PhUSE 2009

Paper CS04

Zooming in on graphics

Armin Gemperli, BIOP AG, Basel, Switzerland

ABSTRACT

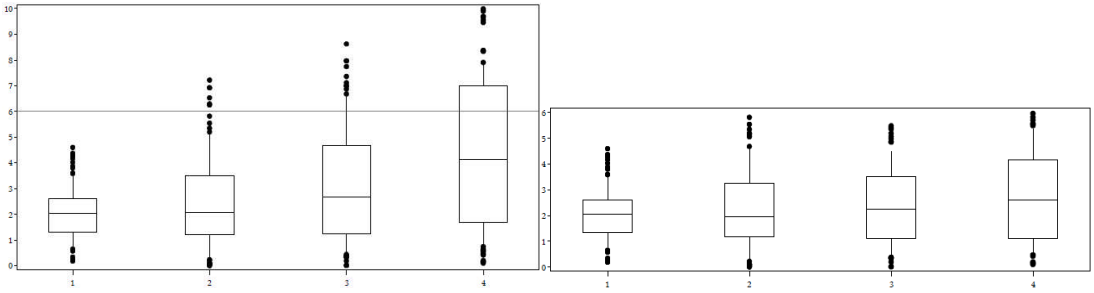
When producing multiple figures for the same outcome (e.g. for different subgroups), the axis range should be maintained across graphs to support comparability. An axis range that encloses every data point over all subgroups might be very large, as a consequence. As a downside, subgroups that cover only a small range of data values may not be very distinctive visually. A well accepted compromise is to shorten the axis range in a way that leaves out points which are less important for the interpretation of the graph. The features that are left out are then indicated in the visible part of the graph.

Clipping extreme values has been found especially important when creating boxplots over many subgroups. In SAS® 9.2 boxplots can be produced either via Proc GPLOT, Proc BOXPLOT or via one of the newly introduced Statistical Graphics (SG) procedures. There are substantial differences in the way these procedures handle clipping of extreme values, as will be demonstrated.

INTRODUCTION

The range of values depicted in a graphic is easily specified via the order option in the axis statement. Its result is not always predictable, unfortunately. If the specified axis range is too short, so that graphic features are left out, the resulting figure may be created 1) under ignorance of the data points left out; 2) under ignorance of the specified axis range; 3) with correctly specified axis range but the graphic will reach outside of the axis area; or 4) as a clipped version of the figure.

Figure 1: Boxplot produced with Proc GPLOT on either full range (left) or under limited range (right) of the vertical axis.



In Figure 1, four boxplots are depicted which are produced with Proc GPLOT and the statement

```
symbol I=box10;
```

The right plot has been produced under the same conditions with additional option under vaxis:

```
order=(0 to 6 by 1)
```

As a result the boxplots are not just cut at the height of 6, but all data points greater than 6 (plus offset!) were ignored in the production of the graph and a message written to the log saying:

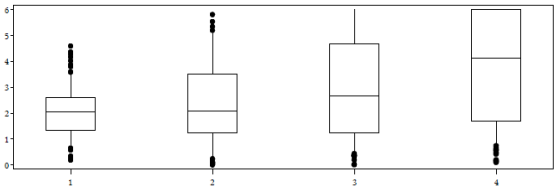
```
NOTE: 56 observation(s) outside the axis range for the y * x request.
WARNING: Values exist outside the axis range, but only values within the displayed range took
part in interpolation calculations for the PLOT statement.
```

If the figure to the right is the desired output, I simply would have used a where statement to discard y-values greater than 6. Why is this feature hidden in an axis statement? Shouldn't an axis statement affect only the way something is presented, not the way something is calculated? Why is SAS doing this to me and what can I do about it?

CLIPPING EXTREMES UNDER PROC GPLOT

By default the axis statement is processed with the options `mode=exclude` which means that all data points not included in the visual area are not considered in any calculation. Setting `mode=include` for the same data as for Figure 1 produces the desired plot.

Figure 2: Boxplot from Figure 1, on a restricted vertical axis range and axis statement option `mode=include`.



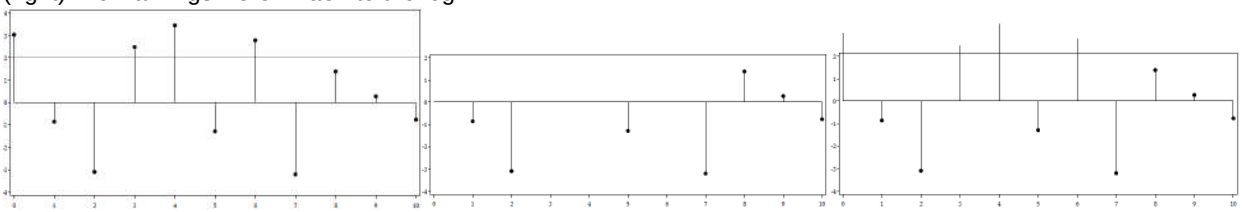
The boxplot produced in Figure 2 produces no warning but a note in the log:

```
NOTE: 56 observation(s) outside the axis range for the y * x request.
```

There is no indication of clipping in the figure itself. The picture could likely lead to the false conclusion of no clipping.

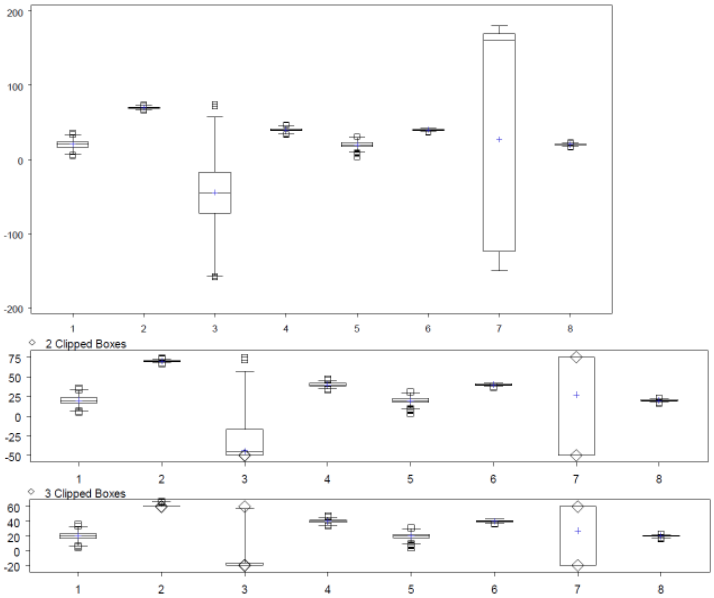
Based on the interpolation method selected in Proc GPLOT the result after clipping may vary. For most interpolation methods a warning and a note is written to the log in case `mode=exclude` (the default) is applied and no warning but a note when `mode=include` is set. This holds true for `Interpol=Step`, `BOX`, `HILO`, `STD`, `R`, `L`, `SM` and `SPLINE`. The same logic is also applied when producing a bubble plot under Proc GPLOT. For `Interpol=JOIN` no warning is produced under the default mode setting, although data is left out. The same holds true for `Interpol=Needle`. If needle or a map/plot-pattern interpolation is chosen together with `mode=include`, then the axis range is correctly displayed but not fully respected in the plot itself, as depicted in Figure 3.

Figure 3: Proc GPLOT with `Interpol=Needle` (left) and option `mode=exclude` (middle) or `mode=include` (right). No warnings were written to the log.



The outcome when annotated objects are not fully enclosed in the visible are is highly unpredictable. Michael Friendly provides a macro (www.math.yorku.ca/SCS/sssg/boxplot.html) that creates boxplots based on the annotate facility. The macro also provides an option to specify the axis range, but any specification that does not enclose the entire data range leads to unwanted features.

Figure 4: Boxplot produced with Proc BOXPLOT and no clipping (top), clipfactor=2 (middle) and clipfactor=1.4 (bottom).



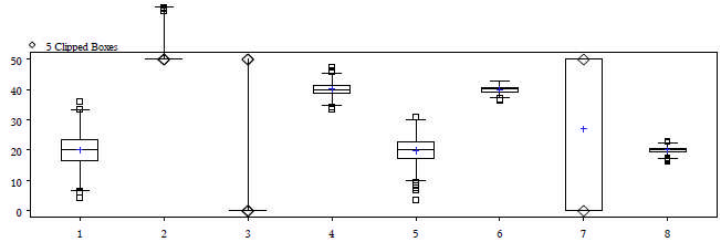
CLIPPING EXTREMES UNDER PROC BOXPLOT

In Proc BOXPLOT, if an axis-range that does not cover the entire range of data values is specified in an axis statement, it is simply ignored. Instead, Proc BOXPLOT provides extra options to clip extreme values.

The range of values to be covered in proc BOXPLOT is defined in the `clipfactor=`*`factor`* option, which takes values greater than 1 as argument. Clipfactor first calculates the mean of the first and third quartile over all groups, here called *Q1* and *Q3*. It then takes the difference *R=Q3-Q1*. The upper limit is finally defined as *Q1+R×factor* and the lower range as *Q3–Q1+R×factor.* The minimum that can be specified this way – for `clipfactor=1` – ranges from *Q1* to *Q3*. Since *Q1* and *Q3* are means over many quartiles, any single boxplot can still be cut at any place, following this procedure.

Any clipping is nicely indicated by an extra symbol at the place where the graph is cut. The amount of clipping is further documented by a note in the log. Figure 4 is produced via the device driver in `GOPTIONS`. If the graph is written to a file via ODS it looks substantially different. Besides the disturbing fact that the figure reaches out of the axis frame, the axis range is also different to the non-ODS version (for exactly the same code, besides the ODS part) as can be seen in Figure 5. To be more specific, Figure 4 has been produced with the pdf device defined under `GOPTIONS` and Figure 5 with the `ODS PDF` statement. Other ODS destinations (e.g. rtf) lead to exactly the same confusion.

Figure 5: Boxplot produced with Proc BOXPLOT and clipfactor=1.4. Output produced via ODS.



Proc BOXPLOT in combination with the clipfactor option has several drawbacks:

1) The axis-value at which to cut cannot be determined directly, but only via clipping factor. E.g. if you want to cut at an upper value of y_1, you first have to calculate the clipping factor as $(y_1 - Q1)/R$.
2) Only one clipping factor can be specified, covering the lower and upper axis cut, simultaneously.
3) Unpredictable result when used together with ODS.

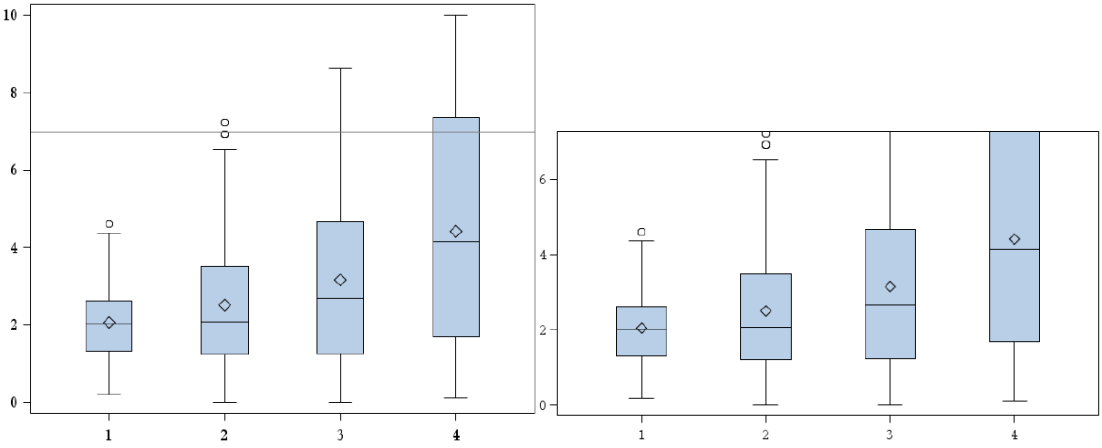
CLIPPING EXTREMES UNDER PROC SGPLOT

A boxplot can be produced in Proc SGPLOT using

```
proc sgplot;
 vbox y / category=x;
 yaxis max=7;
run;
```

The line `yaxis max=7` restricts the vertical axis to a maximum value of 7. As can be seen in Figure 6, the visual area of the graphic is cut, but the boxplots not affected, otherwise.

Figure 6: Boxplot produced with Proc SGPLOT without axis restriction (left) and using the statement `yaxis max=7` (right).



There is a small offset at the top of the graph which makes the top circle in boxplot-category 4 to appear. This offset can be set to zero with the statement `OFFSETMAX=0` in an axis statement, but only in versions SAS 9.2 Phase II (TS2M0) and later.

SOLUTION TO CLIPPING EXTREMES UNDER PROC (S)GPLOT

Figure 7: Boxplot with extreme outliers.

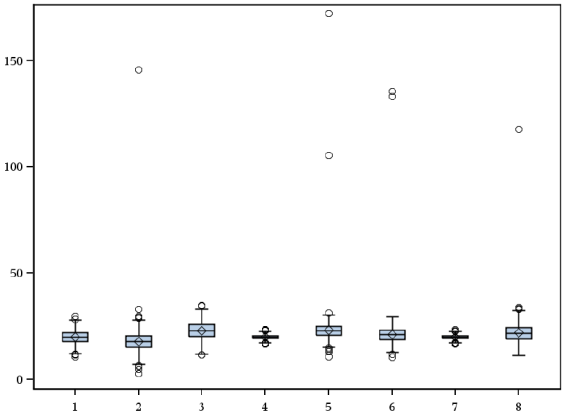


Figure 7 shows boxplots with far outliers, produced with Proc SGPLOT. The interesting features of the plot are visually not very distinct and what is less interesting (the few outliers) attract too much spotlight. As a solution we leave the lower limit of the vertical axis at 0 and bring the upper limit down.

Before restricting the vertical axis we need to determine the maximum value covered by a box. We do not want to cut the whiskers, but the outliers. The whiskers of a boxplots are defined to go from the quartile to the maximum outcome value that is smaller than the fence. The fence being defined as 1.5 times the interquartile range (the distance between quartiles). This is the classical definition for constructing boxplots. SAS provides alternative definitions. `Interpol=box` in the symbol statement supports fence constructed by percentiles of the data. For instance `Interpol=box10` draws whiskers down/up to the 5% and 95% percentile, and `Interpol=box00` up to the extremes.

First we calculate the fence via interquartile range and quartiles (here only upper quartile is needed, since we intend not to shift the axis origin of 0).

```
proc univariate data=phusetalk noprint;
 class x;
 var y;
 output out=IQR qrange=qrange q3=q3;
run;
```

From the range and upper quartile we calculate the minimum at which we are allowed to cut the axis, without hurting the whiskers.

First determine the fence, then the maximum:

```
proc sql noprint;
 select max(q3+1.5*qrange) into: fence
 from IQR;
quit;
Data _Null_;
 retain max -1E10;
 merge phusetalk IQR end=last;
 by x;
 if y le &fence and y gt max then max=y;
 if last then call symput('cutpoint', 10*ceil(max/10));
run;
```

We output 10*ceil(max/10) instead of just max, as we want to have the axis limit at a multiple of 10, rather than such an obscure value as 33.87.

As a next step we add an extra line per group if a value is beyond the axis limit.

```
Data phusetalk2;
 retain index;
 merge phusetalk IQR;
 by x;
```

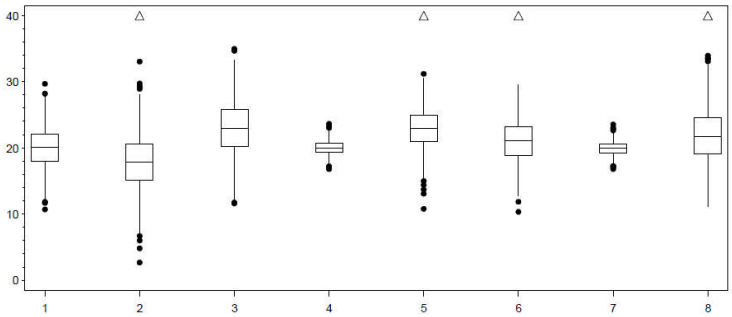
```
 if first.x then index=0;
 if y>&cutpoint then index=1;
 if last.x and index=1 then do;
   mark=0;
   output;
   y=&cutpoint;
   mark=1;
   output;
 end;
 else do;
   mark=0;
   output;
 end;
run;
```

The extra lines are marked with mark=1, the original data with mark=0. The above code could be arranged much simpler by just duplicating every value which is beyond the limit (once at the original value with mark=0 and once at the limit with mark=1). Then we would possibly end up with multiple identical, overlying points at the upper limit, which would not hurt the graph, as these points would not be visually distinct from one single point. This approach is used later in this section when the outliers are labeled.

The plot is finally produced for the two groups of the mark variable with distinct interpolation options:

```
symbol1 I=box  v=dot mode=include;
symbol2 I=none v=triangle;
axis1 order=(0 to &cutpoint by 10);
proc gplot data=phusetalk2;
  plot y*x=mark / vaxis=axis1;
run; quit;
```

Figure 8: Boxplot with restricted vertical axis and extra symbol for outliers.



In Figure 8 the presence of values not shown is indicated by a triangle to give the impression of pointing-up. Unfortunately no inverted triangle is available among the SAS symbols to do the same for pointing downwards, if the axis is cut below the boxes. The value for “f” in the font math does look similar, however and might be a good compromise (▽). It needs to be slightly shrunk compared to the triangle (via the height option) and it has not all sides of equal length (as triangle does), but it will do the job. Specify: `symbol2 interpol=none v=l f=math height=0.3;`

As a next step, let us put some labels on the marks to indicate, how many, and where the outliers are.

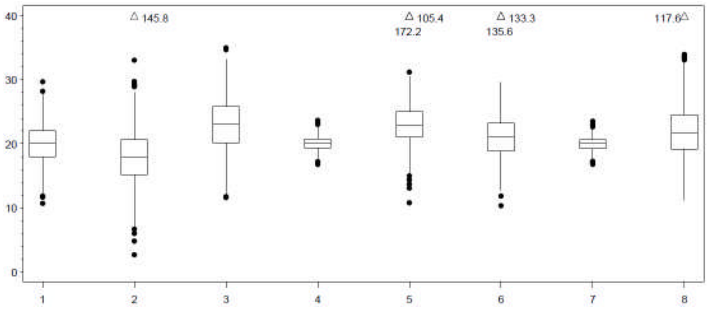
```
Data phusetalk2;
 format label $6.;
 retain index mark 0;
 merge phusetalk IQR;
 by x;
 if y>&cutpoint then do;
   mark=0;
   output;
   label=put(y,5.1);
   y=&cutpoint;
   mark=1;
   output;
 end;
 else do;
   mark=0;
```

```
    output;
  end;
run;
```

The length of the label variable is chosen longer than actually needed. This way the labels are more distinctive when plotted.

```
proc sort data=phusetalk2;
  by x mark descending label;
run;
symbol1 I=box v=dot mode=include;
symbol2 I=none v=triangle
        pointlabel=(justify=right position=Middle nodropcollisions "#label");
axis1 order=(0 to &cutpoint by 10);
proc gplot data=phusetalk2;
plot y*x=mark / vaxis=axis1;
run; quit;
```

Figure 9: Boxplot with vertical axis cut and labels for extreme outliers.



The labeling does not work well in case of many outliers. As the possibilities to place them are limited, labels via the annotate facility might be the preferred option.

The vbox and hbox statements in SGPLOT cannot be combined with any other plot type (e.g. scatter). As a consequence, we can draw boxplots and clip them but have no means to indicate the outliers, as this has been demonstrated with Proc GPLOT above. Also is the annotate facility not compatible with the statistical graphics introduced in version 9.2 of SAS.

QUARTILE DEFINITIONS

We must be cautious in calculation the quartiles. There is no single definition of a quartile. In SAS, we can choose from five options how quartiles are calculated. This is done via the option `PERCENTILE` in Procs SGPLOT and BOXPLOT, or via the option `PCTLDEF` in Proc UNIVARIATE and via the option `QNTLDEF` (in combination with option `QMETHOD`) in Proc SUMMARY/MEANS. These options let you choose among one of 5 percentile definitions. Fortunately, the default is all the same among these procedures (in Proc SUMMARY/ MEANS only in combination with `QMETHOD=OS`).

As an illustration think of n=15 values and calculate the upper quartile (p=0.75). By definition it is the p*n ordered value. In this case the 11.25^{th} value. Since that value does not exist (only full numbers are allowed for the order statistics), we have to come up with a decision. Let's say we order the n value $x_{(1)}$, $x_{(2)}$, …, $x_{(n)}$ and ideally select the value $x_{(np)}$. The possible options in SAS are:

1: Weighted average of adjacent values: $x_{(j)}$ and $x_{(j+1)}$, where $j \le np \le j+1$ (linear interpolation).
2: Closest value: $x_{(j)}$ so that j is closer to np than to any other index.
3: Closest value up: $x_{(j)}$ with j being the smallest integer $j \ge np$.
4: Weighted average at n+1: Calculate (n+1)*p (instead of n*p) and take the weighted average of adjacent values as in 1 ($x_{(j)}$ and $x_{(j+1)}$, where $j \le (n+1)p \le j+1$).
5: Combination of 1 and 3: Take the average between $x_{(j)}$ and $x_{(j+1)}$ if j=np; if $j<np<j+1$ then take $x_{(j+1)}$ as the quartile.

Proc RANK also offers some options for percentiles definitions. These definitions are distinct from the ones offered above. The goal of Proc RANK is to separate the data into groups. Percentiles which lie between values cannot be determined with Proc RANK and it should not be used to calculate percentiles.

Unfortunately, Proc GPLOT offers no way to choose the definition of percentiles. There is also no documentation on which definition is implemented. Some comparisons revealed that definition number 5 is implemented (which is the

default in all other procedures).
Table 1 illustrates the five percentile definitions. The upper quartile is computed in the dataset 1,2,3,4 and 1,2,3,4,5,6.

Table 1: Calculated upper quartile for two dataset and the five percentile definitions.

	Percentile definition				
Dataset	1	2	3	4	5
1,2,3,4	3	3	3	3.75	3.5
1,2,3,4,5,6	4.5	4	5	5.25	5

DISCUSSION

An alternative to clipping extremes would be to rescale the vertical axis (e.g. log-scale). This works well only if there are no extremes on both ends of the scale and if the outliers are still meaningful enough to show them. In many clinical applications patients with measurements far beyond a certain cut-off tend not to show clinical symptoms different from those patients with measurement at the cut-off. We are not more interested in the actual value than in the information that a measurement is beyond a limit. Clipping the outliers maintains the actual scale and supports the ease of interpreting the results as the interesting part of the graph is no longer distorted.

CONCLUSION

The look of clipped graphics is not easy to predict. Standard SAS features to clip boxplots are difficult to handle and not reliable when used in connection with ODS. It is suggested to clip the data beforehand and then run the graphical procedure on the clipped outcome, with two layers, one showing the clipped graph, one indicating the outliers. This is best handled in Proc GPLOT. Proc GPLOT handles different graphical layers easily and is compatible with the annotate facility, what is currently not supported by the new statistical graphics procedures.

CONTACT INFORMATION

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

Armin Gemperli
BIOP AG
Centralbahnstrasse 29
CH-4051 Basel
Switzerland
Armin.Gemperli@biop.ch