $20,503.

References

Cochran, W.G. (1977), *Sampling Techniques*, Third Edition, New York: John Wiley & Sons, Inc.

Fuller, W.A. (1975), "Regression Analysis for Sample Survey," *Sankhyā*, 37, Series C, Pt. 3, 117–132.

Fuller, W.A., Kennedy, W., Schnell, D., Sullivan, G., and Park, H.J. (1989), *PC CARP*, Ames, IA: Statistical Laboratory, Iowa State University.

Hansen, M.H., Hurwitz, W.N., and Madow, W.G. (1953), *Sample Survey Methods and Theory*, Volumes I and II, New York: John Wiley & Sons, Inc.

Hidiroglou, M.A., Fuller, W.A., and Hickman, R.D. (1980), *SUPER CARP*, Ames, IA: Statistical Laboratory, Iowa State University.

Kish, L. (1965), *Survey Sampling*, New York: John Wiley & Sons, Inc.

Chapter 15
The TRANSREG Procedure

Chapter Table of Contents

Chapter 15
The TRANSREG Procedure

Overview

The Box-Cox (1964) transformation has the form

$$\begin{array}{ll} (y^{\lambda}-1)/\lambda & \lambda \neq 0 \\ \log(y) & \lambda = 0 \end{array}$$

This family of transformations of the positive dependent variable y is controlled by the parameter λ. Transformations linearly related to square root, inverse, quadratic, cubic, and so on are all special cases. The limit as λ approaches 0 is the log transformation. More generally, Box-Cox transformations of the following form can be fit:

$$\begin{array}{ll} ((y+c)^{\lambda}-1)/(\lambda g) & \lambda \neq 0 \\ \log(y+c)/g & \lambda = 0 \end{array}$$

By default, $c = 0$. The parameter c can be used to rescale y so that it is strictly positive. By default, $g = 1$. Alternatively, g can be $\dot{y}^{\lambda-1}$ where $\dot{y}$ is the geometric mean of y.

The new BOXCOX transformation in the TRANSREG procedure can be used to perform a Box-Cox transformation of the dependent variable. You can specify a list of power parameters using the LAMBDA= transformation option. By default, LAMBDA=-3 TO 3 BY 0.25. The procedure chooses the optimal power parameter using a maximum likelihood criterion (Draper and Smith 1981, pp. 225-226). You can specify the PARAMETER=c transformation option when you want to shift the values of y, usually to avoid negatives. To divide by $\dot{y}^{\lambda-1}$, specify the GEOMETRICMEAN transformation option. For example,

```
proc transreg;
   model BoxCox(y / lambda=-2 to 2 by 0.1
                    parameter=2 geometricmean) =
         identity(x1-x5);
   output out=results;
   run;
```

Syntax

New Nonoptimal Transformation

Nonoptimal transformations are computed before the iterative algorithm begins. Nonoptimal transformations create a single new transformed variable that replaces the original variable. The new variable is not transformed by the subsequent iterative algorithms (except for a possible linear transformation with missing value estimation).

BOXCOX
BOX
finds a Box-Cox transformation of the specified variables. The BOXCOX transformation can be used only with dependent variables. The ALPHA=, CLL=, CONVENIENT, GEOMETRICMEAN, LAMBDA=, and PARAMETER= transformation options can be used with the BOXCOX transformation.

Transformation Options (t-options)

The *t-options* are specified within the parentheses that enclose variables and are listed after a slash.

ALPHA=*p*
ALP=*p*
is a new transformation option for use with BOXCOX transformations. The ALPHA= option specifies the alpha for the confidence interval for the power parameter. By default, ALPHA=0.05.

CLL=*number-list*
is a new transformation option for use with BOXCOX transformations. CLL stands for Convenient Lambda List. When the confidence interval for the power parameter includes one of the values in this list, PROC TRANSREG reports it and can optionally use the convenient power parameter instead of the more optimal power parameter. The default is CLL=1.0 0.0 0.5 -1.0 -0.5 2.0 -2.0 3.0 -3.0. By default, a linear transformation is preferred over log, square root, inverse, inverse square root, quadratic, inverse quadratic, cubic, and inverse cubic. If you specify the CONVENIENT option, then PROC TRANSREG uses the first convenient power parameter in the list that is in the confidence interval. For example, if the optimal power parameter is 0.25 and 0.0 is in the confidence interval but not 1.0, then the convenient power parameter is 0.0.

CONVENIENT
CON
is a new transformation option for use with BOXCOX transformations. The CONVENIENT option specifies that a power parameter from the CLL= list is to be used for the final transformation instead of the LAMBDA= value if a CLL= value is in the confidence interval. See CLL= for more information on the usage of this option.

GEOMETRICMEAN
GEO

is a new transformation option for use with BOXCOX transformations. The GEOMETRICMEAN option specifies that the transformation is to be divided by $\dot{y}^{\lambda-1}$ where $\dot{y}$ is the geometric mean of the variable to be transformed. This form of the Box-Cox transformation essentially converts the transformation back to original units and hence allows direct comparison of the residual sums of squares for models with different power parameters.

LAMBDA=*number-list*
LAM=*number-list*

is a new transformation option for use with BOXCOX transformations. It provides a list of power parameters. The default is LAMBDA=-3 TO 3 BY 0.25. PROC TRANSREG tries each power parameter in the list and picks the best one. However, when CONVENIENT is specified, if there is a convenient power parameter in the confidence interval, it will be chosen instead.

PARAMETER=*n*
PAR=*n*

has been enhanced to accept a constant to add to each value of the BOXCOX variable before a Box-Cox transformation. By default, PARAMETER=0 for the BOXCOX transformation.

Algorithm Options (a-options)

DETAIL
DET

The output of the DETAIL option has been modified to include Box-Cox results. The new output includes: "Lambda Used," the power parameter used in the final transformation; "Lambda," the power parameter selected from the LAMBDA= option; "Log Likelihood," the log likelihood for the power parameter; "Conv. Lambda," the first convenient power parameter found in the confidence interval (or blank for none found); "Conv. Lambda LL," the log likelihood for the convenient power parameter; "CI Limit," the limit on the confidence limit; and "Alpha," the alpha for the confidence limit. The power parameter used will either come from the LAMBDA= list or it may come from the CLL= list when a convenient power parameter is in the confidence interval.

Details

Here are some examples of usage of the new LAMBDA= option:

```
model BoxCox(y / lambda=0) = identity(x1-x5);
model BoxCox(y / lambda=-2 to 2 by 0.1) = identity(x1-x5);
model BoxCox(y) = identity(x1-x5);
```

In the first example

```
model BoxCox(y / lambda=0) = identity(x1-x5);
```

LAMBDA=0 specifies a Box-Cox transformation with a power parameter of 0. Since a single value of 0 was specified for LAMBDA=, there is no difference between the following models:

```
model BoxCox(y / lambda=0) = identity(x1-x5);
model log(y) = identity(x1-x5);
```

In the second example

```
model BoxCox(y / lambda=-2 to 2 by 0.1) = identity(x1-x5);
```

there is a list of power parameters specified. This tells PROC TRANSREG to find a Box-Cox transformation before the usual iterations begin. PROC TRANSREG tries each power parameter in the list and picks the best transformation. A maximum likelihood approach (Draper and Smith 1981, pp. 225-226) is used. Note that this is quite different from TRANSREG's usual approach of iteratively finding optimal transformations. It is analogous to SMOOTH, RANK, and the other nonoptimal transformations that are performed before the iterations begin.

In the third example

```
model BoxCox(y) = identity(x1-x5);
```

the default list of -3 TO 3 BY 0.25 is used.

The procedure prints the optimal power parameter, a confidence interval on the power parameter (using the ALPHA= transformation option), a "convenient" power parameter (selected from the CLL= option list), and the log likelihood for each power parameter tried.

Examples

Example 15.1. Basic Box-Cox Transformations

This example illustrates finding a Box-Cox transformation of some artificial data. Data were generated from the model

$y = e^{x+\epsilon}$

where $\epsilon \sim \mathrm{N}(0, 1)$. The transformed data can be fit with a linear model

$\log(y) = x + \epsilon$

```
title 'Basic Box-Cox Example';

data x;
   do x = 1 to 8 by 0.025;
      y = exp(x + normal(7));
      output;
      end;
   run;

proc transreg data=x ss2 details;
   title2 'Defaults';
   model boxcox(y) = identity(x);
   run;
```

Output 15.1.1. Basic Box-Cox Example, Default Output

```
                    Basic Box-Cox Example
                           Defaults

                    The TRANSREG Procedure

                  Transformation Information
                        for BoxCox(y)

                Lambda      R-Square    Log Like

                 -3.00          0.03    -4601.01
                 -2.75          0.04    -4266.08
                 -2.50          0.04    -3934.11
                 -2.25          0.05    -3605.75
                 -2.00          0.06    -3281.88
                 -1.75          0.07    -2963.74
                 -1.50          0.10    -2653.14
                 -1.25          0.14    -2352.72
                 -1.00          0.21    -2066.32
                 -0.75          0.34    -1799.25
                 -0.50          0.52    -1558.55
                 -0.25          0.71    -1360.28
                  0.00 +        0.79    -1275.31 <
                  0.25          0.70    -1382.62
                  0.50          0.51    -1589.03
                  0.75          0.34    -1834.53
                  1.00          0.22    -2105.88
                  1.25          0.15    -2397.35
                  1.50          0.11    -2704.64
                  1.75          0.08    -3024.24
                  2.00          0.06    -3353.38
                  2.25          0.05    -3689.91
                  2.50          0.04    -4032.18
                  2.75          0.03    -4378.97
                  3.00          0.03    -4729.37

               < - Best Lambda
               * - Confidence Interval
               + - Convenient Lambda
```

PROC TRANSREG correctly selects the log transformation $\lambda = 0$, with a narrow confidence interval. The maximum of the log likelihood function is flagged with the

less-than sign (<), and the convenient power parameter of $\lambda = 0$ in the confidence interval is flagged by the plus sign (+). The rest of the output is shown next.

Output 15.1.2. Basic Box-Cox Example, Default Output

```
                          Basic Box-Cox Example
                                Defaults

                          The TRANSREG Procedure

       TRANSREG Univariate Algorithm Iteration History for BoxCox(y)

  Iteration    Average    Maximum                Criterion
     Number     Change     Change    R-Square       Change    Note
  ------------------------------------------------------------------------
          1    0.00000    0.00000     0.79064                Converged

  Algorithm converged.


                    Model Statement Specification Details

 Type  DF Variable    Description     Value

 Dep    1 BoxCox(y)   Lambda Used     0
                      Lambda          0
                      Log Likelihood  -1275.3
                      Conv. Lambda    0
                      Conv. Lambda LL -1275.3
                      CI Limit        -1277.2
                      Alpha           0.05

 Ind    1 Identity(x) DF              1


        Univariate ANOVA Table Based on the Usual Degrees of Freedom

                                 Sum of        Mean
   Source                 DF     Squares      Square    F Value    Liberal p

   Model                   1    1145.884    1145.884    1053.66    >= <.0001
   Error                 279     303.421       1.088
   Corrected Total       280    1449.305

   The above statistics are not adjusted for the fact that the dependent
   variable was transformed and so are generally liberal.


                Root MSE           1.04285    R-Square    0.7906
                Dependent Mean     4.49653    Adj R-Sq    0.7899
                Coeff Var         23.19225    Lambda      0.0000


      Univariate Regression Table Based on the Usual Degrees of Freedom

                                    Type II
                                     Sum of       Mean
Variable        DF    Coefficient    Squares     Square    F Value    Liberal p

Intercept        1     0.01551366       0.01       0.01       0.01    >= 0.9185
Identity(x)      1     0.99578183    1145.88    1145.88    1053.66    >= <.0001

The above statistics are not adjusted for the fact that the dependent variable
was transformed and so are generally liberal.
```

This next example uses several options. The LAMBDA= option specifies power parameters sparsely from -2 to -0.5 and from 0.5 to 2 just to get the general shape of the log likelihood function in that region. Between -0.5 and 0.5, more power parameters are tried. The CONVENIENT option is specified so that if a power parameter such as $\lambda = 1$ or $\lambda = 0$ is found in the confidence interval, it will be used instead of the optimal power parameter. PARAMETER=2 is specified to add 2 to each y before performing the transformations. ALPHA=0.00001 specifies a wide confidence interval.

```
proc transreg data=x ss2 details;
   title2 'Several Options Demonstrated';
   model boxcox(y / lambda=-2 -1 -0.5 to 0.5 by 0.05 1 2
                    convenient
                    parameter=2
                    alpha=0.00001)
       = identity(x);
   run;
```

Output 15.1.3. Basic Box-Cox Example, Several Options Demonstrated

```
                    Basic Box-Cox Example
                 Several Options Demonstrated

                    The TRANSREG Procedure

                  Transformation Information
                        for BoxCox(y)

                Lambda      R-Square    Log Like

                -2.000          0.22    -2583.73
                -1.000          0.45    -1779.35
                -0.500          0.67    -1439.82
                -0.450          0.70    -1410.51
                -0.400          0.72    -1382.74
                -0.350          0.74    -1356.92
                -0.300          0.76    -1333.59
                -0.250          0.77    -1313.42
                -0.200          0.79    -1297.21
                -0.150          0.79    -1285.83 *
                -0.100          0.80    -1280.09 <
                -0.050          0.80    -1280.63 *
                 0.000 +        0.79    -1287.71 *
                 0.050          0.78    -1301.19
                 0.100          0.76    -1320.56
                 0.150          0.74    -1345.09
                 0.200          0.72    -1373.99
                 0.250          0.69    -1406.51
                 0.300          0.65    -1442.02
                 0.350          0.62    -1480.02
                 0.400          0.58    -1520.13
                 0.450          0.54    -1562.05
                 0.500          0.50    -1605.57
                 1.000          0.22    -2105.88
                 2.000          0.06    -3320.36

               < - Best Lambda
               * - Confidence Interval
               + - Convenient Lambda
```

The results show that the optimal power parameter is -0.1 but 0 is in the confidence interval, hence a log transformation is chosen. The rest of the output is shown next.

Output 15.1.4. Basic Box-Cox Example, Several Options Demonstrated

```
                        Basic Box-Cox Example
                     Several Options Demonstrated

                        The TRANSREG Procedure

       TRANSREG Univariate Algorithm Iteration History for BoxCox(y)

  Iteration    Average    Maximum                Criterion
     Number     Change     Change    R-Square       Change    Note
  ------------------------------------------------------------------------
          1    0.00000    0.00000     0.79238                 Converged

  Algorithm converged.


                   Model Statement Specification Details

Type  DF Variable    Description     Value

Dep    1 BoxCox(y)   Lambda Used     0
                     Lambda          -0.1
                     Log Likelihood  -1280.1
                     Conv. Lambda    0
                     Conv. Lambda LL -1287.7
                     CI Limit        -1289.9
                     Alpha           0.00001
                     Parameter       2
                     Options         Convenient Lambda Used

Ind    1 Identity(x) DF              1
```

```
                          Basic Box-Cox Example
                       Several Options Demonstrated

                          The TRANSREG Procedure

        Univariate ANOVA Table Based on the Usual Degrees of Freedom

                                 Sum of        Mean
    Source                DF     Squares      Square    F Value    Liberal p

    Model                  1     999.438    999.4381    1064.82    >= <.0001
    Error                279     261.868      0.9386
    Corrected Total      280    1261.306

    The above statistics are not adjusted for the fact that the dependent
    variable was transformed and so are generally liberal.


                Root MSE           0.96881    R-Square    0.7924
                Dependent Mean     4.61429    Adj R-Sq    0.7916
                Coeff Var         20.99591    Lambda      0.0000


       Univariate Regression Table Based on the Usual Degrees of Freedom

                                    Type II
                                     Sum of       Mean
Variable         DF    Coefficient    Squares     Square    F Value    Liberal p

Intercept         1     0.42939328      8.746      8.746       9.32    >= 0.0025
Identity(x)       1     0.92997620    999.438    999.438    1064.82    >= <.0001

The above statistics are not adjusted for the fact that the dependent variable
was transformed and so are generally liberal.
```

Example 15.2. Graphical Displays of Box-Cox Transformations

This example shows how to make graphical displays of the Box-Cox transformation results. Plots include the log likelihood function with the confidence interval, root mean squared error as a function of the power parameter, R^2 as a function of the power parameter, the Box-Cox transformation of the variable y, the original scatter plot based on the untransformed data, and the new scatter plot based on the transformed data. Also, a condensed version of the log likelihood table with the confidence interval is printed.

```
title h=1 'Box-Cox Graphical Displays';

data x;
   input y x @@;
   datalines;
10.0  3.0  72.6  8.3  59.7  8.1  20.1  4.8  90.1  9.8   1.1  0.9
78.2  8.5  87.4  9.0   9.5  3.4   0.1  1.4   0.1  1.1  42.5  5.1
57.0  7.5   9.9  1.9   0.5  1.0 121.1  9.9  37.5  5.9  49.5  6.7
 8.3  1.8   0.6  1.8  53.0  6.7 112.8 10.0  40.7  6.4   5.1  2.4
73.3  9.5 122.4  9.9  87.2  9.4 121.2  9.9  23.1  4.3   7.1  3.5
12.4  3.3   5.6  2.7 113.0  9.6 110.5 10.0   3.1  1.5  52.4  7.9
80.4  8.1   0.6  1.6 115.1  9.1  15.9  3.1  56.5  7.3  85.4  9.8
```

```
32.5  5.8  43.0  6.2   0.1  0.8  21.8  5.2  15.2  3.5   5.2  3.0
 0.2  0.8  73.5  8.2   4.9  3.2   0.2  0.3  69.0  9.2   3.6  3.5
 0.2  0.9 101.3  9.9  10.0  3.7  16.9  3.0  11.2  5.0   0.2  0.4
80.8  9.4  24.9  5.7 113.5  9.7   6.2  2.1  12.5  3.2   4.8  1.8
80.1  8.3  26.4  4.8  13.4  3.8  99.8  9.7  44.1  6.2  15.3  3.8
 2.2  1.5  10.3  2.7  13.8  4.7  38.6  4.5  79.1  9.8  33.6  5.8
 9.1  4.5  89.3  9.1   5.5  2.6  20.0  4.8   2.9  2.9  82.9  8.4
 7.0  3.5  14.5  2.9  16.0  3.7  29.3  6.1  48.9  6.3   1.6  1.9
34.7  6.2  33.5  6.5  26.0  5.6  12.7  3.1   0.1  0.3  15.4  4.2
 2.6  1.8  58.6  7.9  81.2  8.1  37.2  6.9
;
```

The TRANSREG procedure is run to find the Box-Cox transformation. The lambda list is -2 TO 2 BY 0.01, which produces 401 lambdas. This many power parameters makes a nice graphical display with plenty of detail around the confidence interval. However, 401 values is a lot to print, so for this reason, the usual Box-Cox transformation information table is excluded from the printed output. Instead, it is output to a SAS data set using ODS so a sample of it can be printed. Just the confidence interval and the rows corresponding to power parameters that are multiples of 0.5 are printed. Null labels are provided for the columns that need to be printed without headers. The details table is also output to a SAS data set using ODS, since it contains information that will be incorporated into some of the plots.

```
* Fit Box-Cox model, output results to output data sets;
ods output boxcox=b details=d;
ods exclude boxcox;
proc transreg details data=x;
   model boxcox(y / convenient lambda=-2 to 2 by 0.01) = identity(x);
   output out=trans;
   run;

proc print noobs label data=b(drop=rmse);
   title2 'Confidence Interval';
   where ci ne ' ' or abs(lambda - round(lambda, 0.5)) < 1e-6;
   label convenient = '00'x ci = '00'x;
   run;
```

Output 15.2.1. Box-Cox Graphical Displays

Box-Cox Graphical Displays

The TRANSREG Procedure

TRANSREG Univariate Algorithm Iteration History for BoxCox(y)

Iteration Number	Average Change	Maximum Change	R-Square	Criterion Change	Note
1	0.00000	0.00000	0.95396		Converged

Algorithm converged.

Model Statement Specification Details

Type	DF	Variable	Description	Value
Dep	1	BoxCox(y)	Lambda Used	0.5
			Lambda	0.46
			Log Likelihood	-167.0
			Conv. Lambda	0.5
			Conv. Lambda LL	-168.3
			CI Limit	-169.0
			Alpha	0.05
			Options	Convenient Lambda Used
Ind	1	Identity(x)	DF	1

Box-Cox Graphical Displays
Confidence Interval

Lambda		R-Square	Log Like	
-2.00		0.14	-1030.56	
-1.50		0.17	-810.50	
-1.00		0.22	-602.53	
-0.50		0.39	-415.56	
0.00		0.78	-257.92	
0.41		0.95	-168.40	*
0.42		0.95	-167.86	*
0.43		0.95	-167.46	*
0.44		0.95	-167.19	*
0.45		0.95	-167.05	*
0.46		0.95	-167.04	<
0.47		0.95	-167.16	*
0.48		0.95	-167.41	*
0.49		0.95	-167.79	*
0.50	+	0.95	-168.28	*
0.51		0.95	-168.89	*
1.00		0.89	-253.09	
1.50		0.79	-345.35	
2.00		0.70	-435.01	

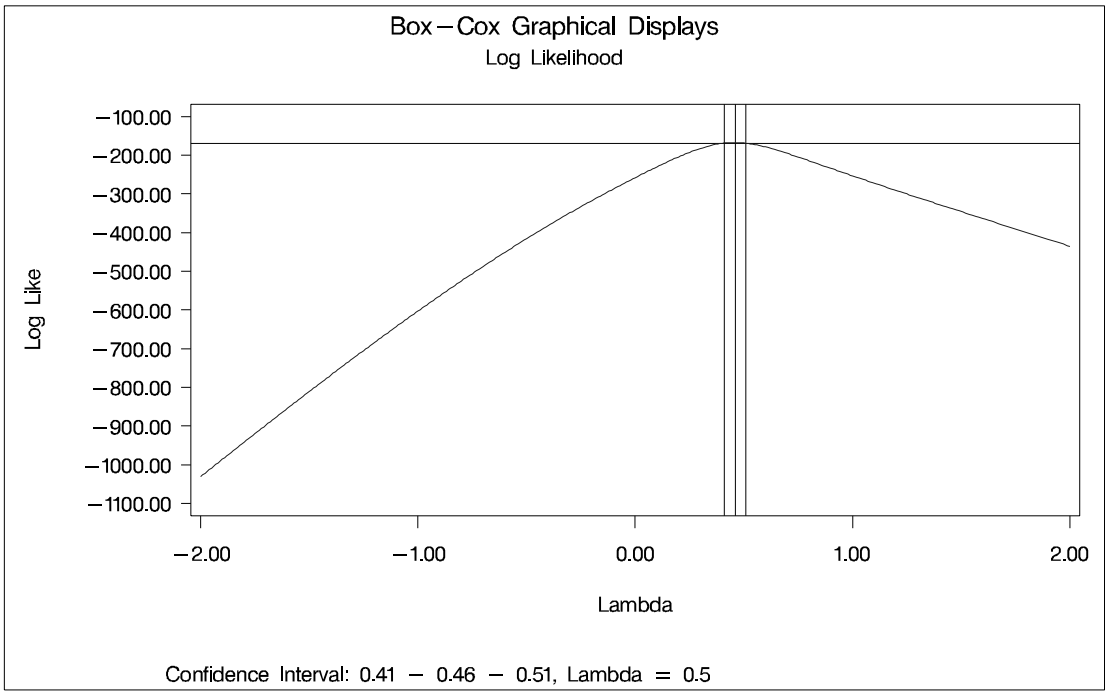
These next steps extract information from the Box-Cox transformation and details tables, and store the information in macro variables. The confidence interval limit from the details table provides a vertical axis reference line for the log likelihood plot. The convenient power parameter ('Lambda Used') is extracted from the footnote. The confidence interval is extracted from the confidence interval observations of the Box-Cox transformation table and will be used in the footnote and for horizontal axis reference lines in the log likelihood plot.

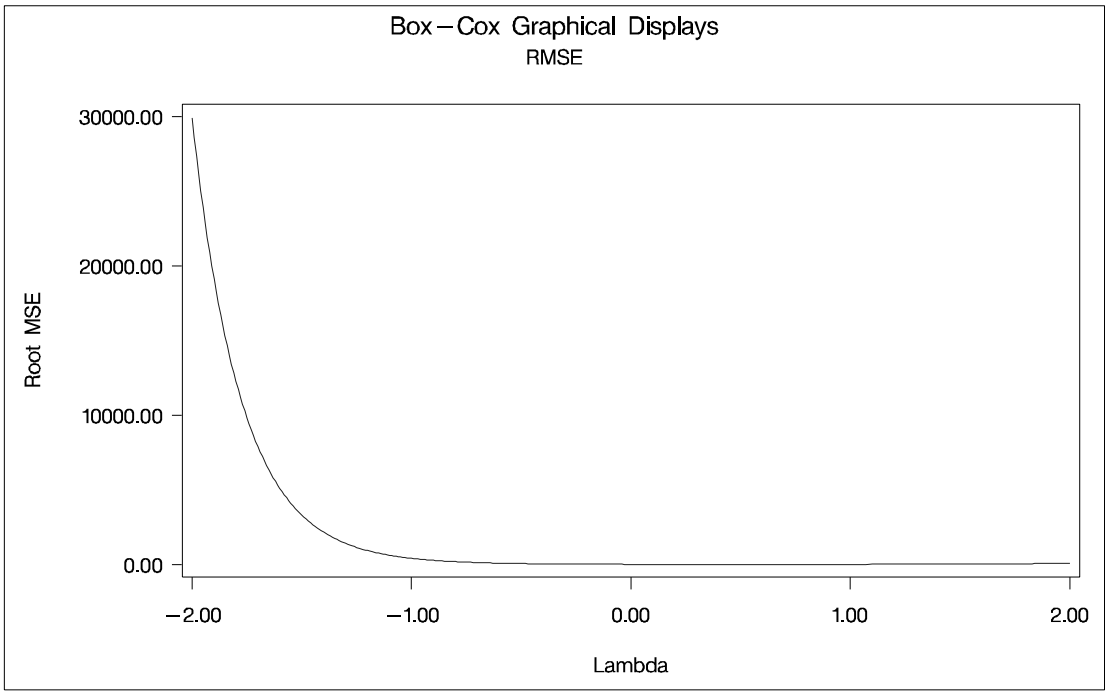
```
* Store values for reference lines;
data _null_;
   set d;
   if description = 'CI Limit'
      then call symput('vref',   formattedvalue);
   if description = 'Lambda Used'
      then call symput('lambda', formattedvalue);
   run;

data _null_;
   set b end=eof;
   where ci ne ' ';
   if _n_ = 1
      then call symput('href1', compress(put(lambda, best12.)));
   if ci  = '<'
      then call symput('href2', compress(put(lambda, best12.)));
   if eof
      then call symput('href3', compress(put(lambda, best12.)));
   run;
```

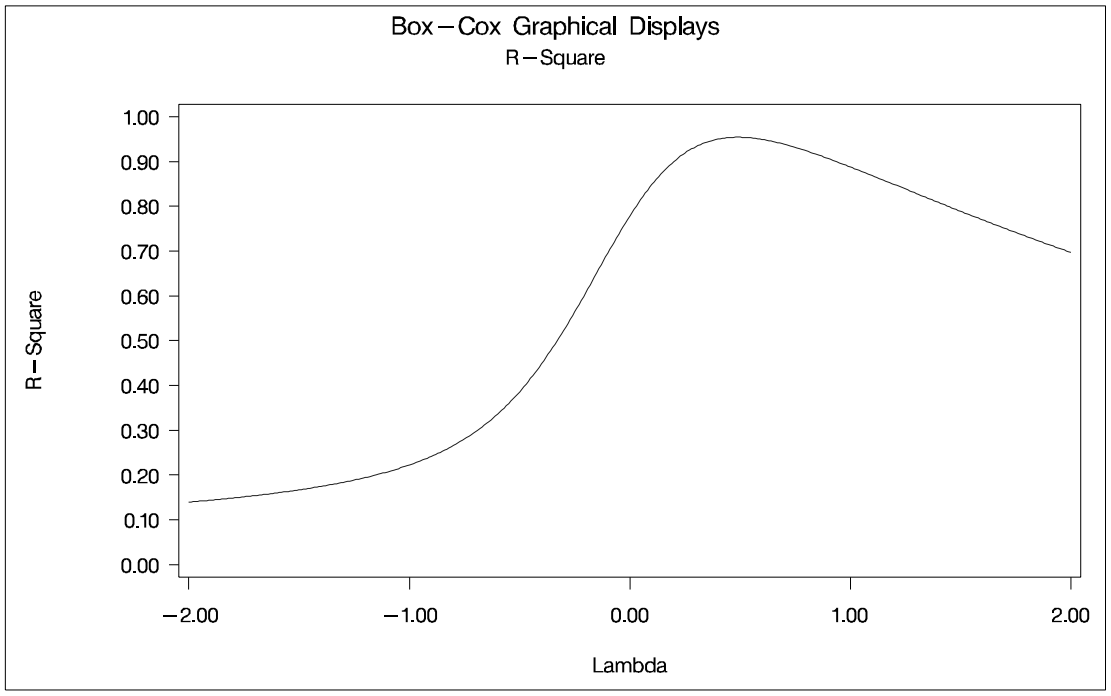
These steps plot the log likelihood, root mean square error, and R^2. The input data set is the Box-Cox transformation table, which was output using ODS.

```
* Plot log likelihood, confidence interval;
axis1 label=(angle=90 rotate=0) minor=none;
axis2 minor=none;
proc gplot data=b;
   title2 'Log Likelihood';
   plot loglike * lambda / vref=&vref href=&href1 &href2 &href3
                           vaxis=axis1 haxis=axis2 frame cframe=ligr;
   footnote "Confidence Interval: &href1 - &href2 - &href3, "
            "Lambda = &lambda";
   symbol v=none i=spline c=blue;
   run;

   footnote;
   title2 'RMSE';
   plot rmse * lambda / vaxis=axis1 haxis=axis2 frame cframe=ligr;
   run;

   title2 'R-Square';
   plot rsquare * lambda / vaxis=axis1 haxis=axis2 frame cframe=ligr;
   axis1 order=(0 to 1 by 0.1) label=(angle=90 rotate=0) minor=none;
   run; quit;
```

Output 15.2.2. Box-Cox Graphical Displays





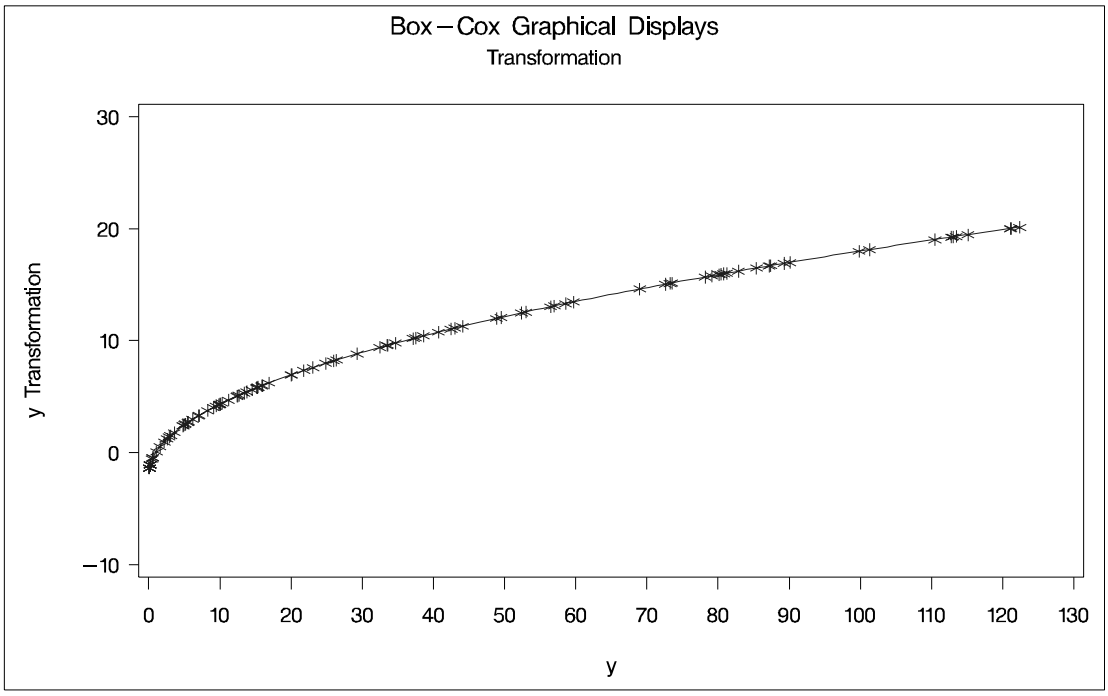


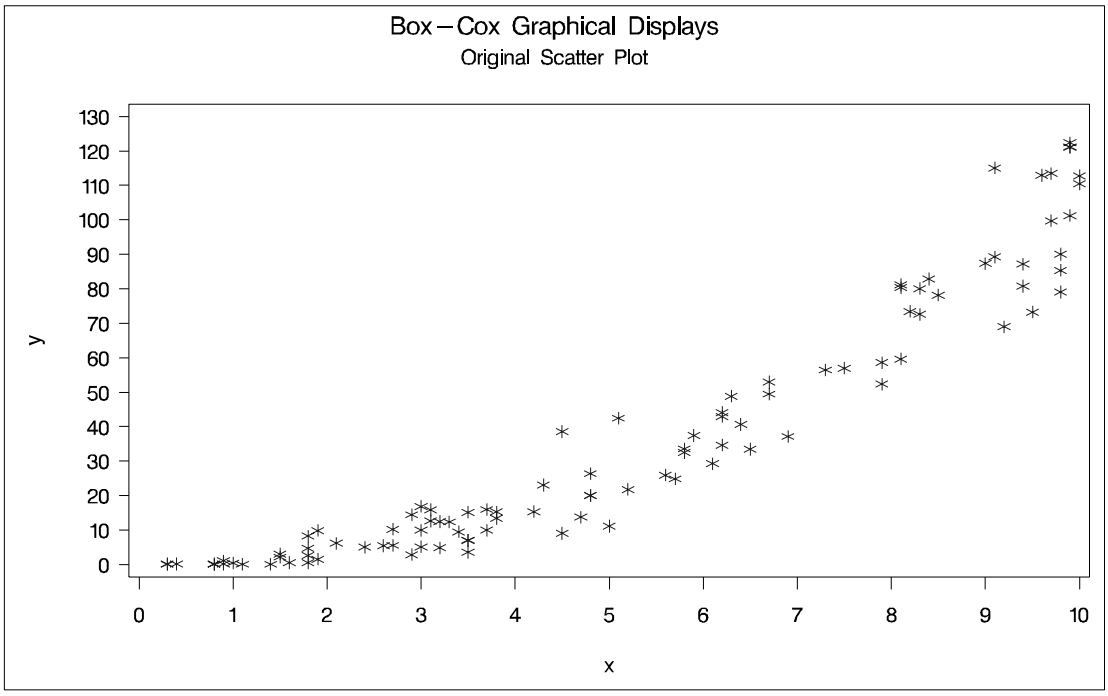
The optimal power parameter is 0.46, but since 0.5 is in the confidence interval, and since the CONVENIENT option was specified, the procedure chooses a square root transformation.

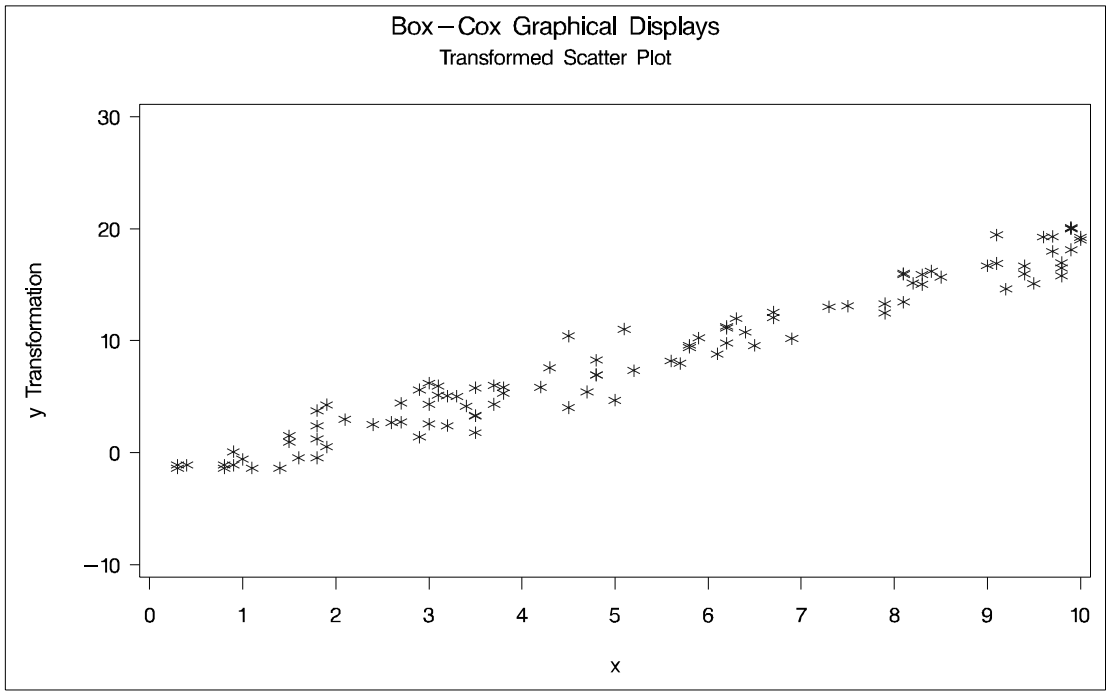
The next steps plot the transformation of y, the original scatter plot based on the untransformed data, and the new scatter plot based on the transformed data. The input data set is the ordinary output data set from PROC TRANSREG. The transformation of the variable y by default is ty.

```
axis1 label=(angle=90 rotate=0) minor=none;
axis2 minor=none;
proc gplot data=trans;
   title2 'Transformation';
   symbol i=splines v=star c=blue;
   plot ty * y / vaxis=axis1 haxis=axis2 frame cframe=ligr;
   run;

   title2 'Original Scatter Plot';
   symbol i=none v=star c=blue;
   plot y * x / vaxis=axis1 haxis=axis2 frame cframe=ligr;
   run;

   title2 'Transformed Scatter Plot';
   symbol i=none v=star c=blue;
   plot ty * x / vaxis=axis1 haxis=axis2 frame cframe=ligr;
   run; quit;
```

Output 15.2.3. Box-Cox Graphical Displays







The square root transformation makes the scatter plot more linear.

References

Box, G.E.P. and Cox, D.R. (1964), "An Analysis of Transformations," *Journal of the Royal Statistics Society*, B-26, 211–252.

Draper, N.R. and Smith, H. (1981), *Applied Regression Analysis*, Second Edition, New York: John Wiley & Sons, Inc.

Subject Index

S

T

X

Syntax Index

G

H

I

K

L

M

N

O

P

R

S

T

V

W

X