Paper 088-2007

Using ODS Styles with SAS/GRAPH®

Jeff Cartier and Dan Heath, SAS Institute, Cary, NC

INTRODUCTION

Today, most SAS® users are taking advantage of the Output Delivery System (ODS) to produce documents that contain output from SAS procedures. Users are aware of the existence of ODS styles and how a style can be specified to alter the fonts, colors, and other appearance aspects of their tabular output. The good news is that, in SAS® 9.2, all SAS/GRAPH graphical output can now be formatted in a similar fashion with an ODS style. ODS styles have been extended to include elements that affect graphical procedure output as well as tabular output. This paper shows how easy it is to apply any of the supplied style definitions to SAS/GRAPH, SAS/STAT®, and SAS/ETS® output. You will also see how SAS/GRAPH coding and supplied STATGRAPH templates interact with information supplied by a style.

THE NEW GSTYLE SAS OPTION

In SAS 9.2, a new SAS system option GSTYLE has been added. By default, this option is set to GSTYLE. This option applies to the GRSEG output from these existing SAS/GRAPH procedures: GCHART, GPLOT, GMAP, GBARLINE, G3D, GCONTOUR, and GRADAR. Figures 1 and 2 show the output from a simple GBARLINE procedure using this option. Some of the differences you may notice are different fonts, different levels of font sizes and weights, different bar colors, and different plot line colors. There were no additional options (or goptions) in the program that specified the colors or fonts to be used–just whether GSTYLE was in effect or not.

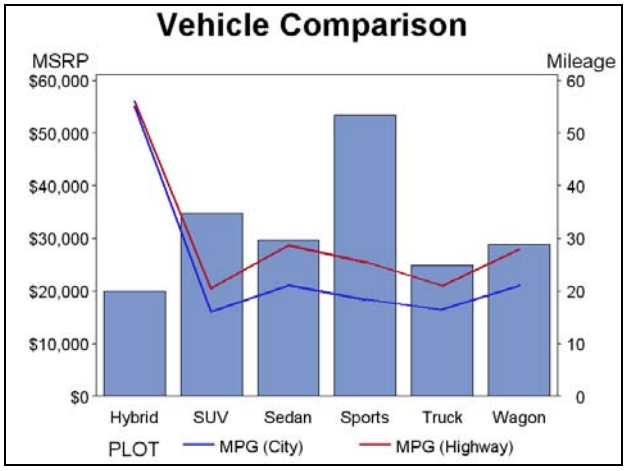


Figure 1. GSTYLE Option

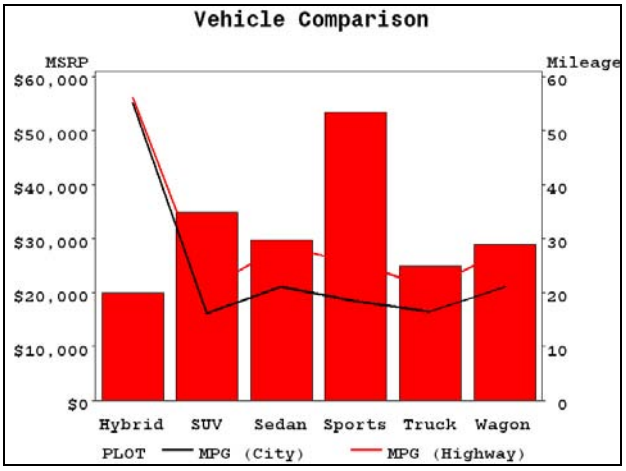


Figure 2. NOGSTYLE Option

Reasons for introducing the GSTYLE option include:

- Integrating SAS/GRAPH output better in ODS output that includes tables.
- Creating attractive graphs by default.
- Simplifying SAS/GRAPH coding by eliminating the need to add many graphical options just to get attractive colors and fonts. Graphical appearance options can still be added when desired to augment the style.
- Making styles available for all device drivers, not just ACTIVEX and JAVA.[1]
- Allowing for the SAS® 9.1 and earlier coding techniques.

What impact will this have on existing SAS/GRAPH programs that use many options to adjust colors and fonts?

- Existing options that modify colors and fonts continue to work whenever coded. They simply act as an override to any corresponding information supplied by the style in effect. For some existing programs, an undesirable combination of colors or fonts may result.
- For complete compatibility with SAS 9.1 behavior, you should set the NOSTYLE option in the program.

[1] The ACTIVEX, ACTXIMG, JAVA, and JAVAIMG devices always use styles, regardless of the GSTYLE setting.

WHAT STYLES ARE AVAILABLE AND HOW DO I USE THEM?

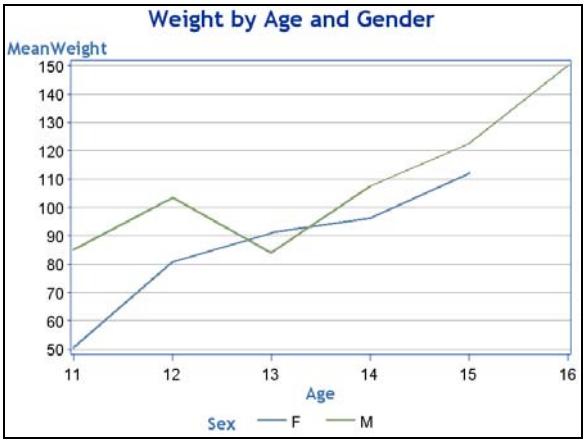
Using ODS styles is simple–you include an ODS statement indicating the destination and a style is assigned by default. You change the style by adding a STYLE= option to the ODS destination statement. Here are the defaults:

ODS Destination	Default Style *	Default Device Driver
LISTING	Listing	Host driver (WIN, XCOLOR, and so on)
HTML	Default	PNG
PDF	Printer	SASPRTC
PRINTER	Printer	SASPRTC
RTF	RTF	SASEMF

* The default style for each destination is set in the SAS Registry.

Example: Produce a graph and table in the RTF destination with the Normal style (see Figure 3)

```
ods rtf   style=normal;
proc means data=sashelp.class maxdec=1 mean nway;
  class age sex;
  var weight;
  output out=class mean=MeanWeight;
run;
title 'Weight by Age and Gender';
symbol i=join;
proc gplot data=class;
  plot MeanWeight*age=sex / autovref;
run; quit;
ods rtf close;
```



Analysis Variable : Weight			
Age	**Sex**	**N Obs**	**Mean**
11	F	1	50.5
	M	1	85.0
12	F	2	80.8
	M	3	103.5
13	F	2	91.0
	M	1	84.0
14	F	2	96.3
	M	2	107.5
15	F	2	112.3
	M	2	122.5
16	M	1	150.0

Figure 3. Graph and Table Output

There are more than 50 ODS styles supplied in SAS 9.2. Here are some of the most commonly used:

ODS Style	Parent Style	Font Family	Images	Color Scheme	Wall	Background
Analysis	Default	<sans-serif>	No	Zuni	White	Light
Default		<sans-serif>	No	Terra	White	Gray
Journal	Default	<sans-serif>	No	Grayscale	White	White
Listing	Default	<sans-serif>	No	Terra	White	White
Normal	Default	Trebuchet MS	No	Seaside	White	White
Printer	Default	<serif>	No	Terra	White	White
RTF	Printer	<serif>	No	Terra	White	White
Statistical	Default	<sans-serif>	No	Deco	White	White

This example shows a choropleth map created with PROC GMAP where a continuous variable (population) has been binned into ten ranges and a unique color assigned to each level. With NOGSTYLE, the colors are defined by the current color list (Figure 4). Notice how you must keep referring to the legend to get any sense of population density. With GSTYLE, a *color ramp* defined within the style establishes a start and end color (Figure 5). New colors (based on the number of levels requested) are computed by the procedure and applied as needed. Notice how much easier it is to visualize the relative state populations, even without the legend.

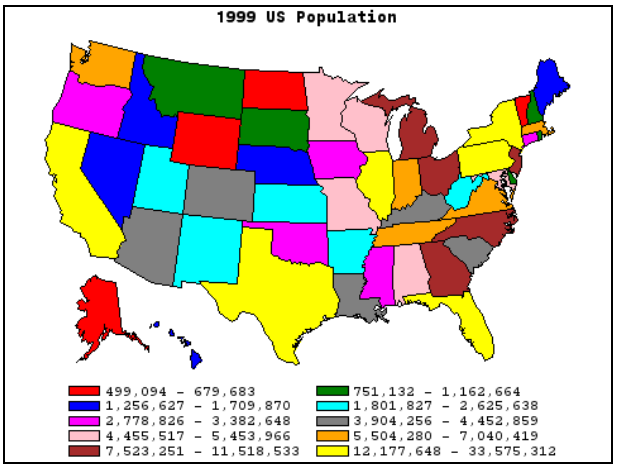


Figure 4. NOGSTYLE Option

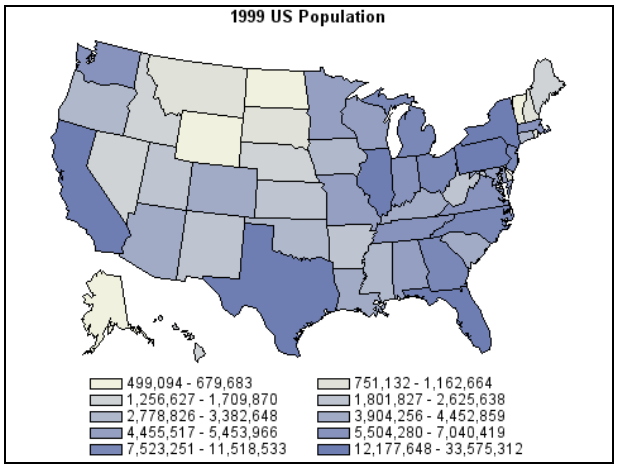


Figure 5. GSTYLE Option

WRITING SAS/GRAPH PROGRAMS WITH STYLES IN MIND

Recall the SAS/GRAPH coding from an earlier example:

```
proc gplot data=class;
   plot MeanWeight*age=sex / autovref;
run; quit;
```

Notice that this program does NOT contain any of the numerous SAS/GRAPH options that change the colors or fonts of the output. If any of these options were to appear in the program, they would have precedence over any style attribute that may address the same feature. In general, a style does not enable a graphical feature–you must do this in your SAS/GRAPH program. In this example, the style does not turn on reference lines, but it does control the line style, thickness, and color of these lines once they are enabled with AUTOVREF. Another example is the FRAME option used by GCHART and GPLOT action statements. In general, you only need to enable or disable this feature. If you use CFRAME to turn on the frame you will also override the color defined in the style. Here is a list of some common SAS/GRAPH options that affect the same graph features that a style does:

GOPTIONS	SYMBOL / PATTERN	AXIS	LEGEND	GPLOT – PLOT / BUBBLE	GCHART – VBAR / HBAR
COLORS CBACK CTEXT CTITLE CSYMBOL CPATTERN FTEXT FTITLE	COLOR	COLOR STYLE WIDTH LABEL=(COLOR FONT) VALUE=(COLOR FONT)	CBACK CFRAME CBORDER CSHADOW CBLOCK LABEL=(COLOR FONT) VALUE=(COLOR FONT)	CAXIS CFRAME CTEXT CAUTOHREF CAUTOVREF LAUTOVREF LAUTOHREF	CAXIS CFRAME CTEXT CAUTOREF LAUTOREF COUTLINE

NEW SAS/GRAPH PROCEDURES INCORPORATE STYLES

SAS 9.2 introduces several new procedures that are designed to use ODS styles.

SGPLOT Creates plots such as scatter, series, fit line, box, dot, and band. Creates charts such as histogram, bar, and line. Allows these plots and charts to be overlaid.

SGSCATTER Creates comparative scatter plots and scatter plot matrices. Confidence ellipses and fit lines can be added.

SGPANEL Creates a data-driven grid (trellis) of plots and charts.

SGRENDER Renders the graph defined with the Graph Template Language.

The SGPLOT procedure creates an assortment of graphs including horizontal and vertical box plots (Figure 6).

```
ods rtf style=analysis;

title 'Vehicle Weights';
proc sgplot data=sashelp.cars;
  vbox weight / category=type
                datalabel=model;
run;

ods rtf close;
```

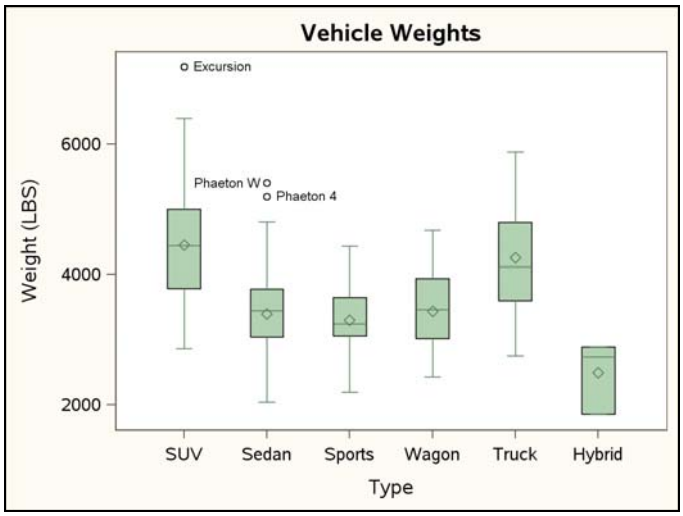


Figure 6. Box Plots from PROC SGPLOT

The SGPANEL procedure offers dynamic gridding of plots and charts (Figure 7).

```
proc sort data=sashelp.cars out=cars;
  by origin;
run;

ods rtf style=statistical;

proc sgpanel data=cars;

  panelby origin / layout=panel
                   novarname columns=1;

  dot type / response=msrp stat=mean
             limitstat=clm  limits=both
             name='dot' legendlabel=
             'Mean with CLM (alpha=.05)';

  discretelegend 'dot';
run;

ods rtf close;
```

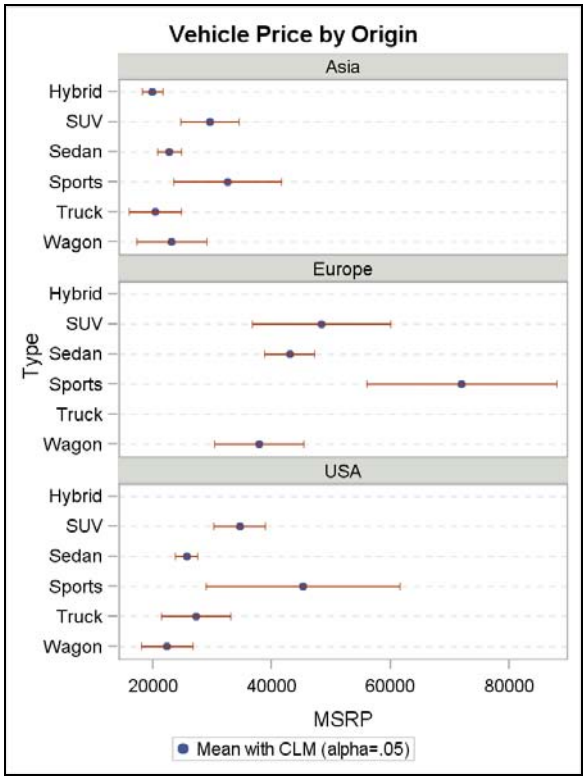


Figure 7. Dot Plots from PROC SGPANEL

The underlying technology for these new procedures is very different from traditional SAS/GRAPH procedures and consequently there are many differences in the way things work:

- TITLE*n*, FOOTNOTE*n*, FORMAT, and LABEL statements are supported.
- Global statements such as GOPTIONS, AXIS, LEGEND, SYMBOL, or PATTERN are not supported. Instead, statements such as DISCRETELEGEND, XAXIS and YAXIS can be used as procedure statements.
- Output is image-based and no GRSEG is produced. There is no notion of device driver.
- The ODS GRAPHICS statement is used for output control (image name, size, format, and so on)
- The GSYTLE option has no effect on their output because styles are always used.

ODS STATISTICAL GRAPHICS AND STYLES

You are probably familiar with ODS Statistical Graphics, a new form of graphics introduced in SAS®9 by SAS/STAT and SAS/ETS procedures:

- This software is production in SAS 9.2.
- Graphs are not produced by default. You must enable or disable graphics with the ODS GRAPHICS statement. When graphics is enabled, procedures produce a set of standard graphs.
- To further customize graphical output, procedures support a PLOTS= option to request specific graphs and plotting options for them.
- As with the new SAS/GRAPH procedures, ODS styles are fully supported (Figure 8).

```
ods rtf style=listing;
ods graphics on / width=3.5in;

proc lifetest data=exposed method=km;
  time days*status(0);
  strata treatment;
  survival confband=hw
           plots=(survival(atrisk cl));
run;

ods graphics off;
ods rtf close;
```

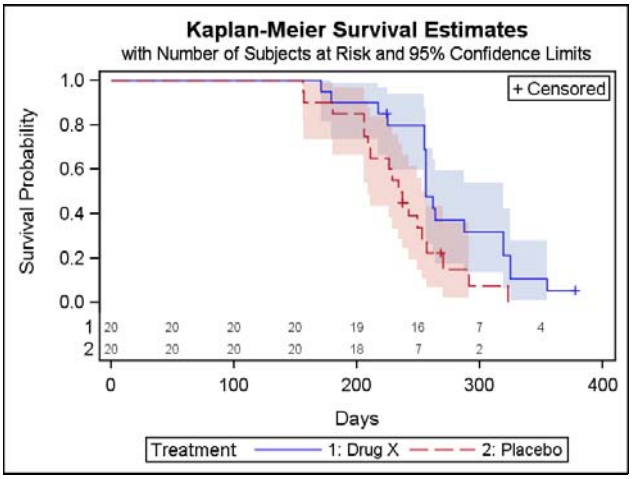


Figure 8. Survival Plot from PROC LIFETEST

There are grayscale styles as well as color styles. Shown below is a fit plot from PROC REG using the Journal and Journal2 styles (Figures 9 and 10). The styles are identical except that the Journal2 style uses minimal ink and does not fill any areas such as confidence bands, ellipses, histograms, or contours.

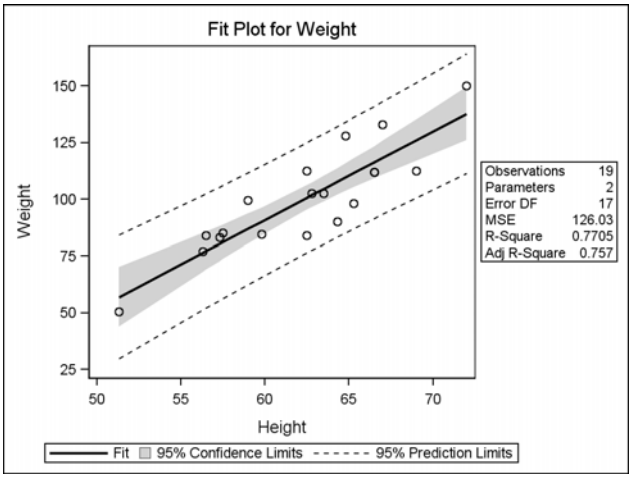


Figure 9. Journal Style

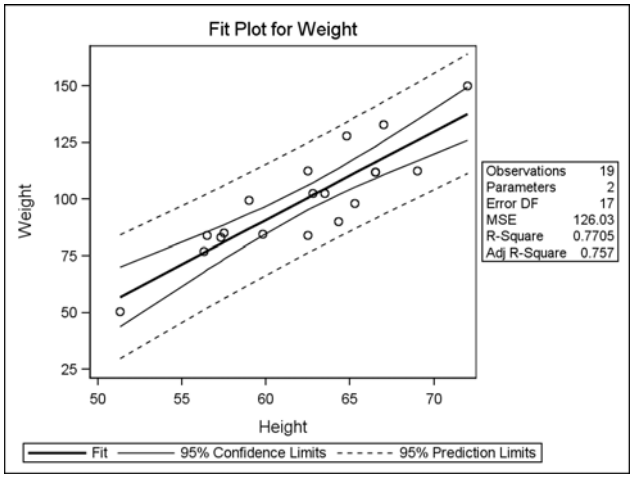


Figure 10. Journal2 Style

A guiding philosophy behind the ODS Statistical Graphics project has been to let the style dictate the overall appearance of the graph. To achieve this goal, several new ODS style elements have been added to all styles to allow more customization of graphical appearance for specific types of plots and common plot features such as confidence lines and bands, fit lines, histograms, box plots, ellipses, and contours. Some of these new style features will be discussed later.

CREATING CUSTOM STYLES

The following sections show some of the graphical style elements and the graphical features they control. The examples show how you can change existing styles to create new appearance schemes.

CHANGING FONTS

There are specific fonts assigned to various parts of graphs and tables. Figure 11 shows several of these.

GraphTitleFont	all titles
GraphFootnoteFont	all footnotes
GraphLabelFont	axis labels, legend title
GraphValueFont	axis tick values, legend values, inset text
GraphDataFont	data values in graph
GraphUnicodeFont	font for special glyphs
GraphAnnoFont	font for editor annotation

The values of the font family can be set as actual font names or as references to font names defined in the SAS Registry. For example, <sans-serif> translates to 'Arial' in many locales.

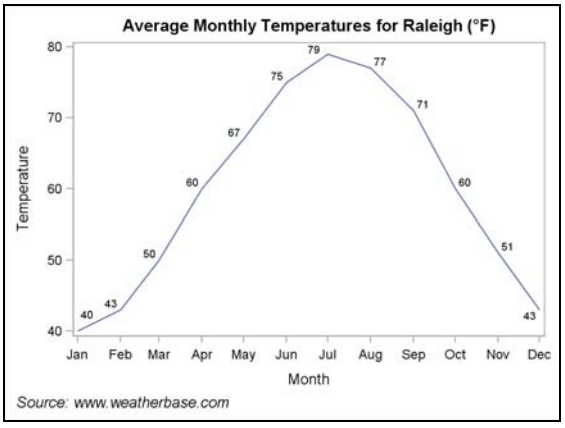


Figure 11. Plot Showing Use of Five Font Styles

```
proc template;
   define style Styles.Statistical;
      parent = styles.default;
      style GraphFonts /
         'GraphTitleFont' = ("<sans-serif>, <MTsans-serif>",11pt,bold)
         'GraphFootnoteFont' = ("<sans-serif>, <MTsans-serif>",10pt,italic)
         'GraphLabelFont' = ("<sans-serif>, <MTsans-serif>",10pt)
         'GraphValueFont' = ("<sans-serif>, <MTsans-serif>",9pt)
         'GraphDataFont' = ("<sans-serif>, <MTsans-serif>",7pt)
         'GraphUnicodeFont' = ("<MTsans-serif-unicode>",9pt)
         'GraphAnnoFont' = ("<sans-serif>, <MTsans-serif>",10pt);
      /* more elements */
   end;
run;
```

The **GraphFonts** style element could be changed to use any font defined on your computer. When making font changes, be sure to make corresponding changes to the Fonts style element, which controls the fonts in other parts of the output. You can adjust font size, weight (`bold`), and style (`italic`), if desired. Multiple font families can be listed in a specific to generic order to allow for appropriate substitutions when a specific font is not installed. The font references prefixed with "MT" refer to the new Monotype series of fonts that are part of every SAS 9.2 installation.

```
proc template;
   define style Styles.Statistical2;
      parent = styles.Statistical;
      style GraphFonts /
         'GraphTitleFont' = ("'Century', <MTserif>",12pt,bold)
         'GraphFootnoteFont' = ("'Century', <MTserif>",11pt,italic)
         'GraphLabelFont' = ("'Century', <MTserif>",10pt,bold)
         'GraphValueFont' = ("'Century', <MTserif>",9pt)
         'GraphDataFont' = ("'Century', <MTserif>",7pt)
         'GraphUnicodeFont' = ("'Century', <MTserif-unicode>",9pt)
         'GraphAnnoFont' = ("'Century', <MTserif>",,10pt);
      /* more elements */
   end;
run;
```

The attributes defined by the **GraphFonts** and **Fonts** elements are referenced in other style elements that associate both a font and a color to some text usage. Typically, there is no need to change these elements.

```
style GraphLabelText /
   font =
GraphFonts('GraphLabelFont')
   color = GraphColors('glabel');
```

MAPPING COLORS, MARKERS AND LINES TO DATA

Every line, filled area, or marker in the graph is assigned its visual attributes from predefined style elements.

For non-grouped data, the **GraphDataDefault** style element defines the attributes for all non-specialized[2] lines, markers, filled areas, and color ramps (Figure 11).

For grouped data, the style elements **GraphData1** - **GraphData12** define line, marker, and fill attributes (Figure 12).

For most styles, color, line style, and marker symbol change per group value for maximum visual discrimination.

Two color attributes are assigned to these elements:

- ConstrastColor affects color for lines and markers.
- Color affects filled areas only.

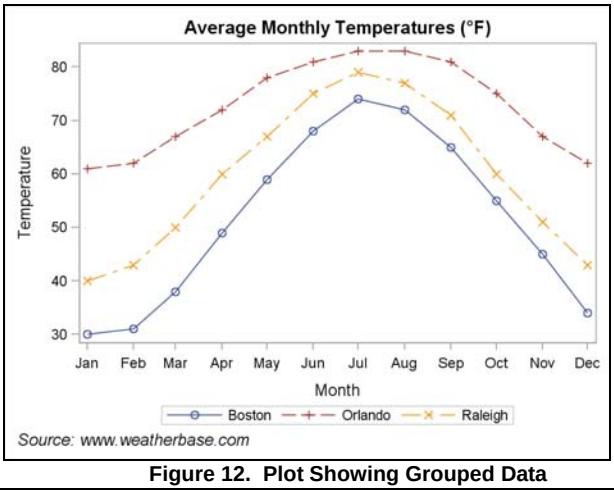


Figure 12. Plot Showing Grouped Data

This is the definition of the **GraphData1** - **GraphData3** style elements in the DEFAULT style.

```
proc template;
 define style styles.default;

 class GraphData1 /
    markersymbol = "circle"
    linestyle = 1
    contrastcolor =
GraphColors('gcdata1')
    color = GraphColors('gdata1');

 class GraphData2 /
    markersymbol = "plus"
    linestyle = 4
    contrastcolor =
GraphColors('gcdata2')
    color = GraphColors('gdata2');

class GraphData3 /
    markersymbol = "X"
    linestyle = 8
    contrastcolor =
GraphColors('gcdata3')
    color = GraphColors('gdata3');

 /* more elements */
 end;
run;
```

To hold the line style constant and allow only color and marker changes per group value, you could modify the style. `Linestyle=1` is a solid line.

```
proc template;
 define style mydefault;
   parent = styles.default;

 style GraphData1 from GraphData1 /
    linestyle = 1;

 style GraphData2 from GraphData2 /
    linestyle = 1;

 style GraphData3 from GraphData3 /
    linestyle = 1

 /* more elements */
 end;
run;
```

`GraphData1 from GraphData1` indicates that the attributes that do not appear are inherited from the parent style element. This avoids having to assign values to all attributes.

MODIFYING THE APPEARANCE OF SPECIFIC PLOT TYPES

There are style elements that apply only to certain plot types. For example,

```
style GraphContour /
    startcolor = GraphColors('gramp3cstart')
    neutralcolor = GraphColors('gramp3cneutral')
    endcolor = GraphColors('gramp3cend')
    displayopts = "LabeledLineGradient";
```

[2] Specialized data-related items include reference lines, fit lines, confidence lines/bands, etc. These have their own style elements.

The StartColor, NeutralColor, and EndColor attributes define a color ramp. The DisplayOpts attribute defines whether the contour is filled, whether contour lines appear, and whether the lines are labeled (Figure 13).

`DisplayOpts="LabeledLineGradient"` `DisplayOpts="LabeledLineFill"` `DisplayOpts="LabeledLine"`

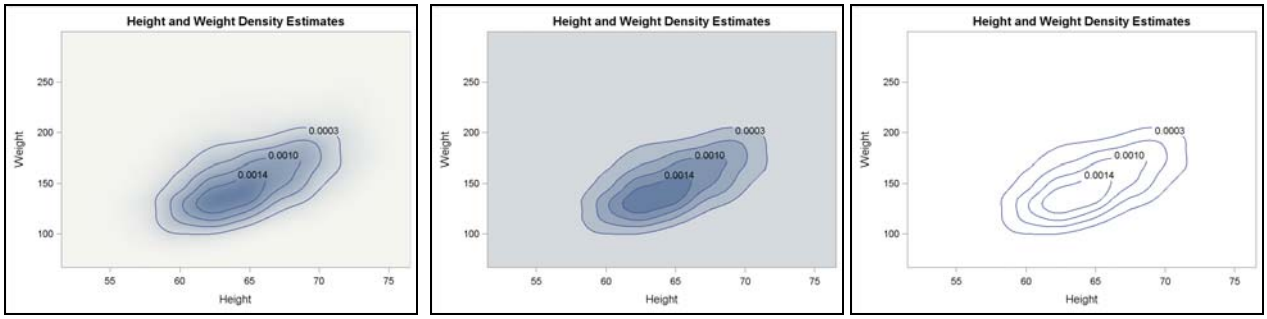


Figure 13. Presentation Variations for Contour Plots

Box plots have seven style elements that affect such things as box outline, whiskers, median, mean, outliers, and connect line. The **GraphBox** style element controls which box plot features are displayed, the statistic to be connected if a connect line is shown, and the style of the whisker cap (Figure 14).

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

CAPSTYLE = "SERIF" | "LINE" | "BRACKET"
CONNECT = "MEAN" | "MEDIAN" | "MIN" | "MAX" | "Q1" | " Q3"
DISPLAYOPTS ="FILL | CAPS | MEDIAN | MEAN | OUTLIERS | CONNECT | NOTCHES"

Default Box Plot Display Notches and Connect Line Added Median Only

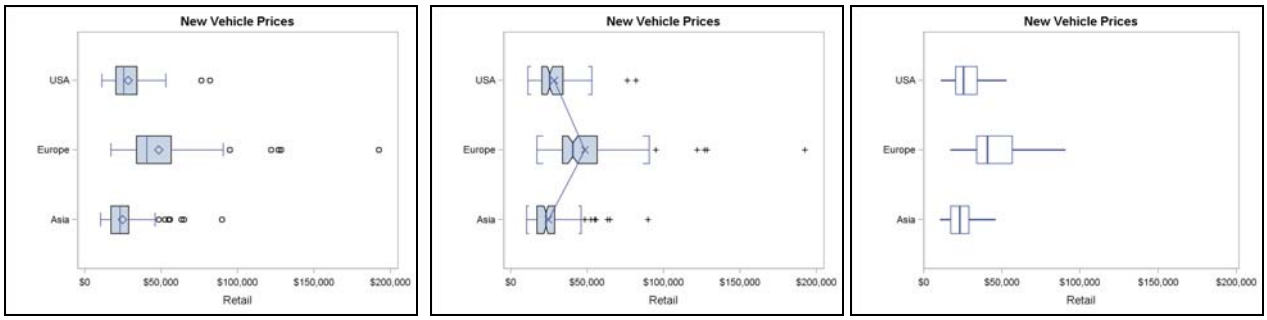


Figure 14. Presentation Variations for Box Plots

CONCLUSION

ODS styles offer a way to get professional looking output from SAS software. They provide a coordinated appearance scheme for tables and graphs that is consistent whether using HTML, RTF, PDF, or other output formats. In SAS 9.2, the primary SAS/GRAPH procedures now support styles for GRSEG output. New graphical procedures all fully support styles. Style elements have been added to offer finer levels of control over graphs. New style elements are added to provide more choices for appearance schemes. ODS documentation will now include thorough reference and usage examples on style elements and helpful tips on how to modify them.

Benefits from adopting a style-based approach to producing reports include

- Reduced program size. Programming options that would normally be needed to specify output appearance can be omitted. Programs can now be simplified to focus on information content.
- Reduced time spent juggling appearance option values to get attractive output.