Graph Additions

Dave Lorenz

October 2, 2015

Abstract

These examples demonstrate how to add features to an existing graph. Adding a plot or a title to an existing graph is discussed in the GraphSetup vignette. These general procedures apply to most high-level graphics functions within the smwrGraphs package. All of the examples use randomly generated sets of data. **NOTE:** to use any high-level graphics function in the smwrGraphs package, 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`.

Contents

1 Introduction

All of the examples use randomly generated sets of data for simple line or scatter plots. The data are generated in the following code

```
> # Load the smwrGraphs package
> library(smwrGraphs)
> # Generate the random data
> set.seed(3636)
> X <- rnorm(32)
> Y <- X + rnorm(32)
```

2 Reference Line with Annotation

This example draws a simple scatter plot, then adds a line representing the median y value and annotates that line. Adding annotation generally requires a trail and error approach to placement of the annotation. The simple method used in this example works because X and Y are correlated.

```
> # Set up the graphics environment, the equivalent call for an on screen
> #  device would be setPage("square")
> setSweave("graph01", 6 ,6)
> #
> AA.pl <- xyPlot(X, Y)
> # Add the median line of Y and annotation
> refLine(horizontal = median(Y), current=AA.pl)
> addAnnotation(min(X), median(Y), "Median Y", current=AA.pl)
> graphics.off()
```

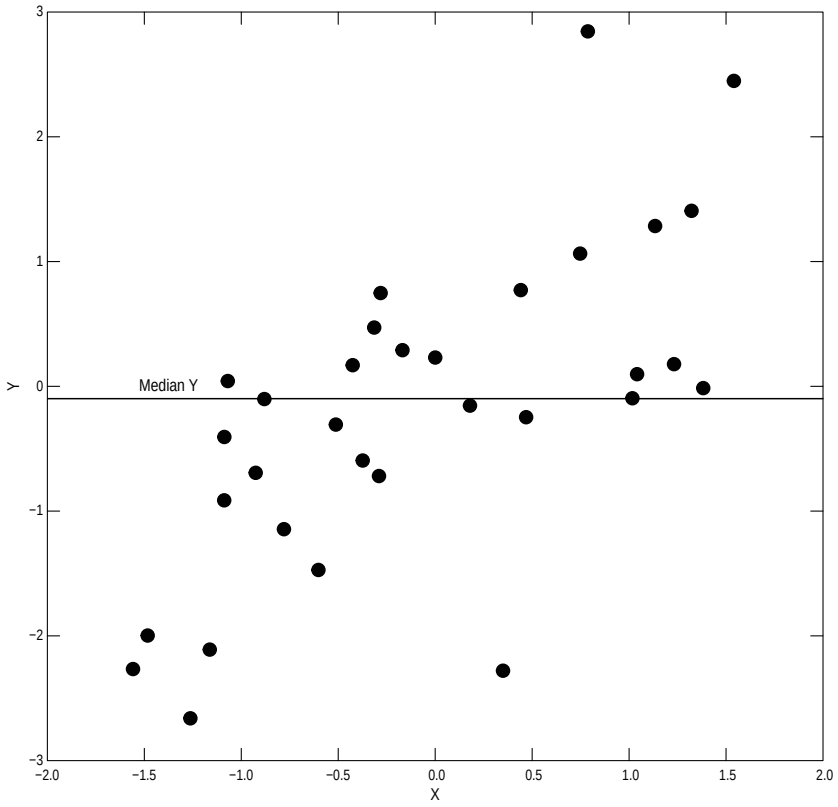

Figure 1. Scatter plot with reference line and annotation.

3 Grid Lines

Adding grid lines is a 3-step process—1 create the graph with nothing plotted, 2 add the grid lines, and 3 then add the plotted data. This process guarantees that the data will over plot the grid lines.

```
> # Set up the graphics environment, the equivalent call for an on screen
> #  device would be setPage("square")
> setSweave("graph02", 6 ,6)
> # Step 1
> AA.pl <- xyPlot(X, Y, Plot=list(what="none"))
> # Step 2
> addGrid(AA.pl)
> # Step 3
> AA.pl <- addXY(X, Y, Plot=list(what="points"))
> # Required call to close PDF output graphics
> graphics.off()
```

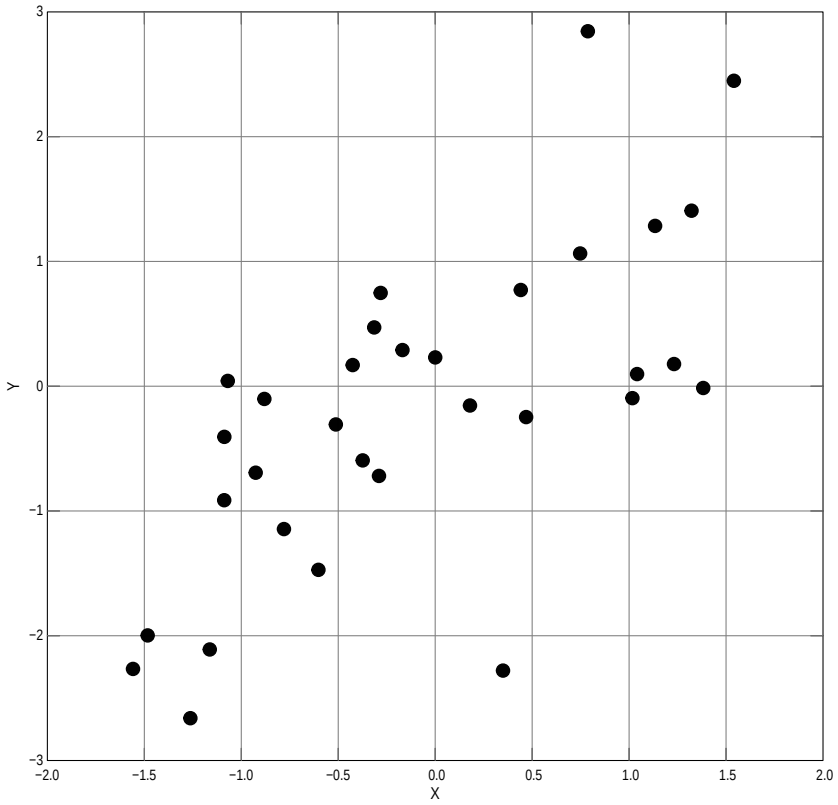

Figure 2. Scatter plot with grid lines.

4 Add a Smoothed Line

It is sometimes desirable to add a smoothed line to a scatter plot to show the general trend or relation between the data.

```
> # Set up the graphics environment, the equivalent call for an on screen
> #  device would be setPage("square")
> setSweave("graph03", 6 ,6)
> # Create a scatter plot from the X and Y data. The name of the output (AA.pl)
> #  is completely arbiutrary, but consistently used through these examples.
> AA.pl <- xyPlot(X, Y)
> # The addSmooth function will compute the smmothed line and add the plot to the
> # graph. Accept all defaults for this example. A very useful additional
> # argument would be span for the loess.smooth
> AA.pl <- addSmooth(X, Y, current=AA.pl)
> # Required call to close PDF output graphics
> graphics.off()
```

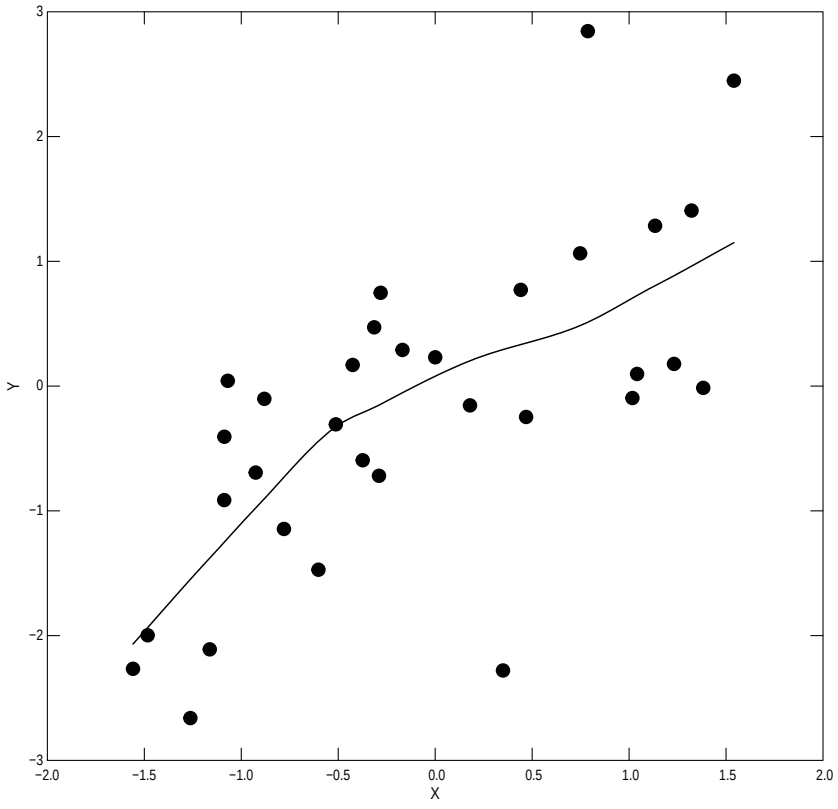

Figure 3. Scatter plot with smooth line.

5 Add a Regression Line with Confidence Intervals and Table

Sometimes a table needs to be added to a graph. That table may represent some statistical summary or other additional data.

```
> # Set up the graphics environment, the equivalent call for an on screen
> #  device would be setPage(layout=list(width=6, height=4)).
> setSweave("graph04", 6 ,4)
> # Create a scatter plot from the X and Y data.
> AA.pl <- xyPlot(X, Y, Plot=list(what="points", color="black"))
> # Create and add the regresion line and 95% confidence intervals
> AA.pl <- addSLR(AA.pl)
> # The output is discarded in this case becuase an explanation is not created
> addCI("SLR", current=AA.pl)
> # Create the table and add it to the graph
> # Note may actually want to reformat the last p-value so not 0
> AA.tbl <- format(round(coefficients(summary(AA.pl$lm)), 4))
> AA.tbl <- cbind(" "=c("Intercept", "X"), AA.tbl)
> addTable(AA.tbl, where="ul")
> # Required call to close PDF output graphics
> graphics.off()
```

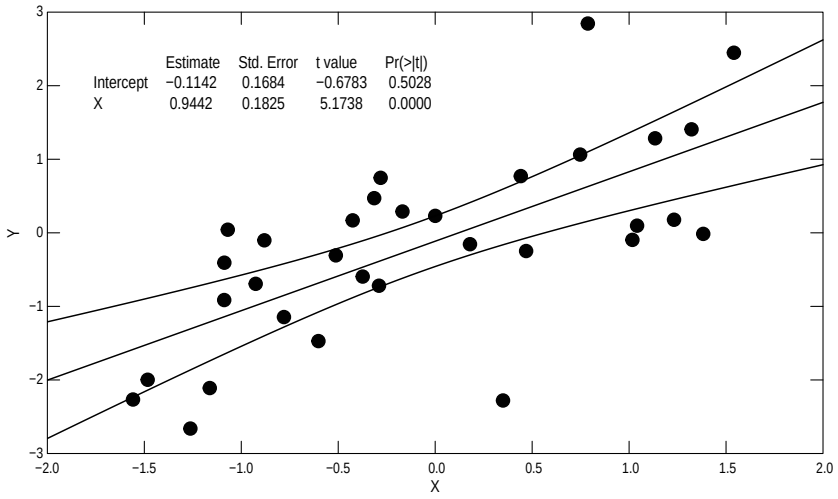

Figure 4. Scatter plot with simple linear regression.

6 Show area covered by data

Occasionally there is a need to show the general area covered by a set of data. The `smwrGraphs` package has several functions that help the user draw areas covered by the data: `dataEllipse`, `cov2Ellipse`, and `hull` functions will provide closed polygons that represent some measure of the area covered by data. This example demonstrates `dataEllipse` and `hull`.

This example also identifies and labels selected points lying outside of the enclosing polygon. It demonstrates the use of the `lablePoints` function.

```
> # Set up the graphics environment, the equivalent call for an on screen
> #  device would be setPage(layout=list(width=6, height=4)).
> setSweave("graph05", 6, 3.5)
> # Create a scatter plot from the X and Y data.
> AA.pl <- xyPlot(X, Y, Plot=list(what="points", color="black"))
> # Create and draw an ellipse that covers 90 percent of the data
> AA.el <- dataEllipse(X, Y, percent=90)
> with(AA.el, addXY(x, y, Plot=list(what="lines", color="darkred"), current=AA.pl))
> # Now do the same with a smooth hull
> AA.hl <- hull(X, Y, percent=90, smooth=TRUE)
> with(AA.hl, addXY(x, y, Plot=list(what="lines", color="magenta"), current=AA.pl))
> # Now find the distance from the center of the ellipse and which are greater than
> # the 90th percentile
> AA.ds <- mahalanobis(cbind(X,Y), c(mean(X), mean(Y)), var(cbind(X,Y)))
> AA.sel <- which(AA.ds > quantile(AA.ds, probs=0.9, type=2))
> # Add the labels--the sequence number of the point
> labelPoints(X[AA.sel], Y[AA.sel], as.character(AA.sel), current=AA.pl)
> # Required call to close PDF output graphics
> graphics.off()
```

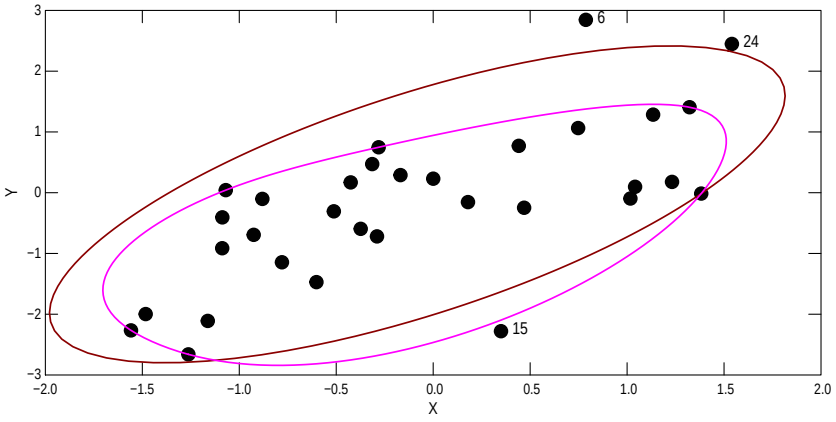

Figure 5. Scatter plot showing area covered by the data.