Plotting Censored Data

Dave Lorenz

May 20, 2015

plot.qw dotPlot boxPlot ecdfPlot probPlot qqPlot timePlot xyPlot?

These examples demonstrate variations of types of box plots that can be generated using the boxPlot function in the smwrGraphs package. All of the examples use randomly generated sets of data. **NOTE:** to use the boxPlot function, you must first call a function to set up the graphics environment like `setPage` or `setPDF`, but these are not included here to use the graphics tools in `Sweave`.

```
> # Load the smwrGraphs package
> library(smwrGraphs)
> # Generate a random sample for the box plot
> set.seed(27036)
> BP <- rchisq(32, 3)
> # Generate a small random sample
> bp <- rchisq(4, 3)
> # Create grouping variables
> Gchar <- rep(c("A", "B"), 16)
> Gnum <- rep(c(1998, 2002), 16)
```

1 Box Plot Types

The truncated box plot is the default type with truncation at the 10 and 90 percentiles. The other types can be created with a simple revision to the `Box` argument. The simple box plot extends the whiskers to the minimum and maximum values. For the Tukey box plot, `type="tukey"`, the whiskers are extended to the observed value that is within 1.5 times the interquartile range above or below the upper or lower quartile and observed values outside of that range are plotted as individual symbols. The extended box plot is a variation on the simple box plot where all observed values outside of the upper and lower percentiles are plotted as individual symbols. For the default 10 and 90 percentile limits, no more than 10 percent of the total number of observed value will be individually plotted. Note that the y-axis range is set by the range of the data and not the range of the box plot.

```
> # setSweave is a specialized function that sets up the graphics page for
> # Sweave scripts. It should be replaced by a call to setPage or setPDF
> # in a regular script.
> setSweave("boxplot01", 6 ,6)
> # Set layout for 4 graphs
> AA.lo <- setLayout(width=rep(1.25, 4), height=4, xtop=1.5)
> # Only need to create the margins once in this case
> AA.gr <- setGraph(1, AA.lo)
> boxPlot(BP, margin=AA.gr)
> addTitle("Truncated")
> setGraph(2, AA.lo)
> boxPlot(BP, Box=list(type="simple"), margin=AA.gr)
> addTitle("Simple")
> setGraph(3, AA.lo)
> boxPlot(BP, Box=list(type="tukey"), margin=AA.gr)
> addTitle("Tukey")
> setGraph(4, AA.lo)
> boxPlot(BP, Box=list(type="extended"), margin=AA.gr)
> addTitle("Extended")
> # Required call to close PDF output graphics
> graphics.off()
```

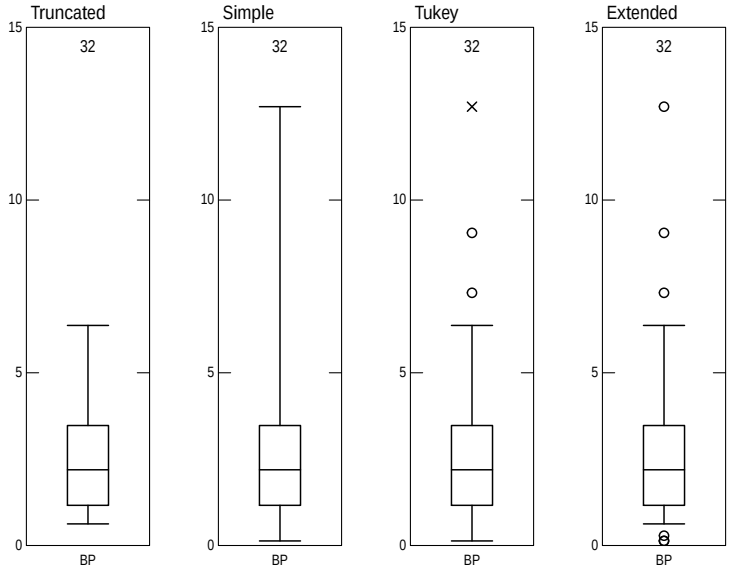

Figure 1. The four basic types of box plots.

2 Example with Explanation

The explanation for the box plot is best created to the right of the graph rather than below. This example shows the explanation for the truncated box plot. The explaantion for the Tukey box plot requires a widht of about 2.5 inches, rather than the 1.5 inches in the example that follows.

```
> setSweave("boxplot02", 6 ,6)
> # Set layout for 1 graph and an explanation
> AA.lo <- setLayout(width=1.5, height=4, explanation=list(right=1.5))
> # Only need to create the margins once in this case
> AA.gr <- setGraph(1, AA.lo)
> AA.bp <- boxPlot(BP, margin=AA.gr)
> setGraph("explanation", AA.lo)
> addExplanation(AA.bp, title="Truncated Boxplot")
> graphics.off()
```

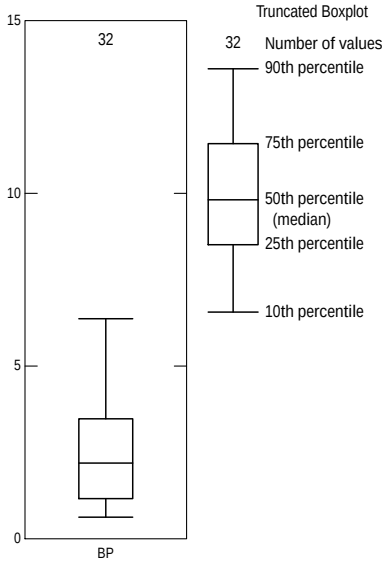

Figure 2. Box plot with explanation.

3 Other variations in the Box argument

The Box argument includes the show.counts components, which is a logical value (`TRUE` or `FALSE`) that controls whethere the number of observations are shown above the box; censorbox and censortype components, which are valid only for specialized data and require specific methods; the nobox component, which sets the upper limit for the number of observations where individual values are plotted rather than a box; width, which is a single numeric value that specifies the width of the box in inches; fill, which specifes the color of the box; and truncated, which specify the limits for the truncated and extended types of boxplot.

This example demonstrates how to create a boxplot with a filled box, change the width, and demonstrates the individuals plotted for a small sample size. It also shows how to change the default x-axis labels. The user should verify the order before changing `xlables`.

```
> setSweave("boxplot03", 6 ,5)
> # The figure is set to a heigth of 5 inches to fit on the page.
> # The color gray80 is a very light gray and works well for the fill
> boxPlot(BP, bp, Box=list(fill="gray80", width=1.0), xlabels=c("Big", "Small"))
> graphics.off()
```

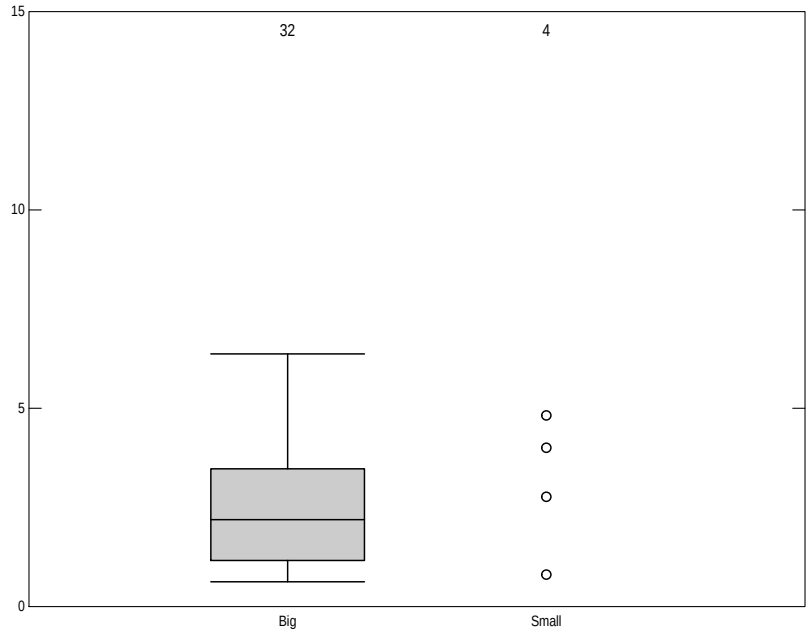

Figure 3. Box plot variations.

4 Grouped Box Plots

The Box argument includes the `group` argument, which can be used to split a single variable into multiple variables corresponding to each unique value in the group. This example demonstrates how to create a grouped boxplot. The user should verify the order before changing `xlables`.

```
> setSweave("boxplot04", 6 ,6)
> # Accept default graph size for this example
> boxPlot(BP, group=Gchar)
> graphics.off()
```

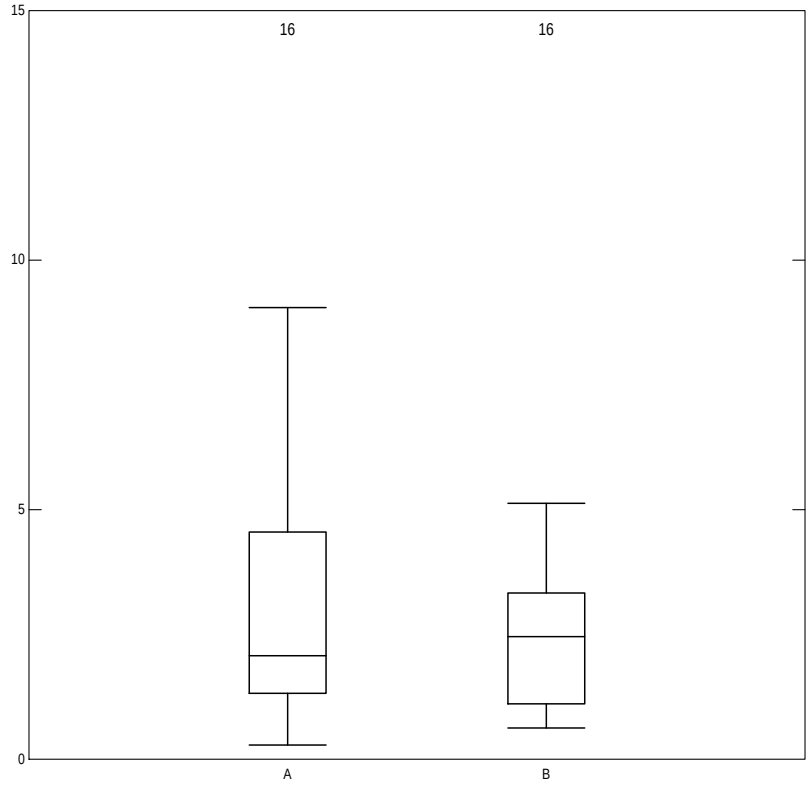

Figure 4. A grouped box plot.

5 Numeric Grouped Box Plots

If the `group` argument is numeric, then the x-axis set up along a continuous scale. This example was selected to demonstrate a quirk in the logic of numeric groups—the values may represent discontinuous years, for example. The numeric column can be forced to be treated as a normal grouping variable by converting it to type factor.

```
> setSweave("boxplot05", 6 ,6)
> # Accept default graph size for this example
> boxPlot(BP, group=Gnum)
> graphics.off()
```

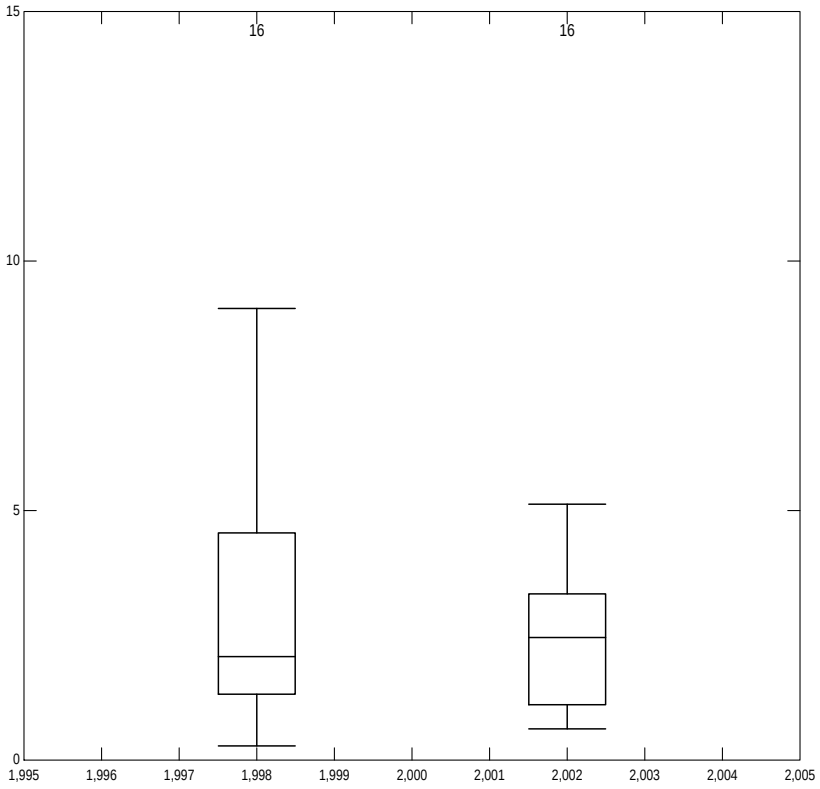

Figure 5. A box plot grouped by a numeric variable.