Paper 193-2007

SAS/GRAPH® Procedures for Creating Statistical Graphics in Data Analysis

Dan Heath, SAS Institute Inc., Cary, NC

ABSTRACT

Making a plot of the data is often the first step in a data analysis or a statistical study. In SAS® 9.2, SAS/GRAPH introduces a family of new procedures developed for this purpose. These procedures are designed to create stand-alone displays that complement the more specialized graphs produced by the statistical procedures that use the ODS Statistical Graphics infrastructure. Since these new procedures and the statistical procedures use the same infrastructure, graphs created with all of these procedures are consistent in their appearance and in the way they are produced and managed.

The new “SG” family of procedures includes SGPLOT, SGPANEL, SGSCATTER, and SGRENDER. The SGPLOT procedure creates single-celled plots that can be constructed with a variety of plot types, along with data smoothers, fitted curves, and other statistical features. The SGPANEL procedure creates paneled plots and charts where the paneling is driven by classification variables. The SGSCATTER procedure creates paneled scatter plots and matrices with support for fitted lines, confidence bands, and computed ellipses. These three procedures are designed with a syntax that is powerful yet concise. The SGRENDER procedure is unique in that it creates customized plots by rendering user-defined templates written in the Graph Template Language. This presentation provides examples showing how you can use these procedures in your own work.

INTRODUCTION

An effective plot can reveal trends or patterns in your data that might have otherwise remained hidden in their tabular form. It is also important that the process for creating these plots be straightforward enough not to detract from the analysis. If a large amount of effort is required to create the plots you need for the analysis, then valuable time is lost. In addition, when the time comes to report your findings, you need features that make it easy for you to create graphics ready for publication or presentation.

In SAS 9.2, SAS/GRAPH introduces the first installment of a new family of procedures designed to create statistical graphics to assist in general-purpose data analysis. These procedures enable you to create graphics ranging from simple scatter plots to paneled displays with classification, all with a syntax that is clear and concise. The names of the new procedures all begin with “SG” to differentiate them from the “traditional” SAS/GRAPH procedures. These new procedures include the following:

- SGPLOT – used to create individual plots and charts with overlay capabilities
- SGPANEL – used to create paneled plots and charts driven by classification variables
- SGSCATTER – used to create comparative scatter plot panels, with the capability to overlay fits and confidences

To create a graph that is beyond the capabilities of these procedures, you can define a graph template by using the Graph Template Language (GTL) and use PROC SGRENDER to create the graph. For more information about GTL, see Jeff Cartier's SUGI 31 paper, “A Programmer's Introduction to the Graph Template Language” (2006).

As you read through these procedure examples, please note that SAS 9.2 is still under development. Therefore, code examples are subject to change or enhancement.

HOW TO USE THE NEW PROCEDURES IN STATISTICAL ANALYSIS

In SAS 9.2, many procedures in SAS/STAT®, SAS/ETS®, SAS/QC®, SAS High-Performance Forecasting, and Base SAS use the ODS Statistical Graphics infrastructure to create statistical graphics as automatically as they create tables (see the SUGI 31 paper by Bob Rodriguez and Tonya Balan (2006) for an overview of the ODS Statistical Graphics infrastructure). However, the graphics produced by these procedures are specific to the analysis done by the procedure. When you are faced with new data, using graphical procedures to create preliminary stand-alone plots of the data can be useful to learn about the data and can help set the direction for a subsequent statistical analysis.

The “SG” procedures can also be useful in the course of a statistical analysis, as in these tasks:

- combining the results from multiple procedures into a single graphical display – for example, creating a scatter plot of the raw data and overlaying computed fits obtained with different statistical methods
- creating custom displays not currently produced by the statistical procedures

The key is that these new procedures enable you to do these tasks with a syntax that is clear and concise, allowing you to focus on the analysis rather than on creating the graphics.

The following example illustrates how a preliminary data analysis with the SGPANEL procedure suggests a model to be analyzed with PROC GLM, which in turn produces a more specialized graphical display. Measurements of a response variable (Y) in an experiment are recorded in a SAS data set named MEASURE for levels of two factors (A and B). The following statements in the SGPANEL procedure create the paneled display shown in Figure 1:

```
proc sgpanel data=measure;
   panelby b / columns=3;
   vline a / response=y stat=mean
             limits=both;
run;
```

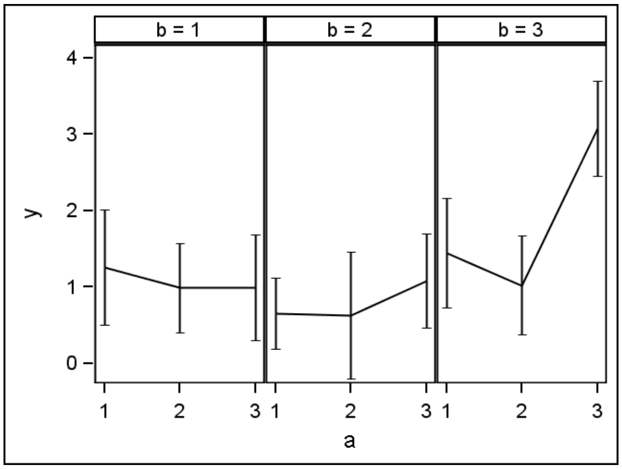


Figure 1. Interaction Plot of Factors A and B

In the past, creating a plot like the one in Figure 1 might have taken a large amount of graph code. Now you can quickly create graphs like these without disrupting the work flow of your analysis.

Each cell in the panel corresponds to the measurements for a level of factor B. The display within each cell plots the mean response at each level of A, along with 95% confidence limits for each mean. The change across levels of A for the third level of B, compared to the flat trends for the other two levels of B, suggests the presence of an interaction effect. An analysis of the variability due to the A and B factors should account for this possible interaction. The following statements request a two-way ANOVA with interaction by using the GLM procedure (Figures 2a and 2b).

```
ods graphics on;
proc glm data=measure;
   class a b;
   model y = a|b / ss3;
   lsmeans a*b / pdiff=all;
   ods select ModelANOVA DiffPlot;
run;
quit;
ods graphics off;
```

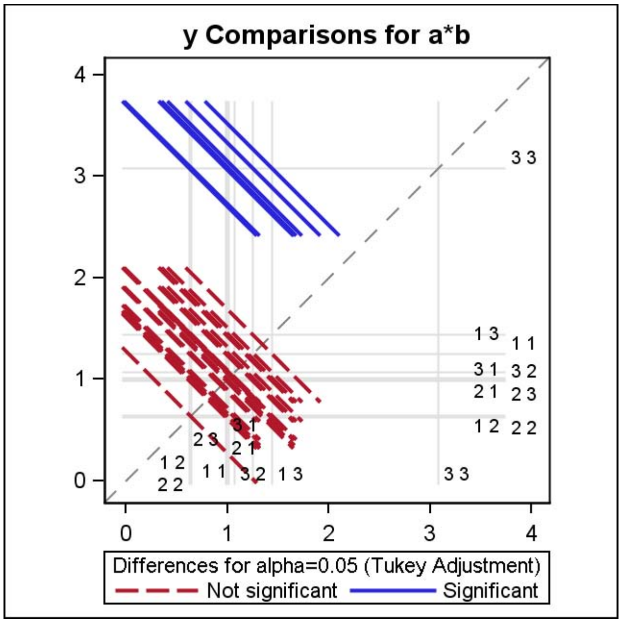
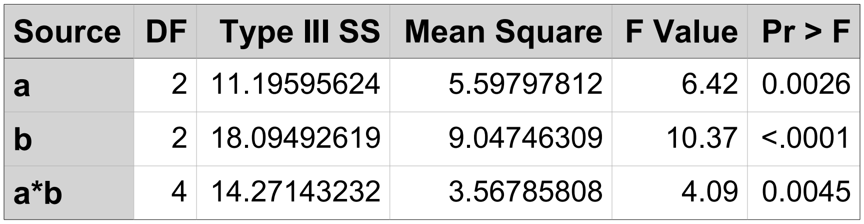


Figure 2a. A Diffogram Produced by GLM

Note that for the GLM procedure, as well as other statistical procedures, the graphics facility must be enabled by the ODS GRAPHICS statement for the graphics to be produced. This statement is not necessary for the "SG" procedures, since the sole intent behind these procedures is to produce graphics. However, the ODS GRAPHICS statement can be used to control different aspects of the "SG" procedure output, including image size and image output type.



Source	DF	Type III SS	Mean Square	F Value	Pr > F
a	2	11.19595624	5.59797812	6.42	0.0026
b	2	18.09492619	9.04746309	10.37	<.0001
a*b	4	14.27143232	3.56785808	4.09	0.0045

Figure 2b. ANOVA Table Produced by PROC GLM

The ANOVA table shown in Figure 2b confirms that there is a statistically significant interaction. The LSMEANS statement requests a plot (referred to as a diffogram) of all pairwise least-squares means differences (Figure 2a). This plot reveals that the (A=3, B=3) cell is involved in all of the significant differences among the A*B cell means. The lines representing these differences do not cross the 45-degree reference line.

USING THE PROCEDURES

In the following overview of the new “SG” procedures, a number of the graphs that are shown can also be produced with statistical procedures such as UNIVARIATE, LOESS, REG, and CORR, which use the ODS Statistical Graphics infrastructure in SAS 9.2. Deciding which procedure to use is a question of context and control:

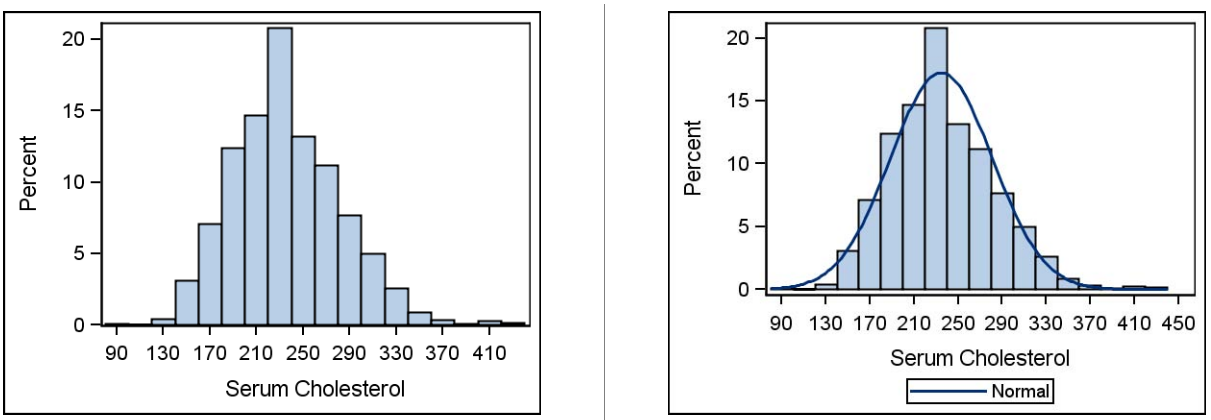
- Using a statistical procedure is recommended for creating graphs in the context of the analysis performed by a procedure. The user decides which graphs to request (some are produced by default), and the procedure does the work of creating and enhancing the graphs based on the information available from the analysis.
- Using an “SG” procedure is recommended for creating general-purpose, stand-alone graphs, where the user decides which display to make for a given set of data and controls the details.

SGPLOT PROCEDURE

The SGPLOT procedure is designed to create single-celled plots with powerful overlaying capabilities. A variety of plot types are supported, including the following:

- basic plots – scatter, series, step, band, and needle
- fits and confidence – loess, regression, penalized B-spline, and ellipse
- distributions – horizontal and vertical box plots, histogram, normal curve, and kernel density estimate
- categorization – dot plot, horizontal and vertical bar charts, horizontal and vertical line charts

These plot types can be combined to create more complex plots. As a simple example, suppose you have completed a heart study and you want to examine the cholesterol levels of the members of the study group. You might start out with a simple histogram (Figure 3a) to see the distribution. However, you might also want add a normal curve. Adding that curve is a simple operation (Figure 3b).



```
proc sgplot data=Fram_hrt;
   histogram chol;
run;
```

```
proc sgplot data=Fram_hrt;
   histogram chol;
   density chol; /* default: type=normal */
run;
```

Figure 3a and 3b. Histogram of Serum Cholesterol Levels

Notice that neither example requires appearance-type options, such as color, font, etc., to produce a high-quality graph. Since the “SG” procedures are based on the ODS Statistical Graphics infrastructure, the output is automatically sensitive to ODS styles. Using styles enables you to quickly create professional-looking output without spending extra time trying out different appearance values.

The order of the plot statements (and the REFLINE statement) is significant. The plots are drawn in the order in which they are specified in the procedure. The order of axis or legend statements has no impact on graph output. Notice the example in Figure 4. Since the reference line was specified first, it was drawn first, with the box plot drawn on top of it. To draw the line on top of the box plot, simply interchange the REFLINE and HBOX statements.

```
proc sgplot data=sashelp.heart
            (where=(smoking_status ne ' '));
   refline 80 / axis=x name="normal"
     lineattrs=(color=red)
     legendlabel="Normal diastolic pressure";
   hbox diastolic / category=smoking_status;
   keylegend "normal";
run;
```

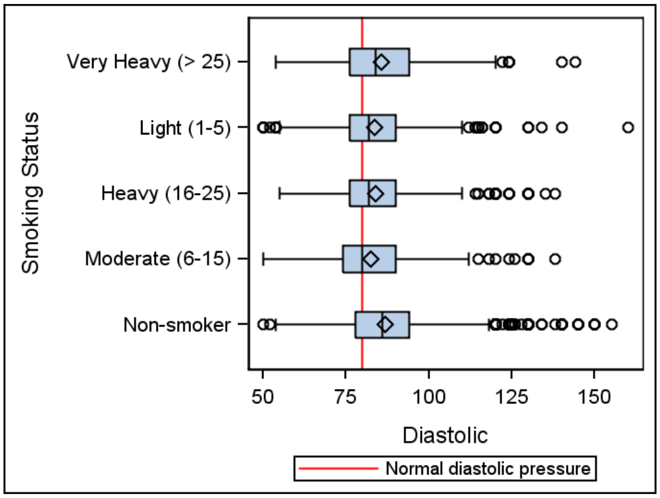


Figure 4. A KEYLEGEND Statement with an Assigned Reference Line

The statement order can be an important consideration in combining plot types that contain both filled areas and markers or lines. By default, filled areas drawn on top of markers obscure them; therefore, you typically want to draw your filled areas first. However, it is possible to add transparency to a filled area to see what is drawn behind it, as shown in Figure 5.

```
proc sgplot data=sashelp.class;
   scatter x=weight y=height;
   loess x=weight y=height / clm
         nomarkers clmtransparency=0.3
         name="loess";
   keylegend "loess";
run;
```

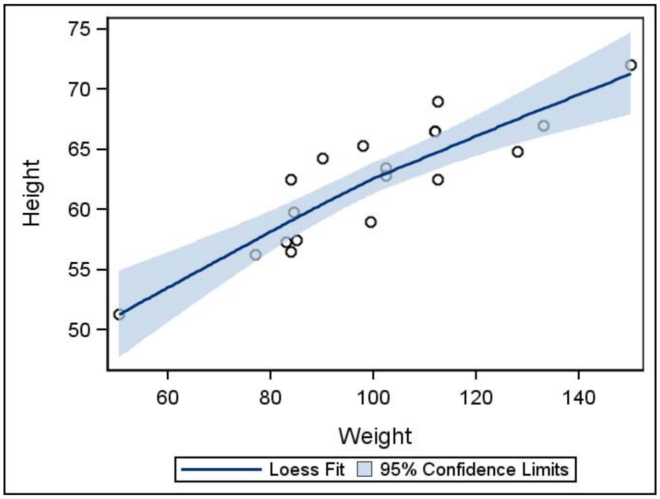


Figure 5. Confidence Limits with Transparency

Note that the loess smoother and the confidence limits in Figure 5 were computed using the same default methods as in the LOESS procedure.

Like most other SAS procedures, the SGPLOT procedure supports BY-group processing using the BY statement. When using BY groups, PROC SGPLOT supports a UNIFORM option that enables you to control the uniformity in the plot, which includes uniform group values, uniform axis scales, or complete uniformity.

SGPANEL PROCEDURE

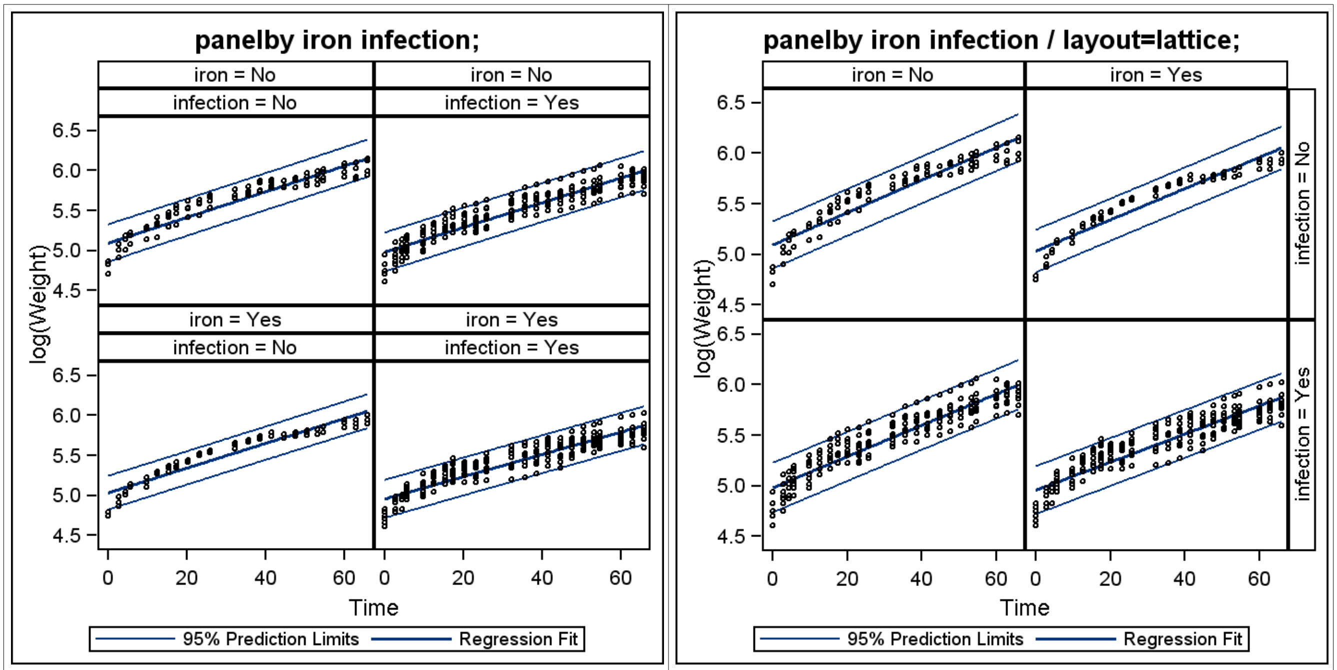
The SGPANEL procedure is designed to produce a paneled plot based on classification levels. The plots contained within the cell of each panel are defined much like the plots in the SGPLOT procedure, with the following plot types supported:

- basic plots – scatter, series, step, band, and needle
- fits and confidence – loess, regression, and penalized B-spline
- distributions – horizontal and vertical box plots, histogram, normal curve, and kernel density estimate
- categorization – dot plots, horizontal and vertical bar charts, horizontal and vertical line charts

As with PROC SGPLOT, these plot types can be combined to create more complex graphs.

The classification variables that drive the paneling are specified in the PANELBY statement. There are two types of paneling supported: panel (default) and lattice. When the panel type is used (Figure 6a), there is no limit to the number of classification variables you can specify, although each variable takes up more of your graph space in the panel. When the lattice type is used (Figure 6b), you are limited to two classification variables, one for the column and one for the row.

In addition to the PANELBY statement, the SGPANEL procedure also supports BY-group processing with the BY statement. A powerful approach for plotting multiple classifiers is to use only one or two of the most significant classification variables in the PANELBY statement and to use the BY statement to cycle through the remaining variables. This approach helps you to maximize the amount of space used by the plots in each cell. Another feature of PROC SGPANEL that makes this approach work well is automatic graph paging support. By default, PROC SGPANEL calculates an optimum number of cells per graph page and automatically generates each graph, even within a BY group. If you want to adjust the number of cells per graph, you can specify the number of rows and columns in the PANELBY statement. If only ROWS or COLUMNS is specified, PROC SGPANEL calculates an optimum count for the other value. By default, the axis scales are uniform across each graph page, even if the BY statement is not used.



```
title "panelby iron infection;";

proc sgpanel data=cows;
panelby iron infection;
reg x=tpoint y=weight / cli
             markerattrs=(size=3px);
run;
```

```
title "panelby iron infection /
layout=lattice;";

proc sgpanel data=cows;
panelby iron infection / layout=lattice;
reg x=tpoint y=weight / cli
             markerattrs=(size=3px);
run;
```

Figure 6a and 6b. Examples of Panel-Type and Lattice-Type Paneling

What makes the SGPANEL procedure such a powerful tool for data analysis is that it enables you to create complex plots with a small amount of procedure code. In Figure 7, notice that only a few statements are required to create a dot plot panel that shows an anomaly with the “Morris” data.

```
proc sgpanel data=barley;
   panelby site / novarname;
   dot variety / response=yield
                 group=year;
   keylegend / title="Year";
run;
```

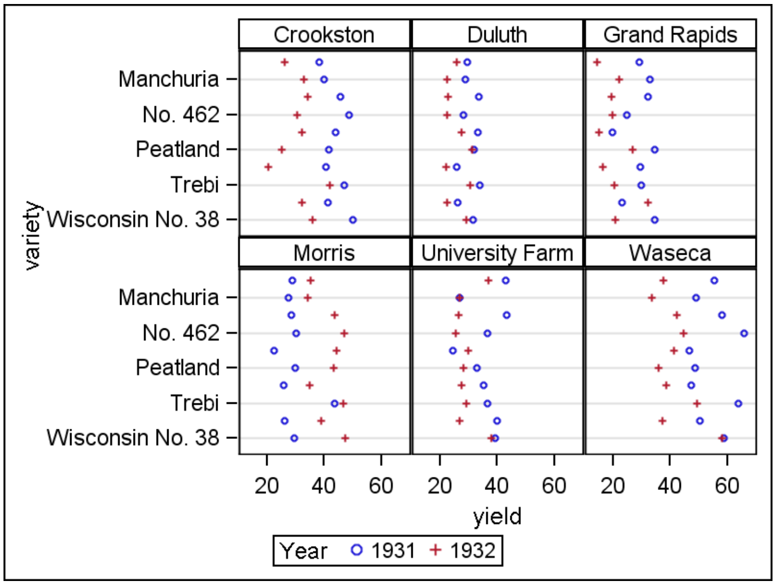


Figure 7. DOT Plot Using PROC SGPANEL

For SAS 9.2, only discrete classification is supported in the SGPANEL procedure. However, a simple way to handle interval classifier data is through the use of user-defined formats, as in the following box plot example (Figure 8):

```
/* Create classification ranges */
/* using a format               */
proc format;
   value ageint
      40-49 = "40-49"
      50-59 = "50-59"
      60-69 = "60-69";
run;

proc sgpanel data=Fram_hrt;
   format age ageint.;
   panelby age sex / layout=lattice;
   hbox chol;
run;
```

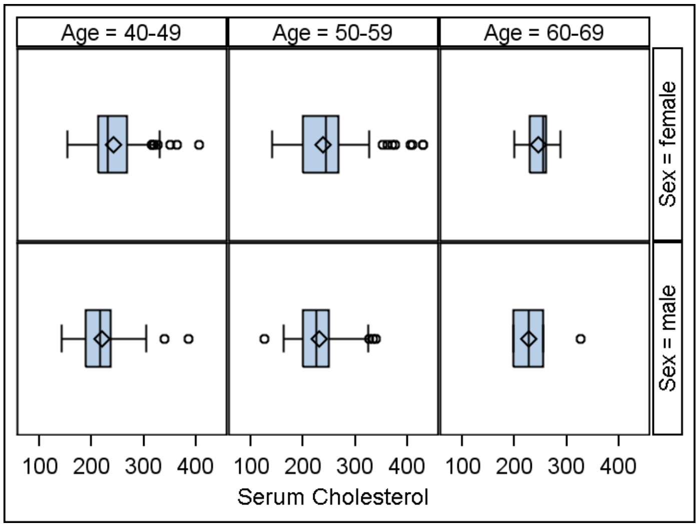


Figure 8. Box Plot with Age Intervals

Notice also the use of the lattice layout in Figure 8. If the example had used the default panel layout, the male and female boxes would have been adjacent to each other for each age group, making the comparison between males and females in different age groups more difficult. By using the lattice, you can quickly compare the serum cholesterol levels between males and females in any age group.

SGSCATTER PROCEDURE

The SGSCATTER procedure is designed to create panels of scatter plots, from independent scatter plots to scatter plot matrices. Although it is possible to create a single-cell scatter plot with this procedure, that plot is best created with PROC SGPLOT, which gives you more options for customization. This procedure also supports BY-group processing, which enables you to create a panel per unique crossing of classifiers. Unlike the SGPLOT and SGPANEL procedures, which have over 20 statements each, this procedure has only three statements:

- PLOT – creates independent plots for each Y*X pair in a panel
- COMPARE – creates a matrix that crosses a list of X variables with a list of Y variables
- MATRIX – creates a scatter plot matrix of a list of variables

In the PLOT statement, the method of specifying Y*X pairs can be any of the following forms (PROC GPLOT users should recognize these forms):

- Y_0*X_0...Y_n*X_n
- Y*(X_0...X_n)
- (Y_0...Y_n)*X
- (Y_0...Y_n)*(X_0...X_n)

Each resulting pair produces a scatter plot in the panel. You can control either the number of rows or the number of columns in the panel, but not both. The panel will grow in either the row or column direction, depending on the option specified. If neither option is specified, PROC SGSCATTER automatically generates a balanced panel based on the number of pairs. There is no paging in this procedure as there is in PROC SGPANEL, except for standard BY-group paging. Here is a simple example that uses some data on car performance (Figure 9):

```
proc sgscatter data=sashelp.cars;
   plot (mpg_city mpg_highway)
       *(weight horsepower) /
         group=type;
run;
```

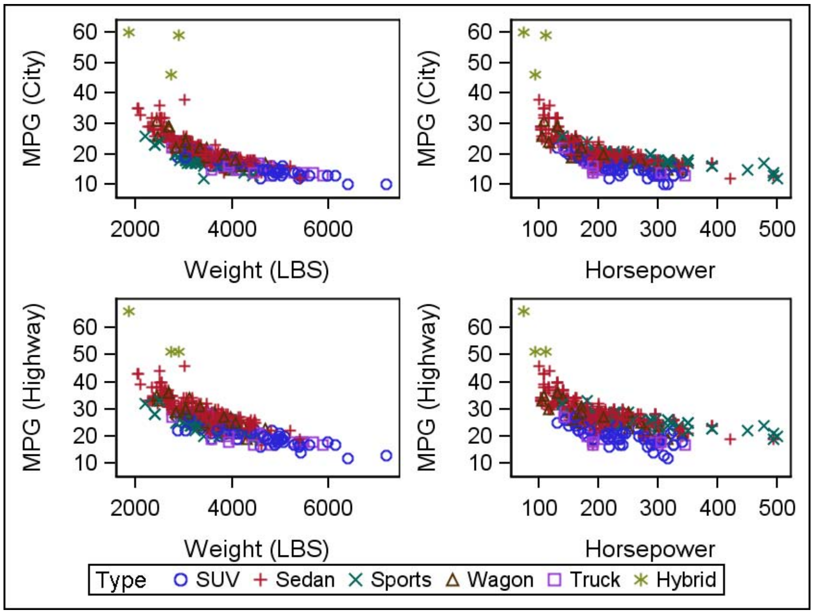


Figure 9. SGSCATTER Using the PLOT Statement

Notice that you are allowed one group variable, and it is applied to all plots. In addition, the existence of a group variable automatically displays the legend. The LEGEND option enables you to control the position and appearance of the legend when it is displayed.

The PLOT statement is not limited to scatter plots. You can request loess, regression, or penalized B-spline fits of the scatter points, as well as computed ellipses. Each of these features has options that you can use to control them. Let's take the example from Figure 9 and view the loess fit for truck-type vehicles (Figure 10):

```
proc sgscatter data=sashelp.cars
      (where=(type eq 'Truck'));
   plot (mpg_city mpg_highway)*
      (weight horsepower) / loess;
run;
```

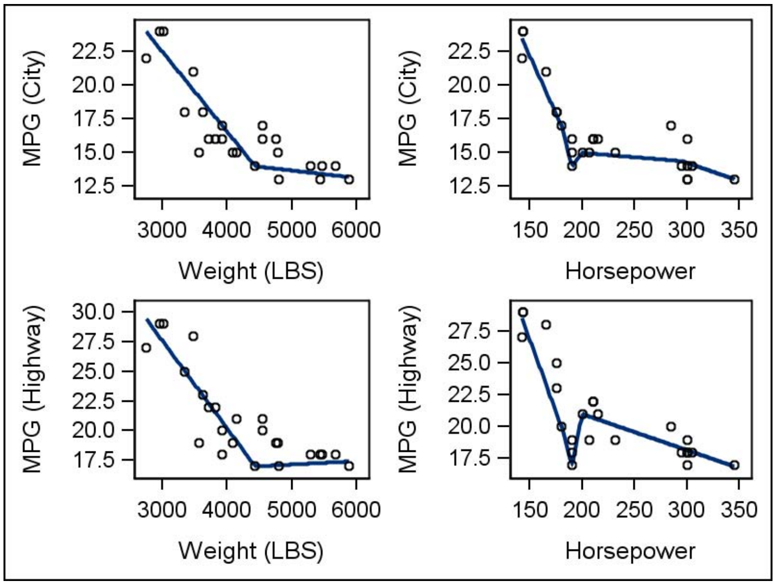


Figure 10. The PLOT Statement with LOESS Fits

Each of these plots has individual axis ranges; however, there might be times when you want your axes to be uniform for comparison purposes. The PLOT statement has an option called UNISCALE that enables you to set the X or Y axis, or both axes, to a uniform scale.

Scatter point labels are also possible with the PLOT statement. If you specify the DATALABEL option without a variable name, the Y variable values from each plot are used to label the points. However, you might have a column of observation numbers or ID values you want to display across all plots. In that case, specify the variable in the DATALABEL option, and only the values from that column are used to label the scatter points.

The COMPARE statement enables you to create a comparison matrix of plot pairs that share axes. The pairs are created by crossing the list of X and Y variables specified in the statement. It is also possible to create a plot sharing a single X or Y axis by simply specifying one X or Y variable. The statement has the same fit and confidence features as the PLOT statement, as well as the data labeling capability. As with the PLOT statement, only one group variable is allowed, and it is used by all variable crossings. Using the car data from the previous example, you can create a simple plot with a shared Y axis (Figure 11). By specifying lists of variables for both X and Y, you can create a larger comparison matrix.

```
proc sgscatter data=sashelp.cars;
   compare x=(mpg_city mpg_highway)
           y=horsepower;
run;
```

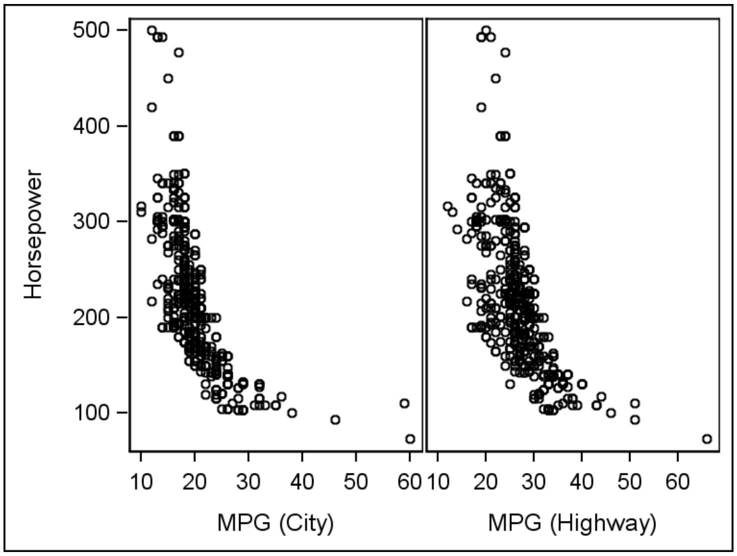
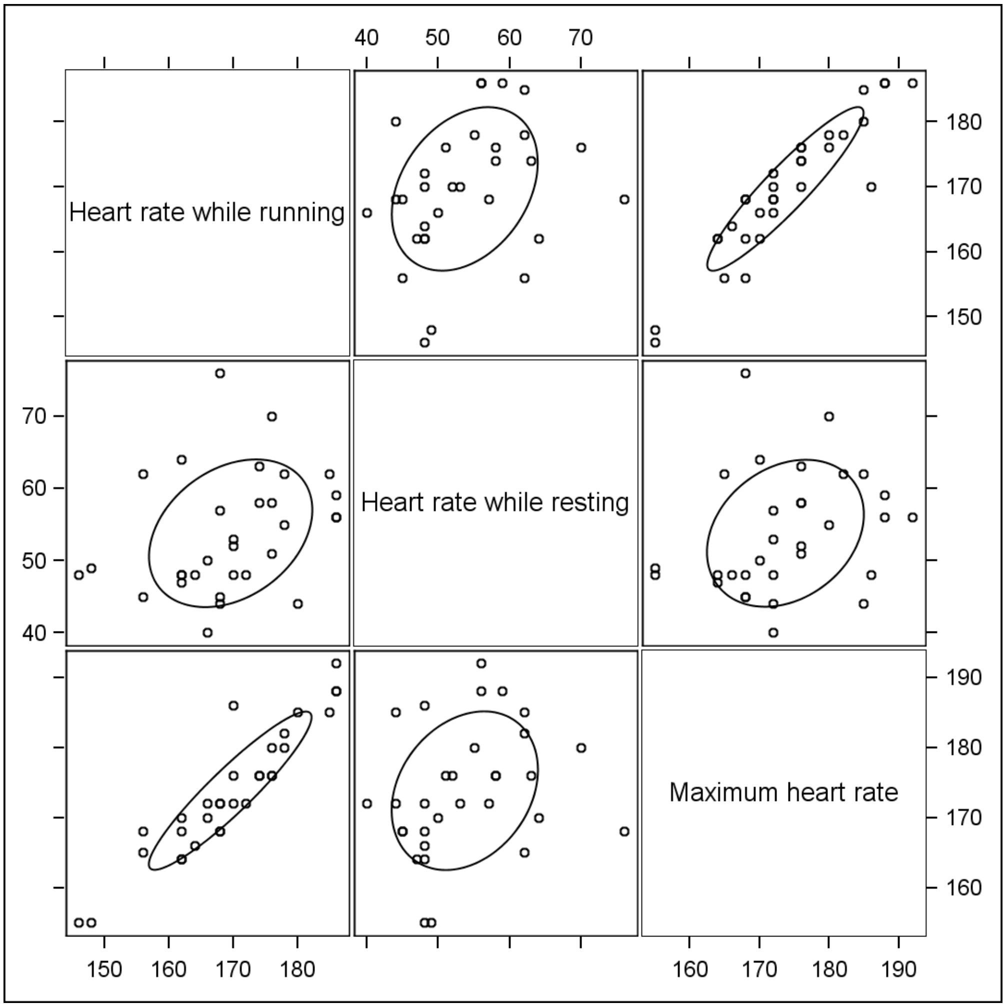


Figure 11. Shared Axis Using the COMPARE Statement

The MATRIX statement is used to create a scatter plot matrix from a list of variables. The fit options are not supported with this statement, but the ELLIPSE and DATALABEL options are available.

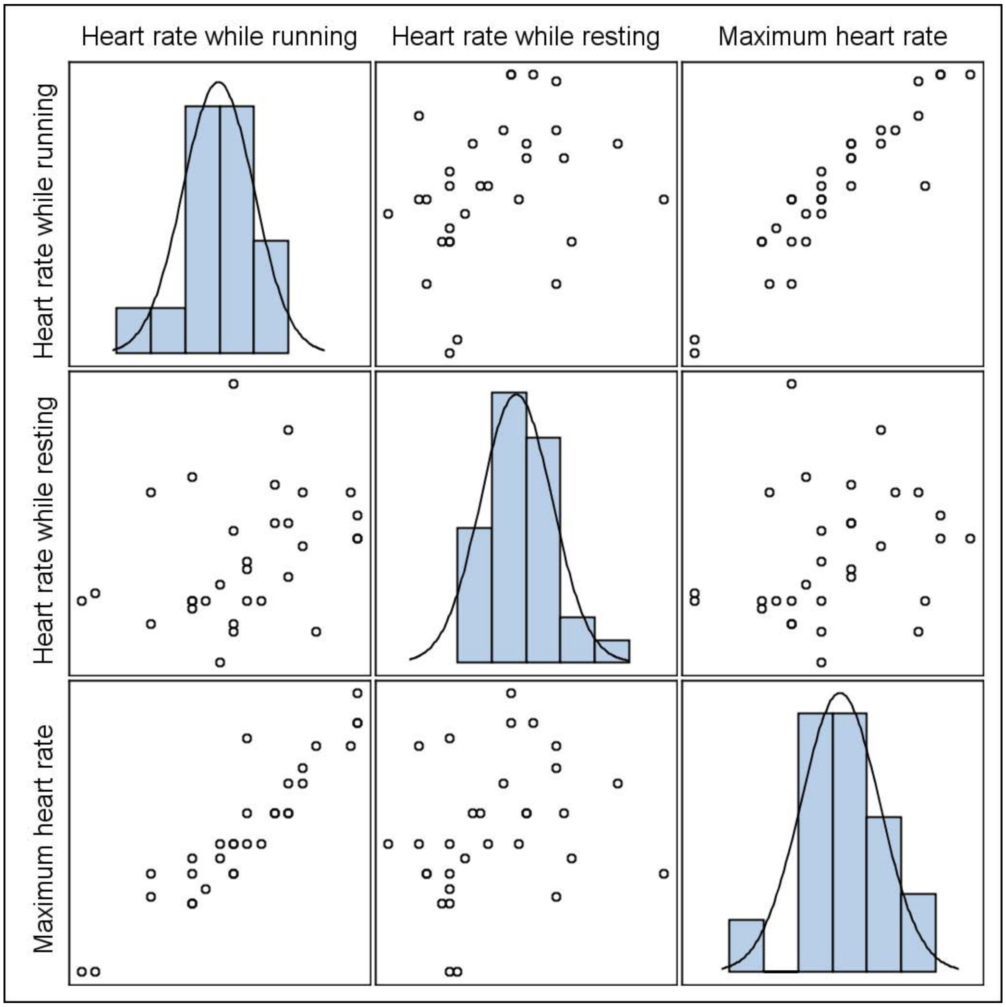
Using some fitness data, here is a MATRIX example that uses the ELLIPSE option to create computed prediction ellipses (Figure 12):



```
proc sgscatter data=fitness;
   matrix runpulse rstpulse maxpulse / ellipse=(alpha=0.5);
run;
```

Figure 12. Scatter Plot Matrix with the ELLIPSE Option

A unique feature of the MATRIX statement is the capability to add features to the diagonal (Figure 13). The DIAGONAL option enables you to add histograms, normal curves, and kernel density estimates to see the distribution of the matrix variables. These features can also be overlaid. When the DIAGONAL option is used, however, all axes are removed, since the scales between the scatter plots and the histograms are incompatible.



```
proc sgscatter data=fitness;
   matrix runpulse rstpulse maxpulse / diagonal=(histogram normal);
run;
```

Figure 13. Scatter Plot Matrix with the DIAGONAL Option

SGRENDER PROCEDURE

If you find a situation where you are not able to create the desired display by using one of the previously described "SG" procedures, you might want to use the Graph Template Language (GTL) to design a custom display and then use PROC SGRENDER to create the output by using that graph template and your data. GTL is the language used by the statistical procedures to create their displays by using the ODS Statistical Graphics infrastructure. GTL enables you to create custom displays with a high degree of control and flexibility, but with that capability comes a higher degree of complexity than that of the "SG" procedure syntax. A discussion of GTL is beyond the scope of this paper (see Jeff Cartier's SUGI 31 paper, "A Programmer's Introduction to the Graph Template Language" (2006)); however, if you use GTL to create custom displays, the SGRENDER procedure supports the following useful features:

- BY-group support – enables your templates to be rendered using BY-grouped data
- special dynamics – dynamics such as _BYLINE_, _BYVAR_, _BYVAR2_, _BYVAL_, _BYVAL2_, _LIBNAME_, and _MEMNAME_ are surfaced through the procedure and are available for use in your template
- paging support – automatic paging support when a DATAPANEL or DATALATTICE layout is used in your template

PROCEDURE CONCEPTS

DIFFERENCES FROM TRADITIONAL SAS/GRAPH PROCEDURES

When you are beginning to use the new “SG” procedures, you need to understand certain important concepts that are different from the concepts of “traditional” SAS/GRAPH procedures. A primary concept deals with the use of global statements. Many traditional SAS/GRAPH global statements do not apply to these procedures, as shown in the following list:

- Apply – TITLE, FOOTNOTE, FORMAT, LABEL
- Do not apply – GOPTIONS, AXIS, LEGEND, PATTERN, SYMBOL, NOTE

However, in the following sections, you will see how each procedure enables you to control each of these graphical features.

All visual attributes for plots and charts, such as color, marker shape, line style, etc., are derived either from the active ODS style or from the syntax within the procedure. Plot types are also determined directly from the statements or options used within the procedure. This approach for coding plot attributes directly in the procedure eliminates the need for global statements such as PATTERN and SYMBOL and adds clarity to your programs.

Instead of the GOPTIONS statement, the “SG” procedures use the ODS GRAPHICS statement, the same statement used to control the global attributes for ODS Statistical Graphics output from other statistical procedures. This statement can be used to control attributes such as output image type, image size, and image map support.

Another concept that does not apply to the new “SG” procedures is that output from these procedures does not generate an entry in a GSEG catalog. However, you can use the ODS DOCUMENT destination to save generated output from these procedures and replay the output later using PROC DOCUMENT.

TITLES AND FOOTNOTES

Except for SGRENDER, all of the “SG” procedures support the TITLE and FOOTNOTE global statements. These procedures support the majority of TITLE and FOOTNOTE statement options and functionality, with the exception of text angling, annotation, and hyper-linking options. There is also one behavioral difference to note with these procedures regarding the JUSTIFY option: if you try to justify two strings in the same location (such as j=r) in the statement, the strings *append* instead of moving to the next line. If you want to have two lines of right-justified text, you need to use two statements.

Since these procedures use ODS styles through the ODS Statistical Graphics infrastructure, titles and footnotes automatically acquire their text attributes from the style, minimizing the need for appearance options. Here is a typical example (Figure 14):

```
title1 "This is the Main Plot Title";
title2 "This is a subtitle";
footnote j=l "NOTE: Created with SGPLOT"
         j=r "Date: 12/14/2006";
proc sgplot data=sashelp.class;
  scatter x=weight y=height;
run;
```

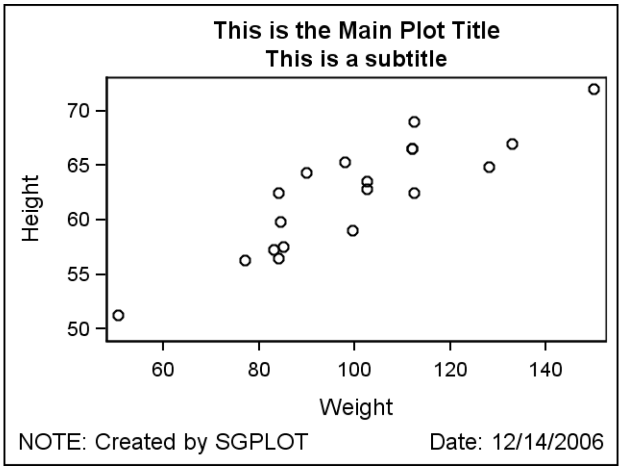
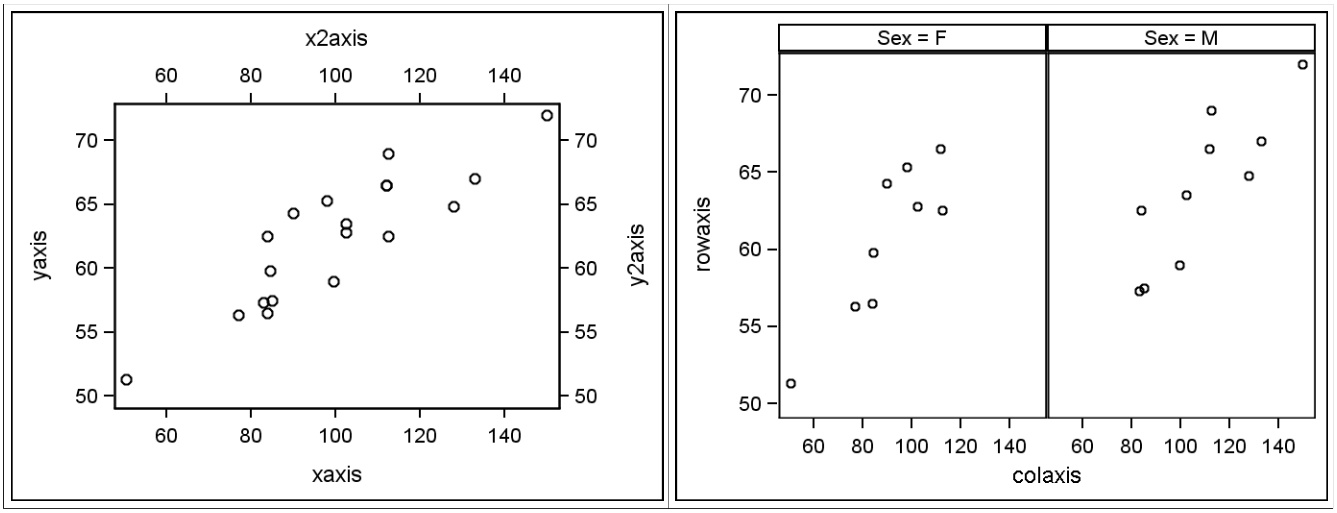


Figure 14. Titles and Footnotes Using PROC SGPLOT

AXES

Axes are controlled from within the procedures. Within PROC SGPLOT, there are four axis statements: XAXIS, YAXIS, X2AXIS, and Y2AXIS. Each statement enables you to control a different axis around the plot (Figure 15a).
The axis statements support four types of scales: linear, discrete, log, and time. The default scale is determined by the data; however, that scale can be changed by the user. A typical case for changing the scale might be to change the axis from a linear axis to a log axis. The syntax of these statement is focused on features such as range setting, value fitting, and scale

configuration options. Appearance options, such as color and fonts, are acquired from the ODS style.



Figures 15a and 15b. Axes in the SGPLOT and SGPANEL Procedures

The SGPANEL procedure follows the same philosophy as PROC SGPLOT, except there are only two statements: ROWAXIS and COLAXIS (Figure 15b).

The SGSCATTER procedure has no axis statements, since plot panels are created from a single statement.

LEGENDS

The SGPLOT and SGPANEL procedures automatically create a legend for you based on the content of the graph, not simply based on whether a group variable is used. This default legend can be overridden in one of two ways:

- using the NOAUTOLEGEND procedure option to disable the legend
- defining your own legend(s) by using the KEYLEGEND statement(s), which disables the automatic legend

The KEYLEGEND statement enables you to control not only the title, position, and arrangement of a legend, but also its content. You can choose which parts of an overlay are represented in a legend. For example, suppose you have some car data, and you want to see the trend of weight vs. horsepower across vehicle types (Figure 16). In this example, you would want to see the vehicle types in the legend but not the fit line. You can exclude the fit line by referencing only the SCATTER statement in the KEYLEGEND statement. Any number of plot or REFLINE statements can be referenced in a KEYLEGEND statement, enabling you to control the legend content.

```
proc sgplot data=sashelp.cars;
   loess x=weight y=horsepower /
           clm nomarkers;
   scatter x=weight y=horsepower /
           group=type name="scatter";
   keylegend "scatter";
run;
```

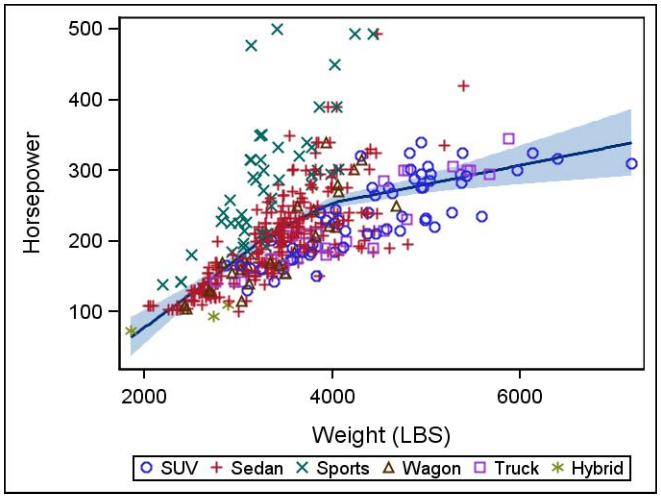


Figure 16. Customized Legend in PROC SGPLOT

The SGPLOT and SGPANEL procedures also support multiple legends. For example, if you have two plot statements with different groups, you might want to assign each plot to its own legend:

```
proc sgplot data=mydata;
  scatter x=x1 y=y1 / group=g1 name=”scatter”;
  series x=x2 y=y2 / group=g2 name=”series”;
  keylegend ”scatter”;
  keylegend ”series”;
run;
```

The SGSCATTER procedure does not have a legend statement; however, a legend will appear if a group variable is used. Legend options exist in each statement that enable you to control different aspects of the legend, including position and arrangement. One position option unique to PROC SGSCATTER is the capability to position a legend in an empty cell of a panel. For example, if you request a 3-column-by-2-row panel by using the PLOT statement, but you have only 5 pairs to plot, you can request that the legend be placed in the remaining empty cell of the panel, maximizing the amount of space for the graph.

INSETS

With traditional SAS/GRAPH procedures, the NOTE statement or Annotate is typically used to insert additional information, such as statistics, directly into a graph. With PROC SGPLOT, this kind of information can be added using the procedure's INSET statement. There are two types of insets supported: text and tables. The text form of inset enables you to add statements, even small paragraphs, to the plot. The table form enables you to add label-value pairs to a table that is inset in the plot. Both of these forms enable you to position the inset in any of eight positions inside the plot area. Here is a plot showing both forms of inset (Figure 17):

```
proc sgplot data=sashelp.class;
   scatter x=weight y=height;
   inset "This is some important information"
         "in the bottom-right corner" /
         position=BottomRight;
   inset ("A"="37" "B"="209" "C"="Value") /
         position=TopLeft border;
run;
```

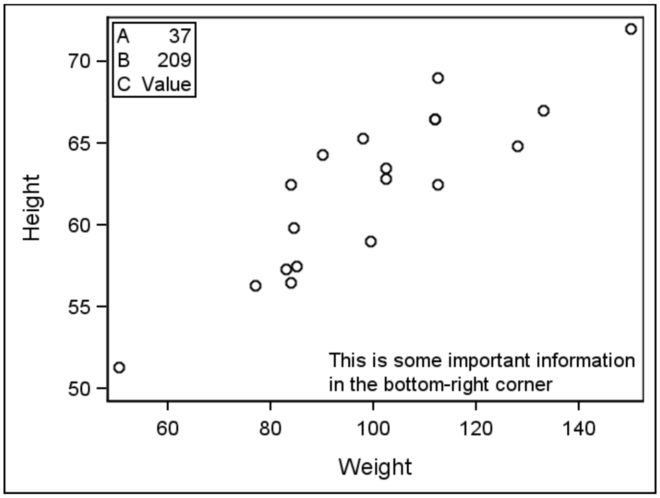


Figure 17. Insets Using PROC SGPLOT

The INSET statement also supports Unicode characters and superscripting/subscripting of text. This support is important for adding standard statistical symbols and statistics to insets. Although direct Unicode values can be used, there are predefined names you can use without having to look up the Unicode values. This list includes the Greek characters, as well as BAR, BAR2 (double bar), HAT, PRIME, and TILDE, for the respective diacritical marks. Figure 18 replaces the table in the previous example with some text that uses these features.

```
ods escapechar='\';
proc sgplot data=sashelp.class;
   reg x=weight y=height / clm cli;
   inset ("Y\{unicode bar}"="62.34"
          "R\{sup '2'}"="0.94"
          "\{unicode alpha}"=".05") /
          position=TopLeft border;
run;
```

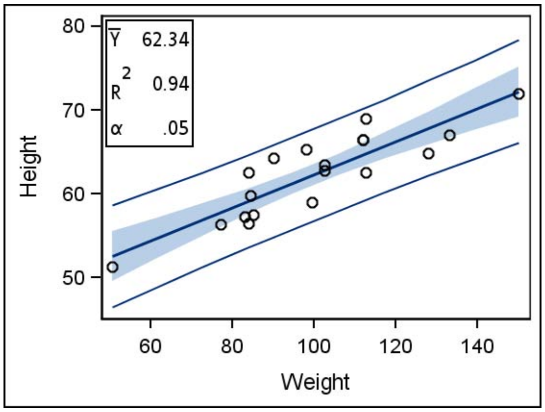


Figure 18. Inset with Unicode Values

Notice that the Unicode or superscript/subscript functionality requires you to use the defined ODS escape character, or the default ODS escape sequence if a character is not defined (the sequence is *(*ESC*)*). The value for the UNICODE function must be either one of the previously mentioned keywords, or a Unicode hex value defined by '*hex-value*'x. The value for the SUP or SUB function must be a quoted string.

This text functionality is not limited to table insets: it can be used with the text form as well. For example, you might want to set the alpha value in the top-left corner with the following statement:

```
      inset "\{unicode alpha} = .05" / position=TopLeft;
```

USING THE PROCEDURES WITH ODS STYLES

When you generate output from the “SG” procedures, ODS styles are always applied to the plots, even when the user does not specify a style. In fact, all of the previous examples in this paper were generated using the LISTING style, which is the default style for the ODS LISTING destination. In addition to the default styles for each ODS destination, a few additional styles are specifically tuned to work well with statistical graphics:

- STATISTICAL – color-based style that works well for HTML and PDF output
- ANALYSIS – color-based style that works well for HTML and PDF output
- JOURNAL – gray-scale-based style that works well for all output types
- JOURNAL2 – same as JOURNAL, with fills turned off for histograms, confidence bands, box plots, and computed ellipses

In the “SG” procedures, ODS style elements can be accessed directly to modify the attributes of plot features. There are a number of graph-specific style elements that define the appearance of such features as fit lines, confidence limits, reference lines, and outliers. There are many elements that are used automatically based on the plot request in the procedure; however, at times, you might want or need to use a different style element. For example, suppose you compute your own custom fit through a collection of data. By default, the SERIES line uses a default appearance acquired through the GraphDataDefault style element. However, the GraphFit style element is tuned to give a better fit-line appearance. If you reference that element, it can save you the time of trying out custom attributes; and it guarantees that if you change the ODS style or open multiple ODS destinations, your plot features will still blend correctly with each style without changing your graph code (Figures 19a and 19b).

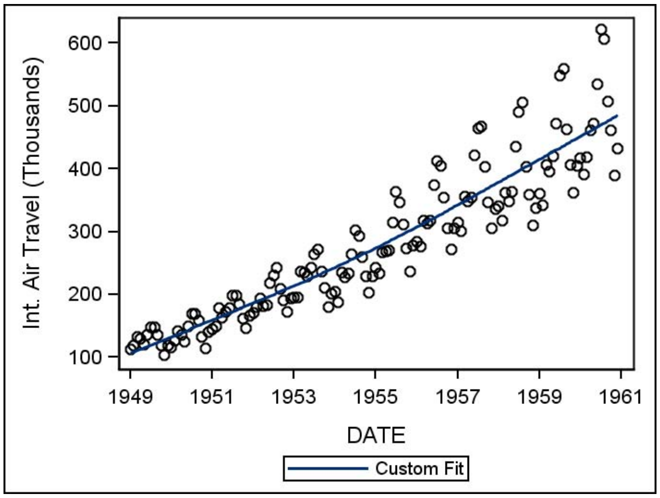


Figure 19a. Custom Fit with the LISTING Style

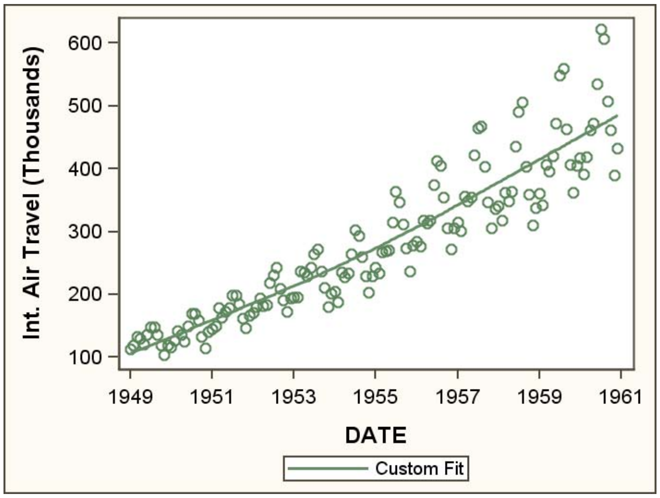


Figure 19b. Custom Fit with the ANALYSIS Style

```
proc sgplot data=DataWithFit;
   label air "Int. Air Travel (Thousands)";
   scatter x=date y=air;
   series x=date y=fit / lineattrs=GraphFit legendlabel="Custom Fit" name="fit";
   keylegend "fit";
run;
```

There might be times when you want to override an attribute of a style element or not use a style element at all. You can choose either option directly through procedure syntax (Figure 20).

```
proc sgpanel data=Fram_hrt;
   panelby sex;
   histogram chol;
   density chol /
      lineattrs=GraphFit(color=red);
   density chol / type=kernel
      lineattrs=(color=green pattern=dash
                 thickness=2);
run;
```

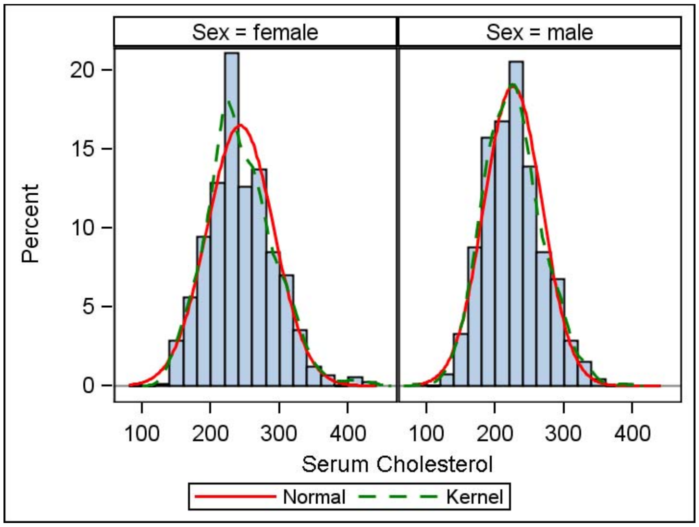


Figure 20. Custom Line Attributes

CONCLUSION

As you can see from the code examples and resulting plots, the new “SG” procedures enable you to produce a wide range of graphics, from simple to complex, with just a small amount of coding effort. Custom features such as insets and classification panels are now simple operations. The use of ODS styles minimizes the need to use plot attribute syntax, as well as the need to try a lot of appearance values to find the ones that work well together. The examples in this paper have barely scratched the surface of the ways these procedures can be used to help you accomplish your data analysis and reporting tasks. As you use these procedures, please give us feedback so we can continue to refine and improve these tools to meet your needs.

ACKNOWLEDGMENTS

I want to thank Bob Rodriguez, Sanjay Matange, and Jeff Cartier for their valuable input to this paper, as well as Susan Schwartz and Melisa Turner for their review time.

CONTACT INFORMATION

Dan Heath
SAS Institute Inc.
500 SAS Campus Dr.
Cary, NC 27513
Dan.Heath@sas.com