Graph Set Up

Dave Lorenz

March 16, 2015

These examples demonstrate how to set up a complete figure beginning with a simple scatter plot. The general procedures apply to any other 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`.

```
> # Load the smwrGraphs package
> library(smwrGraphs)
> # Generate the random data
> set.seed(27036)
> X <- rnorm(32)
> Y <- X + rnorm(32)
> Z <- rnorm(32, sd=1.2)
> Zfill <- runif(32, -2, 0)
```

Some basic nomenclature is necessary to understand the instructions in this vignette.

figure The completed product, consisting of one or more graphs and an optional explanation.

graph The area containing the plot, axis labels and titles and graph title.

plot The data plotted as lines, points, or other.

axis labels Labels for each major tick.

axis title A brief description of the axis.

graph title A description or identifier for the graph.

explanation An explanation of the symbols in the graphs, also called legend or key.

1 A Simple Graph

The simplest graph can be created using only two functions, one to set up the graphics environment and one to create the graph. An additional call would be required to close the graph if the output were a PDF file created by `setPDF` or `setSweave`.

```
> # Set up the graphics environment, the equivalent call for an on screen
> #  device would be setPage("square")
> setSweave("graph01", 6 ,6)
> #
> xyPlot(X, Y)
> # Required call to close PDF output graphics
> graphics.off()
```

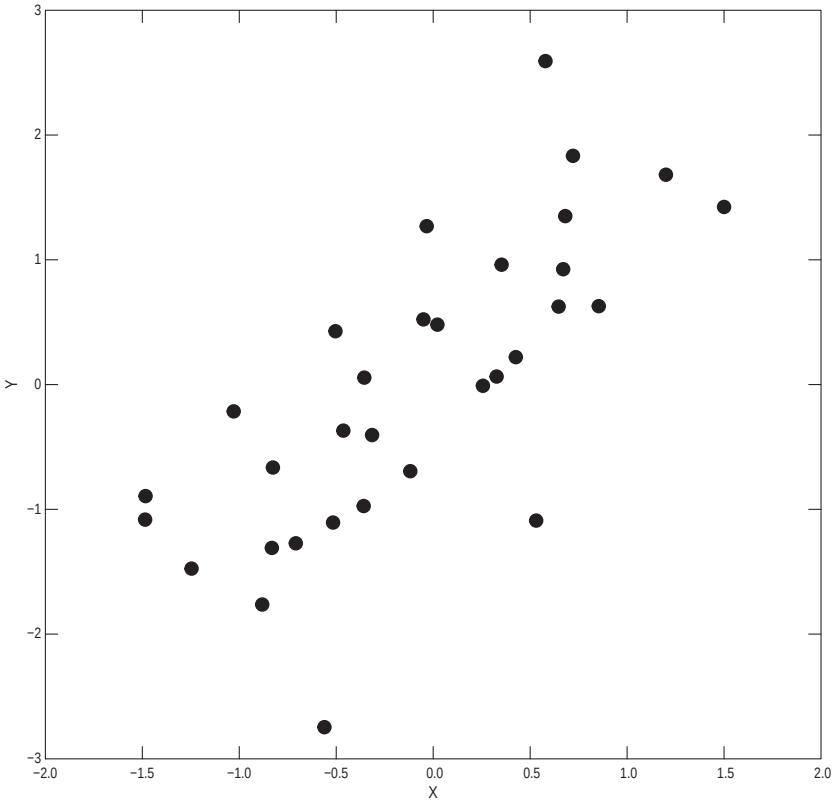

Figure 1. A simple x-y scatter plot.

2 Plot Customization

The default plot created by `xyPlot` is a simple x-y scatter plot. The axes are scaled to a range that fits the data and labeled according accepted practices within the USGS The x- and y-axis titles are derived from the names of the `x` and `y` arguments. Each of those options are controlled by a simple argument to `xyPlot`. The plot is filled black circles. The `Plot` argument is a list that controls what is actually plotted, any individual component of that list can be omitted and the default value will be used. This example shows several options. Each line is commented to describe the desired change. they would not necessarily need to be commented in any call.

```
> # Set up the graphics environment, the equivalent call for an on screen
> #  device would be setPage("square")
> setSweave("graph02", 6 ,6)
> #
> xyPlot(X, Y,
+ # Change from solod black circles to blue plus signs
+ Plot=list(symbol="+", color="blue"),
+ # Set the x-axis range to -2 to 2, for symmetry
+ xaxis.range=c(-2,2),
+ # label at the five integral values: -2, -1, 0, 1, 2
+ xlabels=5,
+ # can also be manually set: see figure 3
+ # Change the x- and y-axis titles and end the call
+ xtitle="Random Data",
+ ytitle="Correlated Data")
> # Required call to close PDF output graphics
> graphics.off()
```

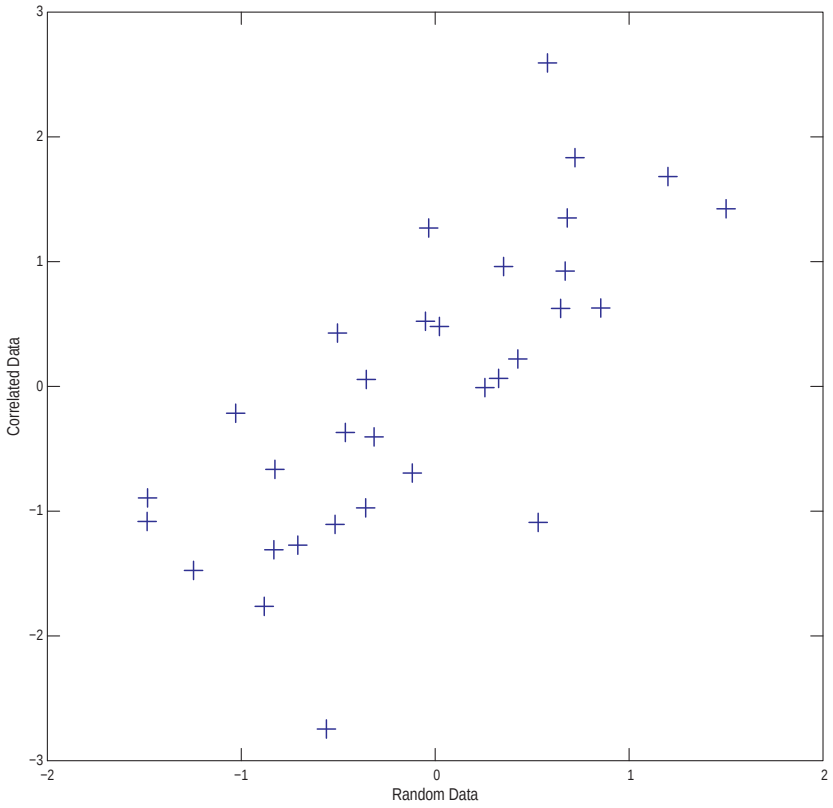

Figure 2. Scatter plot variations.

3 Axis Customization

The previous section showed some simple ways to modify axis labels. The user has a great deal of control over axis labels by using the arguments to `linearPretty` for linear axes, `logPretty` for log axes, `datePretty` for date axes, and `probPretty` for probability axes. For all label arguments as list, the `labels` argument must be supplied.

```
> # Set up the graphics environment, the equivalent call for an on screen
> #  device would be setPage("square")
> setSweave("graph03", 6 ,6)
> #
> bigY <- exp(Y)
> xyPlot(X, bigY,
+ # Change from solod black circles to blue plus signs
+ Plot=list(symbol="uptri", color="green"),
+ # Set the x-axis range to -2 to 2, for symmetry
+ xaxis.range=c(-2,2),
+ # label at the five integral values: -2, -1, 0, 1, 2
+ xlabels=c(-2, -1, 0, 1, 2),
+ yaxis.log=TRUE,
+ # The default behaviour would be numeric labels for this range.
+ ylabels=list(labels="Auto", style="scientific"),
+ # Change the x- and y-axis titles and end the call
+ xtitle="Random Data",
+ ytitle="Correlated Data")
> # Required call to close PDF output graphics
> graphics.off()
```

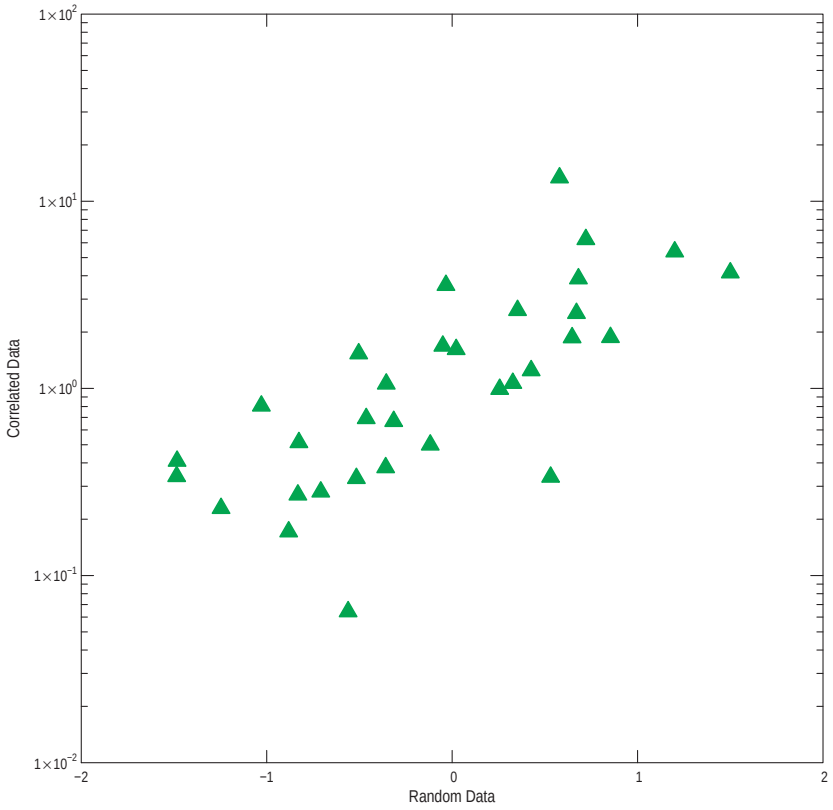

Figure 3. Scatter plot label variations.

4 Adding a plot to a graph

All of the high-level plotting functions in the smwrGraphs package return much information about the graph and the plot. This information can be used to add a plot to an existing graph or add an explanation. The information returned from a high-level plotting function is passed to other functions that add to the graph as the `current` argument.

```
> # Set up the graphics environment, the equivalent call for an on screen
> #  device would be setPage("square")
> setSweave("graph04", 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, Plot=list(name="Correlated Data", color="blue"),
+ xaxis.range=c(-2,2), xlabels=5,
+ xtitle="Random Data",
+ ytitle="Response Data")
> # Use the addXY function to add a plot to the graph. The output contains
> #  information about both plots and cen be used to create an explanation
> #  or legend.
> AA.pl <- addXY(X, Z, Plot=list(name="Uncorrelated Data", what="points",
+   color="darkred"), current=AA.pl)
> # The addExplanation function processes the information in the output to
> #  create an explanaiton of the data shown in the plots.
> addExplanation(AA.pl, where="ul", title="")
> # Required call to close PDF output graphics
> graphics.off()
```

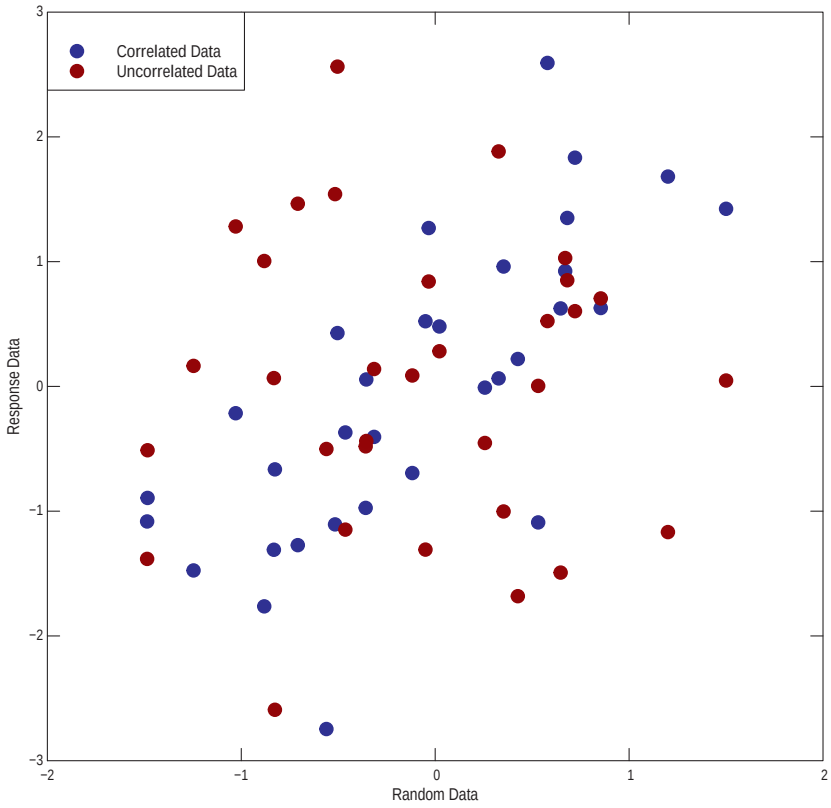

Figure 4. Example of multiple plots in a graph.

5 Multiple Graphs

Sometimes, complete figures need multiple graphs instead of multiple plots within a graph. In that case, or when the graph area needs to be changed from the default, two additional functions (`setLayout` and `setGraph`) are needed. The `setLayout` function can be used to set up a grid of graphs and the `setGraph` functions sets up each individual graph. The graphs set up by `setLayout` can share axes or stand alone.

```
> # 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 ,4)
> # Set the layout for 2 graphs in one row. and allocate room at the top for
> #  a graph title
> AA.lo <- setLayout(num.cols=2)
> # The first graph is the left-most graph
> AA.gr <- setGraph(1, AA.lo)
> # Create a scatter plot from the X and Y data.
> AA.pl <- xyPlot(X, Y, Plot=list(color="blue"),
+   xaxis.range=c(-2,2), xlabels=5,
+   xtitle="Random Data", ytitle="Correlated Data",
+   margin=AA.gr)
> # Add the title
> addTitle("A")
> # The figure caption should always by the lower-left most graph
> addCaption("Figure 5. Example Graphs.")
> # Subsequent graphs are placed to the right in each row
> AA.gr <- setGraph(2, AA.lo)
> # Create a scatter plot from the X and Y data.
> AA.pl <- xyPlot(X, Z, Plot=list(color="darkred"),
+   xaxis.range=c(-2,2), xlabels=5,
+   xtitle="Random Data", ytitle="Uncorrelated Data",
+   margin=AA.gr)
> # Add the title
> addTitle("B")
> # Required call to close PDF output graphics
> graphics.off()
```

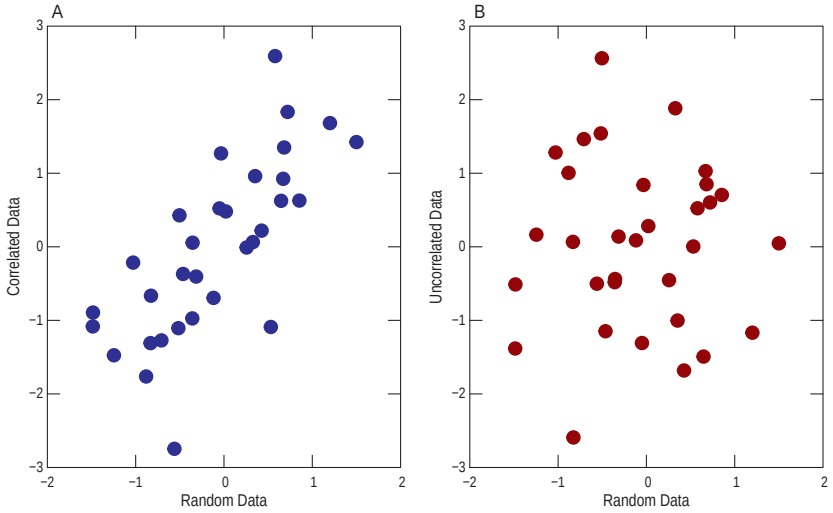

Figure 5. Example Graphs.

6 Multiple Graphs with an Explanation

Sometimes multiple graphs require a separate explanation, also called a legend, to describe the information that is plotted. The `setLayout` function can allocate space for the explanation to the right, at the bottom, or in a grid cell. When placing an explanation in a grid cell, it is important to distinguish between the grid cell number, which ranges from 1 (in the upper left corner) to the number of rows times the number of columns (in the lower right corner), from the graph number, which ranges from 1 to the number of graphs. The graph numbers are laid out much like the grid cell numbers, but skip the grid cell allocated to the explanation. Note that not all graphs numbers need to be used and that they can be used in any order.

The example below illustates how to set up an explanation within a grid cell. The layout is printed to illustrate the configuration of the graph number is the cells.

```
> # Set up the graphics environment, the equivalent call for an on screen
> #  device would be setPage(layout=list(width=6, height=4)).
> setSweave("graph06", 6 ,6)
> # Set the layout for 2 rows and 2 columns with the explanation in grid cell 2
> # Note that num.graphs must be set
> AA.lo <- setLayout(num.cols=2, num.rows=2, num.graphs=3, explanation=list(grid=2))
> # Print the layout to see how the graph number are assigned to the grid cells
> print(AA.lo)

Graph numbers 1 through  3
Explanation is e
Grid cell not available is .

 1e
 23

> # The first graph (box plot) is in the upper-left corner
> AA.gr <- setGraph(1, AA.lo)
> AA.bp <- boxPlot(Y, Z, Box=list(type="simple", show.counts=FALSE),
+   xlabels=c("Correlated", "Uncorrelated"))
> # The explanation for the box plot can be added now or after the other graphs, but add now
> AA.gr <- setGraph("explanation", AA.lo)
> addExplanation(AA.bp, title=expression(bold("Box Plot Description")))
> # Create a scatter plot from the X and Y data.
> AA.gr <- setGraph(2, AA.lo)
> AA.pl <- xyPlot(X, Y, Plot=list(color="blue"),
+   xaxis.range=c(-2,2), xlabels=5,
+   xtitle="Random Data", ytitle="Correlated Data",
+   margin=AA.gr)
> # ANd the final graph
> AA.gr <- setGraph(3, AA.lo)
> # Create a scatter plot from the X and Y data.
> AA.pl <- xyPlot(X, Z, Plot=list(color="darkred"),
```

```
+   xaxis.range=c(-2,2), xlabels=5,
+   xtitle="Random Data", ytitle="Uncorrelated Data",
+   margin=AA.gr)
> # Required call to close PDF output graphics
> graphics.off()
```

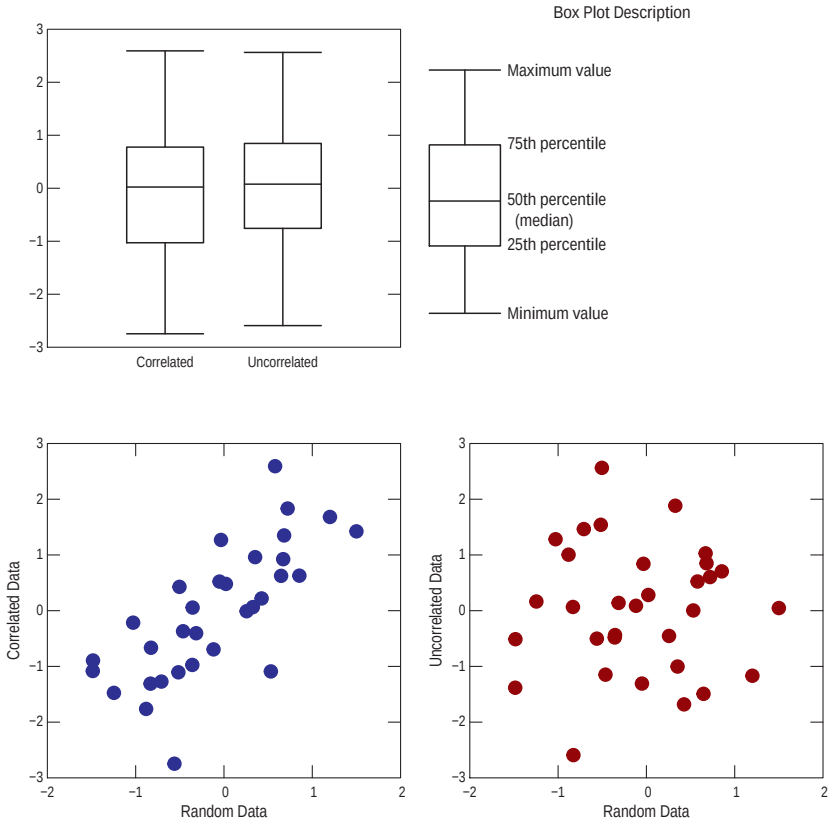

Figure 6. Figure with multiple graphs and an explanation.

7 Multiple Graphs with a Shared Axis

Sometimes it is desireable that either x- or y-axes be shared across a column or row. Sharing axes requires that the axis ranges be controlled for all graphs in the column or row. The `setLayout` function controls axis sharing by the `shared.x` and `shared.y` arguments. Nonnegative values to those arguments indicate sharing—a value of 0 indicates touching axes, positive value are relative spacing based on the default character size. The example below illustates a simple case.

```
> # Set up the graphics environment, the equivalent call for an on screen
> #  device would be setPage(layout=list(width=6, height=4)).
> setSweave("graph07", 4 ,6)
> # Set the layout for 2 graphs in one column with shared x-axes
> AA.lo <- setLayout(num.rows=2, shared.x=1)
> # The first graph is the upper graph
> AA.gr <- setGraph(1, AA.lo)
> # Create a scatter plot from the X and Y data.
> AA.pl <- xyPlot(X, Y, Plot=list(color="blue"),
+   xaxis.range=c(-2,2), xlabels=5,
+   ytitle="Correlated Data",
+   margin=AA.gr)
> # The second graph is placed immediately below
> AA.gr <- setGraph(2, AA.lo)
> # Create a scatter plot from the X and Y data.
> AA.pl <- xyPlot(X, Z, Plot=list(color="darkred"),
+   xaxis.range=c(-2,2), xlabels=5,
+   xtitle="Random Data", ytitle="Uncorrelated Data",
+   margin=AA.gr)
> # Required call to close PDF output graphics
> graphics.off()
```

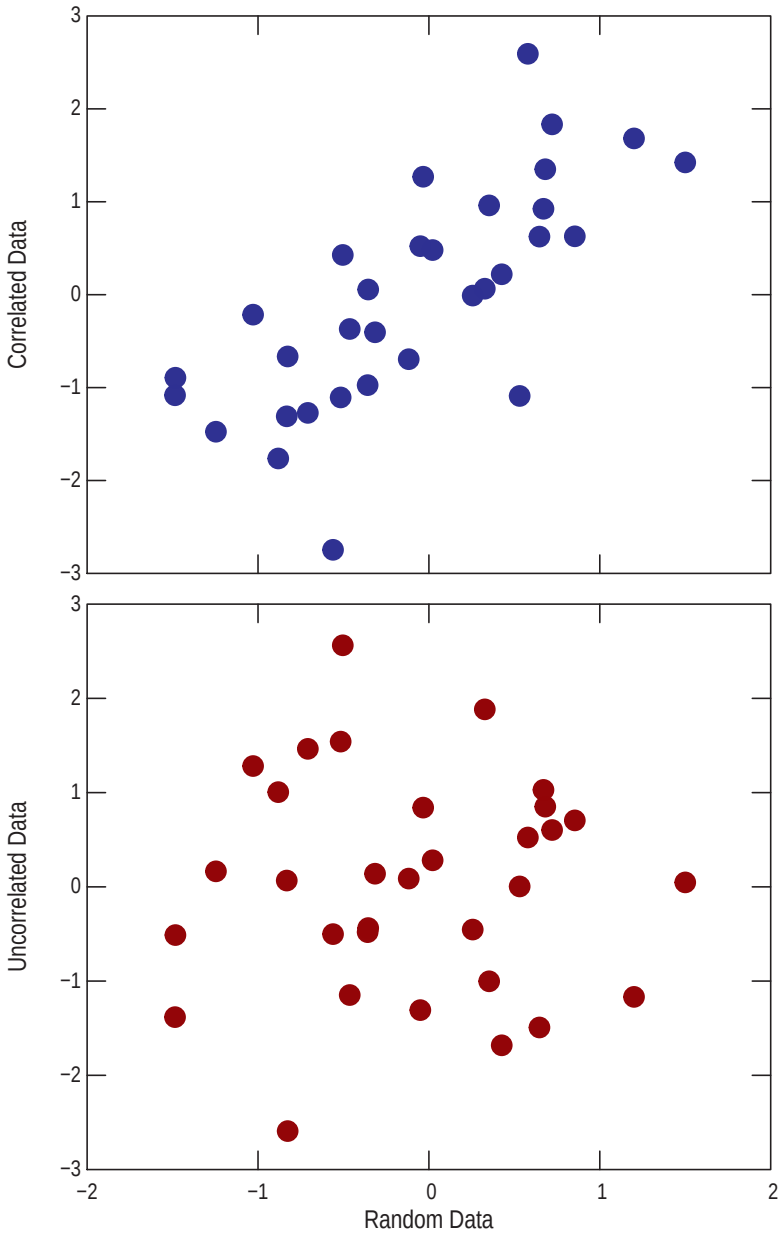

Figure 7. Figure of multiple graphs with a shared x-axis.

8 Restoring Axis Labels

Sometimes it is necessary to suppress drawing the axis ticks, labels, and title until the graph is complete. This is unusual, but can easily happen when drawing shaded areas. This example illustrates how to add the ticks, labels, and title after adding a shaded area.

```
> # Set up the graphics environment, the equivalent call for an on screen
> #  device would be setPage(layout=list(width=6, height=4)).
> setSweave("graph08", 6 ,5)
> # Set the layout for 2 graphs in one column with shared x-axes
> # Create a scatter plot from the X and Z data.
> AA.pl <- xyPlot(X, Z, Plot=list(color="blue"),
+   xaxis.range=c(-2,2), xlabels=-5,
+   yaxis.range=c(-3,2),
+   ytitle="Uncorrelated Data", xtitle="")
> # Add the shaded are just above the bottom x-axis
> addArea(sort(X), Zfill, ybase=-3, current=AA.pl)
> # Now add ticks, labels and title
> addAxisLabels("bottom", AA.pl, ticks=TRUE, labels=TRUE, title="X")
> # Required call to close PDF output graphics
> graphics.off()
```

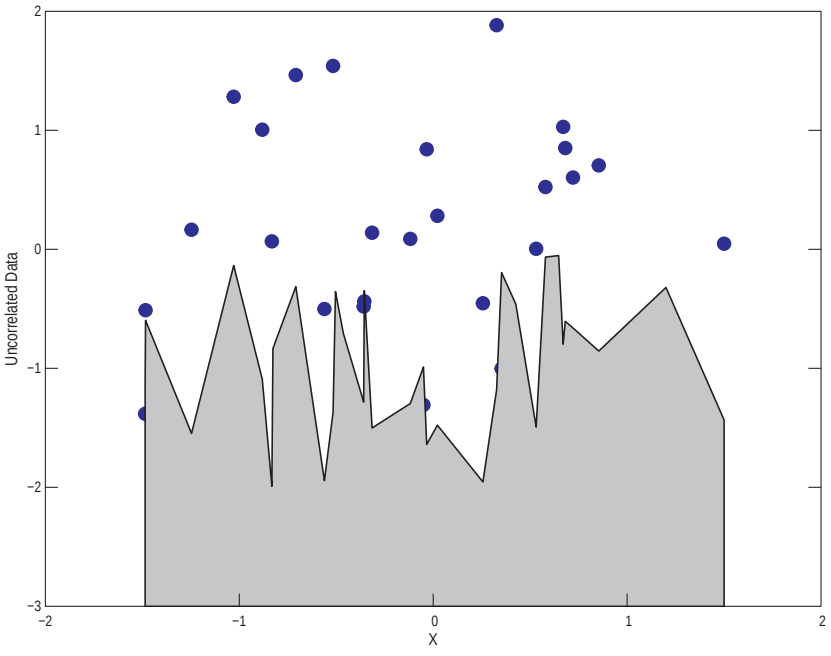

Figure 8. Graph showing axis ticks on top of shaded area.