Paper 198-2023

# Getting Started with the SGPLOT Procedure

Joshua M. Horstman, Nested Loop Consulting

## ABSTRACT

Do you want to create highly-customizable, publication-ready graphics in just minutes using SAS®? This workshop introduces the SGPLOT procedure, which is part of ODS Statistical Graphics, included in Base SAS®. Starting with the basic building blocks, you can construct basic plots and charts in no time. We work through several plot types, and you learn some simple ways to customize each one.

## INTRODUCTION

The SGPLOT procedure is the workhorse for producing single-cell plots in modern SAS environments. This paper covers the basic syntax of the SGPLOT procedure and provides a series of exercises that can be used to learn its basic functionality and features. This paper is intended as a companion to a hands-on workshop taught in a live classroom setting, but it can be used on its own for independent study.

## BACKGROUND ON THE SGPLOT PROCEDURE

### THE OUTPUT DELIVERY SYSTEM (ODS)

The Output Delivery System (ODS) is a component of the SAS system that allows for extensive customization of the format and appearance of output generated by SAS. Prior to the development of ODS, output from SAS was limited to a text-based listing output.

With ODS, the SAS programmer can create output in many common formats such as PDF, RTF, HTML, Microsoft Office file formats like Excel and PowerPoint, and more. This output can make use of complex layouts, graphics, colors, fonts, and other visual elements to meet nearly any need. ODS has been part of the Base SAS product since version 7, so no separate license is required.

### ODS STATISTICAL GRAPHICS

ODS Statistical Graphics is an extension to ODS used to create analytical graphs. It was introduced as part of SAS/GRAPH in SAS 9.2, but was moved into the Base SAS product starting with version 9.3.

ODS Statistical Graphics are based on templates created using the Graph Template Language (GTL). Fortunately, it is not necessary to learn the details of GTL to produce highly customized graphs because much of this functionality is available to the SAS programmer through a procedural interface known as the Statistical Graphics (SG) procedures. Using these procedures, publication-quality graphs can be created with just a few lines of code.

These procedures should not be confused with the older SAS/GRAPH procedures such as GPLOT, GCHART, and GMAP. This is an entirely different graphing system with its own statements and options.

### THE SGPLOT PROCEDURE

The SGPLOT procedure is one of the SG procedures that comprise the ODS Statistical Graphics package. It is used to create single-cell plots of many different types. These include scatter plots, bar charts, box plots, bubble plots, line charts, heat maps, histograms, and many more.

Here is the basic syntax of the SGPLOT procedure:

```
proc sgplot data=<input-data-set> <options>;
  <one or more plot requests>
  <other optional statements>
run;
```

We start with the SGPLOT statement itself. This allows us to specify an input data set as well as numerous other procedure options.

Next, we include one or more plot request statements. There are dozens of plot request statements available. Some of these include SCATTER, SERIES, VBOX, VBAR, HIGHLOW, and BUBBLE. We'll examine these and others as we progress through the exercises. We'll also see how we can combine multiple plot requests into a single plot.

Finally, there are several optional statements that control certain plot features such as XAXIS, YAXIS, REFLINE, INSET, and KEYLEGEND. These statements are not covered in this workshop, but some are included in the companion workshop, "Doing More with the SGPLOT Procedure" (Horstman 2018).

### ODS DESTINATIONS

To create ODS graphs, a valid ODS destination must be open when the graph procedure is executed. For example, to invoke the SGPLOT procedure and direct the output to a PDF file, the ODS PDF statement is used to open and close the file as follows:

```
ods pdf file="c:\example.pdf";
  <SG procedure code goes here...>;
ods pdf close;
```

There are similar statements associated with other ODS destinations such as ODS HTML and ODS RTF. You can also have multiple destinations open simultaneously if you wish.

## ABOUT THE EXERCISES

### USING THE EXERCISES

These exercises were created as part of a hands-on workshop to be presented in a classroom setting. If you are using them on your own, it is recommended that you progress through them sequentially as they build on each other. To maximize your learning, try to complete each exercise on your own before looking at the solution provided. Also, keep in mind that there are often multiple ways to perform a task in SAS, so the code provided may not be the only correct solution.

### EXAMPLE DATA SETS

Throughout this workshop, we will make use of several data sets from the SASHELP library. These data sets are included with SAS, which means these exercises should work anywhere you have SAS installed.

We will use the following data sets:

- SASHELP.CLASS – demographics on 19 students in a grade school classroom
- SASHELP.CARS – technical data about 428 car models
- SASHELP.HEART – health information regarding 5,209 patients in a heart study
- SASHELP.STOCKS – monthly stock data for three companies over ten years

Take a few moments to familiarize yourself with these data sets before proceeding with the exercises.

## EXERCISE 1: BASIC SCATTER PLOT

### THE SCATTER STATEMENT

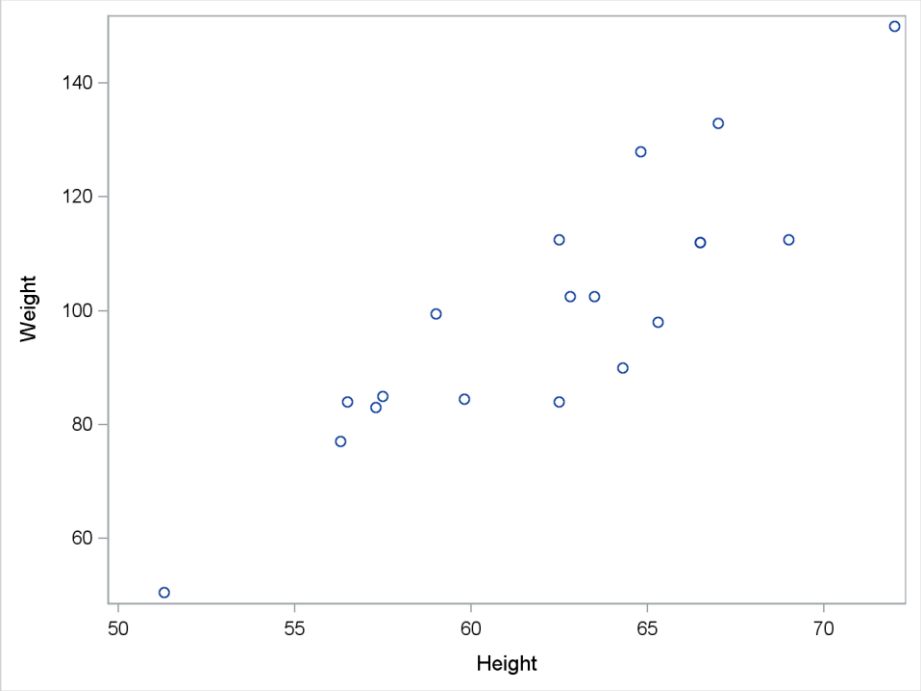
The first plot request we will try is the SCATTER statement, which is used to create a scatter plot. There are two required arguments, X= and Y=, which specify the variables to plot. Here is the syntax:

```
proc sgplot data=<input-data-set> <options>;
  scatter x=variable y=variable < / options>;
run;
```

Notice that there are additional options available on the SCATTER statement, but they won't be needed for the first exercise.

### EXERCISE

Using the SASHELP.CLASS data set, create a scatter plot of WEIGHT vs. HEIGHT. You will need to use a SCATTER statement with the required X= and Y= arguments. Refer to the syntax above. Your output should resemble the one shown below.




**Exercise 1. Basic Scatter Plot**

### SOLUTION

```
proc sgplot data=sashelp.class;
  scatter x=height y=weight;
run;
```

## EXERCISE 2: GROUPED SCATTER PLOT

### THE GROUP= OPTION

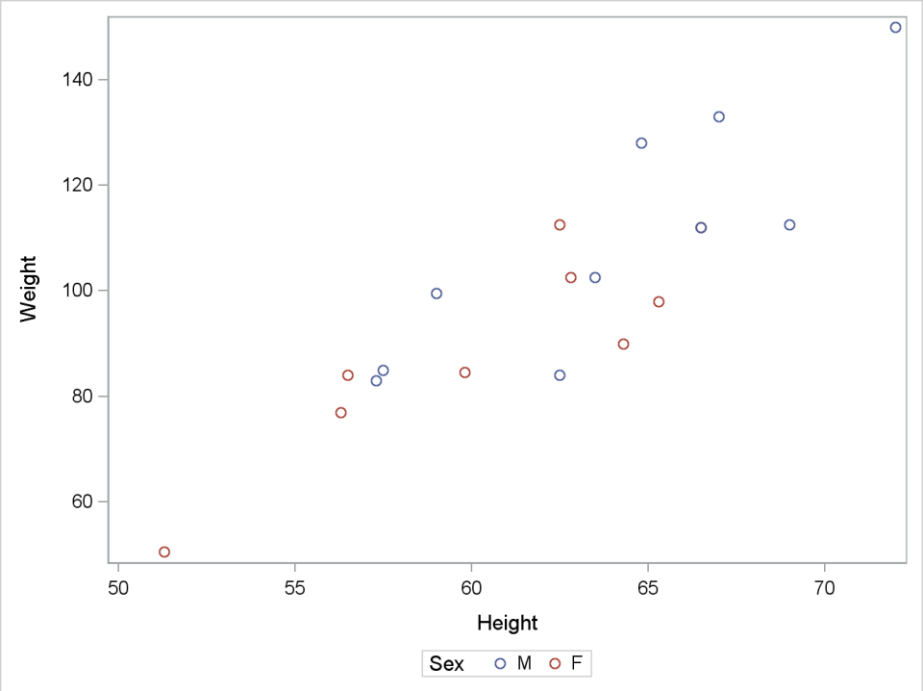
In many cases, our data contain values from multiple groups we wish to compare. The GROUP= option allows us to specify a grouping variable and is available on many different plot requests, including the SCATTER statement. When you use a grouping variable, plot elements will be given different visual attributes for each unique value of the grouping variable.

```
proc sgplot data=<input-data-set> <options>;
  scatter x=variable y=variable
          / group=variable <more options>;
run;
```

The slash character is used to separate the required arguments from any options. Because GROUP= is optional, it comes after the slash along with any other options desired.

### EXERCISE

Modify the scatter plot from Exercise 1 to use SEX as a grouping variable. You'll need to add the GROUP= option to the SCATTER statement. Don't forget that it must come after the slash.




**Exercise 2. Grouped Scatter Plot**

### SOLUTION

```
proc sgplot data=sashelp.class;
  scatter x=height y=weight / group=sex;
run;
```

Notice how SGPLOT automatically includes a legend when we add a grouping variable. What happens if you add the option DATALABEL=NAME?

## EXERCISE 3: GROUPED BUBBLE PLOT

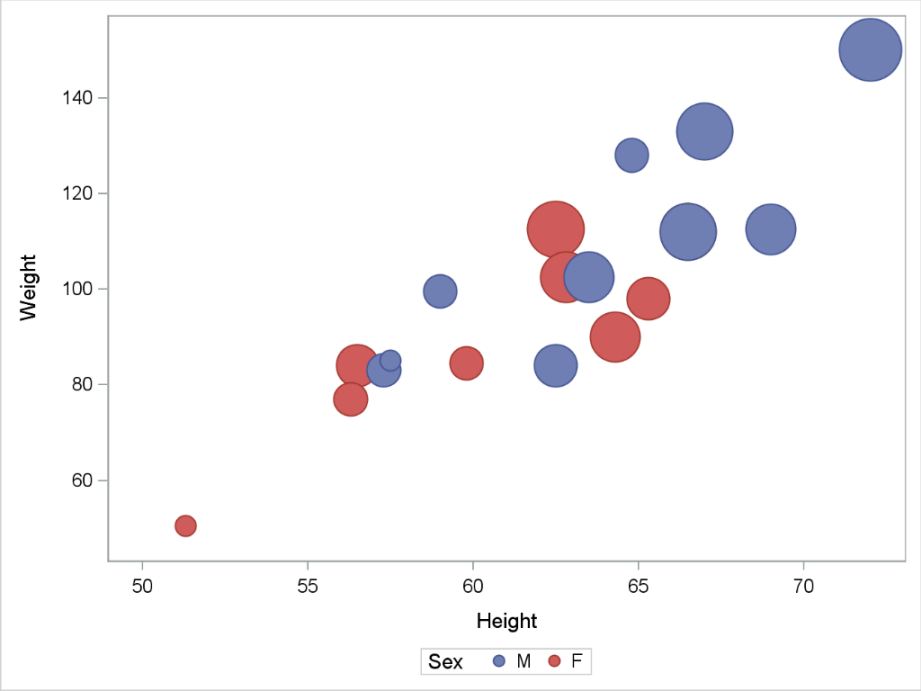
### THE BUBBLE STATEMENT

Bubble plots provide a way to visualize the relationships between three different variables and can be requested with the BUBBLE statement. In addition to X= and Y=, the SIZE= argument is required. It specifies a variable that controls the size of the bubbles.

```
proc sgplot data=<input-data-set> <options>;
  bubble x=variable y=variable size=variable < / options>;
run;
```

### EXERCISE

Using the SASHELP.CLASS data set, create a bubble plot of WEIGHT vs HEIGHT, grouped by SEX with bubbles sized by AGE. You will need to use the BUBBLE statement, the required arguments X=, Y=, and SIZE=, and the GROUP= option.




**Exercise 3. Grouped Bubble Plot**

### SOLUTION

```
proc sgplot data=sashelp.class;
  bubble x=height y=weight size=age / group=sex;
run;
```

## EXERCISE 4: GROUPED SERIES PLOT

### THE SERIES STATEMENT

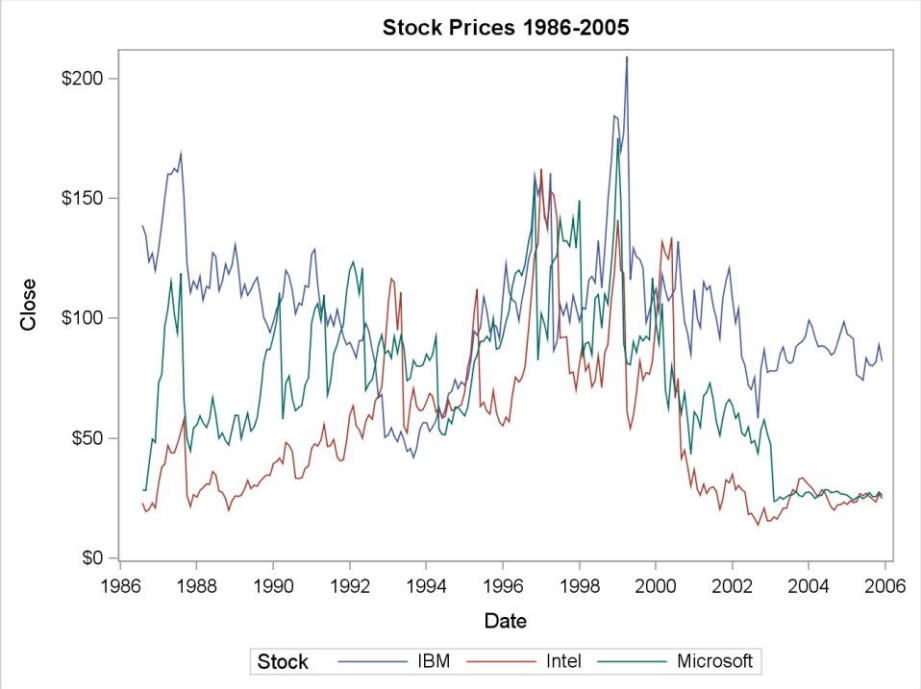
The SERIES statement is used to request a line plot. The individual data points are connected by line segments. A grouping variable results in multiple lines.

```
proc sgplot data=<input-data-set> <options>;
  series x=variable y=variable < / options>;
run;
```

As with the SCATTER statement, the required arguments are X= and Y=. By default, only the lines are shown, not the data values themselves. To add plot markers at each value, use the MARKERS option.

### EXERCISE

Using the SASHELP.STOCKS data set, create a series plot of closing price (CLOSE) by date (DATE), grouped by company (STOCK). Use a title statement to add a title to the plot.




**Exercise 4. Grouped Series Plot**

### SOLUTION

```
proc sgplot data=sashelp.stocks;
  title "Stock Prices 1986-2005";
  series x=date y=close / group=stock;
run;
```

## EXERCISE 5: HIGH-LOW PLOT

### THE HIGHLOW STATEMENT

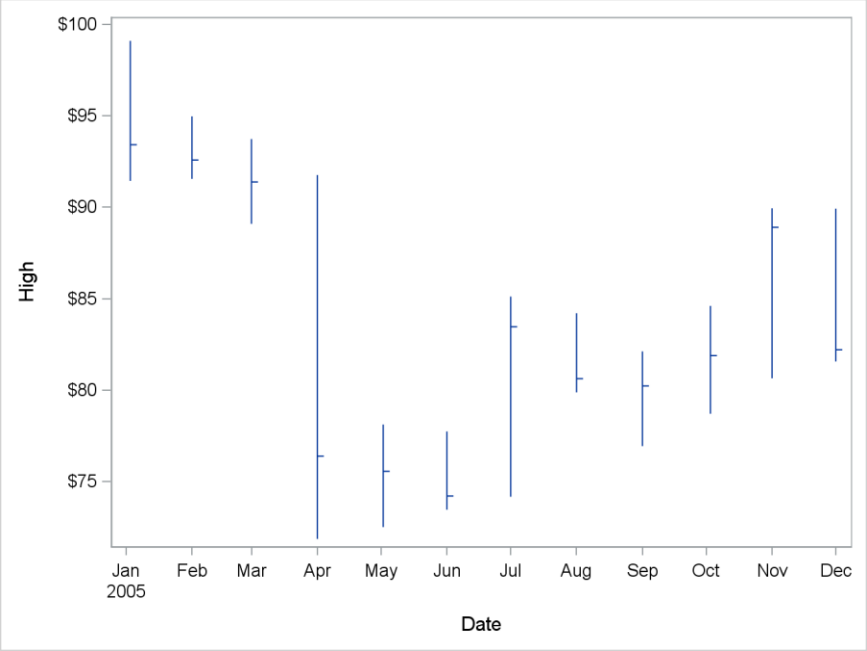
The HIGHLOW statement creates a plot consisting of floating vertical or horizontal line segments. Each line segment connects a high value with a low value. These values are specified by the two required arguments HIGH= and LOW=. You must also include either the X= or Y= argument. Using the X= argument results in vertical line segments, while the Y= argument creates horizontal line segments.

```
proc sgplot data=<input-data-set> <options>;
  highlow x=variable | y=variable
       high=variable low=variable < / options>;
run;
```

The optional argument CLOSE= is used to specify a variable that contains values for closing tick marks. This is commonly used when presenting stock data where the HIGH= and LOW= variables represent the high and low price of a stock over some period and a closing tick denotes the final price.

### EXERCISE

From the SASHELP.STOCKS data set, create a high-low plot of monthly stock prices with closing ticks for the stock IBM during the year 2005. The relevant variables are called HIGH, LOW, and CLOSE (which also happen to be the names of the corresponding arguments on the HIGHLOW statement). Values of DATE should appear on the X axis. You'll need to use a WHERE statement to subset the data.




**Exercise 5. High-Low Plot**

### SOLUTION

```
proc sgplot data=sashelp.stocks;
  where stock='IBM' and year(date)=2005;
  highlow x=date high=high low=low / close=close;
run;
```

## EXERCISE 6: HORIZONTAL BOX PLOT

### THE HBOX STATEMENT

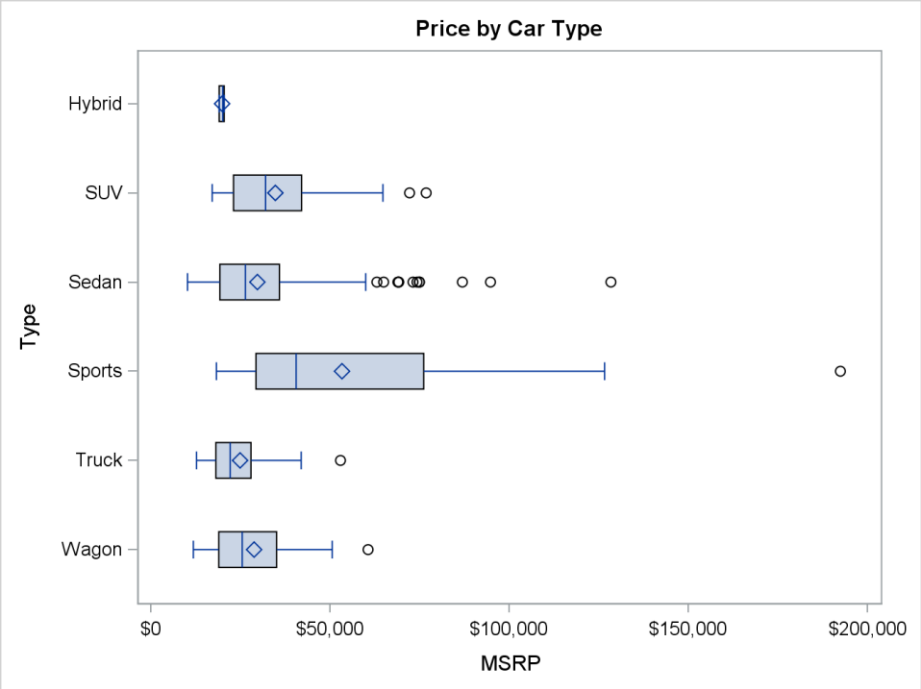
Box plots provide information about the distribution of a continuous variable. The HBOX statement is used to create a horizontal box plot, while VBOX generates a vertical box plot. The required argument is the name of a numeric analysis variable. Since there is only one required argument, it needs no name and is simply included directly after the HBOX or VBOX keyword.

```
proc sgplot data=<input-data-set> <options>;
  hbox variable < / options>;
run;
```

The CATEGORY= option can be used to create a separate box for each distinct value of a categorical variable. This can also be combined with the GROUP= option to add groups within each category.

### EXERCISE

Using the SASHELP.CARS data set, create a horizontal box plot of vehicle price (MSRP) by vehicle type (TYPE). You will need to use the HBOX statement along with a numeric analysis variable and the CATEGORY= option. Add a title.




**Exercise 6. Horizontal Box Plot**

### SOLUTION

```
proc sgplot data=sashelp.cars;
  title "Price by Car Type";
  hbox msrp / category=type;
run;
```

## EXERCISE 7: VERTICAL BAR CHART

### THE VBAR STATEMENT

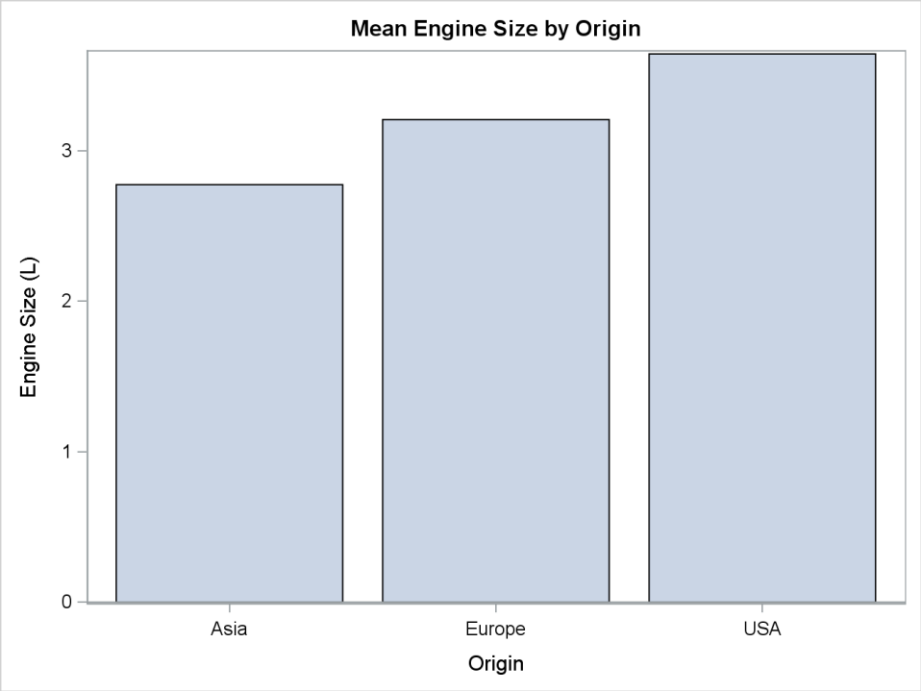
Bar charts are a common way of summarizing information about a categorical variable. The HBAR statement will produce a horizontal bar chart, while vertical bar charts are created using VBAR. In either case, the only required argument is a categorical variable, which may be either numeric or character. As with the HBOX statement in the previous exercise, this argument is unnamed. A separate bar will be drawn for each value of this categorical variable.

```
proc sgplot data=<input-data-set> <options>;
  vbar categorical-variable < / options>;
run;
```

The optional RESPONSE= argument can be used to specify a response variable that will control the lengths of the bars. This is frequently used in conjunction with the STAT= option which specifies the statistic that determines the bar lengths. The default statistic is the sum when a response variable is specified, or a frequency count otherwise.

### EXERCISE

Using SASHELP.CARS, create a vertical bar chart of mean engine size (ENGINESIZE) by vehicle origin (ORIGIN). Use the VBAR statement with a categorical variable and the RESPONSE= and STAT= options. Add a title. (Bonus: Try adding the option LIMITS=BOTH and see what happens.)




**Exercise 7. Vertical Bar Chart**

### SOLUTION

```
proc sgplot data=sashelp.cars;
  title "Mean Engine Size by Origin";
  vbar origin / response=enginesize stat=mean;
run;
```

## EXERCISE 8: GROUPED VERTICAL BAR CHART

### THE GROUP= OPTION FOR BAR CHARTS

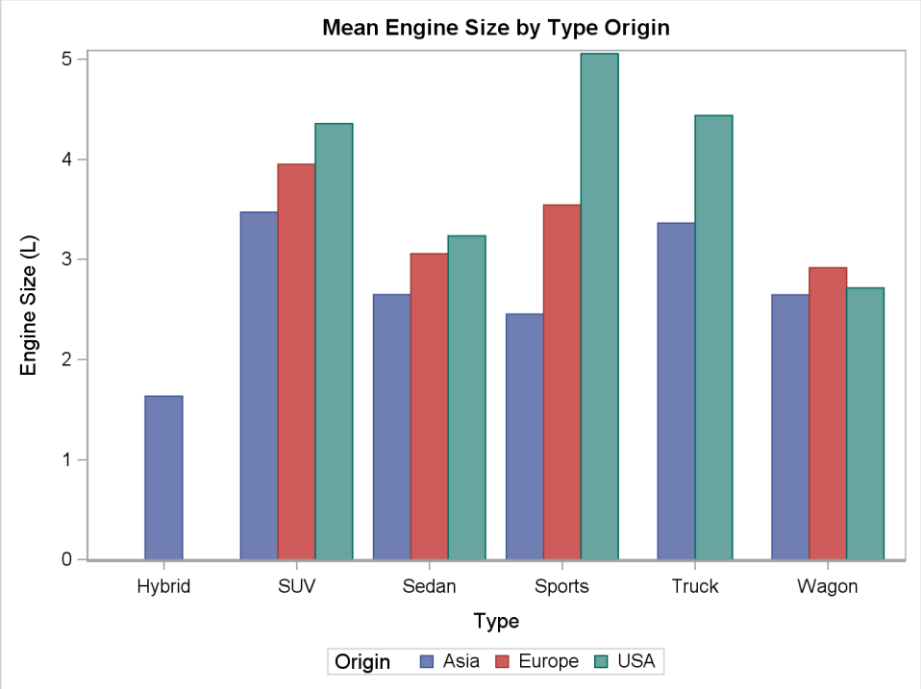
Like most plot types, VBAR and HBAR support the GROUP= option. This creates a separate bar for each distinct value of a grouping variable within each value of the main categorical variable.

```
proc sgplot data=<input-data-set> <options>;
  vbar categorical-variable
          / group=variable <more options>;
run;
```

The GROUPDISPLAY= option can be used to control how the bars are grouped. Specifying GROUPDISPLAY=CLUSTER will cluster the grouped bars together for each category. In contrast, GROUPDISPLAY=STACK will stack the bars for each group within each category.

### EXERCISE

Using SASHELP.CARS, create a vertical bar chart of mean engine size (ENGINESIZE) by vehicle type (TYPE) grouped into clusters by vehicle origin (ORIGIN). Add a title. (Bonus: Stack the bars instead.)




**Exercise 8. Grouped Vertical Bar Chart**

### SOLUTION

```
proc sgplot data=sashelp.cars;
  title "Mean Engine Size by Type and Origin";
  vbar type / response=enginesize stat=mean
              group=origin groupdisplay=cluster;
run;
```

## EXERCISE 9: HEAT MAP

### THE HEATMAP STATEMENT

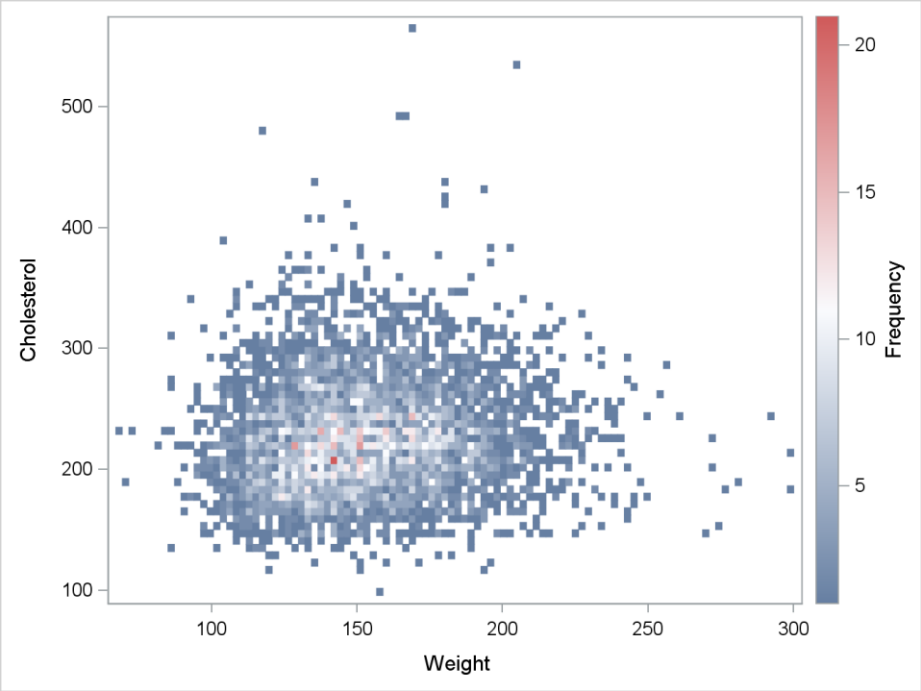
The HEATMAP creates a grid of color-coded rectangles based on a two-dimensional binning of the data values. The required arguments X= and Y= specify the variables to be binned and plotted.

```
proc sgplot data=<input-data-set> <options>;
  heatmap x=variable y=variable < / options>;
run;
```

The SGPLOT procedure has a built-in algorithm to decide the number and size of the bins in each dimension. There are also options available to explicitly control this process such as NXBINS=, NYBINS=, XBINSIZE=, and YBINSIZE=. Try out these options to see how they affect the output.

### EXERCISE

Using the SASHELP.HEART data set, create a heat map of CHOLESTEROL vs. WEIGHT. You'll need to use the HEATMAP statement along with the X= and Y= arguments.




**Exercise 9. Heat Map**

### SOLUTION

```
proc sgplot data=sashelp.heart;
  heatmap x=weight y=cholesterol;
run;
```

What happens if you add the options NXBINS=50 and NYBINS=50? Don't forget the slash!

## EXERCISE #10: GROUPED VERTICAL LINE CHART

### THE VLINE STATEMENT

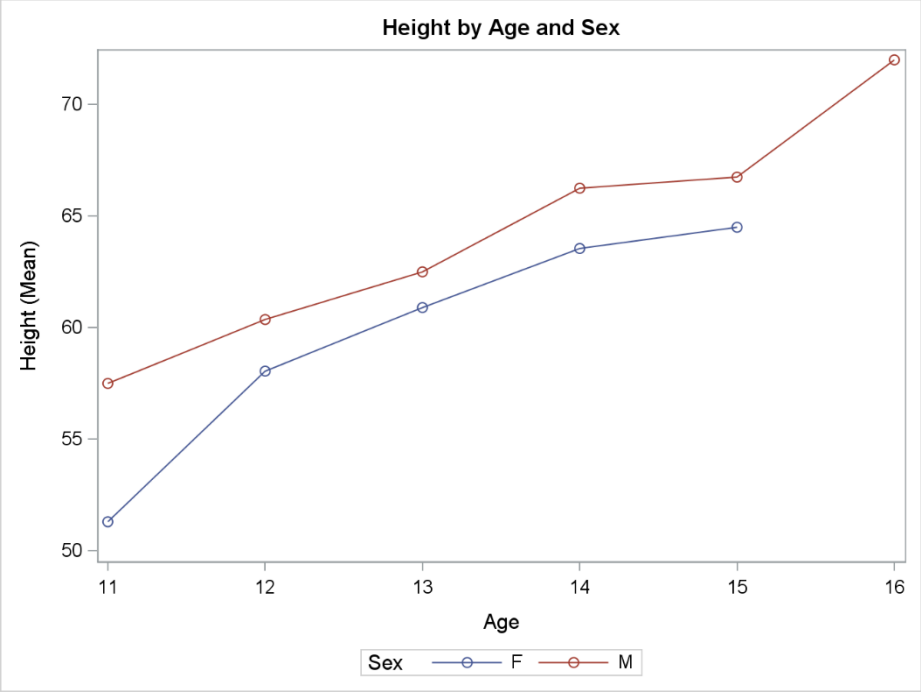
The VLINE statement is used to create a vertical line chart (which consists of horizontal lines). At first, this may sound like the SERIES statement. However, the values being plotted are statistics based on a categorical variable as opposed to raw data values. In that sense, VLINE is more like VBAR, and the syntax reflects that similarity.

```
proc sgplot data=<input-data-set> <options>;
  vline categorical-variable < / options>;
run;
```

The RESPONSE= and STAT= options have the same effects on the VLINE statement as they have on VBAR. Like the SERIES statement, the VLINE statement doesn't add plot markers to the line unless the MARKERS option is specified. To create a horizontal line chart, there is an analogous HLINE statement.

### EXERCISE

Using the SASHELP.CLASS data set, create a vertical line chart of mean HEIGHT by AGE grouped by SEX and include plot markers. Use the VLINE statement with a categorical variable and the RESPONSE=, STAT=, GROUP=, and MARKERS options. Add a title. (Bonus: Try adding the option LIMITS=BOTH.)




**Exercise 10. Grouped Vertical Line Chart**

### SOLUTION

```
proc sgplot data=sashelp.class;
  title "Height by Age and Sex";
  vline age / response=height stat=mean markers group=sex;
run;
```

## EXERCISE 11: REGRESSION PLOT

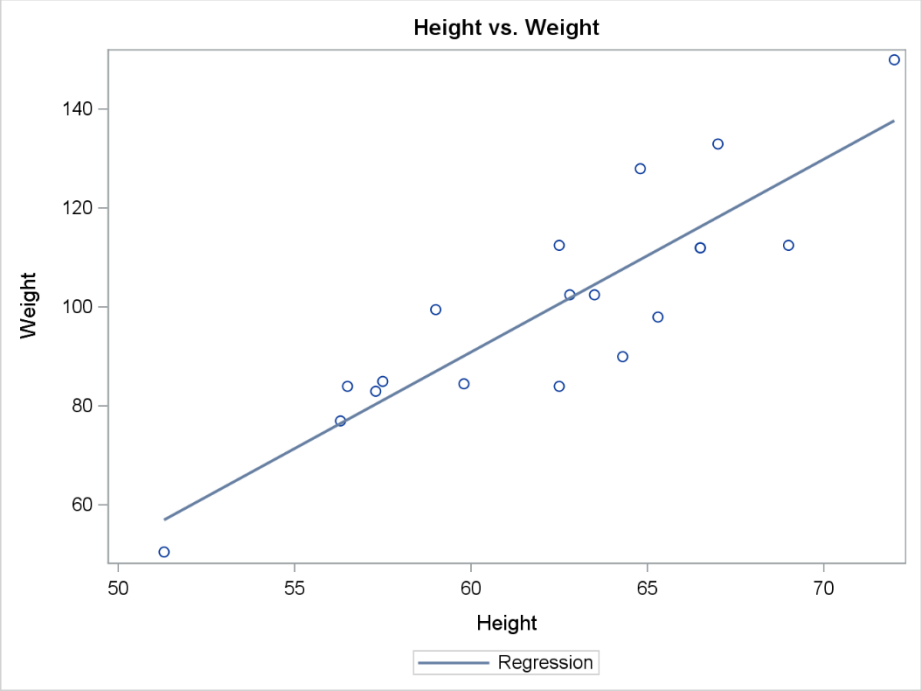
### THE REG STATEMENT

The REG statement adds a regression line to a scatter plot showing the relationship between two variables. The required arguments X= and Y= specify the variables to plot.

```
proc sgplot data=<input-data-set> <options>;
  reg x=variable y=variable < / options>;
run;
```

### EXERCISE

Working with the SASHELP.CLASS data set, create a regression plot of WEIGHT vs. HEIGHT. Use the REG statement with the X= and Y= arguments. Add a title.




**Exercise 11. Regression Plot**

### SOLUTION

```
proc sgplot data=sashelp.class;
  title "Height vs. Weight";
  reg x=height y=weight;
run;
```

What happens if you add the options CLM (confidence limits) and/or CLI (prediction limits)? Don't forget the slash!

## EXERCISE 12: HISTOGRAM AND DENSITY PLOT

### COMBINATION PLOTS

The SGPLOT procedure can be used to create combination plots by simply including multiple plot request statements. All plots will be overlaid atop one another in the same graph space and using the same axis system. Plots are drawn in the order listed within the procedure call, with subsequent plot requests drawn over earlier plots.

```
proc sgplot data=<input-data-set> <options>;
  <plot-request-statement-1>
  <plot-request-statement-2>
  <plot-request-statement-3>
  ...
  <other optional statements>
run;
```

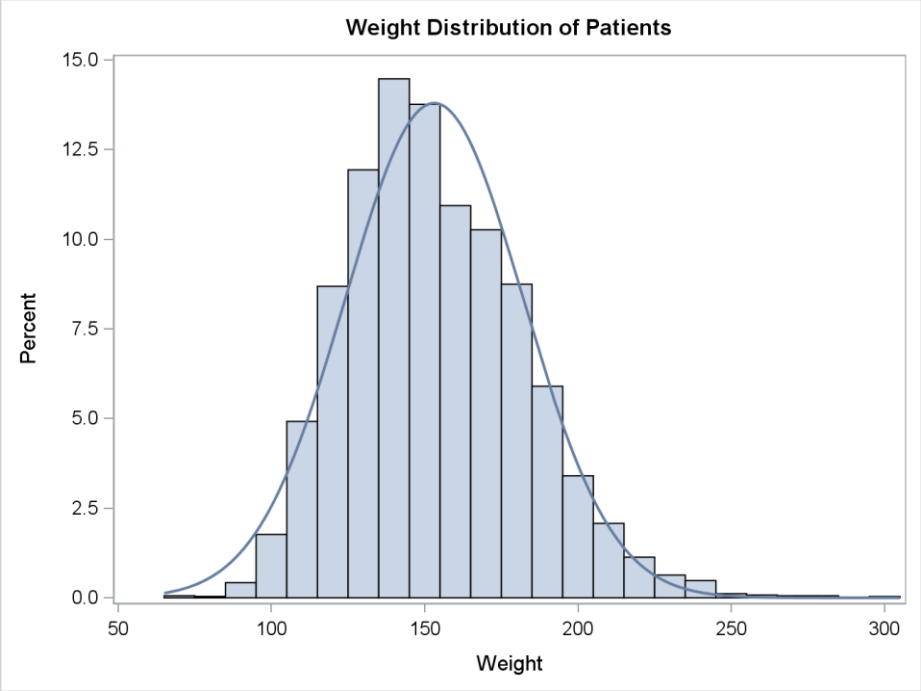
### THE HISTOGRAM AND DENSITY STATEMENTS

The HISTOGRAM statement creates a histogram while the DENSITY statement draws a density curve. These are two ways of visualizing the distribution of the values of a variable. Both require only a single response variable as the required argument, which is unnamed.

```
proc sgplot data=<input-data-set> <options>;
  histogram response-variable < / options>;
  density response-variable < / options>;
run;
```

### EXERCISE

Using the SASHELP.HEART data set, create a combination plot including both a histogram and density plot of WEIGHT. Use the HISTOGRAM statement with a response variable followed by the DENSITY statement with a response variable. In addition, suppress the automatic legend using the NOAUTOLEGEND option on the PROC SGPLOT statement. Add a title.




**Exercise 12. History and Density Plot**

**SOLUTION**

```
proc sgplot data=sashelp.heart noautolegend;
  title "Weight Distribution of Patients";
  histogram weight;
  density weight;
run;
```

The density curve is plotted over the histogram because it comes later in the SGPLOT procedure call. Try reversing the order and see what happens.

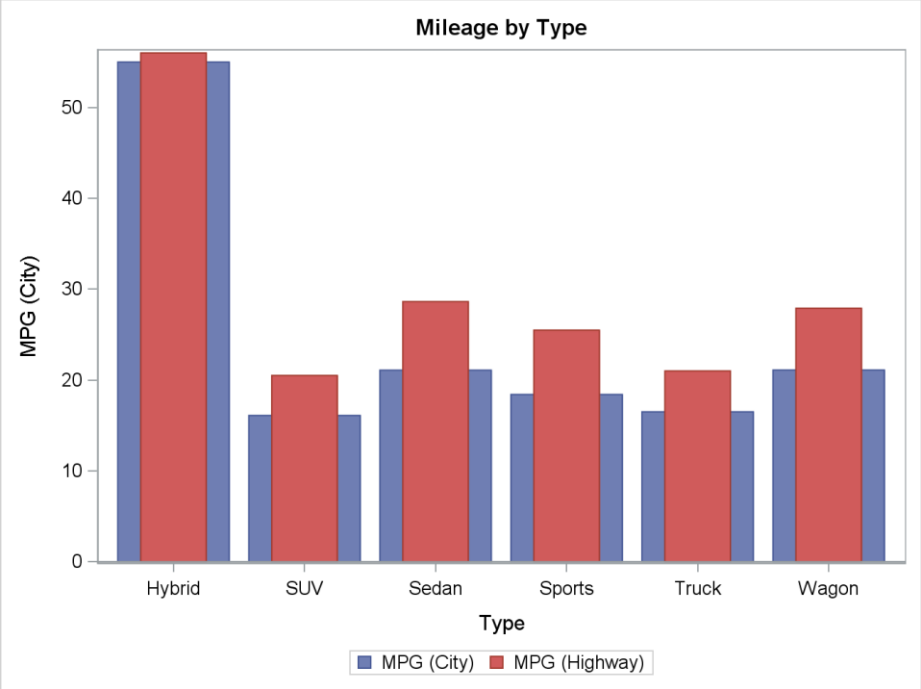
## EXERCISE 13: LAYERED VERTICAL BAR CHART

### BARWIDTH= OPTION FOR BAR CHARTS

One of the many options available on the VBAR and HBAR statements is the BARWIDTH= option. This controls the width of the bars and is expressed as a proportion of the maximum possible width. For example, BARWIDTH=1 would result in no space at all between the bars while BARWIDTH=0.5 would produce bars only half as wide. The default value is 0.8.

### EXERCISE

Working with the SASHELP.CARS data set, create a layered vertical bar chart showing both city mileage (MPG_CITY) and highway mileage (MPG_HIGHWAY) by vehicle type (TYPE). Use the VBAR statement twice to overlay one bar chart on the other. The second bar chart should use the BARWIDTH=0.5 option so the bars from the first bar chart will not be completely obscured. Each VBAR statement will also need a categorical response variable as well as the RESPONSE= and STAT= options. Add a title.




**Exercise 13. Layered Vertical Bar Chart**

### SOLUTION

```
proc sgplot data=sashelp.cars;
  title 'Mileage by Type';
  vbar type / response=mpg_city stat=mean;
  vbar type / response=mpg_highway stat=mean barwidth=0.5;
run;
```

The second vertical bar chart is drawn on top of the first. What happens if you reverse the order of the VBAR statements? What if you leave off the BARWIDTH option?

## EXERCISE 14: BAR AND LINE CHART

### THE VBARBASIC STATEMENT

When combining plot requests to create combination plots, some plot types are incompatible with others. For example, the VBAR and SCATTER plots cannot be used together. Any attempt to do so will result in the following error:

```
ERROR: Attempting to overlay incompatible plot or chart types.
```

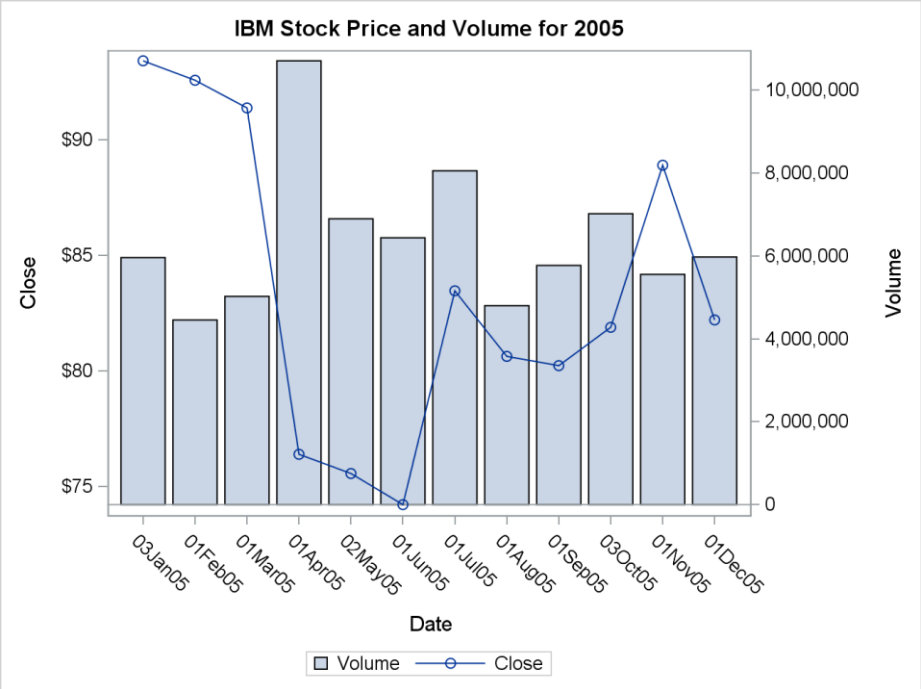
However, there is an alternate version of the VBAR statement called VBARBASIC that eliminates the incompatibility. The VBARBASIC statement can be used in combination with basic plot types such as SCATTER and SERIES. There are some minor differences in functionality between VBAR and VBARBASIC, but the syntax and most of the options are the same. There is also an analogous HBARBASIC statement which can be used in lieu of HBAR in combination with basic plot types.

### MULTIPLE AXIS SYSTEMS IN COMBINATION PLOTS

By default, all plot requests in a combination plot use the same axes. This can cause undesired results when the plots have different ranges of values. Using the secondary axis system can help in this situation. The secondary X axis is located along the top of the plot area, and the secondary Y axis is to the right-hand side. To specify that one or both secondary axes should be used for a plot, simply include the X2AXIS and/or Y2AXIS options on the corresponding plot request statement.

### EXERCISE

Using the SASHELP.STOCKS data set, combine a series plot of closing price (CLOSE) and a vertical bar chart of trading volume (VOLUME) for monthly IBM stock data for 2005. Use the VBARBASIC statement with a categorical variable and the RESPONSE= and Y2AXIS options, followed by a SERIES statement with the X= and Y= arguments and the MARKERS option. You'll also need a WHERE statement to subset the data. Add a title as well.




**Exercise 14. Bar and Line Chart**

**SOLUTION**

```
proc sgplot data=sashelp.stocks;
  title IBM Stock Price and Volume for 2005';
  where stock='IBM' and year(date)=2005;
  vbarbasic date / response=volume y2axis;
  series x=date y=close / markers;
run;
```

What happens if you leave off the Y2AXIS option? What if you reverse the order of the plot statements?

## WRAP-UP

The SGPLOT procedure provides an extremely flexible mechanism for creating publication-quality statistical graphs with minimal coding. There are several more plot types available beyond those covered in this paper, and many more options available on those that were covered. Consult the *SAS 9.4 ODS Graphics: Procedures Guide, Sixth Edition* (SAS Institute, 2016) for complete reference material on all the capabilities of SGPLOT.

These exercises have only scratched the surface of the customization available through the SGPLOT procedure. To delve deeper into the SGPLOT procedure, refer to the companion paper *Doing More with the SGPLOT Procedure* (Horstman 2018).

## REFERENCES

Horstman, Joshua M. *Doing More with the SGPLOT Procedure*. Proceedings of the SouthEast SAS Users Group 2018 Conference, 2018. Paper 205. https://www.lexjansen.com/sesug/2018/SESUG2018_Paper-205_Final_PDF.pdf

*SAS 9.4 ODS Graphics: Procedures Guide, Sixth Edition*. Cary, NC: SAS Institute, 2016. https://documentation.sas.com/api/docsets/grstatproc/9.4/content/grstatproc.pdf.

## CONTACT INFORMATION

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

Joshua M. Horstman
Nested Loop Consulting
317-721-1009
josh@nestedloopconsulting.com