Paper SAS1780-2015

Graphs are Easy with SAS® 9.4

Sanjay Matange, SAS Institute Inc., Cary, NC

ABSTRACT

Axis tables, polygon plot, text plot, and more features have been added to Statistical Graphics (SG) procedures and Graph Template Language (GTL) for SAS® 9.4. These additions are a direct result of your feedback and are designed to make creating graphs easier. Axis tables let you add multiple tables of data to your graphs and to correctly align with the axis values with the right colors for group values in your data. Text plots can have rotated and aligned text anywhere in the graph.

You can overlay jittered markers on box plots, use images and font glyphs as markers, specify group attributes without making style changes, and create entirely new custom graphs using the polygon plot. All this without using the annotation facility, which is now supported both for SG procedures and GTL. This paper guides you through these exciting new features now available in SG procedures and GTL.

INTRODUCTION

The introduction of Statistical Graphics (SG) Procedures and Graph Template Language (GTL) as part of the ODS Graphics system in SAS® 9.2 has been a great leap forward for data presentation using SAS. The SG procedures provide an easy to use, yet flexible syntax to create most commonly used graphs for data analysis and presentation. GTL allows the creation of complex multi-cell graphs using a structured syntax. Every new feature in the SG procedures has a corresponding feature in GTL, so we will cover these new features by discussing their SG versions.

First, some history and timeline. In SAS 9.2, the focus was to provide the graphs necessary for the SAS analytical procedures.

In SAS 9.3, the focus shifted from needs of the statistical procedures to the needs of the SAS users who create custom graphs for their analyses and presentations. Many new features enabled the creation of the most commonly used graphs in many industries.

In SAS 9.4, we have taken the next step in this evolution. Now the goal is not only to make these graphs possible but to make them easy. This paper describes many of these new features with actual use cases in SAS 9.4, the first maintenance release of SAS 9.4 (SAS 9.4M1), and the second maintenance release of SAS 9.4 (SAS 9.4M2). This paper is for any SAS user who wants to create their own graphs for analysis and presentations and for SAS users of all skill levels.

REVIEW OF KEY NEW FEATURES

To be sure, there are a lot of new features to discuss. Most or all of these features are in areas that we observed need help, or they are features that you directly requested. The full list is too long to cover each item, but we will cover the marquee items in the following sections.

New statements:

1. XAXISTABLE and YAXISTABLE statements
2. STYLEATTRS statement
3. POLYGON statement
4. TEXT statement
5. SYMBOLCHAR and SYMBOLIMAGE statements
6. DROPLINE statement

Key new features:

1. Splitting of text values for axis values, labels, data labels, and more
2. Fitting data labels in bar charts
3. Broken axes
4. Overlaying basic plot with vertical box plots.
5. Histogram with groups
6. Series plots with color group, pattern group, and symbol group
7. Proportional bars, insets, header order, and cell order for panels

Other new features:

1. Axis label position
2. Discrete color bands for an axis
3. Equal scale for Y & Y2 axes
4. Minor grids
5. Bar chart baseline
6. Gradient fill for bar charts and histograms
7. Segment labels

NEW STATEMENTS

XAXISTABLE

A key feature of analytical and clinical graphs is the inclusion of raw and derived statistical data correctly aligned with other elements of the graph. One common use case is the display of the "Subjects-At-Risk" values in a survival plot.

The XAXISTABLE statement is designed to handle the display of data inside or outside a graph and to correctly position this data along a numeric or discrete x-axis. See in Figure 1.

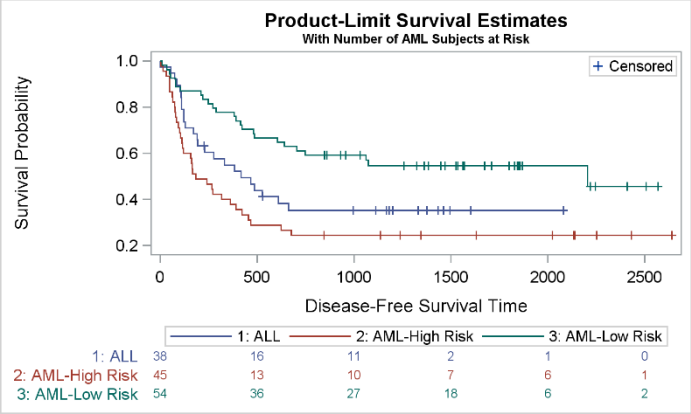


Figure 1. Survival Plot with Outer Risk Table

Here is the code for this graph:

```
title 'Product-Limit Survival Estimates';
title2  h=0.8 'With Number of AML Subjects at Risk';
proc sgplot data=SurvivalPlotData;
  step x=time y=survival / group=stratum name='s';
  scatter x=time y=censored / markerattrs=(symbol=plus) name='c';
  scatter x=time y=censored / markerattrs=(symbol=plus) GROUP=stratum;
  xaxistable atrisk / x=tatrisk class=stratum colorgroup=stratum;
  keylegend 'c' / location=inside position=topright;
  keylegend 's';
run;
```

Note in Figure 1, the "Subjects-At-Risk" values are displayed at the bottom of the graph for each stratum. The values are shown only for time values of 0, 500, 1000, and so on. We call this an "Axis Table" because the values are aligned with values on the x-axis.

The CLASS=stratum option displays all the unique risk values for the stratum variable. The LOCATION=outside option (which is the default) displays the table of values at the bottom of the graph area.

Prior to SAS 9.4, the display of the at-risk values at the bottom of the graph could be done only by using the SG Annotation facility. This requires considerable more work than the one line statement above.

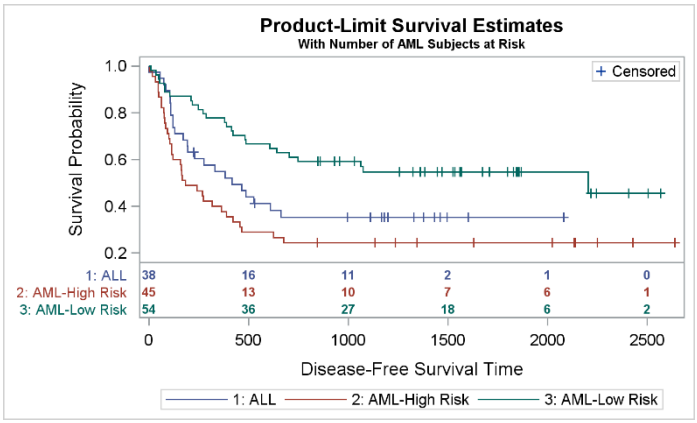


Figure 2. Survival Plot with Inner Risk Table

Figure 2 shows the survival plot with the "Subjects-At-Risk" values aligned with the x-axis inside the data area and above the x-axis. You can generate this output by setting the LOCATION=Inside option in the XAXISTABLE statement. Here is that code:

```
xaxistable atrisk / x=tatrisk class=stratum colorgroup=stratum
                    location=inside;
```

Placing the "At-Risk" data closer to the rest of the data in the graph reduces the eye movement required to process the data and increases the effectiveness of this graph. The COLORGROUP=stratum option colors the values by the stratum variable. As a result, the association with the survival curves is easier to determine. Prior to SAS 9.4, we could create this graph using SG Annotate or the MARKERCHAR option in the SCATTER plot statement.

YAXISTABLE

Just like we placed rows of textual values along the x-axis at specified values, we often need to place columns of textual values along the y-axis. A popular example of such a graph is the "Forest Plot." See Figure 3.

This graph consists of the odds ratio graph in the middle, a column of study names on the left, and multiple columns of statistics on the right. Once again, it is important to align the rows of textual values with the rows of the odds ration plot.

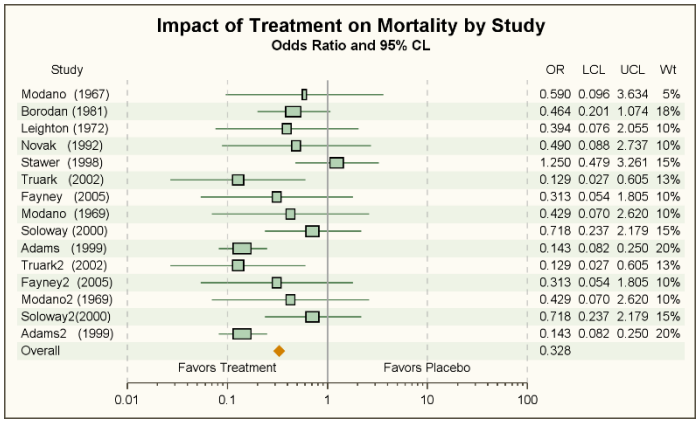


Figure 3 – Forest Plot with Weighted Odds-Ratios

The graph is constructed as follows:

1. The odds-ratio plot in the middle displays OR by Study. The y-axis of the graph is suppressed.
2. A SCATTER statement displays the odds-ratio for the "Overall" case.
3. A HIGHLOW statement plots the weighted odds ratio and confidence limits.
4. A YAXISTABLE statement displays "Study" names on the left by using the POSITION=left option.
5. A YAXISTABLE statement displays four statistics on the right using the POSITION=right option.
6. A TEXT statement places the "Favors Treatment" and "Favors Placebo" labels at the bottom.

7. Alternate color bands are used for the y-axis to help the eye across the graph.

Here is the SGPLOT code for the graph in Figure 3:

```
title "Impact of Treatment on Mortality by Study";
title2 h=8pt 'Odds Ratio and 95% CL';
proc sgplot data=forest noautolegend nocycleattrs nowall noborder;
  scatter y=study x=or2 / markerattrs=graphdata2(symbol=diamondfilled);
  highlow y=study low=lcl high=ucl / type=line;
  highlow y=study low=q1 high=q3 / type=bar barwidth=0.6;
  yaxistable study / y=study location=inside position=left;
  yaxistable or lcl ucl wt / y=study location=inside position=right;
  refline 1  / axis=x noclip;
  refline 0.01 0.1 10 100 / axis=x lineattrs=(pattern=shortdash)
          transparency=0.5 noclip;
  text y=study x=xlbl text=lbl  / position=center contributeoffsets=none;
  xaxis type=log  max=100 minor display=(nolabel)  valueattrs=(size=7);
  yaxis display=none fitpolicy=none reverse valueshalign=left
        colorbands=even colorbandsattrs=Graphdatadefault(transparency=0.8);
run;
```

YAXISTABLE WITH INDENTATION

A popular version of the Forest Plot is the graph shown in Figure 4 – the "Forest Plot with Subgroups."

The key difference between this graph and the simple Forest Plot in Figure 3 is the way the study names are grouped. Instead of having a list of study names, the study names are grouped into smaller chunks.

The group names are displayed in a bold font, and the values in the group are displayed in a normal font, with or without indention. Only the "Value" rows have hazard ratio plots and the statistics on the right.

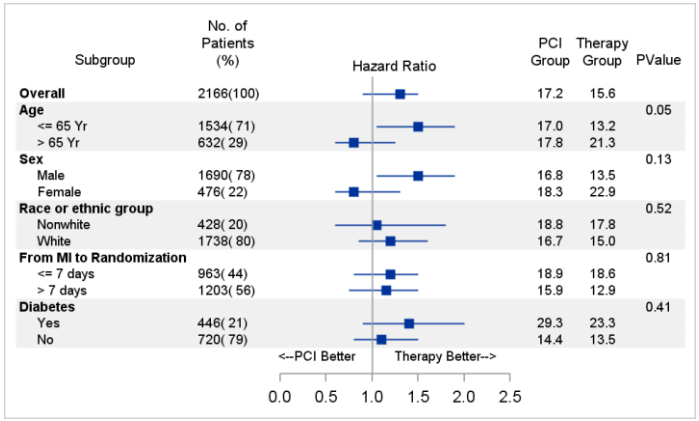


Figure 4. Sub-grouped Forest Plot with Indented Values

The YAXISTABLE statement has special options to create such a graph. Here is the code to create the graph in Figure 4:

```
yaxistable subgroup  / location=inside position=left textgroup=id
          textgroupid=text indentweight=indentWt;
```

The XAXISTABLE statement and YAXISTABLE statement have many options to control the output. Here are some key options:

- LOCATION – places the table inside or outside the data region bounded by the axes.
- POSITION – places the table in one of the two directions for each statement.
- CLASS – creates multiple rows (for XAXISTABLE) or columns (for YAXISTABLE).
- COLORGROUP – specifies a color for the values of each classifier.
- CLASSDISPLAY – positions the class values as a stack (which is the default) or clustered.
- CLASSORDER – controls the order in which to display the rows or columns.
- STAT – specifies the statistic to use to summarize the numeric columns.
- TEXTGROUP – specifies a classifier for controlling text attributes like size, font, weight.

- TEXTGROUPID – specifies the attribute map ID to control the text size, font, weight, and so on.
- DROPONMISSING – specifies whether a row or column is displayed if all values are missing.

STYLEATTRS

Users of the SGPLOT procedures often create graphs that are grouped by one or more classifier variables as shown in the grouped scatter plot in Figure 5.

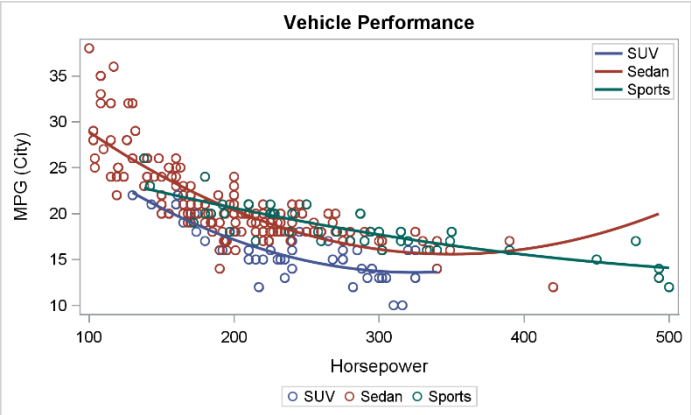


Figure 5 – Grouped Scatter Plot

In such cases, the elements in the graphs derive their visual properties from the GRAPHDATA1 -12 elements of the active style. In this case, the "Type" column is used as the classifier, and the data is subset to include "Sedan", "Sports" and "SUV".

The graph in Figure 5 uses the HTMLBlue style, which is a COLOR priority style. This means, that color for the symbols and lines are changed only if the number of unique group values exceed 12 (the number of group colors in the style).

To see changes in marker symbols and line patterns, we can change the style to one that has an attribute priority of NONE, or set the ATTRPRIORITY=NONE option in the ODS GRAPHICS statement (as shown in the following code).

Now, to group and use symbols that are different from the ones in the style, we can specify the symbols to use in the STYLEATTRS statement as shown in the following code.

```
ods graphics / reset attrpriority=none;
title 'Vehicle Performance';
proc sgplot data=sashelp.cars(where=(type in ('Sedan' 'SUV' 'Sports')));
  styleattrs datasymbols=(circlefilled trianglefilled diamondfilled);
  scatter x=horsepower y=mpg_city / group=type name='a';
  reg x=horsepower y=mpg_city / degree=2 group=type name='b' nomarkers;
  keylegend 'a';
  keylegend 'b' / location=inside position=topright across=1 linelength=30;
run;
```

The STYLEATTRS statement enables you to specify your own list of group attributes to use. You can provide a list of data colors, data contrast colors, data symbols, and data line patterns by using the appropriate option.

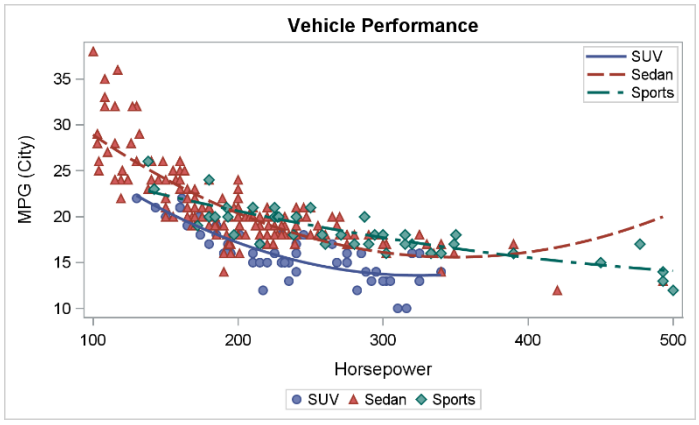


Figure 6 – Grouped Scatter Plot with Custom Attributes

The list of values that you provide override the corresponding list of attributes in the active style.

In the previous SAS code, we named three symbols to use for groups. The resulting graph is in Figure 6.

Prior to SAS 9.4, you would have to derive a new style and change the order of the

symbols and colors yourself. The STYLEATTRS option was added in response to your request for an easier way to do this.

ATTRIBUTES MAP

While Figure 6 successfully changed the marker symbols to use, the order of symbol assignment is still based on the order of the group values in the data. Since "SUV" is encountered first in the data, it gets the attributes from GRAPHDATA1.

This can be a problem because the assignments are data dependent. Next time around, "SUV" might not be first, and the attribute assignment will change from graph to graph based on the order of the data.

To address this issue, the Attributes Map feature was introduced in SAS 9.3. The attribute map works like a format. You can specify the visual attributes you want to assign to each group value. Then, regardless of their position in the data, the attribute values remain consistent. The map is defined in the SAS data set shown in Figure 8.

With SAS 9.4M3, this feature will be extended. The legend values can be populated from an Attributes Map instead of from the plot. In this case, if there are more values in the attributes map than are in the data, the values are still drawn. Also, the order of the values in the legend is the same as the order in the attributes map, as shown in Figure 7.

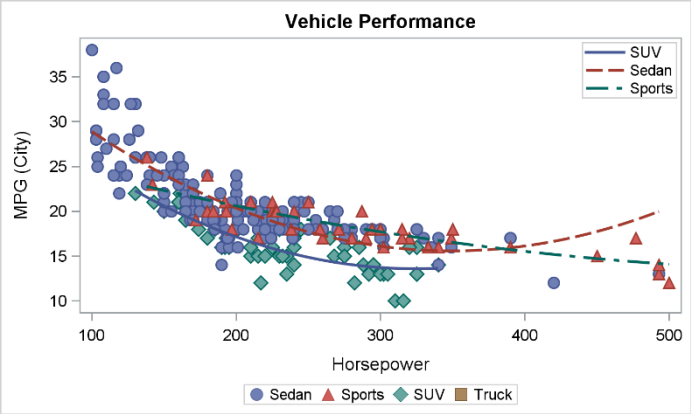


Figure 7 – Group Attributes from Map

Obs	Id	Value	MarkerStyleElement	MarkerSymbol
1	Cars	Sedan	GraphData1	CircleFilled
2	Cars	Sports	GraphData2	TriangleFilled
3	Cars	SUV	GraphData3	DiamondFilled
4	Cars	Truck	GraphData4	SquareFilled

Figure 8 – Attributes Map

Notice that the legend in Figure 7 has four entries, but there is no group value of "Truck" in the data. The attribute maps feature allows us to include the color swatch for "Truck" even if that value is not in today's data. This functionality is useful in many domains, where you might have "Mild", "Moderate" and "Severe" events. Although the data of a particular subject might have only some of the events, we still want to show all the possible events in the legend.

POLYGON PLOT

SAS 9.4M1 includes the new POLYGON plot statement, which allows you to create a custom graph. The polygon plot works like SG Annotate with the addition that the axes are aware of the data ranges in this plot.

To describe its features, let us create a custom graph that cannot be created by any statement in the SGPLOT procedure. We will use the data set in Figure 9, which shows the list of names in a fundraising group, the number of items sold, and the amount of money raised.

Obs	Name	Sex	Age	Items	Money
1	Alfred	M	14	8	$29
2	Alice	F	13	12	$20
3	Barbara	F	13	13	$29
4	Carol	F	14	5	$11
5	Henry	M	14	5	$13
6	James	M	12	11	$11

Figure 9 – Fund Raising Data set

Let us create what is sometimes referred to as an "Area Bar Chart" as shown in Figure 10. The graph shows the items sold by each student. However, the x-axis is not discrete with the name of each student like the usual bar chart.

The x-axis is numeric, and displays the cumulative funds raised as we go to the right. The bar for "Alfred" is drawn first. The bar width is equal to the money raised by Alfred (which is $28), and the bar height is equal to the number of items sold.

Then, the bar for "Alice" is drawn, positioned just to the right of Alfred. The width of this bar is the amount raised by Alice, which is $20. A bar is drawn for all the students, and the total money raised is shown by the range on the x-axis.

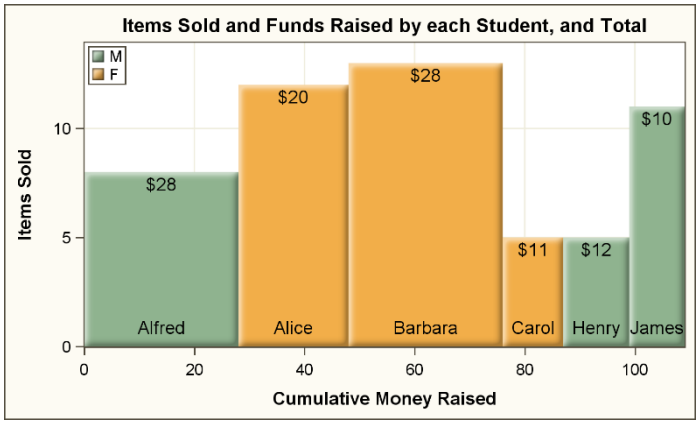


Figure 10 – Area Bar Chart

The code for this graph is shown below. A POLYGON statement is used to draw each bar. Each polygon is identified by the ID. Each polygon has 4 vertices, which are computed in a DATA step as previously described. The position for the name labels is also computed.

```
title 'Items Sold and Funds Raised by each Student and Total';
proc sgplot data=Area_Bar;
  polygon id=name x=x y=y / group=sex fill dataskin=sheen label=money
          labelattrs=(color=black size=10) labelpos=ymax name='a';
  text x=xlbl y=ylbl text=name / position=top textattrs=(size=10);
  xaxis offsetmin=0 offsetmax=0 grid;
  yaxis offsetmin=0 integer grid;
  keylegend 'a' / location=inside position=topleft across=1;
run;
```

The money raised by each student is displayed near the top of the polygon, inside the bounding box. The POLYGON statement gives you flexibility in placing the label anywhere relative to the polygon bounding box, either inside or outside the axes.

A view of the polygon data set is shown in Figure 11. "Name" can be used as the "ID" for each polygon. Each polygon is a rectangle, so there are four observations defining the (x, y) coordinates for each corner. (xlbl, ylbl) is used to draw the names using the TEXT plo statement. Here is the DATA step code:

```
data Area_Bar;
  label x='Cumulative Money Raised'
        y='Items Sold';
  retain prev 0 ylbl 0.4;
  set Fund_Raising;
  id=_n_; y=0; xlbl=prev + money/2;
  x=prev; output;
  xlbl=.;
  x=prev+money; output;
  y=items; output;
  x=prev; output;
  prev+money;
run;
```

Obs	Name	Sex	Items	Money	x	y	xlbl	ylbl
1	Alfred	M	8	$28	0	0	14.0	0.4
2	Alfred	M	8	$28	28	0	.	0.4
3	Alfred	M	8	$28	28	8	.	0.4
4	Alfred	M	8	$28	0	8	.	0.4
5	Alice	F	12	$20	28	0	38.0	0.4
6	Alice	F	12	$20	48	0	.	0.4
7	Alice	F	12	$20	48	12	.	0.4
8	Alice	F	12	$20	28	12	.	0.4

Figure 11 – Data for Polygon Plot

TEXT PLOT

Figure 10 uses the new TEXT plot statement to draw the text strings at specified locations in the graph. Prior to SAS 9.4M2, we often used the SCATTER plot statement with the MARKERCHAR option to draw text strings at specified locations in the graph. However, as we needed more features for text display, we

could not justify continuing to overload the SCATTER plot statement with text features. So, while these features are available in the SCATTER plot statement, our intention is to promote the use of the TEXT plot statement for such functions.

The TEXT statement provides many new features for displaying text in a graph. Here is the syntax for the TEXT statement:

```
TEXT x=col y=col text=col;
```

The available options include Group, GroupDisplay, ColorResponse, ColorModel, BackFill, BackLight, Position, Rotate, SizeResponse, Split options, ContributeOffsets and more.

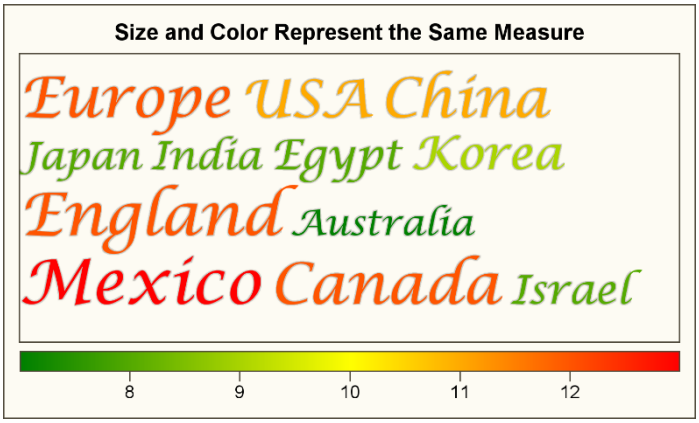


Figure 12 – Word Cloud

These options can be used to build various unique graphs including the Word Cloud shown in Figure 12. A special (experimental) feedback option will be included in SAS 9.4M3 TEXT plot to enable tight placement of such text strings.

SYMBOLCHAR AND SYMBOLIMAGE

SG procedures provide you with a set of 30 built-in marker symbols. These symbols are scalable, so you get symbols that scale proportionately (both in size and thickness for unfilled shapes) for higher DPI. These symbols also scale in SVG output.

However, there is always a need for more symbols. Starting in SAS 9.4M1, the SGPLOT procedure supports creating symbols using Unicode fonts or image icons. Now, you have an unlimited supply of symbol shapes.

The SYMBOLIMAGE statement defines a symbol from any image. In the following example code, we defined four new symbols from images with assigned names. Because all images are rectangular in shape, it helps to have transparent images. Images can be rotated as shown in the code for the ribbon symbol.

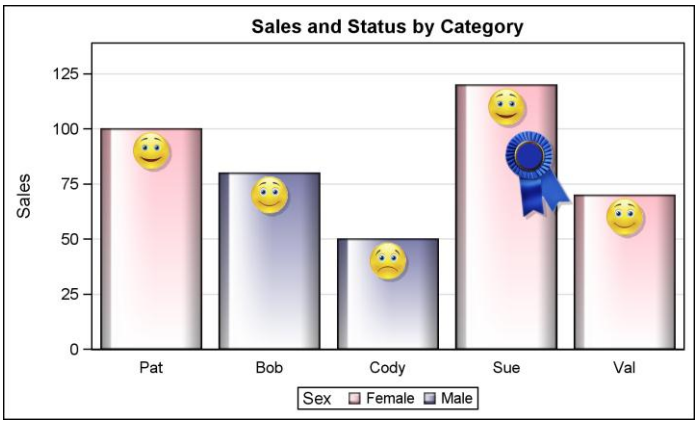


Figure 13 – Image Symbols

```
title 'Sales and Status by Category';
proc sgplot data=bars;
  symbolimage name=Great image="C:\VeryHappy_Tran.png";
  symbolimage name=Good  image="C:\Happy_Tran.png";
  symbolimage name=Poor  image="C:\Sad_Tran.png";
  symbolimage name=Ribbon  image="C:\Blue_Ribbon_Tran.png" / rotate=15;
  styleattrs datasymbols=(Great Good Poor) datacolors=(pink cx7f7faf);
  vbarparm category=cat response=Sales / group=sex dataskin=gloss;
  scatter x=cat y=loc / group=status markerattrs=(size=30) dataskin=crisp;
  xaxis display=(nolabel) offsetmin=0.1 offsetmax=0.1;
  yaxis offsetmin=0 offsetmax=0.1;
run;
```

After new symbol names are defined, these names can be used in the STYLEATTRS statement like any built-in symbol name (such as CIRCLEFILLED). The list of symbols that you specify are used in place of the symbols coming from the style. The SYMBOLIMAGE statement supports additional options such as Rotate, Scale, HOFFSET, and VOFFSET.

In the graph in Figure 13, the bar color indicates the gender of the salesperson. Here, we use the "Face" markers to show the target level of the sales. The graph also identifies the top salesperson with a blue ribbon.

The SYMBOLCHAR statement defines a symbol from any character of any font. Unicode fonts provide you thousands of characters, and many fonts also provide special shapes.

```
SYMBOLCHAR name=symbolname char="hex-string"x / < options>;
```

Supported options include Rotate, Scale, HOFFSET, and VOFFSET and TEXTATTRS.

DROPLINE

The DROPLINE statement is very similar to the REFLINE statement, except that instead of drawing a line across the entire width or height of the data area, a line is drawn from an (x, y) point in the data to one of the axes. The DROPLINE statement can be very useful when only a partial line is desired. See Figure 14.

This graph shows the Maximum LFT lab values by treatment. For ALAT, ASAT and ALKPH, the level of concern is 2.0, while for BILTOT it is 1.5. In this graph, it is preferable to draw the level of concern lines to cover only the appropriate lab values.

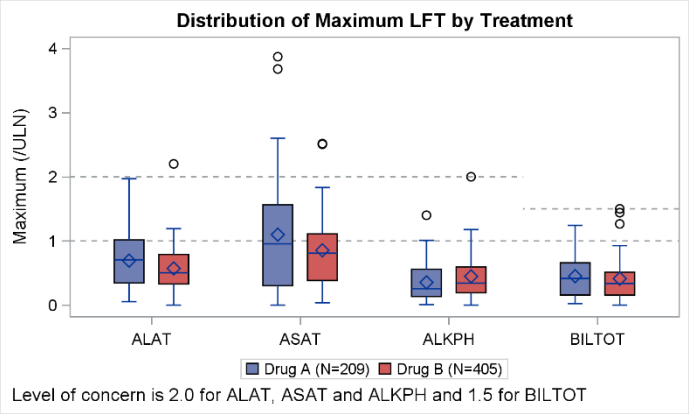


Figure 14 – Distribution of Maximum LFT by Treatment

Because the lower level of "1.0" applies to all lab values, we can use a REFLINE statement to draw the line for the lower level. However, we need two lines for the upper level. Both lines need to start halfway between ALKPH and BILTOT on the x-axis. The dropline at y=2.0 drops to the left y-axis, and the dropline at 1.5 drops to the right y-axis. The DISCRETEOFFSET=-0.5 option positions the starting point to left of the value.

```
DROPLINE x='BILTOT' y=2.0 / dropto=y discreteoffset=-0.5;
```

KEY NEW FEATURES

In addition to the new statements, many new features have been added to existing statements to make graphs easier to create. Many of these features are our effort to simplify the creation of graphs. Other features are in direct response to requests and suggestions by you. There are a lot of new features included in SAS 9.4, SAS 9.4M1, and SAS 9.4M2. This paper covers the key items.

SPLIT TEXT FOR AXIS TICK VALUES AND ALL LABELS

Starting in SAS 9.4, discrete tick values for the x-axis are automatically split at the default white space.

A new set of options was added to control this behavior. These options are FITPOLICY, SPLITCHAR, SPLITJUSTIFY, and SPLITCHARNODROP. The default split character is one or more "white space" character, such as blank, tab, and so on. The values on the x-axis are split based on the length of the longest word that will fit in the midpoint spacing.

The FITPOLICY=SPLITALWAYS option can be used to force a split at the split character. Figure 15 shows a graph with split tick values for the x-axis and y-axis. For the y-axis, we use "Split Always" and also specify the SPLITJUSTIFY=left option.

```
title 'Sales by Book and Author';
proc sgplot data=Books;
  bubble x=Book y=author size=sales /
         dataskin=gloss;
  yaxis display=(nolabel)
        fitpolicy=splitalways
        splitjustify=left;
  xaxis display=(nolabel;
  run;
```

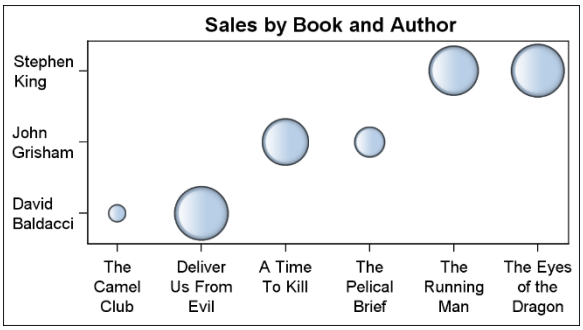


Figure 15 – Split Axis Tick Values

Axis labels can be split over multiple lines, and curve and data labels can also be split using a split character.

BAR LABEL FITTING

Bar labels are displayed based on the data type. For a vertical bar chart (VBAR), numeric bar labels are rotated when they do not fit in the midpoint spacing between the bars. See Figure 16. Discrete bar labels are split.

```
title 'Sales and Count by Author';
proc sgplot data=Books;
  vbar author / response=sales dataskin=gloss
       datalabel=count nostatlabel;
  xaxis display=(nolabel) valueattrs=(size=6)
       discreteorder=data;
  run;
```

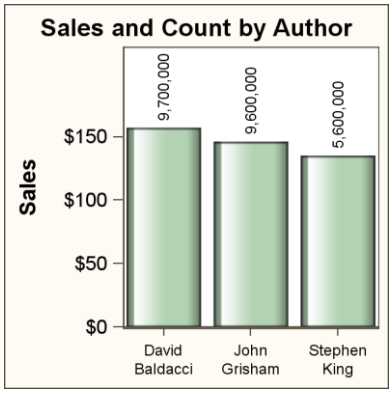


Figure 16 – Rotated Data Labels

BROKEN AXIS

A feature often requested by you is support for a "broken axis" to address use cases where the data contains extreme outliers. Plotting data to include the outliers can skew the scale on the axis, and as a result, the rest of the data is hard to view.

Figure 17 shows a case where a broken axis is useful. The graph on the left shows 4 bars with different response values. Bars "A," "B," and "C" have values under 50, while bar "D" has a value over 1200. Displaying all bars on one axis severely skews the scaling on the y-axis, making bars "A," "B," and "C" barely visible.

The ranges for the y-axis are specified in the following code to create a broken y-axis. A "Full" break is shown here. Options to show the break only on the axis are coming in a future release.

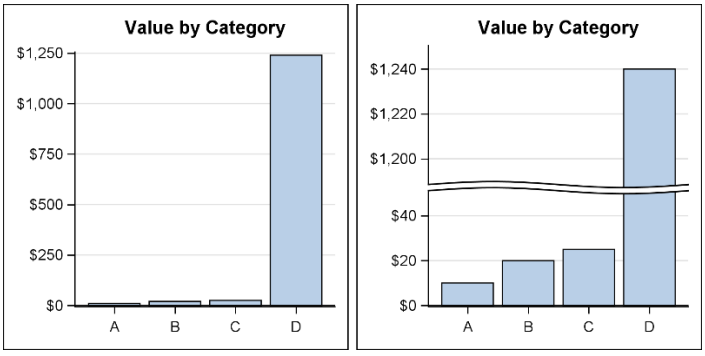


Figure 17 – Broken axis

```
proc sgplot data=broken;
  vbarcat / response=value;
  xaxis display=(nolabel);
  yaxis display=(nolabel) ranges=(0-50 1190-1250) values=(0 to 1250 by 20);
  run;
```

VBOX OVERLAY WITH BASIC PLOTS

One of the most powerful features of the SGPLOT procedure is the ability to mix and match plot statements to create unique graphs. Many basic and computed plot statements can be combine in innumerable ways to create your custom graph.

One limitation is that we can combine only basic plots such as scatter plots or series plots, with each other or with fit plots. Even when the axis data types are the same, we cannot combine a bar chart and a box plot with each other or other basic plots.

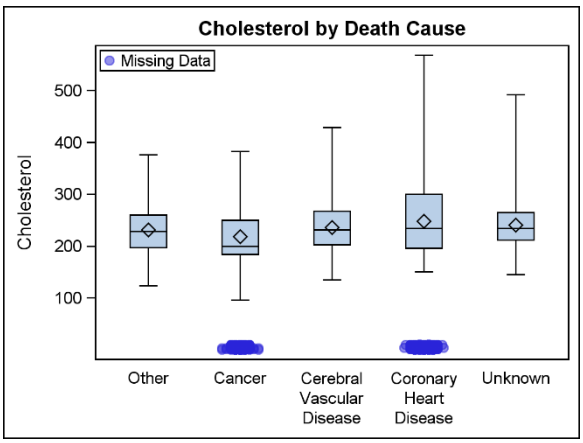


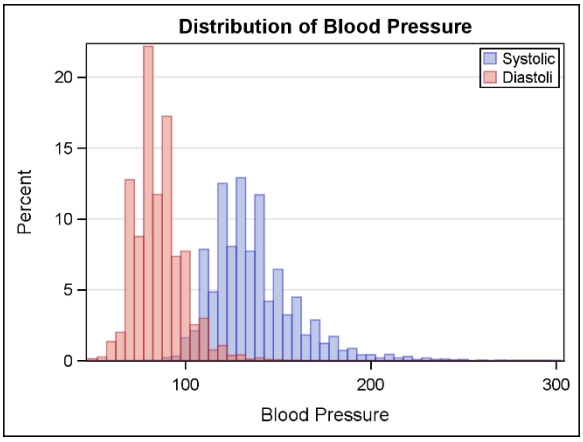
Figure 18 – Combining a Vertical Box Plot and a Scatter Plot

Starting in SAS 9.4M1, one of these restrictions has been removed. Now, you can combine box plots with basic plot types, such as a scatter plot. Combing these plots can show you the distribution of the data and the presence of bad or missing data. Figure 18 was created using the following code. Note use of the EXTREME option for the VBOX statement and the JITTER option for the SCATTER statement.

```
title 'Cholesterol by Death Cause';
proc sgplot data=heart_Box noautolegend;
  vbox cholesterol / category=deathcause extreme;
  scatter x=deathcause y=chol / markerattrs=graphdata1(symbol=circlefilled)
          transparency=0.5  name='s' jitter jitterwidth=0.5
          legendlabel='Missing Data';
  keylegend 's' / location=inside position=topleft;
  xaxis display=(noticks nolabel);
  yaxis values=(100 to 500 by 100) min=0 valueshint;
run;
```

HISTOGRAMS WITH GROUPS

SAS 9.4M2 now supports a GROUP option in the HISTOGRAM statement. This option makes it much easier to create a comparative histogram. The histograms in Figure 19 were created using the following code.




```
proc sgplot data=heart;
  histogram bp / group=type
    transparency=0.5 name='a';
  keylegend 'a' / location=inside
    position=topright across=1
    opaque;
  yaxis grid;
```

```
run;
```

Figure 19 – Grouped Histogram

Comparative histograms can also be created by using multiple histograms from different columns in the data set. Creating comparative histograms in this manner requires some customizations of the axis and the creation of bins using the BINSTART and BINWIDTH options.

SERIES WITH COLOR AND PATTERN GROUPS

Spaghetti plots are popular for representing multiple classifications for events over time. Figure 20 shows a graph of adoption rate over time.

The connectivity of each series line is determined by the GROUP classifier. Normally, the color and pattern for each line group in the data is also determined by the group value.

However, often it is necessary to group the series plot segments by additional classifiers, such as drug class or study name. In this example, each series is colored by the location and patterned by the year.

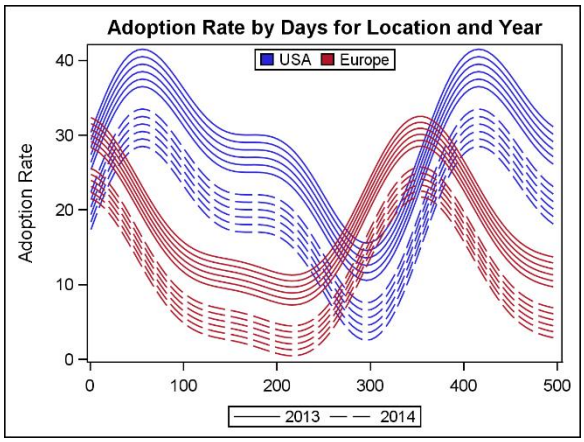


Figure 20 – Spaghetti Plot

SAS 9.4M2 adds support for additional classification grouping for the SERIES plot. These options are GROUPLC and GROUPLP for line color and pattern and GROUPMC and GROUPMS for marker color and symbol. SAS 9.4M2 also provides a TYPE option for the KEYLEGEND statement. This option enables us to create special legends for colors, patterns, and so on. The LINELENGTH option can be used to reduce the length of the line segment in the legend.

```
title 'Adoption Rate by Days for Location and Year';
proc sgplot data=spaghetti;
  series x=x y=y / group=id grouplc=location grouplp=year name='a';
  keylegend 'a' / type=linecolor location=inside position=top;
  keylegend 'a' / type=linepattern linelength=24;
  xaxis display=(nolabel);
run;
```

PROPORTIONAL PANEL

The SGPANEL procedure creates paneled graphs for one or more class variables. Normally, the width and height of each row or column of the panel are equal, and the data is fitted onto each axis.

```
proc sgpanel data=cars noautolegend;
  panelby origin / columns=3
     proportional uniscale=row
     novarname noheader;
  vbar type / group=type <opts>;
  rowaxis display=(nolabel noticks);
  colaxis display=(nolabel noticks);
  inset origin / nolabel  <opts>;
run;
```

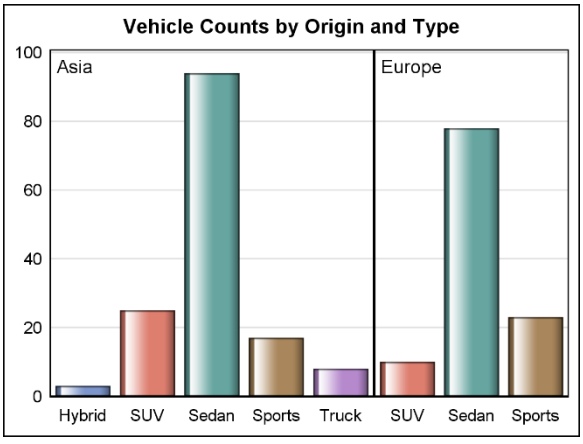


Figure 21 shows a panel with two cells--one for "Asia" and one for "Europe." Each cell contains a vertical bar chart by TYPE. The number of categories in each cell are different. Asia has five categories, and Europe has only three. However, each bar has the same width across all cells, and the width of each cell is based on the number of categories in the cell. This is done by using the PROPORTIONAL option. We have also used the NOHEADER option to suppress headers and the INSET option to add our own insets.

Figure 21 – Proportional Panel

OTHER NEW FEATURES

In addition to the new statements and key features described above, various new features have been added in the SAS 9.4 releases. Some of these features were requested by you and provide a more complete set of features.

AXIS LABELS

Various options were added for placement of axis labels. Now, labels are placed in the center of the data extent of the axis, as can be seen by the x-axis label in Figure 22.

X-axis labels can be placed at the axis center, the data center, to the left, or to the right.

Y-axis labels can be placed at the axis center, the data center, at the top, or at the bottom. For the axis center and the data center, the axis label is oriented vertically. At the top and bottom, the label is oriented horizontally as shown in Figure 22.

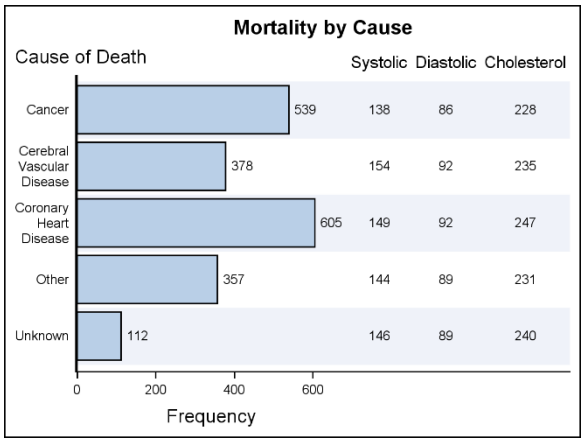


Figure 22- Axis Labels

DISCRETE AXIS COLOR BANDS

Figure 22 also shows the usage of y-axis color bands. These bands help the eye line up items across a wide graph and is especially helpful with columns of data. This feature was also used in Figures 3 and 4. These bands can be displayed for discrete x or y axis using the COLORBANDS option with "ODD" or "EVEN" value.

EQUAL SCALES FOR Y AND Y2 AXES

New options simplify the equalization of scales for the Y and Y2 axes. You can now display labels and values for an axis opposite to the main axis. Equalizing scale on the axis can be done by setting the same variables on both axes.

MINOR GRIDS AND TICKS

Minor grids and ticks are supported for linear, time, and log axes.

BAR CHART BASELINE

The bar charts always have a baseline at zero. This baseline can be changed to any other value on the response axis by using the BASELINE option. The bar is drawn from the baseline to the bar value. Sometimes, for aesthetic reasons you might want to remove the display of the baseline at zero. To do this, you can set the BASELINEATTRS(THICKNESS=0) option.

GRADIENT FILL

Starting in SAS 9.4M2, a new visual option for FILLTYPE is available for bar charts and histograms. This new option draws bars with an alpha gradient, starting with the set bar transparency at the data value and fading to fully transparent at the baseline. This transparency can be combined with data skins to provide a different aesthetic look.

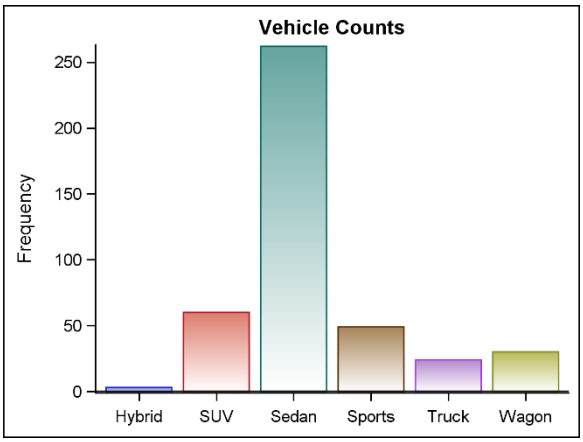


Figure 23 – Gradient Fills

SEGMENT LABELS

Displaying segment values in a stacked bar chart has been possible by overlaying a scatter plot with the MARKERCHAR option in a VBARPARM statement.

However, this requires some extra work to compute the exact location of each label. Also, segment labels can be a problem if the segment is not big enough to fit the label, and sometimes a busy graph is the result.

Now, you can request segment labels by using a simple option. Segment and overall bar labels can be requested independently as shown in Figure 24.

Options are also provided to manage the display of the labels when they do not fit within the segment.

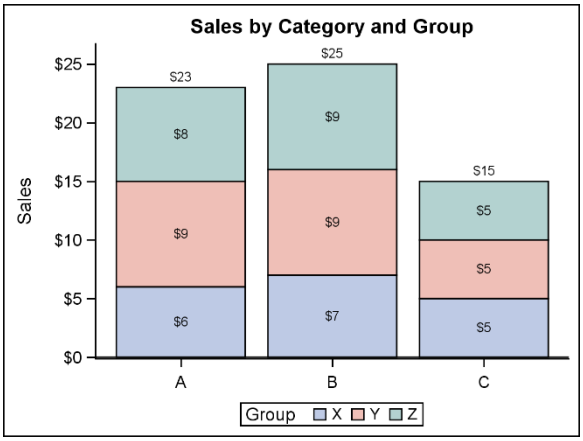


Figure 24 – Segment Labels

CONCLUSION

Many new features have been added to the SG procedures in SAS 9.4, SAS 9.4M1 and SAS 9.4M2. These features include new plot statements such as AXISTABLE and convenience statements for management of group attributes and for the creation of new symbols. New features were also added to manage axis tick values and labels and to overlay scatter markers on box plots.

With these additions, creating graphs is easier. Many items that previously required custom coding or annotation can now be done using a statement or an option. While annotation is still available, our goal is to make its usage really rare.

RESOURCES

The PDF file of this paper and the SAS code for all the programs is available at the links below:

Paper: http://support.sas.com/resources/papers/proceedings15/SAS1780-2015.pdf

Code: http://support.sas.com/resources/papers/proceedings15/SAS1780-2015.zip

REFERENCES

Matange, Sanjay and Dan Heath. 2011. *Statistical Graphics Procedures by Example: Effective Graphics using SAS*: Cary, NC: SAS Institute Inc.

Heath, Dan. 2014. "Putting on the Ritz: New Ways to Style your ODS Graphics to the Max." *Proceedings of the SAS Global Forum 2014 Conference*. Cary, NC. SAS Institute Inc.

Matange, Sanjay. 2014. Graphically Speaking blog. SAS Institute Inc., Available at http://blogs.sas.com/content/graphicallyspeaking/

RECOMMENDED READING

- *Base SAS® Procedures Guide*
- *Statistical Graphics Procedures by Example: Effective Graphics using SAS* by Sanjay Matange and Dan Heath

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author:

Sanjay Matange
100 SAS Campus Drive
Cary, NC 27513
SAS Institute Inc.
Sanjay.Matange@sas.com
http://www.sas.com