Graph Gallery

Dave Lorenz

October 13, 2015

Abstract

These examples demonstrate selected high-level plotting functions in the smwrGraphs package that are not covered in other vignettes of demos. **NOTE:** to use any of these functions, 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

Most of the examples use data from the smwrData package. The data are retrieved and modified in the following code. Randomly generated data required for one example is generated in that example.

```
> # Load the smwrGraphs package
> library(smwrGraphs)
> # Load the smwrData package and some data
> library(smwrData)
> data(IonBalance)
> data(MiscGW)
> # Transform the data. These examples will ignore potassium, fluoride,
> # nitrate, and carbonate.
> PD <- transform(MiscGW, Ca.meq = conc2meq(Calcium, "calcium"),
+   Mg.meq = conc2meq(Magnesium, "magnesium"),
+   Na.meq = conc2meq(Sodium, "sodium"),
+   Cl.meq = conc2meq(Chloride, "chloride"),
+   SO4.meq = conc2meq(Sulfate, "sulfate"),
+   HCO3.meq = conc2meq(Bicarbonate, "bicarb")) # abbreviations allowed
> data(MC11_1993)
```

2 Graph with Error Bars

The `xyPlot` function plots paired x- and y-coordinate data. As of version 1.0, there are methods for factor and numeric x-coordinate data and numeric y-coordinate data. This example plots factor data and over plots the error bars showing the confidence interval.

```
> # 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("ggplot01", 6 ,6)
> # Construct means and 95 percent confidence intervals for selected constituents
> # in the IonBalance dataset (smwrData)
> Stats <- lapply(IonBalance[, c("Ca", "Mg", "Na")],
+   function(x) {
+     stats <- t.test(x)
+     return(c(mean=as.vector(stats$estimate),
+       U95=stats$conf.int[2], L95=stats$conf.int[1]))
+   })
> Stats <- as.data.frame(do.call(rbind, Stats))
> Stats$Constituent <- as.factor(row.names(Stats))
> # Now the plot and add error bars
> AA.pl <- with(Stats, xyPlot(Constituent, mean, yaxis.range=c(0,4),
+   Plot=list(name="Mean"),
+   ytitle="Concentation in milli-equivalents per liter"))
> AA.pl <- with(Stats, addErrorBars(Constituent, L95, U95,
+         Bars=list(name="95 percent confidence interval"), current=AA.pl))
> # And the explanation
> addExplanation(AA.pl, "ur")
> # Required call to close PDF output graphics
> graphics.off()
```

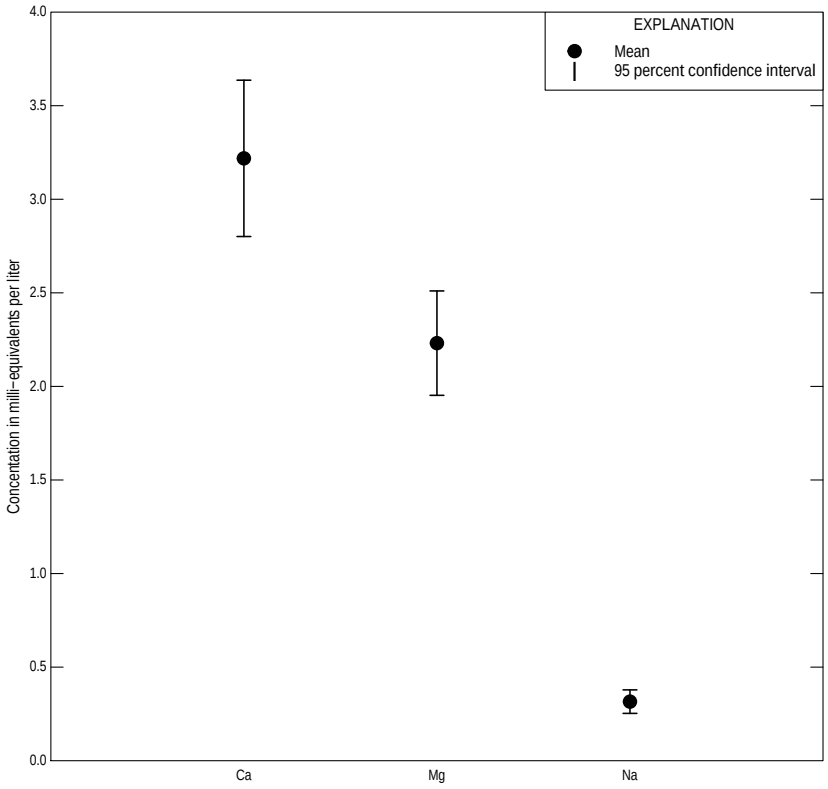

Figure 1. A graph with error bars.

3 Biplot

The biplot is a type of exploratory graph to show a simple two-variable scatter plot. A biplot allows information about both rows and columns of a data matrix to be displayed graphically. Row information is displayed as points and column information is displayed typically as vectors. The `biPlot` function will construct a biplot. The default method requires a matrix to represent the rows and columns of the data matrix. The biplot is typically used to display information from a principal component analysis and a method exists for the output from `princomp` function. That method is is demonstrated in this example.

The scaling between observations and variables is controlled by the `Scale` argument, which can take any value between 0 and 1 or a character string indicating a specific scaling. The options for the character string are "distance," which produces a plot where the observations retain their approximate relation with respect to Euclidean distances and corresponds to a numeric value of 1; "variance," which produces a plot where the cosine of the angle between the variable vectors is related to the correlation between the variables and corresponds to a numeric value of 0; "symmetric," which tries to balance the range of values for observations and variables to give a pleasing graph and corresponds to a numeric value of 0.5; or "Auto," which is the same as "variance." This scaling scheme echoes the presentation in Gower and Hand (1996).

```
> # 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("ggplot02", 6 ,6)
> # Construct the PCA from calcium, magnesium, and sodium and print it
> MGW.pp <- princomp(data.matrix(MiscGW[, c("Calcium", "Magnesium", "Sodium")]),
+   cor=TRUE, scores=TRUE)
> print(MGW.pp)

Call:
princomp(x = data.matrix(MiscGW[, c("Calcium", "Magnesium", "Sodium")]),
    cor = TRUE, scores = TRUE)

Standard deviations:
   Comp.1    Comp.2    Comp.3
1.5562434 0.6042572 0.4614971

 3  variables and  4 observations.

> # Create the biplot using the default settings
> biPlot(MGW.pp)
> # Required call to close PDF output graphics
> graphics.off()
```

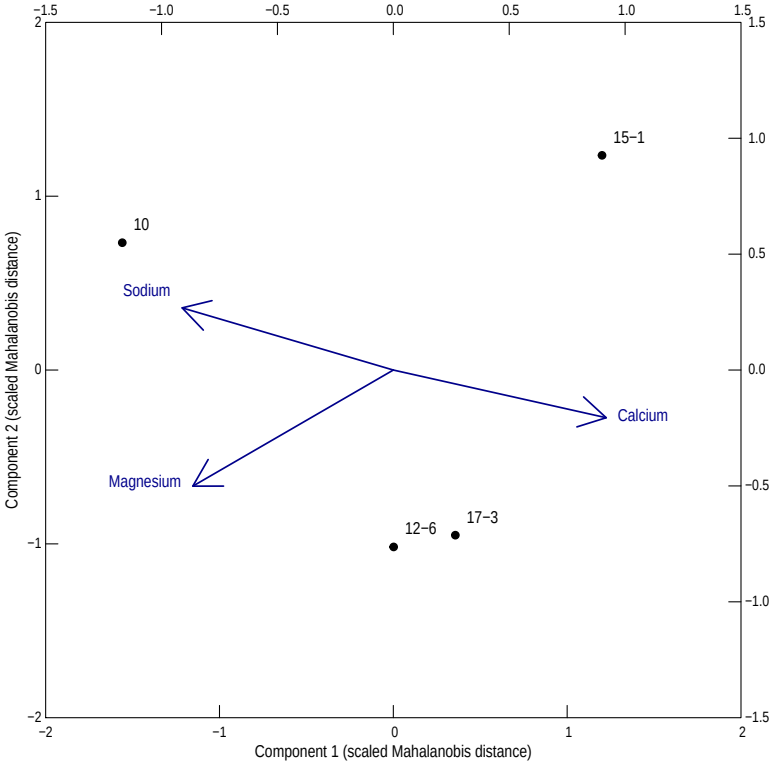

Figure 2. A biplot from a principal component analysis.

4 Scaleless Graph

Scales on ordination graphs are relative to the data shown and have no absolute value. When general relations between parameters are shown without the use of a grid or scale, arrows must be added to show general direction of increasing amount. The actual value of the scale from a principal component analysis or non-metric multidimensional scaling are relative to the data and can be shown with an arrow indicating the general increasing value.

This example shows a scaleless graph from the previous principal component analysis and illustrates the use of `addStiff` to show the actual ionic strengths.

```
> # 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("ggplot03", 6 ,6)
> # Use the PCA from the previous example. The scores component provides the
> # coordinate information for each observation.
> # From the biplot, the first axis repesent increasing calcium and the second
> # axis increasing difference between sodium and magnesium.
> #
> # First set up the graph
> AA.pl <- with(MGW.pp, xyPlot(scores[,1], scores[,2], Plot=list(what="none"),
+   ylabels=0, xlabels=0, ytitle="", xtitle=""))
> # Add the axis Labels, these can be placed at the axis minimum, given by
> # the "usr" parameter.
> addLabel(expression("Increasing Relative Calcium" %->% ""), par("usr")[1],
+   "bottom", justification="left", current=AA.pl)
> addLabel(expression("Increasing Sodium - Magnesium difference" %->% ""),
+   par("usr")[3],
+   "left", justification="left", current=AA.pl)
> # Add the Stiff diagrams
> AA.pl <- with(MiscGW, addStiff(MGW.pp$scores[,1], MGW.pp$scores[,2],
+   width=1, height=0.5, cations=cbind(Calcium, Magnesium, Sodium),
+   anions=cbind(Bicarbonate, Sulfate, Chloride),
+   catlabels=c("Ca", "Mg", "Na"), anlabels=c("HCO3", "SO4", "Cl"),
+   current=AA.pl))
> # Note that there will be overlapping of the Stiff diagram
> # Required call to close PDF output graphics
> graphics.off()
```

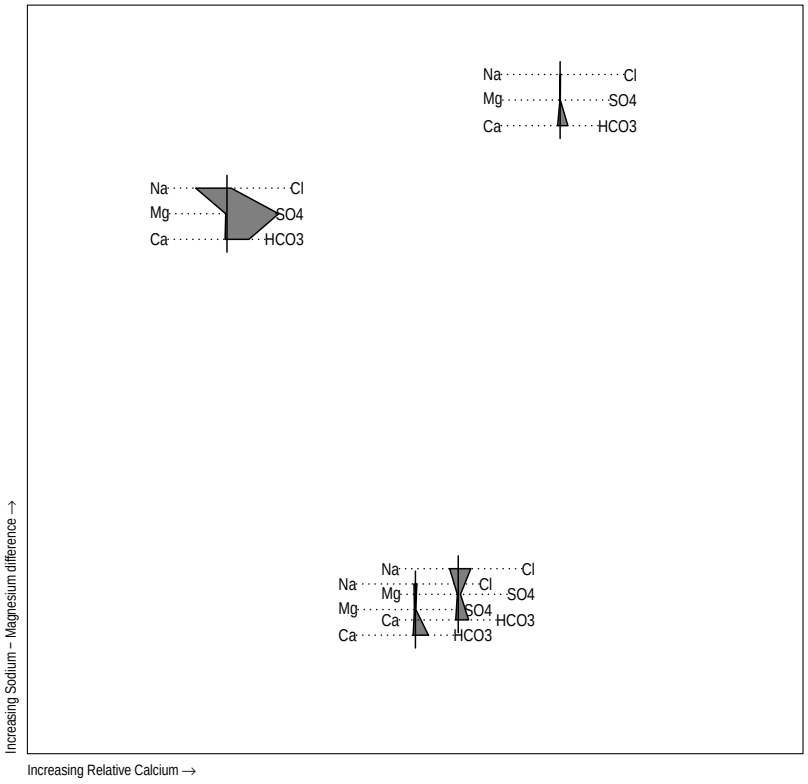

Figure 3. A scaleless graph also showing Stiff diagrams.

5 Area Graph

This example shows a simple area graph that shows the area under a log-Pearson type III density curve. The function `dlpearsonIII` is in the `smwrBase` package that is automatically loaded by when the `smwrGraphs` package is loaded.

The `areaPlot` function only works with numeric x-coordinate data. See the demo `durationHydrograph.r` to see an example using date/time x-coordinate data.

```
> # 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("ggplot04", 6 ,6)
> # First generate the sequence of x-coordinate values
> Xdata <- seq(0, 5, length.out=101)
> # Draw the curve
> areaPlot(Xdata, dlpearsonIII(Xdata, 0, 1, .5), Areas=list(fillDir="under",
+   fillColors="gray"), xaxis.range=c(0,5))
> # Required call to close PDF output graphics
> graphics.off()
```

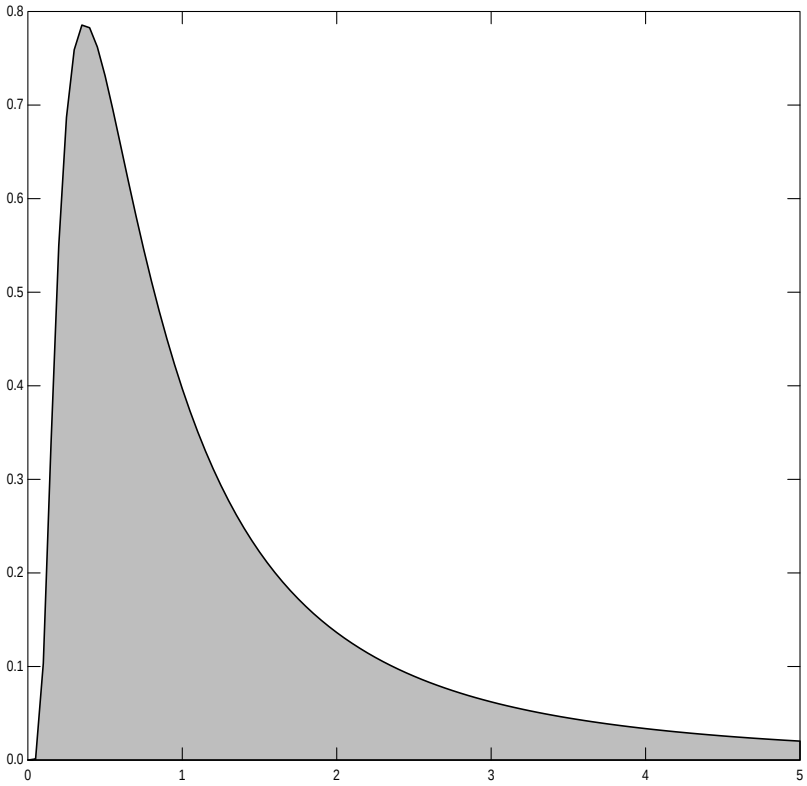

Figure 4. An example area graph.

6 Contour Plot

The `contourPlot` function can draw contour lines or filled contours from either x, y, and z data or from a matrix with x and y grid locations. The example in this section creates a filled contour graph of soil-temperature data from data collected at known depths at regular intervals, but uses the methods for arbitrary x, y, and z data.

```
> # 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("ggplot05", 6 ,6)
> #
> # First set up the data in proper form--separate columns for the x- (JULIAN),
> # y-(Depth), and z-axis (Temperature) data.
> MC11_stack <- reshape(MC11_1993[, c(2, 6:10)], direction="long", varying=list(2:6),
+   timevar="Depth", times=c(0.5, 1.0, 1.5, 2.0, 2.5), v.names="Temperature")
> # Set up for an explanation
> AA.lo <- setLayout(explanation=list(right=1.5))
> # Create the contour plot, Note that z is the first argument and no option for
> # reversing the sense of the y axis!
> setGraph(1, AA.lo)
> AA.pl <- with(MC11_stack, contourPlot(Temperature,JULIAN, -Depth,
+   Contours=list(name="Soil Temperature", filled=TRUE),
+   xaxis.range=c(50, 350), ytitle="Depth in meters below land surface",
+   xtitle="Julian day, 1993"))
> # Note to set the specific levels of the contours, for example to draw only
> # unfrozen ground, add levels=seq(0, 20, by=2) to the list defining Contours
> # Add the explanation
> setGraph("explanation", AA.lo)
> addExplanation(AA.pl)
> # Required call to close PDF output graphics
> graphics.off()
```

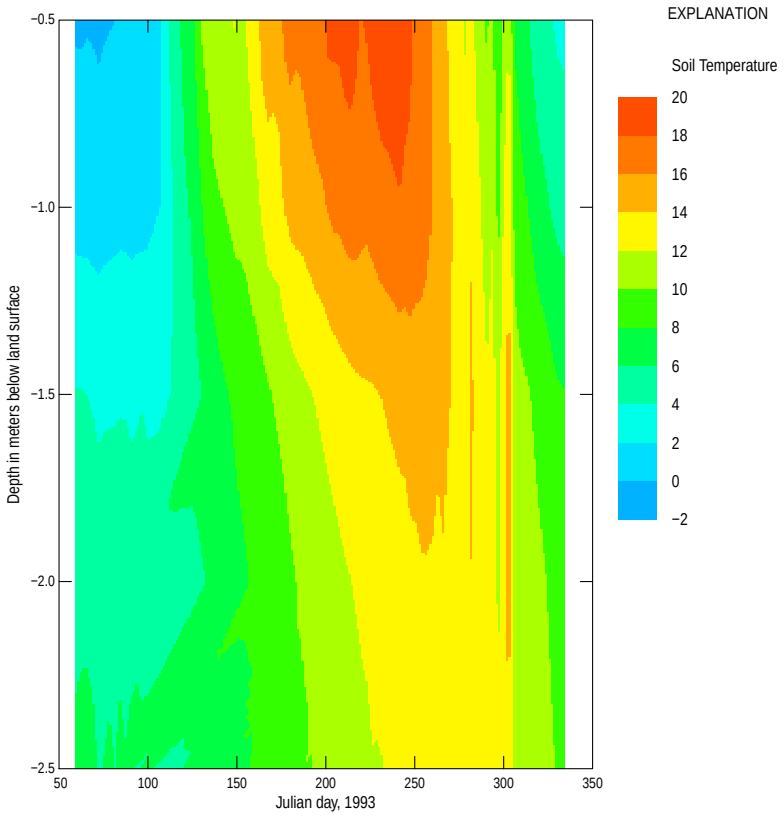

Figure 5. A contour plot with filled contours.

7 Correlogram

The `corGram` function is used primarily as a diagnostic plot for regressions on irregularly spaced samples. The term correlogram can also be used in time-series analysis, where the observations are regularly spaced and in spatial analysis to display spatial correlation.

This example shows 5 conditions typically shown in regression diagnostics: no correlations structure, serial correlation, seasonal or temporal lack of fit, and time-series anomalies. It uses randomly generated data to display the conditions. Each of the y values for the conditions will be scaled to a mean of 0 and standard deviation of 1 because the y values are not centered by `corGram`.

```
> # 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("ggplot06", 6 ,6)
> #
> # Generate the random data.
> set.seed(1236)
> # Use sorted observations in decimal format for 4 years of data collection
> Rdates <- sort(runif(100, 2010, 2014))
> # The random, no serial correlation data
> Yrand <- scale(rnorm(100))
> # Add serial correlation
> Yser <- scale(Yrand + c(0, Yrand[-100]) + c(0, 0, Yrand[-c(99,100)])/3)
> # Seasonal and temporal lack of fit
> Yseas <- scale(Yrand + cos(2*pi*Rdates))
> Ytime <- scale(Yrand + seq(-1, 1, length.out=100))
> # And the anomaly
> Yanom <- scale(Yrand + 0.75*(Rdates > 2011.1 & Rdates < 2011.7))
> # Set up for the graphs and create the correlograms
> # These use gray90 color and reset the yaxis range to emphasize the line
> AA.lo <- setLayout(num.rows=3, num.cols=2)
> setGraph(1, AA.lo)
> corGram(Rdates, Yrand, Plot=list(color="gray90"), yaxis.range = c(-1, 1))
> addTitle("Random data")
> setGraph(2, AA.lo)
> corGram(Rdates, Yser, Plot=list(color="gray90"), yaxis.range = c(-1, 1))
> addTitle("Serial Correlation")
> setGraph(3, AA.lo)
> corGram(Rdates, Yseas, Plot=list(color="gray90"), yaxis.range = c(-1, 1))
> addTitle("Seasonal Lack of Fit")
> setGraph(4, AA.lo)
> corGram(Rdates, Ytime, Plot=list(color="gray90"), yaxis.range = c(-1, 1))
> addTitle("Temporal Lack of Fit")
> setGraph(5, AA.lo)
> corGram(Rdates, Yanom, Plot=list(color="gray90"), yaxis.range = c(-1, 1))
> addTitle("Anomaly")
> # Required call to close PDF output graphics
> graphics.off()
```

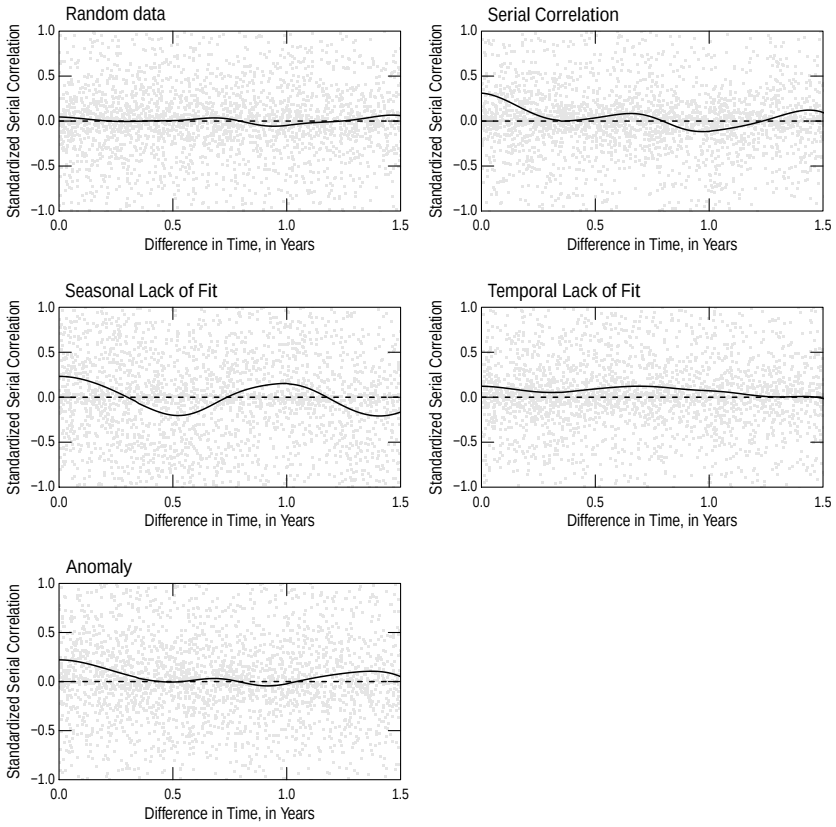

Figure 6. Selected correlograms.

The smooth line correlogram for random, uncorrelated data shows only small and irregular fluctuations around the zero line, much like the example. Serial correlation is indicated when the smooth line is substantially greater than zero at a time difference of zero years, gradually approaches zero and then fluctuates around the zero line; the fluctuation in the example are a bit larger than expected. A seasonal lack of fit is indicated by a smooth line that shows a distinct periodic pattern—a single period is indicated in the example, above zero at a difference in 0 and 1 year and less than zero at a difference of 0.5 year; a double period show a line above 0 at 0, .5, and 1 year difference in time and less than zero at 0.25 and 0.75 year difference; and so forth. A lack of fit over time is indicated by a smooth line that persists above zero for a time difference greater than about 0.5 year, much as indicated in the example graph. The anomaly is more difficult to diagnose; it can show characteristics of the smoothed serial and seasonal correlogram lines. It is distinctly different from the seasonal smooth line in that there is no peak at a time difference of 1 year. The example shows a curve that is not substantially different from one showing serial correlation, but shows no peak near a time difference of 1 year.

8 Surface Plot

A surface plot is a three dimensional presentation of a surface in contrast to a contour plot that uses either lines or colors to indicate the surface. There are two steps to displaying a surface plot using the functions in `smwrGraphs`: the first step is to set the projection for the surface plot using `preSurface`, the second step is to draw the surface using `surfacePlot`. This example sets up a simple surface plot of the MC11 data used in the contour plot example.

```
> # First set up the projection. The preSurface function can be interactive,
> # This script sets the projection to A and supresses the request for user input.
> AA.pre <- with(MC11_1993, preSurface(JULIAN, c(-2.5, -2.0, -1.5, -1.0, -0.5),
+   cbind(TEMP.2.5, TEMP.2.0, TEMP.1.5, TEMP.1.0, TEMP.0.5), batch="A"))
> # 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("ggplot07", 6 ,6)
> #
> # Create the graph
> surfacePlot(AA.pre, xtitle="Julian Day, 1993", ytitle="Depth below landsurface",
+   ztitle="Soil Temperature")
> # Required call to close PDF output graphics
> graphics.off()
```

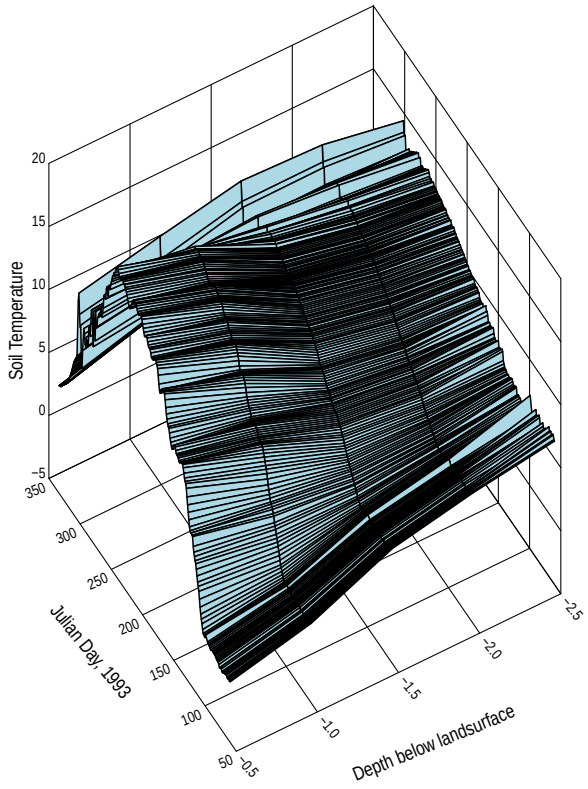

Figure 7. A surface plot.

9 A Report Graph

A report graph is simply formatted text written to a graphics device. The intent is generally to produce some kind of documentation for a statistical analysis. Oftentimes following pages will contain diagnostic or other graphs from the analysis. The function `reportGraph` will try to put the printed output from any R object onto a graph page. It does not check to verify that the width or length will fit in the graph area; any text that is to the right or below the text that can be seen is simply not shown. The example in this section simply shows the summary report from a simple linear regression.

```
> # 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("ggplot08", 6 ,4)
> #
> # Construct the regression
> AA.lm <- lm(Mg ~ Ca, data=IonBalance)
> # And write the report
> reportGraph(summary(AA.lm))
> # Required call to close PDF output graphics
> graphics.off()
```

```
Call:
lm(formula = Mg ~ Ca, data = IonBalance)

Residuals:
     Min       1Q   Median       3Q      Max
−0.71038 −0.22443  0.00127  0.17983  0.72960

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.44362    0.27754   1.598    0.126
Ca           0.55542    0.08291   6.699 1.61e−06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3579 on 20 degrees of freedom
Multiple R−squared:  0.6917, Adjusted R−squared:  0.6763
F−statistic: 44.88 on 1 and 20 DF,  p−value: 1.61e−06
```

Figure 8. A graph showing the report from a simple linear regression.

10 A Dendrogram

A dendrogram presents the results of a hierarchical cluster analysis in graphical form. This example will perform a cluster analysis of the the Calcium and Magnesium data in the IonBalance dataset. The example dendrogram suggests two clusters: samples taken in the fall and winter and samples taken in the spring and summer.

```
> # 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("ggplot09", 6 ,6)
> # Extract the data and assigne rownames based on sample date
> CaMg <- data.matrix(IonBalance[, c("Ca", "Mg")])
> rownames(CaMg) <- as.character(IonBalance$DATES)
> # Construct the distance matrix and the cluster analysis
> CaMg.dist <- dist(CaMg)
> CaMg.hclust <- hclust(CaMg.dist, method="average")
> # Dreaw the dendrogram
> dendGram(CaMg.hclust, ytitle="Tree Height")
> # Required call to close PDF output graphics
> graphics.off()
```

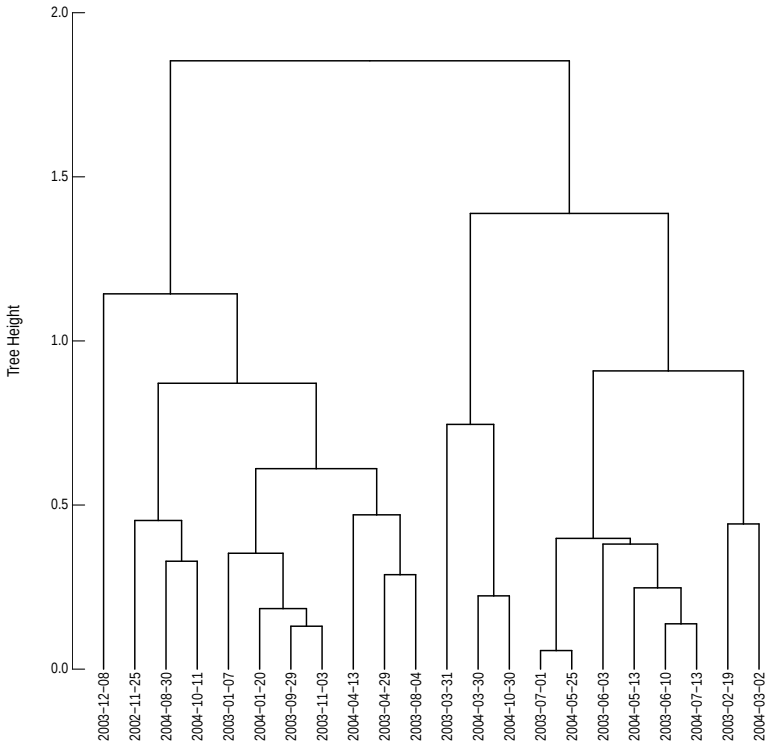

Figure 9. The dendrogram of the example cluster analysis.

References

[1] Gower, J.C. and Hand, D.J., 1996, Biplots, Chapman and Hall, London, 277 p.