CHAPTER 1

Scatter Plots

Purpose: This chapter demonstrates how to create basic scatter plots using Proc Gplot, and control the markers, axes, and text labels.

Basic Scatter Plot

Scatter plots are probably the simplest kind of graph, and provide a great way to visually look for relationships between two variables. Let's start with a very simple scatter plot, using the SASHELP.CLASS sample data that ships with SAS. The data set contains the sex, age, height, and weight for 19 students. Here are the first few lines of data:

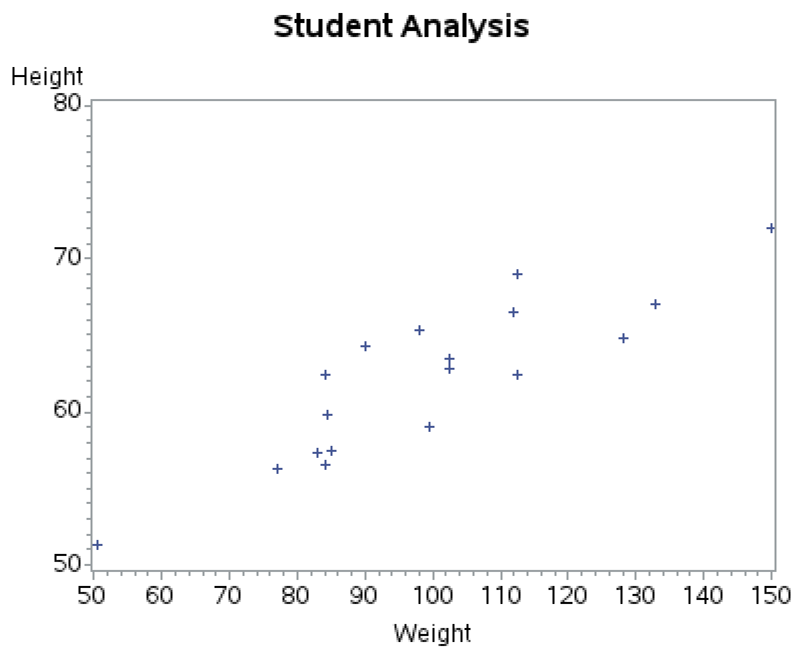
sashelp.class

Obs	Name	Sex	Age	Height	Weight
1	William	M	15	66.5	112.0
2	Thomas	M	11	57.5	85.0
3	Ronald	M	15	67.0	133.0
4	Robert	M	12	64.8	128.0
5	Philip	M	16	72.0	150.0

In this example, we will use a scatter plot to look for a relationship between the height and weight of the students.

```
title1 ls=1.5 "Student Analysis";
proc gplot data=sashelp.class;
plot height*weight;
run;
```

The code produces the following default plot, which shows that the taller students generally weigh more, and shorter students generally weigh less.

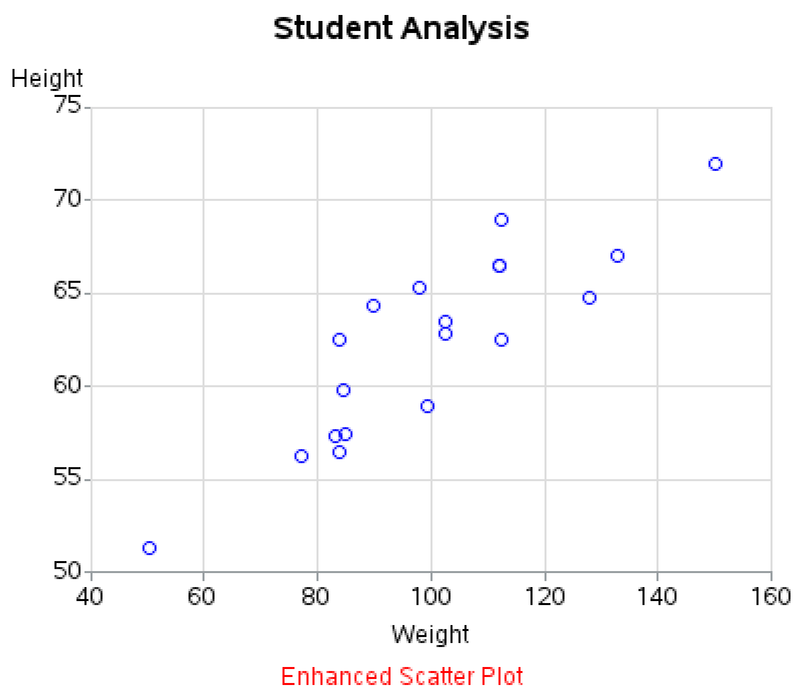


Default Scatter Plot

As with most graphs, the default settings are ok in a generic sort of way, but we can produce a much better graph by specifying a few options. Let's use a better plot marker, clean up the axes, and add some light gray reference lines. Use a SYMBOL statement to specify a blue circle as the plot marker. Use AXIS statements for the VAXIS and HAXIS to specify the numeric ranges, suppress the minor tick marks, and get rid of the offset gap at the ends of the ranges. Use the AUTOVREF and AUTOHREF options to add light gray reference lines at the major axis tick marks. Then use the NOFRAME option to get rid of the right and top edges around the graph area (the light gray reference lines will suffice).

```
title1 ls=1.5 "Student Analysis";
symbol1 value=circle height=3 interpol=none color=blue;
axis1 order=(50 to 75 by 5) minor=none offset=(0,0);
axis2 order=(40 to 160 by 20) minor=none offset=(0,0);
proc gplot data=sashelp.class;
plot height*weight /
 vaxis=axis1 haxis=axis2 noframe
 autovref cvref=graydd
 autohref chref=graydd;
run;
```

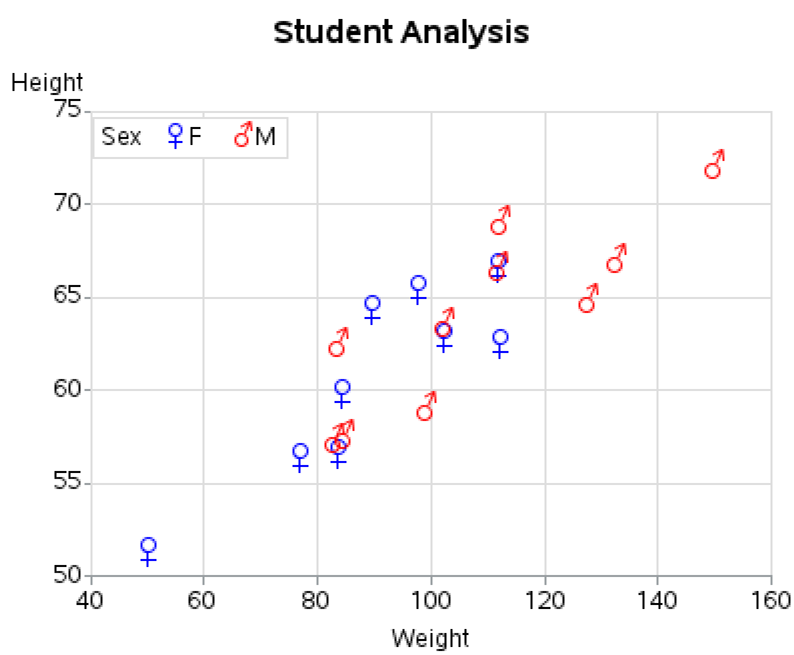
The resulting scatter plot is easy to read and visually pleasing.



In the previous graph, we controlled the shape of the marker (value=circle) – what if we want various different groups of data to be represented by different markers? First, make sure you have a variable in your data that contains a different unique value for each marker shape, and then instead of just plotting Y*X, you plot Y*X=V (where V is the name of that variable). In this case, we have a variable called SEX with values of M and F (male and female), therefore we can plot HEIGHT*WEIGHT=SEX.

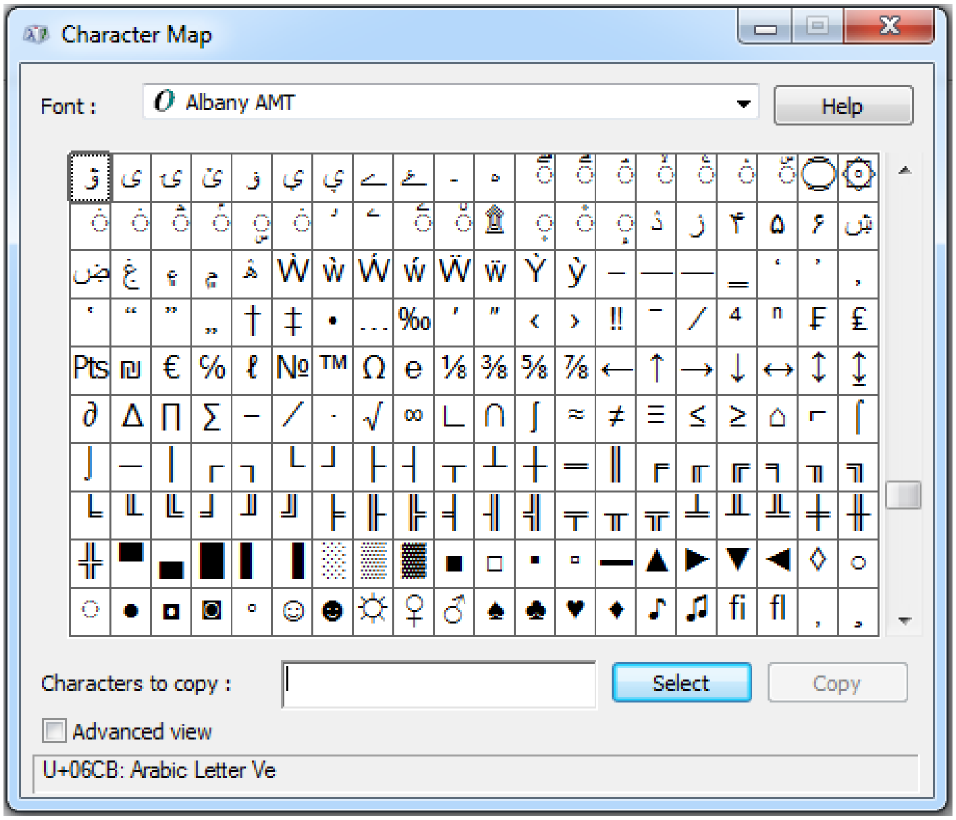
Note that this third variable does not contain the actual shapes to use, but rather it only needs to contain unique values for each group. These values are then assigned alphabetically to the marker shapes specified in the SYMBOL statements. SAS has many built-in shapes with mnemonic names (such as circle, dot, diamond, and square), and you can also use any character from any font by specifying the font name and the hexadecimal code for the character. In this case, since the values represent ‘male’ and ‘female,’ let’s use the male and female symbols. I think it is also useful to make the size of the symbols in the legend closely match the size of the symbols in the legend, therefore I use the SHAPE option of the LEGEND statement to control it.

```
title1 ls=1.5 "Student Analysis";
symbol1 font='albany amt/unicode' value='2640'x
 height=4.5 interpol=none color=blue;
symbol2 font='albany amt/unicode' value='2642'x
 height=4.5 interpol=none color=red;
legend1 position=(top left inside) shape=symbol(.,4)
 repeat=1 mode=protect cborder=graydd;
axis1 order=(50 to 75 by 5) minor=none offset=(0,0);
axis2 order=(40 to 160 by 20) minor=none offset=(0,0);
proc gplot data=sashelp.class;
plot height*weight=sex / legend=legend1
 vaxis=axis1 haxis=axis2 noframe
 autovref cvref=graydd
 autohref chref=graydd;
run;
```



Data-Driven Marker Shapes

Let's Talk: You probably like the idea of using font characters for the plot markers, but you're wondering how to find the hexadecimal code for the characters. The technique I would recommend is to select the desired font in the Windows "Character Map," and after you find the character you want, you can click on it and see the hexadecimal code at the bottom of the window.

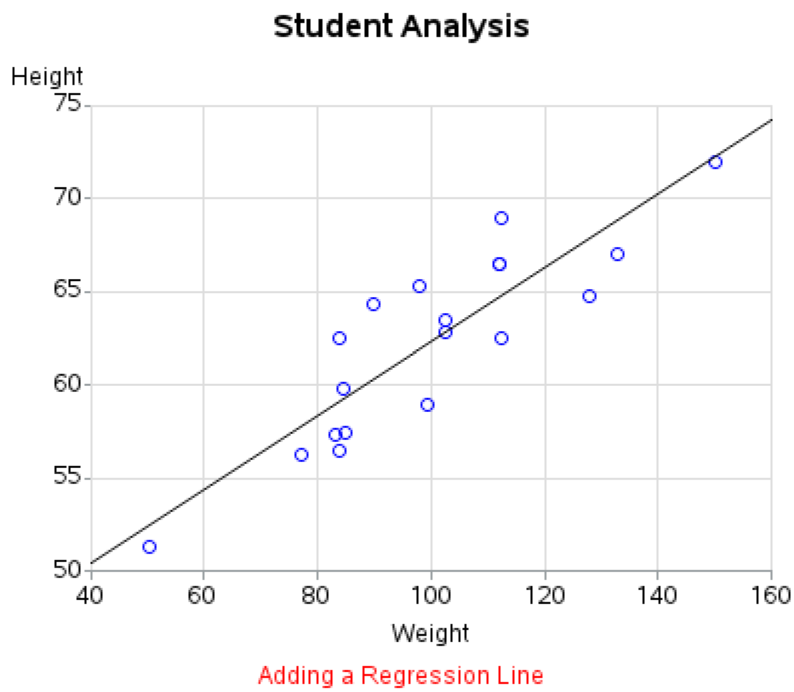


Regression Line

Scatter plots are often used to look for relationships between two variables, and a powerful analytic tool that can augment such plots is the regression line. SAS has specialized statistical procedures to help with in-depth regression analyses, but if you just want to add a simple regression line then you can use the capabilities that are built into Proc Gplot.

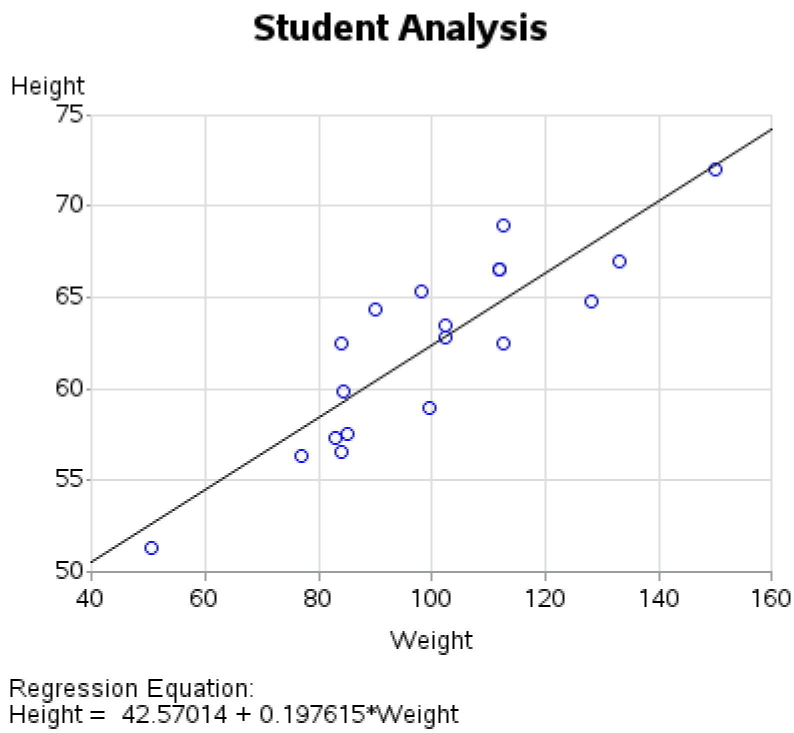
In the previous scatter plots, we used INTERPOL=NONE so there was no line or curve connecting the markers. If you specify INTERPOL=RL a regression line will be drawn through the markers. You can specify the color of the markers separately from the color of the line, using the CV (color of markers) and the CI (color of interpolation line) options on the SYMBOL statement.

```
title1 ls=1.5 "Student Analysis";
goptions reset=symbol;
symbol1 value=circle height=3 cv=blue interpol=rl ci=black;
axis1 order=(50 to 75 by 5) minor=none offset=(0,0);
axis2 order=(40 to 160 by 20) minor=none offset=(0,0);
proc gplot data=sashelp.class;
plot height*weight=1 /
 vaxis=axis1 haxis=axis2 noframe
 autovref cvref=graydd
 autohref chref=graydd;
run;
```



As you can see in the plot above, the markers do generally follow the regression line (taller students are generally heavier students), but it's difficult to tell just by looking at the line exactly how the height and weight are related. If you add the REGEQN option, then the equation used to draw the regression line is used, so you can easily see what the mathematical relationship is.

```
title1 ls=1.5 "Student Analysis";
symbol1 value=circle height=3 cv=blue interpol=rl ci=black;
axis1 order=(50 to 75 by 5) minor=none offset=(0,0);
axis2 order=(40 to 160 by 20) minor=none offset=(0,0);
proc gplot data=sashelp.class;
plot height*weight=1 /
 vaxis=axis1 haxis=axis2 noframe regeqn
 autovref cvref=graydd
 autohref chref=graydd;
run;
```

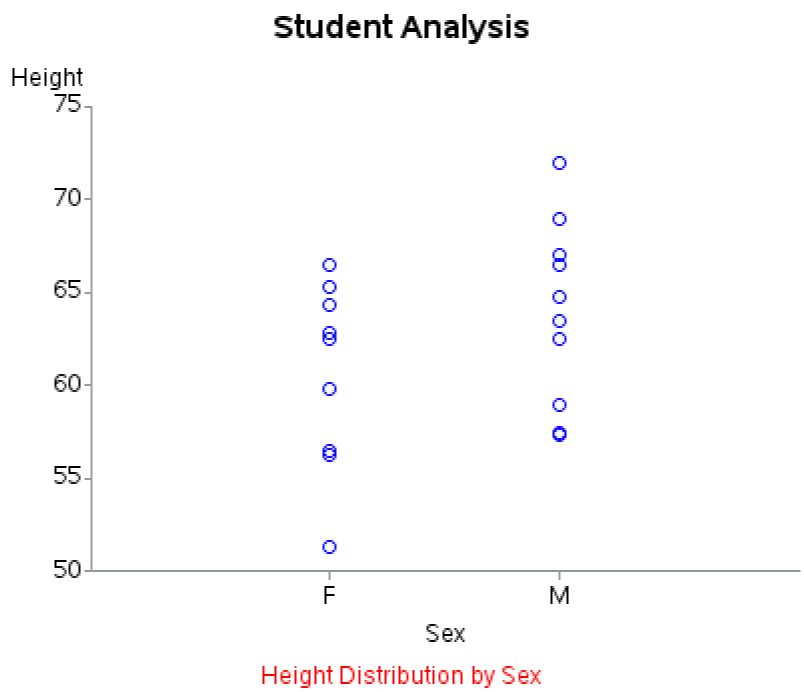


Adding Regression Equation

Box Plot

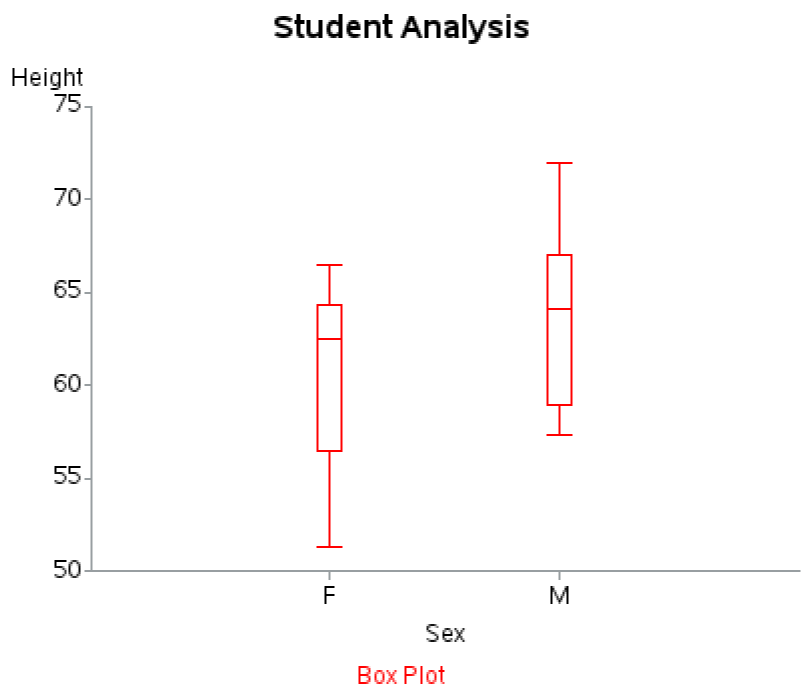
Another variation that can help increase the analytic power of a scatter plot is a box plot (in the special case where the variable plotted on the horizontal axis represents discrete value, not continuous). For example, let's say you want to analyze the height distribution of the students by sex. You might start with a simple scatter plot like the following (note that I add some OFFSET to the left and right side of the HAXIS so that the plot markers will be shifted more towards the middle of the plot).

```
title1 ls=1.5 "Student Analysis";
symbol1 value=circle height=4 interpol=none color=blue;
axis1 order=(50 to 75 by 5) minor=none offset=(0,0);
axis2 offset=(30,30);
proc gplot data=sashelp.class;
plot height*sex=1 /
 vaxis=axis1 haxis=axis2 noframe;
run;
```



In general, this scatter plot shows that the males are taller than the females (if you can assume that there are not too many markers overlaid on the exact same spot, which could bias the visual interpretation of the plot). But it sure would be nice to add some more quantitative summary information to the plot. For example, it would be great to know the median height for the males and females. A box plot is great for this, as it shows the median, as well as the 25th and 75th percentiles. You can easily generate a box plot using INTERPOL=BOX on the symbol statement (BOXT adds the optional top and bottom whiskers).

```
title1 ls=1.5 "Student Analysis";
symbol1 interpol=boxt bwidth=4 color=red;
axis1 order=(50 to 75 by 5) minor=none offset=(0,0);
axis2 offset=(30,30);
proc gplot data=sashelp.class;
plot height*sex=1 /
 vaxis=axis1 haxis=axis2 noframe;
run;
```

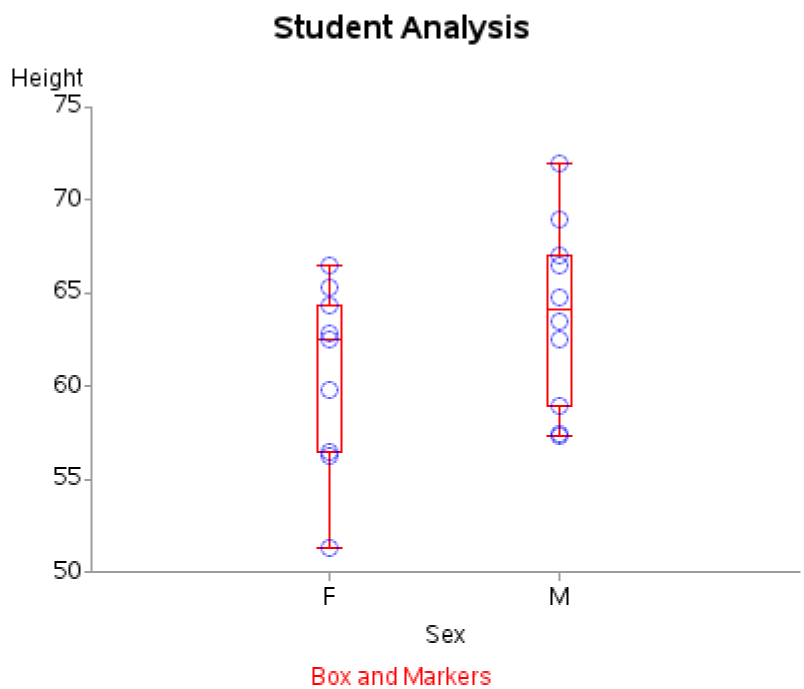


I often find it useful to overlay the individual markers on the box plot (using the OVERLAY option). This provides a little more insight into the distribution of the data within the percentile ranges, and so on. This is also a good occasion to utilize transparent colors (new in SAS 9.3) for the plot markers – it keeps the markers from obscuring the box plot, and when multiple markers are stacked in the same location the transparent colors combine and produce a darker marker. You can specify transparent colors using SAS RGBA color codes in the form aRRGGBBxx, where xx is the intensity (opacity) of the color (if you do not have SAS 9.3 yet, just use color=RED).

> **Let's Talk**: I recommend that you always specify which SYMBOL to use in your PLOT statement – for example PLOT Y*X=2 means use SYMBOL2. If you do not specify which to use, then SAS has an algorithm it follows to assign them. Also, I recommend you always specify a COLOR on your SYMBOL

statements. If you do not specify a color, then SAS will typically repeat that symbol using each of the colors in its color list.

```
title1 ls=1.5 "Student Analysis";
symbol1 interpol=boxt bwidth=4 color=red;
symbol2 value=circle height=4 interpol=none color=a0000ff77;
axis1 order=(50 to 75 by 5) minor=none offset=(0,0);
axis2 offset=(30,30);
proc gplot data=sashelp.class;
plot height*sex=1 height*sex=2 / overlay
 vaxis=axis1 haxis=axis2 noframe;
run;
```



Proportional Axes Plot

When the values being plotted on both axes are in the same units, it is often desirable to plot them to the same scale. By default, each axis is auto-scaled, and the lengths of the axes are determined by the size and proportions of the available area. This example demonstrates techniques you can use to override those defaults.

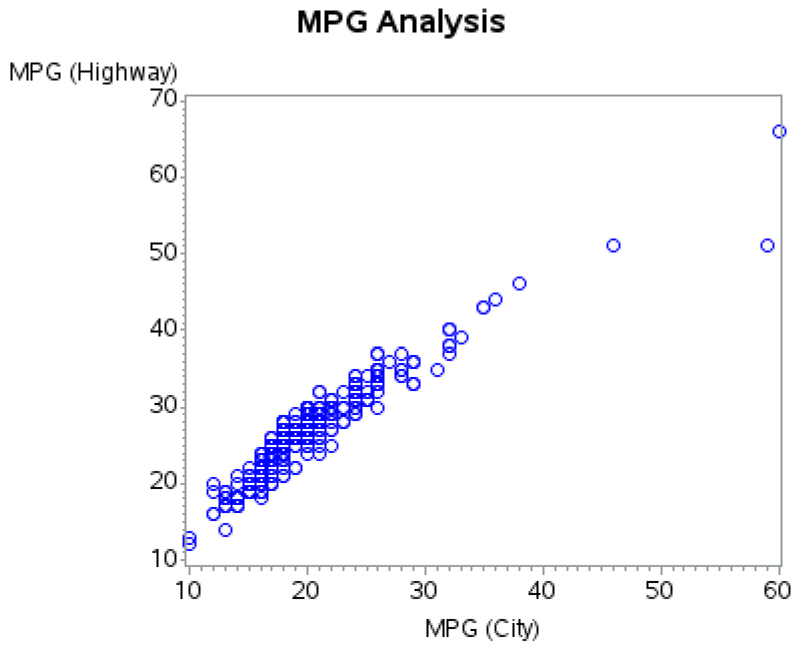
We'll be using the SASHELP.CARS data for this example. It contains miles per gallon (mpg) data for several cars produced in the year 2004. Below are a few of the observations from the data. (If some of the mpg values look a little high, it's because these are the original numbers that were obtained using the pre-2008 test standards, which did not measure the hybrid vehicles correctly, for example.)

sashelp.cars

Make	Model	MPG_City	MPG_Highway
Toyota	Prius 4dr (gas/electric)	59	51
Toyota	Sequoia SR5	14	17
Toyota	4Runner SR5 V6	18	21
Toyota	Highlander V6	18	24
Toyota	Land Cruiser	13	17

With minimal code, you can easily produce a plot with the default axes. Notice that both axes are plotting mpg, but the axes are auto-scaled differently, and the horizontal axis is longer than the vertical axis.

```
title1 ls=1.5 "MPG Analysis";
symbol1 value=circle height=4 interpol=none color=blue;
proc gplot data=sashelp.cars;
plot mpg_highway*mpg_city=1;
run;
```

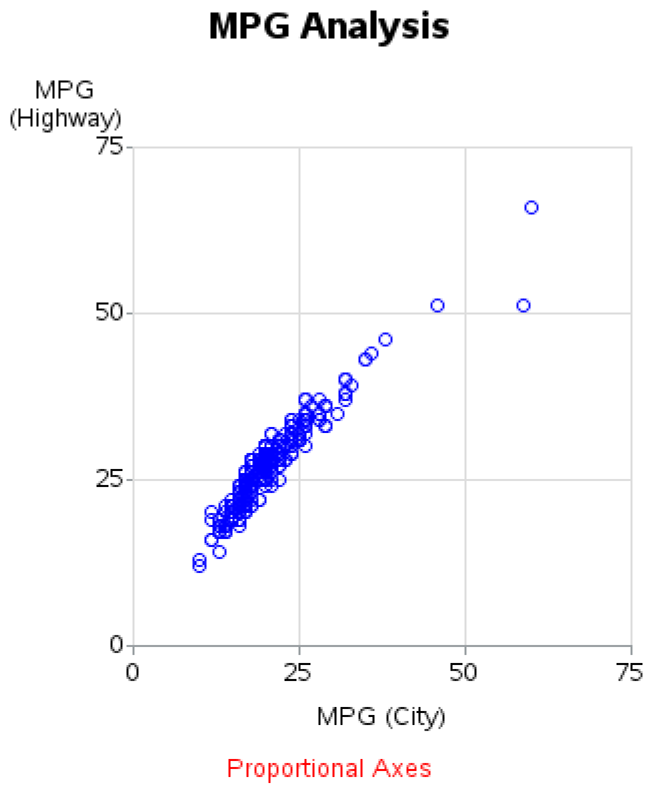


Default Axes

By specifying axis statements, we can force both axes to be the exact same physical length (2.6 inches), and cover the exact same range of values (0 to 75). We'll also turn on the automatic reference lines at the major tick marks to create a grid – the grid forms squares which help the users recognize that the axes are proportional. Also, since the default vertical axis label is a bit long, we can split it into two pieces by specifying those two pieces in the LABEL of the AXIS1, with a JUSTIFY (j=c) between them.

```
title1 ls=1.5 "MPG Analysis";
symbol1 value=circle height=3 interpol=none color=blue;
axis1 length=2.6in order=(0 to 75 by 25) minor=none
 offset=(0,0) label=(j=c 'MPG' j=c '(Highway)');
axis2 length=2.6in order=(0 to 75 by 25) minor=none
 offset=(0,0);
proc gplot data=sashelp.cars;
plot mpg_highway*mpg_city=1 /
 vaxis=axis1 haxis=axis2 noframe
 autovref cvref=graydd
 autohref chref=graydd;
run;
```

The resulting graph is now proportional, so that 1 mpg in the vertical direction is equal to 1 mpg in the horizontal direction.

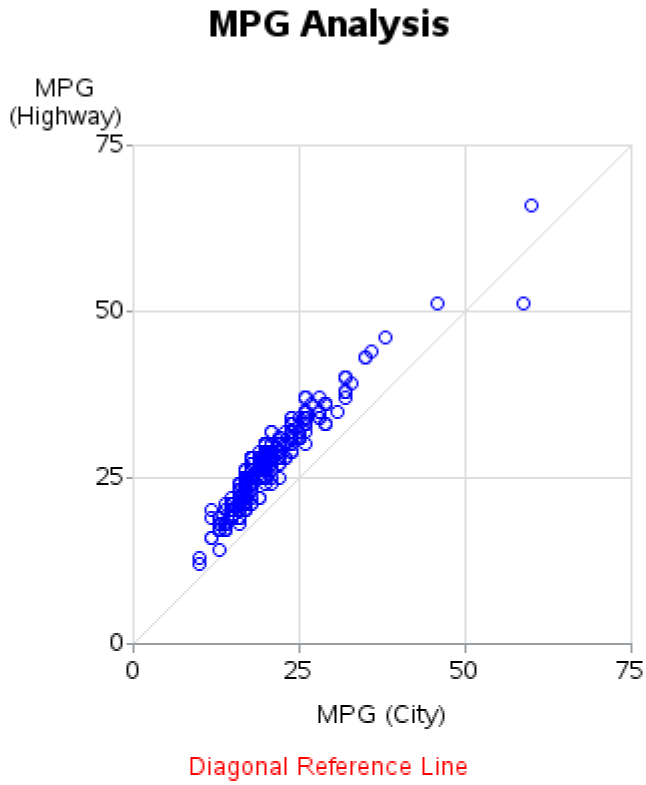


With mpg data, is the highway or city mpg usually higher? From personal experience, we probably all know that the highway mpg is usually higher (for non-hybrid vehicles, at least). If we draw a diagonal reference line through the plot, we can more easily see that characteristic of the data. There is no built-in option to draw a diagonal reference line, therefore we must annotate it. First you create a data set containing the annotate commands to MOVE to the bottom left of the graph, and then DRAW a gray line to the top right. Then specify that data set as the ANNO= in the Proc Gplot. Make the line the same color as the other reference lines, so the user will know it's a reference line (as opposed to a regression line).

```
data anno_diagonal;
xsys='2'; ysys='2';
x=0; y=0; function='move'; output;
x=75; y=75; function='draw'; color='graydd'; output;
run;

title1 ls=1.5 "MPG Analysis";
symbol1 value=circle height=3 interpol=none color=blue;
axis1 length=2.6in order=(0 to 75 by 25) minor=none
 offset=(0,0) label=(j=c 'MPG' j=c '(Highway)');
axis2 length=2.6in order=(0 to 75 by 25) minor=none
 offset=(0,0);
proc gplot data=sashelp.cars anno=anno_diagonal;
plot mpg_highway*mpg_city=1 /
 vaxis=axis1 haxis=axis2 noframe
 autovref cvref=graydd
 autohref chref=graydd;
run;
```

We now have a great plot, where we can symmetrically compare the highway and city mpg around a diagonal reference line. It is evident that most vehicles get noticeably better mileage on the highway.

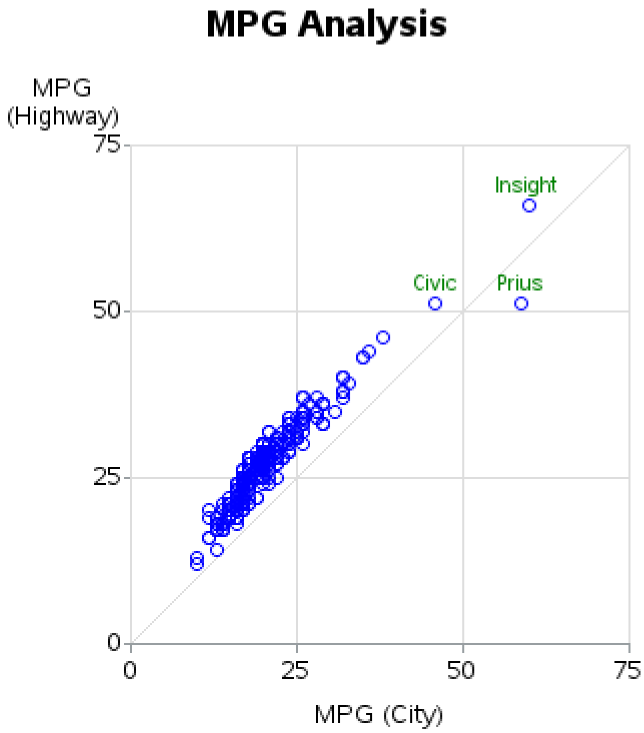


But, what about those outliers, getting extremely good mileage – users can't tell by looking at the plot what those are. Let's selectively add some labels to those markers. You can do this using the POINTLABEL functionality of the SYMBOL statement. If you want to label all the points, then you can tell it to use one of the variables in the data set as the label. But, in most cases I prefer to only label a subset of outliers, therefore I add a variable to the data (POINT_TEXT), containing text for the markers I want to label, and blank for all the others. In this case, I only show labels on markers over 50 mpg, and since the values for MODEL are a bit long for labels, I just use just the first word of that string.

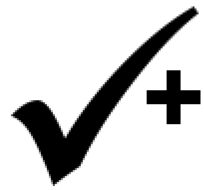
```
data my_data; set sashelp.cars;
length point_text $50;
if mpg_highway>50 or mpg_city>50 then
 point_text=scan(model,1,' ');
run;

title1 ls=1.5 "MPG Analysis";
symbol1 value=circle height=3 interpol=none color=blue
 pointlabel=(height=8pt color=green "#point_text");
axis1 length=2.6in order=(0 to 75 by 25) minor=none
 offset=(0,0) label=(j=c 'MPG' j=c '(Highway)');
axis2 length=2.6in order=(0 to 75 by 25) minor=none
offset=(0,0);
proc gplot data=my_data anno=anno_diagonal;
plot mpg_highway*mpg_city=1 /
 vaxis=axis1 haxis=axis2 noframe
 autovref cvref=graydd
 autohref chref=graydd;
run;
```

Now, I can easily tell at a glance that the Insight, Prius, and Civic hybrids are the vehicles that get extremely good mileage.



Point Labels



Bubble Plot

This variation of the scatter plot uses bubbles of varying size to represent a third variable. Whereas marker shapes are a good choice to represent character or categorical variables, the bubble size is better suited to represent the continuous values of a numeric variable (such as population, dollar amounts, and so on).

Let's use a different data set for this example – and in doing so, you will learn a few useful tricks for subsetting and customizing data. SASHELP.DEMOGRAPHICS is one of the sample data sets that ships with SAS. It contains various statistics about all the countries of the world. We will create a bubble plot where the size of the bubble represents the population, and the position of the bubble represents the literacy rate and the income per person.

A plot with a bubble for every country would be a bit crowded, therefore we will subset the data to only show the countries with over 75 million people. While we are subsetting the data, we can also do a bit of data cleaning, to get rid of rows that contain missing values for any of the variables (missing numeric values are represented by a '.' in SAS code). This is also a good time to apply the desired formats, and convert the upper case country names to mixed case.

```
data my_data; set sashelp.demographics
 (where=(pop>75000000 and AdultLiteracyPct^=. and gni^=.));
format gni dollar8.0;
format AdultLiteracyPct percent7.0;
name=propcase(name);
run;
```

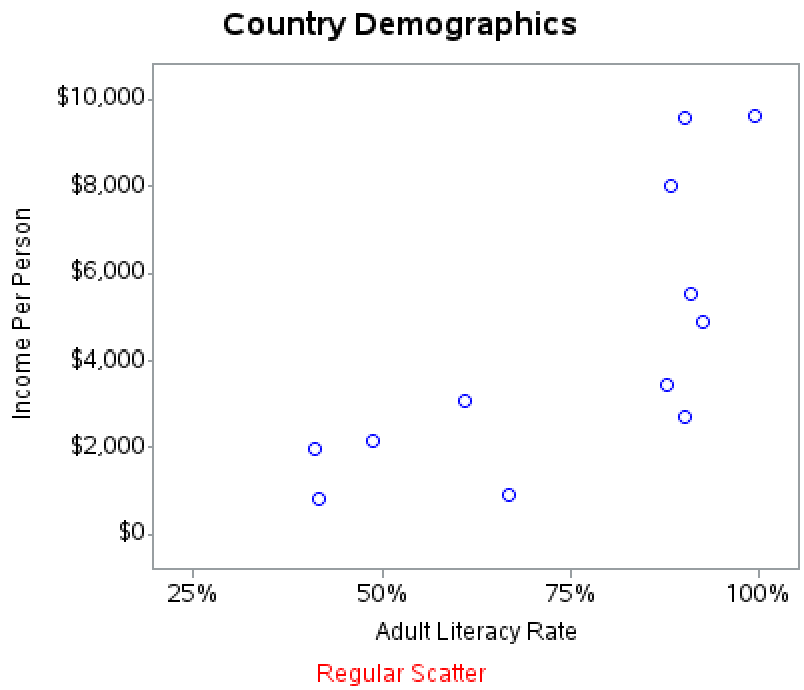
Here are the first few rows of the resulting data:

sashelp.demographics

NAME	GNI	AdultLiteracypct	pop
Mexico	$9,590	90%	107,029,360
Brazil	$8,020	88%	186,404,913
Ethiopia	$810	42%	77,430,702
Nigeria	$930	67%	131,529,669

Let's create a simple scatter plot first, to get all the layout logistics worked out. There are a couple of differences from the previous scatter plots. This time we'll add ANGLE=90 to the vertical axis label, so the long text will fit better. Also add OFFSET=(5,5) to the ends of each axis, so there will be some space between the last tick mark and the axis edge – this allows space for the bubbles that might extend outside the axis min/max values.

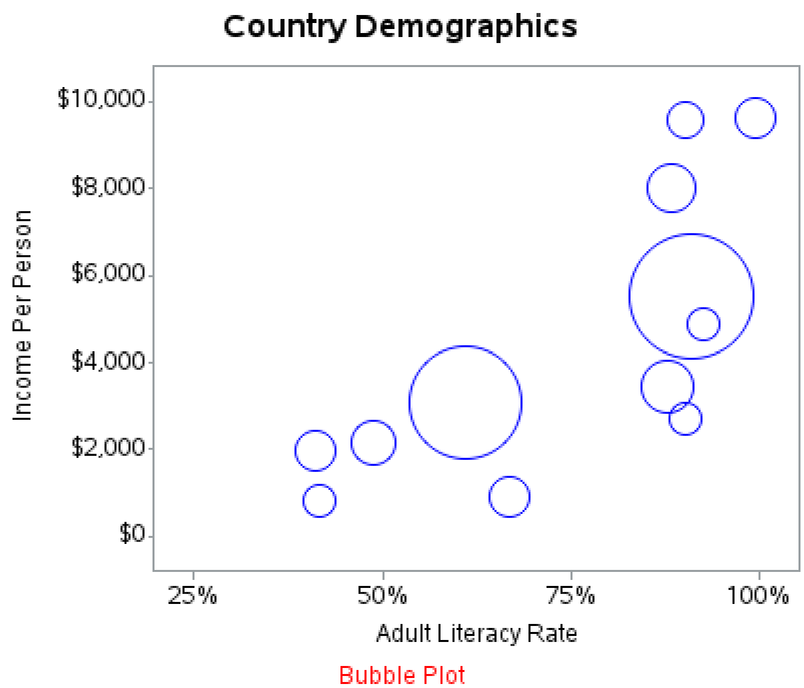
```
title1 ls=1.5 "Country Demographics";
goptions reset=symbol;
symbol1 value=circle height=4 interpol=none color=blue;
axis1 label=(angle=90 "Income Per Person")
 order=(0 to 10000 by 2000) minor=none offset=(5,5);
axis2 label=("Adult Literacy Rate")
 order=(.25 to 1 by .25) minor=none offset=(5,5);
proc gplot data=my_data;
plot gni*AdultLiteracyPct=1 /
 vaxis=axis1 haxis=axis2;
run;
```



The coding difference in a bubble plot and a regular scatter plot is that the bubble plot does not use a SYMBOL statement – the size and color of the bubble markers are controlled via options on the BUBBLE statement.

```
title1 ls=1.5 "Country Demographics";
axis1 label=(angle=90 "Income Per Person")
 order=(0 to 10000 by 2000) minor=none offset=(5,5);
axis2 label=("Adult Literacy Rate")
 order=(.25 to 1 by .25) minor=none offset=(5,5);
proc gplot data=my_data;
bubble gni*AdultLiteracyPct=pop /
 vaxis=axis1 haxis=axis2
 bcolor=blue bscale=area bsize=15;
run;
```

This produces a nice simple bubble plot, where the size (area) of the bubble represents the population of the country.



The Proc Gplot BUBBLE statement produces a nice simple bubble plot, with minimal code required, but it lacks two features that I often find desirable – labeling and coloring the individual bubbles in a data driven way. For example, adding labels with the country name, and coloring the bubbles by the region.

But there is a way to create such a bubble plot in SAS/GRAPH – you can use annotate to draw the bubbles, and color and label them any way you want! This book is geared towards the beginner, so I am not going to explain in depth how annotate works. But you should be able to easily re-use the code below with your own data.

First, you will need to determine the maximum population value in your data (which we will do using Proc SQL), and also declare what maximum bubble size you want to correspond to that maximum population. Also, we need to sort the data, so that the larger bubbles will be printed behind the smaller ones, so as not to obscure them.

```
proc sql noprint;
select max(pop) into :maxpop from my_data;
quit; run;

%let minbub=0;
%let maxbub=10;

proc sort data=my_data out=my_data;
by descending pop;
run;
```

Now we can create a data set with the annotate commands to draw the bubbles and labels. This might look like a lot of code, but notice that most of the code is just mapping the country names to the desired colors. The code assigns the X and Y location for the bubbles, then calculates a size (radius) so that the areas of the bubbles are proportional to the population, and then assigns a color. We create a solid colored bubble first, and then draw an outline around the colored bubble with a gray empty bubble. And last, we label the bubble with the country name.

```
data anno_bubbles; set my_data;
length function color $8;
xsys='2'; ysys='2'; hsys='3'; when='b';
x=AdultLiteracyPct; y=gni;
function='pie'; rotate=360;
size=&minbub+((sqrt(pop/3.14)/sqrt(&maxpop/3.14))*&maxbub);
if name in ('China' 'Philippines' 'Indonesia' 'Vietnam')
 then color='cxff2f2f';
if name in ('India' 'Pakistan' 'Bangladesh')
 then color='cx2fbfe5';
if name in ('Ethiopia' 'Nigeria') then color='cxD15FEE';
if name in ('Mexico' 'Brazil') then color='cxe5ff2f';
if name in ('Russia') then color='orange';
style='psolid'; output;
style='pempty'; color='grayaa'; output;
function='label'; position='2'; size=3.0; text=name;
 style=''; color=''; rotate=.; output;
run;
```

Since we’re annotating the colored bubbles, we do not really need for Gplot to draw any markers. Therefore we use VALUE=NONE on the symbol statement to suppress the regular markers, and then use ANNO= ANNO_BUBBLES to overlay our annotation on the plot.

```
title1 ls=1.5 "Country Demographics";
symbol1 value=none interpol=none;
axis1 label=(angle=90 "Income Per Person")
 order=(0 to 10000 by 2000) minor=none offset=(5,5);
axis2 label=("Adult Literacy Rate")
 order=(.25 to 1 by .25) minor=none offset=(5,5);
proc gplot data=my_data anno=anno_bubbles;
plot gni*AdultLiteracyPct=1 /
 vaxis=axis1 haxis=axis2;
run;
```

And there you have it – a nice bubble plot, with color-coded bubbles, and labels!