SAS/STAT® 9.3 User's Guide

Statistical Graphics Using ODS (Chapter)

SAS® Documentation

This document is an individual chapter from *SAS/STAT® 9.3 User's Guide*.

The correct bibliographic citation for the complete manual is as follows: SAS Institute Inc. 2011. *SAS/STAT® 9.3 User's Guide*. Cary, NC: SAS Institute Inc.

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

1st electronic book, July 2011

SAS® Publishing provides a complete selection of books and electronic products to help customers use SAS software to its fullest potential. For more information about our e-books, e-learning products, CDs, and hard-copy books, visit the SAS Publishing Web site at **support.sas.com/publishing** or call 1-800-727-3228.

Chapter 21

Statistical Graphics Using ODS

Contents

Introduction

Effective graphics are indispensable for modern statistical analysis. They reveal patterns, differences, and uncertainty that are not readily apparent in tabular output. Graphics provoke questions that stimulate deeper investigation, and they add visual clarity and rich content to reports and presentations.

In earlier SAS releases, creating graphs with statistical procedures typically required additional programming steps such as creating output data sets with the values to plot, modifying these data sets with a DATA step program, and using traditional SAS/GRAPH procedures to produce the plots.

ODS Graphics eliminates the need for additional programming. ODS Graphics is an extension of ODS (the Output Delivery System). ODS manages procedure output and lets you display it in a variety of destinations, such as HTML and RTF. With ODS Graphics, statistical procedures produce graphs as automatically as they produce tables, and graphs are integrated with tables in the ODS output. ODS Graphics is available in procedures in SAS/STAT, Base SAS, SAS/ETS, SAS/QC, and other products (see the section "Procedures That Support ODS Graphics" on page 619). Note that ODS Graphics is automatically provided with Base SAS software.

ODS Graphics might or might not be enabled by default depending on your operating system, whether you are in the SAS windowing environment, your registry, system options, and configuration file settings. For more information about default settings and enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 610.

You can enable ODS Graphics with the following statement:

```
ods graphics on;
```

When ODS Graphics is enabled, procedures that support ODS Graphics create appropriate graphs, either by default or when you specify procedure options for requesting specific graphs. These options are documented in the "Syntax" section of each procedure chapter, and the "Details" section of each chapter provides an "ODS Graphics" subsection that lists the graphs that are available. Once ODS Graphics is enabled, it stays enabled for the duration of your SAS session unless you disable it.

You can disable ODS Graphics with the following statement:

```
ods graphics off;
```

You might consider disabling ODS Graphics if your goal is solely to produce computational results. Often though, you can enable ODS Graphics and then leave it enabled. Throughout this chapter, ODS Graphics is enabled only once per section.

Chapter Reading Guide

This chapter provides a basic introduction to ODS Graphics along with more detailed information. The following list provides a guide to reading this chapter:

- If you want to see a few of the many graphs that are produced by statistical procedures by using ODS Graphics, see the section "Getting Started with ODS Statistical Graphics" on page 592.
- If you are using ODS Graphics for the first time, read the section "A Primer on ODS Statistical Graphics" on page 609, which provides the minimum information that you need to get started.
- If you need to create plots of raw data or your own customized plots of statistical results, see the section "Statistical Graphics Procedures" on page 689, which describes SAS procedures that use ODS Graphics.
- If you need information about specialized topics such as accessing your graphs, making changes to your graphs, and working with ODS styles, see the detailed discussions starting with the section "Syntax" on page 620 and including the section "Examples of ODS Statistical Graphics" on page 699.

If you are unfamiliar with ODS, see Chapter 20, "Using the Output Delivery System." For complete documentation about the Output Delivery System, see the *SAS Output Delivery System: User's Guide*. For an introduction to graph template modification, see Chapter 22, "ODS Graphics Template Modification." For an introduction to ODS Graphics, ODS styles, the graph template language, the style template language, the statistical graphics procedures, and graph template modification, see Kuhfeld (2010). For complete documentation about ODS graph templates, see the *SAS Graph Template Language: User's Guide* and the *SAS Graph Template Language: Reference Guide*. For complete documentation about the ODS Graphics Editor, see the *SAS ODS Graphics Editor: User's Guide*. Also see the *SAS ODS Graphics: Procedures Guide* for information about the statistical graphics procedures.

Assumptions about ODS Defaults in This Chapter

Default settings such as destinations and whether or not ODS Graphics is enabled vary depending on your operating system, registry settings, configuration file settings, system options, and whether you are using the SAS windowing environment or batch mode. For this reason, this chapter makes no assumptions about these defaults. Instead, all destinations are often explicitly closed without assuming which destination (usually LISTING or HTML) is open, destinations are explicitly opened when needed, and ODS Graphics is explicitly enabled and disabled as needed. In some examples, when all destinations are closed, the LISTING destination is opened at the end of the step so that some destination is available for subsequent output. If you know the defaults for your environment, you do not need to use many of the ODS statements that are used in this chapter.

Getting Started with ODS Statistical Graphics

This section provides examples that illustrate the most basic uses of ODS Graphics with a few of the many plots that are produced by statistical procedures.

Default Plots for Simple Linear Regression with PROC REG

This example is based on the section "Getting Started: REG Procedure" on page 6307 of Chapter 76, "The REG Procedure." The Class data set used in this example is available in the Sashelp library. The following statements use PROC REG to fit a simple linear regression model in which Weight is the response variable and Height is the independent variable:

```
ods graphics on;

proc reg data=sashelp.class;
   model Weight = Height;
run; quit;
```

The ODS GRAPHICS ON statement requests ODS Graphics in addition to the usual tabular output. The statement ODS GRAPHICS OFF is not used here, but it can be specified to disable ODS Graphics.

The graphical output consists of a fit diagnostics panel, a residual plot, and a fit plot. These plots are integrated with the tabular output and are shown in Figure 21.1, Figure 21.2, and Figure 21.3, respectively. The results are displayed in the HTMLBLUE style.

Figure 21.1 Fit Diagnostics Panel

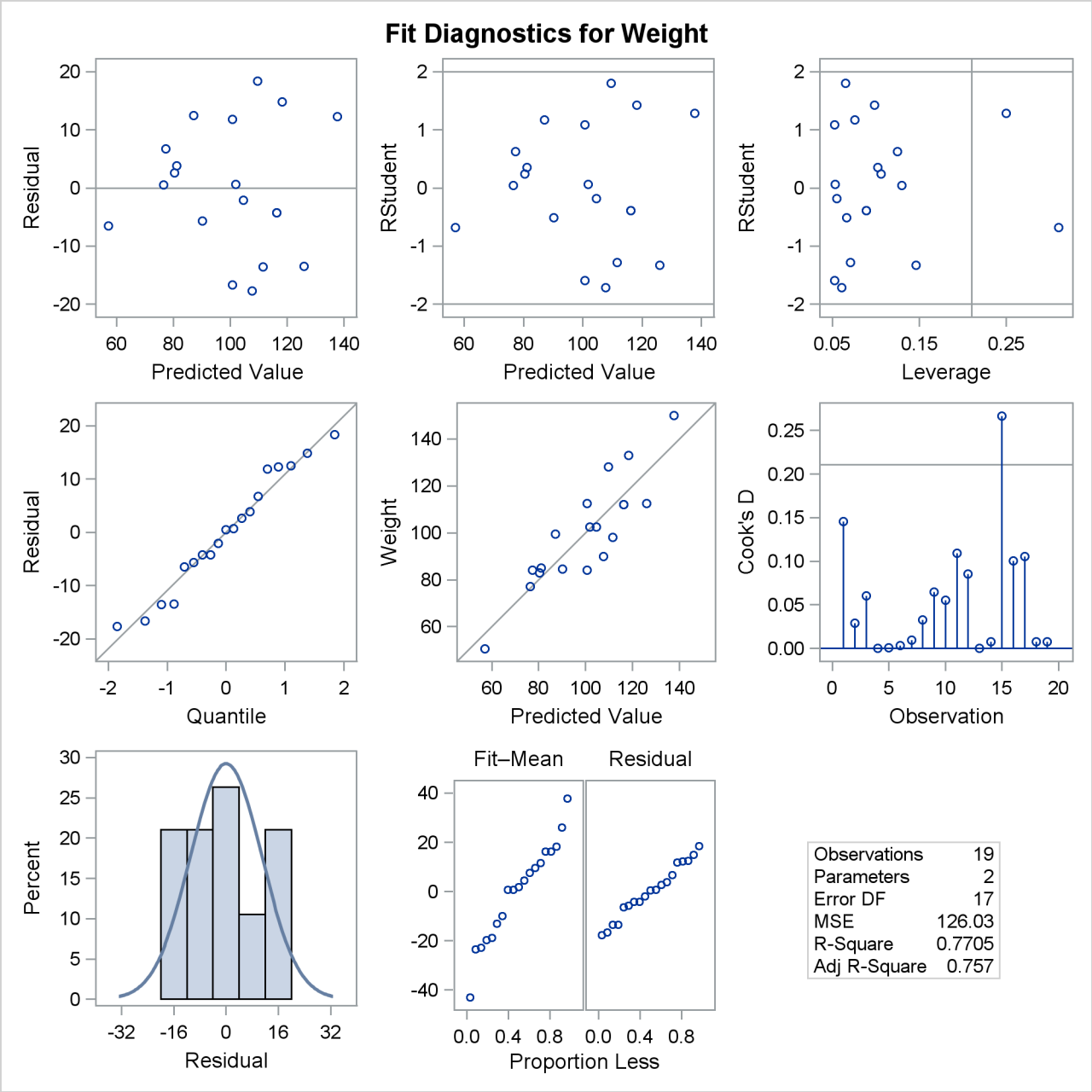


Figure 21.2 Residual Plot

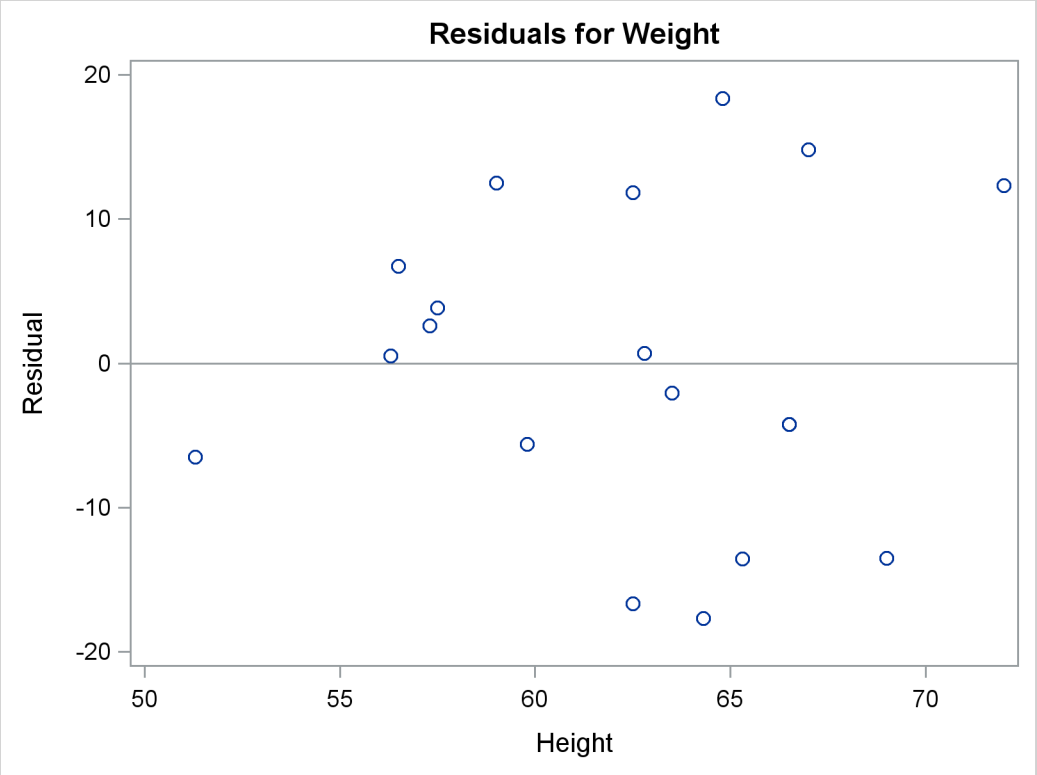
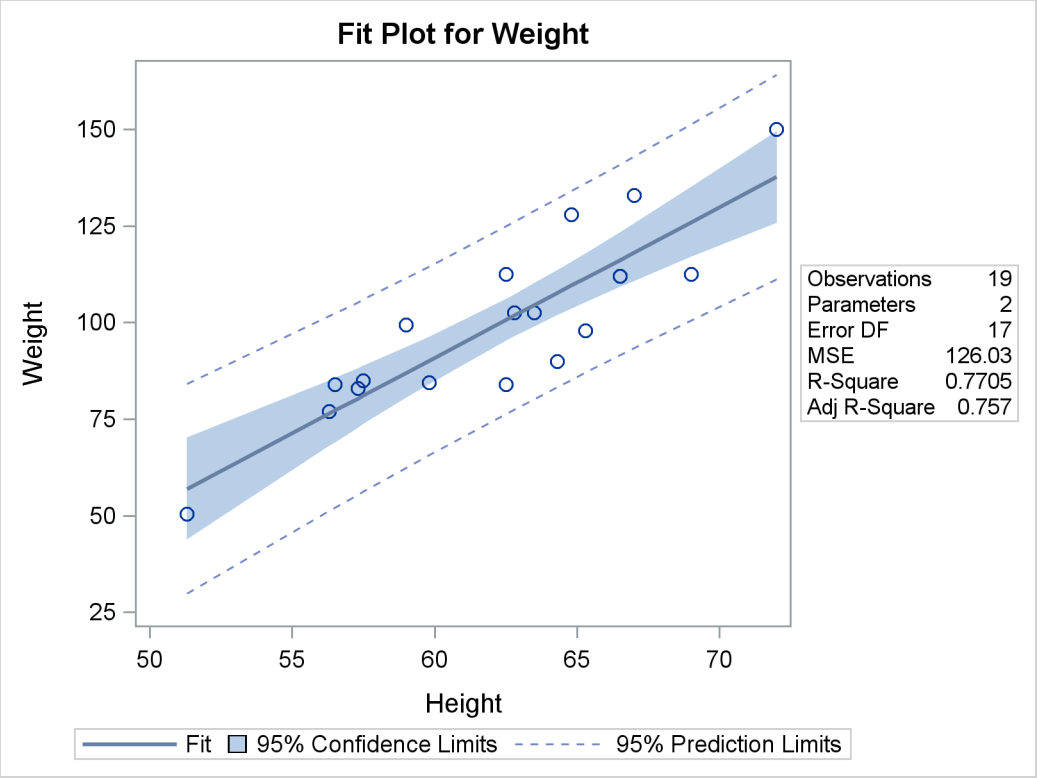


Figure 21.3 Fit Plot



ODS styles control the colors and general appearance of all graphs and tables, and the SAS System provides several styles that are recommended for use with statistical graphics. The default style that you see when you run SAS depends on the ODS destination, system options, and SAS registry settings. For more information about styles, see the section "Graph Styles" on page 611 and the section "Styles" on page 646.

Survival Estimate Plot with PROC LIFETEST

This example is taken from Example 51.2 of Chapter 51, "The LIFETEST Procedure." It shows how to construct a product-limit survival estimate plot. Both the ODS GRAPHICS statement and procedure options are used to request the plot. This examples uses the bone marrow transplant data set, which is available from the Sashelp library. The data set contains disease-free times for three risk categories.

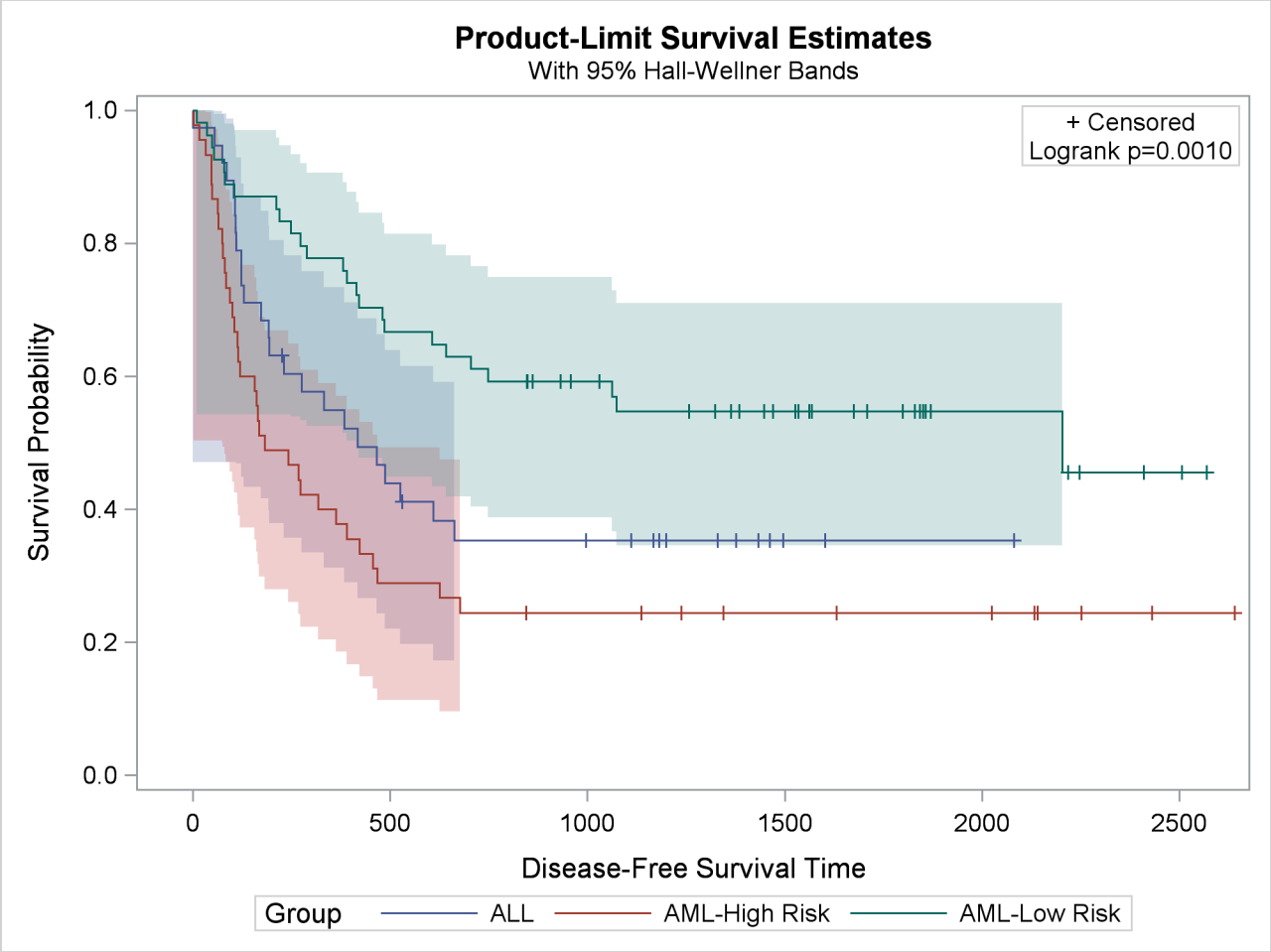
The following statements use PROC LIFETEST to compute the product-limit estimate of the survivor function for each risk category:

```
ods graphics on;

proc lifetest data=sashelp.BMT plots=survival(cb=hw test);
   time T * Status(0);
   strata Group / test=logrank;
run;
```

The ODS GRAPHICS ON statement enables ODS Graphics, and the PLOTS=SURVIVAL option requests a plot of the estimated survival curves. The CB=HW suboption requests Hall-Wellner confidence bands, and the TEST suboption displays the *p*-value for the log-rank test in a plot inset.

Figure 21.4 displays the plot; note that tabular output is not shown. Patients in the AML-Low Risk group are disease-free longer than those in the ALL group, who in turn fare better than those in the AML-High Risk group.

Figure 21.4 Survival Plot



Contour and Surface Plots with PROC KDE

This example is taken from the section "Getting Started: KDE Procedure" on page 3606 in Chapter 47, "The KDE Procedure." Here, in addition to the ODS GRAPHICS statement, procedure options are used to request plots. The following statements simulate 1,000 observations from a bivariate normal density with means (0,0), variances (10,10), and covariance 9:

```
data bivnormal;
   do i = 1 to 1000;
      z1 = rannor(104);
      z2 = rannor(104);
      z3 = rannor(104);
      x  = 3*z1+z2;
      y  = 3*z1+z3;
      output;
   end;
run;
```

The following statements request a bivariate kernel density estimate for the variables x and y:

```
ods graphics on;

proc kde data=bivnormal;
   bivar x y / plots=contour surface;
run;
```

The PLOTS= option requests a contour plot and a surface plot of the estimate (displayed in Figure 21.5 and Figure 21.6, respectively). For more information about the graphs available in PROC KDE, see the section "ODS Graphics" on page 3625 of Chapter 47, "The KDE Procedure."

Figure 21.5 Contour Plot of Estimated Density

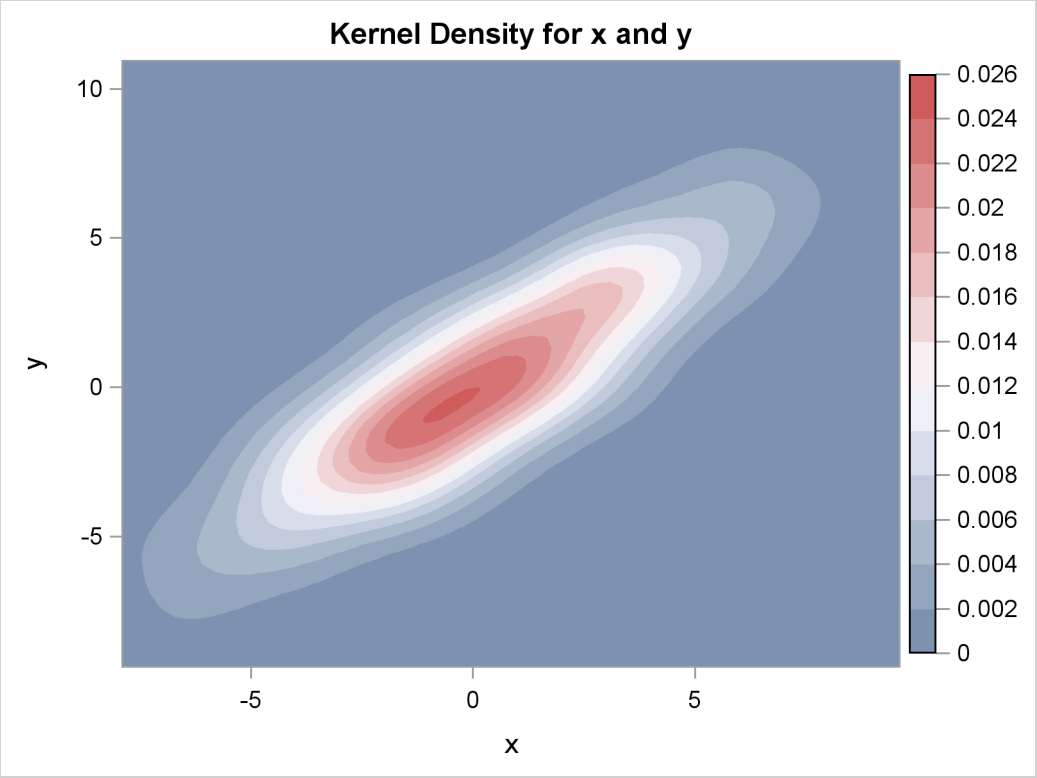
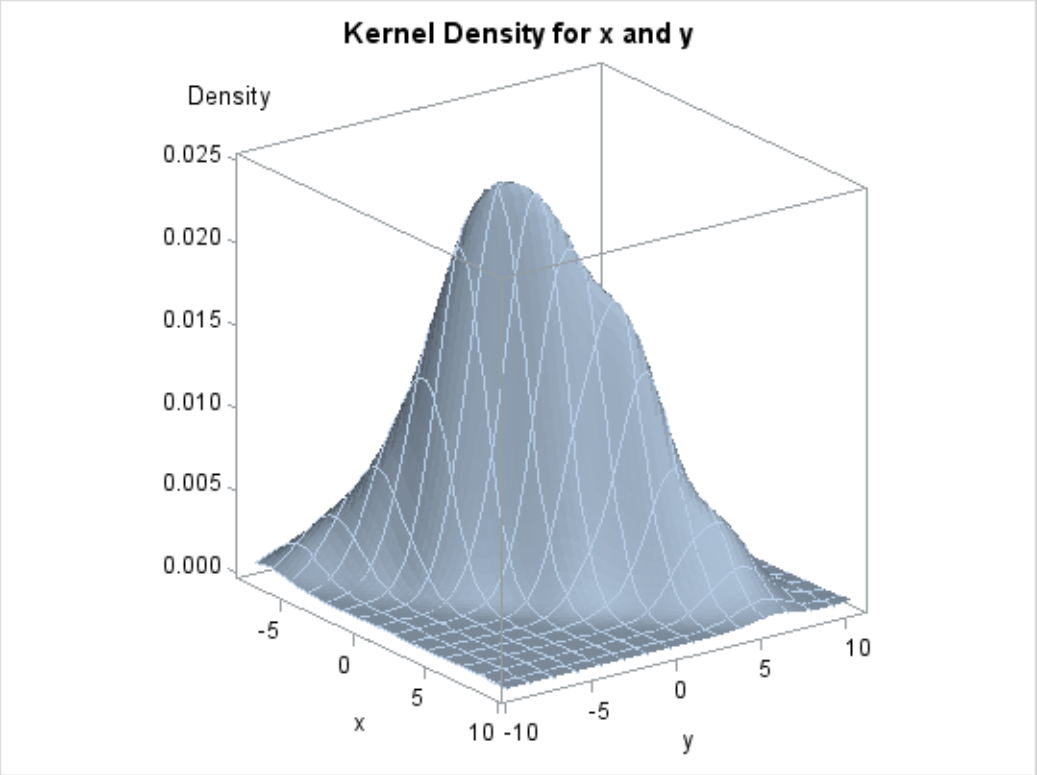


Figure 21.6 Surface Plot of Estimated Density



Contour Plots with PROC KRIGE2D

This example is taken from Example 48.2 of Chapter 48, "The KRIGE2D Procedure." The coal seam thickness data set is available from the Sashelp library. The following statements create a SAS data set that contains a copy of these data along with some artificially added missing data:

```
data thick;
   set sashelp.thick;
   if _n_ in (41, 42, 73) then thick = .;
run;
```

The following statements run PROC KRIGE2D:

```
ods graphics on;

proc krige2d data=thick outest=predictions
             plots=(observ(showmissing)
                    pred(fill=pred line=pred obs=linegrad)
                    pred(fill=se line=se obs=linegrad));
   coordinates xc=East yc=North;
   predict var=Thick r=60;
   model scale=7.2881 range=30.6239 form=gauss;
   grid x=0 to 100 by 2.5 y=0 to 100 by 2.5;
run;
```

The PLOTS=OBSERV(SHOWMISSING) option produces a scatter plot of the data along with the locations of any missing data. The PLOTS=PRED option produces maps of the kriging predictions and standard errors. Two instances of the PLOTS=PRED option are specified with suboptions that customize the plots. The results are shown in Figure 21.7.

Figure 21.7 Spatial Distribution

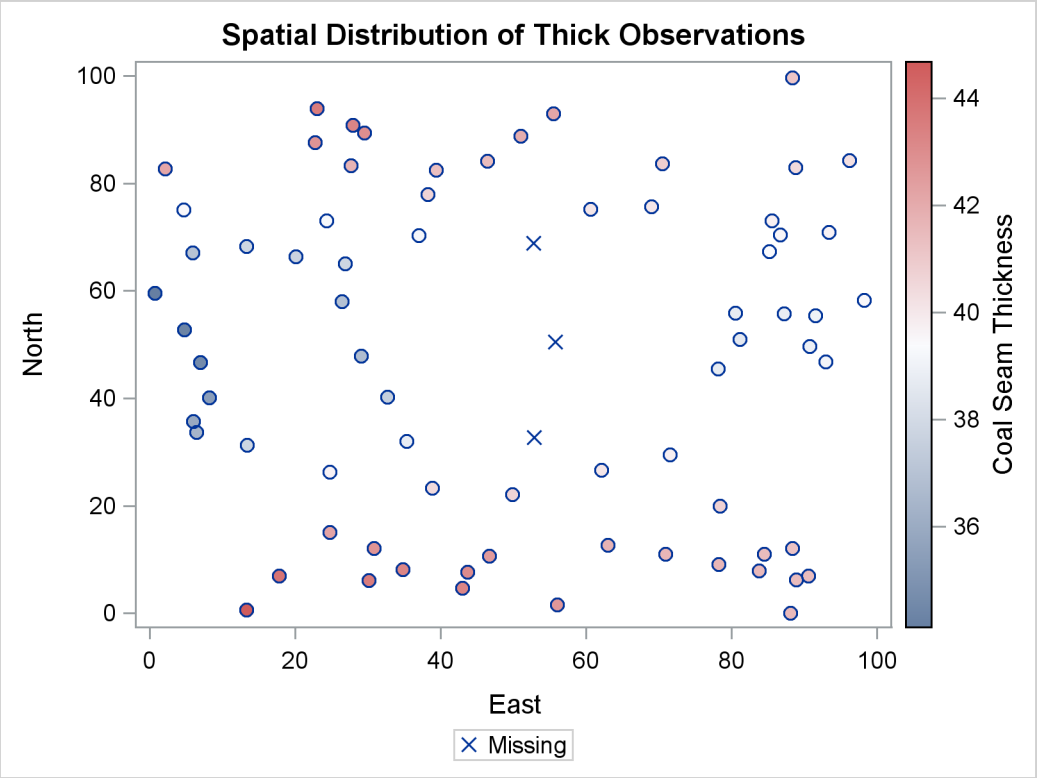
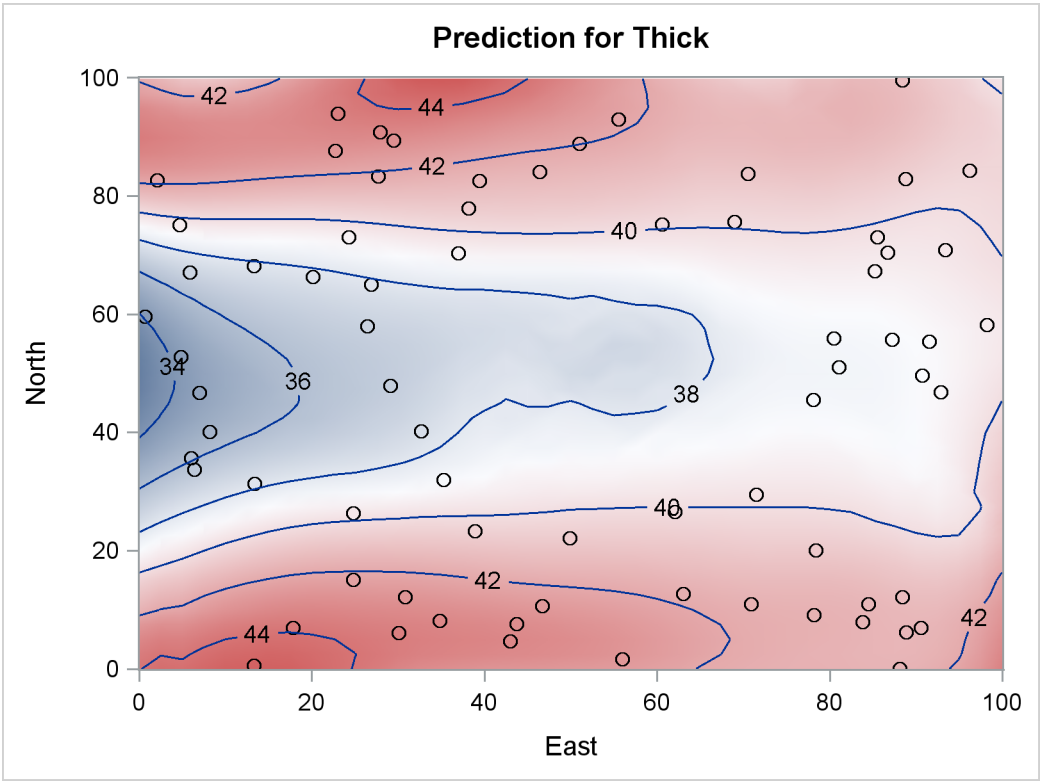
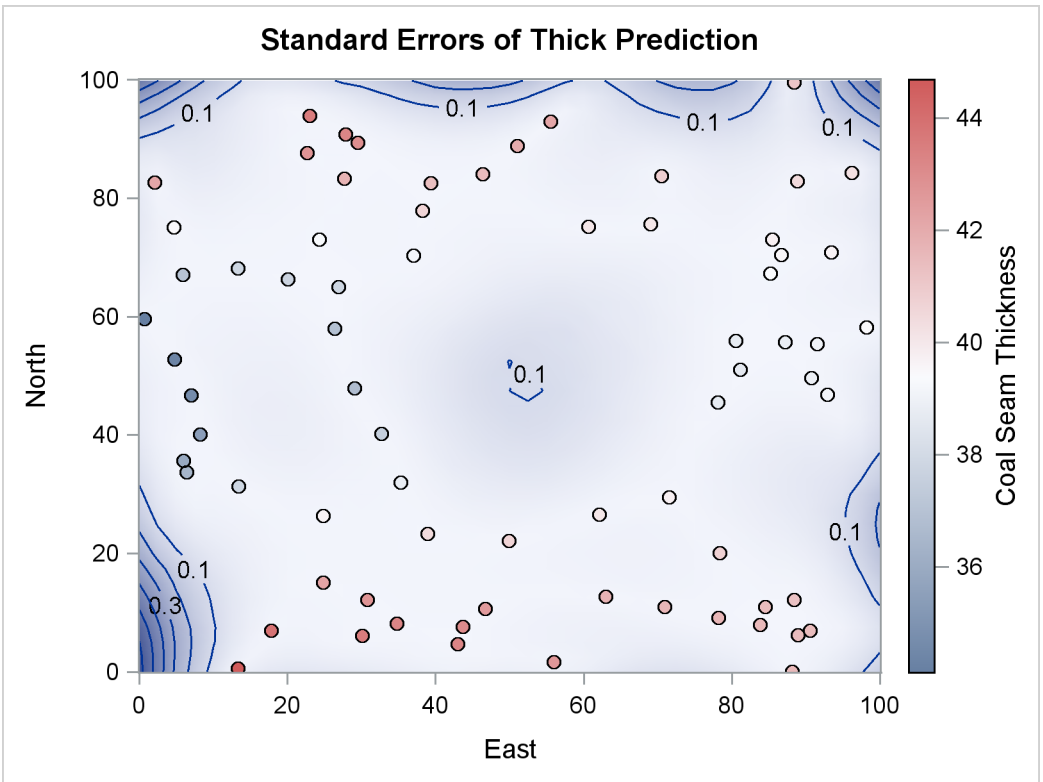


Figure 21.7 *continued*





Partial Least Squares Plots with PROC PLS

This example is taken from the section "Getting Started: PLS Procedure" on page 5641 of Chapter 69, "The PLS Procedure." The following statements create a SAS data set that contains measurements of biological activity in the Baltic Sea:

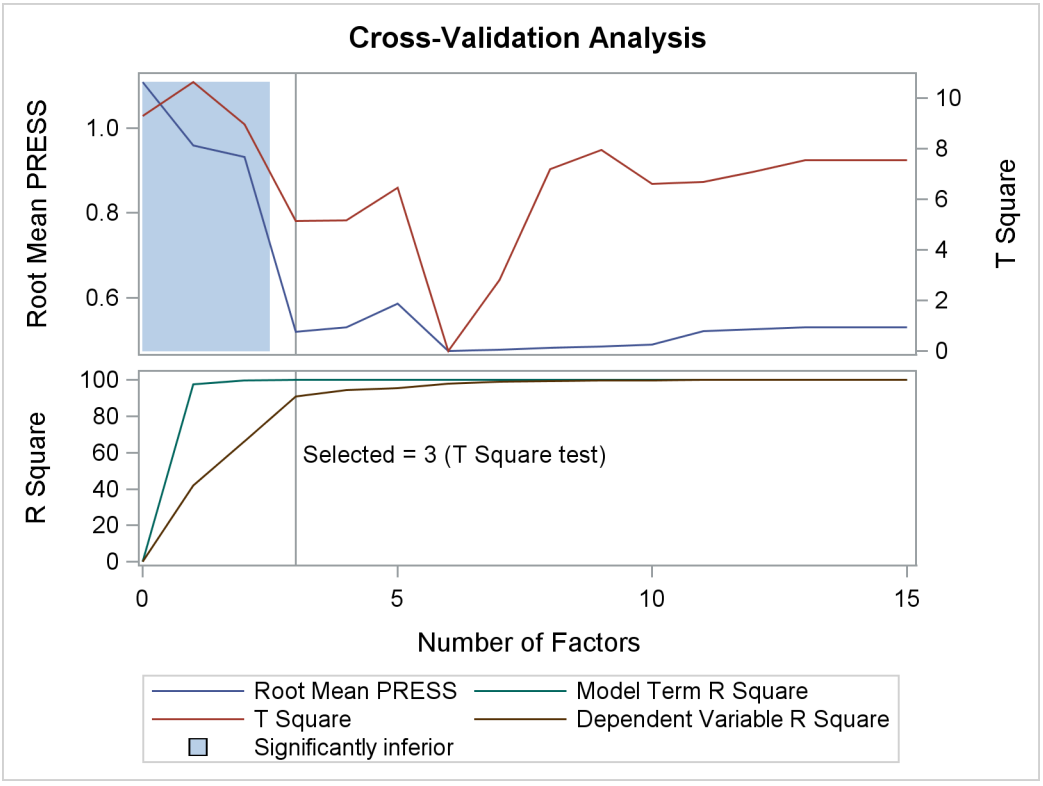
```
data Sample;
   input obsnam $ v1-v27 ls ha dt @@;
   datalines;
EM1   2766 2610 3306 3630 3600 3438 3213 3051 2907 2844 2796
      2787 2760 2754 2670 2520 2310 2100 1917 1755 1602 1467
      1353 1260 1167 1101 1017         3.0110  0.0000   0.00

   ... more lines ...

;
```

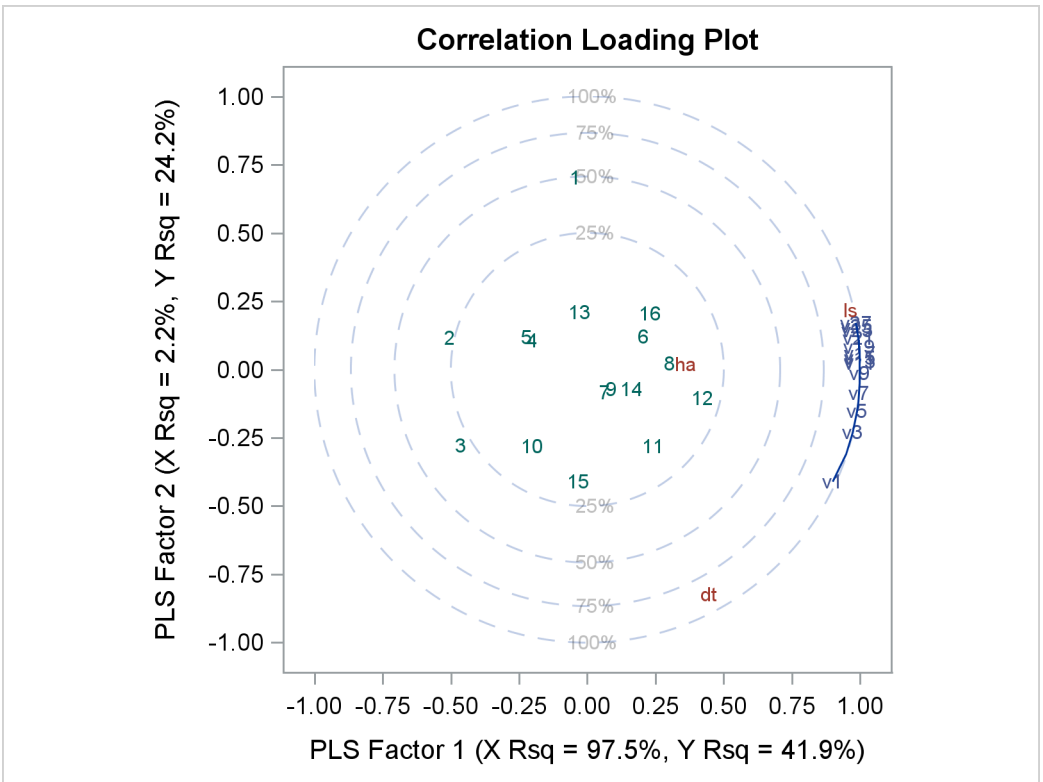
The following statements run PROC PLS:

```
ods graphics on;

proc pls data=sample cv=split cvtest(seed=104);
   model ls ha dt = v1-v27;
run;
```

By default, the procedure produces a plot for the cross validation analysis and a correlation loading plot (see Figure 21.8).

Figure 21.8 Partial Least Squares





Box-Cox Transformation Plot with PROC TRANSREG

This example is taken from Example 93.2 of Chapter 93, "The TRANSREG Procedure." The following statements create a SAS data set that contains failure times for yarn:

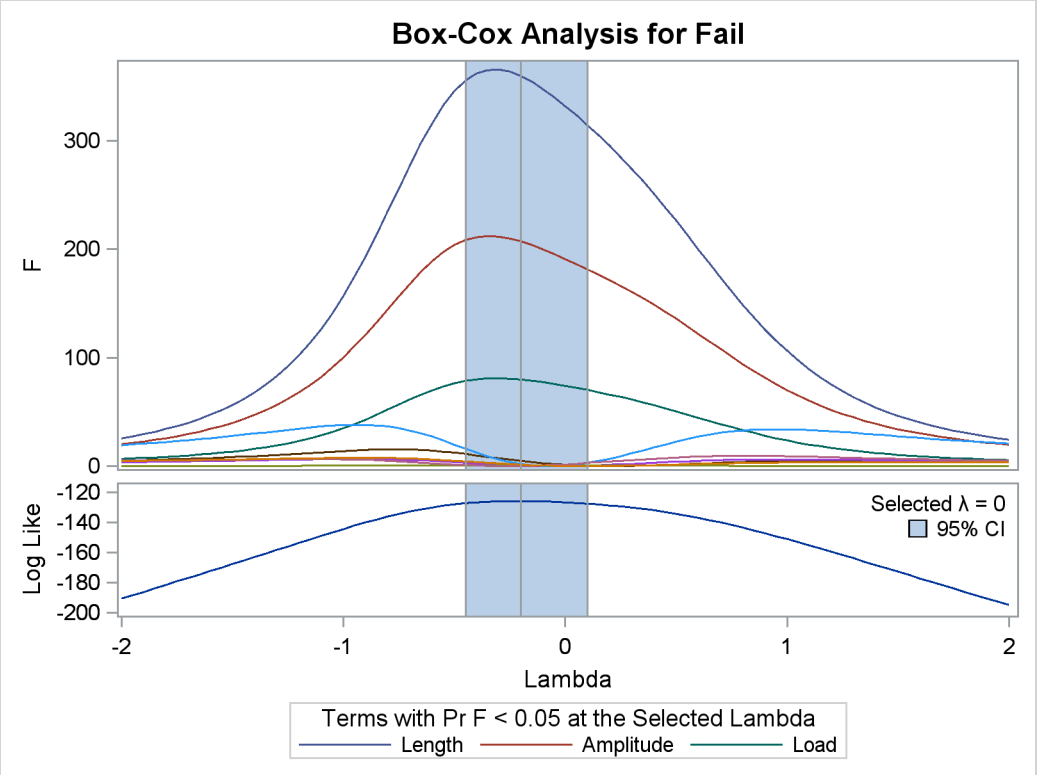
```
proc format;
   value a -1 =   8 0 =   9 1 =  10;
   value l -1 = 250 0 = 300 1 = 350;
   value o -1 =  40 0 =  45 1 =  50;
run;

data yarn;
   input Fail Amplitude Length Load @@;
   format amplitude a. length l. load o.;
   label fail = 'Time in Cycles until Failure';
   datalines;
 674 -1 -1 -1    370 -1 -1  0    292 -1 -1  1    338  0 -1 -1

   ... more lines ...

;
```

The following statements run PROC TRANSREG:

```
ods graphics on;

proc transreg data=yarn;
   model BoxCox(fail / convenient lambda=-2 to 2 by 0.05) =
         qpoint(length amplitude load);
run;
```

The log-likelihood plot in Figure 21.9 suggests a Box-Cox transformation with $\lambda = 0$.

Figure 21.9 Box-Cox "Significant Effects"



LS-Means Diffogram with PROC GLIMMIX

This example is taken from the section "Graphics for LS-Mean Comparisons" on page 2991 of Chapter 40, "The GLIMMIX Procedure." The following statements create a SAS data set that contains measurements from an experiment that investigates how snapdragons grow in various soils:

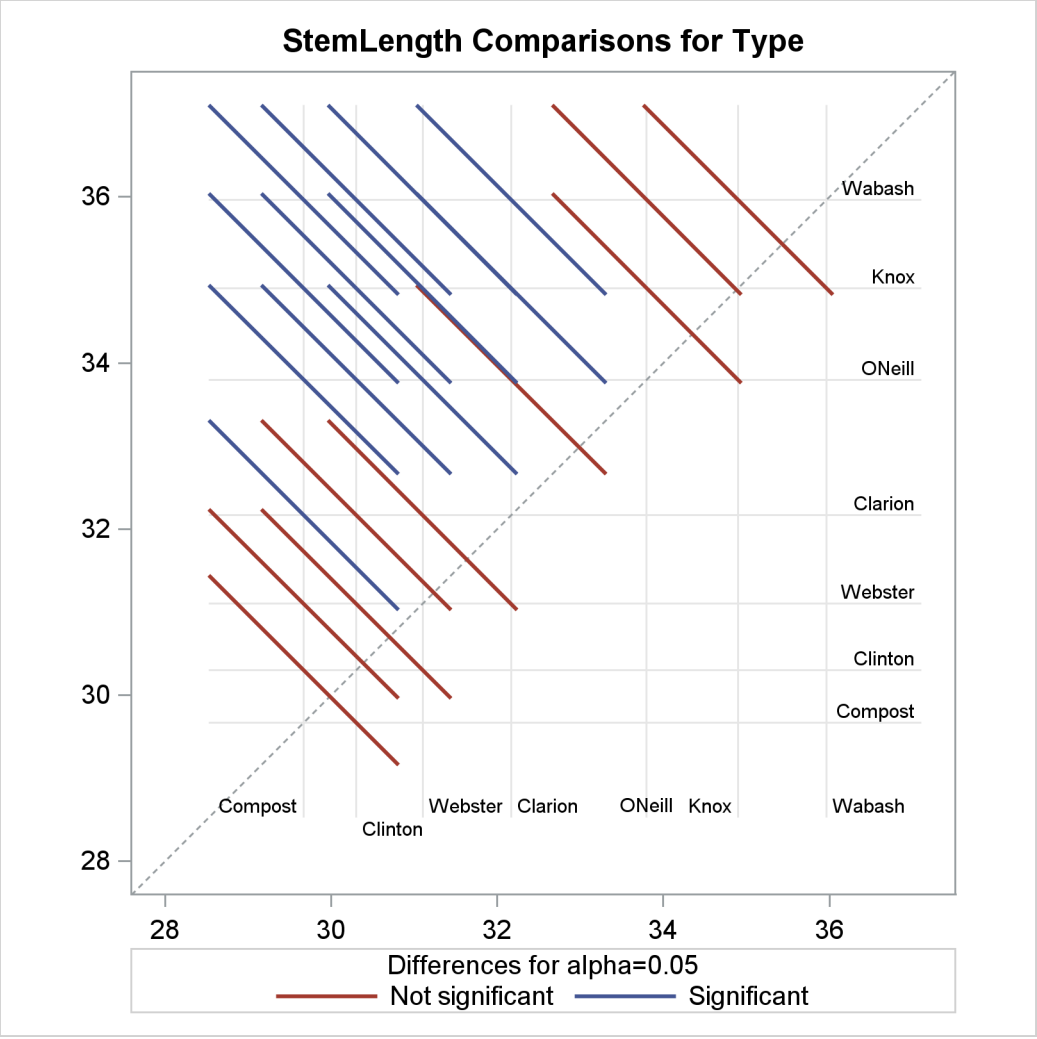
```
data plants;
   input Type $ @;
   do Block = 1 to 3;
      input StemLength @;
      output;
   end;
   datalines;
Clarion   32.7 32.3 31.5

   ... more lines ...

;
```

The following statements run PROC GLIMMIX:

```
ods graphics on;

proc glimmix data=plants order=data plots=diffogram;
   class Block Type;
   model StemLength = Block Type;
   lsmeans Type;
run;
```

The PLOTS=DIFFOGRAM option produces a diffogram, shown in Figure 21.10, that displays all of the pairwise least squares mean differences and indicates which are significant.

Figure 21.10 LS-Means Diffogram



Principal Component Analysis Plots with PROC PRINCOMP

This example is taken from Example 72.3 of Chapter 72, "The PRINCOMP Procedure." The following statements create a SAS data set that contains ratings of job performance of police officers:

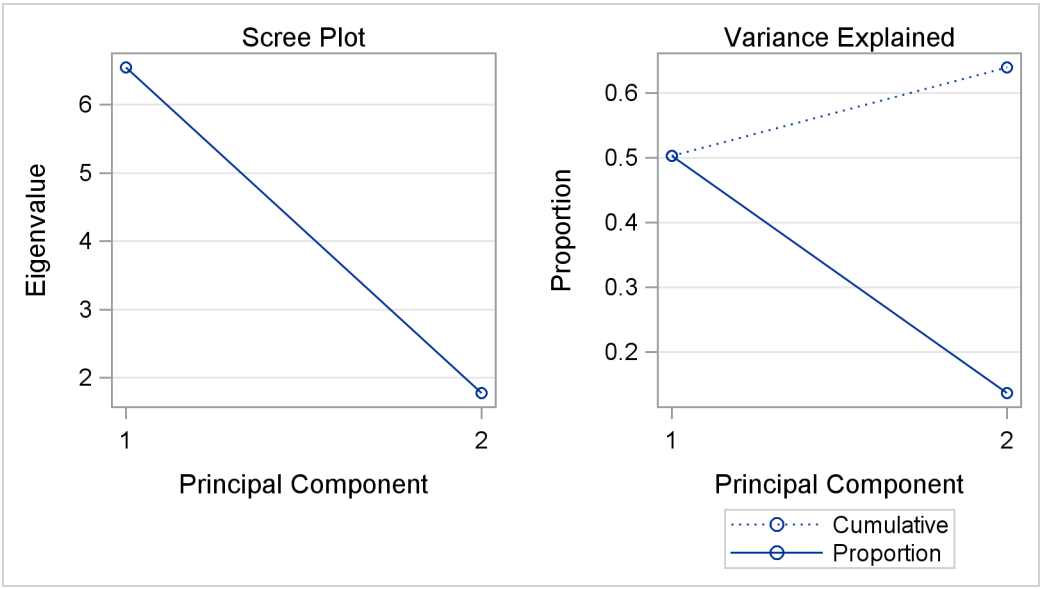
```
options validvarname=any;

data Jobratings;
   input ('Communication Skills'n
          'Problem Solving'n
          'Learning Ability'n
          'Judgment Under Pressure'n
          'Observational Skills'n
          'Willingness to Confront Problems'n
          'Interest in People'n
          'Interpersonal Sensitivity'n
          'Desire for Self-Improvement'n
          'Appearance'n
          'Dependability'n
          'Physical Ability'n
          'Integrity'n
          'Overall Rating'n) (1.);
   datalines;
26838853879867

   ... more lines ...

;
```

The following statements run PROC PRINCOMP:

```
ods graphics on;

proc princomp data=Jobratings(drop='Overall Rating'n) n=2
              plots=(Matrix PatternProfile);
run;
```

The plots are requested by the PLOTS=(MATRIX PATTERNPROFILE) option. The results, shown in Figure 21.11, contain the default scree and variance-explained plots, along with a scatter plot matrix of component scores and a pattern profile plot.

Figure 21.11 Principal Component Analysis



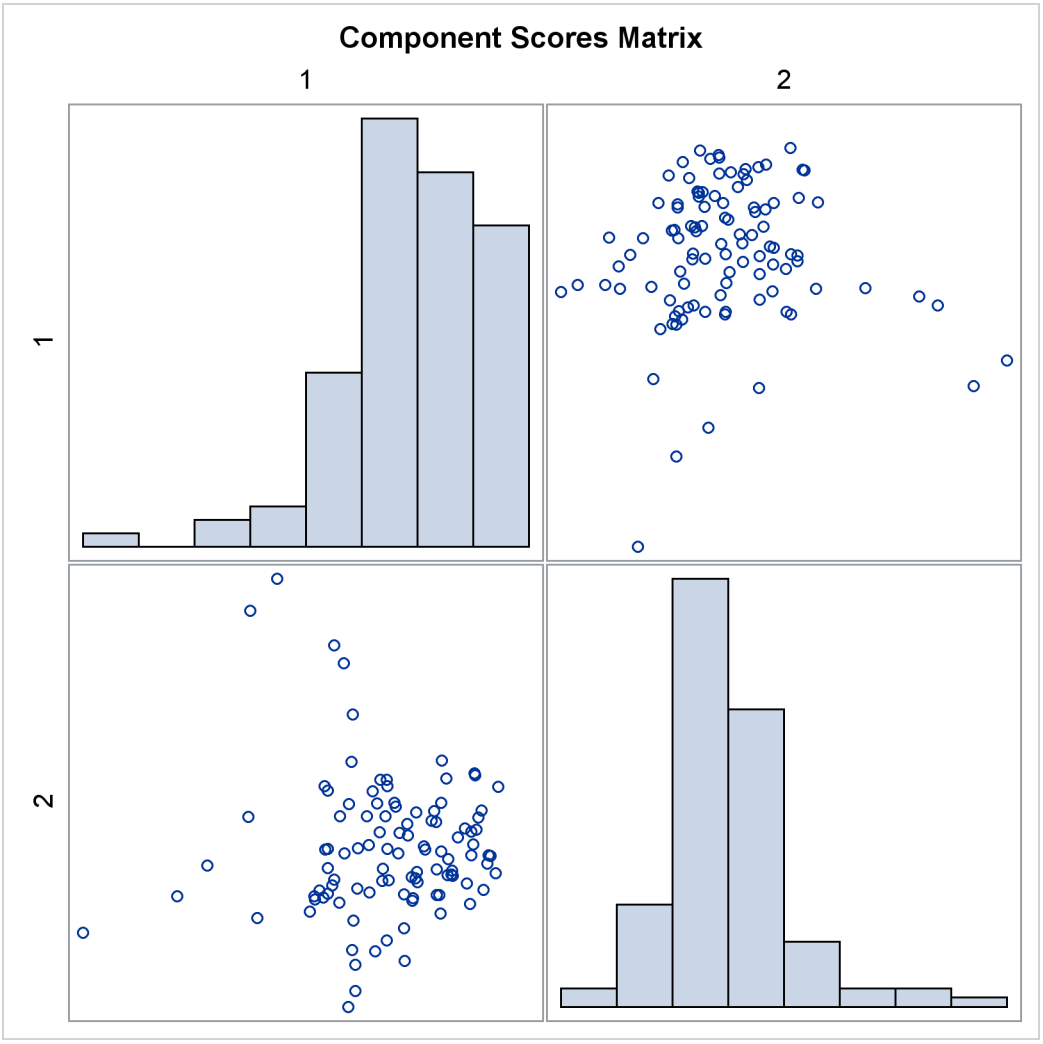
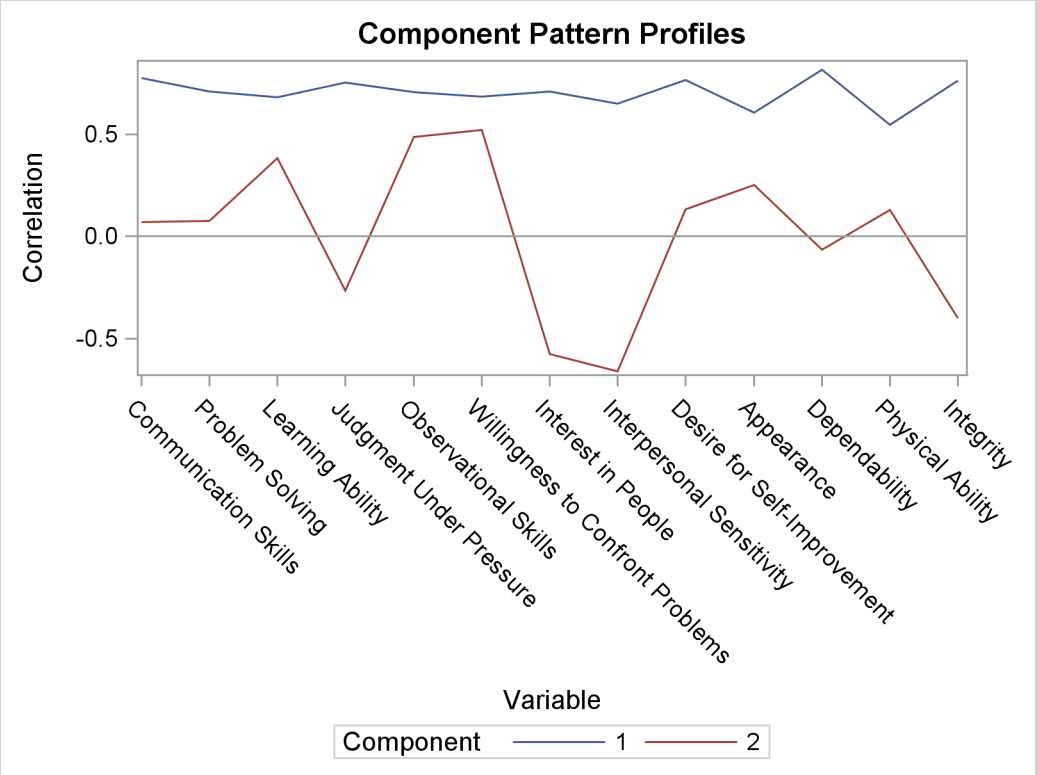


Figure 21.11 *continued*



Grouped Scatter Plot with PROC SGPLOT

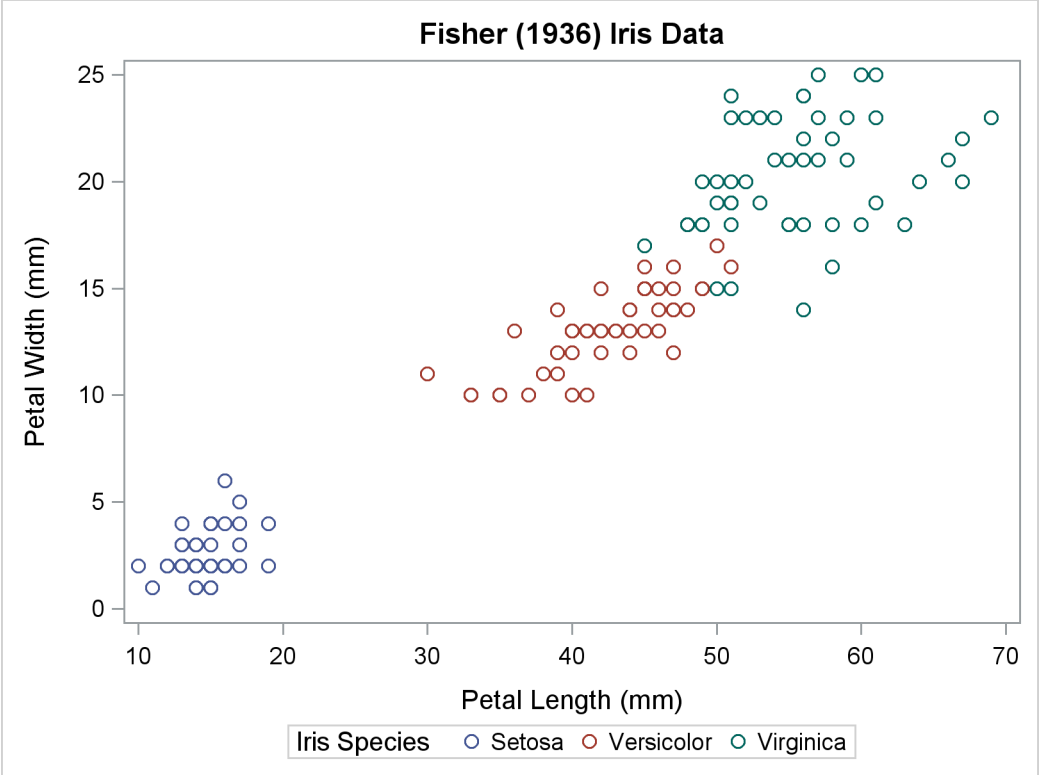
This example is taken from Example 32.1 of Chapter 32, "The DISCRIM Procedure." It uses the Fisher iris data set, which is available from the Sashelp library.

The following statements run PROC SGPLOT to make a scatter plot, grouped by iris species:

```
proc sgplot data=sashelp.iris;
   title 'Fisher (1936) Iris Data';
   scatter x=petallength y=petalwidth / group=species;
run;
```

The results are shown in Figure 21.12.

Figure 21.12 Iris Data



See the section "Statistical Graphics Procedures" on page 689 and the *SAS ODS Graphics: Procedures Guide* for more information about PROC SGPLOT (statistical graphics plot) and other SG procedures. You do not need to enable ODS Graphics in order to use SG procedures (because making plots with ODS Graphics is their sole function).

A Primer on ODS Statistical Graphics

You can enable ODS Graphics by specifying the following statement:

```
ods graphics on;
```

ODS Graphics remains enabled for all procedure steps until you disable it with the following statement:

```
ods graphics off;
```

Once ODS Graphics is enabled, creating graphical output with procedures is as simple as creating tabular output. For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 610. See the section "Syntax" on page 620 for details about the more commonly used ODS GRAPHICS statement options.

You can control your output in the following ways:

- ODS destination statements (such as ODS HTML or ODS RTF) specify where you want your graphs displayed. See Figure 21.20 for an example of HTML output. See the section "ODS Destination Statements" on page 623 for a list of the supported destinations. See the section "Syntax" on page 620 for details about the more commonly used ODS destination statement options.
- ODS SELECT and ODS EXCLUDE statements select and exclude graphs from your output. See the section "Selecting and Excluding Graphs" on page 631 for an example of how to select graphs.
- ODS OUTPUT statements create SAS data sets from the data object used to make the plot. See the section "Specifying an ODS Destination for Graphics" on page 626 for an example.
- Procedure options specify which graphs to create. For each procedure, these options are described in the "Syntax" section of the procedure chapter. Typically, you use the PLOTS= option to control all graphs. The available graphs are listed in the "ODS Graphics" section, which is found in the "Details" section of each procedure chapter. Many graphs are produced by default.
- ODS styles control the general appearance and consistency of all graphs and tables. See the sections "Graph Styles" on page 611 and "Styles" on page 646 for more information about styles.
- ODS templates modify the layout and details of each graph. See the section "Graph Templates" on page 711 in Chapter 22, "ODS Graphics Template Modification," for more information about templates.

  NOTE: A default template is provided by SAS for each graph, so you do not need to know anything about templates to create statistical graphics.

You can also access individual graphs, control the resolution and size of graphs, and modify your graphs (as explained in the sections beginning with "Selecting and Viewing Graphs" on page 626). Alternatively, you can use special statistical graphics procedures to create custom graphs directly (see the section "Statistical Graphics Procedures" on page 689).

Enabling and Disabling ODS Graphics

You can enable ODS Graphics by specifying the following statement:

```
ods graphics on;
```

ODS Graphics remains enabled for all procedure steps until you disable it with the following statement:

```
ods graphics off;
```

ODS Graphics might or might not be enabled by default. This depends on a number of factors. ODS Graphics is typically enabled by default in the SAS windowing environment; ODS Graphics is typically disabled by default when you invoke SAS in other ways. However, these defaults can be changed in a number of ways. You can enable or disable ODS Graphics by default in an *autoexec.sas* file, a configuration file such as *SASV9.CFG*, or in the SAS registry. You can change the default in the SAS windowing environment by

selecting **Tools ► Options ► Preferences** from the menu at the top of the main SAS window. Then on the **Results** tab, select the **Use ODS Graphics** check box to enable ODS Graphics by default or clear the check box to disable ODS Graphics by default. You can also change the default output destination (HTML or LISTING) on the **Results** tab. See the section "HTML Output in the SAS Windowing Environment" on page 524 for more information about default ODS Graphics settings and default destinations.

When ODS Graphics is enabled, procedures that support ODS Graphics create graphs, either by default or when you specify procedure options for requesting specific graphs. Often, you can leave ODS Graphics enabled for the duration of your SAS session. However, you might consider disabling ODS Graphics if your goal is solely to produce computational results, particularly for large data sets or with many BY groups.

Graph Styles

ODS styles control the overall appearance of graphs and tables. They specify colors, fonts, line styles, symbol markers, and other attributes of graph elements. The following styles (among the many ODS styles) are recommended for statistical work:

- The HTMLBLUE style is a color style that is recommended for use in Web pages or color print media. See Figure 21.20 for an example. The HTMLBLUE style inherits most of its attributes from the STATISTICAL style, which inherits from the DEFAULT style. The HTMLBLUE style has a brighter appearance than its parents with color coordination between the tables and graphs. The dominant color is blue.

  The HTMLBLUE style is one of the default styles for the HTML destination (depending on SAS option and registry settings). It is also the default style in SAS/STAT documentation. It is an all-color style; groups of observations are distinguished by color instead of by line style or symbol changes.[1] Most other styles simultaneously vary colors, line styles, and marker symbols to show group membership. Output created with the HTMLBLUE style does not print well on black-and-white devices. If you need an alternative to the HTMLBLUE style that varies colors, lines, and markers, use the HTMLBLUECML style or some other style.

- The HTMLBLUECML style is a color style that is recommended for use in Web pages or color print media. It inherits most of its attributes from the HTMLBLUE style. See Figure 21.21 for an example. Groups of observations are distinguished by simultaneous color, line style, and symbol changes. If you need an alternative to the HTMLBLUECML style that is all-color, use the HTMLBLUE style instead.

- The DEFAULT style is a color style. See Figure 21.19 for an example. Most other styles inherit some of their elements from this style. The DEFAULT style is one of the default styles for the HTML destination (depending on SAS registry and option settings). Groups of observations are distinguished by simultaneous color, line style, and symbol changes. Output created with the DEFAULT style might not print well on black-and-white devices.

- The STATISTICAL style is a color style. See Figure 21.22 for an example. Output created with the STATISTICAL style might not print well on black-and-white devices. Groups of observations

[1] More precisely, the HTMLBLUE style is an all-color style for the first 12 groups of observations, which are more than are shown in most analyses. Markers and lines change for groups 13–24 and then again for groups 25–36. Figure 21.36 shows how colors, markers, and line styles change in the HTMLBLUE style, and Figure 21.35 shows how these change in most other styles.

are distinguished by simultaneous color, line style, and symbol changes. The STATISTICAL style inherits elements from the DEFAULT style.

- The ANALYSIS style is a color style with a somewhat different appearance from the STATISTICAL style. See Figure 21.23 for an example. Groups of observations are distinguished by simultaneous color, line style, and symbol changes. The ANALYSIS style inherits elements from the DEFAULT style. Output created with the ANALYSIS style might not print well on black-and-white devices.
- The JOURNAL family of styles (JOURNAL, JOURNAL2, and JOURNAL3) consists of black-and-white or gray-scale styles that are recommended for graphs that appear in journals and in other black-and-white publications. See Figure 21.24 for an example of the JOURNAL style, see Figure 21.9 for an example of the JOURNAL2 style, and see Example 21.3 for a comparison of the three styles.
- The RTF style is used to produce graphs to insert into a Microsoft Word document or a Microsoft PowerPoint slide. See Figure 21.26 for an example of the RTF style, which is the default style for the RTF destination. Groups of observations are distinguished by simultaneous color, line style, and symbol changes. The RTF style inherits elements from the DEFAULT style. Output created with the RTF style might not print well on black-and-white devices.
- The LISTING style is similar to the DEFAULT style, but with a lighter background. See Figure 21.25 for an example. It is the default style for the LISTING destination. Groups of observations are distinguished by simultaneous color, line style, and symbol changes. The LISTING style inherits elements from the DEFAULT style. Output created with the LISTING style might not print well on black-and-white devices.

You specify a style with the STYLE= option in the ODS destination statement. For example, the following statement requests RTF output produced with the JOURNAL style:

```
ods rtf style=Journal;
```

The following statement sets the style for the LISTING destination:

```
ods listing style=HTMLBlue;
```

The style specified with the STYLE= option in the ODS LISTING statement applies only to graphs. SAS monospace format is used for tables.

Most color styles (except the HTMLBLUE style) are compromise styles in the sense that some graph elements are intentionally over-distinguished to facilitate black-and-white printing. For example, fit lines that correspond to different classification levels are distinguished by both colors and line patterns. You can use the HTMLBLUE style when you want groups to be distinguished only by color. You can easily modify any style to be an all-color style like HTMLBLUE. For example:

```
proc template;
   define style styles.Default2;
      parent = default;
      style Graph from Graph / attrpriority = "Color";
      end;
   run;
```

The **AttrPriority = "Color"** option makes a style an all-color style.

You can instead use the %MODSTYLE SAS autocall macro (see the sections "Creating an All-Color Style" on page 676 and "Style Template Modification Macro" on page 674) to modify some other style so that it relies only on color for distinguishability. More generally, you can modify the colors, fonts, and other attributes of graph elements in a style by editing the style template. More information is provided in the section "Styles" on page 646, and detailed information is in the *SAS Output Delivery System: User's Guide*.

ODS Destinations

ODS can send your graphs and tables to a number of different destinations including RTF (rich text format), HTML (hypertext markup language), LISTING (the SAS LISTING destination), DOCUMENT (the ODS document), and PDF (portable document format). You use an ODS statement to open a destination, as in the following examples:

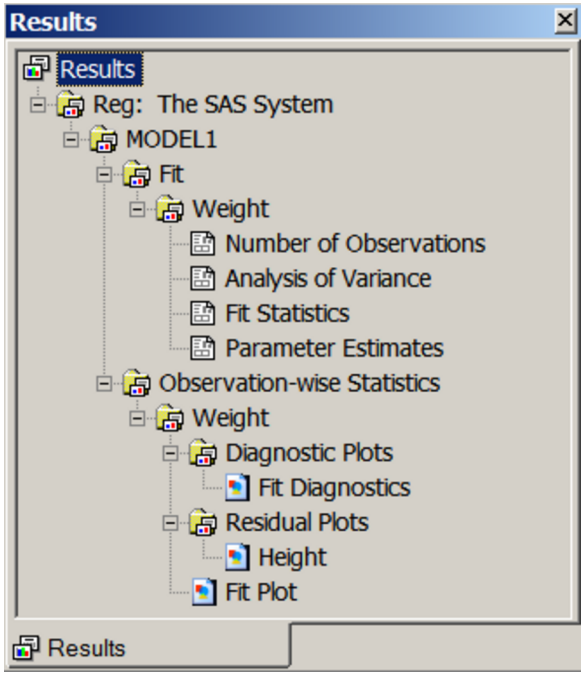
```
ods html body='b.htm';
ods rtf;
ods listing;
ods document name=MyDoc(write);
ods pdf file="contour.pdf";
```

You can close destinations individually or all at once, as in the following examples:

```
ods html close;
ods rtf close;
ods listing close;
ods document close;
ods pdf close;
ods _all_ close;
```

For most ODS destinations (for example, HTML, RTF, and PDF), graphs and tables are integrated in the output, and you view your output with an appropriate viewer, such as a web browser for HTML. However, the LISTING destination is different. If you are using the LISTING destination in the SAS windowing environment, you view your graphs individually by clicking the graph icons in the Results window, shown in Figure 21.13. This action invokes a host-dependent graph viewer (for example, Microsoft Photo Editor on Windows). The graphs produced with ODS Graphics are *not* displayed with traditional graphs in the Graph window.

Figure 21.13 SAS Results Window



If you are using the SAS windowing environment and you prefer to view integrated output, you should use a destination such as HTML or RTF. In many cases, HTML is the default destination in the SAS windowing environment (see the section "HTML Output in the SAS Windowing Environment" on page 524). You can change destinations in the SAS windowing environment by selecting **Tools ► Options ► Preferences** from the menu at the top of the main SAS window and then selecting the **Results** tab.

Instead, you can prevent the Output window from appearing by using ODS statements to close the LISTING destination, as follows:

```
ods listing close;
ods html;
```

A graph is created for every open destination. When you open a new destination, you should close all destinations that you do not need. Closing destinations makes your jobs run faster and with fewer resources, because fewer tables and graphs are produced.

Accessing Individual Graphs

If you are writing a paper or creating a presentation, you need to access your graphs individually. There are various ways to do this, depending on the ODS destination. Three particularly useful methods are as follows:

- If you are viewing RTF output, you can simply copy and paste your graphs from the viewer into a Microsoft Word document or a Microsoft PowerPoint slide.

- If you are viewing HTML output, you can copy and paste your graphs from the viewer, or you can right-click the graph and save it to a file. Copying and pasting from RTF is preferable because the default resolution is higher than with HTML. See the section "Specifying the Size and Resolution of Graphs" on page 615 for details.
- You can save your graphs in image files and then include them into a paper or presentation. For example, you can save your graphs as PNG files and include them into a paper that you are writing with LaTeX or into an HTML document.

You can specify the graphics image format and the filename in the ODS GRAPHICS statement. For example, the following statements, when submitted before a procedure step that produces multiple graphs, save the graphs in PostScript files named *myname.ps*, *myname1.ps*, and so on:

```
ods _all_ close;
ods latex;
ods graphics on / outputfmt=ps imagename='myname';
```

See the section "Image File Types" on page 632 for details about the file types available with various destinations, how they are named, and how they are saved.

If you are using the LISTING destination and the SAS windowing environment, you can also copy from the viewer into a Microsoft Word document or a Microsoft PowerPoint slide.

Specifying the Size and Resolution of Graphs

Two factors to consider when you are creating graphs for a paper or presentation are the size of the graph and its resolution. You can specify the size of a graph in the ODS GRAPHICS statement. The following examples show typical ways to change the size of your graphs:

```
ods graphics on / width=6in;
ods graphics on / height=4in;
ods graphics on / width=4.5in height=3.5in;
```

You can change the resolution with the IMAGE_DPI= option in any ODS destination statement, as in the following example:

```
ods html image_dpi=300;
```

The default resolution of graphs created with the HTML and LISTING destinations is 96 DPI (dots per inch), whereas the default with the RTF destination is 200 DPI. An increase in resolution often improves the quality of the graphs, but it also increases the size of the image file. See the section "Graph Size and Resolution" on page 639 for more information about graph size and resolution.

Modifying Your Graphs

Although ODS Graphics is designed to automate the creation of high-quality statistical graphics, on occasion you might need to modify your graphs. There are two ways you can make modifications, depending on whether the changes you want to make are data-dependent and immediate (for a specific graph you are preparing for a paper or presentation), or whether they are persistent (applied to a graph each time you run the procedure). You can make immediate, ad hoc changes by using the ODS Graphics Editor, which provides a point-and-click interface. You can make persistent changes by modifying the ODS graph template for a particular plot. For an introduction to graph template modification, see Chapter 22, "ODS Graphics Template Modification." A graph template is a program, written in the Graph Template Language (GTL), that specifies the layout and details of a graph.

NOTE: The SAS system provides a template for each graph it creates, so you do not need to know anything about templates to create statistical graphics.

You can use the ODS Graphics Editor to customize titles and labels, annotate data points, add text, and change the properties of graph elements. After you have modified your graph, you can save it as a PNG image file or as an SGE file, which preserves the editing context. You can open SGE files with the ODS Graphics Editor and resume editing.

You can invoke the ODS Graphics Editor in the SAS windowing environment, provided that you have enabled ODS Graphics to create editable graphs. The steps for doing this are described in the section "ODS Graphics Editor" on page 640. Also see *SAS ODS Graphics Editor: User's Guide*.

Figure 21.14 shows the ODS Graphics Editor window for a fit plot created by PROC REG. Figure 21.15 shows modifications made with tools in the ODS Graphics Editor. The title has been changed, and the legend has been repositioned.

Figure 21.14 ODS Graphics Editor Invoked with a Fit Plot

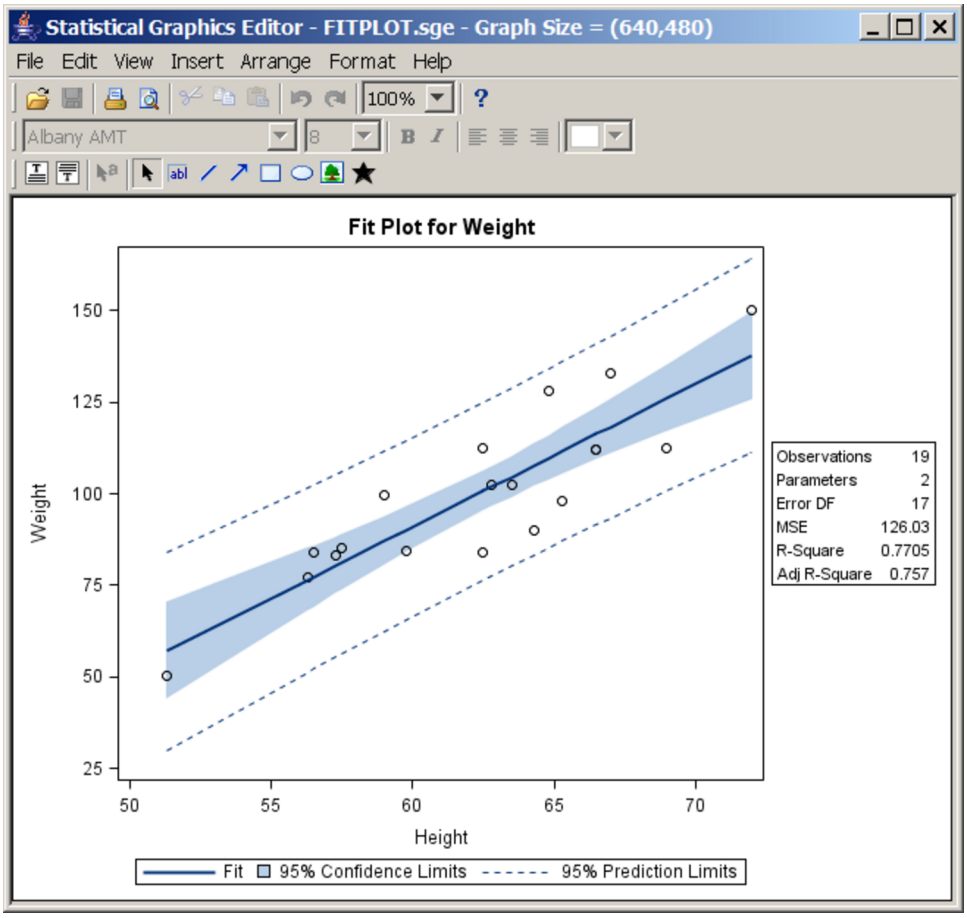
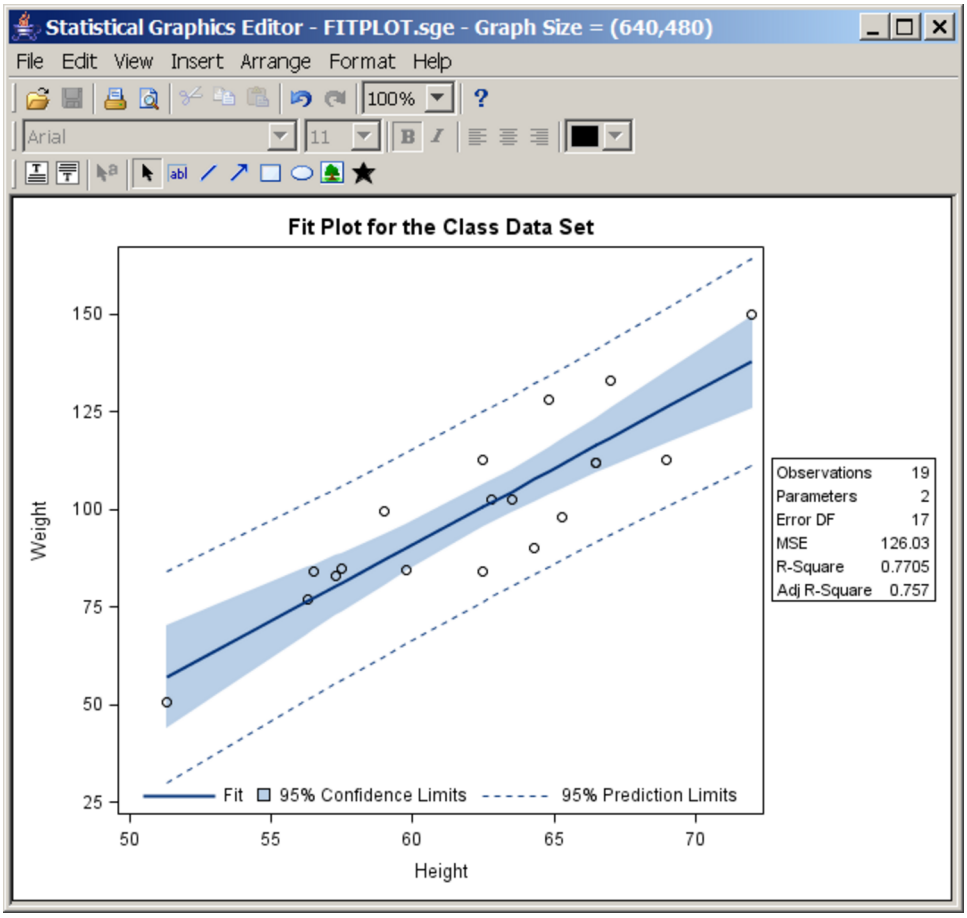


Figure 21.15 Point-and-Click Modifications Made with the ODS Graphics Editor



Procedures That Support ODS Graphics

SAS procedures that support ODS Graphics include the following:

SAS/STAT		SAS/QC	SAS/ETS
ANOVA	NPAR1WAY	ANOM	ARIMA
BOXPLOT	ORTHOREG	CAPABILITY	AUTOREG
CALIS	PHREG	CUSUM	COPULA
CLUSTER	PLM	MACONTROL	ENTROPY
CORRESP	PLS	MVPCHART	ESM
FACTOR	POWER	MVPMODEL	EXPAND
FMM	PRINCOMP	PARETO	MODEL
FREQ	PRINQUAL	RELIABILITY	PANEL
GAM	PROBIT	SHEWHART	SEVERITY
GENMOD	QUANTREG		SIMILARITY
GLIMMIX	REG	**Base SAS**	SYSLIN
GLM	ROBUSTREG	CORR	TIMEID
GLMPOWER	RSREG	FREQ	TIMESERIES
GLMSELECT	SEQDESIGN	UNIVARIATE	UCM
KDE	SEQTEST		VARMAX
KRIGE2D	SIM2D		X12
LIFEREG	SURVEYFREQ	**Other**	
LIFETEST	SURVEYLOGISTIC	HPF	
LOESS	SURVEYPHREG	HPFENGINE	
LOGISTIC	SURVEYREG		
MCMC	TPSPLINE	SAS Risk	
MDS	TRANSREG	Dimensions	
MI	TTEST		
MIXED	VARCLUS		
MULTTEST	VARIOGRAM		
NLIN			

For details about the specific graphs available with a particular procedure, see the PLOTS= option syntax and the "ODS Graphics" section in the corresponding procedure chapter. For the SAS/STAT procedures, the procedure names in the preceding table are links to the "ODS Graphics" section.

Procedures That Support ODS Graphics and Traditional Graphics

A number of procedures that support ODS Graphics produced traditional graphics in previous releases of SAS. These include the UNIVARIATE procedure in Base SAS software; the LIFEREG, LIFETEST, and REG procedures in SAS/STAT software; and the ANOM, CAPABILITY, CUSUM, MACONTROL, PARETO, RELIABILITY, and SHEWHART procedures in SAS/QC software. All of these procedures continue to produce traditional graphics, but in some cases, they do so only when ODS Graphics is not enabled. For more information about the interaction between traditional graphics and ODS graphics in other procedures, see the documentation for that procedure.

Traditional graphs are saved in SAS graphics catalogs and are controlled by the GOPTIONS statement. In contrast, ODS Graphics produces graphs in standard image file formats (not graphics catalogs), and their appearance and layout are controlled by ODS styles and templates.

Syntax

The following sections document some of the most commonly used options in the ODS GRAPHICS statement (section "ODS GRAPHICS Statement" on page 620) and other statements used with ODS Graphics (section "ODS Destination Statements" on page 623). You can find the complete syntax in the *SAS Output Delivery System: User's Guide*. In addition, information about the PLOTS= option is provided in the section "PLOTS= Option" on page 624. Statistical procedures that produce ODS Graphics all have a PLOTS= option that is used to select graphs and control some aspects of the graphs.

ODS GRAPHICS Statement

ODS GRAPHICS < **OFF** | **ON** > < / *options* > **;**

The ODS GRAPHICS statement enables ODS to create graphs. You can enable ODS Graphics by using either of the following equivalent statements:

```
ods graphics on;
ods graphics;
```

You specify one of these statements prior to your procedure invocation, as illustrated in the examples beginning with "Default Plots for Simple Linear Regression with PROC REG" on page 592. Any procedure that supports ODS Graphics then produces graphs, either by default or when you specify procedure options for requesting particular graphs.

To disable ODS Graphics, specify the following statement:

```
ods graphics off;
```

ODS Graphics might or might not be enabled by default depending on your operating system, whether you are in the SAS windowing environment, your registry, system options, and configuration file settings. For more information about default settings and enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 610.

The following is a subset of the options, syntax, and capabilities available in the ODS GRAPHICS statement. See the *SAS Output Delivery System: User's Guide* for more information.

ANTIALIAS=ON | OFF

controls the use of antialiasing to smooth the components of a graph. Without antialiasing, pixels are simply set or not set. With antialiasing, pixels at the edge of a line or other object are set to an intermediate color, which makes smoother and more professional looking graphics. Text displayed in a graph is always antialiased. Antialiasing is very time-consuming for larger graphical displays, and its benefits decrease as the number of points increases, so it is turned off by default for plots with many points. If the number of observations in the ODS output object exceeds the ANTIALIASMAX= threshold (10,000 by default), then antialiasing is not used, even if you specify the option ANTIALIAS=ON. The default is ANTIALIAS=ON.

ANTIALIASMAX=*n*

specifies the maximum number of markers or lines to be antialiased before antialiasing is disabled. For example, if there are more than 10,000 point markers and ANTIALIASMAX=10,000 (the default), then no markers are antialiased.

BORDER=ON | OFF

specifies whether to draw the graph with a border. BORDER=ON is the default.

HEIGHT=*dimension*

specifies the height of the graph. The default is HEIGHT=480PX (480 pixels). You can also specify height in inches (for example, HEIGHT=5IN) or centimeters (for example, HEIGHT=12CM).

IMAGEMAP=ON | OFF

controls tooltip generation in the HTML destination. The default is IMAGEMAP=OFF, which means that no tooltips are generated. Tooltips are text boxes that appear in HTML output when you rest your mouse pointer over a part of the plot (see Example 21.1).

IMAGENAME=< *base-file-name* **>**

specifies the base image filename. The default is the name of the output object. You can determine the name of the output object by using the ODS TRACE statement (see the section "Determining Graph Names and Labels" on page 628). The base image name should not include an extension. ODS automatically adds the increment value and the appropriate extension (which is specific to the output destination). See the section "Specifying Base Filenames" on page 635 for an example.

LABELMAX=*n*

specifies the maximum number of labeled areas before labeling is disabled. For example, if LABELMAX=50, and there are more than 50 points with labels, then no points are labeled. The default is LABELMAX=200.

MAXLEGENDAREA=*n*

specifies the maximum percentage of the overall graph area that a legend can occupy. The default is MAXLEGENDAREA=20. Larger legends are dropped from the display.

OUTPUTFMT=< *image-file-type* **| STATIC >**

specifies the image format for graphs. The OUTPUTFMT= option was previously named the IMAGEFMT= option. By default, OUTPUTFMT=STATIC and ODS dynamically uses the best quality static image format for the active output destination. The available image formats include: BMP (Microsoft Windows device-independent bitmap), DIB (Microsoft Windows device-independent bitmap), EMF (Microsoft NT enhanced metafile), EPSI (Adobe encapsulated PostScript interchange), GIF (graphic interchange format), JFIF (JPEG file interchange format), JPEG (Joint Photographic Experts Group), PBM (portable bitmap), PCD (Photo CD), PCL (Printer Command Language), PDF

(portable document format), PICT (the QuickDraw picture format), PNG (Portable Network Graphics), PS (PostScript image file format), SVG (Scalable Vector Graphics), TIFF (Tagged Image File Format), WMF (Microsoft Windows Metafile format), XBM (X Bitmap), and XPM (X-Windows Pixelmap). If the specified image format is not valid for the active output destination, the device is automatically remapped to the default image format.

RESET< *=option* **>**
resets one or more ODS GRAPHICS options to their default settings. The RESET and RESET=ALL options are equivalent. If you want to reset more than one option, but not all of the options, then you must specify RESET= separately for each option you reset (for example, **`ods graphics on / reset=antialias reset=index;`**). The RESET= options include the following:

ALL
resets all of the resettable options to their defaults.

ANTIALIAS
resets the ANTIALIAS= option to its default.

ANTIALIASMAX
resets the ANTIALIASMAX= option to its default.

BORDER
resets the BORDER= option to its default.

INDEX
resets the index counter that is appended to static image files.

HEIGHT
resets the HEIGHT= option to its default.

IMAGEMAP
resets the IMAGEMAP= option to its default.

LABELMAX
resets the LABELMAX= option to its default.

SCALE
resets the SCALE= option to its default.

TIPMAX
resets the TIPMAX= option to its default.

WIDTH
resets the WIDTH= option to its default.

SCALE=ON | OFF
specifies whether the fonts and symbol markers are scaled proportionally with the size of the graph. The default is SCALE=ON. For examples, see Figure 21.52 and Figure 21.53.

SCALEMARKERS=ON | OFF
specifies whether markers are scaled in nested layouts. The default is SCALE=ON.

TIPMAX=*n*

specifies the maximum number of distinct tooltips permitted before tooltips are disabled. Tooltips are text boxes that appear when you rest your mouse pointer over a part of the plot. For example, if TIPMAX=400, and there are more than 400 points in a scatter plot, then no tooltips appear. The default is TIPMAX=500.

WIDTH=*dimension*

specifies the width of the graph. The default is WIDTH=640PX (640 pixels). You can also specify widths in inches (for example, WIDTH=5in) or centimeters (for example, WIDTH=12cm).

ODS Destination Statements

ODS has a number of statements that control the destination of ODS output. The ODS destination statements that are most commonly used with ODS Graphics are: ODS DOCUMENT, ODS HTML, ODS LATEX, ODS LISTING, ODS PCL, ODS PDF, ODS PS, and ODS RTF. Specifying a statement opens a destination, unless the CLOSE option is specified. Each of the following statements opens an ODS destination:

```
ods html;
ods rtf;
ods html image_dpi=300;
ods listing style=HTMLBlue;
```

Each of the following statements closes an ODS destination:

```
ods html close;
ods rtf close;
ods listing close;
```

The following statement closes all open destinations:

```
ods _all_ close;
```

The following two options are commonly used in ODS destination statements to control aspects of ODS Graphics:

IMAGE_DPI=*dpi*

specifies the dots per inch (DPI), which is the image resolution for graphical output. The default varies depending on the destination. For example, the default is 96 for HTML and 200 for RTF.

STYLE=*style-name*

specifies the output style. Commonly used styles include HTMLBLUE, HTMLBLUECML, DEFAULT, LISTING, STATISTICAL, JOURNAL, JOURNAL2, JOURNAL3, RTF, and ANALYSIS.

Other options provide you with ways to control the files that are created. For example, the following statement opens the HTML destination:

```
ods html body='b.html' contents='c.html' frame='a.html';
```

This statement also writes the body of the output to the file *b.html*, the table of contents to the file *c.html*, and an overall frame that contains both the contents and the output to the file *a.html*. Alternatively, you can specify FILE= instead of BODY=.

If you are using a destination for which individual graphs are created (for example, LISTING or HTML), you can use the GPATH= option to specify the directory where your graphics files are saved, as in the following example:

```
ods html gpath="C:\figures";
```

See the sections "Image File Types" on page 632, "Saving Graphics Image Files" on page 636, "LISTING Destination" on page 637, "HTML Destination" on page 637, and "LATEX Destination" on page 637 for more information about individual image files and options specified in the ODS Destination statements. For complete details about the ODS destination statements, see the *SAS Output Delivery System: User's Guide*.

PLOTS= Option

Each statistical procedure that supports ODS Graphics has a PLOTS= option that is used to select graphs and specify some options. The syntax of the PLOTS= option is as follows:

PLOTS <(*global-plot-options*)> <= *plot-request* <*(options)*>>
PLOTS <(*global-plot-options*)> <= (*plot-request* <*(options)*> <... *plot-request* <*(options)*>>)>

The PLOTS= option has a common overall syntax for all statistical procedures, but the specific global plot options, plot requests, and plot options vary across procedures. This section discusses only a few of the options available in the PLOTS= option. For more information about the PLOTS= option, see the "Syntax" section for each procedure that produces ODS Graphics. There are only a limited number of things that you can control with the PLOTS= option. Most graphical details are controlled either by graph templates (see the section "Graph Templates" on page 711 in Chapter 22, "ODS Graphics Template Modification,") or by styles (see the section "Styles" on page 646).

The PLOTS= option usually appears in the PROC statement. However, for some procedures, certain analyses and hence certain plots can appear only if an additional statement is specified. These procedures often have a PLOTS= option in that other statement. For example, the PHREG procedure has a PLOTS= option in the BAYES statement, which is used to perform a Bayesian analysis. See the "Syntax" section of each procedure chapter for more information. The following examples illustrate the syntax of the PLOTS= option:

```
plots=all
plots=none
plots=residuals
plots=residuals(smooth)
plots=(trace autocorr)
plots(unpack)
plots(unpack)=diagnostics
plots=diagnostics(unpack)
plots(only)=freqplot
plots=(scree(unpack) loadings(plotref) preloadings(flip))
plots(unpack maxparmlabel=0 stepaxis=number)=coefficients
plots(sigonly)=(rawprob adjusted(unpack))
```

Also see the "Getting Started" sections "Survival Estimate Plot with PROC LIFETEST" on page 595, "Contour and Surface Plots with PROC KDE" on page 596, "Contour Plots with PROC KRIGE2D" on page 598, "LS-Means Diffogram with PROC GLIMMIX" on page 604, and "Principal Component Analysis Plots with PROC PRINCOMP" on page 606 for examples of the PLOTS= option.

The simplest PLOTS= specifications are of the form PLOTS=*plot-request* or PLOTS=(*plot-requests*). When there is more than one plot request, the plot-request list must appear in parentheses. Each plot request either requests a plot (for example, RESIDUALS) or provides you with a place to specify plot-specific options (for example, DIAGNOSTICS(UNPACK)). Some simple and typical plot requests are explained next:

- PLOTS=ALL requests all plots that are relevant to the analysis. This does not mean that all plots that the procedure can produce are produced. Plots that are produced for one set of options might not appear with PLOTS=ALL and a different set of options. In some cases, certain plots are not produced unless certain options or statements outside the PLOTS= option are specified.
- PLOTS=NONE disables ODS Graphics for just that step. You can use this option instead of specifying ODS GRAPHICS OFF before a procedure step and ODS GRAPHICS ON after the step when you want to suppress graphics for only that step.
- PLOTS=RESIDUALS requests a plot of residuals in a modeling procedure such as PROC REG.
- PLOTS=RESIDUALS(SMOOTH) requests the residuals plot along with a smooth fit function.
- PLOTS=(TRACE AUTOCORR) requests trace and autocorrelation plots in procedures with Bayesian analysis options.

Global plot options appear in parentheses after the option name and before the equal sign. These options affect many or all of the plots. The UNPACK option is a commonly used global plot option. It specifies that plots that are normally produced with multiple plots per panel (or "packed") should be unpacked and appear in multiple panels with one plot in each panel. The specification PLOTS(UNPACK)=(*plot-requests*) unpacks all paneled plots. The UNPACK option is also used as an option in a plot request when you want to unpack only certain panels. For example, the option PLOTS=(DIAGNOSTICS(UNPACK) PARTIAL PREDICTIONS) unpacks only the diagnostics panel. In some cases, unpacked plots contain additional information that is not found in the smaller packed versions. The UNPACK option is not available for all plot requests; it is available only with plots that have multiple panels by default.

Another commonly used global plot option is the ONLY option. Many procedures produce default plots, and additional plots can be requested in the PLOTS= option. Specifying PLOTS=(*plot-requests*) while omitting the default plots does not prevent the default plots from being produced. The ONLY option is used when you want to see only the plots specifically listed in the plot-request list. Procedures that produce no default plots typically do not provide an ONLY option. You can use ODS SELECT and ODS EXCLUDE (see the section "Selecting and Excluding Graphs" on page 631) to select and exclude graphs, but in some situations the ONLY option is more convenient. It is typically more efficient to select plots by using the PLOTS(ONLY)= option, because the procedure does not do extra work to generate a plot that is excluded by the PLOTS(ONLY)= option. In contrast, ODS SELECT and ODS EXCLUDE have their effect after the procedure has done the work to generate the plot.

Selecting and Viewing Graphs

This section describes techniques for selecting and viewing your graphs. Topics include:

- specifying an ODS destination for graphics
- viewing your graphs in the SAS windowing environment
- referring to graphs by name when using ODS
- selecting and excluding graphs from your output

Specifying an ODS Destination for Graphics

If you do not specify an ODS destination, then either the LISTING or the HTML destination is used by default. Here is an example of how you can explicitly specify the HTML destination:

```
ods graphics on;
ods html;

proc reg data=sashelp.class;
   model Weight = Height;
run; quit;

ods html close;
```

This ODS HTML statement creates an HTML file with a default name. See the section "Specifying a File for ODS Output" on page 627 to see how to specify a filename. Other destinations are specified in a similar way. For example, you can specify an RTF destination with the following statements:

```
ods graphics on;
ods rtf;

. . .

ods rtf close;
```

The destinations that ODS supports for graphics are as follows:

Destination	Destination Family
DOCUMENT	
HTML	MARKUP
LATEX	MARKUP
LISTING	
PCL	PRINTER
PDF	PRINTER
PS	PRINTER
RTF	

You can close all open destinations if you are interested only in displaying your output in a nondefault destination. For example, if you want to see your output only in the RTF destination, you can specify the following statements:

```
ods graphics on;
ods _all_ close;
ods rtf;

. . .

ods rtf close;
ods listing;
```

Closing unneeded destinations makes your jobs run faster and creates fewer files. More generally, it makes your jobs consume fewer resources, because a graph is otherwise created for every open destination. The last statement opens the LISTING destination after you are finished using the RTF destination.

You can also use the ODS OUTPUT destination to create an output data set from the data object used to make a plot. Here is an example:

```
ods graphics on;

proc reg data=sashelp.class;
   ods output fitplot=myfitplot;
   model Weight = Height;
run; quit;
```

Specifying a File for ODS Output

You can specify a filename for your output with the FILE= option in the ODS destination statement, as in the following example:

```
ods html file="test.htm";
```

The output is written to the file *test.htm*, which is saved in the SAS current folder. At start-up, the SAS current folder is the same directory in which you started your SAS session. If you are using the SAS windowing environment, then the current folder is displayed in the status line at the bottom of the main SAS

window. If you do not specify a filename for your output, then the SAS System provides a default filename, which depends on the ODS destination. This file is saved in the SAS current folder. You can always check the SAS log to verify the name of the file in which your output is saved. For example, suppose you specify the following statement:

```
ods html;
```

Then the following message is displayed in the SAS log:

```
NOTE: Writing HTML Body file: sashtml.htm
```

The default filenames for each destination are specified in the SAS Registry. For example, Figure 21.54 shows that the default filename in the SAS Registry for the RTF destination is *sasrtf.rtf*. For more information, see the *SAS Companion* for your operating system.

Viewing Your Graphs in the SAS Windowing Environment

The mechanism for viewing graphs created with ODS can vary depending on your operating system, which viewers are installed on your computer, and the ODS destination you have selected. If you do not specify an ODS destination, then the default destination is either HTML or LISTING.

If you are using the SAS windowing environment and the HTML destination, then the results are displayed by default in the SAS Results Viewer unless you chose to use an external browser. To use an external viewer, select **Tools ► Options ► Preferences** from the menu at the top of the main SAS window. Then select the **Results** and **Web** tabs to make your selection.

If you are using the LATEX or the PS destinations, you must use a PostScript viewer, such as GSview. For information about the windowing environment in a different operating system, see the *SAS Companion* for that operating system.

If you do not want to view the results as they are being generated, then select **Tools ► Options ► Preferences** from the menu at the top of the main SAS window. Then on the **Results** tab, clear the **View results as they are generated** checkbox.

If you are using the SAS windowing environment and the LISTING destination, go to the Results window and find the icon for the corresponding graph. You can double-click the graph icon to display the graph in the default viewer that is configured on your computer for the corresponding image file type (see Figure 21.13).

Determining Graph Names and Labels

Procedures assign a name to each graph they create with ODS Graphics. This enables you to refer to ODS graphs in the same way that you refer to ODS tables (see the section "The ODS Statement" on page 531 in Chapter 20, "Using the Output Delivery System,"). You can determine the names of graphs in several ways:

- You can look up graph names in the "ODS Graphics" section of chapters for procedures that use ODS Graphics. For example, see the section "ODS Graphics" on page 6437 in Chapter 76, "The REG Procedure."
- You can use the Results window to view the names of ODS graphs created in your SAS session. See the section "The SAS Results Window" on page 536 in Chapter 20, "Using the Output Delivery System," for more information.
- You can use the ODS TRACE ON statement to list the names of graphs created by your SAS session. This statement adds identifying information in the SAS log (or optionally in the SAS LISTING) for each graph that is produced. See the section "The ODS Statement" on page 531 in Chapter 20, "Using the Output Delivery System," for more information.

The graph name is not the same as the name of the image file that contains the graph (see the section "Naming Graphics Image Files" on page 634).

This example revisits the analysis described in the section "Contour and Surface Plots with PROC KDE" on page 596. To determine which output objects are created by ODS, you specify the ODS TRACE ON statement prior to the procedure statements as follows:

```
ods graphics on;
ods trace on;

proc kde data=bivnormal;
   bivar x y / plots=contour surface;
run;

ods trace off;
```

The trace record from the SAS log is as follows:

```
Output Added:
-------------
Name:        Inputs
Template:    Stat.KDE.Inputs
Path:        KDE.Bivar1.x_y.Inputs
-------------

Output Added:
-------------
Name:        Controls
Template:    Stat.KDE.Controls
Path:        KDE.Bivar1.x_y.Controls
-------------

Output Added:
-------------
Name:        ContourPlot
Label:       Contour Plot
Template:    Stat.KDE.Graphics.Contour
Path:        KDE.Bivar1.x_y.ContourPlot
-------------
```

```
Output Added:
-------------
Name:        SurfacePlot
Label:       Density Surface
Template:    Stat.KDE.Graphics.Surface
Path:        KDE.Bivar1.x_y.SurfacePlot
-------------
```

By default, PROC KDE creates table objects named **Inputs** and **Controls**, and it creates graph objects named **ContourPlot** and **SurfacePlot**. In addition to the name, the trace record provides the label, template, and path for each output object. Graph templates are distinguished from table templates by a naming convention that uses the procedure name in the second level and **Graphics** in the third level. For example, the fully qualified template name for the surface plot created by PROC KDE is **Stat.KDE.Graphics.SurfacePlot**.

You can specify the LISTING option in the ODS TRACE ON statement to write the trace record to the LISTING destination as follows:

```
ods trace on / listing;
```

Each table and graph has a path (or name path), which was previously shown in the trace output. The path consists of the plot name preceded by the names of one or more output groups. Each table and graph also has a label path, which can be seen by adding the LABEL option to the ODS TRACE ON statement, after a slash, as follows:

```
ods trace on / label;

proc kde data=bivnormal;
   bivar x y / plots=contour surface;
run;

ods trace off;
```

A portion of the trace output is shown next:

```
Path:       KDE.Bivar1.x_y.Inputs
Label Path: 'The KDE Procedure'.'Bivariate Analysis'.'x and y'.'KDE.Bivar1.x_y'

Path:       KDE.Bivar1.x_y.Controls
Label Path: 'The KDE Procedure'.'Bivariate Analysis'.'x and y'.'KDE.Bivar1.x_y'

Path:       KDE.Bivar1.x_y.ContourPlot
Label Path: 'The KDE Procedure'.'Bivariate Analysis'.'x and y'.'Contour Plot'

Path:       KDE.Bivar1.x_y.SurfacePlot
Label Path: 'The KDE Procedure'.'Bivariate Analysis'.'x and y'.'Density Surface'
```

The label path contains the information that you see in the HTML table of contents. Names are fixed, they do not vary, and they are not data- or context-dependent. In contrast, labels often reflect data- or context-dependent information.

Selecting and Excluding Graphs

You can use the ODS SELECT and ODS EXCLUDE statements along with graph and table names to specify which ODS outputs are displayed. See the section "The ODS Statement" on page 531 in Chapter 20, "Using the Output Delivery System," for more information about how to use these statements.

This section shows several examples of selecting and excluding graphs by using the data set and trace output created in the section "Determining Graph Names and Labels" on page 628. The following statements use the ODS SELECT statement to select only two graphs, **ContourPlot** and **SurfacePlot**, for display in the output:

```
proc kde data=bivnormal;
   ods select ContourPlot SurfacePlot;
   bivar x y / plots=contour surface;
run;
```

Equivalently, the following statements use the ODS EXCLUDE statement to exclude the two tables:

```
proc kde data=bivnormal;
   ods exclude Inputs Controls;
   bivar x y / plots=contour surface;
run;
```

You can select or exclude graphs by using either the name or the label. Labels must be specified in quotes. In the context of this example, the following two statements are equivalent:

```
ods select contourplot;
ods select 'Contour Plot';
```

You can also specify multiple levels of the path, as in the following example:

```
ods select x_y.contourplot;
ods select 'x and y'.'Contour Plot';
ods select 'x and y'.contourplot;
ods select x_y.'Contour Plot';
```

Name and label paths can be mixed, as in the last two statements. All four of the preceding statements select the same plot. Furthermore, selection based directly on the names and labels is case-insensitive. The following statements all select the same plot:

```
ods select x_y.contourplot;
ods select 'x and y'.'Contour Plot';
ods select X_Y.CONTOURPLOT;
ods select 'X AND Y'.'CONTOUR PLOT';
```

It is sometimes useful to specify a WHERE clause in an ODS SELECT or ODS EXCLUDE statement. This enables you to specify expressions based on either the name path or the label path. You can base your selection on two automatic variables _path_ and _label_. The following two statements select every object whose path contains the string 'plot' and every object whose label path contains the string 'plot', respectively, ignoring the case in the name and label:

```
ods select where = (lowcase(_path_)  ? 'plot');
ods select where = (lowcase(_label_) ? 'plot');
```

The question mark operator means that the second expression (the string 'plot') is contained in the first expression (the lowercase version of the name or label). For example, all of the following names match 'plot' in the WHERE clause: plot, SurfacePlot, SURFACEPLOT, FitPlot, pLoTtInG, Splotch, and so on. Since WHERE clause selection is based on SAS string comparisons, selection is case-sensitive. The LOWCASE function is used to ensure a match even when the specified string does not match the case of the actual name or label.

WHERE clauses are particularly useful when you want to select all of the objects in a group. A group is a level of the name path or label path hierarchy before the last level. In the following step, all of the objects whose name path contains 'DiagnosticPlots' are selected:

```
proc reg data=sashelp.class plots(unpack);
   ods select where = (_path_ ? 'DiagnosticPlots');
   model Weight = Height;
run; quit;
```

These are the plots that come from unpacking the PROC REG diagnostics panel of plots. All are in a group named 'DiagnosticPlots'.

Graphics Image Files

Accessing your graphs as individual image files is useful when you want to include them in various types of documents. The default image file type depends on the ODS destination, but there are other supported image file types that you can specify. You can also specify the names for your graphics image files and the directory in which you want to save them. This section describes the image file types supported by ODS Graphics, and it explains how to name and save graphics image files.

Image File Types

If you are using the LISTING, HTML, or LATEX destinations, your graphs are individually produced in a specific image file type, such as PNG (Portable Network Graphics). If you are using a destination in the PRINTER family or the RTF destination, the graphs are contained in the ODS output file and cannot be accessed as individual image files. However, you can open an RTF output file in Microsoft Word and then copy and paste the graphs into another document, such as a Microsoft PowerPoint presentation. This is illustrated in Example 21.2.

Table 21.1 shows the various ODS destinations supported by ODS Graphics, the viewer that is appropriate for displaying graphs in each destination, and the image file types supported for each destination.

Table 21.1 Destinations and Image File Types Supported by ODS Graphics

Destination	Destination Family	Recommended Viewer	Image File Types
DOCUMENT		Not applicable	Not applicable
HTML	MARKUP	Web browser	PNG (default), GIF, JPEG,
LATEX	MARKUP	PostScript or PDF viewer after compiling the LaTeX file	PostScript (default), EPSI, GIF, JPEG, PDF, PNG
LISTING		Default viewer in your system for the specified file type	PNG (default), GIF, BMP, DIB, EMF, EPSI, GIF, JFIF, JPEG, PBM, PS, TIFF, WMF
PCL	PRINTER	Not applicable	Contained in PRN file
PDF	PRINTER	PDF viewer, such as Adobe Reader	Contained in PDF file
PS	PRINTER	PostScript viewer, such as GSview	Contained in PostScript file
RTF		Word processor, such as Microsoft Word	Contained in RTF file

For destinations such as PDF and RTF, you can control the types of the images that are contained in the file even though individual files are not made for each image. The default image file type is PNG, and other image types are available. See the *SAS Output Delivery System: User's Guide* for more information.

Scalable Vector Graphics

Scalable vector graphics output is now supported in ODS Graphics. The output type support varies based on the ODS destination that you use. You can specify the OUTPUTFMT= option in the ODS GRAPHICS statement to specify the output type for any destination. For destinations that generate vector graphics by default, you can get image output by specifying the option OUTPUTFMT=STATIC.

Vector graphics are not supported for all graph types. When vector graphics are requested but not supported, the graph automatically changes to image output. Vector graphics are not supported for the following graph types:

- three-dimensional graph
- contour plots with smooth gradient fills
- graphs with continuous legends
- graphs with data skins
- graphs with rotated annotation images
- graphs with transparency (EMF and PS only)

The LISTING destination can generate all of the supported forms of vector-based output: PDF, PS, EMF, SVG, and PCL. Each graph is generated in a separate file that can be included into a larger report. The default output format is a PNG image.

Like the LISTING destination, the ODS PRINTER destination can generate all of the supported vector output types. The output format depends on the type of printer you select. If you select the PDF, SVG, or PCL5C printers, vector-based output is automatically produced. However, if you select PS or EMF printers, you need to set the OUTPUTFMT= option in the ODS GRAPHICS statement to PS or EMF, respectively, to create vector-based output. By default, the output from this destination is in one file instead of individual files for each graph.

The PDF destination renders PDF vector output by default, except for the exceptions noted. You can specify the OUTPUTFMT=STATIC option in the ODS GRAPHICS statement to produce an embedded image in the PDF file.

The LATEX destination renders PS vector output by default, except for the exceptions noted. You can specify the OUTPUTFMT=STATIC option in the ODS GRAPHICS statement to produce an embedded image in the postscript file.

The RTF destination renders PNG image output by default. Vector-based EMF output is also supported for this destination. You can specify the OUTPUTFMT=EMF option in the ODS GRAPHICS statement to select this output type. If one of the noted exceptions occurs, the output type for that graph changes to a PNG image.

The HTML destination renders PNG image output by default. Vector-based SVG output is also supported for this destination. You can specify the OUTPUTFMT=SVG option in the ODS GRAPHICS statement to select this output type. If one of the noted exceptions occurs, the output type for that graph changes to a PNG image.

In most cases, the file size with vector graphics is much smaller than a comparable static image file. However, in some cases, the vector graphics file size is larger than the image version. This is likely for scatter plots of data sets with a large number of observations.

Naming Graphics Image Files

The following discussion applies to the destinations where ODS graphs are created as individual image files (for example, HTML, LISTING, and LATEX). The names of graphics image files are determined by a base filename, an index counter, and an extension. By default, the base filename is the ODS graph name (see the section "Determining Graph Names and Labels" on page 628). There is an index counter for each base filename. The extension indicates the image file type. The first time a graph object with a given base filename is created, the filename consists only of the base filename and the extension. If a graph with the same base filename is created multiple times, then an index counter is appended to the base filename to avoid overwriting previously created images.

To illustrate, consider the following statements:

```
proc kde data=bivnormal;
   ods select ContourPlot SurfacePlot;
   bivar x y / plots=contour surface;
run;
```

If you run this step at the beginning of a SAS session, the two graphics image files created are *ContourPlot.png* and *SurfacePlot.png*. If you immediately rerun these statements, then ODS creates the same graphs in different image files named *ContourPlot1.png* and *SurfacePlot1.png*. The next time, the image files are named *ContourPlot2.png* and *SurfacePlot2.png*. The index starts at zero, and one is added each time the same name is used. Note, however, that when the index is at zero, 0 is not added to the filename.

Resetting the Index Counter

You can specify the RESET=INDEX option in the ODS GRAPHICS statement to reset the index counter. This is useful when you need to have predictable names. It is particularly useful when you are running a SAS program multiple times in the same session. The following statement resets the index:

```
ods graphics on / reset=index;
```

The index counter is reinitialized at the beginning of your SAS session or if you specify the RESET=INDEX option in the ODS GRAPHICS statement. Graphics image files with the same name are overwritten.

Specifying Base Filenames

You can specify a base filename for all your graphics image files with the IMAGENAME= option in the ODS GRAPHICS statement as follows:

```
ods graphics on / imagename="MyName";
```

You can also specify the RESET=INDEX option as follows:

```
ods graphics on / reset=index imagename="MyName";
```

The IMAGENAME= option overrides the default base filename. With the preceding statement, the graphics image files are named *MyName*, *MyName1*, *MyName2*, and so on.

Specifying Image File Types

You can specify the image file type for the LISTING, HTML, or LATEX destinations with the OUTPUTFMT= option in the ODS GRAPHICS statement as follows:

```
ods graphics on / outputfmts=gif;
```

For more information, see the section "ODS GRAPHICS Statement" on page 620.

Naming Graphics Image Files with Multiple Destinations

Since the index counter depends only on the base filename, if you specify multiple ODS destinations for your output, then the index counter is increased independently of the destination. For example, the following statements create image files named *ContourPlot.png* and *SurfacePlot.png* that correspond to the LISTING destination, and *ContourPlot1.png* and *SurfacePlot1.png* that correspond to the HTML destination:

```
ods listing;
ods html;
ods graphics on / reset;

proc kde data=bivnormal;
   ods select ContourPlot SurfacePlot;
   bivar x y / plots=contour surface;
run;

ods _all_ close;
ods listing;
```

When you specify one of the destinations in the PRINTER family or the RTF destination, your ODS graphs are embedded in the document, so the index counter is not affected. For example, the following statements create image files *ContourPlot.png* and *SurfacePlot.png* for the LISTING destinations, but no image files for the RTF destination:

```
ods listing;
ods rtf;
ods graphics on / reset;

proc kde data=bivnormal;
   ods select ContourPlot SurfacePlot;
   bivar x y / plots=contour surface;
run;

ods _all_ close;
```

Saving Graphics Image Files

Knowing where your graphics image files are saved and how they are named is particularly important if you are running in batch mode, if you have disabled the SAS Results window (see the section "Viewing Your Graphs in the SAS Windowing Environment" on page 628), or if you plan to access the files for inclusion in a paper or presentation. The following discussion assumes you are running SAS under the Windows operating system. If you are running on a different operating system, see the *SAS Companion* for your operating system.

In the SAS windowing environment, the current folder is displayed in the status line at the bottom of the main SAS window. When **Use WORK folder** is cleared in the **Tools ► Options ► Preferences ► Results** tab, graph image files are saved in the current folder and are available after your SAS session ends. They can accumulate with time and take up a great deal of space. When **Use WORK folder** is selected, graph image files are stored in the Work folder and are not available after your SAS session ends.

If you are running your SAS programs in batch mode, the graphs are saved by default in the same directory where you started your SAS session. For example, suppose the SAS current folder is *C:\myfiles*. If ODS Graphics is enabled, then your graphics image files are saved in the directory *C:\myfiles*. Traditional graphics are always saved in a catalog in your Work directory.

With the LISTING, HTML, and LATEX destinations, you can specify a directory for saving your graphics image files. With a destination in the PRINTER family and with the RTF destination, you can specify a directory only for your output file. The remainder of this discussion provides details for each destination type.

LISTING Destination

If you are using the LISTING destination, the individual graphs are created as PNG files by default. You can use the GPATH= option in the ODS LISTING statement to specify the directory where your graphics files are saved. For example, if you want to save your graphics image files in *C:\figures*, then you can specify the following:

```
ods listing gpath="C:\figures";
```

It is important to note that the GPATH= option applies only to ODS Graphics. It does not affect the behavior of graphs created with traditional SAS/GRAPH procedures.

HTML Destination

If you are using the HTML destination, the individual graphs are created as PNG files by default. You can use the PATH= and GPATH= options in the ODS HTML statement to specify the directory where your HTML and graphics files are saved, respectively. This also gives you more control over your graphs. For example, if you want to save your HTML file named *test.htm* in the *C:\myfiles* directory, but you want to save your graphics image files in *C:\myfiles\png*, then you can specify the following:

```
ods html path  = "C:\myfiles"
         gpath = "C:\myfiles\png"
         file  = "test.htm";
```

When you specify the URL= suboption with the GPATH= option, SAS creates relative paths for the links and references to the graphics image files in the HTML file. This is useful for building output files that are easily moved from one location to another. For example, the following statements create a relative path to the *png* directory in all the links and references contained in *test.htm*:

```
ods html path  = "C:\myfiles"
         gpath = "C:\myfiles\png" (url="png/")
         file  = "test.htm";
```

If you do not specify the URL= suboption, SAS creates absolute paths that are hard-coded in the HTML file. These can cause broken links if you move the files. For more information, see the ODS HTML statement in the *SAS Output Delivery System: User's Guide*.

LATEX Destination

LaTeX is a document preparation system for high-quality typesetting. The ODS LATEX statement produces output in the form of a LaTeX source file that is ready to compile in LaTeX. When you request ODS Graphics for a LATEX destination, ODS creates the requested graphs as PostScript files by default, and the LaTeX

source file includes references to these image graphics files. You can compile the LaTeX file, or you can ignore this file and simply access the individual PostScript files to include your graphs in a different LaTeX document, such as a paper that you are writing. You can specify the PATH= and GPATH= options in the ODS LATEX statement, as explained previously for the ODS HTML statement. See Example 21.3 for an illustration. The ODS LATEX statement is an alias for the ODS MARKUP statement with the TAGSET=LATEX option. For more information, see the *SAS Output Delivery System: User's Guide*.

The default image file type for the LATEX destination is PostScript. When you use LaTeX to compile your document, the graphics format for included images is Postscript. However, if you prefer to use pdfLaTeX, you can specify a different format such as JPEG, PDF, or PNG, any of which can be directly included into your pdfLaTeXdocument. To specify one of these formats, you use the OUTPUTFMT= option in the ODS GRAPHICS statement. For more information, see the LaTeX documentation for the `graphicx` package.

Creating Graphs in Multiple Destinations

This section illustrates how to send your output to more than one destination with a single execution of your SAS statements. For example, to create LISTING, HTML, and RTF output, you can specify the ODS LISTING, ODS HTML, and the ODS RTF statements before your procedure statements. The ODS _ALL_ CLOSE statement closes all open destinations before and after the other statements are run.

```
ods _all_ close;
ods listing;
ods html;
ods rtf;

. . .

ods _all_ close;
```

You can also specify multiple instances of the same destination. For example, using the data in the section "Contour and Surface Plots with PROC KDE" on page 596, the following statements save the contour plot to the file *contour.pdf* and the surface plot to the file *surface.pdf*:

```
ods _all_ close;
ods pdf file="contour.pdf";
ods pdf select ContourPlot;
ods pdf(id=srf) file="surface.pdf";
ods pdf(id=srf) select SurfacePlot;
ods graphics on;

proc kde data=bivnormal;
   ods select ContourPlot SurfacePlot;
   bivar x y / plots=contour surface;
run;

ods _all_ close;
```

The ID= option assigns the name **srf** to the second instance of the PDF destination. Without the ID= option, the second ODS PDF statement closes the destination that was opened by the previous ODS PDF statement,

and it opens a new instance of the PDF destination. In that case, the file *contour.pdf* is not created. For more information, see the ODS PDF statement in the *SAS Output Delivery System: User's Guide*.

Graph Size and Resolution

ODS provides options for specifying the size and resolution of graphs. You can specify the size of a graph in the ODS GRAPHICS statement and the resolution in an ODS destination statement. There are two other ways to change the size of a graph, but they are rarely needed. The three methods are as follows:

- Usually, you specify the WIDTH= or HEIGHT= option (or both) in the ODS GRAPHICS statement to change the size of a graph.
- To modify the size of a particular graph, specify the dimensions with the DESIGNHEIGHT= and DESIGNWIDTH= options in the BEGINGRAPH statement in the template. Some templates contain the specification DESIGNWIDTH=DEFAULTDESIGNHEIGHT, which sets the width of the graph to the default height, or DESIGNHEIGHT=DEFAULTDESIGNWIDTH, which sets the height of the graph to the default width.
- To modify the size of all of your ODS graphs, specify the dimensions with the OUTPUTHEIGHT= and OUTPUTWIDTH= options in the style template.

The following examples show typical ways to change the size of your graphs:

```
ods graphics on / width=6in;
ods graphics on / height=4in;
ods graphics on / width=4.5in height=3.5in;
```

The dimensions of the graph can be specified in pixels (for example, 200PX), inches (for example, 3IN), or centimeters (for example, 8CM). The default dimensions of ODS Graphics are 640 pixels wide and 480 pixels high, and these values determine the default aspect ratio. The actual size of the graph in inches depends on your printer or display device. For example, if the resolution of your printer is 100 dots per inch and you want a graph that is 4 inches wide, you should set the width to 400 pixels.

If you specify only one dimension, the other is determined by the default aspect ratio—that is, height=0.75 $\times$ width. For best results, you should create your graphs by using the exact size that is used to display the graphs in your paper or presentation. In other words, avoid generating them at one size and then expanding or shrinking them for inclusion into the your document.

By default, fonts and symbol markers are automatically scaled with the size of the graph. You can suppress this scaling with the SCALE= option, as in the following example:

```
ods graphics on / scale=off;
```

The default resolution of graphs created with HTML and LISTING is 96 DPI (dots per inch), whereas the default with RTF is 200 DPI. The 200 DPI value is recommended if you are copying and pasting graphs into a Microsoft PowerPoint presentation or a Microsoft Word document. Graphs shown in SAS/STAT documentation are typically generated at 300 DPI for display in PDF and 96 DPI for display in HTML.

You can change the resolution with the IMAGE_DPI= option in any ODS destination statement, as in the following example:

```
ods html image_dpi=300;
```

An increase in resolution often improves the quality of the graphs, but it also greatly increases the size of the image file. Going from 96 DPI to 300 DPI increases the size of the image file by roughly a factor of $(300/96)^2 = 9.77$. Even when you are using a higher DPI for most of your graphs, you should consider using a lower DPI for some, such as contour plots, that create large files even at a lower DPI.

If you increase the resolution, you might need to compensate by reducing the size of the graph, as in the following example:

```
ods graphics on / width=4.5in  height=3.5in;
```

Increasing DPI also increases the amount of memory needed for your program to complete. You can increase the amount of memory available to ODS Graphics with an option when you invoke SAS, as in the following example:

```
-jreoptions '(-Xmx256m)'
```

You can modify the default amount of memory available to ODS Graphics by changing JREOPTIONS in your SAS configuration file to the settings –Xmx*nnn*m –Xms*nnn*m, where *nnn* is the amount of memory in megabytes. An example is –Xmx256m –Xms256m. In either case, the exact syntax varies depending on your operating system and the amount of memory that you can allocate varies from system to system. For more information, see the *SAS Companion* for your operating system.

ODS Graphics Editor

The ODS Graphics Editor is a point-and-click interface that you can use to modify a specific graph created by ODS Graphics. For example, if you need to enhance a graph for a paper or presentation, you can use the ODS Graphics Editor to customize the title, modify the axis labels, annotate particular data points, and change graph element properties such as fonts, colors, and line styles.

This section explains how to enable ODS Graphics to create editable graphs and how to invoke the ODS Graphics Editor. You can use the ODS Graphics Editor in the SAS windowing environment, provided that the LISTING destination is open and that you have first enabled ODS Graphics to create editable graphs. **NOTE:** The LISTING destination is typically open by default. There are three steps you must take to edit a graph:

1 You must first enable the creation of editable graphs in one of three ways:
   - use an ODS statement to temporarily enable this feature
   - use a SAS command to temporarily enable this feature
   - use the SAS Registry Editor to permanently enable this feature

   Creating editable graphs takes additional resources, so you might not want to permanently enable this feature.

2 You submit your SAS code and create editable graphs.

3 You invoke the ODS Graphics Editor and edit the plot.

Step 2 involves submitting SAS code in the usual way, and no special instructions are needed for creating graphs that can be edited. Steps 1 and 3 are explained in more detail in the following sections.

Enabling the Creation of Editable Graphs

Temporarily Enable Creation of Editable Graphs by Using an ODS Statement

You can enable the creation of editable graphs within a SAS session by submitting one of the following statements:

```
ods listing sge=on;
ods html sge=on;
```

You can disable the creation of editable graphs by submitting one of the following statements:

```
ods listing sge=off;
ods html sge=off;
```

Temporarily Enable Creation of Editable Graphs by Using a SAS Command

Alternatively, you can enable the creation of editable graphs for the duration of your SAS session by first selecting the Results window and then entering **`sgedit on`** on the command line. SAS confirms that the creation of editable graphs is enabled by displaying a message in the bottom left corner of the SAS window. The command must be entered from the Results window. If you enter it from any other window, it is ignored.

Permanently Enable Creation of Editable Graphs across SAS Sessions

You can create a default setting that enables or disables the creation of editable graphs across SAS sessions via the 'ODS Graphics Editor' setting in the SAS Registry. You can change this setting in the SAS windowing environment as follows:

1 Open the Registry Editor by entering **regedit** on the command line.

2 Select **SAS_REGISTRY ► ODS ► GUI ► RESULTS**.

3 In the **Value Data** field, click **ODS Graphics Editor** to open the Edit String Value window, and type **On** to enable the creation of editable graphs or type **Off** to disable it.

4 Click **OK**.

Editing a Graph with the ODS Graphics Editor

The ODS Graphics Editor is illustrated using the following example:

```
data sasuser.growth;
   input country $ GDP LFG EQP NEQ GAP;
   datalines;
Argentin  0.0089 0.0118 0.0214 0.2286 0.6079
Austria   0.0332 0.0014 0.0991 0.1349 0.5809

   ... more lines ...

Zambia   -0.0110 0.0275 0.0702 0.2012 0.8695
Zimbabwe  0.0110 0.0309 0.0843 0.1257 0.8875
;

ods graphics on;
ods html style=Statistical sge=on;

proc robustreg data=sasuser.growth
               plots=(ddplot histogram);
   model GDP  = LFG GAP EQP NEQ / diagnostics leverage;
   output out=robout r=resid sr=stdres;
run;

ods _all_ close;
ods listing;
```

The DATA and the PROC ROBUSTREG steps are submitted to the SAS System, in this case from the SAS windowing environment, as shown in Figure 21.16. Two versions of the graph are created: one in an uneditable PNG file (for example, *DDPlot.png*) and one in an editable SGE file (for example, *DDPlot.sge*). Both are saved in the SAS current folder. You can edit the graph in one of three ways:

- In the Results window, double-click the second graph icon for the graph you want to edit (see Figure 21.16). The second icon corresponds to the SGE file, and the first icon corresponds to the PNG file. Clicking the first graph icon invokes a host-dependent graph viewer (for example, Microsoft Photo Editor on Windows), not the ODS Graphics Editor. **NOTE:** The Editor window might be hidden behind other windows in the SAS windowing environment.
- You can edit the graph by selecting it in the SAS Explorer window. You must first navigate to the SAS current folder and to the SGE files.
- You can open the graph from outside of the SAS System. For example, if you are running the SAS System under the Windows operating system, you can click on the graph's SGE file to open it with the ODS Graphics Editor.

Figure 21.16 Results Window with Icons for Editable Plots

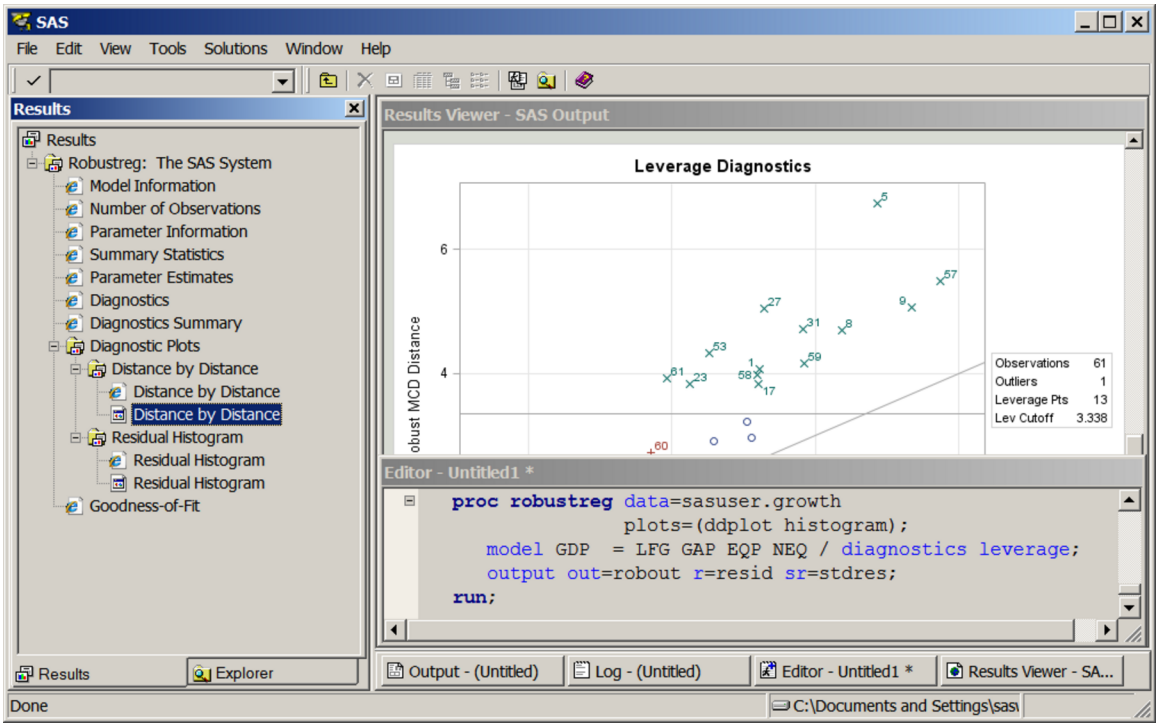


Figure 21.17 shows the ODS Graphics Editor window for the editable diagnostic plot created by PROC ROBUSTREG. In Figure 21.18, various tools in the ODS Graphics Editor are used to modify the title and annotate a particular point. The edited plot can be saved as a PNG file or as an SGE file by selecting **File ► Save As**. After saving the plot, you can edit it again through the SAS Explorer window or by selecting **File ► Open** from the ODS Graphics Editor window. Alternatively, you can reopen the saved plot for editing without first invoking the SAS System. For example, if you are running the SAS System under the Windows operating system, you can click on the plot to open it with the ODS Graphics Editor.

The ODS Graphics Editor does not permit you to make structural changes to a graph (such as moving the positions of data points). The ODS Graphics Editor provides you with a point-and-click way to make one-time changes to a specific graph, whereas the template language (see the section "Graph Templates" on page 711 in Chapter 22, "ODS Graphics Template Modification,") provides you with a programmatic way to make template changes that persist every time you run the procedure. For complete details about the tools available in the ODS Graphics Editor, see *SAS ODS Graphics Editor: User's Guide*.

Figure 21.17 Diagnostic Plot before Editing

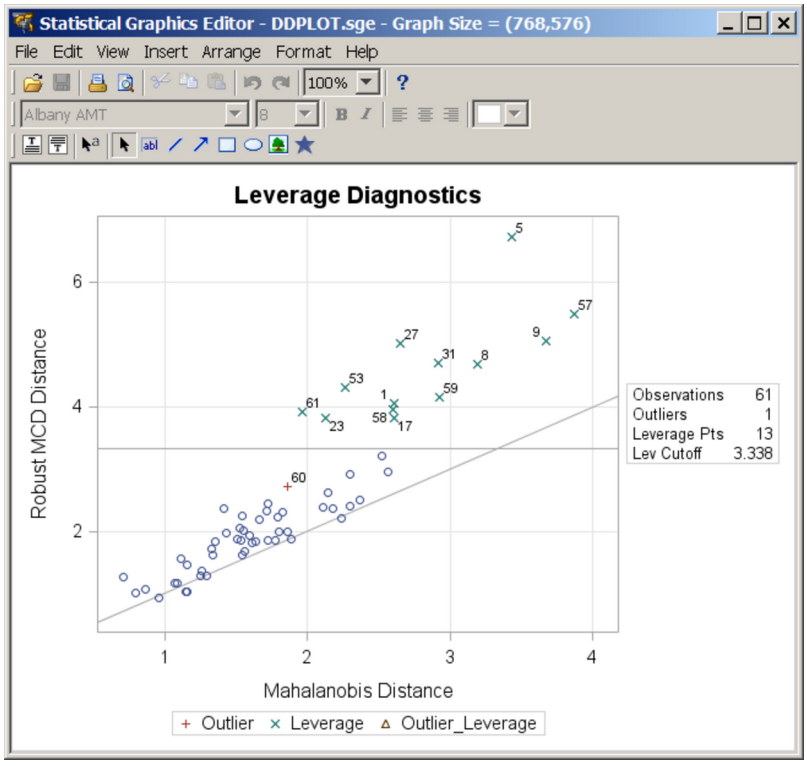
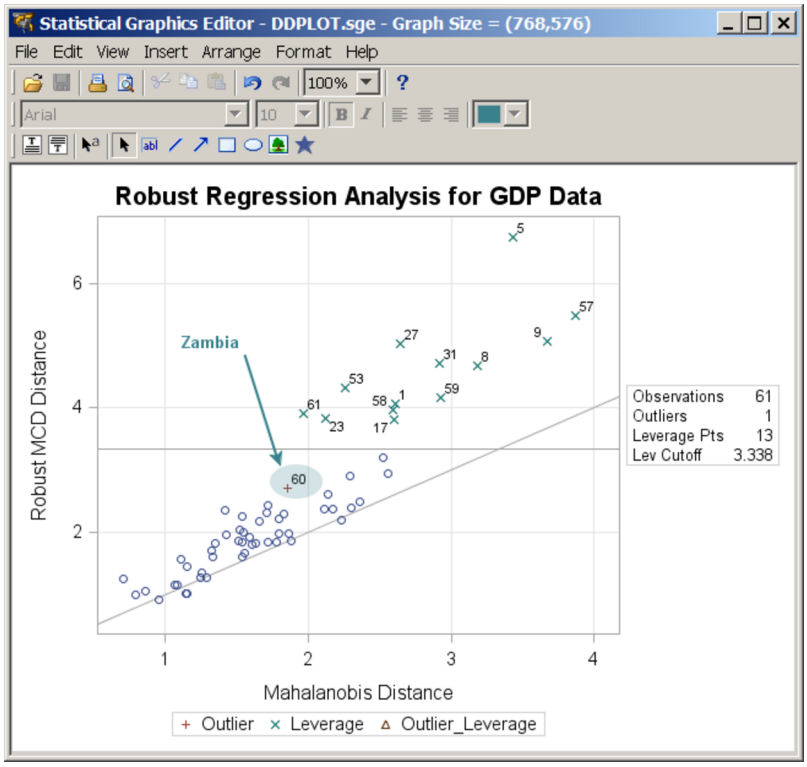


Figure 21.18 Diagnostic Plot after Editing



The Default Template Stores and the Template Search Path

Compiled templates are stored in a template store, which is a type of item store. (An item store is a special type of SAS file.) You can see the list of template stores by submitting the following statement:

```
ods path show;
```

The results are as follows:

```
Current ODS PATH list is:

1. SASUSER.TEMPLAT(UPDATE)
2. SASHELP.TMPLMST(READ)
```

These results show that the default template search path consists of Sasuser.Templat followed by Sashelp.Tmplmst. You can add template stores that you create or change the order in which the template stores are searched. This is discussed in detail in the sections "Saving Customized Templates" on page 720, "Using Customized Templates" on page 721, and "Reverting to the Default Templates" on page 722 in Chapter 22, "ODS Graphics Template Modification."

This section discusses the default template stores that you use when you have not modified the template search path with the ODS PATH statement. By default, templates that you write are stored in Sasuser.Templat. If you stored a modified template in Sasuser.Templat, ODS finds and uses your modified template. Otherwise, ODS finds the templates it provides in Sashelp.Tmplmst. You can see a list of all of the templates that you have modified as follows:

```
proc template;
   list / store=sasuser.templat;
run;
```

You can delete any template that you modified (so that ODS finds the default SAS template) by specifying it in a DELETE statement, as in the following statement:

```
proc template;
   delete Stat.REG.Graphics.ResidualPlot;
run;
```

Unless you have administrator privileges, ODS never deletes a template in Sashelp.Tmplmst, so you can safely run the preceding step, even if the template you specify does not exist in Sasuser.Templat. You can run the following step to delete the entire Sasuser.Templat template store of customized templates so that ODS uses only the templates supplied by the SAS System:

```
ods path sashelp.tmplmst(read);
proc datasets library=sasuser nolist;
   delete templat(memtype=itemstor);
run;
ods path sasuser.templat(update) sashelp.tmplmst(read);
```

It is good practice to delete templates that you have customized when you are done with them, so that they are not unexpectedly used later. See the section "Reverting to the Default Templates" on page 722 in Chapter 22, "ODS Graphics Template Modification," for more information.

Styles

ODS styles control the overall appearance of your output. Usually, the only thing you need to do with styles is specify them in an ODS destination statement, as in the following example:

```
ods html body='b.html' style=HTMLBlue;
```

However, you can also modify existing styles and even write your own styles. You can also specify style elements in custom templates that you write, or you can modify which style elements are used in templates supplied by SAS. This section provides an overview of styles and style elements, which are the components of a style. It also describes how to customize a style template and how to specify a default style for your output. Only the most commonly used styles, style elements, and style changes are discussed here. For complete details about styles, see the *SAS Output Delivery System: User's Guide*.

An Overview of Styles

An ODS style template provides formatting information for specific visual aspects of your SAS output (see the section "Style Elements and Attributes" on page 648). The appearance of tables and graphs is coordinated within a particular style. For tables, this information includes a list of fonts and a list of colors. Each font definition specifies a family, size, weight, and style. Colors are associated with common areas of output, including titles, footnotes, BY groups, table headers, and table cells. For graphs, styles also control the appearance of graph elements including lines, markers, fonts and colors. ODS styles also include elements specific to statistical graphics, such as the style of fitted lines, confidence bands, and prediction limits. For more information about styles, see Kuhfeld (2010) and the *SAS Output Delivery System: User's Guide*.

You can specify a style by using the STYLE= option in an ODS destination statement such as HTML, PDF, RTF, or PRINTER. You can also specify a style in the LISTING destination; however, it affects graphs but not tables. Output produced with different styles has the same content, but a different visual appearance. For example, the following statement requests output produced with the JOURNAL style:

```
ods rtf style=Journal;
```

You can use any SAS style or any style that you define yourself. The following statements list the names of all of the styles and then display five of them:

```
proc template;
   list styles;
   source Styles.Default;
   source Styles.Statistical;
   source Styles.Journal;
   source Styles.RTF;
   source Styles.HTMLBlue;
run;
```

The results of this step (not shown) are a list of over fifty styles in the SAS listing and five style templates in the SAS log. Style templates are often hundreds of lines long. See the section "Style Templates and Colors" on page 649 for more information about style templates. Although you can use any style, only a few styles are typically used with ODS Graphics. They are described in Table 21.2.

Table 21.2 Styles

Style	Default in	Description
HTMLBLUE	HTML and SAS/STAT documentation	An all-color style whose dominant colors are shades of blue with sans-serif fonts. See Figure 21.20.
HTMLBLUECML		A color style whose dominant colors are shades of blue with sans-serif fonts. See Figure 21.21.
DEFAULT	HTML	A color style whose dominant colors are gray, blue, and white, with bold sans-serif fonts. See Figure 21.19.
STATISTICAL		A color style whose dominant colors are blue, creamy gray, and white, with sans-serif fonts. See Figure 21.22.
LISTING	LISTING	A color style, similar to DEFAULT but with a white background. See Figure 21.25.
JOURNAL		A black-and-white style with filled areas, with sans-serif fonts. See Figure 21.24.
JOURNAL2		A black-and-white style, similar to JOURNAL but with empty areas. Grouped bar charts use crosshatching to show groups. See Output 21.3.2.
JOURNAL3		A black-and-white style, similar to JOURNAL2 but with a mix of filled areas and crosshatching in grouped bar charts. See Output 21.3.3.
RTF	RTF	A color style whose dominant colors are blue, white, and black, with Times Roman fonts. See Figure 21.26.
ANALYSIS		A color style, similar to STATISTICAL, whose dominant color is tan. See Figure 21.23.

Each ODS destination has its own default style, as shown in Table 21.2. Most output in SAS/STAT documentation uses the HTMLBLUE style. However, throughout this chapter, you can see examples of other styles. For more information about styles, see the *SAS Output Delivery System: User's Guide*.

Style Elements and Attributes

An ODS style template is composed of a set of *style elements*. A style element is a collection of *style attributes* that applies to a particular feature or aspect of the output. A value is specified for each attribute in a style template. For example, **`GraphFit`** is the style element used for fit lines, and its attributes include: **`LineThickness`**, **`LineStyle`**, **`MarkerSize`**, **`MarkerSymbol`**, **`ContrastColor`**, and **`Color`**.

In general, style templates control the overall appearance of ODS tables and graphs. For tables, style templates specify features such as background color, table borders, and color scheme, and they specify the fonts, sizes, and color for the text and values in a table and its headers. For graphs, style templates specify the following features:

- background color
- graph dimensions (height and width)
- borders
- line styles for axes and grid lines
- fonts, sizes, and colors for titles, footnotes, axis labels, axis values, and data labels (see the section "Modifying Graph Fonts in Styles" on page 682 for an illustration)
- marker symbols, colors, and sizes for data points and outliers
- line styles for needles
- line and curve styles for fitted models and predicted values (see the section "Modifying Other Graph Elements in Styles" on page 685 for an illustration)
- line and curve styles for confidence and prediction limits
- fill colors for histogram bars, confidence bands, and confidence ellipses
- colors for box plot features
- colors for surfaces
- color ramps for contour plots

The SAS System supplies a graph template for each graph that is created by statistical procedures. A graph template is a program that specifies the layout and details of a graph. See the section "Graph Templates" on page 711 in Chapter 22, "ODS Graphics Template Modification," for more information about templates. Some template options are specified with a style reference of the form **`style-element`**, or occasionally

style-element:attribute. For example, the symbol, color, and size of markers for basic scatter plots are specified in a template SCATTERPLOT statement as follows:

```
scatterplot x=x y=y / markerattrs=GraphDataDefault;
```

The preceding statement specifies that the appearance for markers is controlled by the **GraphDataDefault** element. Consistent use of this element guarantees a common appearance of markers across all scatter plots, based on the style template that you are using.

In general, ODS Graphics features are determined by style element attributes unless they are overridden by a statement or option in the graph template. For example, suppose that a classification variable is specified with the GROUP= option in a SCATTERPLOT template statement as follows:

```
scatterplot x=X y=Y / group=GroupVar;
```

Then the colors for markers that correspond to the classification levels are assigned by using the style element attributes **GraphData1:ContrastColor** through **GraphData12:ContrastColor**.

Style templates are created and modified with PROC TEMPLATE. For more information, see the *SAS Output Delivery System: User's Guide*. You need to understand the relationships between style elements and graph features if you want to create your own style template or modify a style template. These relationships are explained in the following sections.

Style Templates and Colors

The default style templates that the SAS System provides are stored in the *Styles* directory of Sashelp.Tmplmst. You can display, edit, and save style templates by using the same methods available for modifying graph and table templates, as explained in the section "The Default Template Stores and the Template Search Path" on page 645 and the series of sections beginning with the section "Displaying Templates" on page 717 in Chapter 22, "ODS Graphics Template Modification." In particular, you can display a style template by using one of these methods:

- From the Templates window in the SAS windowing environment, expand the Sashelp.Tmplmst node under **Templates**, and then select **Styles** to display the contents of this folder. To open the Templates window, type **odst** on the command line.
- Use the SOURCE statement in PROC TEMPLATE.

For example, the following statements display the DEFAULT style template in the SAS log:

```
proc template;
   source Styles.Default;
run;
```

Some of the results are as follows:

```
define style Styles.Default;
   . . .
   class GraphColors
      "Abstract colors used in graph styles" /
      . . .
      'gconramp3cend' = cxFF0000
      'gconramp3cneutral' = cxFF00FF
      'gconramp3cstart' = cx0000FF
      . . .
      'gdata12' = cxDDD17E
      'gdata11' = cxB7AEF1
      'gdata10' = cx87C873
      'gdata9' = cxCF974B
      'gdata8' = cxCD7BA1
      'gdata6' = cxBABC5C
      'gdata7' = cx94BDE1
      'gdata4' = cxA9865B
      'gdata5' = cxB689CD
      'gdata3' = cx66A5A0
      'gdata2' = cxDE7E6F
      'gdata1' = cx7C95CA;
   . . .
```

The first part of this list shows that the shading for certain filled plots, such as some contour plots goes from blue (**'gconramp3cstart' = cx0000FF**) to magenta (**'gconramp3cneutral' = cxFF00FF**) to red (**'gconramp3cend' = cxFF0000**). All colors are specified in values of the form CX*rrggbb*, where the last six characters specify RGB (red, green, blue) values on the hexadecimal scale of 00 to FF (or 0 to 255 base 10). The second part of the list (**'gdata1' = cx7C95CA**) shows that the dominant component of the **GraphData1** color is blue because the blue component of the color (CA, which corresponds to 202 base 10) is greater than both the green component (95, which corresponds to 149 base 10) and the red component (7C, which corresponds to 124 base 10).

You can change any part of the style and then submit the style back into the SAS System, after first submitting a PROC TEMPLATE statement. See the sections "Saving Customized Templates" on page 720, "Using Customized Templates" on page 721, and "Reverting to the Default Templates" on page 722 in Chapter 22, "ODS Graphics Template Modification," for more information about modifying, using, and restoring templates. The principles discussed in those sections apply to all templates—table, style, and graph.

Some Common Style Elements

This section explains some common style elements and produces most of the graphs displayed in the section "Style Comparisons" on page 656.

The DEFAULT style is the parent for the styles used for statistical graphics work. You can see all of the elements of the DEFAULT style by running the following step:

```
proc template;
   source styles.default;
run;
```

The source listing of the definition of the DEFAULT style is hundreds of lines long. If you run PROC TEMPLATE with the SOURCE statement for most other styles, you see **`parent = styles.default`** (or in the case of the HTMLBLUE style, you see **`parent = styles.statistical`**, which inherits from the DEFAULT style), and you do not see all of the elements in the style unless you also run the preceding step with a SOURCE statement for all parent styles.

Only a few of the style elements are referenced in the templates that the SAS System provides for statistical procedures. The most commonly used style elements, along with the defaults for the noncolor attributes of the DEFAULT style, are shown next (**`Color`** applies to filled areas, and **`ContrastColor`** applies to markers and lines):

`Graph`	graph size, outer border appearance, and background color
	`Padding = 0`
	`BackgroundColor`
`GraphConfidence`	primary fit confidence interval
	`LineThickness = 1px`
	`LineStyle = 1`
	`MarkerSize = 7px`
	`MarkerSymbol = "triangle"`
	`ContrastColor`
	`Color`
`GraphData1`	attributes related to first grouped data items
	`MarkerSymbol = "circle"`
	`LineStyle = 1`
	`ContrastColor`
	`Color`
`GraphData2`	attributes related to second grouped data items
	`MarkerSymbol = "plus"`
	`LineStyle = 4`
	`ContrastColor`
	`Color`
`GraphData3`	attributes related to third grouped data items
	`MarkerSymbol = "X"`
	`LineStyle = 8`
	`ContrastColor`
	`Color`
`GraphData4`	attributes related to fourth grouped data items
	`MarkerSymbol = "triangle"`
	`LineStyle = 5`
	`ContrastColor`
	`Color`
`GraphData`*n*	attributes related to *n*th grouped data items
	`MarkerSymbol`
	`LineStyle`
	`ContrastColor`
	`Color`

GraphDataDefault	attributes related to data items that are not grouped
	EndColor
	NeutralColor
	StartColor
	MarkerSize = 7px
	MarkerSymbol = "circle"
	LineThickness = 1px
	LineStyle = 1
	ContrastColor
	Color
GraphFit	primary fit line, such as a normal density curve
	LineThickness = 2px
	LineStyle = 1
	MarkerSize = 7px
	MarkerSymbol = "circle"
	ContrastColor
	Color
GraphFit2	secondary fit line, such as a kernel density curve
	LineThickness = 2px
	LineStyle = 4
	MarkerSize = 7px
	MarkerSymbol = "X"
	ContrastColor
	Color
GraphGridLines	horizontal and vertical grid lines drawn at major tick marks
	Displayopts = "auto"
	LineThickness = 1px
	LineStyle = 1
	ContrastColor
	Color
GraphOutlier	outlier data for the graph
	LineThickness = 2px
	LineStyle = 42
	MarkerSize = 7px
	MarkerSymbol = "circle"
	ContrastColor
	Color
GraphPredictionLimits	fills for prediction limits
	LineThickness = 1px
	LineStyle = 2
	MarkerSize = 7px
	MarkerSymbol = "chain"
	ContrastColor
	Color

GraphReference	horizontal and vertical reference lines and drop lines **LineThickness = 1px** **LineStyle = 1** **ContrastColor**
GraphDataText	text font and color for point and line labels **Font = GraphFonts('GraphDataFont')** (where **'GraphDataFont' = ("<sans-serif>, <MTsans-serif>",7pt)**) **Color**
GraphValueText	text font and color for axis tick values and legend values **Font = GraphFonts('GraphValueFont')** (where **'GraphValueFont' = ("<sans-serif>, <MTsans-serif>",9pt)**) **Color**
GraphLabelText	text font and color for axis labels and legend title **Font = GraphFonts('GraphLabelFont')** (where **'GraphLabelFont' = ("<sans-serif>, <MTsans-serif>",10pt,bold)**) **Color**
GraphFootnoteText	text font and color for footnotes **Font = GraphFonts('GraphFootnoteFont')** (where **'GraphFootnoteFont' = ("<sans-serif>, <MTsans-serif>",10pt)**) **Color**
GraphTitleText	text font and color for titles **Font = GraphFonts('GraphTitleFont')** (where **'GraphTitleFont' = ("<sans-serif>, <MTsans-serif>",11pt,bold)**) **Color**
GraphWalls	vertical walls bounded by axes **LineThickness = 1px** **LineStyle = 1** **FrameBorder = on** **ContrastColor** **BackgroundColor** **Color**

You refer to these elements in graph templates as **style-element** or as **style-element:attribute** (for example **GraphDataDefault:ContrastColor**). The default values are not shown for the color attributes since they are typically defined indirectly. For example, **Graph:BackgroundColor** (the color that fills the box outside the graph) is defined elsewhere in the style as **colors('docbg')**. The style also defines **'docbg' = color_list('bgA')** and **'bgA' = cxE0E0E0**. This shows that the background is a shade of gray that is much closer to white (CXFFFFFF) than to black (CX000000). You can see the background color in Figure 21.27. This shade of gray might seem darker (closer to CX000000) than you might expect based on just the RGB values. Your perception of a color change is not a linear function of the change in RGB values.

You can use the following program to see the color and other attributes for a number of style elements:

```
proc format; value vf 5 = 'GraphValueText'; run;

data x1;
   array y[20] y0 - y19;
   do x = 1 to 20; y[x] = x - 0.5; end;
   do x = 0 to 10 by 5; output; end;
   label y0 = 'GraphLabelText' x = 'GraphLabelText';
   format x y0 vf.;
run;

%macro d;
   %do i = 1 %to 12;
      reg y=y%eval(19-&i) x=x / lineattrs=GraphData&i markerattrs=GraphData&i
                                curvelabel="  GraphData&i" curvelabelpos=max;
   %end;
%mend;

%macro l(i, l);
   reg y=y&i x=x / lineattrs=&l markerattrs=&l curvelabel="  &l"
                   curvelabelpos=max;
%mend;

ods listing style=default;

proc sgplot noautolegend data=x1;
   title 'GraphTitleText';
   %d
   %l(19, GraphDataDefault)
   %l( 6, GraphFit)
   %l( 5, GraphFit2)
   %l( 4, GraphPredictionLimits)
   %l( 3, GraphConfidence)
   %l( 2, GraphGridLines)
   %l( 1, GraphOutlier)
   %l( 0, GraphReference)
   xaxis values=(0 5 10);
run;
```

The results in Figure 21.27 display the attributes for a number of the elements of the DEFAULT style.

When there is a group or classification variable, the colors, markers, and lines that distinguish the groups are derived from the **GraphData***n* elements that are defined with the style. In the DEFAULT style, these are elements **GraphData1** through **GraphData12**. There can be any number of groups even though only 12 **GraphData***n* style elements are defined in the DEFAULT style. The following steps create a data set with 40 groups, display one line per group, and produce Figure 21.35:

```
data x2;
   do y = 40 to 1 by -1;
      group = 'Group' || put(41 - y, 2. -L);
      do x = 0 to 10 by 5;
         if x = 10 then do; z = 11; l = group; end;
         else           do; z = .;  l = ' ';   end;
         output;
      end;
   end;
run;

proc sgplot data=x2;
   title 'Colors, Markers, Lines Patterns for Groups';
   series  y=y x=x / group=group markers;
   scatter y=y x=z / group=group markerchar=l;
run;
```

The colors, markers, and line patterns in Figure 21.35 repeat in cycles. The **GraphData1** – **GraphData8** lines in Figure 21.27 exactly match the **Group1** – **Group8** lines in Figure 21.35. After that, there are differences due to the cyclic construction of the grouped style. This is explained next.

The DEFAULT style defines a marker symbol only in **GraphData1** through **GraphData7**. The seven markers are: circle, plus, X, triangle, square, asterisk, and diamond. With the explicit style reference in Figure 21.27, the actual symbol, when no symbol is specified, is the circle. This is what you see for **GraphData8** through **GraphData12**. With the group variable in Figure 21.35, the symbols repeat in cycles. Hence, **Group1**, **Group8**, **Group15**, and so on, are all circles. Similarly, **Group2**, **Group9**, **Group16**, and so on, are all pluses. The DEFAULT style defines 11 different line styles for **GraphData1** through **GraphData11**. You specify line styles by specifying an integer. The default lines styles are: 1, 4, 8, 5, 14, 26, 15, 20, 41, 42, and 2. Hence, **Group1**, **Group12**, **Group23**, and so on, all have the same line style, which is a solid line (line style 1). Similarly, **Group2**, **Group13**, **Group24**, and so on, all have line style 4. There are twelve different colors, so **Group1**, **Group13**, **Group25**, and so on, all have the same colors. Overall, there are $12 \times 11 \times 7 = 924$ color/line/marker combinations that appear before any combination repeats. You can use the %MODSTYLE SAS autocall macro (see the sections "Creating an All-Color Style" on page 676 and "Style Template Modification Macro" on page 674) to conveniently change these style attributes.

The HTMLBLUE style is an all-color style for the first 12 groups of observations. Most analyses have fewer than 12 groups. Markers and lines change for groups 13–24 and then again for groups 25–36. Figure 21.36 shows how colors, markers, and line styles change in the HTMLBLUE style. Figure 21.35 and Figure 21.36 through Figure 21.42 show how these elements change in other styles.

Style Comparisons

In this section, some of the most commonly used styles are compared with a series of figures, most of which were generated in the preceding section. Figure 21.19 through Figure 21.26 show tables and graphs in each of eight styles, for the following analysis:

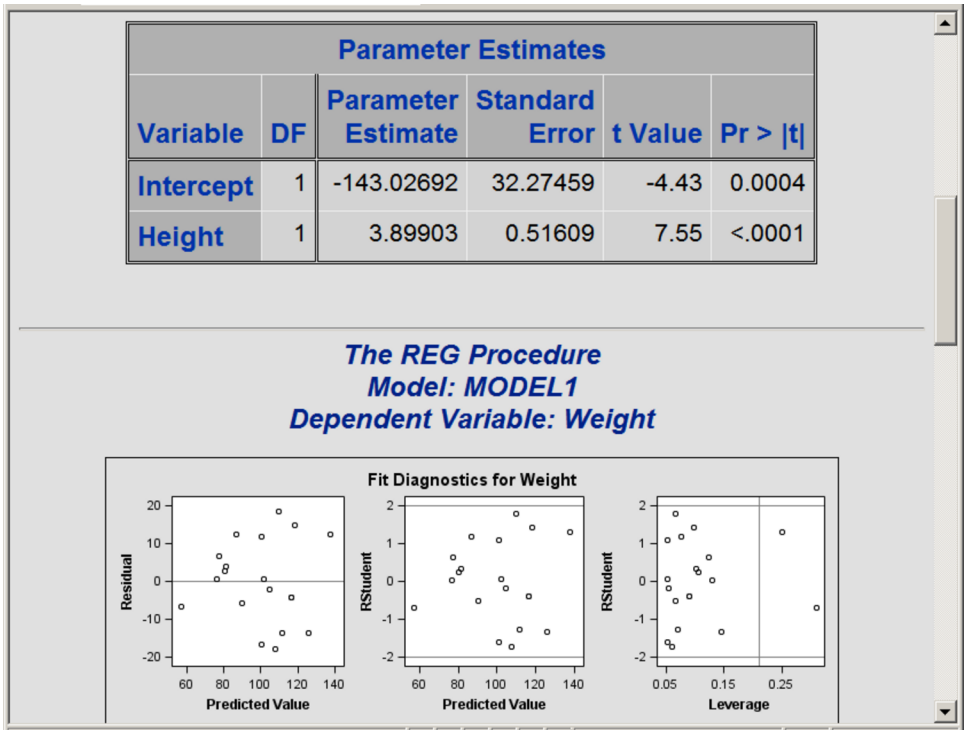
```
proc reg data=sashelp.class;
   model Weight = Height;
run; quit;
```

Figure 21.27 through Figure 21.34 show some of the more common style elements. Figure 21.35 through Figure 21.42 show how groups of observations are displayed in the graph.

The style comparisons are as follows:

- Figure 21.19, Figure 21.27, and Figure 21.35 show the DEFAULT style.
- Figure 21.20, Figure 21.28, and Figure 21.36 show the HTMLBLUE style.
- Figure 21.21, Figure 21.29, and Figure 21.37 show the HTMLBLUECML style.
- Figure 21.22, Figure 21.30, and Figure 21.38 show the STATISTICAL style.
- Figure 21.23, Figure 21.31, and Figure 21.39 show the ANALYSIS style.
- Figure 21.24, Figure 21.32, and Figure 21.40 show the JOURNAL style.
- Figure 21.25, Figure 21.33, and Figure 21.41 show the LISTING style.
- Figure 21.26, Figure 21.34, and Figure 21.42 show the RTF style.

Figure 21.19 Statistical Output with the DEFAULT Style

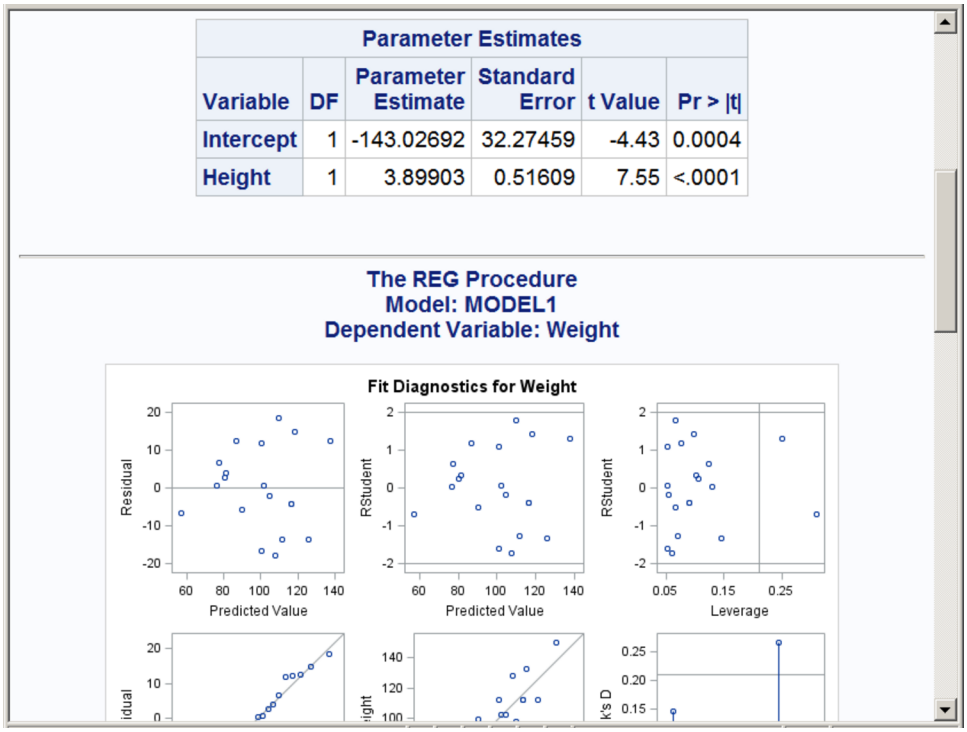


Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-143.02692	32.27459	-4.43	0.0004
Height	1	3.89903	0.51609	7.55	<.0001

The REG Procedure
Model: MODEL1
Dependent Variable: Weight

Figure 21.20 Statistical Output with the HTMLBLUE Style

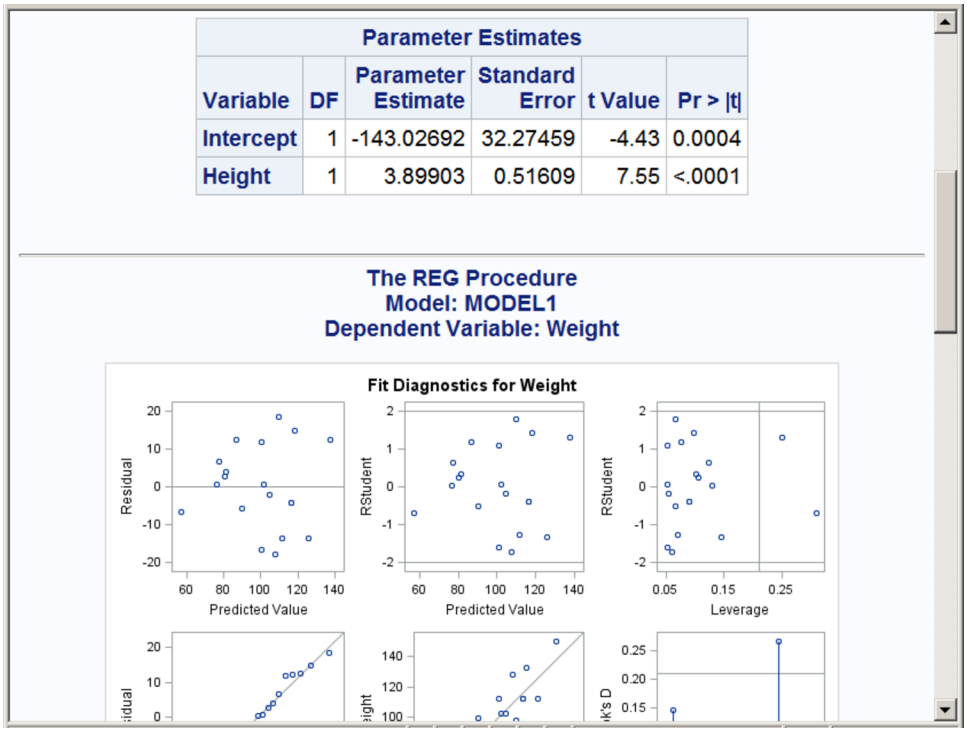


Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-143.02692	32.27459	-4.43	0.0004
Height	1	3.89903	0.51609	7.55	<.0001

The REG Procedure
Model: MODEL1
Dependent Variable: Weight

Figure 21.21 Statistical Output with the HTMLBLUECML Style

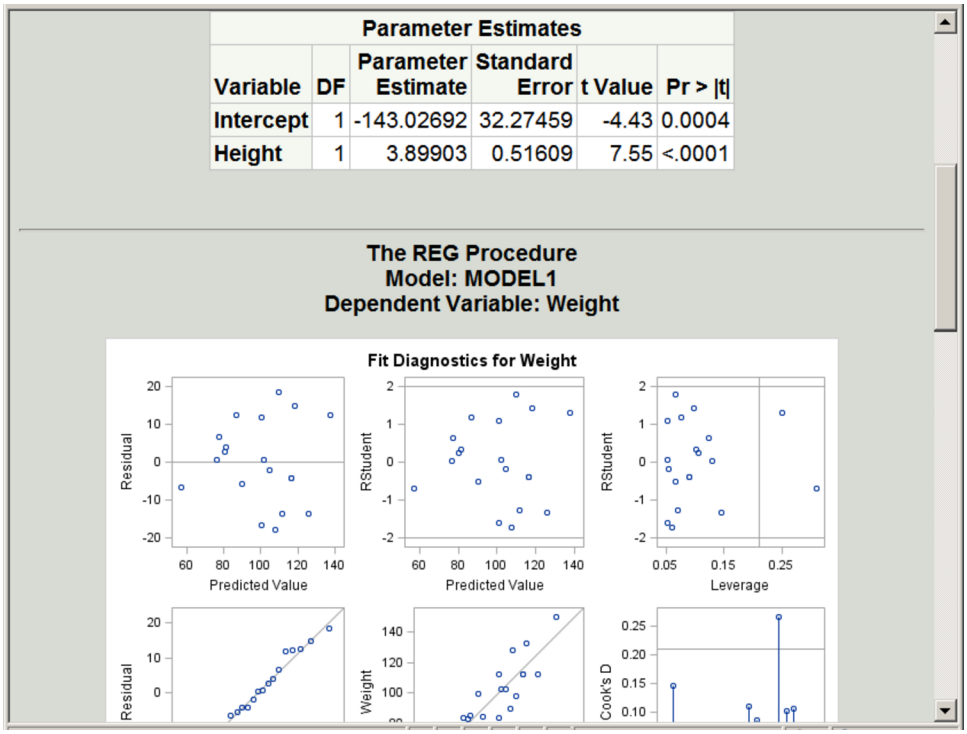


Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-143.02692	32.27459	-4.43	0.0004
Height	1	3.89903	0.51609	7.55	<.0001

The REG Procedure
Model: MODEL1
Dependent Variable: Weight

Figure 21.22 Statistical Output with the STATISTICAL Style

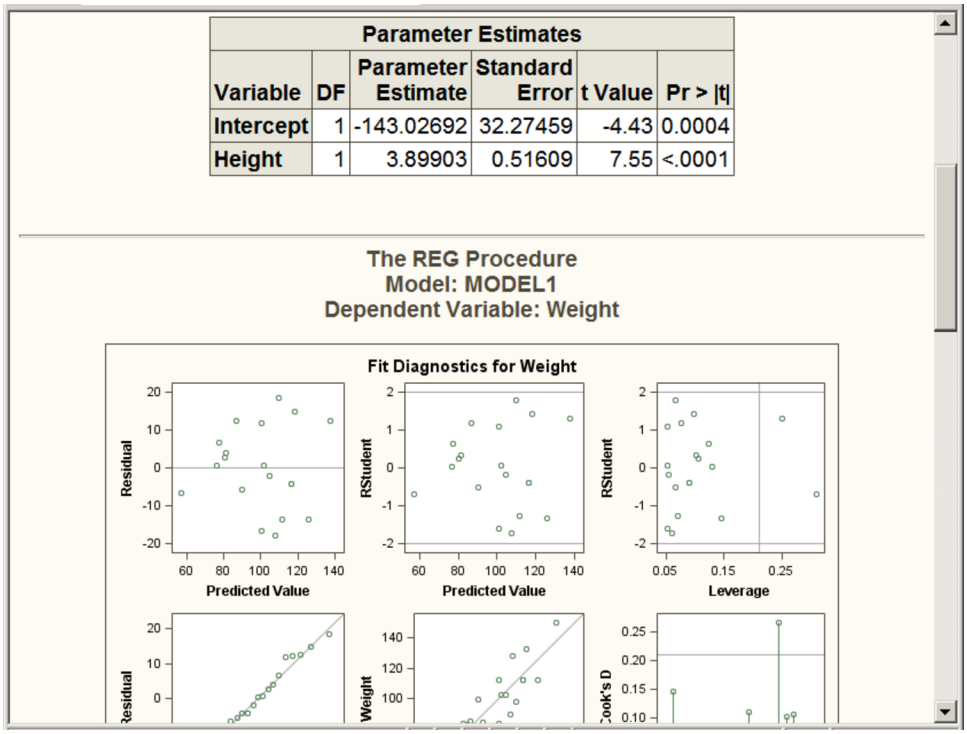


Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-143.02692	32.27459	-4.43	0.0004
Height	1	3.89903	0.51609	7.55	<.0001

The REG Procedure
Model: MODEL1
Dependent Variable: Weight

Figure 21.23 Statistical Output with the ANALYSIS Style

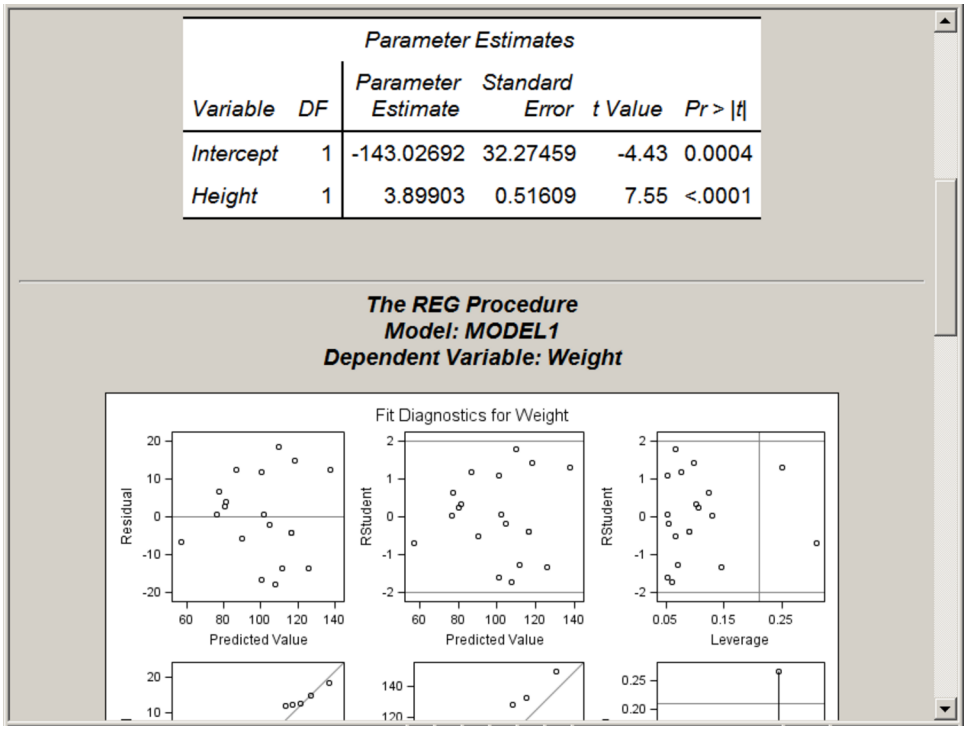


Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-143.02692	32.27459	-4.43	0.0004
Height	1	3.89903	0.51609	7.55	<.0001

The REG Procedure
Model: MODEL1
Dependent Variable: Weight

Figure 21.24 Statistical Output with the JOURNAL Style

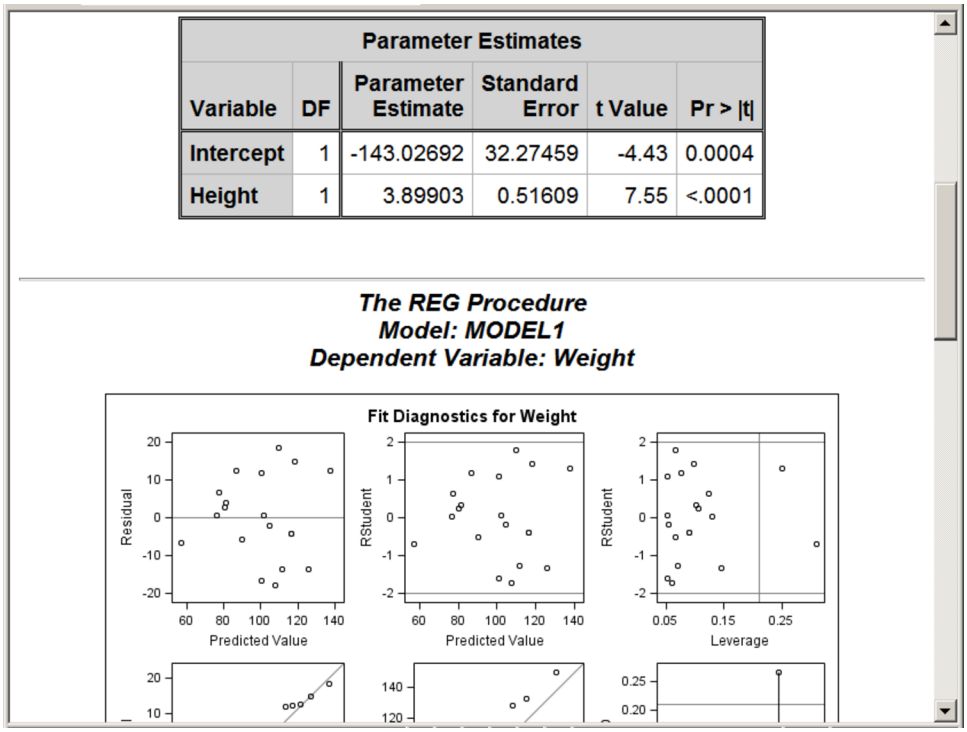


Parameter Estimates

Variable	*DF*	*Parameter Estimate*	*Standard Error*	*t Value*	*Pr > |t|*
Intercept	1	-143.02692	32.27459	-4.43	0.0004
Height	1	3.89903	0.51609	7.55	<.0001

The REG Procedure
Model: MODEL1
Dependent Variable: Weight

Figure 21.25 Statistical Output with the LISTING Style

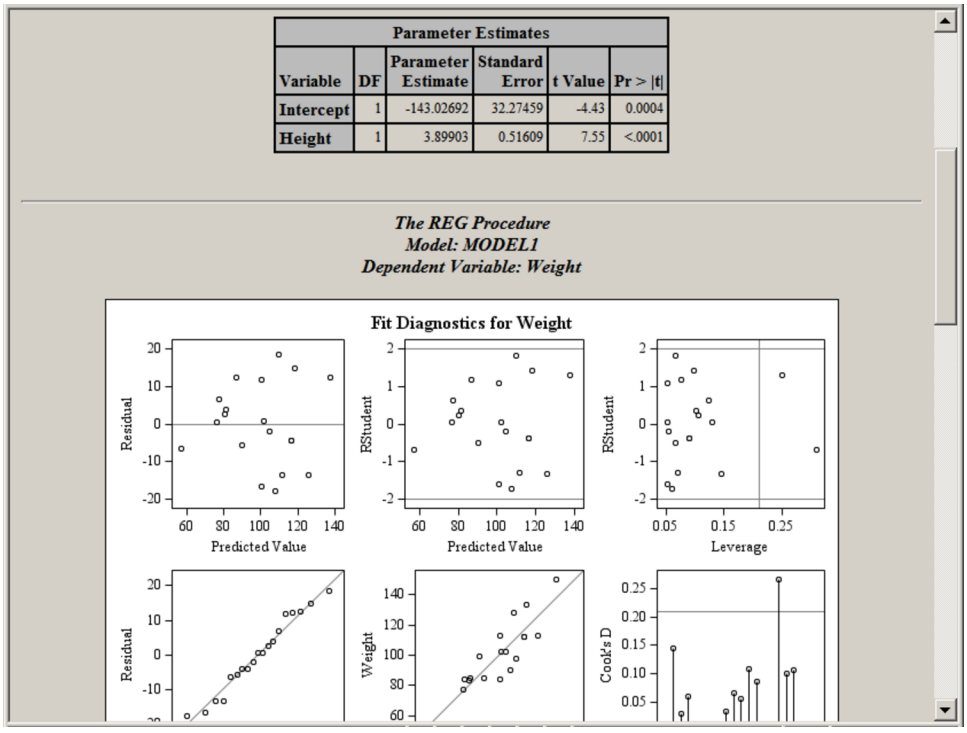


Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-143.02692	32.27459	-4.43	0.0004
Height	1	3.89903	0.51609	7.55	<.0001

The REG Procedure
Model: MODEL1
Dependent Variable: Weight

Figure 21.26 Statistical Output with the RTF Style



Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-143.02692	32.27459	-4.43	0.0004
Height	1	3.89903	0.51609	7.55	<.0001

The REG Procedure
Model: MODEL1
Dependent Variable: Weight

Figure 21.27 Attributes of Style Elements in the DEFAULT Style

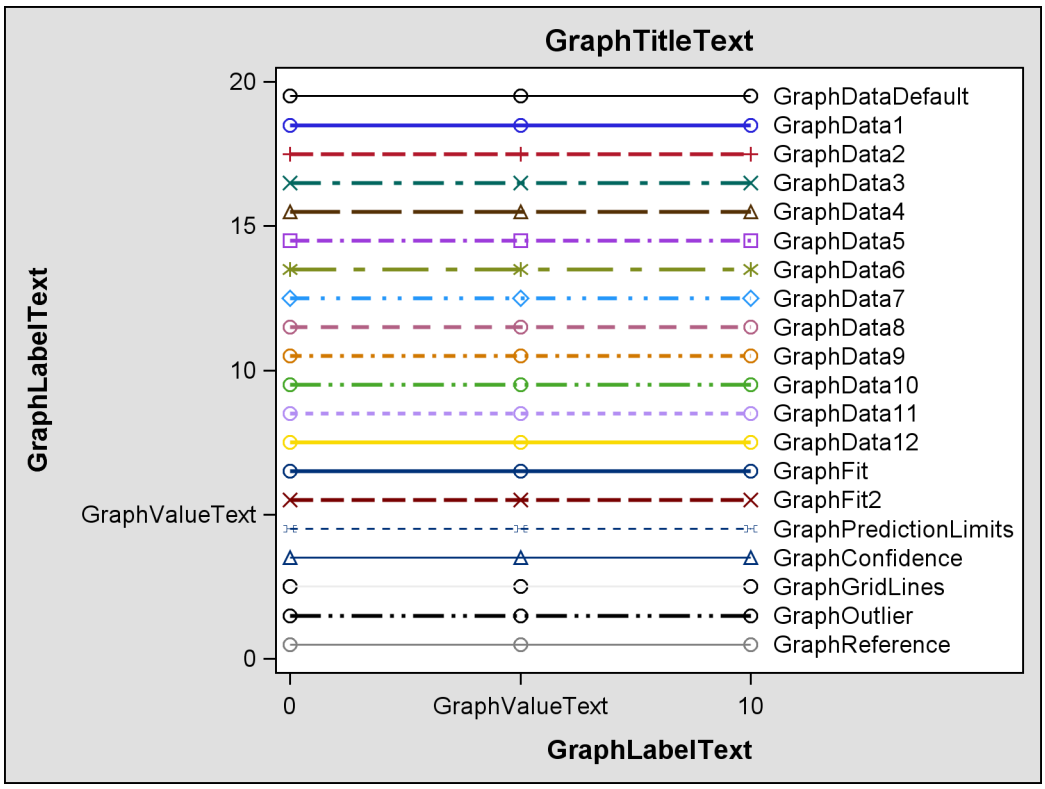


Figure 21.28 Attributes of Style Elements in the HTMLBLUE Style

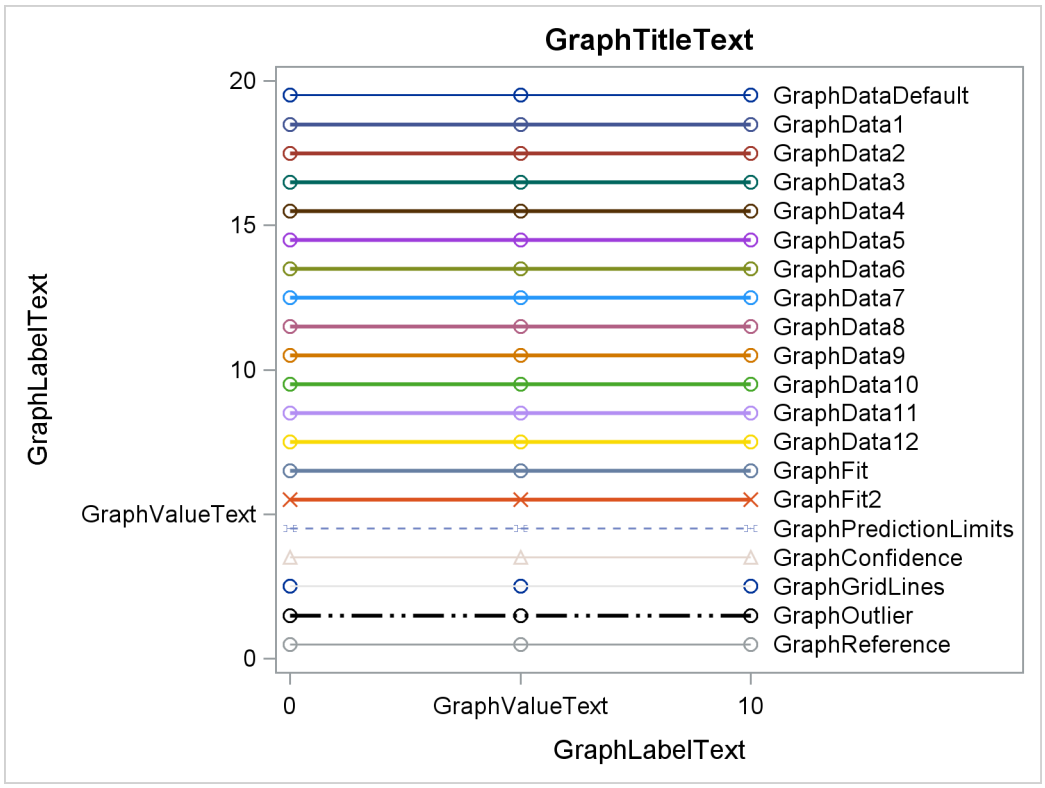


Figure 21.29 Attributes of Style Elements in the HTMLBLUECML Style

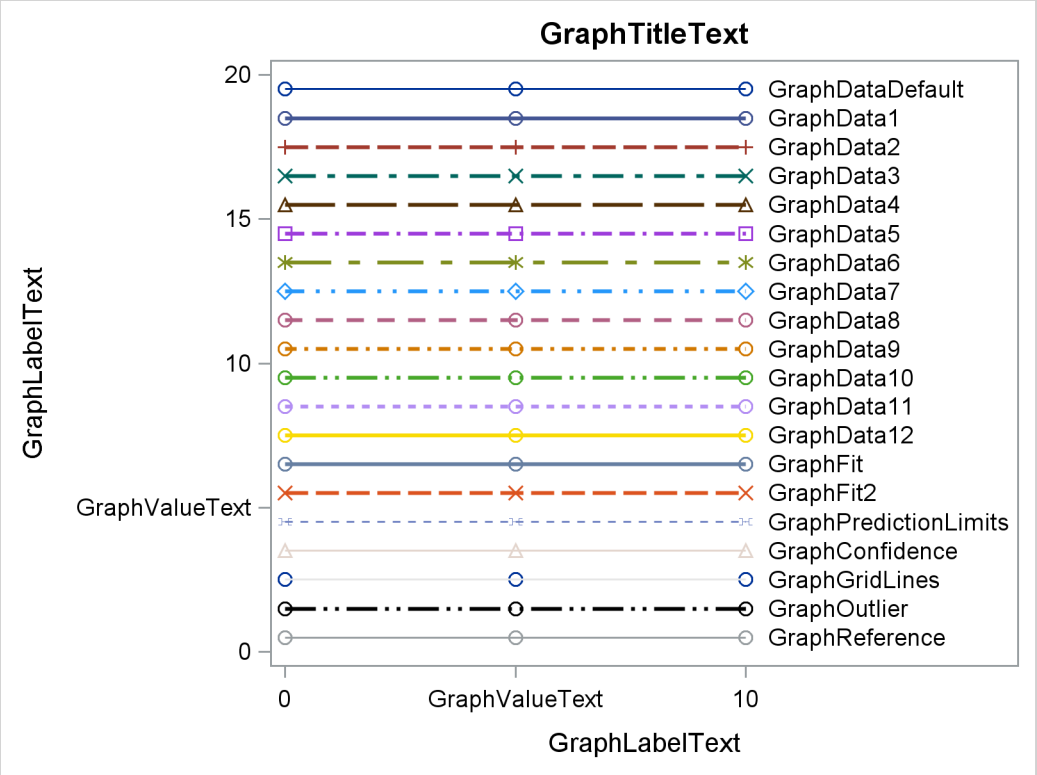


Figure 21.30 Attributes of Style Elements in the STATISTICAL Style

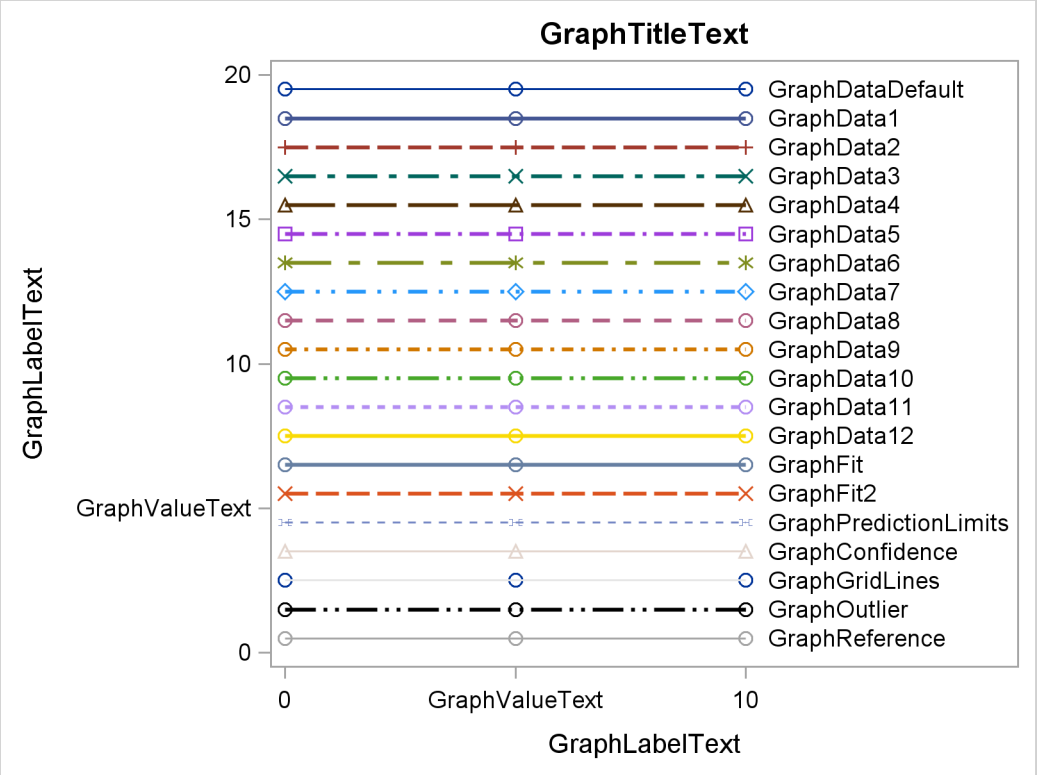


Figure 21.31 Attributes of Style Elements in the ANALYSIS Style

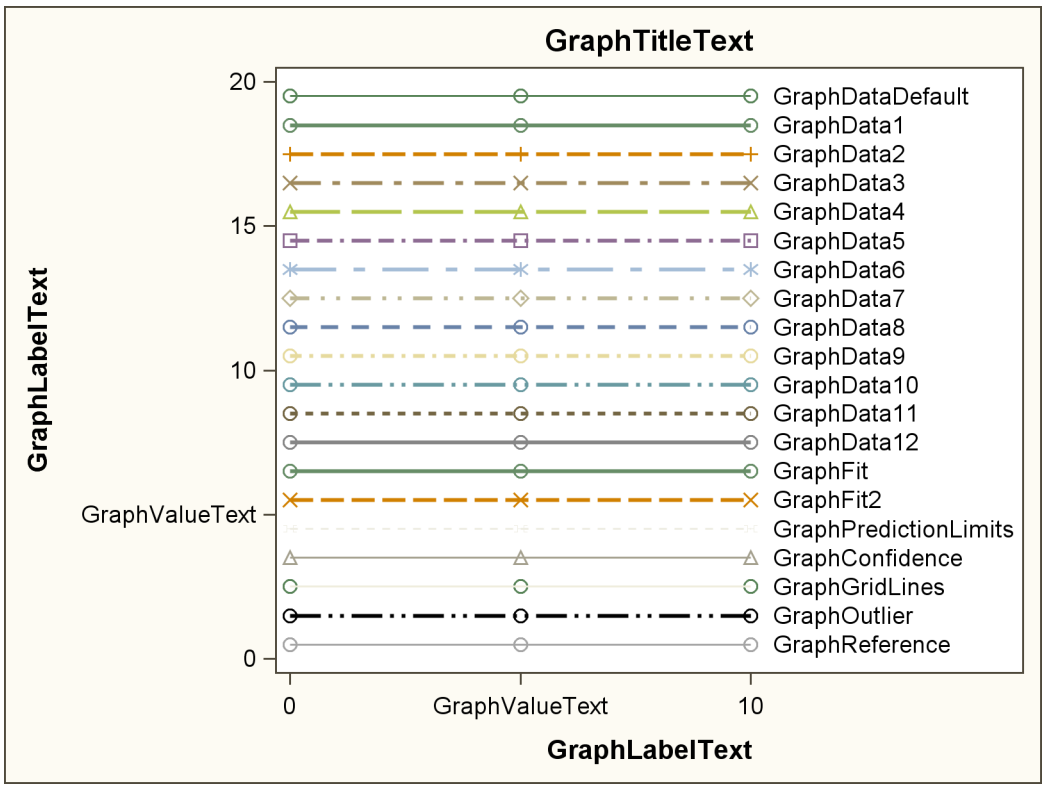


Figure 21.32 Attributes of Style Elements in the JOURNAL Style

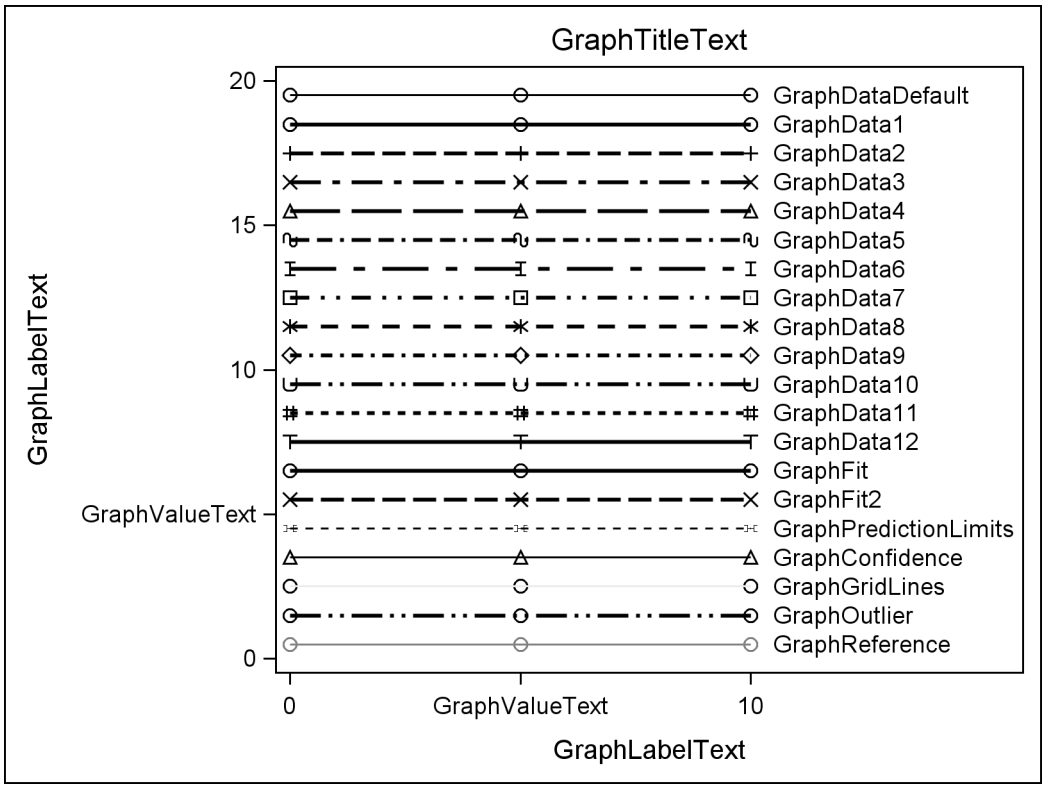


Figure 21.33 Attributes of Style Elements in the LISTING Style

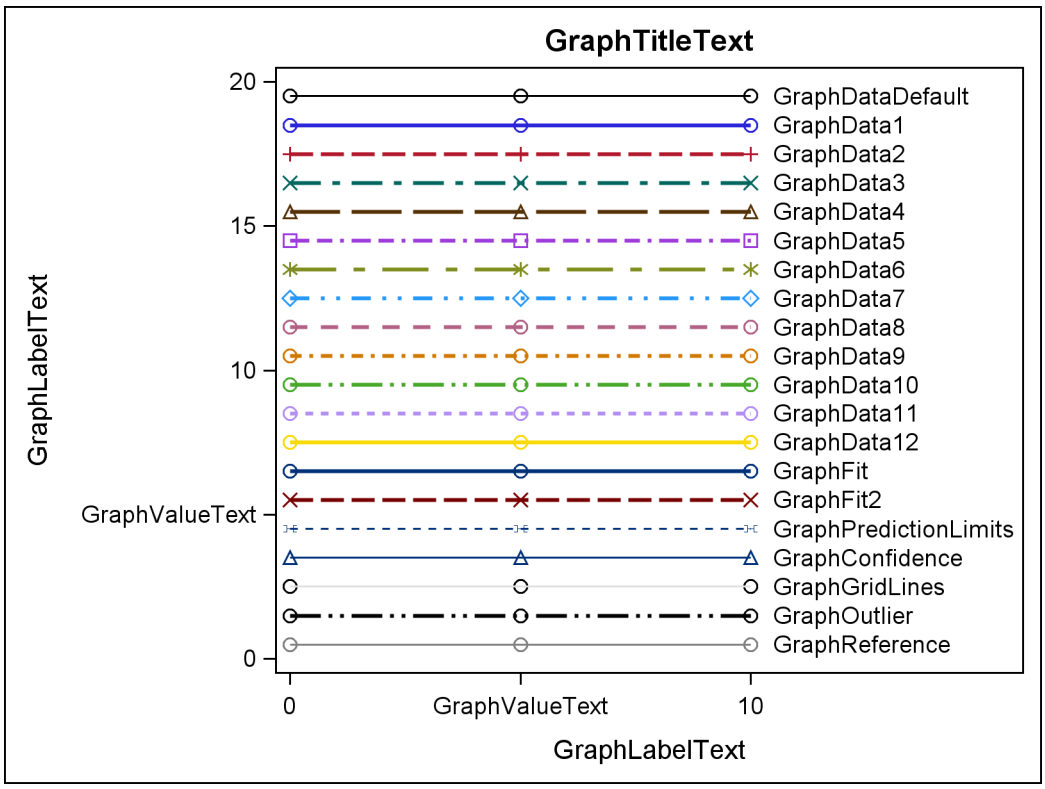


Figure 21.34 Attributes of Style Elements in the RTF Style

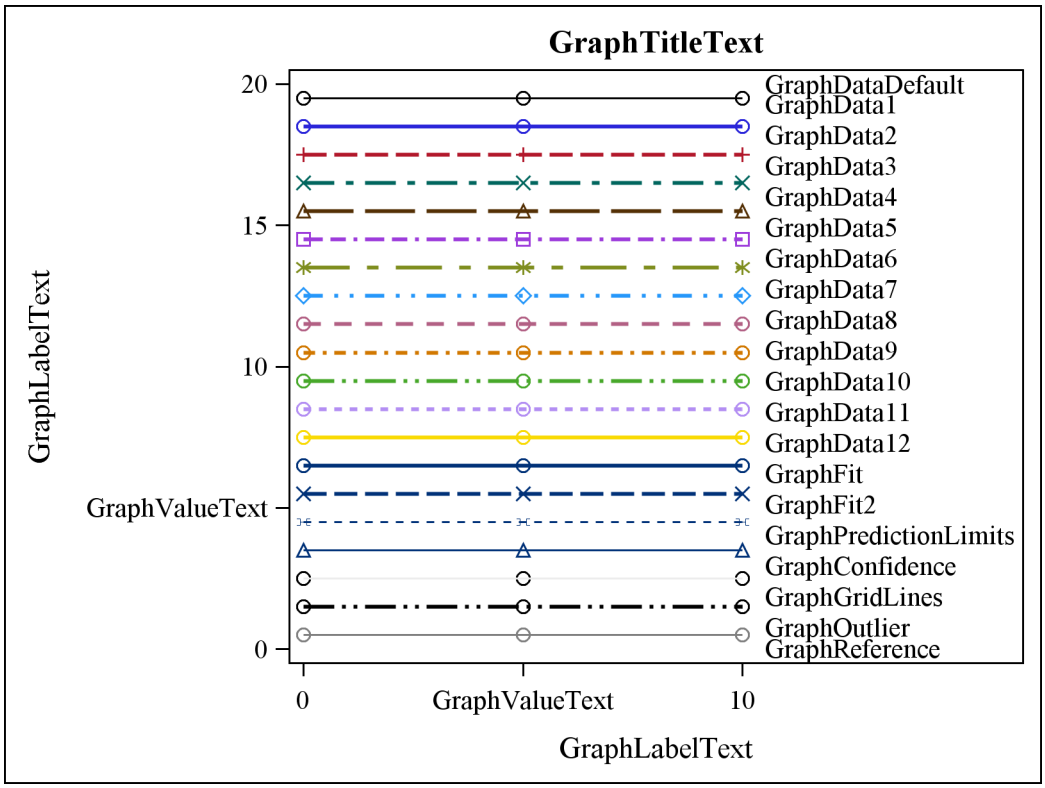


Figure 21.35 Markers, Lines, and Colors with Groups in the DEFAULT Style

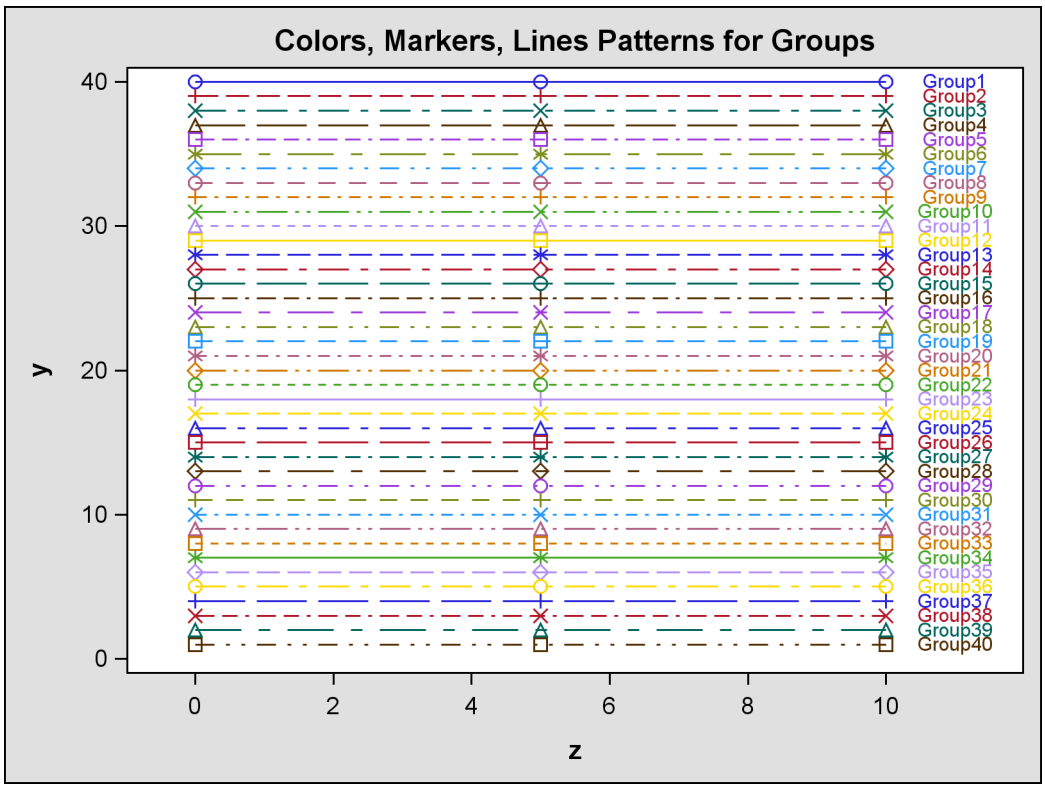


Figure 21.36 Markers, Lines, and Colors with Groups in the HTMLBLUE Style

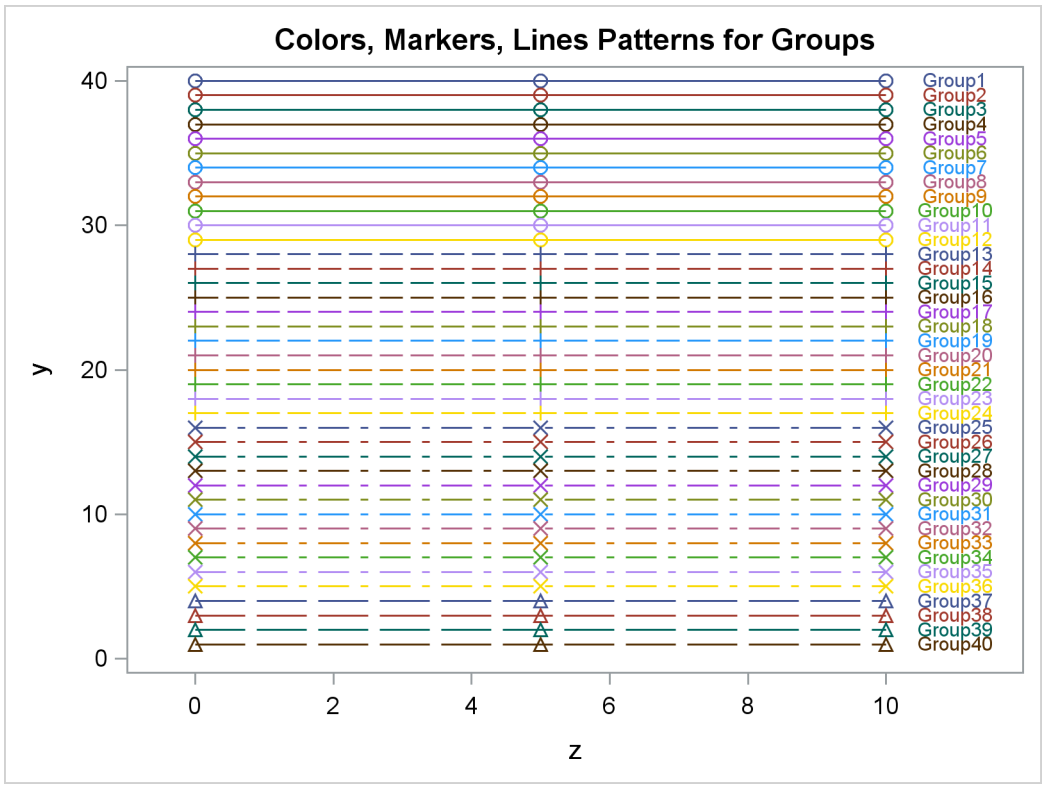


Figure 21.37 Markers, Lines, and Colors with Groups in the HTMLBLUECML Style

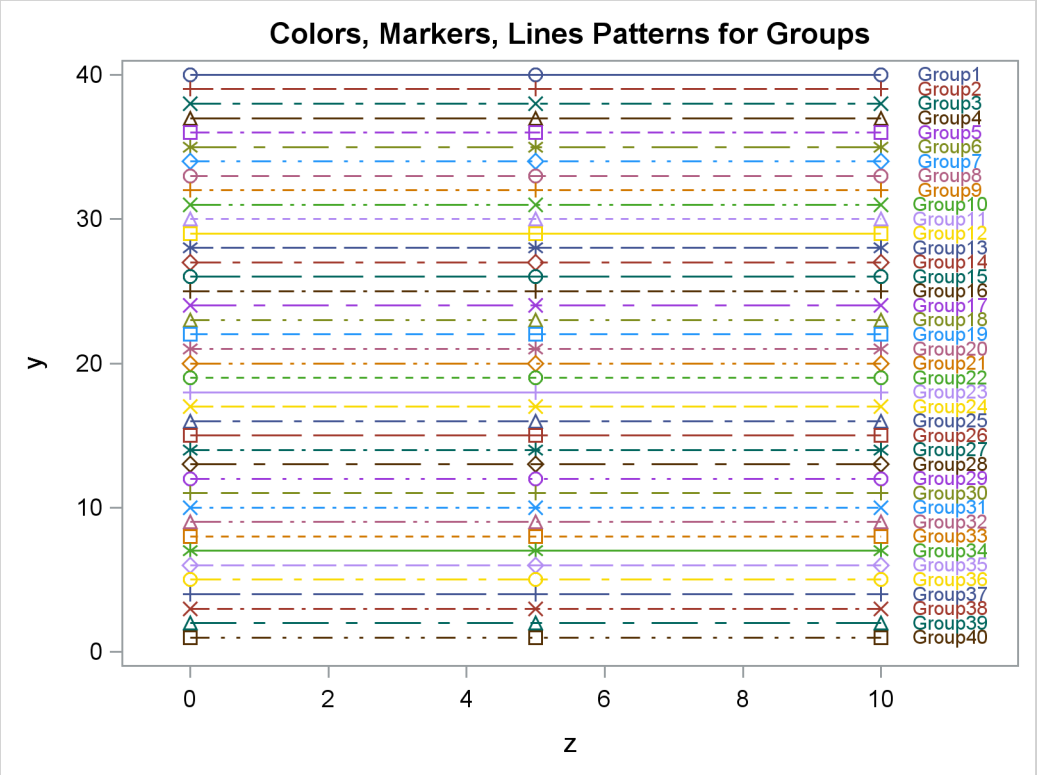


Figure 21.38 Markers, Lines, and Colors with Groups in the STATISTICAL Style

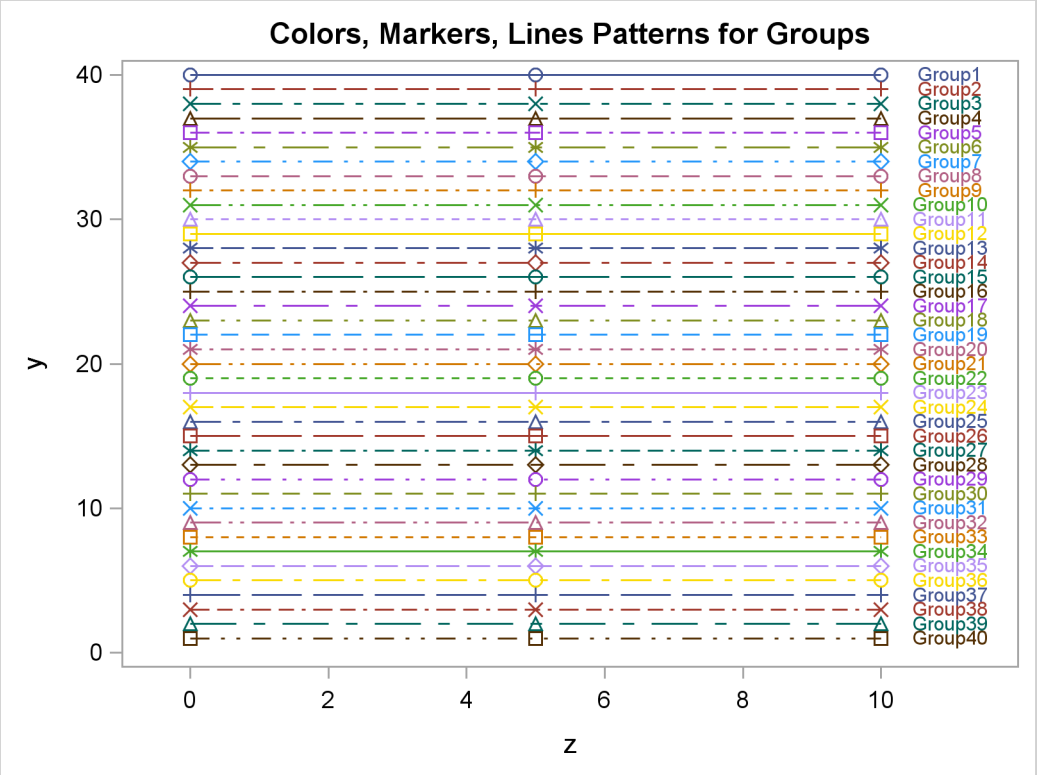


Figure 21.39 Markers, Lines, and Colors with Groups in the ANALYSIS Style

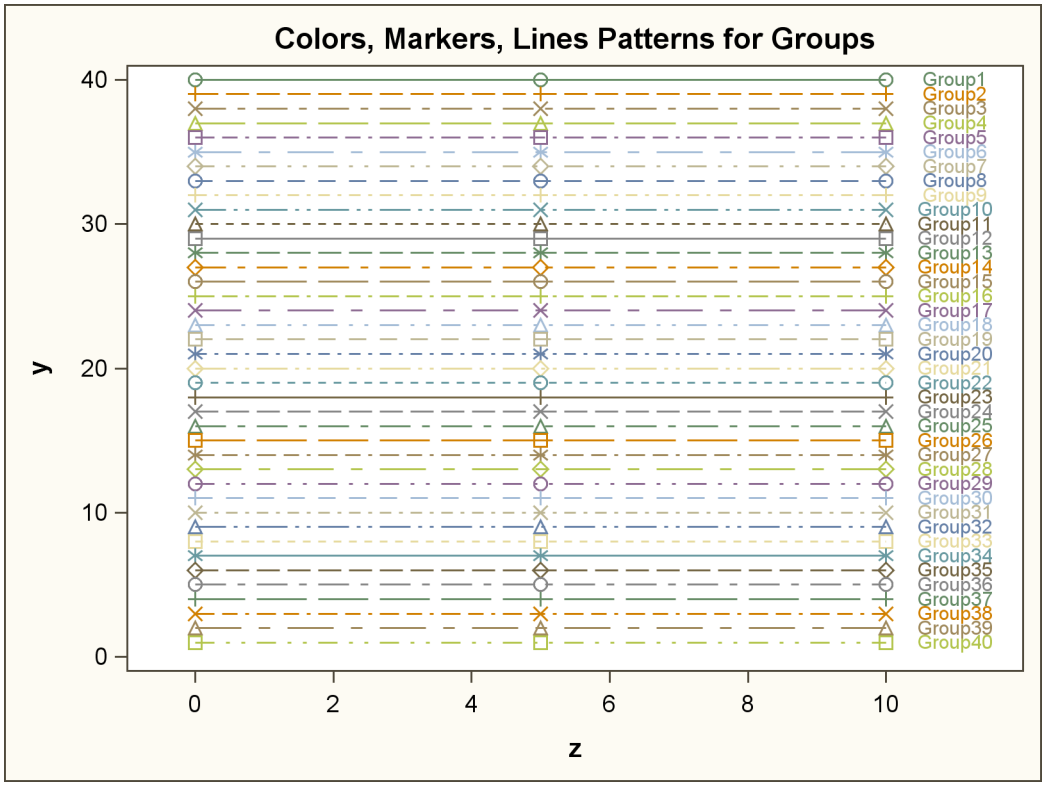


Figure 21.40 Markers, Lines, and Colors with Groups in the JOURNAL Style

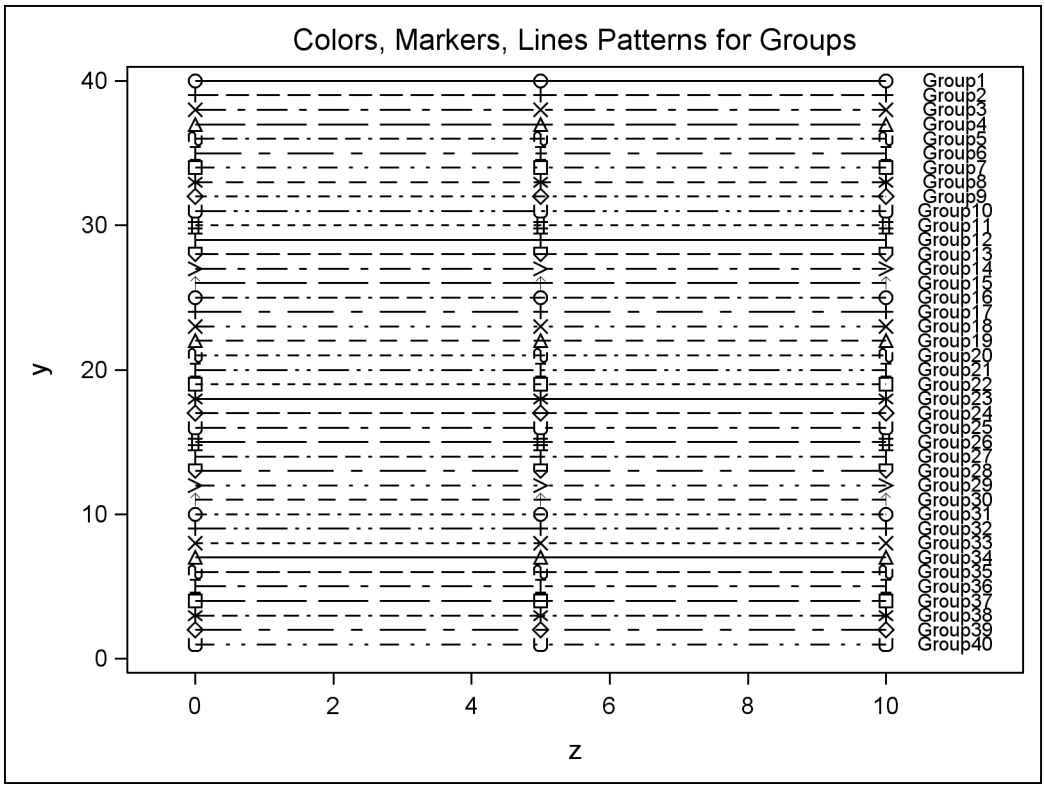


Figure 21.41 Markers, Lines, and Colors with Groups in the LISTING Style

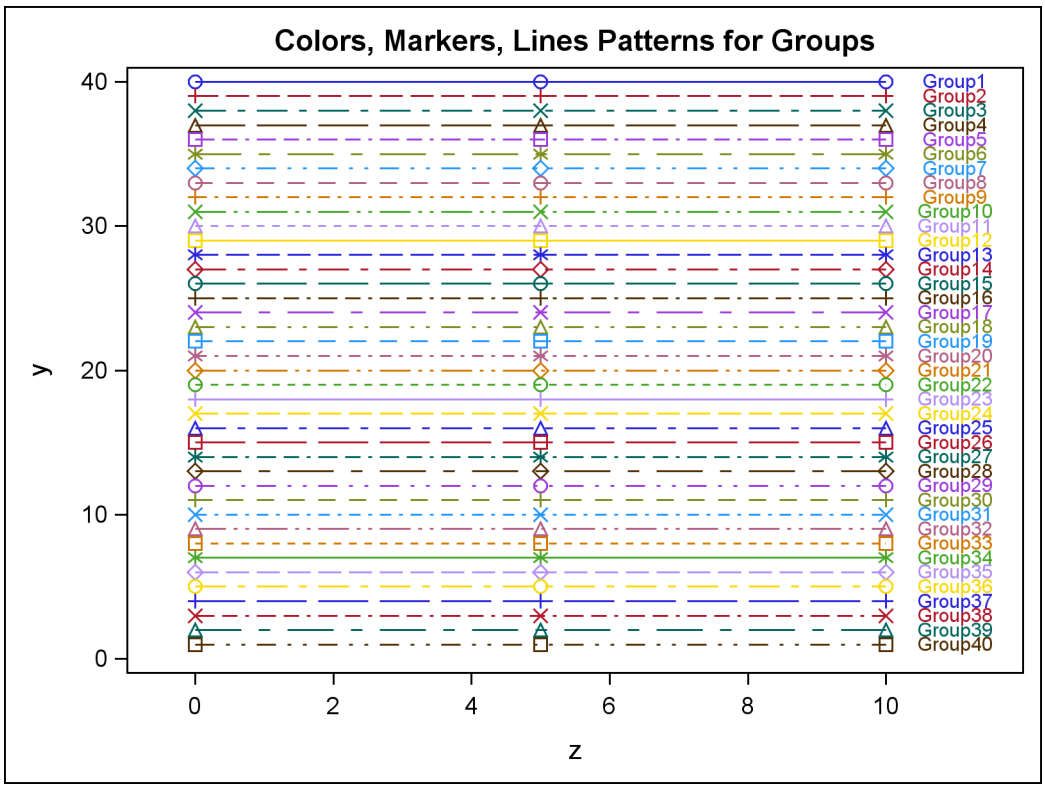
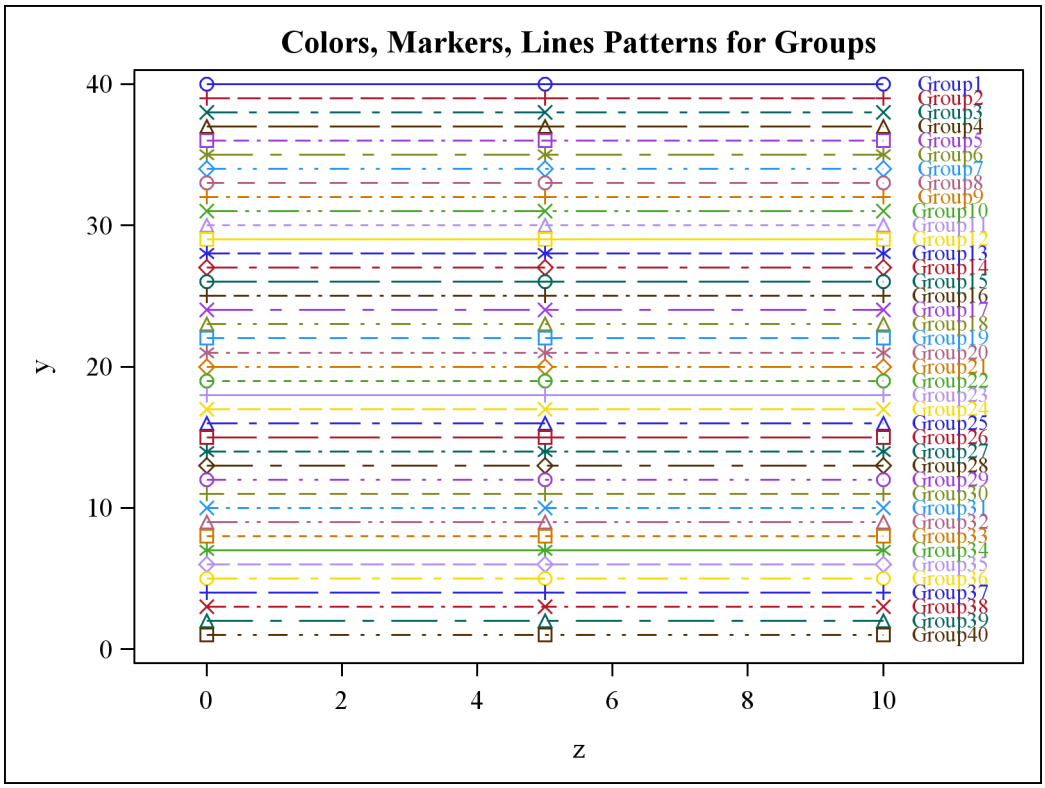


Figure 21.42 Markers, Lines, and Colors with Groups in the RTF Style



Modifying the HTMLBLUE Style

The HTMLBLUE style is an all-color style for the first 12 groups of observations. After each set of 12 groups, the line style and marker change for the next 12 groups. See Figure 21.36. The HTMLBLUECML style is a color style in which groups of observations are distinguished by simultaneous color, line style, and symbol changes. See Figure 21.37. For some graphs, you might want more differentiation than you get in an all-color style like HTMLBLUE but without the overkill differentiation of the HTMLBLUECML style and other styles. This section defines four new styles for this purpose:

HTMLBLUEL – line styles and colors vary together with fixed markers for each set of 11 groups
HTMLBLUEM – markers and colors vary together with fixed line styles for each set of 11 groups
HTMLBLUEFL – line styles and colors vary together with fixed filled markers for each set of 11 groups
HTMLBLUEFM – filled markers and colors vary together with fixed line styles for each set of 5 groups

The following statements show part of the style definition for each of these styles:

```
define style Styles.HTMLBlueL;    parent = styles.htmlbluecml;
   style GraphFit2   from GraphFit2   /                          linestyle = 1;
   style GraphData1  from GraphData1  / markersymbol = "circle" linestyle = 1;
   style GraphData2  from GraphData2  / markersymbol = "circle" linestyle = 4;
   style GraphData3  from GraphData3  / markersymbol = "circle" linestyle = 8;
   style GraphData4  from GraphData4  / markersymbol = "circle" linestyle = 5;
   style GraphData5  from GraphData5  / markersymbol = "circle" linestyle = 14;
   style GraphData6  from GraphData6  / markersymbol = "circle" linestyle = 26;
   style GraphData7  from GraphData7  / markersymbol = "circle" linestyle = 15;
   style GraphData8  from GraphData8  / markersymbol = "circle" linestyle = 20;
   style GraphData9  from GraphData9  / markersymbol = "circle" linestyle = 41;
   style GraphData10 from GraphData10 / markersymbol = "circle" linestyle = 42;
   style GraphData11 from GraphData11 / markersymbol = "circle" linestyle = 2;
   style GraphData12 from GraphData12 / markersymbol = "square" linestyle = 1;
   style GraphData13 from GraphData1  / markersymbol = "square" linestyle = 4;
   style GraphData14 from GraphData2  / markersymbol = "square" linestyle = 8;
   style GraphData15 from GraphData3  / markersymbol = "square" linestyle = 5;
   . . .
end;
define style Styles.HTMLBlueM;    parent = styles.htmlbluecml;
   style GraphFit2   from GraphFit2   /                                  linestyle = 1;
   style GraphData1  from GraphData1  / markersymbol = "circle"          linestyle = 1;
   style GraphData2  from GraphData2  / markersymbol = "square"          linestyle = 1;
   style GraphData3  from GraphData3  / markersymbol = "diamond"         linestyle = 1;
   style GraphData4  from GraphData4  / markersymbol = "asterisk"        linestyle = 1;
   style GraphData5  from GraphData5  / markersymbol = "plus"            linestyle = 1;
   style GraphData6  from GraphData6  / markersymbol = "triangle"        linestyle = 1;
   style GraphData7  from GraphData7  / markersymbol = "circlefilled"    linestyle = 1;
   style GraphData8  from GraphData8  / markersymbol = "starfilled"      linestyle = 1;
   style GraphData9  from GraphData9  / markersymbol = "squarefilled"    linestyle = 1;
   style GraphData10 from GraphData10 / markersymbol = "diamondfilled"   linestyle = 1;
   style GraphData11 from GraphData11 / markersymbol = "trianglefilled" linestyle = 1;
   style GraphData12 from GraphData12 / markersymbol = "circle"          linestyle = 4;
   style GraphData13 from GraphData1  / markersymbol = "square"          linestyle = 4;
   style GraphData14 from GraphData2  / markersymbol = "diamond"         linestyle = 4;
   style GraphData15 from GraphData3  / markersymbol = "asterisk"        linestyle = 4;
   . . .
```

```
end;
define style Styles.HTMLBlueFL;    parent = styles.htmlbluecml;
   style GraphFit2   from GraphFit2   /                                linestyle = 1;
   style GraphData1  from GraphData1  / markersymbol = "circlefilled" linestyle = 1;
   style GraphData2  from GraphData2  / markersymbol = "circlefilled" linestyle = 4;
   style GraphData3  from GraphData3  / markersymbol = "circlefilled" linestyle = 8;
   style GraphData4  from GraphData4  / markersymbol = "circlefilled" linestyle = 5;
   style GraphData5  from GraphData5  / markersymbol = "circlefilled" linestyle = 14;
   style GraphData6  from GraphData6  / markersymbol = "circlefilled" linestyle = 26;
   style GraphData7  from GraphData7  / markersymbol = "circlefilled" linestyle = 15;
   style GraphData8  from GraphData8  / markersymbol = "circlefilled" linestyle = 20;
   style GraphData9  from GraphData9  / markersymbol = "circlefilled" linestyle = 41;
   style GraphData10 from GraphData10 / markersymbol = "circlefilled" linestyle = 42;
   style GraphData11 from GraphData11 / markersymbol = "circlefilled" linestyle = 2;
   style GraphData12 from GraphData12 / markersymbol = "starfilled"   linestyle = 1;
   style GraphData13 from GraphData1  / markersymbol = "starfilled"   linestyle = 4;
   style GraphData14 from GraphData2  / markersymbol = "starfilled"   linestyle = 8;
   style GraphData15 from GraphData3  / markersymbol = "starfilled"   linestyle = 5;
   . . .
end;
define style Styles.HTMLBlueFM;    parent = styles.htmlbluecml;
   style GraphFit2   from GraphFit2   /                                  linestyle = 1;
   style GraphData1  from GraphData1  / markersymbol = "circlefilled"   linestyle = 1;
   style GraphData2  from GraphData2  / markersymbol = "starfilled"     linestyle = 1;
   style GraphData3  from GraphData3  / markersymbol = "squarefilled"   linestyle = 1;
   style GraphData4  from GraphData4  / markersymbol = "diamondfilled"  linestyle = 1;
   style GraphData5  from GraphData5  / markersymbol = "trianglefilled" linestyle = 1;
   style GraphData6  from GraphData6  / markersymbol = "circlefilled"   linestyle = 4;
   style GraphData7  from GraphData7  / markersymbol = "starfilled"     linestyle = 4;
   style GraphData8  from GraphData8  / markersymbol = "squarefilled"   linestyle = 4;
   style GraphData9  from GraphData9  / markersymbol = "diamondfilled"  linestyle = 4;
   style GraphData10 from GraphData10 / markersymbol = "trianglefilled" linestyle = 4;
   style GraphData11 from GraphData11 / markersymbol = "circlefilled"   linestyle = 8;
   style GraphData12 from GraphData12 / markersymbol = "starfilled"     linestyle = 8;
   style GraphData13 from GraphData1  / markersymbol = "squarefilled"   linestyle = 8;
   style GraphData14 from GraphData2  / markersymbol = "diamondfilled"  linestyle = 8;
   style GraphData15 from GraphData3  / markersymbol = "trianglefilled" linestyle = 8;
   . . .
end;
```

New **GraphData***n* style elements are created that inherit colors from the **GraphData1** through **GraphData12** style elements. The line styles and markers are explicitly set in the new style definitions. The **style GraphFit2 from GraphFit2 / linestyle = 1** statement creates a solid second fit line. You can remove that statement if you prefer a dashed second fit line.

The following statements use SAS macros to generate these four new styles:

```
proc template;
   %let m = circle square diamond asterisk plus triangle circlefilled
            starfilled squarefilled diamondfilled trianglefilled;
   %let ls = 1 4 8 5 14 26 15 20 41 42 2;
   %macro makestyle;
      %let l = %eval(%sysfunc(mod(&k,12))+1);
      %let k = %eval(&k+1);
      style GraphData&k from GraphData&l /
            linestyle=%scan(&ls, &j) markersymbol="%scan(&m, &i)";
      %mend;
```

```
define style styles.HTMLBlueL;
   parent=styles.htmlbluecml;
   style GraphFit2 from GraphFit2 / linestyle = 1;
   %macro htmlbluel;
      %let k = 0;
      %do i = 1 %to 11; %do j = 1 %to 11; %makestyle %end; %end;
      %mend;
    %htmlbluel
    end;
define style styles.HTMLBlueM;
   parent=styles.htmlbluecml;
   style GraphFit2 from GraphFit2 / linestyle = 1;
   %macro htmlbluem;
      %let k = 0;
      %do j = 1 %to 11; %do i = 1 %to 11; %makestyle %end; %end;
      %mend;
    %htmlbluem
    end;
%let m = circlefilled starfilled squarefilled diamondfilled trianglefilled;
define style styles.HTMLBlueFL;
   parent=styles.htmlbluecml;
   style GraphFit2 from GraphFit2 / linestyle = 1;
   %macro htmlbluel;
      %let k = 0;
      %do i = 1 %to 5; %do j = 1 %to 11; %makestyle %end; %end;
      %mend;
    %htmlbluel
    end;
define style styles.HTMLBlueFM;
   parent=styles.htmlbluecml;
   style GraphFit2 from GraphFit2 / linestyle = 1;
   %macro htmlbluem;
      %let k = 0;
      %do j = 1 %to 11; %do i = 1 %to 5; %makestyle %end; %end;
      %mend;
    %htmlbluem
    end;
run;
```

The %LET m statement provides the list of markers. The %LET ls statement provides the list of line styles. The MAKESTYLE macro makes the kth style element from the **GraphData**n style element for $n = \text{mod}(k - 1, 12) + 1$. The remaining macros vary markers and line styles in the appropriate order over the elements in each list.

The following step that was used in the section "Style Comparisons" on page 656 is used with the different styles to produce Figure 21.43 through Figure 21.46:

```
proc sgplot data=x2;
   title 'Colors, Markers, Lines Patterns for Groups';
   series  y=y x=x / group=group markers;
   scatter y=y x=z / group=group markerchar=l;
run;
```

Figure 21.43 Markers, Lines, and Colors with Groups in the HTMLBLUEL Style

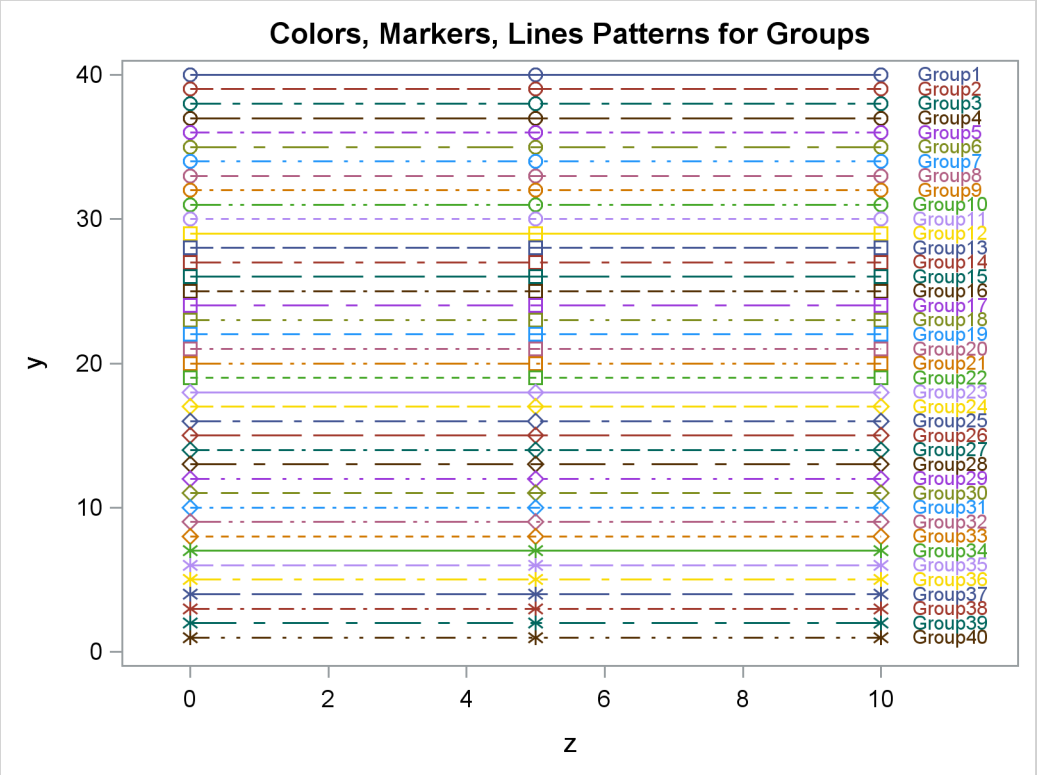


Figure 21.44 Markers, Lines, and Colors with Groups in the HTMLBLUEM Style

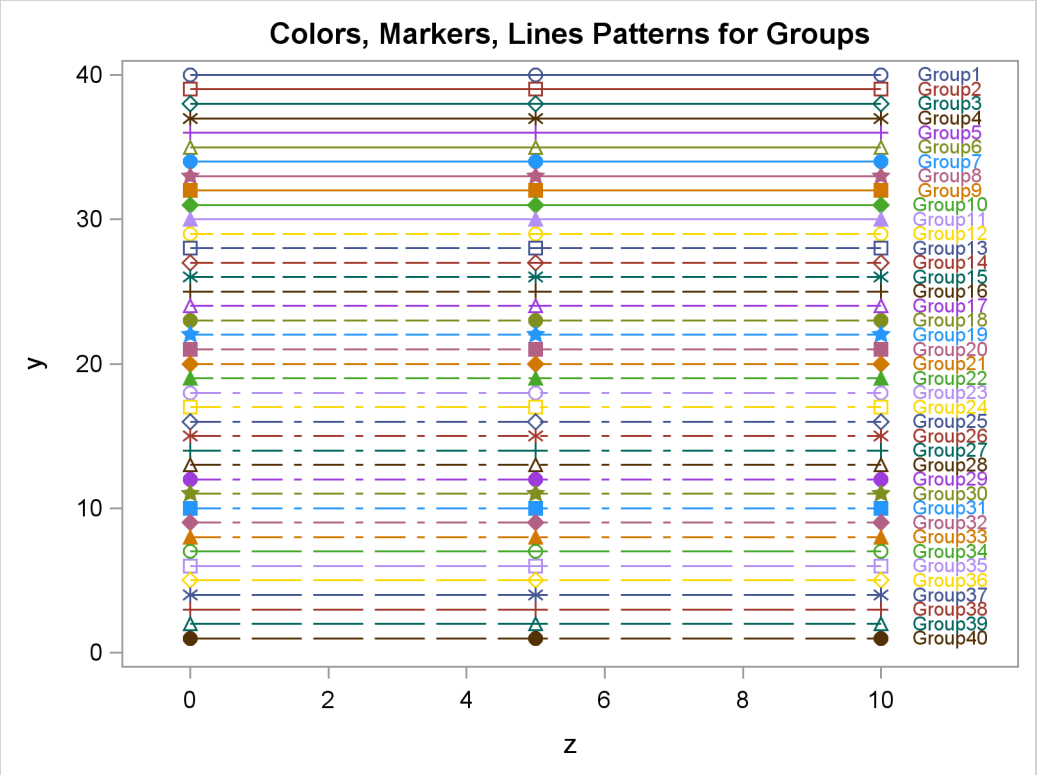


Figure 21.45 Markers, Lines, and Colors with Groups in the HTMLBLUEFL Style

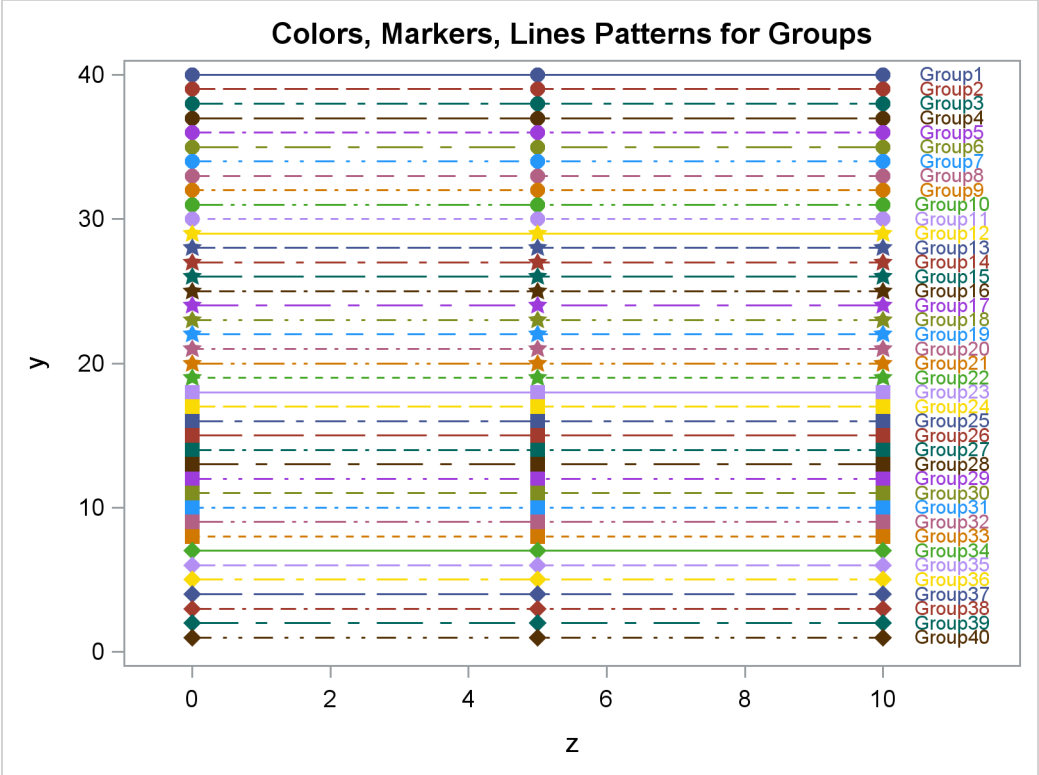
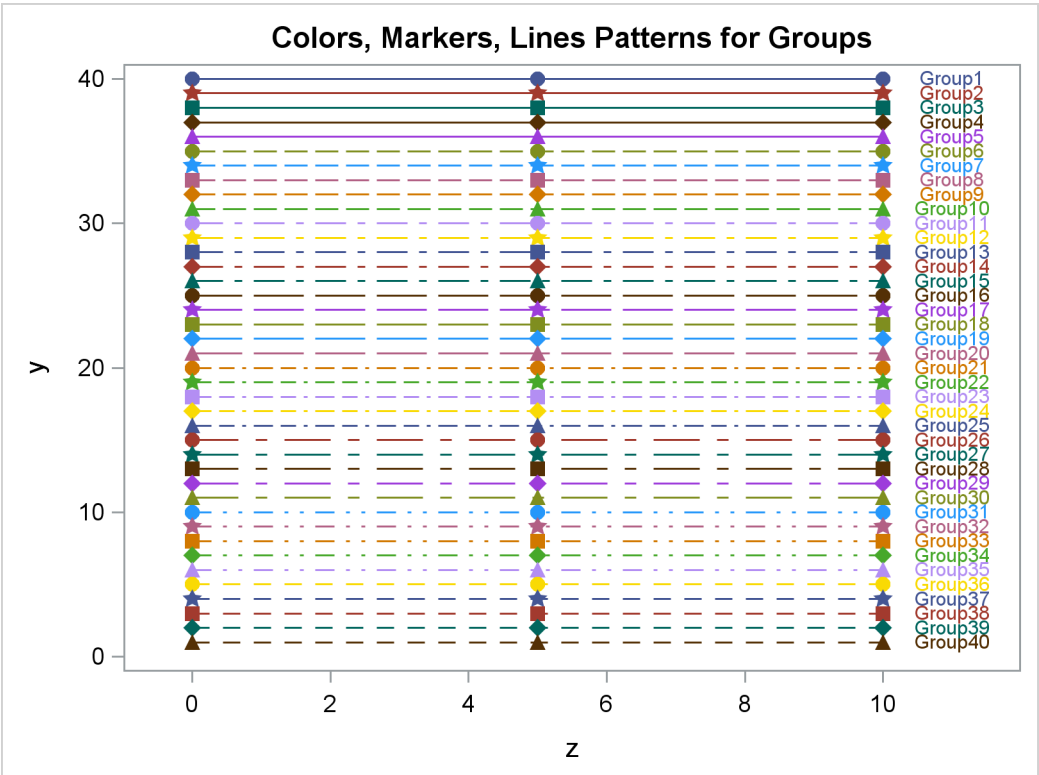


Figure 21.46 Markers, Lines, and Colors with Groups in the HTMLBLUEFM Style



Style Template Modification Macro

The **%ModStyle** macro provides easy ways to customize the style elements (**GraphData1**–**GraphData***n*) that control how groups of observations are distinguished. Examples of using the **%ModStyle** macro can be found in the sections "Creating an All-Color Style" on page 676 and "Changing the Default Markers and Lines" on page 678. Also see Kuhfeld (2009) for more information about this macro.

You do not need to include autocall macros (for example, with a **%include** statement). You can call them directly once they are properly installed. If your site has installed the autocall libraries supplied by the SAS System and uses the standard configuration of SAS software, you need to ensure that the SAS system option MAUTOSOURCE is in effect to begin using the autocall macros. For more information about autocall libraries, see the *SAS Macro Language: Reference*. For details about installing autocall macros, consult your host documentation.

The **%ModStyle** macro has the following options:

COLORS=*color-list*

specifies a space-delimited list of colors for markers and lines. If you do not specify this option, then the colors from the parent style are used. You can specify the colors using any SAS color notation such as CX*rrggbb*.

COLORS=GRAYS generates seven distinguishable grayscale colors from blackest to whitest. The colors should be mixed up to be more easily distinguished when you need fewer colors, but you can do that with your own COLORS= list. The HLS (hue/light/saturation) coding generates colors by setting hue and saturation to 0 and incrementing the lightness for each gray. You can also use the keywords BLUES, PURPLES, MAGENTAS, REDS, ORANGES, YELLOWS, GREENS, and CYANS to generate seven colors with a fixed hue and a saturation of AA (hex).

COLORS=SHADES INT generates seven colors as described previously, except that you specify an integer $0 \leq \text{INT} < 360$. See *SAS/GRAPH: Reference*. The available hues include: GRAY, GREY, BLUE=0, PURPLE=30, MAGENTA=60, RED=120, ORANGE=150, YELLOW=180, GREEN=240, and CYAN=300.

DISPLAY=*n*

specifies whether to display the generated template. By default, the template is not displayed. Specify DISPLAY=1 to display the generated template.

FILLCOLORS=*color-list*

specifies a space-delimited list of colors for bands and fills. If you do not specify this option, then the colors from the parent style are used.

Fill colors from the parent style are designed to work well with the colors from the parent style. If you specify a COLORS= list, then you might want to redefine the FILLCOLORS= list as well. You need to have at least as many fill colors as you have colors (any extra fill colors are ignored). Two shortcuts are available: FILLCOLORS=COLORS uses the COLORS= colors for the fills (your confidence bands should have transparency for this to be useful) and FILLCOLORS=LIGHTCOLORS modifies the lightness associated with each color generated by COLORS=SHADES (this is allowed only with COLORS=SHADES).

LINESTYLES=*line-style-list*

specifies a space-delimited list of line styles. The default is:

```
LineStyles=Solid MediumDash MediumDashShortDash LongDash
           DashDashDot LongDashShortDash DashDotDot Dash
           ShortDashDot MediumDashDotDot ShortDash
```

Line style numbers can range from 1 to 46. Some line styles have names associated with them. You can specify either the name or the number for the following number/name pairs: 1 Solid, 2 ShortDash, 4 MediumDash, 5 LongDash, 8 MediumDashShortDash, 14 DashDashDot, 15 DashDotDot, 20 Dash, 26 LongDashShortDash, 34 Dot, 35 ThinDot, 41 ShortDashDot, and 42 MediumDashDotDot.

MARKERS=*marker-list*

specifies a space-delimited list of marker symbols. By default, **`Markers=Circle Plus X Triangle Square Asterisk Diamond`**. The available marker symbols are listed in *SAS Graph Template Language: Reference Guide*. Two shortcuts are available: MARKERS=FILLED is an alias for the specification **`Markers=CircleFilled TriangleFilled SquareFilled DiamondFilled StarFilled HomeDownFilled`**, and MARKERS=EMPTY is an alias for the specification **`Markers=Circle Triangle Square Diamond Star HomeDown`**.

NAME=*style-name*

specifies the name of the new style that you are creating. This name is used when you specify the style in an ODS destination statement (for example, ODS HTML STYLE=*style-name*). The default is NAME=NEWSTYLE.

NUMBEROFGROUPS=*n*

specifies *n*, the number of **`GraphData`***n* style elements to create. The **`GraphData1`**–**`GraphData`***n* style elements contain *n* combinations of colors, markers, and line styles. By default, 32 combinations are created.

PARENT=*style-name*

specifies the parent style. The new style inherits most of its attributes from the parent style. The default is PARENT=DEFAULT (which is one of the default styles for HTML and the parent style for all of the styles that are recommended for statistical graphics). If your goals are to change colors or create an all-color style, you can use any style as the parent style. However, if your goal is to change markers or line styles without creating an all-color style, do not use the HTMLBLUE style as a parent. The HTMLBLUE style is an all-color style that is different from other styles due to its use of the ATTRPRIORITY= style option.

TYPE=*type-specification*

specifies how your new style cycles through colors, markers, and line styles. The default is TYPE=LMbyC.

These first three methods work well with all plots, because cycling line styles and markers together ensures that both scatterplot markers and series plot lines are distinguishable:

CLM

cycles through colors, line styles, and markers simultaneously. The first group uses the first color, line style, and marker; the second group uses the second color, line style, and marker; and so on. This is the method used by the ODS Graphics styles.

LMbyC
fixes line style and marker, cycles through colors, and then moves to the next line style and marker. This is the default and creates a style where the first groups are distinguished entirely by color.

CbyLM
fixes color, cycles through line style and marker, and then moves to the next color. This option uses the smaller of the number of line styles or the number of markers when cycling within a color.

The following two methods might not work well with all plots:

CbyLbyM
fixes color and line style, then cycles through markers, increments line style, and then cycles through markers. After all line styles have been used, then this option moves to the next color and continues.

LbyMbyC
fixes line style and marker, then cycles through colors, increments marker, and then cycles through colors. After all markers have been used, then this option moves to the next line style and continues. This is closest to the legacy SAS/GRAPH method.

Creating an All-Color Style

Many styles are designed to make color plots where lines, functions, and groups of observations can be distinguished even when the plot is sent to a black-and-white printer. Hence, lines differ not only in color but also in pattern. Similarly, markers differ in both color and symbol. This is not true with the HTMLBLUE style, which is an all-color style.

You can easily modify any style to be an all-color style by using the ATTRPRIORITY= option. For example:

```
proc template;
   define style styles.StatColor;
      parent = statistical;
      style Graph from Graph / attrpriority = "Color";
      end;
   run;
```

Alternatively, you can make an all-color style with the %MODSTYLE autocall macro. It creates a new style by modifying a parent style and reordering the colors, line patterns, and marker symbols in the **GraphData***n* style elements (see the section "Some Common Style Elements" on page 650). By default, the macro creates a new style that distinguishes lines and groups only by color. The macro is documented in the section "Style Template Modification Macro" on page 674.

The following example illustrates the default use of the macro and is taken from the section "Fitting a Curve through a Scatter Plot" on page 7719 of Chapter 93, "The TRANSREG Procedure." The data come from an experiment in which nitrogen oxide emissions from a single cylinder engine are measured for various combinations of fuel and equivalence ratio. This gas data set is available from the Sashelp library.

The following statements fit separate curves for each group and produce Figure 21.47 and Figure 21.48:

```
ods listing style=statistical;
ods graphics on;

proc transreg data=sashelp.Gas ss2 plots=transformation lprefix=0;
   model identity(nox) = class(Fuel / zero=none) * pbspline(EqRatio);
run;

%modstyle(parent=statistical, name=StatColor)
ods listing style=StatColor;

proc transreg data=sashelp.Gas ss2 plots=transformation lprefix=0;
   model identity(nox) = class(Fuel / zero=none) * pbspline(EqRatio);
run;
```

The first PROC TRANSREG step uses the STATISTICAL style to create the fit plot in Figure 21.47, which uses different colors, line patterns, and markers for each group. Then the macro creates a new style, called STATCOLOR, that inherits its characteristics from the STATISTICAL style. Only the attributes of the lines and markers are changed. In Figure 21.48, which is created with the modified style, the groups are differentiated only by color. This is the easiest and most common way for you to use this macro. However, you can use it to perform other style modifications as illustrated in the section "Changing the Default Markers and Lines" on page 678. The macro is documented in the section "Style Template Modification Macro" on page 674.

Figure 21.47 Fit Plot with the STATISTICAL Style

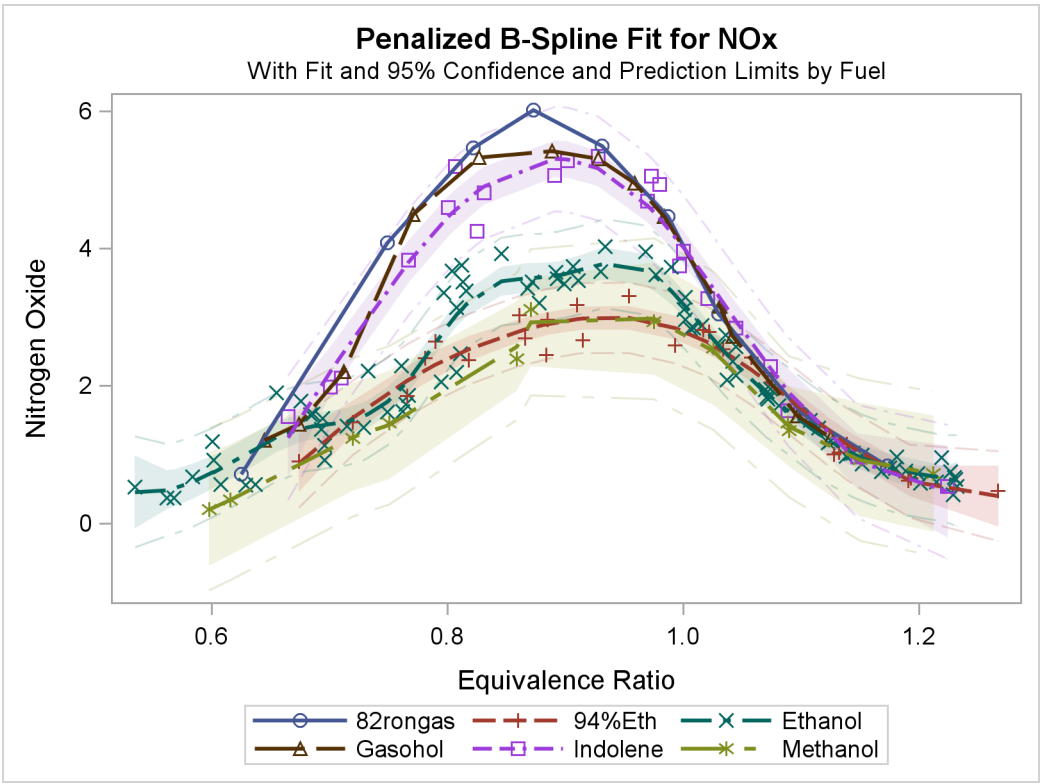
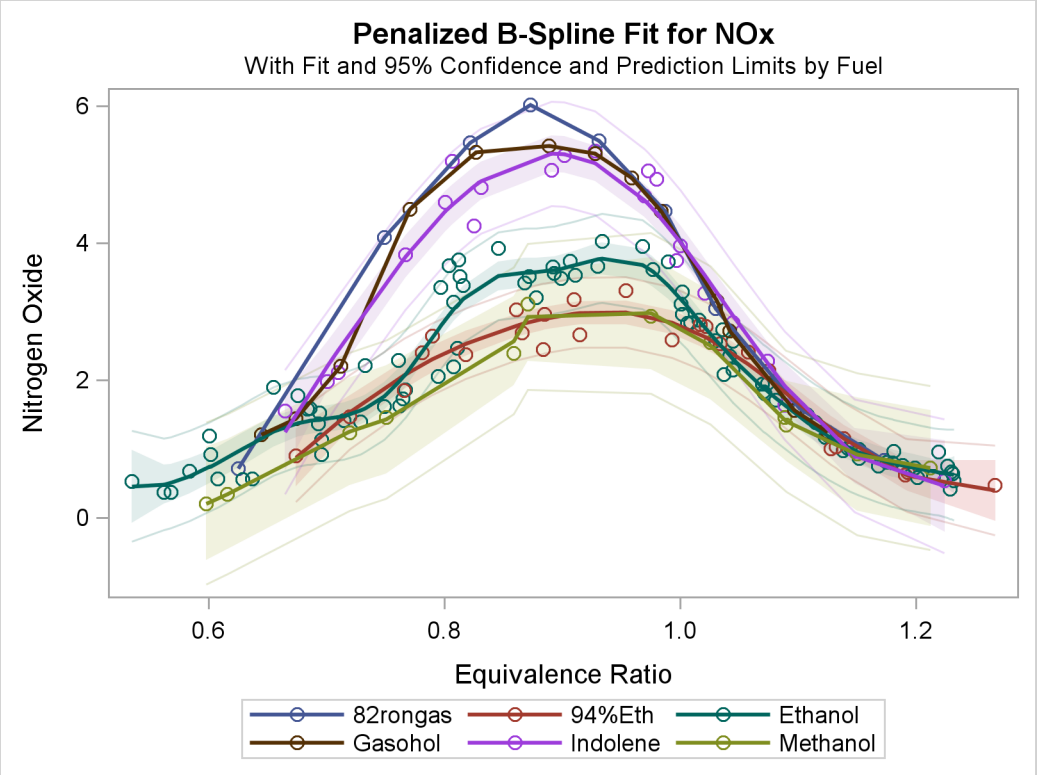


Figure 21.48 Fit Plot with the Modified Style



Changing the Default Markers and Lines

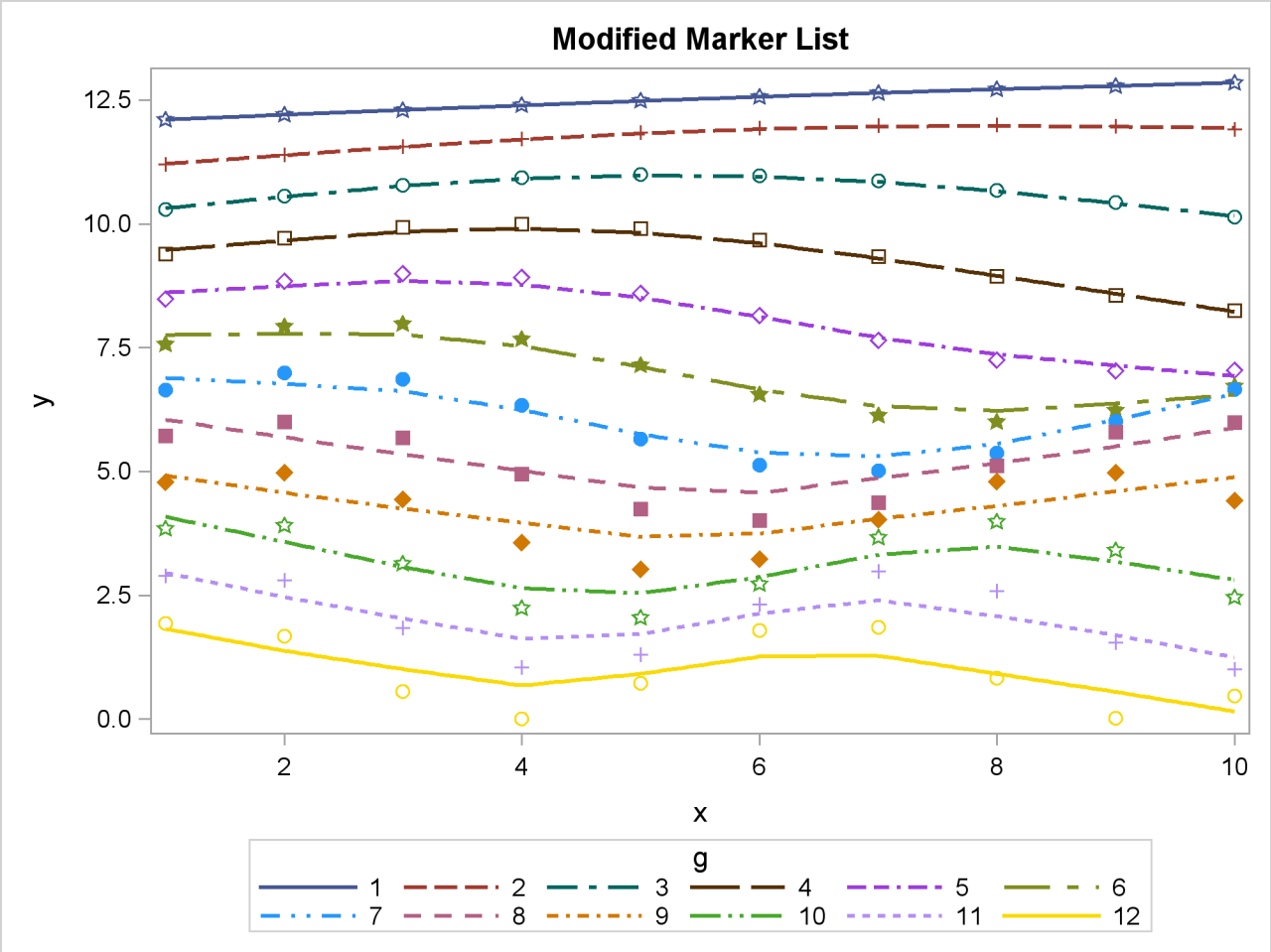
The preceding section shows how to use the %MODSTYLE autocall macro to create an all-color style. You can also use the %MODSTYLE macro to change markers and line styles. This example creates a new style called MARKSTYLE that inherits from the STATISTICAL style but uses a different set of markers. The following statements create artificial data, change the marker list, and display the results:

```
data x;
   do g = 1 to 12;
      do x = 1 to 10;
         y = 13 - g + sin(x * 0.1 * g);
         output;
      end;
   end;
run;

%modstyle(name=markstyle, parent=statistical, type=CLM,
          markers=star plus circle square diamond starfilled
                  circlefilled squarefilled diamondfilled)
```

```
ods listing style=markstyle;

proc sgplot;
   title 'Modified Marker List';
   loess y=y x=x / group=g;
run;
```

The NAME= option specifies the new style name, and the PARENT= option specifies the parent style. The TYPE= option controls the method of cycling through colors, lines, and markers. The default, TYPE=LMbyC, fixes (holds constant) the line styles and markers, while cycling through the color list. This is illustrated in the section "Creating an All-Color Style" on page 676. This example uses TYPE=CLM to cycle through colors, line styles, and markers (holding none of them fixed). Other TYPE= values are described in the section "Style Template Modification Macro" on page 674. The values specified with the TYPE= option are case-sensitive ('by' is lower case and the 'L', 'C', and 'M' are upper case). The new marker list is specified with the MARKERS= option. The results are displayed in Figure 21.49. The marker list is reused in the tenth and subsequent groups since only nine markers are defined.

Figure 21.49 A Modified Style with a New List of Markers

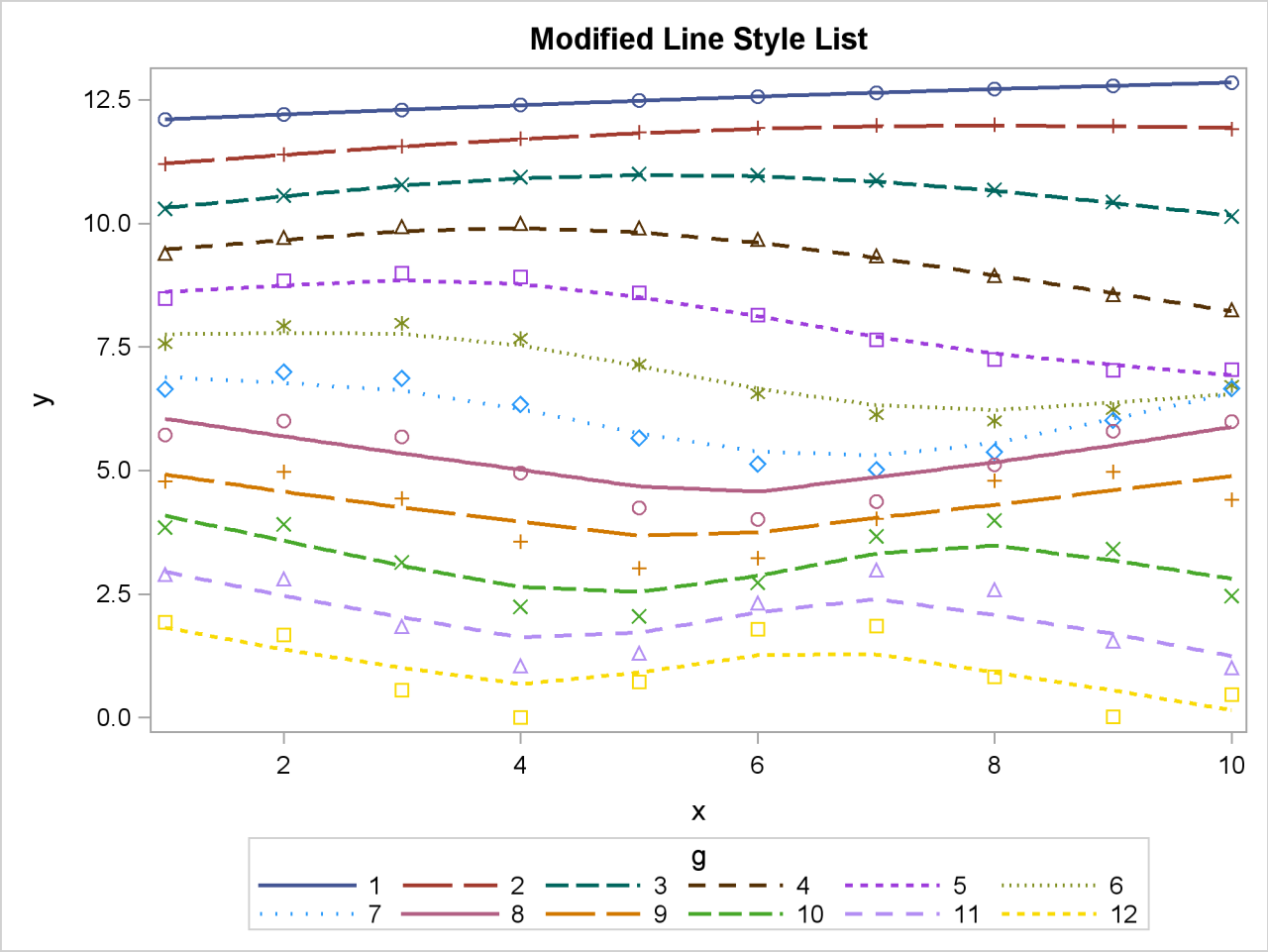


The following statements create a new style called LINESTYLE that inherits from the STATISTICAL style and changes the line list:

```
%modstyle(name=linestyle, parent=statistical, type=CLM,
          linestyles=Solid LongDash MediumDash Dash ShortDash Dot ThinDot)

ods listing style=linestyle;

proc sgplot;
   title 'Modified Line Style List';
   loess y=y x=x / group=g;
run;
```

The new line list is specified with the LINESTYLES= option. The results are displayed in Figure 21.50. In this example, each of the first seven groups uses a dash pattern that is shorter than the previous group. The line list is reused in the eighth and subsequent groups since only seven line patterns are defined.

Figure 21.50 Modified Style with a New List of Line Styles



You can learn more about style modification by examining the new styles, as in the following example:

```
proc template;
   source styles.markstyle;
   source styles.linestyle;
run;
```

The results show the definitions of **GraphData1** through **GraphData32** that the macro created. An abridged listing of the results follows:

```
define style Styles.Markstyle;
   parent = Styles.statistical;
   . . .
   style GraphData1 /
      markersymbol = "star"
      linestyle = 1
      contrastcolor = ColorStyles('c1')
      color = FillStyles('f1');
   . . .
   style GraphData32 /
      markersymbol = "diamond"
      linestyle = 42
      contrastcolor = ColorStyles('c8')
      color = FillStyles('f8');
end;

define style Styles.Linestyle;
   parent = Styles.statistical;
   . . .
   style GraphData1 /
      markersymbol = "circle"
      linestyle = 1
      contrastcolor = ColorStyles('c1')
      color = FillStyles('f1');
   . . .
   style GraphData32 /
      markersymbol = "triangle"
      linestyle = 20
      contrastcolor = ColorStyles('c8')
      color = FillStyles('f8');
end;
```

You can use the NUMBEROFGROUPS= option in the %MODSTYLE macro to control the number of **GraphData***n* style elements created in the new style.

Modifying Graph Fonts in Styles

You can modify an ODS style to customize the general appearance of plots produced with ODS Graphics, just as you can modify a style to customize the general appearance of ODS tables. This section shows you how to customize fonts used in graphs. The following step displays the HTMLBLUE style and its parent styles, STATISTICAL and DEFAULT:

```
proc template;
   source Styles.HTMLBlue;
   source Styles.Statistical;
   source Styles.Default;
run;
```

If you search for 'font', you find the style elements that control graph fonts:

```
style GraphFonts /
   'GraphDataFont' = ("<sans-serif>, <MTsans-serif>",7pt)
   'GraphUnicodeFont' = ("<MTsans-serif-unicode>",9pt)
   'GraphValueFont' = ("<sans-serif>, <MTsans-serif>",9pt)
   'GraphLabelFont' = ("<sans-serif>, <MTsans-serif>",10pt)
   'GraphFootnoteFont' = ("<sans-serif>, <MTsans-serif>",10pt,italic)
   'GraphTitleFont' = ("<sans-serif>, <MTsans-serif>",11pt,bold)
   'GraphTitle1Font' = ("<sans-serif>, <MTsans-serif>",14pt,bold)
   'GraphAnnoFont' = ("<sans-serif>, <MTsans-serif>",10pt);
```

The font **GraphTitle1Font** is used only in traditional graphics; it is not used with ODS Graphics. The following fonts are the ones typically used for the text in most graphs:

- **GraphDataFont** is the smallest font. It is used for text that needs to be small (labels for points in scatter plots, labels for contours, and so on)
- **GraphValueFont** is the next largest font. It is used for axis value (tick marks) labels and legend entry labels.
- **GraphLabelFont** is the next largest font. It is used for axis labels and legend titles.
- **GraphFootnoteFont** is the next largest font. It is used for all footnotes.
- **GraphTitleFont** is the largest font. It is used for all titles.
- **GraphUnicodeFont** is used for special characters. See the section "Unicode and Special Characters" on page 749.

The following statements define a style named NEWSTYLE that replaces the graph fonts in the DEFAULT style with italic Times New Roman fonts, which are available with the Windows operating system:

```
proc template;
   define style Styles.NewStyle;
      parent=Styles.Statistical;
      replace GraphFonts  /
         'GraphDataFont'     = ("<MTserif>, Times New Roman",7pt)
         'GraphUnicodeFont'  = ("<MTserif>, Times New Roman",9pt)
         'GraphValueFont'    = ("<MTserif>, Times New Roman",9pt)
         'GraphLabelFont'    = ("<MTserif>, Times New Roman",10pt)
         'GraphFootnoteFont' = ("<MTserif>, Times New Roman",10pt)
         'GraphTitleFont'    = ("<MTserif>, Times New Roman",11pt)
         'GraphTitle1Font'   = ("<MTserif>, Times New Roman",14pt)
         'GraphAnnoFont'     = ("<MTserif>, Times New Roman",10pt);
   end;
run;
```

For more information about the DEFINE, PARENT, and REPLACE statements, see the *SAS Graph Template Language: Reference Guide*.

The "Getting Started" section of Chapter 77, "The ROBUSTREG Procedure," creates the following data set to illustrate the use of the PROC ROBUSTREG for robust regression:

```
data stack;
   input x1 x2 x3 y @@;
   datalines;
80 27 89 42   80 27 88 37   75 25 90 37   62 24 87 28   62 22 87 18
62 23 87 18   62 24 93 19   62 24 93 20   58 23 87 15   58 18 80 14
58 18 89 14   58 17 88 13   58 18 82 11   58 19 93 12   50 18 89  8
50 18 86  7   50 19 72  8   50 19 79  8   50 20 80  9   56 20 82 15
70 20 91 15
;
```

The following statements create a Q-Q plot that uses the HTMLBLUE style (see Figure 21.51) and the NEWSTYLE style (see Figure 21.52):

```
ods listing style=HTMLBlue;
ods graphics on;

proc robustreg data=stack plots=qqplot;
   ods select QQPlot;
   model y = x1 x2 x3;
run;

ods listing close;
ods listing style=NewStyle;

proc robustreg data=stack plots=qqplot;
   ods select QQPlot;
   model y = x1 x2 x3;
run;
```

Figure 21.51 Q-Q Plot That Uses the HTMLBLUE Style

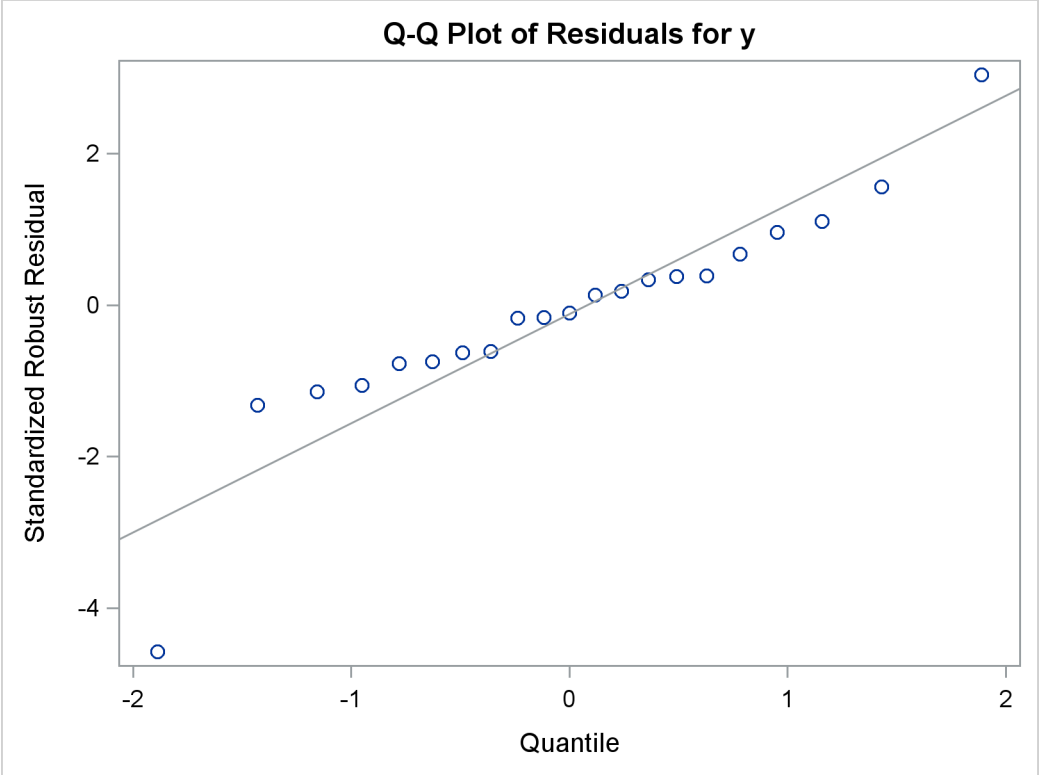
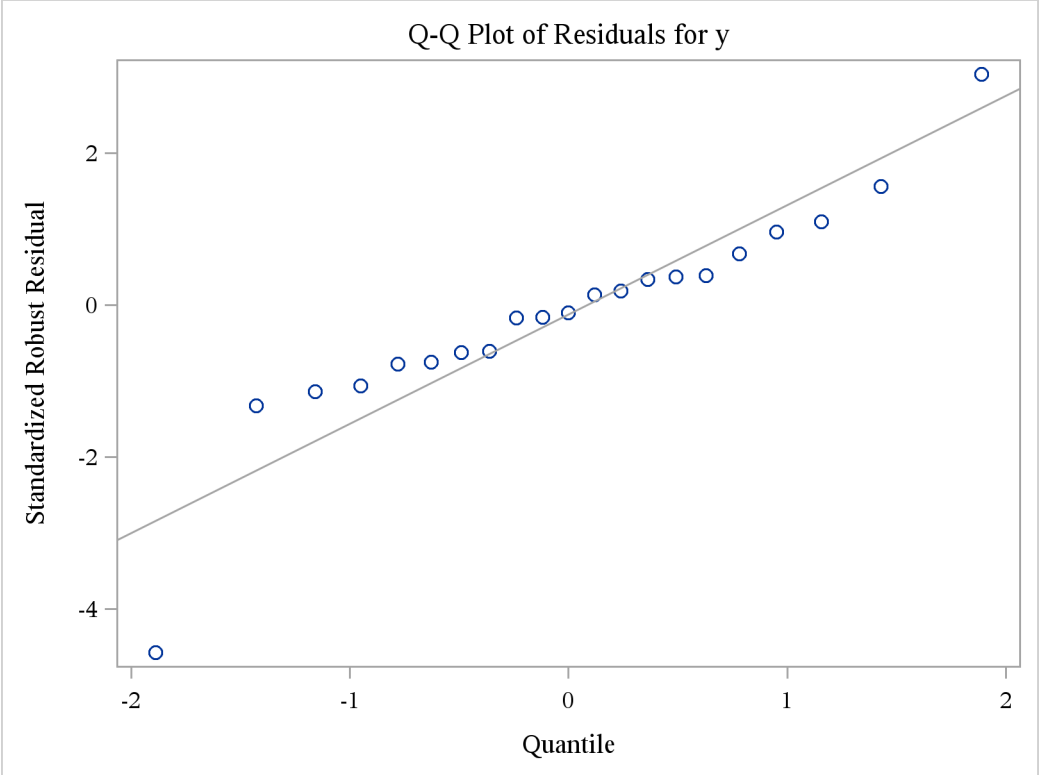


Figure 21.52 Q-Q Plot That Uses the NEWSTYLE Style



Although this example illustrates the use of a style with graphical output from a particular procedure, a style is applied to *all* of your output (graphs and tables) in the destination for which you specify the style. See the section "Changing the Default Style" on page 687 for information about specifying a default style for all your output.

Modifying Other Graph Elements in Styles

This section illustrates how to modify other style elements for graphics, specifically the style element **GraphReference**, which controls the attributes of reference lines. You can run the following statements to learn more about the **GraphReference** style element:

```
proc template;
   source styles.HTMLBlue;
run;
```

The following are the first two lines of the source listing:

```
define style Styles.HTMLBlue;
   parent = styles.statistical;
```

There is no mention of **GraphReference** in the complete listing of the source because **GraphReference** is inherited from a parent style. Most styles inherit many of their attributes from other styles. To find out more, you must list the parent style, as in the following example:

```
proc template;
   source styles.statistical;
run;
```

Most styles that you typically use with ODS Graphics inherit most of their attributes from only one style, the DEFAULT style. The HTMLBLUE style inherits from the STATISTICAL style, which inherits from the DEFAULT style. A few of the other styles inherit from several parents. You might have to repeat this process multiple times to find the first parent. The following step displays the HTMLBLUE style and its parent styles, STATISTICAL and DEFAULT:

```
proc template;
   source Styles.HTMLBlue;
   source Styles.Statistical;
   source Styles.Default;
run;
```

Styles are listed in the order: style of interest, then its parent, and then its grandparent. If you search the results from the top, you will find the most recent specification of a style element first. The **GraphReference** style element is defined as follows:

```
class GraphReference /
   linethickness = 1px
   linestyle = 1
   contrastcolor = GraphColors('greferencelines');
```

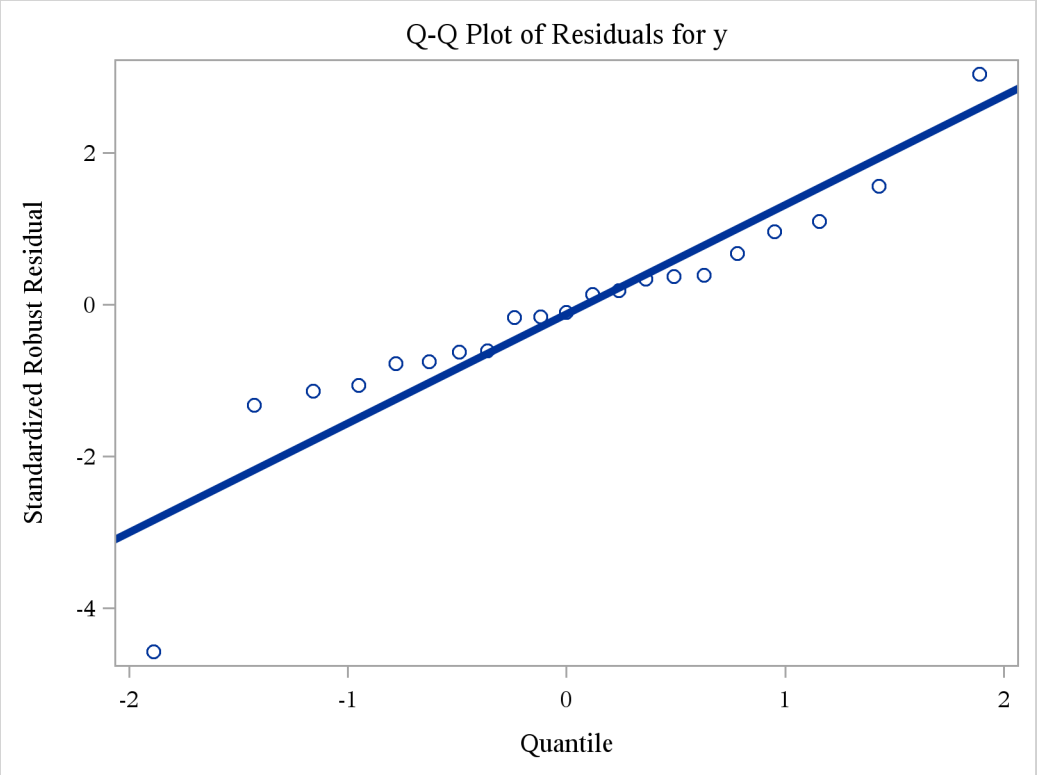
To specify a line thickness of 4 pixels for all reference lines, add the following statement to the definition of the NEWSTYLE style in the section "Modifying Graph Fonts in Styles" on page 682:

```
replace GraphReference / linethickness=4px;
```

The following statements modify the style and produce the Q-Q plot shown in Figure 21.53:

```
proc template;
   define style Styles.NewStyle;
      parent=Styles.Statistical;
      replace GraphFonts  /
         'GraphDataFont'     = ("<MTserif>, Times New Roman",7pt)
         'GraphUnicodeFont'  = ("<MTserif>, Times New Roman",9pt)
         'GraphValueFont'    = ("<MTserif>, Times New Roman",9pt)
         'GraphLabelFont'    = ("<MTserif>, Times New Roman",10pt)
         'GraphFootnoteFont' = ("<MTserif>, Times New Roman",10pt)
         'GraphTitleFont'    = ("<MTserif>, Times New Roman",11pt)
         'GraphTitle1Font'   = ("<MTserif>, Times New Roman",14pt)
         'GraphAnnoFont'     = ("<MTserif>, Times New Roman",10pt);
      replace GraphReference / linethickness=4px;
   end;
run;

ods listing style=NewStyle;
ods graphics on;

proc robustreg data=stack plots=qqplot;
   ods select QQPlot;
   model y = x1 x2 x3;
run;
```

Figure 21.53 Q-Q Plot That Uses the NEWSTYLE Style with a Thicker Line



You can use this approach to modify other attributes of the line, such as **`LineStyle`** and **`ContrastColor`**. These style modifications apply to all graphs that display reference lines, and not just to Q-Q plots produced by PROC ROBUSTREG. You can control the attributes of specific graphs by modifying the graph template, as discussed in the section "Graph Templates" on page 711 in Chapter 22, "ODS Graphics Template Modification." Values specified directly in a graph template override style attributes.

When you are done with the NEWSTYLE style, you do not need to restore the HTMLBLUE style template since you did not modify it. Rather, you inherited from the HTMLBLUE style.

Changing the Default Style

The default style for each ODS destination is specified in the SAS Registry. For example, the default style for the HTML destination is DEFAULT (or HTMLBLUE in the SAS windowing environment) and the default style for the RTF destination is RTF. You can specify a default style for all of your output in a particular ODS destination. This is useful if you want to use a different SAS style, if you have modified one of the styles supplied by the SAS System (see the section "Style Templates and Colors" on page 649), or if you have defined your own style. For example, you can specify the JOURNAL style as the default style for RTF output.

The recommended approach for specifying a default style is as follows. Open the SAS Registry Editor by typing **regedit** on the command line. Expand the node **ODS ► DESTINATIONS** and select a destination

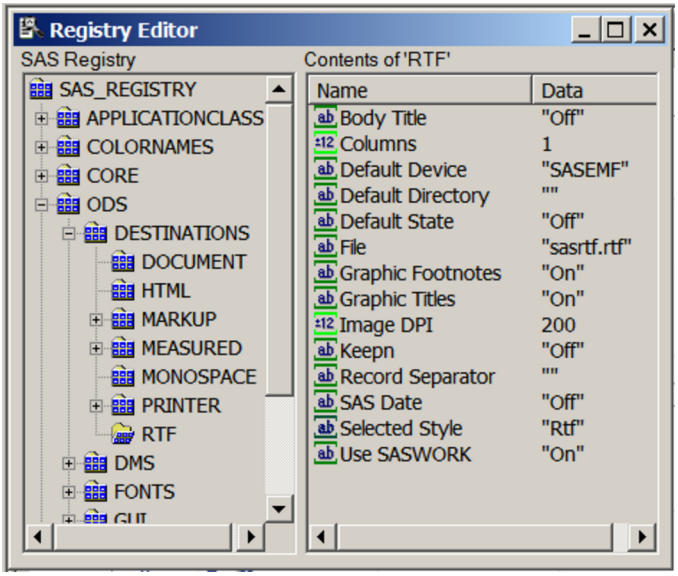
(for example, select **RTF**). Double-click the **Selected Style** item, shown in Figure 21.54, and specify a style. This can be any style supplied by the SAS System or a user-defined style, as long as it can be found with the current template search path (for example, specify **Journal**). You can specify a default style for the other destinations in a similar way.

In a few cases the default style is specified in more than one place. Assume that you are using the SAS windowing environment and Microsoft Windows or UNIX in the following;

- If you expand the node **ODS ► DESTINATIONS ► HTML**, you see that the **Selected Style** is DEFAULT.
- If you expand the node **ODS ► MARKUP ► HTML4**, you see that the **Selected Style** is DEFAULT.
- If you expand the node **ODS ► DMS ► DESTINATIONS ► MARKUP ► HTML4**, you see that the **Selected Style** is HTMLBLUE.

The HTMLBLUE style is the default style for HTML output in the SAS windowing environment, yet the DEFAULT style is the default style for HTML output in other contexts.

Figure 21.54 SAS Registry Editor



ODS searches sequentially through each element of the template search path for the first style template that matches the name of the style specified in the SAS Registry. The first style template found is used. (See the sections "Saving Customized Templates" on page 720, "Using Customized Templates" on page 721, and "Reverting to the Default Templates" on page 722 in Chapter 22, "ODS Graphics Template Modification," for more information about the template search path.) If you are specifying a customized style as your default style, the following are useful suggestions:

- If you save your style in Sasuser.Templat, verify that the name of your default style matches the name of the style specified in the SAS Registry. For example, suppose the RTF style is specified for the RTF destination in the SAS Registry. You can name your style RTF and save it in Sasuser.Templat.

This blocks the RTF style in Sashelp.Tmplmst (provided that you did not alter the default template search path).

- If you save your style in a user-defined template store, verify that this template store is the first in the current template search path. Include the ODS PATH statement in your SAS autoexec file so that it is executed at start-up.

For the HTML destination, an alternative approach for specifying a default style is as follows. From the menu at the top of the main SAS window, select **Tools ► Options ► Preferences**. On the **Results** tab, select the **Create HTML** check box and select a style from the **Style** list.

Statistical Graphics Procedures

Three Base SAS statistical graphics procedures use ODS Graphics and provide a convenient syntax for creating a variety of graphs from raw data or from procedure output.

SGSCATTER	creates single-cell and multi-cell scatter plots and scatter plot matrices with optional fits and ellipses.
SGPLOT	creates single-cell plots with a variety of plot and chart types.
SGPANEL	creates single-page or multi-page panels of plots and charts conditional on classification variables.

You do not need to enable ODS Graphics in order to use these procedures, which are called SG (statistical graphics) procedures. In addition, the Base SAS SGRENDER procedure provides a way to create plots from graph templates that you have modified or written yourself. See the *SAS ODS Graphics: Procedures Guide* and Kuhfeld (2010) for more information about the SG procedures and PROC SGRENDER.

These procedures do much more than make scatter plots. They can produce density plots, dot plots, needle plots, series plots, horizontal and vertical bar charts, histograms, and box plots. They can also compute and display loess fits, polynomial fits, penalized B-spline fits, reference lines, bands, and ellipses. PROC SGRENDER is the most flexible because it uses the Graph Template Language. The syntax for the other SG procedures is much simpler than that of the GTL, and so these procedures are recommended for creating most plots commonly required in statistical work.

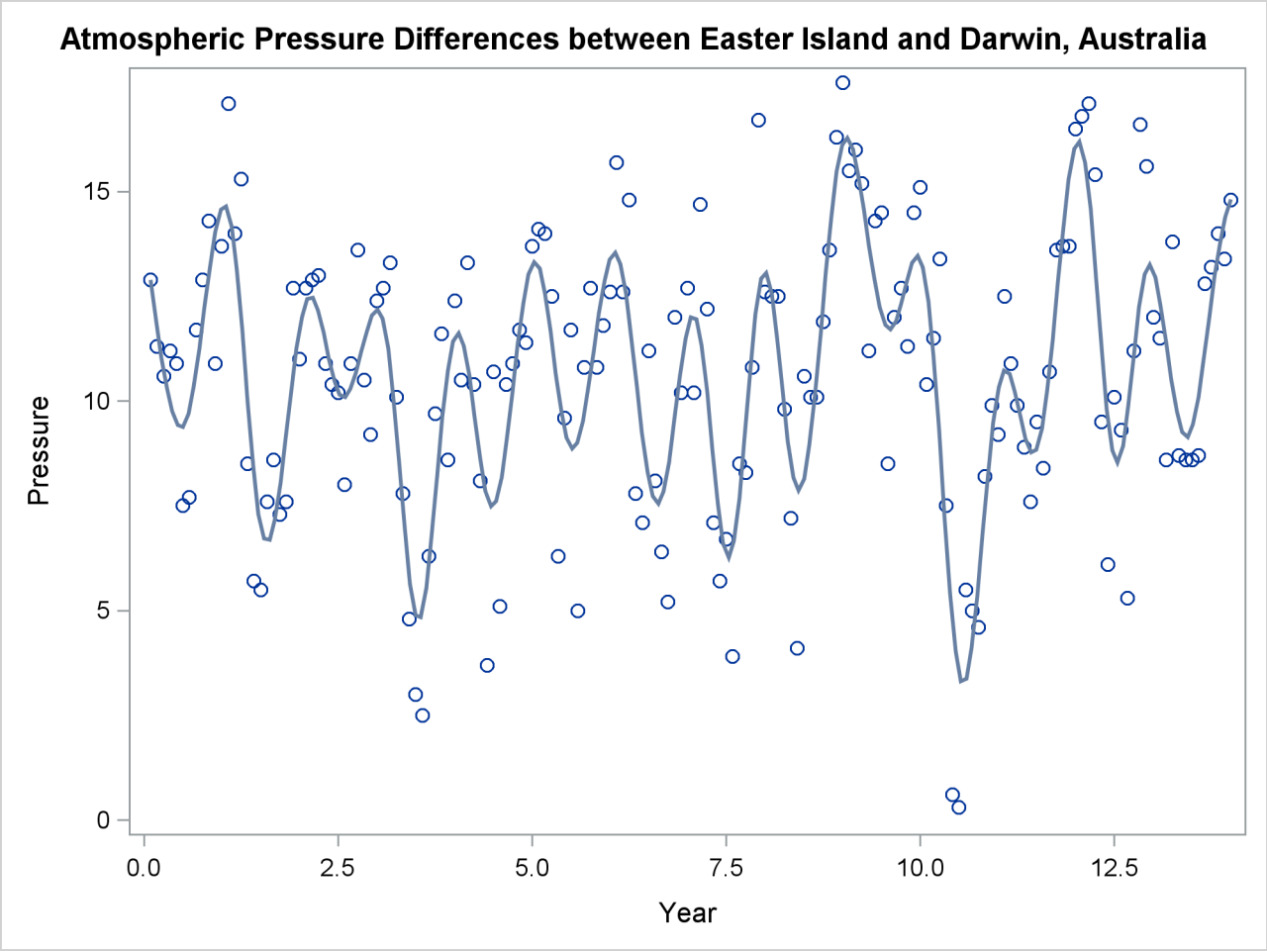
The SGPLOT Procedure

PROC SGPLOT provides a simple way to make a variety of scatter plots. This example is taken from Example 52.4 of Chapter 52, "The LOESS Procedure." The ENSO data set, which contains information about differences in ocean pressure over time, is available from the Sashelp library.

The following statements create a scatter plot of points along with a penalized B-spline fit to the data and produce Figure 21.55:

```
proc sgplot data=sashelp.enso noautolegend;
   title 'Atmospheric Pressure Differences between '
         'Easter Island and Darwin, Australia';
   pbspline y=pressure x=year;
run;
```

Figure 21.55 Penalized B-Spline Fit with PROC SGPLOT



See Chapter 93, "The TRANSREG Procedure," for more information about penalized B-splines. Also see the section "Grouped Scatter Plot with PROC SGPLOT" on page 608 and Figure 21.12 for an example of a scatter plot with groups of observations.

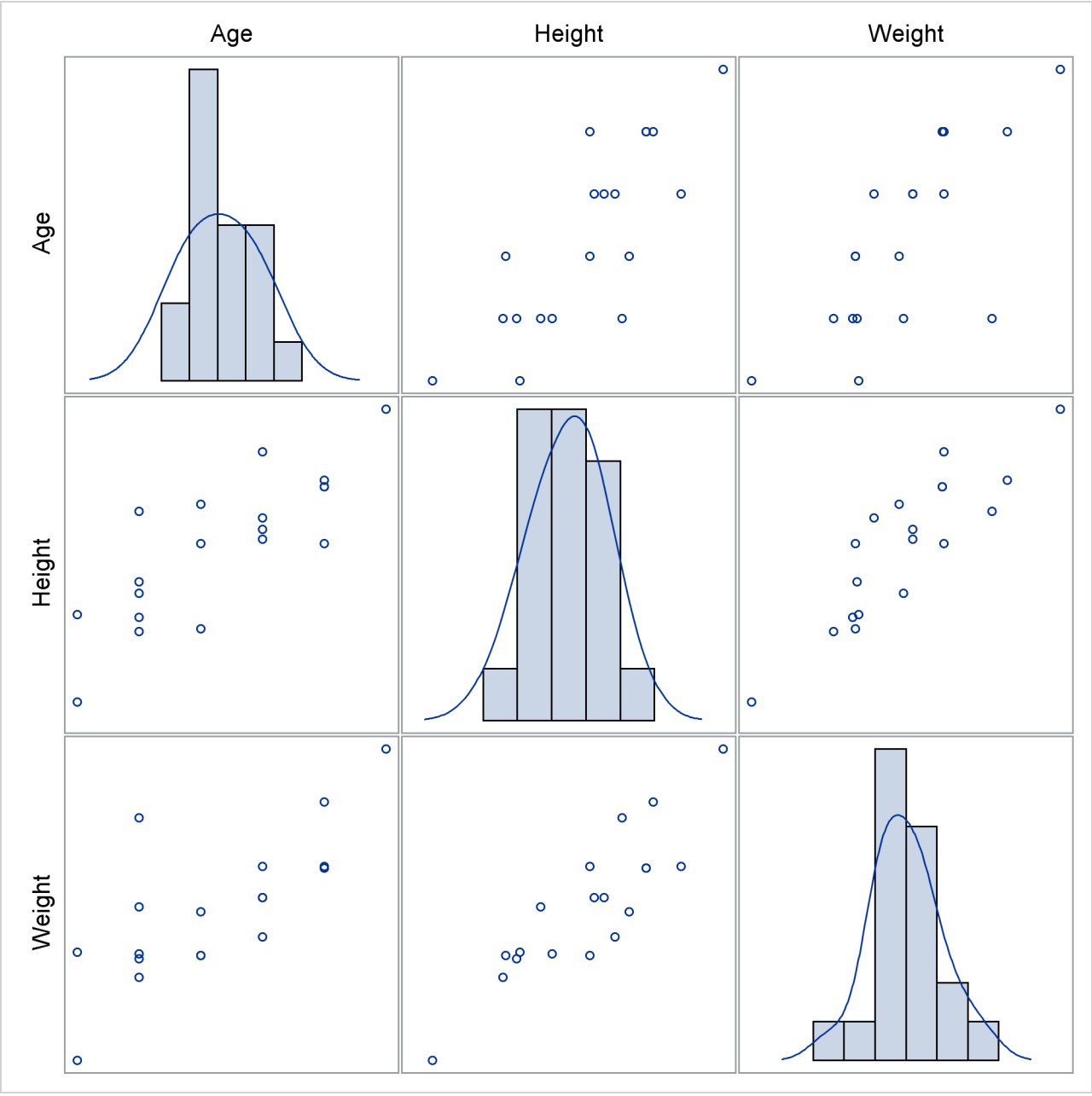
The SGSCATTER Procedure

You can use the SGSCATTER procedure to produce scatter plot matrices. The following step creates a scatter plot matrix from all of the numeric variables in the Class data set (available in the Sashelp library) and produces Figure 21.56:

```
proc sgscatter data=sashelp.class;
   matrix _numeric_ / diagonal=(kernel histogram);
run;
```

The diagonal cells of Figure 21.56 contain a histogram and a kernel density fit. The off-diagonal cells contain all pairs of scatter plots.

Figure 21.56 Scatter Plot Matrix with PROC SGSCATTER



The MATRIX statement creates a symmetric $n \times n$ scatter plot matrix. Other statements are also available. The PLOT statement creates a panel that contains one or more individual scatter plots. The COMPARE statement creates a rectangular $m \times n$ scatter plot matrix. Linear and nonlinear fits can be added, and many graphical features can be requested with options.

The SGPANEL Procedure

The SGPANEL procedure creates paneled plots and charts with one or more classification variables. Classification variables can be designated as row or column variables, or there can be multiple classifications. Graphs are drawn for each combination of the levels of classification variables, showing a subset of the data in each cell.

This example is taken from Example 40.6 of Chapter 40, "The GLIMMIX Procedure." The following statements create the input SAS data sets:

```
data times;
   input time1-time23;
   datalines;
122  150  166  179  219  247  276  296  324  354  380  445
478  508  536  569  599  627  655  668  723  751  781
;

data cows;
   if _n_ = 1 then merge times;
   array t{23} time1 - time23;
   array w{23} weight1 - weight23;
   input cow iron infection weight1-weight23 @@;
   do i=1 to 23;
      weight = w{i};
      tpoint = (t{i}-t{1})/10;
      output;
   end;
   keep cow iron infection tpoint weight;
   datalines;
 1 0 0  4.7    4.905  5.011  5.075  5.136  5.165  5.298  5.323
        5.416  5.438  5.541  5.652  5.687  5.737  5.814  5.799
        5.784  5.844  5.886  5.914  5.979  5.927  5.94

   ... more lines ...

;
```

First, PROC GLIMMIX is run to fit the model, and then the results are prepared for plotting:

```
proc glimmix data=cows;
  t2 = tpoint / 100;
  class cow iron infection;
  model weight = iron infection iron*infection tpoint;
  random t2 / type=rsmooth subject=cow
                  knotmethod=kdtree(bucket=100 knotinfo);
  output out=gmxout pred(blup)=pred;
  nloptions tech=newrap;
run;

data plot;
  set gmxout;
  length Group $ 26;
```

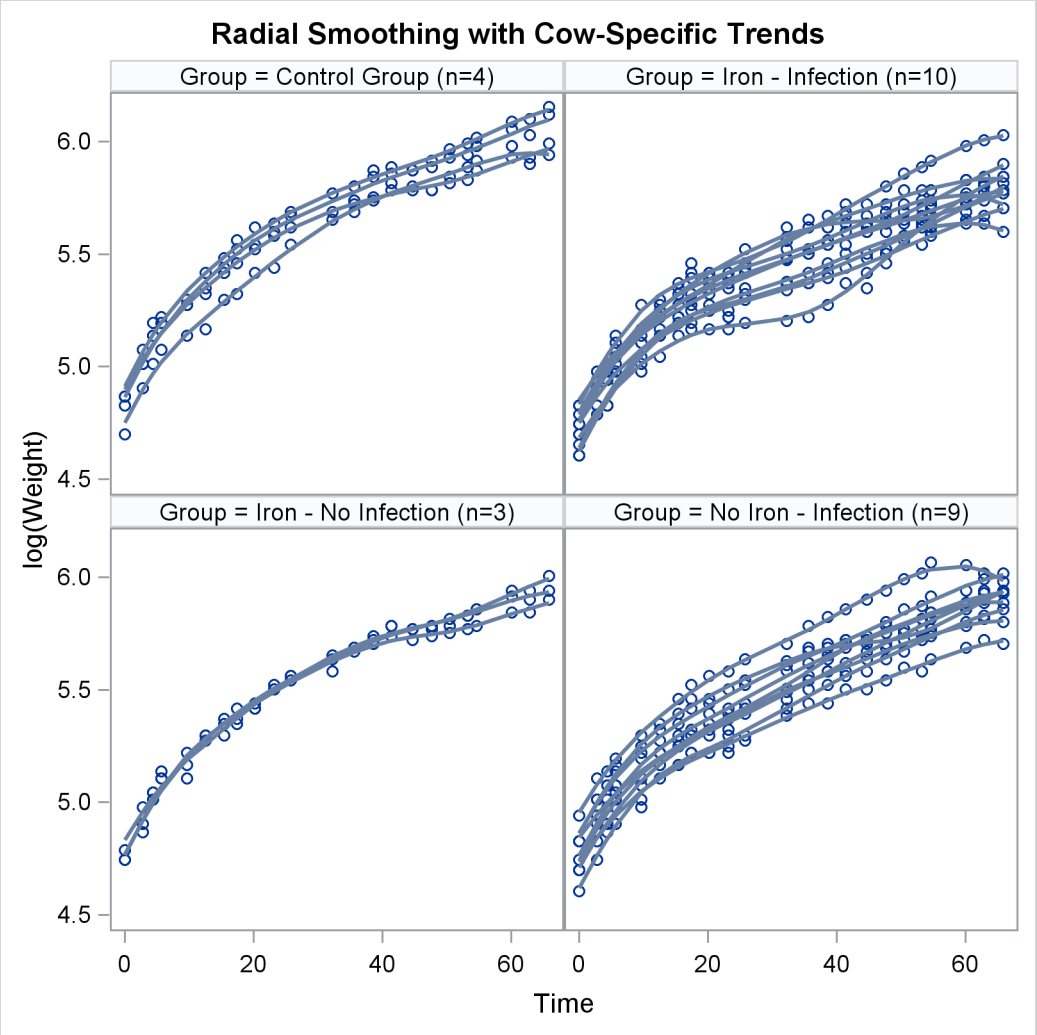
```
      if (iron=0) and (infection=0) then group='Control Group (n=4)';
  else if (iron=1) and (infection=0) then group='Iron - No Infection (n=3)';
  else if (iron=0) and (infection=1) then group='No Iron - Infection (n=9)';
  else group = 'Iron - Infection (n=10)';
run;

proc sort data=plot; by group cow;
run;
```

The following statements produce graphs of the observed data and fitted profiles in the four groups:

```
proc sgpanel data=plot noautolegend;
  title 'Radial Smoothing with Cow-Specific Trends';
  label tpoint='Time' weight='log(Weight)';
  panelby group / columns=2 rows=2;
  scatter x=tpoint y=weight;
  series  x=tpoint y=pred / group=cow lineattrs=GraphFit;
run;
```

The results are shown in Figure 21.57.

Figure 21.57 Fit Using PROC SGPANEL



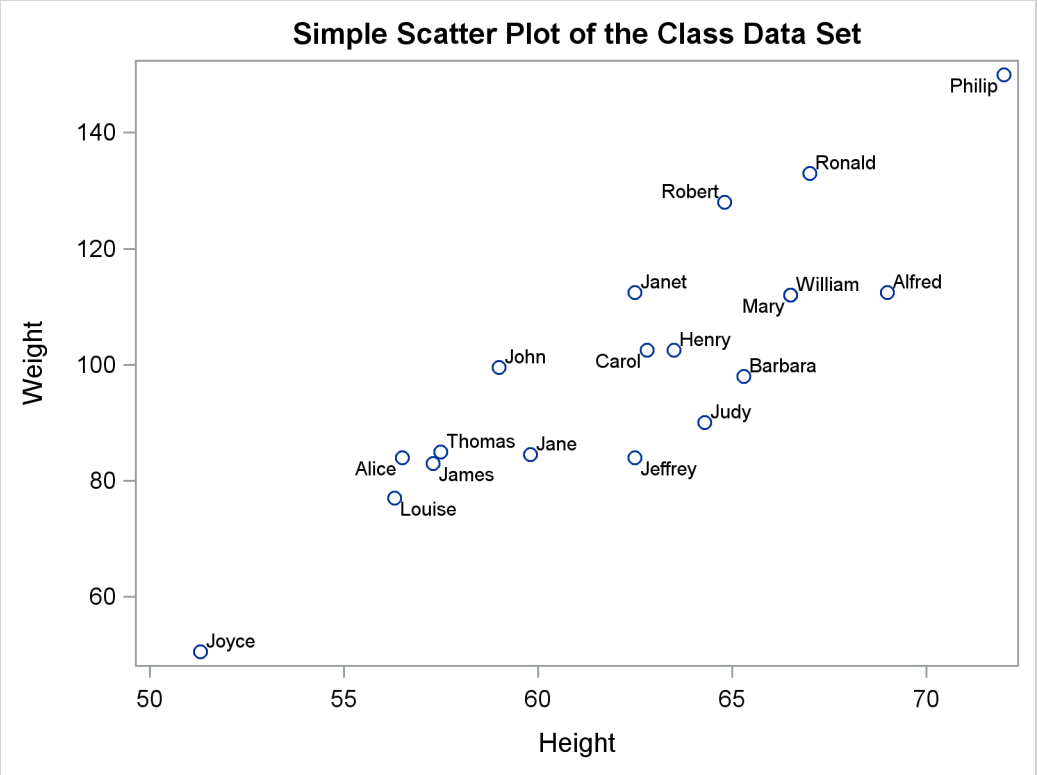
The SGRENDER Procedure

The SGRENDER procedure produces a graph from an input SAS data set and an ODS graph template. With PROC SGRENDER and the Graph Template Language (GTL), you can create highly customized graphs. The following steps create a simple scatter plot of the Class data set (available in the Sashelp library) and produce Figure 21.58:

```
proc template;
   define statgraph Scatter;
      begingraph;
         entrytitle "Simple Scatter Plot of the Class Data Set";
         layout overlay;
            scatterplot y=weight x=height / datalabel=name;
         endlayout;
      endgraph;
   end;
run;

proc sgrender data=sashelp.class template=scatter;
run;
```

The template definition consists of an outer block that begins with a DEFINE statement and ends with an END statement. Inside of that is a BEGINGRAPH/ENDGRAPH block. Inside that block, the ENTRYTITLE statement provides the plot title, and the LAYOUT OVERLAY block contains the statement or statements that define the graph. In this case, there is just a single SCATTERPLOT statement that names the Y-axis (vertical) variable, the X-axis (horizontal) variable, and an optional variable that contains labels for the points. The PROC SGRENDER statement simply specifies the input data set and the template. The real work in using PROC SGRENDER is writing the template.

Figure 21.58 Scatter Plot of Labeled Points with PROC SGRENDER



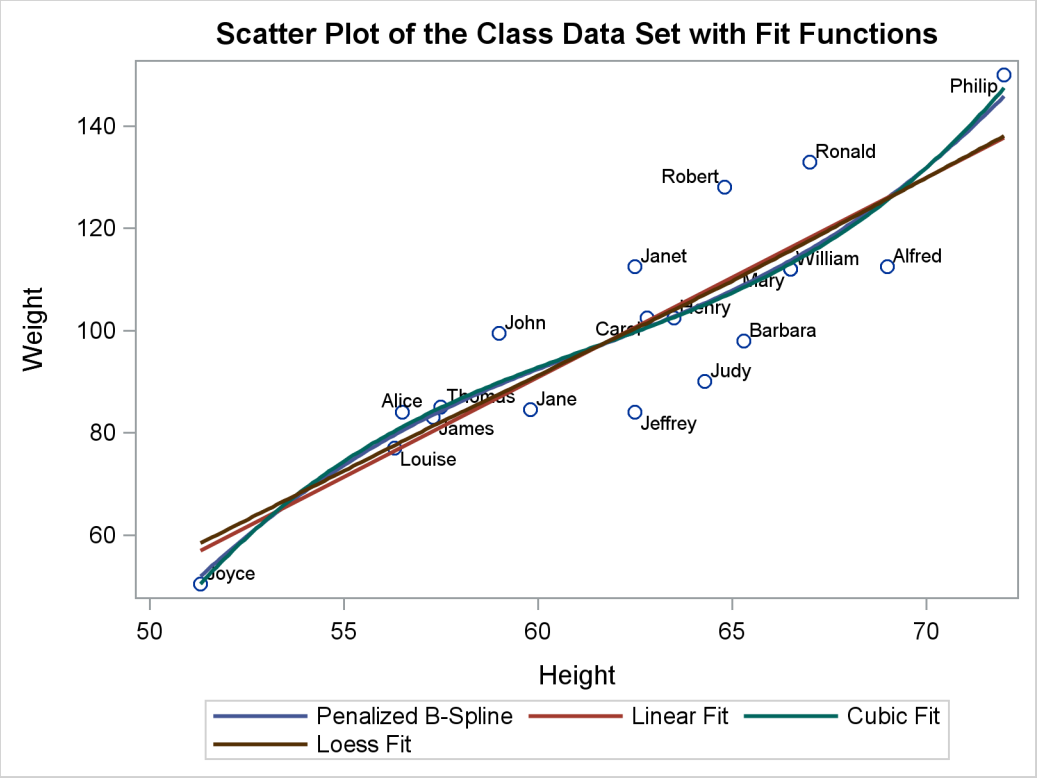
The following steps add a series of fit functions to the scatter plot and create a legend by adding statements to the **Scatter** template:

```
proc template;
   define statgraph Scatter;
      begingraph;
         entrytitle "Scatter Plot of the Class Data Set with Fit Functions";
         layout overlay;
            scatterplot y=weight x=height / datalabel=name;
            pbsplineplot y=weight x=height / name='pbs'
                         legendlabel='Penalized B-Spline'
                         lineattrs=GraphData1;
            regressionplot y=weight x=height / degree=1 name='line'
                         legendlabel='Linear Fit'
                         lineattrs=GraphData2;
            regressionplot y=weight x=height / degree=3 name='cubic'
                         legendlabel='Cubic Fit'
                         lineattrs=GraphData3;
            loessplot y=weight x=height / name='loess'
                         legendlabel='Loess Fit'
                         lineattrs=GraphData4;
            discretelegend 'pbs' 'line' 'cubic' 'loess';
         endlayout;
      endgraph;
   end;
run;
```

```
proc sgrender data=sashelp.class template=scatter;
run;
```

The line attributes for each function are specified with different style elements, **GraphData1** through **GraphData4**, so that the functions are adequately identified in the legend. The preceding statements create Figure 21.59.

Figure 21.59 Scatter Plot and Fit Functions with PROC SGRENDER

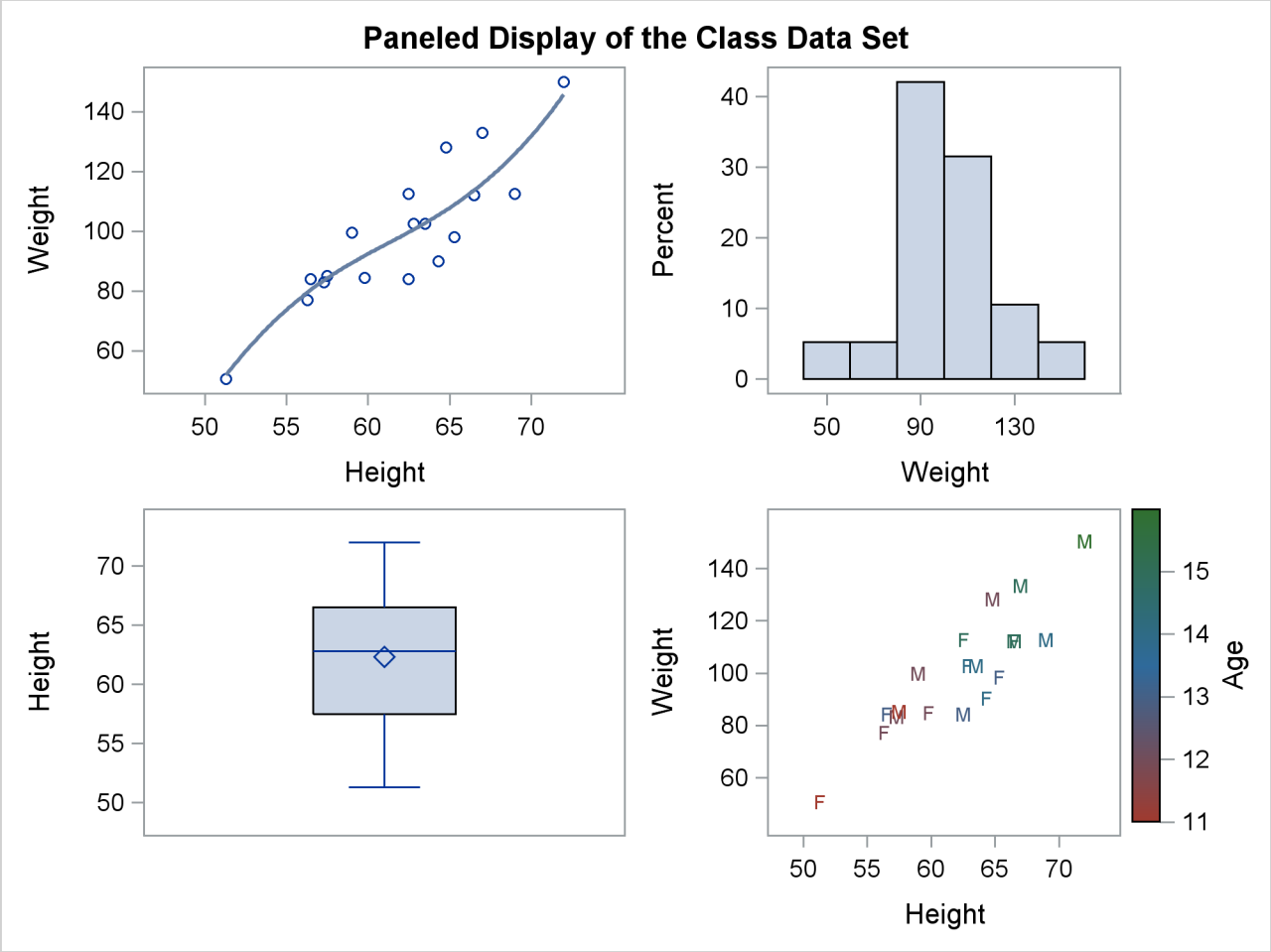


The following statements create a four-panel display of the Class data set and produce Figure 21.60:

```
proc template;
   define statgraph Panel;
      begingraph;
         entrytitle "Paneled Display of the Class Data Set";

         layout lattice / rows=2 columns=2 rowgutter=10 columngutter=10;

            layout overlay;
               scatterplot y=weight x=height;
               pbsplineplot y=weight x=height;
            endlayout;

            layout overlay / xaxisopts=(label='Weight');
               histogram weight;
            endlayout;

            layout overlay / yaxisopts=(label='Height');
               boxplot y=height;
            endlayout;

            layout overlay / xaxisopts=(offsetmin=0.1 offsetmax=0.1)
                             yaxisopts=(offsetmin=0.1 offsetmax=0.1);
               scatterplot  y=weight x=height / markercharacter=sex
                  name='color' markercolorgradient=age;
               continuouslegend 'color'/ title='Age';
            endlayout;

         endlayout;
      endgraph;
   end;
run;

proc sgrender data=sashelp.class template=panel;
run;
```

In this template, the outermost layout is a LAYOUT LATTICE. It creates a 2×2 panel of plots with a 10-pixel separation (or gutter) between each plot. Inside the lattice are four LAYOUT OVERLAY blocks—each defining one of the graphs. The first is a simple scatter plot with a nonlinear penalized B-spline fit. The second is a histogram of the dependent variable Weight. The third is a box plot of the independent variable Height. The fourth simultaneously shows height, weight, age, and sex for the students in the class. Each axis has an offset added at both the maximum and minimum. This provides padding between the axes and the data.

Figure 21.60 Multiple Panels Using PROC SGRENDER



Many other types of graphs are available with the SG procedures. However, even the few examples provided here show the power and flexibility available for making professional-quality statistical graphics. See the *SAS Graph Template Language: User's Guide* and the *SAS ODS Graphics: Procedures Guide* for more information.

Examples of ODS Statistical Graphics

Example 21.1: Creating Graphs with Tool Tips in HTML

This example demonstrates how to request graphs in HTML that are enhanced with tooltip displays, which appear when you move a mouse over certain features of the graph. When you specify the HTML destination and the IMAGEMAP=ON option in the ODS GRAPHICS statement, an image map of coordinates for tooltips is generated along with the HTML output file. Individual graphs are saved as PNG files.

Example 58.2 and Example 58.8 of Chapter 58, "The MIXED Procedure," analyze a data set with repeated growth measurements for 27 children. The following step creates the data set:

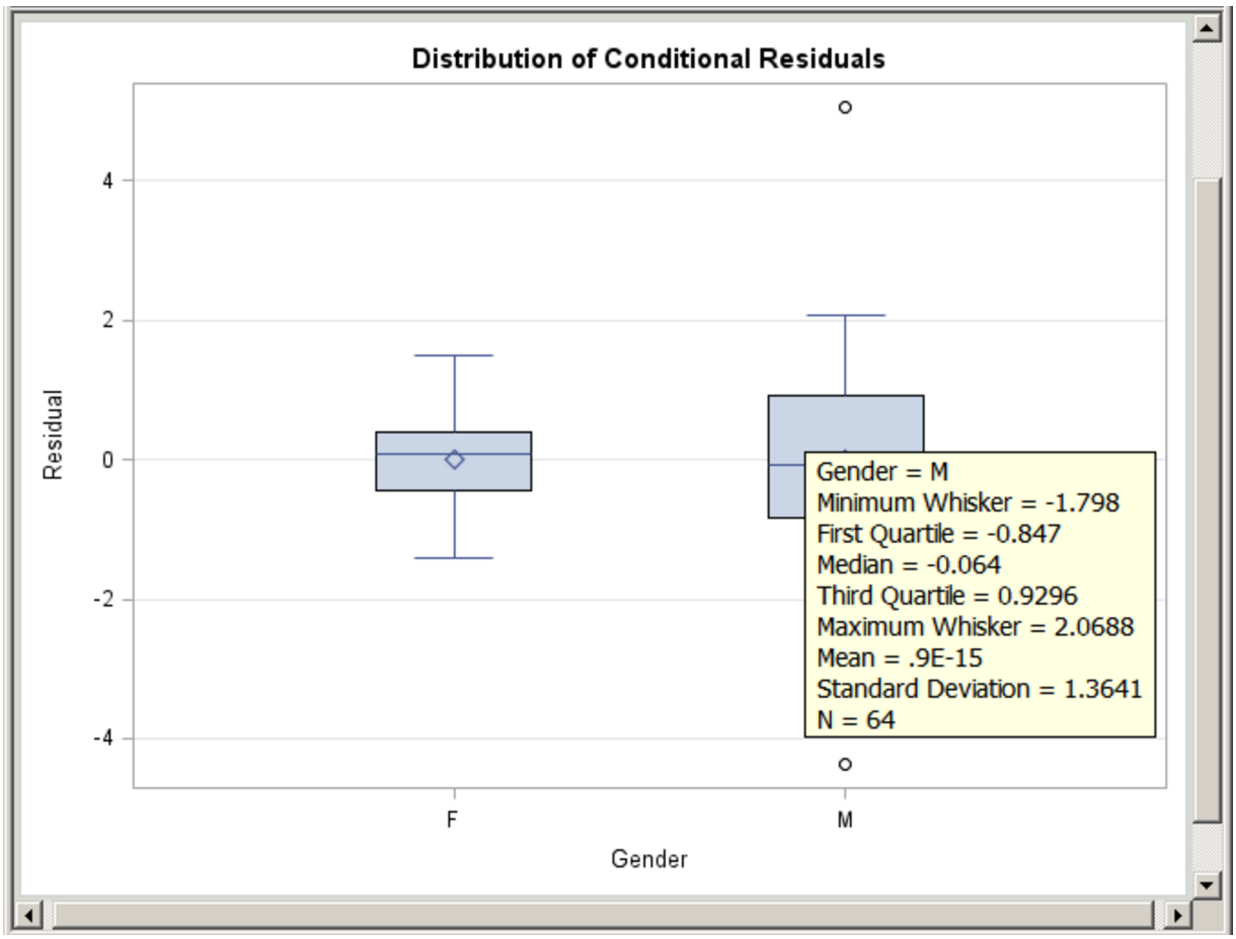
```
data pr;
   input Person Gender $ y1 y2 y3 y4 @@;
   y=y1; Age=8;  output;
   y=y2; Age=10; output;
   y=y3; Age=12; output;
   y=y4; Age=14; output;
   drop y1-y4;
   datalines;
 1  F  21.0  20.0  21.5  23.0      2  F  21.0  21.5  24.0  25.5

   ... more lines ...

;
```

The following statements fit a mixed model with random intercepts and slopes for each child:

```
ods _all_ close;
ods graphics on / imagemap=on;
ods html body='b.html' style=HTMLBlue;

proc mixed data=pr method=ml plots=boxplot;
   ods select 'Conditional Residuals by Gender';
   class Person Gender;
   model y = Gender Age Gender*Age;
   random intercept Age / type=un subject=Person;
run;

ods html close;
```

The PLOTS=BOXPLOT option in the PROC MIXED statement requests box plots of observed values and residuals for each classification main effect in the model (Gender and Person). Only the by-gender box plots are actually created due to the ODS SELECT statement, which uses the plot label to select the plot. Output 21.1.1 displays the results. Moving the mouse over a box plot displays a tooltip with summary statistics for the class level. Graphics with tooltips are supported for only the HTML destination.

Output 21.1.1 Box Plot with Tool Tips



Example 21.2: Creating Graphs for a Presentation

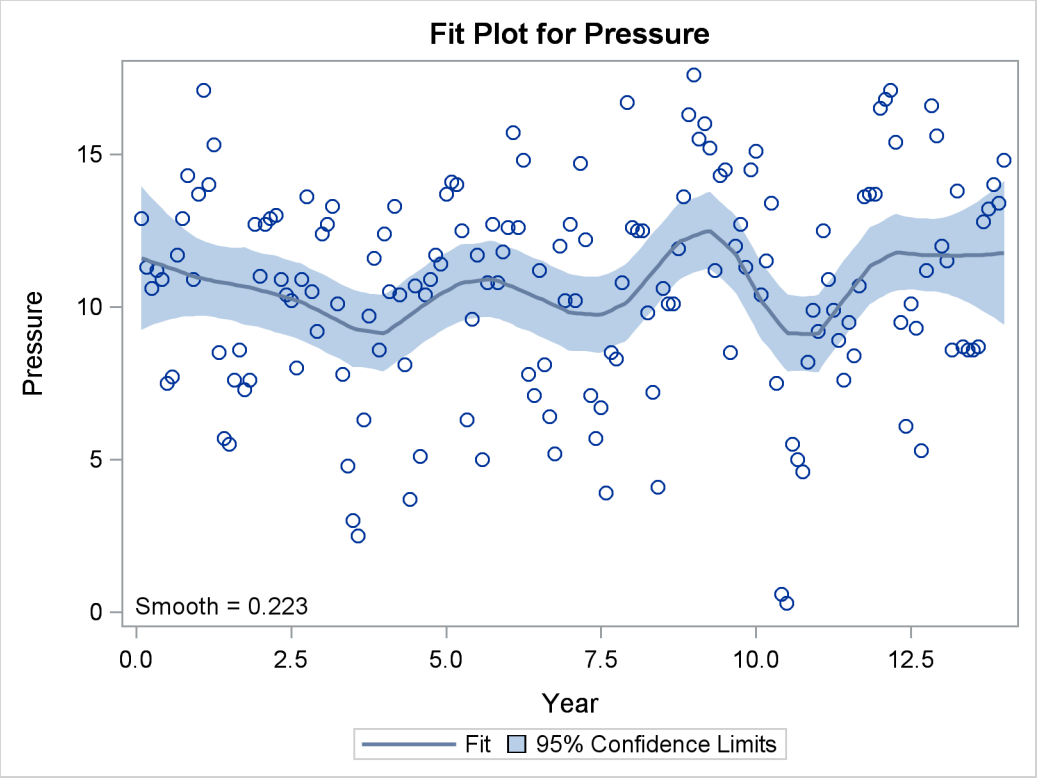
The RTF destination provides an easy way to create graphs for inclusion into a paper or presentation. You can specify the ODS RTF statement to create a file that is easily imported into a document (such as Microsoft Word or WordPerfect) or a presentation (such as Microsoft PowerPoint).

The following statements request a loess fit and save the output in the file *loess.rtf*:

```
ods _all_ close;
ods rtf file="loess.rtf" style=HTMLBlue;
ods graphics on;

proc loess data=sashelp.enso;
   model pressure = year / clm residual;
run;

ods rtf close;
ods listing;
```

The output file includes various tables and the following plots: a plot of the selection criterion versus smoothing parameter, a fit plot with 95% confidence bands, a plot of residual by regressors, and a diagnostics panel. The fit plot is produced with the HTMLBLUE style and is shown in Output 21.2.1.

Output 21.2.1 Loess Fit Plot with the HTMLBLUE Style



If you are running the SAS System in the Microsoft Windows operating system, you can open the RTF file in Microsoft Word and simply copy and paste the graphs into Microsoft PowerPoint. In general, RTF output is convenient for exchange of graphical results between Microsoft Windows applications through the clipboard.

Alternatively, if you use the LISTING or HTML destinations, then your individual graphs are created as PNG files by default. You can insert these files into a Microsoft PowerPoint presentation. See the sections "Naming Graphics Image Files" on page 634 and "Saving Graphics Image Files" on page 636 for information about how the image files are named and saved.

Example 21.3: Creating Graphs in PostScript Files

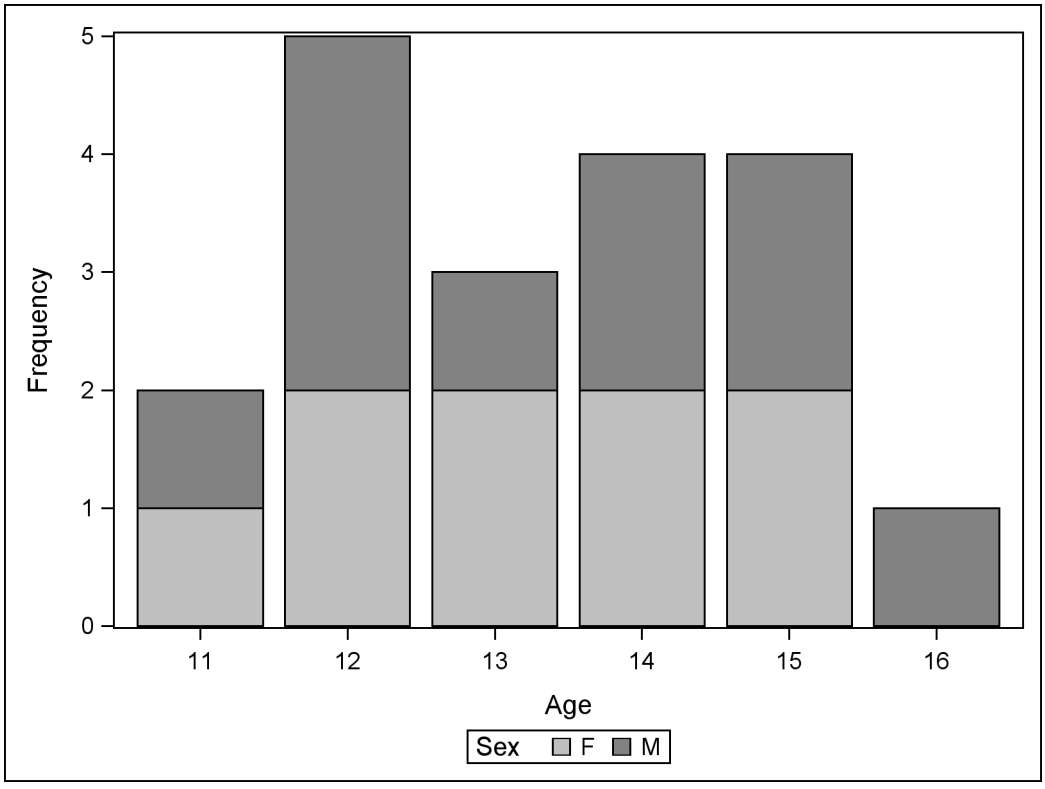
This example illustrates how to create individual graphs in PostScript files. This is particularly useful when you want to include them in a LaTeX document.

The following statements close all open destinations, open the LATEX destination with the JOURNAL style, and request a grouped bar chart for the Sashelp.Class data set:

```
ods graphics on / reset=index;
ods _all_ close;
ods latex style=Journal;

proc sgplot data=sashelp.class;
   vbar age / group=sex;
   run;

ods latex close;
ods listing;
```

The JOURNAL style displays gray-scale graphs that are suitable for a journal. When you specify the ODS LATEX destination, ODS creates a PostScript file for each individual graph in addition to a LaTeX source file that includes the tabular output and references to the PostScript files. By default, these files are saved in the SAS current folder. The bar chart shown in Output 21.3.1 is saved by default in a file named *SGPlot.ps*. See the section "Naming Graphics Image Files" on page 634 for details about how graphics image files are named. If both the default destination (LISTING or HTML) and the LATEX destination are open, then two files are created: *SGPlot.png* and *SGPlot1.ps*. If the RESET=INDEX option is not specified in the ODS GRAPHICS statement and you run the step again, the next names are based on an incremented index (*SGPlot2.png* and *SGPlot3.ps*).

Output 21.3.1 Bar Chart Using the JOURNAL Style

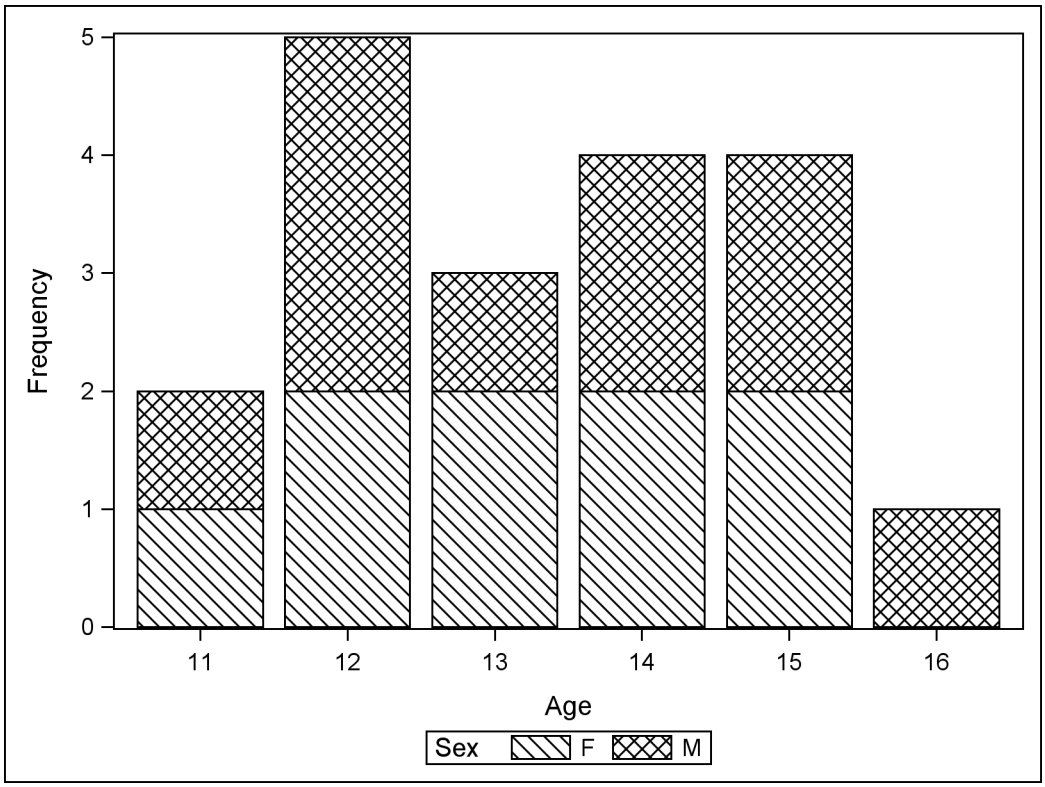


You can use the JOURNAL2 style for a different appearance—the bars are not shaded. Crosshatching is used to indicate group membership. The following step produces Output 21.3.2:

```
ods graphics on / reset=index;
```

```
ods _all_ close;
ods latex style=Journal2;

proc sgplot data=sashelp.class;
   vbar age / group=sex;
   run;

ods latex close;
ods listing;
```

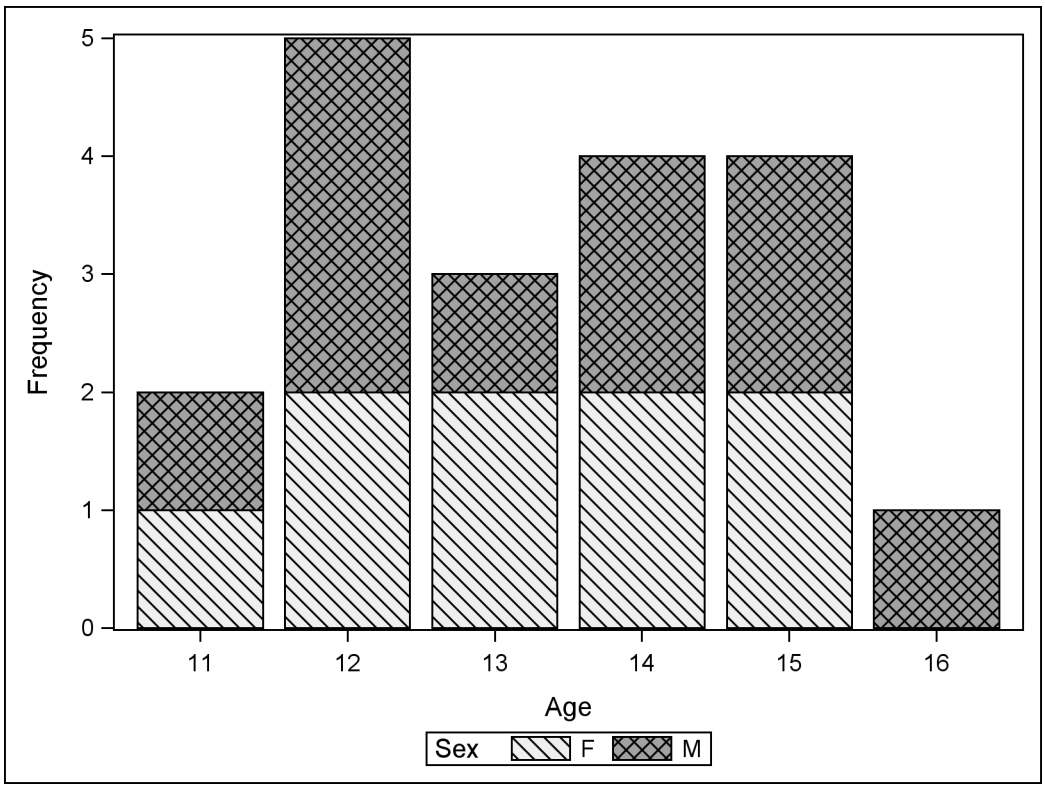
Output 21.3.2 Bar Chart Using the JOURNAL2 Style



You can use the JOURNAL3 style for a different kind of appearance from the JOURNAL style. A mix of filled areas and crosshatching is used in grouped bar charts. The following step produces Output 21.3.3:

```
ods graphics on / reset=index;
ods _all_ close;
ods latex style=Journal3;

proc sgplot data=sashelp.class;
   vbar age / group=sex;
   run;

ods latex close;
ods listing;
```

Output 21.3.3 Bar Chart Using the JOURNAL3 Style



If you are writing a paper, you can include the graphs in your own LaTeX source file by referencing the names of the individual PostScript graphics files. In this situation, you might not find it necessary to use the LaTeX source file created by the SAS System. Alternatively, you can include PNG files into a LaTeX document, after using some other ODS destination (such as HTML) to create the PNG files.

Example 21.4: Displaying Graphs Using the DOCUMENT Procedure

This example illustrates the use of the ODS DOCUMENT destination and the DOCUMENT procedure to display your ODS graphs. You can use this approach whenever you want to generate and save your output (both tables and graphs) and then display or replay it later, potentially in subsets or more than once. This approach is particularly useful when you want to display your output in multiple ODS destinations, or when you want to use different styles without rerunning your SAS program. This approach is also useful when you want to break your output into separate parts for inclusion into different parts of a document such as a LaTeX file.

Consider again the data set Stack created by the following statements:

```
data stack;
   input x1 x2 x3 y @@;
   datalines;
80 27 89 42   80 27 88 37   75 25 90 37   62 24 87 28   62 22 87 18
62 23 87 18   62 24 93 19   62 24 93 20   58 23 87 15   58 18 80 14
58 18 89 14   58 17 88 13   58 18 82 11   58 19 93 12   50 18 89  8
50 18 86  7   50 19 72  8   50 19 79  8   50 20 80  9   56 20 82 15
70 20 91 15
;
```

The following statements request a Q-Q plot from PROC ROBUSTREG with the Stack data:

```
ods _all_ close;
ods document name=QQDoc(write);

proc robustreg data=stack plots=qqplot;
   model y = x1 x2 x3;
run; quit;

ods document close;
ods listing;
```

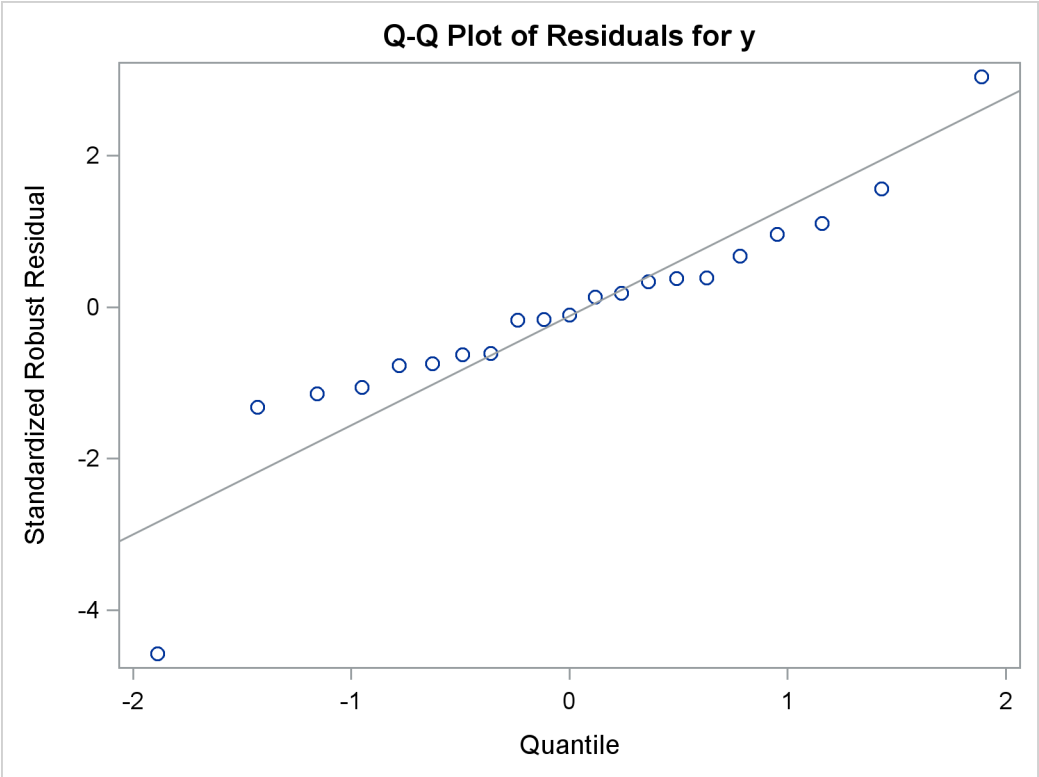
The ODS DOCUMENT statement opens an ODS document named **QQDoc**. All of the results—tables, graphs, titles, notes, footnotes, headers—are stored in the ODS document. None of them are displayed since no other destination is open. In order to display the Q-Q plot with PROC DOCUMENT, you first need to determine its name. You can do this by specifying the ODS TRACE ON statement prior to the procedure statements (see the section "Determining Graph Names and Labels" on page 628 for more information). Alternatively, you can type **odsdocuments** (or **odsd** for short) on the command line to open the Documents window, which you can then use to manage your ODS documents.

The following statements specify an HTML destination and display the residual Q-Q plot by using the REPLAY statement in PROC DOCUMENT:

```
ods html body='b.htm';

proc document name=QQDoc;
   ods select QQPlot;
   replay;
run; quit;

ods html close;
```

Subsequent steps can replay one or more objects from the same ODS document. By default, the REPLAY statement attempts to display every output object stored in the ODS document, but here only the Q-Q plot is displayed because it is specified by the ODS SELECT statement. The plot is displayed in Output 21.4.1.

Output 21.4.1 Q-Q Plot Displayed by PROC DOCUMENT



As an alternative to running PROC DOCUMENT with an ODS SELECT statement, you can run PROC DOCUMENT with a *document path* for the Q-Q plot in the REPLAY statement. This approach is preferable when the ODS document contains a large volume of output, so that PROC DOCUMENT does not attempt to process every piece of output stored in the ODS document.

You can determine the ODS document path for the Q-Q plot by specifying the LIST statement with the LEVELS=ALL option in PROC DOCUMENT as follows:

```
proc document name=QQDoc;
   list / levels=all;
run; quit;
```

The contents of the ODS document **QQDoc** are shown in Output 21.4.2.

Output 21.4.2 Contents of the ODS Document `QQDoc`

```
Listing of: \Work.Qqdoc\
Order by: Insertion
Number of levels: All

  Obs    Path                                                                  Type
-------------------------------------------------------------------------------------
       1 \Robustreg#1                                                          Dir
       2 \Robustreg#1\ModelInfo#1                                              Table
       3 \Robustreg#1\NObs#1                                                   Table
       4 \Robustreg#1\ParmInfo#1                                               Table
       5 \Robustreg#1\SummaryStatistics#1                                      Table
       6 \Robustreg#1\ParameterEstimates#1                                     Table
       7 \Robustreg#1\DiagSummary#1                                            Table
       8 \Robustreg#1\DiagnosticPlots#1                                        Dir
       9 \Robustreg#1\DiagnosticPlots#1\QQPlot#1                               Graph
      10 \Robustreg#1\GoodFit#1                                                Table
```

The ODS document path of the `QQPlot` entry in the `QQDoc` ODS document, as shown in Output 21.4.2, is `\Robustreg#1\DiagnosticPlots#1\QQPlot#1`.

You can use this path to display the residual Q-Q plot with PROC DOCUMENT as follows:

```
proc document name=QQDoc;
   replay \Robustreg#1\DiagnosticPlots#1\QQPlot#1;
run; quit;
```

You can also determine the ODS document path from the Results window or the Documents window. Right-click the object icon and select **Properties**.

The SAS/STAT documentation preparation process uses the ODS document. SAS output is saved into an ODS document that is then replayed into sections of the documentation, which is prepared using LaTeX. In general, when you send your output to the DOCUMENT destination, you can use PROC DOCUMENT to rearrange, duplicate, or remove output from the results of a procedure or a database query. For more information, see the ODS DOCUMENT statement in the section "Dictionary of ODS Language Statements" and the chapter "The DOCUMENT Procedure" in the *SAS Output Delivery System: User's Guide*.

Example 21.5: Customizing the Style for Box Plots

This example demonstrates how to modify the style for box plots. This example is taken from Example 21.1. The following step creates the data set:

```
data pr;
   input Person Gender $ y1 y2 y3 y4 @@;
   y=y1; Age=8;  output;
   y=y2; Age=10; output;
   y=y3; Age=12; output;
   y=y4; Age=14; output;
   drop y1-y4;
   datalines;
 1  F  21.0  20.0  21.5  23.0      2  F  21.0  21.5  24.0  25.5

   ... more lines ...

;
```

The following step displays the HTMLBLUE style and its parent styles, STATISTICAL and DEFAULT:

```
proc template;
   source Styles.HTMLBlue;
   source Styles.Statistical;
   source Styles.Default;
run;
```

If you search for 'box', you find the style element that controls some aspects of the box plot:

```
 class GraphBox /
    capstyle = "serif"
    connect = "mean"
    displayopts = "fill caps median mean outliers";
```

You can learn more about the **GraphBox** style element and its attributes in the section on the BOXPLOT statement in the *SAS Graph Template Language: Reference Guide* and in the section on "ODS Style Elements" in the *SAS Output Delivery System: User's Guide*.

The following statements create two new styles by modifying attributes of the **GraphBox** style element. The first style is a sparse style; the box is outlined (not filled), and the median is shown but not the mean. In contrast, the second style produces a filled box, with caps on the whiskers that shows the mean, median, and outliers. In addition, the box is notched.

The following statements create the two styles:

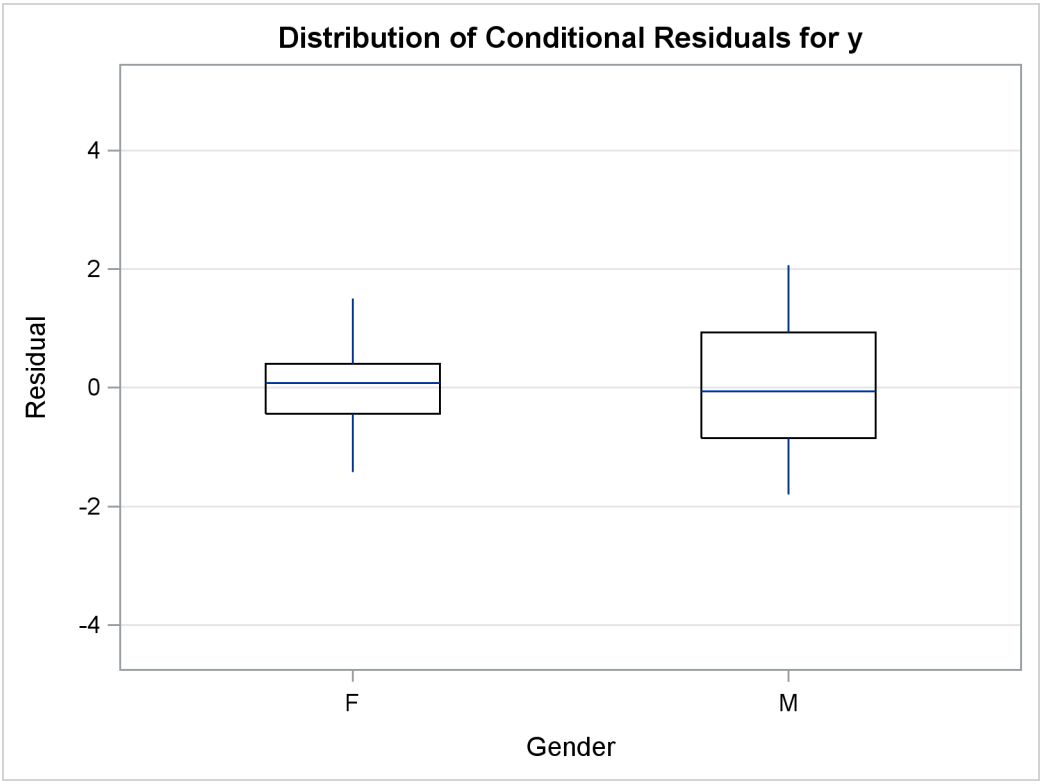
```
proc template;
   define style BoxStyleSparse;
      parent=styles.HTMLBlue;
      style GraphBox / capstyle = "line" displayopts = "median";
   end;
   define style BoxStyleRich;
      parent=styles.HTMLBlue;
      style GraphBox / capstyle = "bracket"
            displayopts = "fill caps median mean outliers notches";
   end;
run;
```

The following steps run PROC MIXED and create box plots that use the two styles:

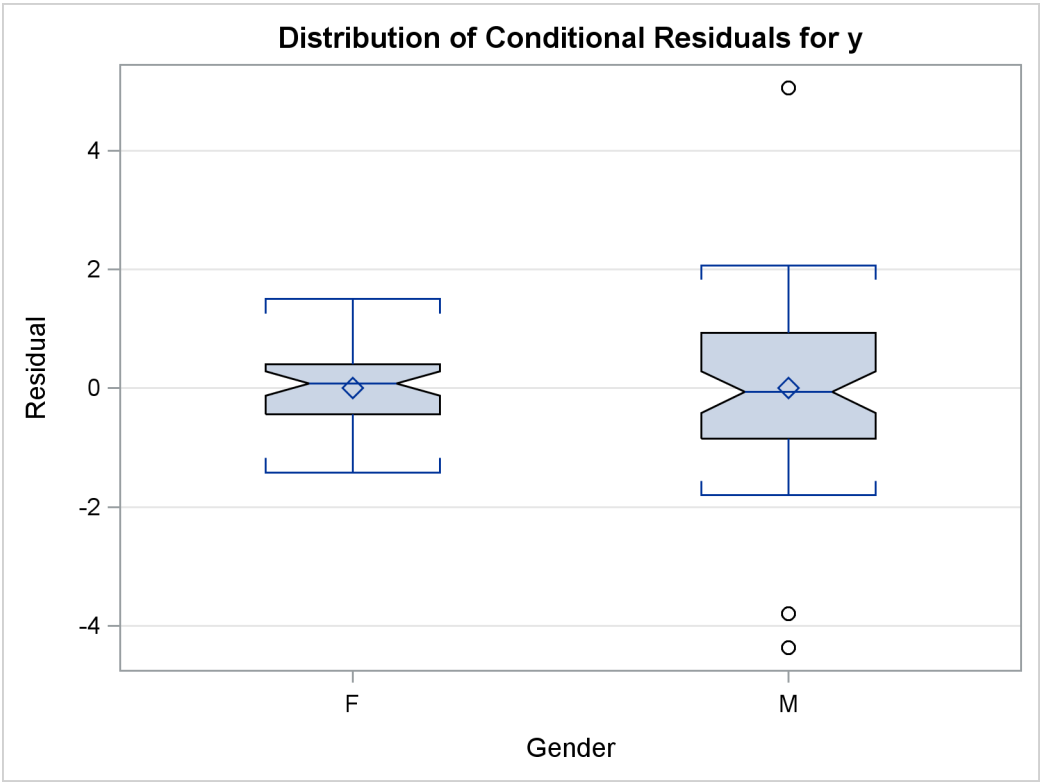
```
ods graphics on;
ods listing style=boxstylesparse;

proc mixed data=pr method=ml plots=boxplot;
   ods select 'Conditional Residuals by Gender';
   class Person Gender;
   model y = Gender Age Gender*Age;
   random intercept Age / type=un subject=Person;
run;

ods listing style=boxstylerich;

proc mixed data=pr method=ml plots=boxplot;
   ods select 'Conditional Residuals by Gender';
   class Person Gender;
   model y = Gender Age Gender*Age;
   random intercept Age / type=un subject=Person;
run;
```

The results with the sparse style are displayed in Output 21.5.1, and the results with the richer style are displayed in Output 21.5.2. See Output 21.1.1 in Example 21.1 to see the results of using the HTMLBLUE style.

Output 21.5.1 Box Plot with the Sparse Style



Output 21.5.2 Box Plot with the Richer Style



References

Kuhfeld, W. F. (2009), "Modifying ODS Statistical Graphics Templates in SAS 9.2," http://support.sas.com/rnd/app/papers/modtmplt.pdf.

Kuhfeld, W. F. (2010), *Statistical Graphics in SAS: An Introduction to the Graph Template Language and the Statistical Graphics Procedures*, Cary, NC: SAS Press.

Subject Index

S

T

Syntax Index

Your Turn

We welcome your feedback.

- If you have comments about this book, please send them to **yourturn@sas.com**. Include the full title and page numbers (if applicable).
- If you have comments about the software, please send them to **suggest@sas.com**.