

SESUG Paper 114-2019

Scatterplots: Basics, Enhancements, Problems, and Solutions

Peter L. Flom, Peter Flom Consulting, New York, NY

ABSTRACT

The scatter plot is a basic tool for presenting information on two continuous variables. While the basic plot is good in many situations, enhancements can increase its utility. I also go over tools to deal with the problem of overplotting.

INTRODUCTION

In this paper, I discuss scatter plots. I start with a very basic example, and then illustrate some enhancements. Next, I show some problems that can occur, and illustrate some solutions.

With the new SG procedures, introduced in SAS®9.2, SAS allows us to make good scatter plots relatively easily. However, there are many options, and applying them well is not always obvious. And, for specialized purposes, PROC SGRENDER can produce highly customized graphics, but its use is not entirely straightforward.

BASIC SCATTER PLOTS AND ENHANCEMENTS

SIMPLE SCATTER PLOTS WITH PROC SGPLOT

The PROC for basic scatter plots is PROC SGPLOT. Rather than list a lot of the options and syntax for this PROC (all of which can be looked up) I will give some specific examples.

As a starting example, let's plot unemployment rate and infant mortality for each of the 50 states plus the District of Columbia. This code:

```
proc sgplot data = UnempIM; *STARTS THE PROC;  
  scatter x = Unemployment y = InfantMortality; *CREATES A PLOT, NOTE THE USE OF X = AND Y =;  
run;
```

Produces this scatter plot:

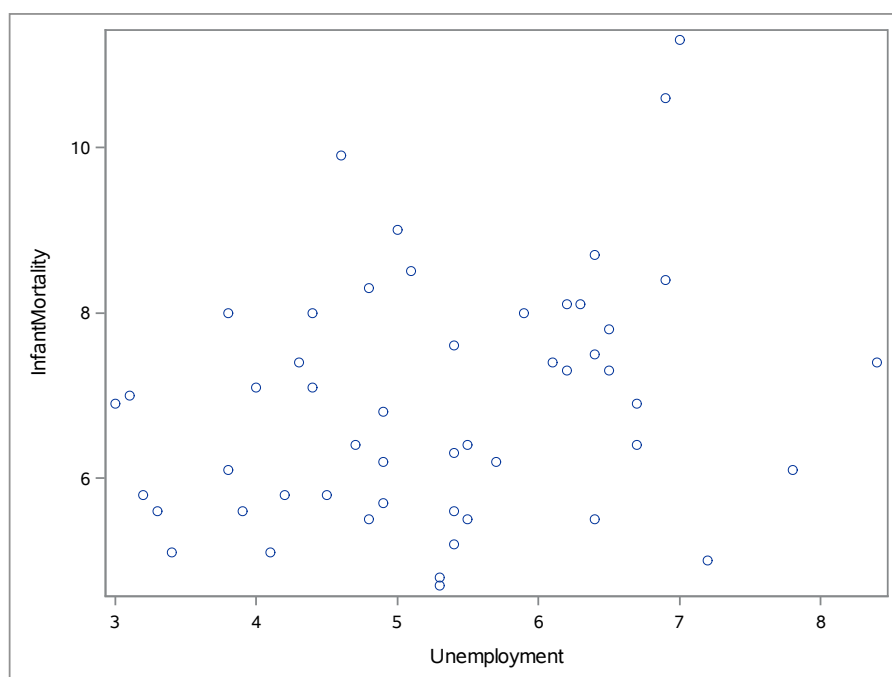


Figure 1: Most basic scatter plot

ENHANCING THE SCATTER PLOT WITH PROC SGPLOT

Next, we should probably make the labels on the axes clearer:

```
proc sgplot data = UnempIM;
  xaxis label = "Unemployment (%)";
  *THIS SHOULD BE SELF EXPLANATORY, THERE ARE OTHER AXIS OPTIONS AS WELL;
  yaxis label = "Infant Mortality (%)";
  scatter x = Unemployment y = InfantMortality;
run;
```

This creates figure 2.

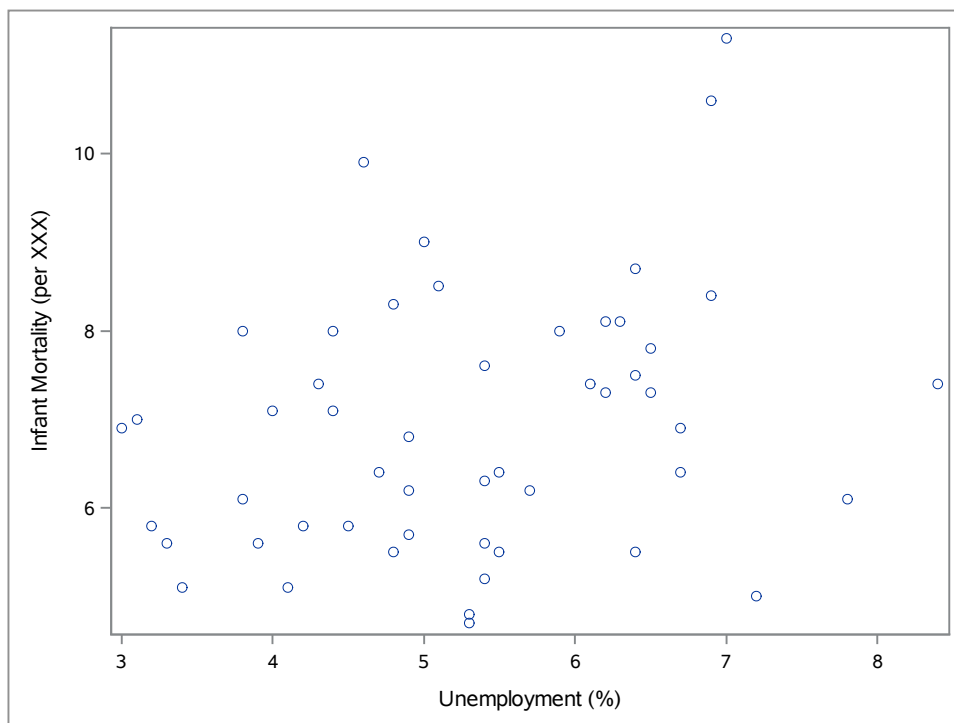


Figure 2: Axes fixed

The scatter plot isn't bad, but we can easily include more information. One thing we might want to do is add a smoothed line for the relationship between the two variables; in fact, we might want more than one, with different amounts of smoothing. One kind of smoothed line is loess. Another is linear regression. We can add loess lines and a regression line as follows:

```
proc sgplot data = UnempIM;
  xaxis label = "Unemployment (%)";
  yaxis label = "Infant Mortality (%)";
  scatter x = Unemployment y = InfantMortality;
  loess x = Unemployment y = InfantMortality/nomarkers;
  loess x = Unemployment y = InfantMortality/nomarkers;
  reg x = Unemployment y = InfantMortality;
  *LOESS WORKS ON THE SAME DATA AS SCATTER, SMOOTH CAN BE ADJUSTED.  NOMARKERS PREVENTS
  SAS FROM PLOTTING EACH POINT 3 TIMES.
  REG PLOTS A LINEAR REGRESSION LINE, IT DOES NOT NEED NOMARKERS;
run;
```

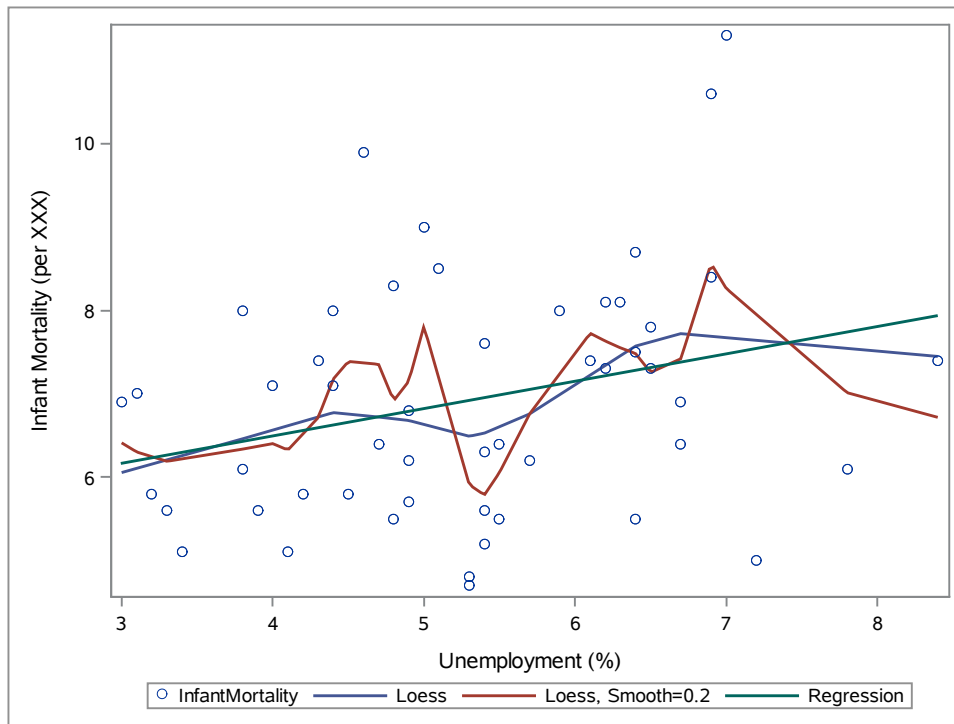


Figure 3: Scatter plot with loess lines

That bumpy smooth is too bumpy, so let's delete it. And we might also want to add an ellipse around the points; by default, the ellipse statement creates a prediction ellipse, that is, an ellipse for predicting a new point. It also approximates a region that contains 95% of the population:

Figure 4 is produced by the following code:

```
proc sgplot data = UnempIM;
  xaxis label = "Unemployment (%)";
  yaxis label = "Infant Mortality (%)";
  scatter x = Unemployment y = InfantMortality;
  loess x = Unemployment y = InfantMortality/nomarkers;
  reg x = Unemployment y = InfantMortality ;
  ellipse x = Unemployment y = InfantMortality;
run;
```

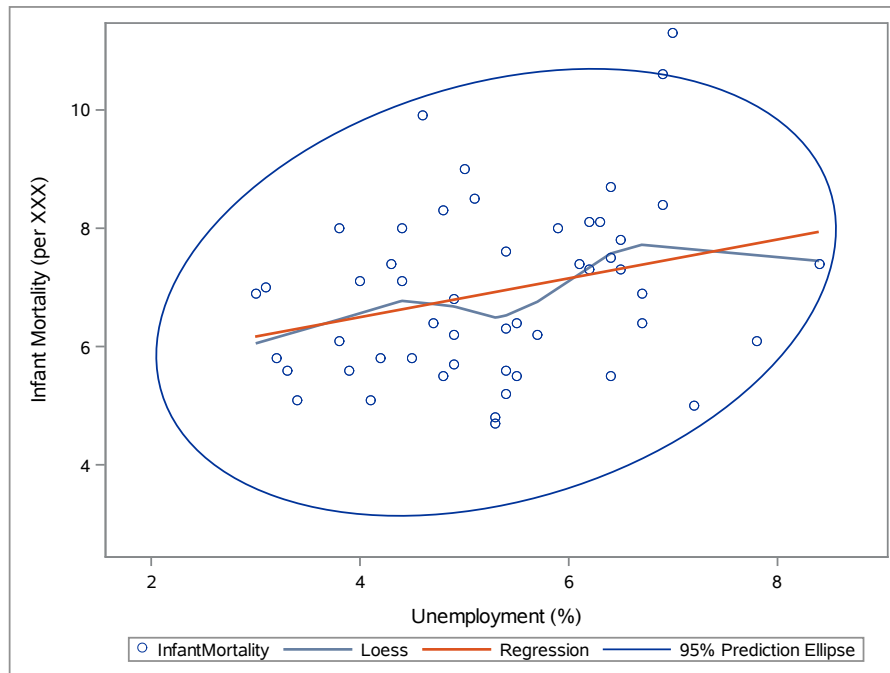


Figure 4: Scatter plot with loess lines and ellipse

That's all simple enough, and certainly adds information. But we can add more; we can look at the distribution of each variable separately and plot these in the margins. This requires use of the graph template language.

MORE COMPLEX ENHANCEMENTS WITH THE GRAPH TEMPLATE LANGUAGE (GTL)

The GTL allows very fine control over every aspect of a graph. It is also the language that SAS uses to create graphics. To use the GTL, you begin with a PROC TEMPLATE. You then use PROC SGRENDER to render (plot) that template. An advantage of this is that, once you have created a template, you can very easily use it with different data sets. A very good reference that includes many starting templates is Kuhfeld [1]. In addition, the GTL Users' Guide and GTL Reference Manual are quite useful. There are a great many possible options, and I will cover only a few.

Suppose we want to produce a graph such as that shown in figure 5

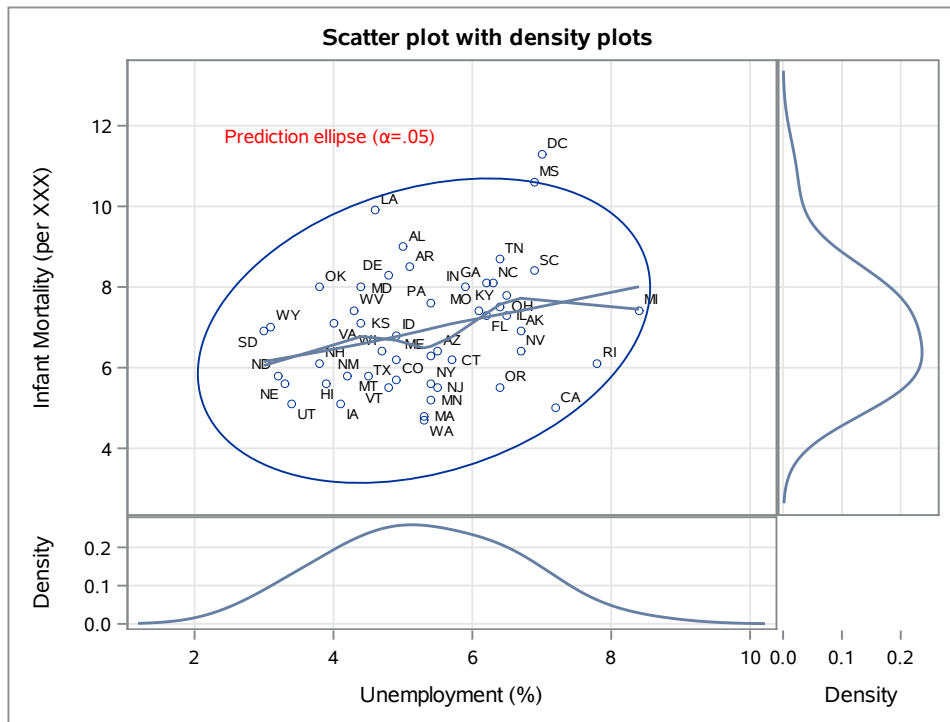


Figure 5: Scatter plot with density plots

I think this is a pretty sophisticated graph. There's a lot of information and it's reasonably clear what that information is. That is, it includes both information on the univariate distributions as well as the bivariate distribution. To produce this figure, we first create a template PROC TEMPLATE.

```
proc template;          *STARTS PROC TEMPLATE;
define statgraph scatdens2; *DEFINES A GRAPH TO BE CALL SCATDENS;
begingraph;    *BEGIN DEFINING THE GRAPH;
  entrytitle "Scatter plot with density plots"; *CREATE A TITLE;
  layout lattice/columns = 2 rows = 2 columnweights = (.8 .2) rowweights = (.8 .2)
    columndatarange = union rowdatarange = union;
  *LAYOUT LATTICE/COLUMNS = 2 ROWS = 2 SETS UP A GRID, OR LATTICE, OF GRAPHS;
  *COLUMNWEIGHTS AND ROWWEIGHTS SETS THE RELATIVE SIZE OF THE INDIVIDUAL COLUMNS AND ROWS;
  columnaxes;
    columnaxis /label = 'Unemployment (%)' griddisplay = on;
  columnaxis /label = '' griddisplay = on;
  endcolumnaxes;
  *COLUMNAXES SETS THE CHARACTERISTICS OF COLUMNS;
  *THE SECOND ONE HAS NO LABEL (NONE WOULD FIT)
  rowaxes;
    rowaxis /label = 'Infant Mortality (%)' griddisplay = on;
  rowaxis /label = '' griddisplay = on;
  endrowaxes;
  layout overlay; *STARTS THE ACTUAL GRAPHING OF DOTS AND SUCH;
    scatter plot x = unemployment y = infantmortality; *GRAPHS THE DOTS;
  loessplot x = unemployment y = infantmortality/nomarkers;
  loessplot x = unemployment y = infantmortality/smooth = 1 nomarkers;
  ellipse x = unemployment y = infantmortality/type = predicted;
  endlayout;
enddefine;
```

```

    densityplot infantmortality/orient = horizontal;
    densityplot unemployment;
endlayout;
endgraph;
end;
run;

```

Then we render it with PROC SGRENDER.

```

proc sgrender data = UnempIM template = scatdens2; *NOW WE RENDER THE TEMPLATE WE CREATED;
run;

```

Rick Wicklin of SAS pointed out that, for people who don't like to program, the %sgdesign macro brings up a GUI interface that allows you to create the second image using drag-and-drop and menus. For details and examples, see [2], I have not used this feature.

OVERPLOTING

Although scatter plots are very useful, they can have problems. The most important of these is *overplotting* which occurs when more than one observation has the same or very similar values. The proper solution depends on the type and amount of overplotting. In some cases, overplotting is due to the discrete nature of the way the data are recorded; for example, when asked their weights and heights, people respond with weights in pounds (or kilograms) and heights in feet and inches (or centimeters), rounded to the nearest unit, or sometimes even to the nearest multiple of 5. In other cases, there is so much data that overplotting occurs even when the data are recorded accurately to several decimal places.

In the first situation, one excellent solution is *jittering* or adding small amounts of random noise to the data. The proper amount to add is partly a matter of trial and error; you want enough jitter so that the overlap is gone, but not so much that the data are distorted. In the latter cases, there are various solutions. If the data set is not enormous, changing the plotting character or its size may be enough. If there is an enormous number of points, then we can change to a parallel box plot.

A DATA SET

Here I create a data set of actual heights and weights (realht and realwt), rounded to the nearest inch and pound (ht and wt). I also jitter these (jitht and jitwt). Ht and wt represent data as it would be reported.

```

data htwt;
  do i = 1 to 10000;
    realht = rannor(1828282)*3 + 66;
    realwt = realht*2 + realht**2*.01 + 10*rannor(12802194);
    ht = round(realht,1);
    wt = round(realwt,1);
    jitht = ht+rannor(1818282);
    jitwt = wt+rannor(199328282);
    output;
  end;
run;

```

MODERATE OVERPLOTING DUE TO DISCRETIZATION

If we have a data set of 500 people with rounded height and weight, the plot will not show all the points clearly; see figure 6

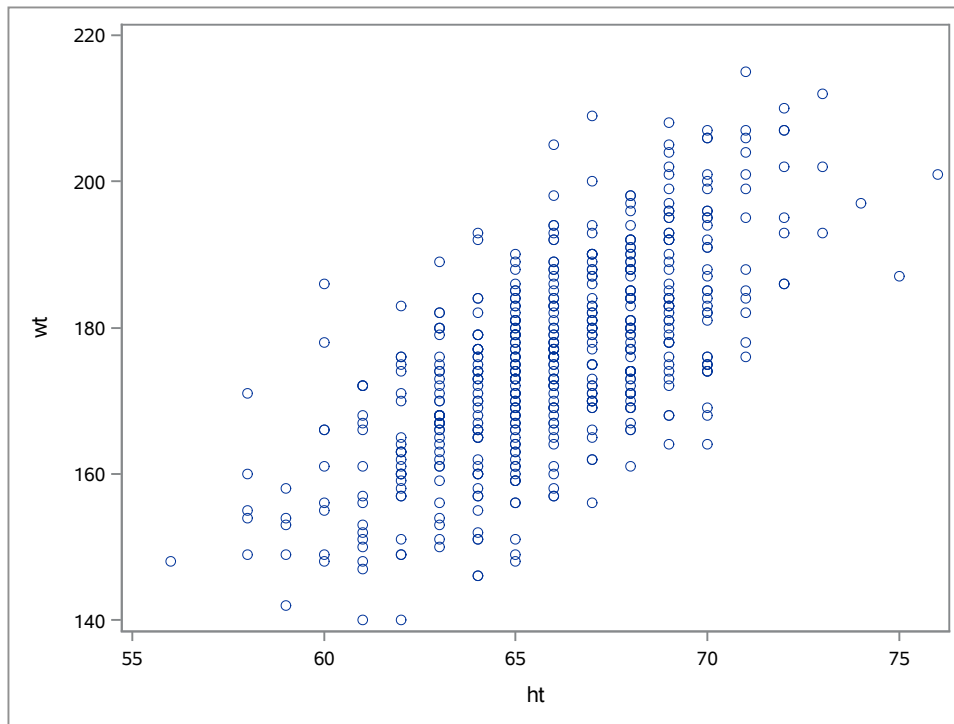


Figure 6: Scatter plot with moderate overplotting

Here, simply jittering the data works well; see figure 7.

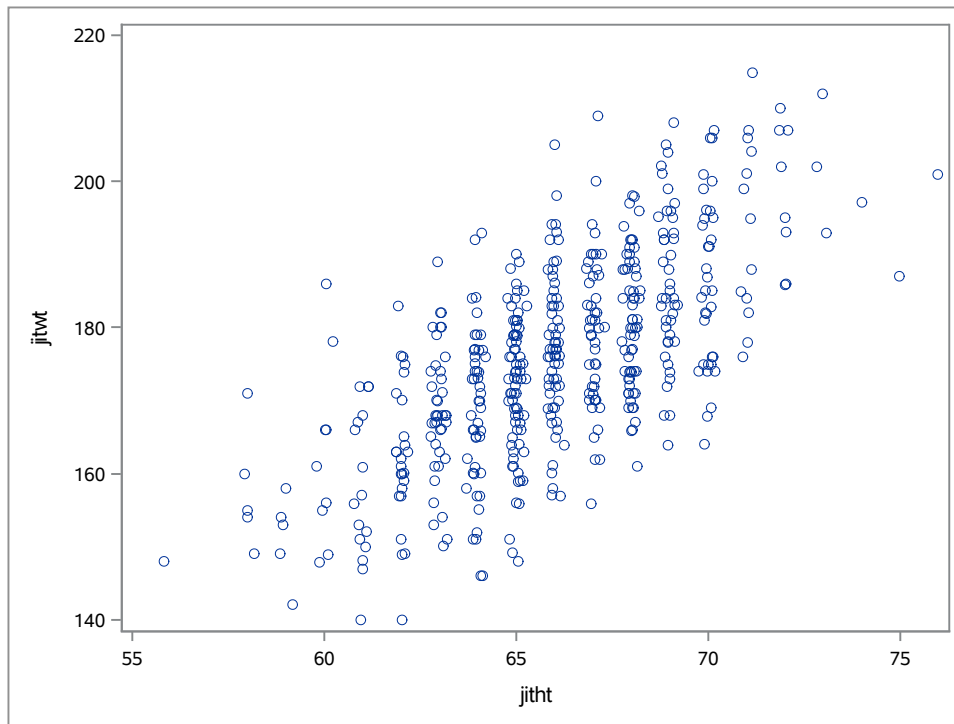


Figure 7: Scatter plot with jittering

However, if we have 10,000 points, jittering is not enough, see figure 8.

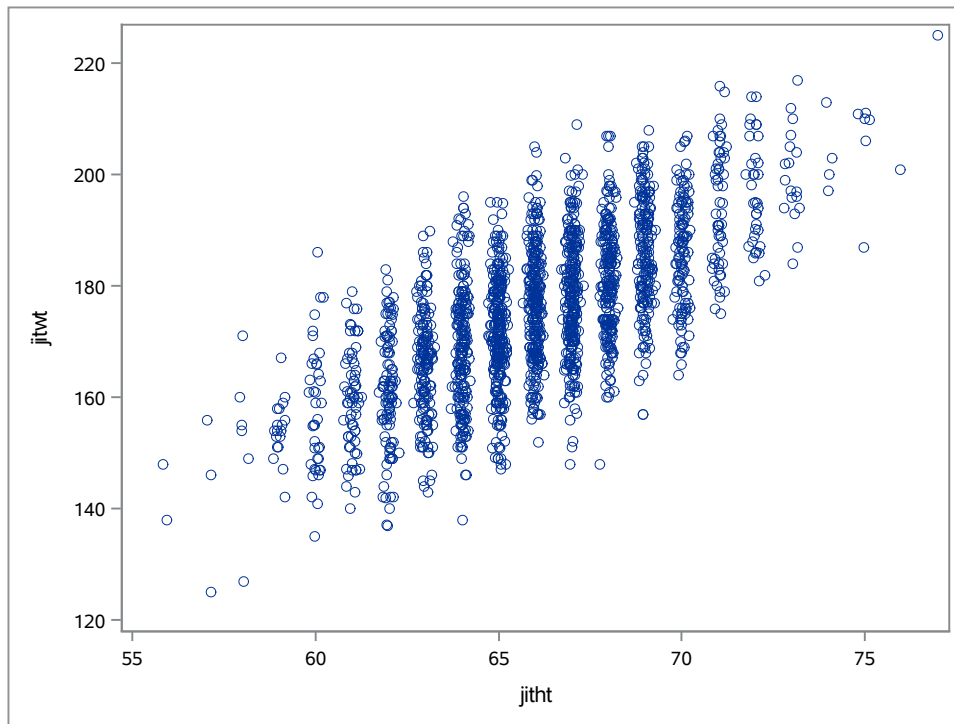


Figure 8: Scatter plot with jittering, but $N = 10000$

We can change the plotting character and its size with the following program:

```
proc sgplot data = htw;
  scatter x = jtht y = jitwt / markerattrs = (size = 2 symbol = circlefilled);
run;
```

producing figure 9

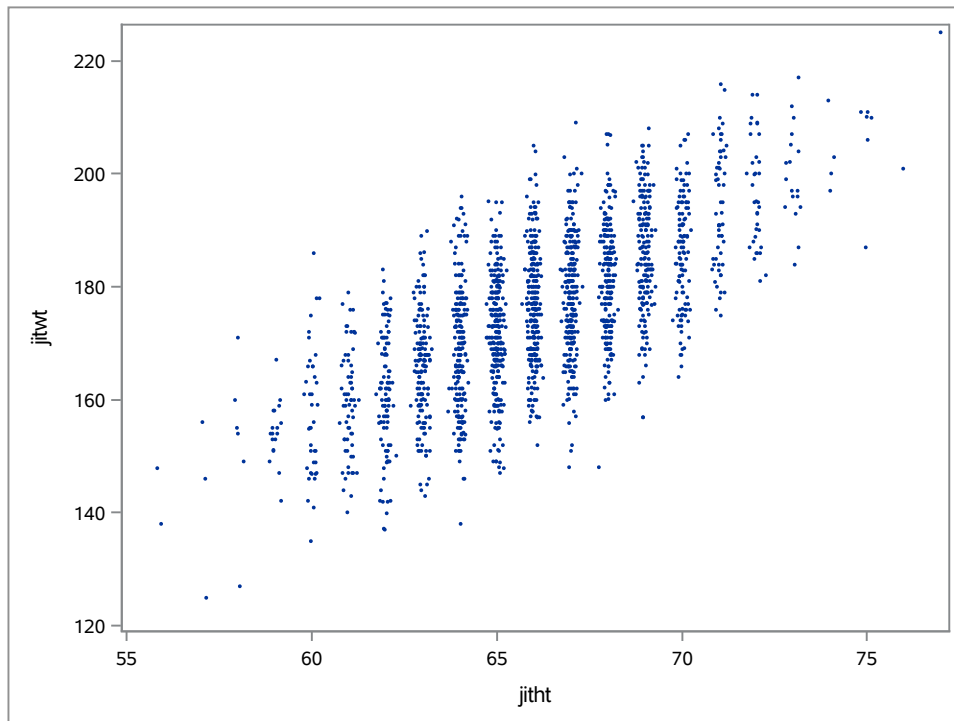


Figure 9: Scatter plot with small symbols, N = 10000

An alternative is to abandon the scatter plot and use parallel boxplots:

```
proc sgplot data = htw;
  vbox wt/category = ht spread; *THE SPREAD OPTION PREVENTS OVERLAP;
run;
```

This produces figure 10

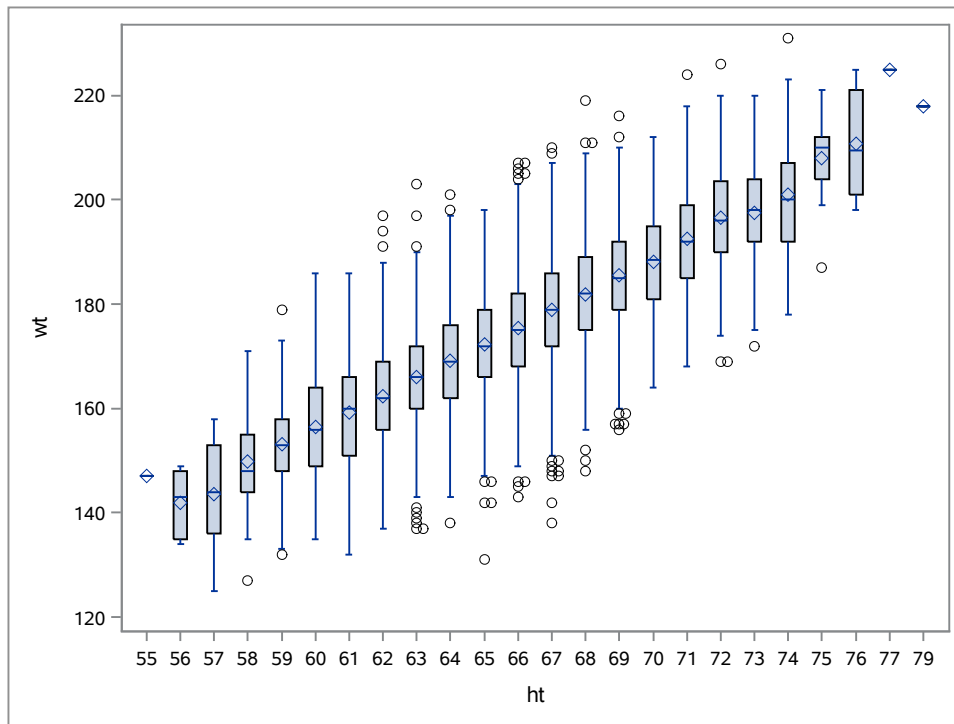


Figure 10: Parallel boxplot, N = 10000

One problem with this is that it does not give any indication of the density of height. Using GTL, we can add a barchart:

```
proc template;
  define statgraph fancybox;
    begingraph;
      entrytitle "Box plot w/histogram";
      layout lattice/rows = 2 columns = 1 order = columnmajor rowweights = (.8 .2);
      columnaxes;
        columnaxis /griddisplay = on;
        columnaxis /label = '' griddisplay = on;
      endcolumnaxes;
      boxplot x = ht y = wt;
      barchart x = ht y = wt;
    endlayout;
  endgraph;
end;
run;

proc sgrender data = htwt template = fancybox;
run;
```

This produces figure 11

The SAS System

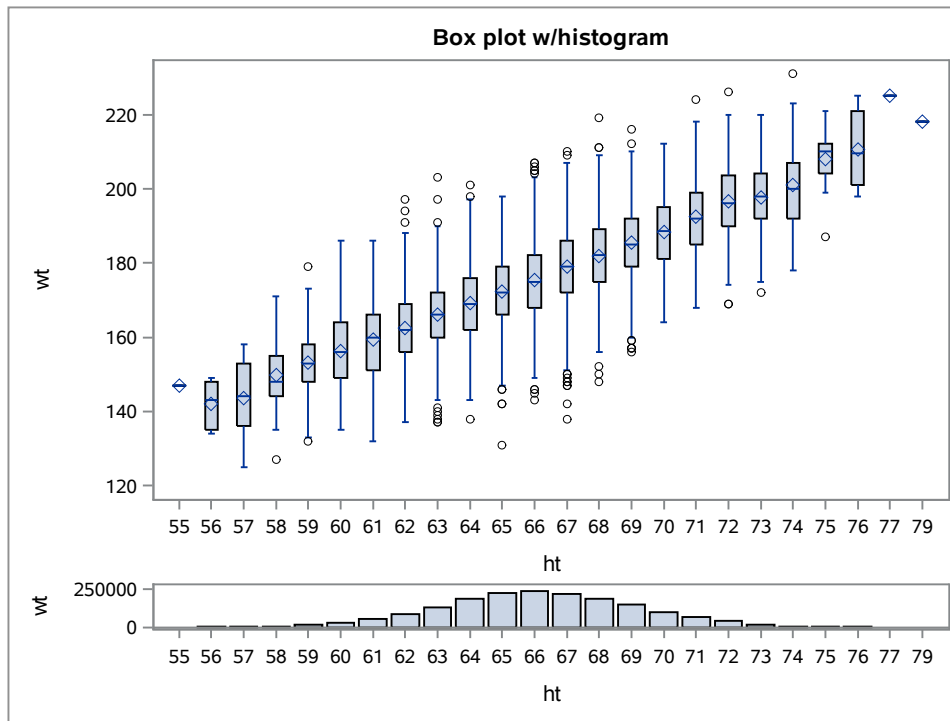


Figure 11: Parallel boxplot, N = 10000

This will look better if we externalize the axes:

```
proc template;
  define statgraph fancybox2;
    begingraph;
      entrytitle "Box plot w/barchart";
      layout lattice/rows = 2 columns = 1 rowweights = (.8 .2)
        rowdata range=union
        rowgutter=3px
        coldatarange = union;
      rowaxes;
        rowaxis / griddisplay=on label="Weight"
        rowaxis / griddisplay=on label="%"
      endrowaxes;
      boxplot x = ht y = wt;
      barchart x = ht y = wt;
    columnaxes;
      columnaxis /griddisplay = on;
    endcolumnaxes;
  endlayout;
endgraph;
end;
run;
```

to get figure 12, below.

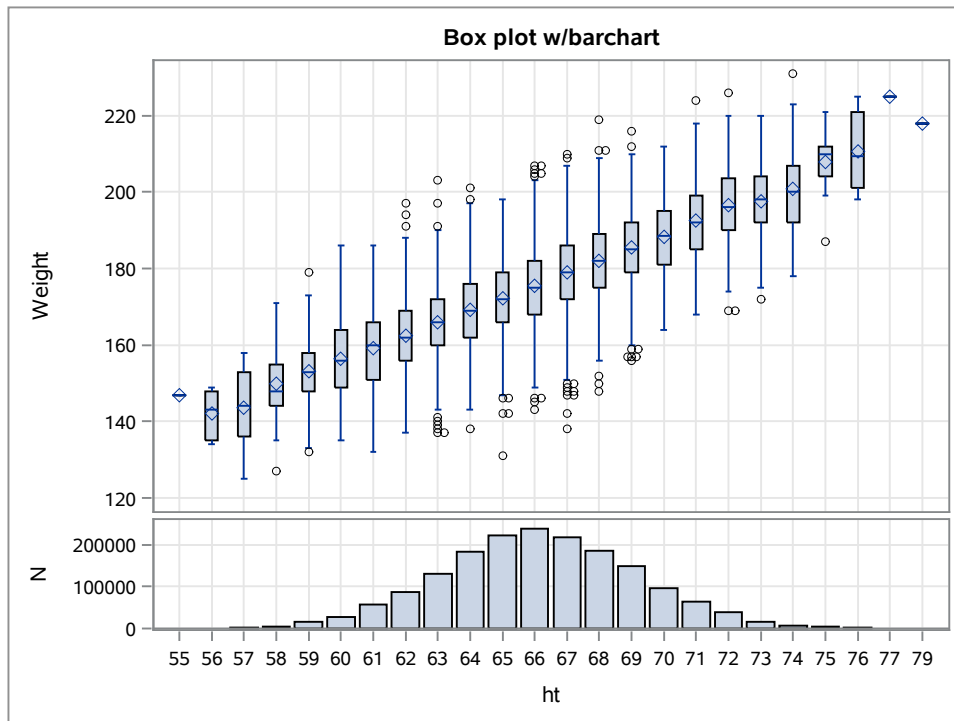


Figure 12: Parallel boxplot, N = 10000

SUMMARY

Scatter plots are a very valuable graphical tool. The SG procedures in SAS allow many scatter plots to be produced easily, and the graph template language allows very fine control over all aspects of a graph.

REFERENCES

- [1] Kuhfeld, W. "Statistical Graphics in SAS: An Introduction to the Graphical Template Language", SAS Press, Cary, NC, 2010.
- [2] <http://support.sas.com/documentation/cdl/en/grstatdesignug/62589/HTML/default/viewer.htm>

CONTACT INFORMATION

Peter L. Flom
 515 West End Ave
 Apt 8C
 New York, NY 10024 peterflomconsulting@mindspring.com
 (917) 488 7176

SAS® and all other SAS Institute Inc., product or service names are registered trademarks or trademarks of SAS Institute Inc., in the USA and other countries. ® indicates USA registration. Other brand names and product names are registered trademarks or trademarks of their respective companies.