

## Box Plots in SAS®: UNIVARIATE, BOXPLOT, or GPLOT?

Robert Adams, Merck & Co., Inc., North Wales, PA

### ABSTRACT

The continuous nature of some clinical trial data makes it well-suited for display using box plots. Box plots convey, at a glance, a wealth of information about the data being graphed. This includes the data's central value, distribution, and variability, as well as how categorical variables compare side-by-side. This paper examines three procedures for creating box plots in SAS: PROC UNIVARIATE, PROC BOXPLOT, and PROC GPLOT. Features and limitations, as well as similarities and differences, are discussed and demonstrated for each of the procedures. As a result, participants are better prepared to choose the SAS procedure which can most effectively present the clinical trial results.

### KEYWORDS

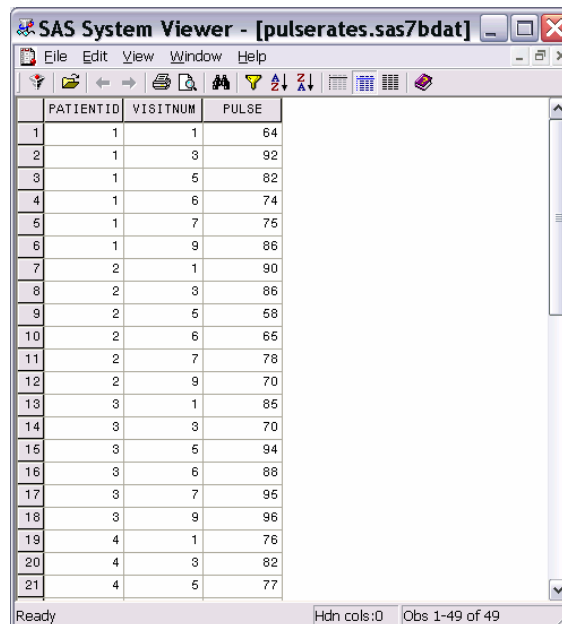
Median, 25<sup>th</sup> percentile, 75<sup>th</sup> percentile, box plot

### INTRODUCTION

Most clinical trials frequently review key measures to monitor patients' response to a drug and alter the trial if necessary. These measures often include continuous data such as lab tests or vital signs. Box plots are a useful tool for communicating these results since they display key measures of the data and show how the data changes by comparing it side-by-side. These key measures include the median, the 25th and 75th percentiles, and the minimum and maximum data values. In this paper, a box plot of patient pulse data over time is reproduced with Windows PC SAS 9.1.3 using 3 different methods: PROC UNIVARIATE, PROC BOXPLOT, and PROC GPLOT. In the process, capabilities as well as limitations of each of the procedures are elicited.

### DATASET

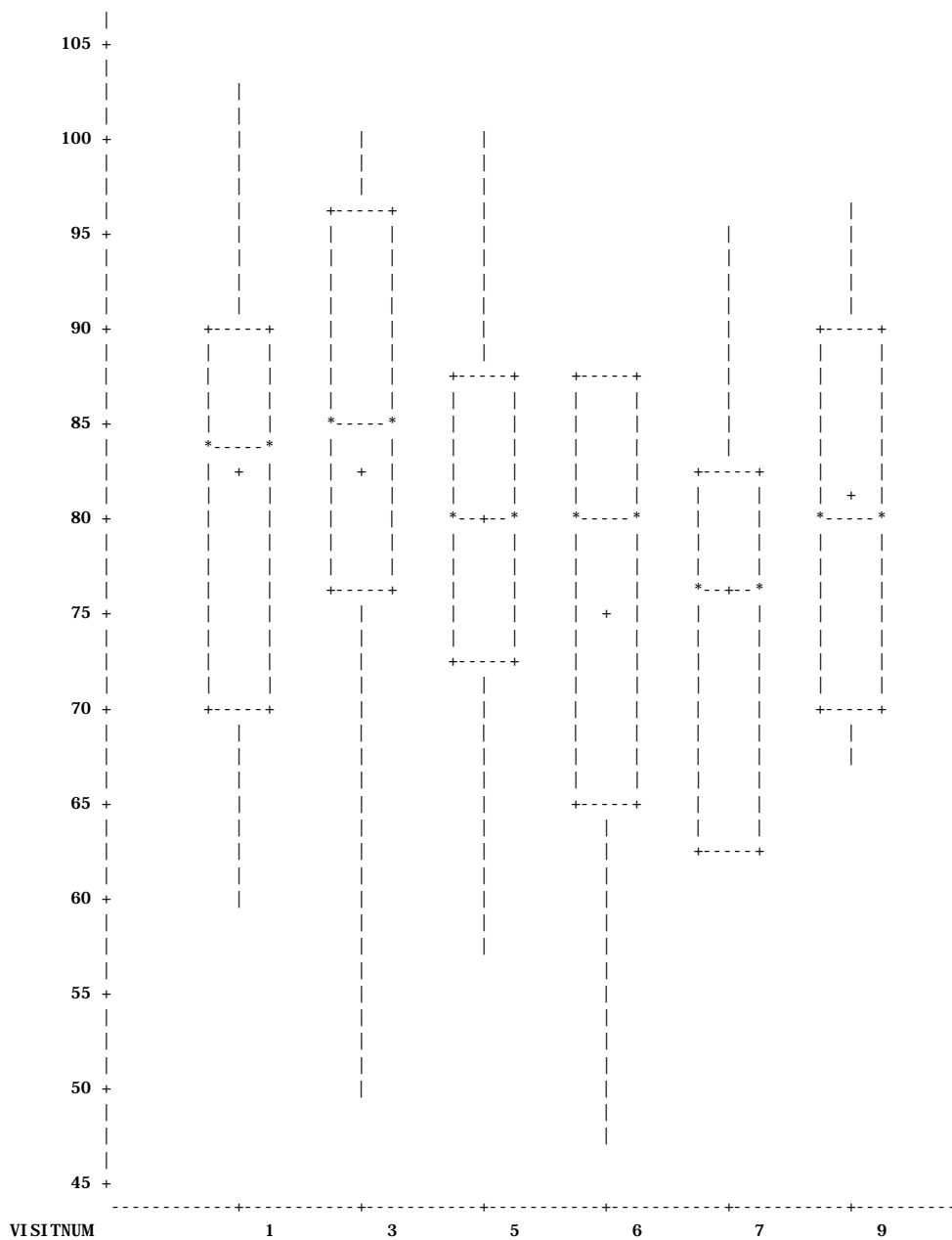
Figure 1 shows part of the example dataset. The dataset contains 8 patients' pulse rate data over each of 6 clinical study visits.



PATIENTID	VISITNUM	PULSE
1	1	64
2	1	92
3	1	82
4	1	74
5	1	75
6	1	86
7	2	90
8	2	86
9	2	58
10	2	65
11	2	78
12	2	70
13	3	85
14	3	70
15	3	94
16	3	88
17	3	95
18	3	96
19	4	76
20	4	82
21	4	77

Figure 1: pulserates.sas example dataset





**Figure 3:** Side-by-side box plots produced when a BY group is combined with the PLOT option on PROC UNIVARIATE

This is the desired box plot. The center horizontal line in each box corresponds to the sample median and the central plus sign corresponds to the sample mean. It also indicates that, at Visit 1 for example, the minimum observation was about 60, 25% of the data falls below 70, 75% of the data falls below 90, and the maximum value was about 104.

To limit the output to just the desired plot in Figure 3, the output table name assigned by PROC UNIVARIATE is referenced. This name is used to request the desired table from the Output Delivery System(ODS). The ODS table name for the side-by-side box plots is *SSPlots* and is used as follows:

```
ods select ssplots ;
```

This outputs only the side-by-side box plots. So the complete call to create the desired box plots with PROC UNIVARIATE is:

```
ods select ssplots ;

proc univariate data = pulserates plot ;
    var PULSE ;
    by visitnum ;
run;
```

When examining the resulting output, it is apparent that the graphs are low-resolution. This is because box plots produced this way are a legacy feature of the UNIVARIATE procedure in earlier versions of SAS. A title statement can be used to add a title to the plot but, other than that, formatting using PROC UNIVARIATE is not very robust. There is no way to label the axes, to add text to the plot, or to add color of any kind. However, it's a familiar procedure to many people and, coupled with the information output in the univariate analysis of the procedure, provides a visualization of a sample as well as valuable statistics about that sample.

## PROC BOXPLOT

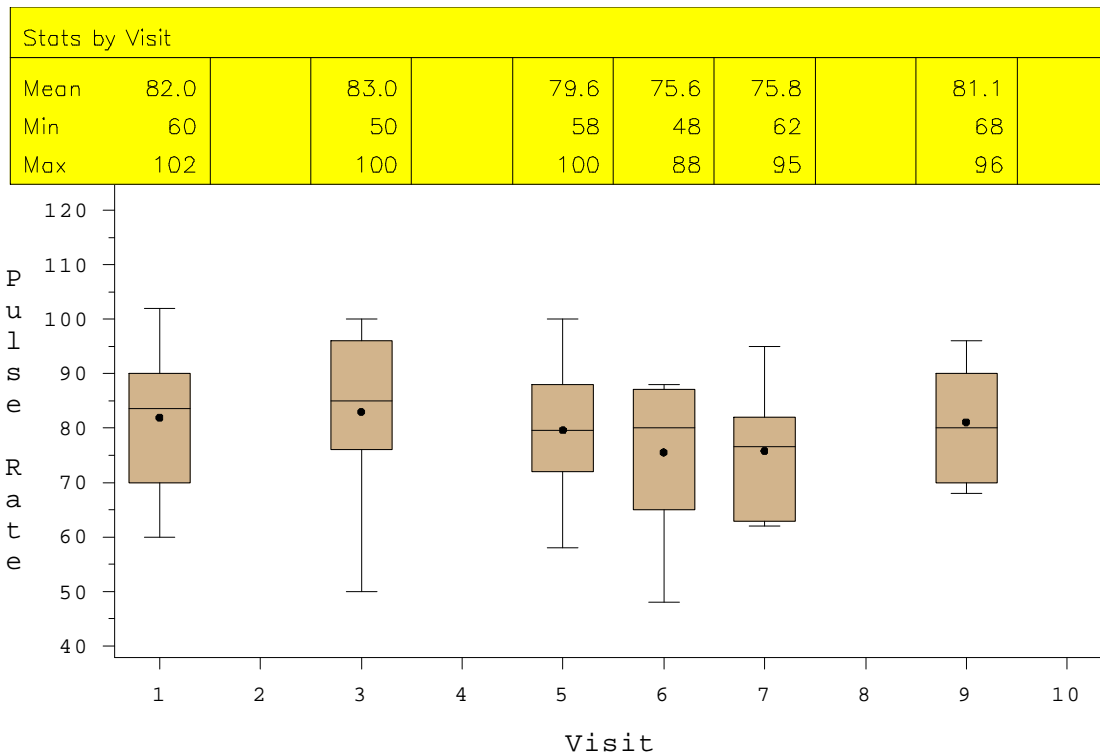
The next procedure is the SAS/STAT procedure PROC BOXPLOT. The syntax of the procedure is:

```
PROC BOXPLOT < options > ;
    PLOT analysis-variable*group-variable < (block-variables ) >
        < =symbol- variable > < / options > ;
    INSET keywords < / options > ;
    INSETGROUP keywords < / options > ;
    BY variables;
    ID variables;
run;
```

Required components of PROC BOXPLOT are the PLOT statement, the analysis variable, and the group variable, with the PLOT statement being the heart of the procedure. The desired plot is again pulse rate over time, so the analysis variable is PULSE. The group variable identifies groups in the data, which is Visit, so the group variable is VISITNUM.

Many options exist for the PLOT statement, which are listed following a slash. Options exist for controlling box appearance, for plotting and labeling points, for creating reference lines, for using a legend, and for creating and labeling axes, among many others.

Figure 4 shows the box plot meeting the requirements.



**Figure 4:** A box plot created with PROC BOXPLOT

The code producing the plot in Figure 4 is:

```

axis1      order = (40 to 120 by 10)
           label = (height = 1.25 'Pulse Rate')
           minor = (number = 1) ;
axis2      order = (1 to 10 by 1)
           label = (height = 1.25 'Visit') ;

symbol     value = dot
           height = 0.5 ;

proc boxplot data = pulserates ;

    plot (PULSE) * visitnum / vaxis = axis1
                                haxis = axis2
                                cboxfill = TAN
                                cboxes = BL ;

    insetgroup mean (5.1) min max / header = 'Stats by Visit'
                                pos = top
                                cfill = YELLOW ;

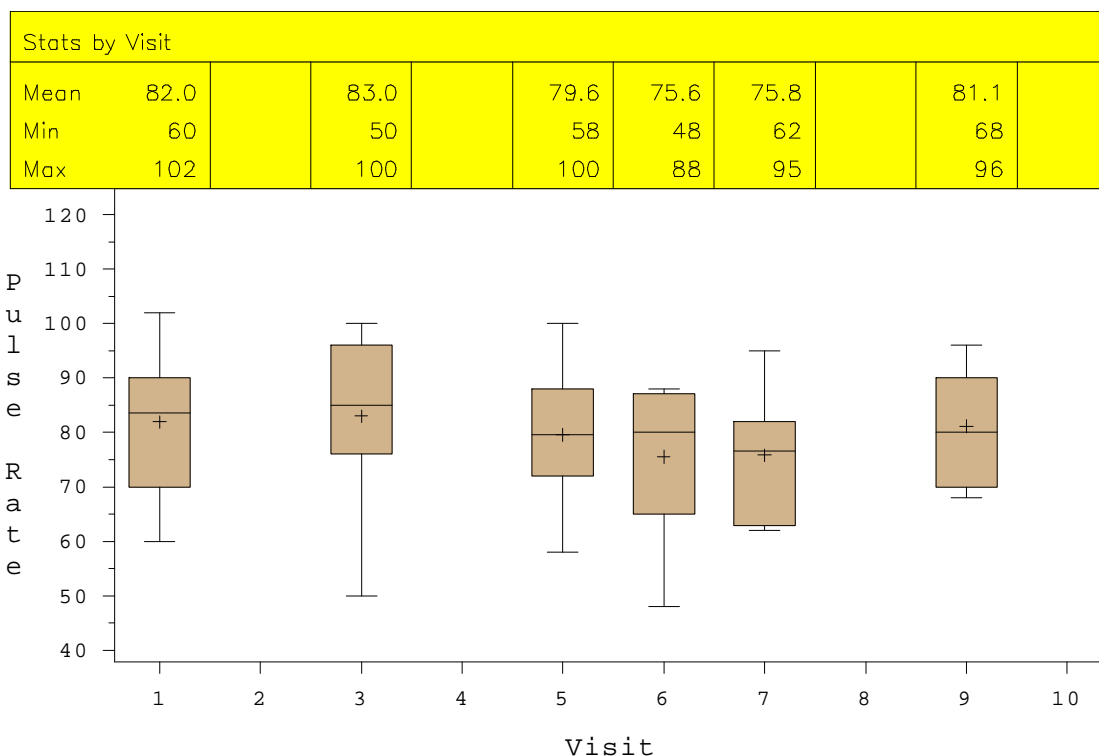
run;

```

The most important parts of the procedure are the *vaxis* and *haxis* options on the PLOT statement. These options either specify tick mark values for their respective axes or specify an axis name defined in a previous AXIS statement. In the above example, the latter is used. It is not documented that the AXIS statement can be used with PROC BOXPLOT, but it can be. The PLOT statement options *vaxis* = 40 to 120 by 10 and *vminor* = 1

accomplish the same things as the above AXIS statement, but there are advantages to using an AXIS definition. It allows for specifying a LABEL = text option to add a descriptive label to the axis. Without this, the axis label is either the variable name or a previously assigned variable label. In addition, because the AXIS statement is a global statement, the axis definition is decoupled from the procedure. As a result, axes are defined once and explicitly assigned in the procedure instead of coded into each call of the BOXPLOT procedure (there can be up to 99 AXIS definitions).

A SYMBOL statement is also used in the above code. This statement is not required. It is used to change the default symbol for the mean from a plus(+) sign to a dot. A box plot without the SYMBOL specification is shown in Figure 5.



**Figure 5:** Displaying the default mean symbol by not using the SYMBOL statement

A different way of changing the default symbol is to specify an optional variable in the input dataset that specifies the symbol marker to be used to plot the means. Symbols that can be used include +, x, and \*; a complete list is available in the SAS/GRAPH OnlineDoc 9.1.3 in the section entitled *Special Symbols for Plotting Data Points* (symbols that can be displayed will depend in part on the operating system being used).

Like the AXIS statement, it is not documented that the SYMBOL statement is used with PROC BOXPLOT. However, it was just shown that the SYMBOL statement can be used with PROC BOXPLOT. Unlike the AXIS statement, the SYMBOL statement is automatically applied to a graph following the SYMBOL definition. Although not required, it provides a greater degree of customization as well as standardization across plots since it is also a global statement.

A third SAS/GRAPH statement can be used to enhance PROC BOXPLOT: the GOPTIONS statement. The syntax is:

```
GOPTIONS <options-list> ;
```

Using GOPTIONS is another way to decouple graph definitions from the procedure. It sets default values for many graphics attributes and device parameters used by SAS/GRAPH procedures (as well as PROC BOXPLOT). A graphics option remains in effect until either the option is specified in another GOPTIONS statement, the RESET= option is used with the GOPTIONS statement to reset the values, or the SAS session is ended. Keep in mind that if RESET = ALL is used, it cancels global statement definitions such as AXIS, FOOTNOTE, SYMBOL, and TITLE in addition to resetting all graphics options to their default values. It is usually a good idea to place a RESET= option first in the GOPTIONS statement.

GOPTIONS has options that set many but not all of the definitions needed for most graphs. For example, it is used to set the horizontal/vertical offsets for a graph, to set the graph's background, symbol, title, and text colors, and to define the interpolation method for the graph, but fine-tuning probably needs to be handled by AXIS and/or SYMBOL statements. When a GOPTIONS statement is used with global AXIS or SYMBOL statements, SAS/GRAPH first searches the statement options and then looks at the value of the corresponding graphics option.

Getting back to the graph of pulse rate by visit, another feature of PROC BOXPLOT that is used is the ability to include tables of summary statistics on the box plot using the INSETGROUP statement. This can't be done with box plots created by PROC UNIVARIATE or PROC GPLOT. It allows SAS to calculate and display the mean, min, and max values (among others) for each group so the user doesn't have to guess at where the plot crosses the y-axis.

One other note about box plots produced with PROC BOXPLOT: they can contain very detailed color specifications for different parts of the graph using the plot statement options.

## PROC GPLOT

Box plots are also generated using PROC GPLOT, from SAS/GRAPH. The syntax of the GPLOT procedure is:

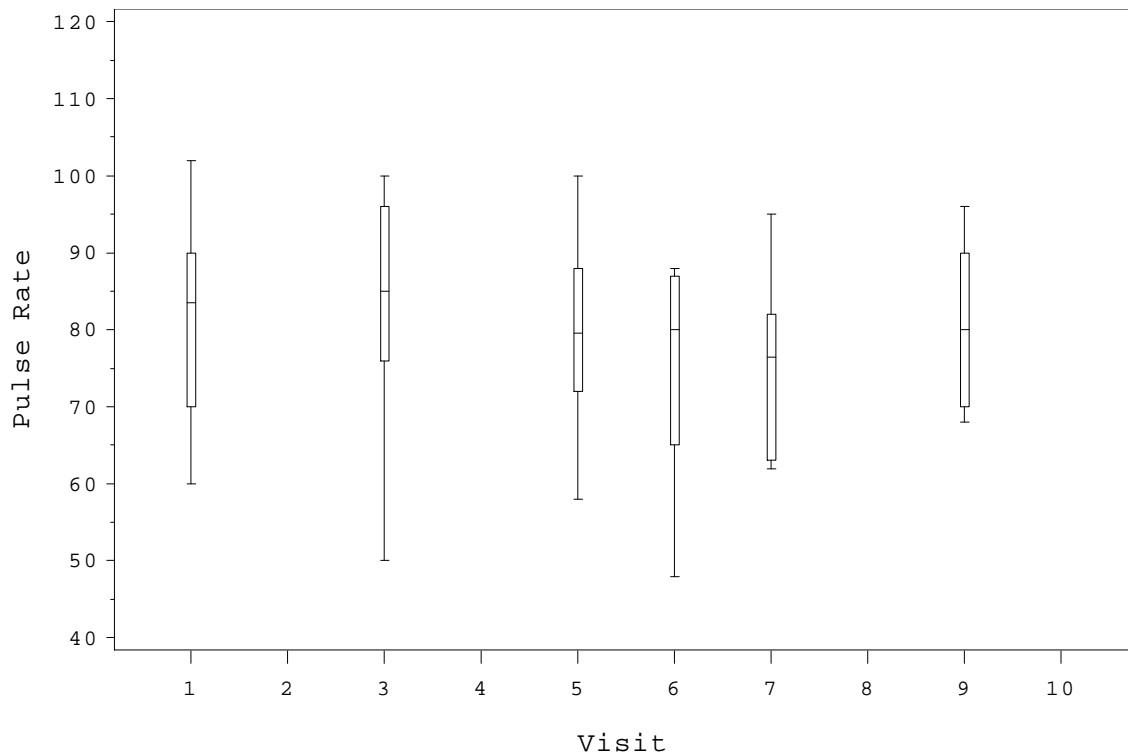
```
PROC GPLOT <DATA=input-data-set> <UNIFORM> ;
      PLOT plot-request(s) </option(s)> ;
quit;
```

PROC GPLOT creates many types of plots. In order to produce a box plot, an interpolation method needs to be specified. One way to do this is in a SYMBOL statement. The statement needed is:

```
symbol      value = dot
            height = 0.4
            interpol = boxtf
            width = 3
            bwidth = 5
            co = BL
            cv = TAN ;
```

The statement *interpol = boxtf* means to create a box plot and the "t" and "f" suffixes mean to draw tops and bottoms on the whiskers, to fill the box with the color specified by CV=, and to outline the box with the color specified by CO=, respectively. It is important to remember that PROC GPLOT works in conjunction with the SYMBOL statement (or a GOPTIONS statement), so one or the other is required. A similar graph (Figure 6) is created by using the following GOPTIONS statement instead of the SYMBOL statement, however GOPTIONS alone does not allow for all of the fine-tuning available with the SYMBOL statement:

```
goptions reset = all interpol = boxt ;
```



**Figure 6:** A box plot created using the `interpol` option of the `goptions` statement

Figure 6 gives an appreciation for some of the `SYMBOL` statement options. For example, the `WIDTH=` and `BWIDTH=` `SYMBOL` statement options are used to adjust the box plot thickness factor and box width, respectively.

The plot statement is also required to produce a box plot. The form of the plot statement used with `PROC GPLOT` is as follows:

```
y-variable * x-variable <=n>
```

The resulting plot statement (and overall procedure specification) needed is:

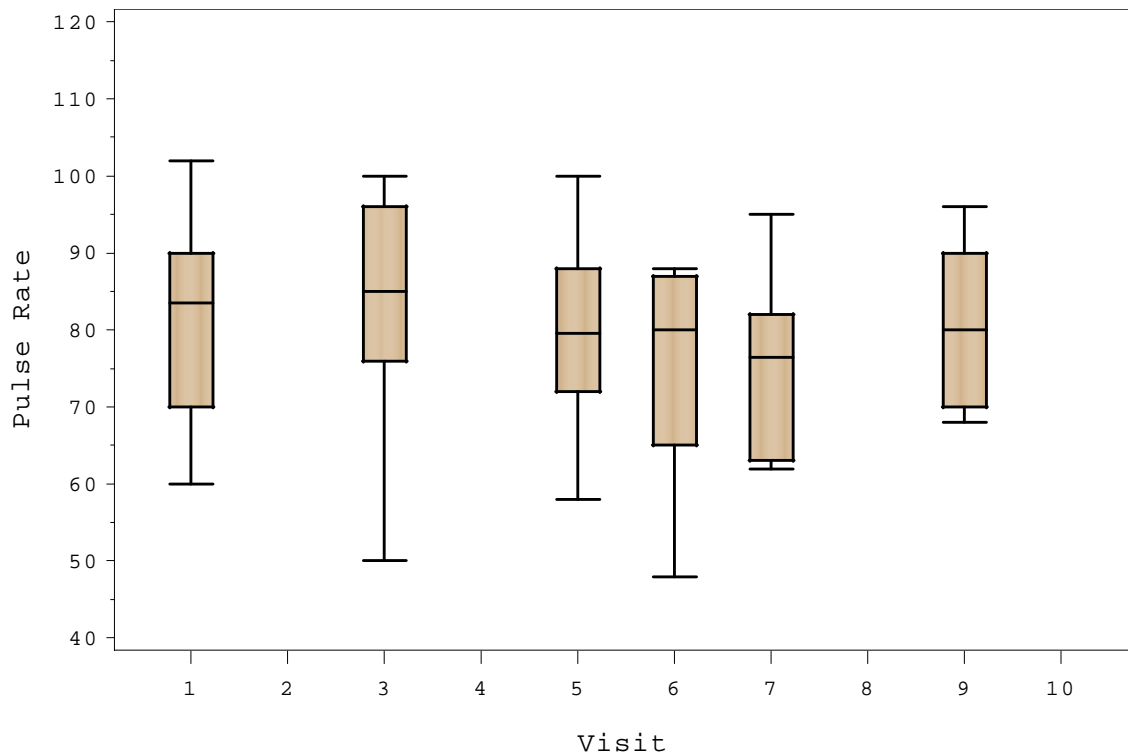
```
symbol      value = dot
            height = 0.4
            interpol = boxtf
            width = 3
            bwidth = 5
            co = BL
            cv = TAN ;

axis1      order = (40 to 120 by 10)
            label = (height = 1.25 angle = 90 'Pulse Rate')
            minor = (number = 1) ;

axis2      order = (1 to 10 by 1)
            label = (height = 1.25 'Visit')
            offset = (5, 5) ;
```

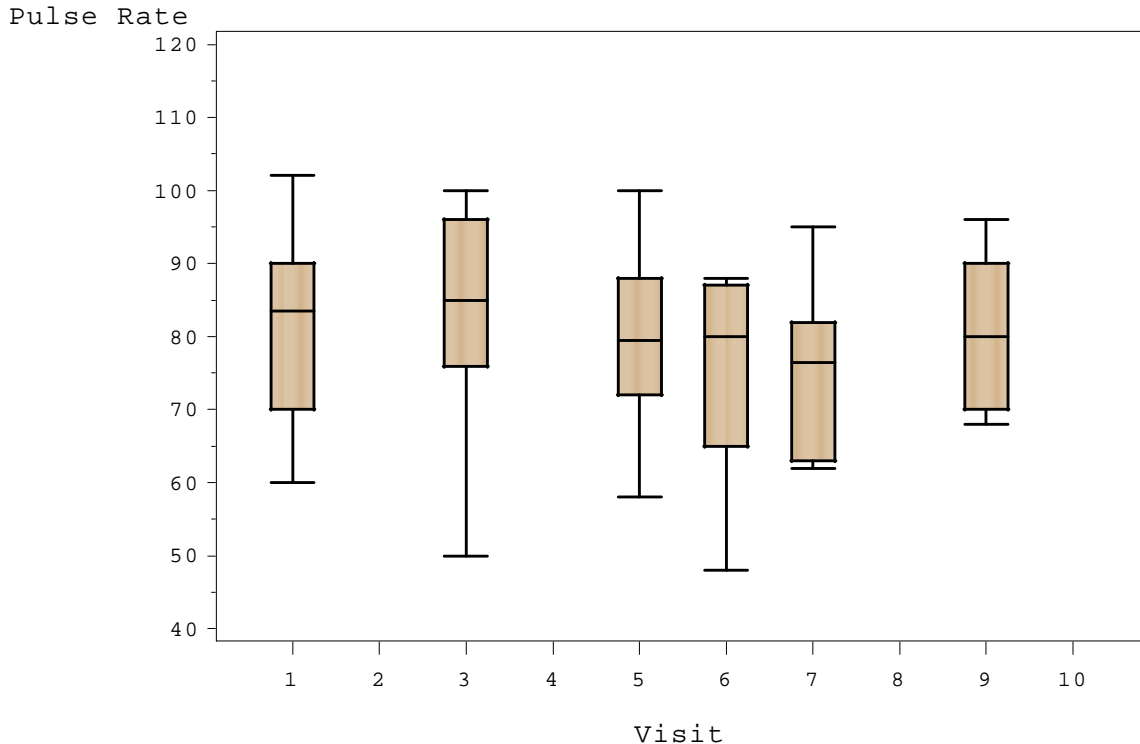
```
proc gplot data = pulserates uniform ;  
  plot PULSE * visitnum /      haxis = axis2  
                               vaxis = axis1  
                               hminor = 0  
                               skipmiss ;  
  
run;  
quit;
```

The above code produces the plot shown in Figure 7.



**Figure 7:** A box plot created using PROC GLOT

This plot is, for the most part, similar to the plot produced by PROC BOXPLOT, with a few exceptions. First of all, the vertical axis label is different. By default, the label appears above the axis, as shown in Figure 8.



**Figure 8:** A box plot created using PROC GPLOT and the default vertical axis label

The ANGLE = option is used on the AXIS statement to get the “Pulse Rate” label to rotate and appear beside the axis. Second, PROC GPLOT does not display mean values. This is somewhat of a limitation for this procedure. INSETGROUPs also can’t be used with GPLOT, which would be nice to compensate for not being able to graphically display the mean.

## CONCLUSIONS

This paper examines 3 methods for generating box plots in SAS: PROC UNIVARIATE, PROC BOXPLOT, and PROC GPLOT. Box plots created with PROC UNIVARIATE leverage familiarity with Base SAS, can be a great addition to the statistics output from the univariate analysis of the procedure, but are low-resolution and somewhat limited. The SAS/STAT procedure PROC BOXPLOT not only is a procedure designed specifically for high-resolution box plots, but also is further enhanced by borrowing some of the features of SAS/GRAPH. The SAS/GRAPH procedure PROC GPLOT creates high-resolution box plots in addition to many other types of plots, and can create box plots almost comparable to those produced by PROC BOXPLOT.

The examples and discussions in this paper allow the reader to see the capabilities of 3 procedures for creating box plots and prepare them to choose a method that will meet their needs most effectively.

## REFERENCES

SAS OnlineDoc 9.1.3.:  
<http://support.sas.com/onlinedoc/913/docMainpage.jsp>

**CONTACT INFORMATION**

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

Robert Adams  
Merck and Company, Inc.

UG1D-10  
351 North Sumneytown Pike  
North Wales, PA 19454  
267-305-8249  
robert\_adams2@merck.com