Transcript of Mick Crawley's R course 2010 Imperial College London, Silwood Park

Emanuel G Heitlinger

Disclaimer: The following document is a private transcript of Mick Crawley's R-course. I am a participant in this course and my writeup has in no way been approved by Mick Crawly (from whom the ideas behind the code and teaching concepts are) or any of his staff.

First day

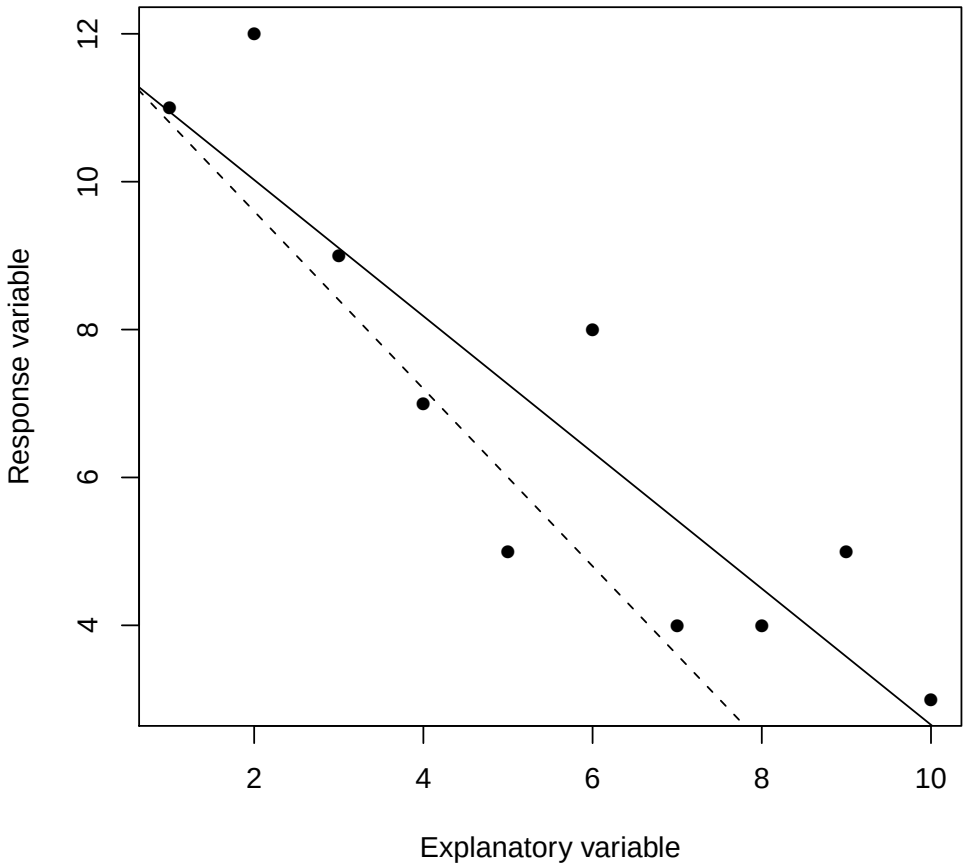
Plotting different categories of variables

First we produce two vectors. One (x) as a sequence of numbers, the second (y) as a collection of numbers we "invent".

```
> x <- 1:10
> y <- c(11, 12, 9, 7, 5, 8, 4, 4, 5, 3)
> x

 [1]  1  2  3  4  5  6  7  8  9 10

> y

 [1] 11 12  9  7  5  8  4  4  5  3
```

We plot these vectors with a regression line and an arbitrary line.

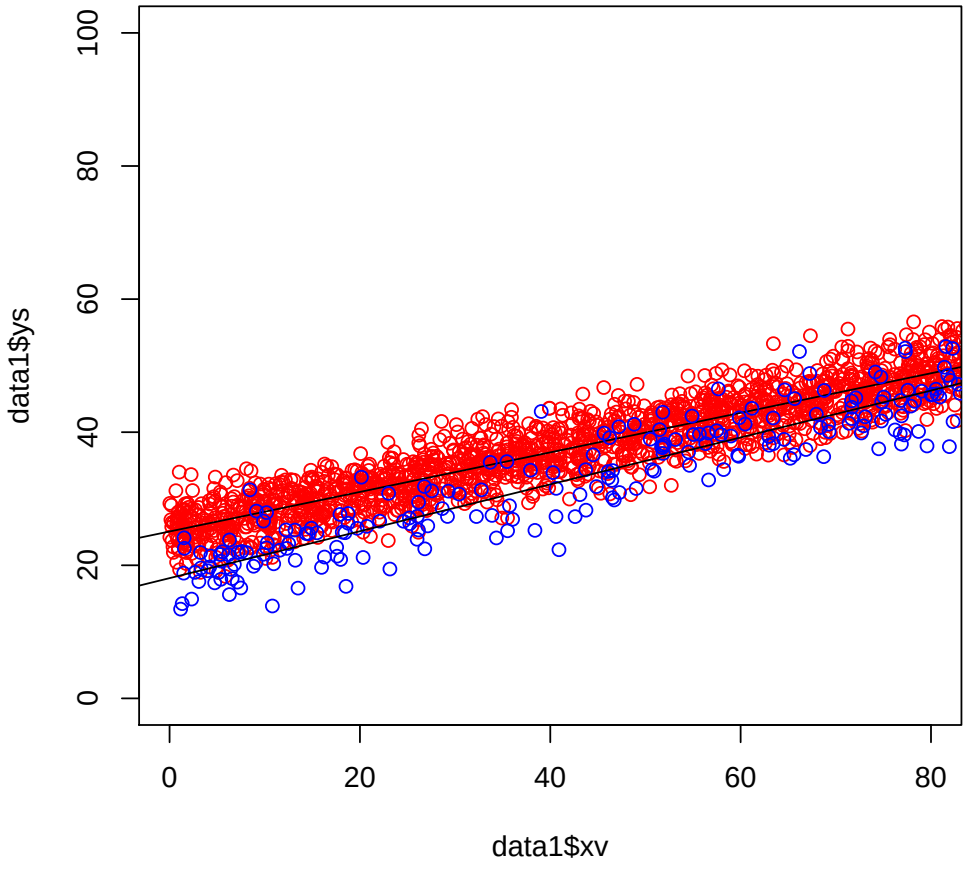
```
> plot(x, y, xlab = "Explanatory variable", ylab = "Response variable",
+     pch = 16)
> abline(lm(y ~ x))
> lines(c(0, 10), c(12, 0), lty = 2)
```



The regression line makes sense. Abline is quite clever to know what lm does and uses this as regression line.

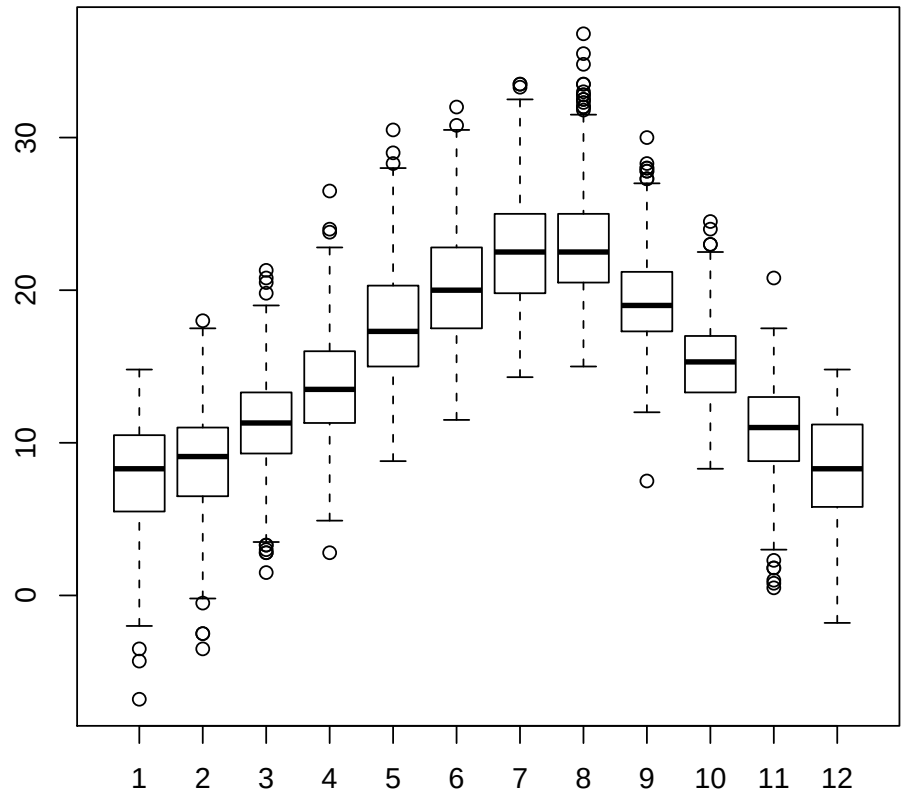
Lets read in some bigger data. We combine two data frames (from two different files) into one plot using points and abline on the original plot.

```
> data1 <- read.table("scatter1.txt", header = TRUE)
> data2 <- read.table("scatter2.txt", header = TRUE)
> plot(data1$xv, data1$ys, col = "red", xlim = c(0, 80), ylim = c(0,
+     100))
> abline(lm(data1$ys ~ data1$xv))
> points(data2$xv2, data2$ys2, col = "blue")
> abline(lm(data2$ys2 ~ data2$xv2))
```



Now look at factors. Here the default in base graphics is to plot them as box and whisker plots.

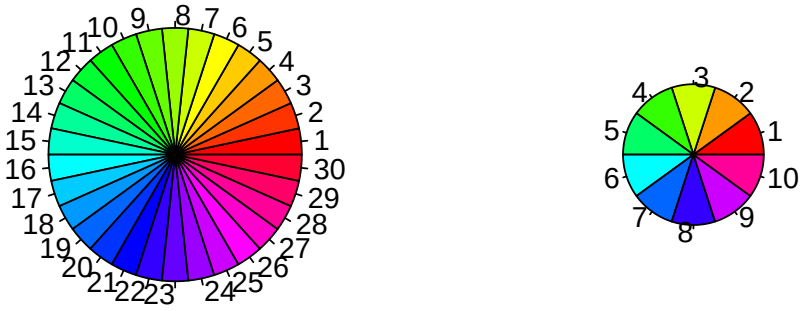
```
> weather <- read.table("SilwoodWeather.txt", header = T)
> weather$month <- factor(weather$month)
> plot(weather$month, weather$upper)
```



Colour in R

Draw a pie with 30 coloured pieces and a certain radius. Draw a smaller pie with 10 pieces on the same plot using par(mfrow=).

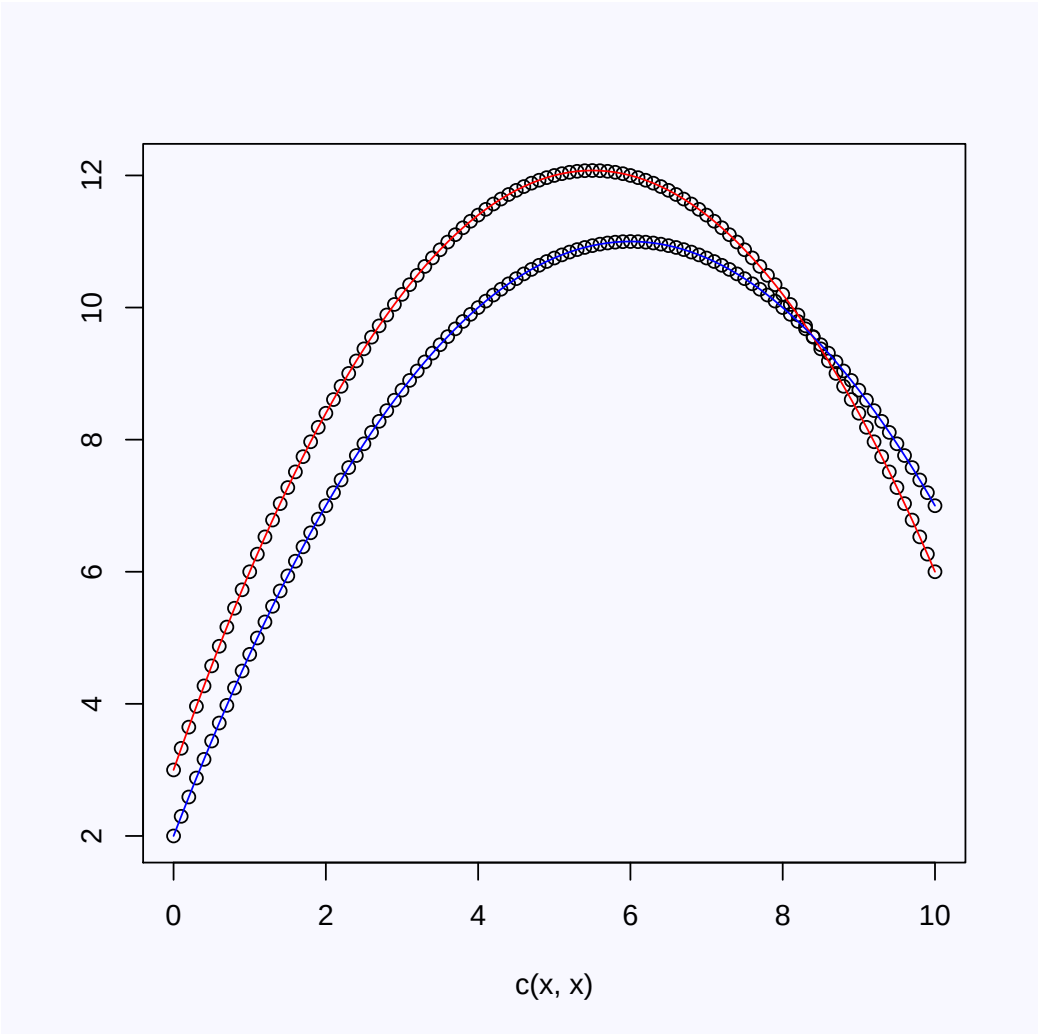
```
> par(mfrow = c(1, 2))
> pie(rep(1, 30), col = rainbow(30), radius = 0.9)
> pie(rep(1, 10), col = rainbow(10), radius = 0.5)
```



Colors in mathematical functions

Of course lines can also be curved and coloured.

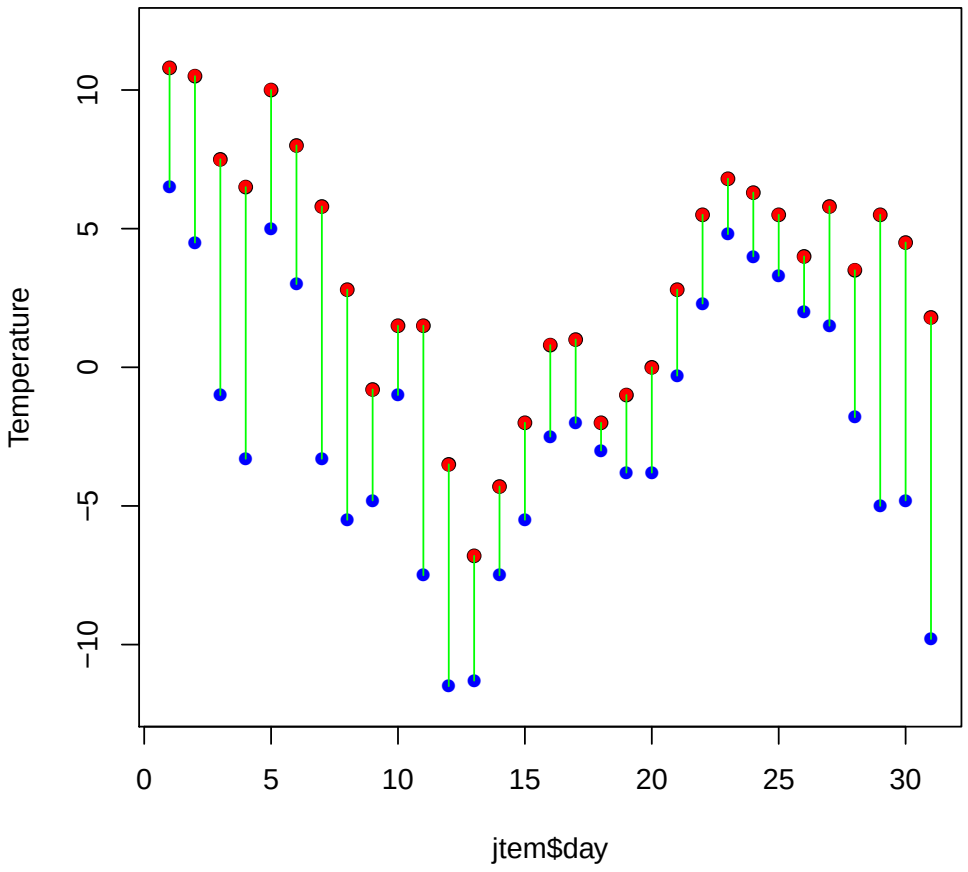
```
> x <- seq(0, 10, 0.1)
> y1 <- 2 + 3 * x - 0.25 * x^2
> y2 <- 3 + 3.3 * x - 0.3 * x^2
> par(bg = "ghostwhite")
> plot(c(x, x), c(y1, y2), ylab = "")
> lines(x, y2, col = "red")
> lines(x, y1, col = "blue")
```



Drawing sets of lines

We can join points in a plot with lines. In our example the minima and maxima.

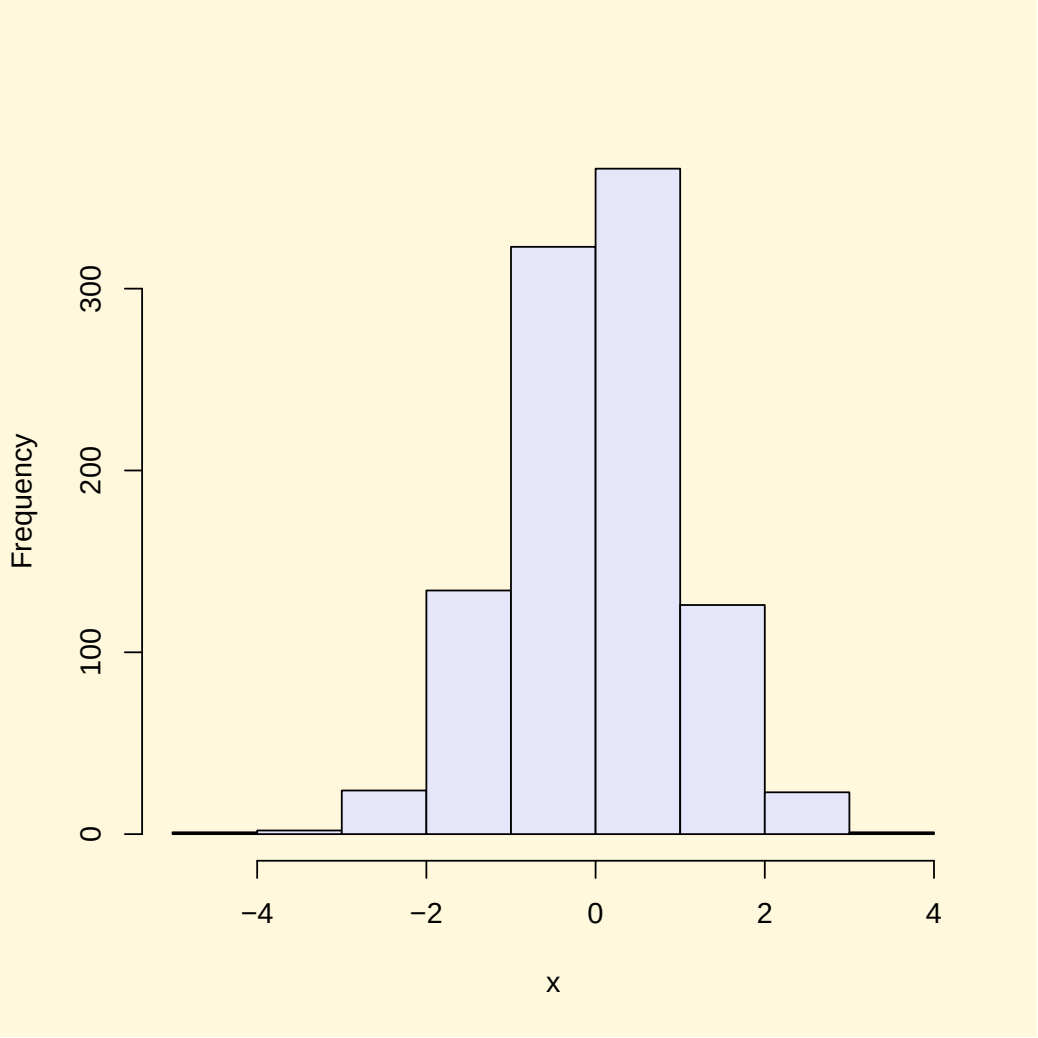
```
> jtem <- read.table("jantemp.txt", header = T)
> attach(jtem)
> plot(jtem$day, jtem$tmax, ylim = c(-12, 12), ylab = "Temperature")
> points(jtem$day, jtem$tmin, col = "blue", pch = 16)
> points(jtem$day, jtem$tmax, col = "red", pch = 16)
> join <- function(i) lines(c(i, i), c(tmin[i], tmax[i]), col = "green")
> bin <- sapply(1:31, join)
> detach(jtem)
```



Colours with histogramms

Some background colours look better thon the default white.

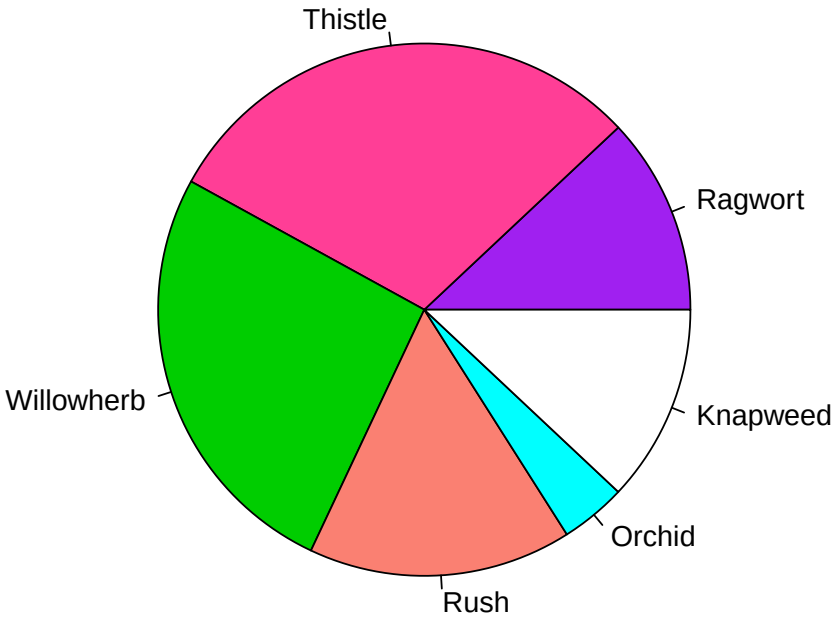
```
> x <- rnorm(1000)
> par(bg = "cornsilk")
> hist(x, col = "lavender", main = "")
```



Colours with labelled pie charts

Piecharts are generally a poor tool to present data. But anyways we can colour them.

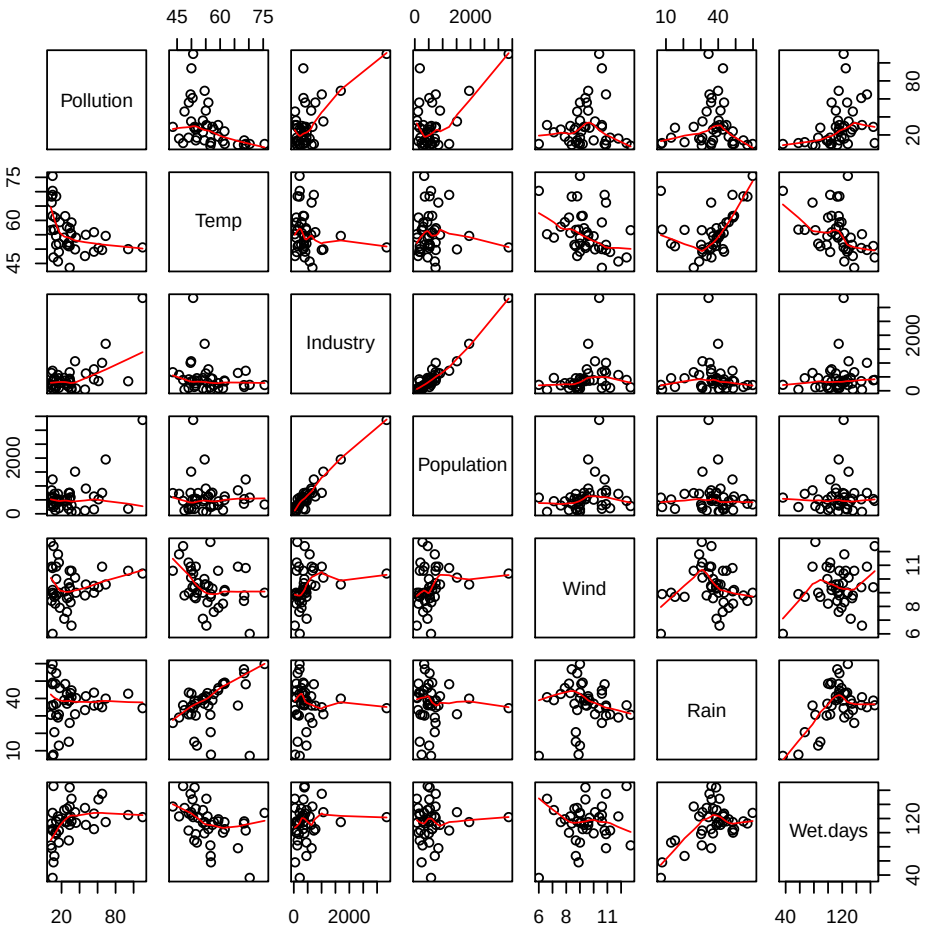
```
> fate <- c(0.12, 0.3, 0.26, 0.16, 0.04, 0.12)
> names(fate) <- c("Ragwort", "Thistle", "Willowherb", "Rush",
+     "Orchid", "Knapweed")
> pie(fate, col = c("purple", "violetred1", "green3", "salmon",
+     "cyan", "white"))
```



Multivariate plots

Multivariate plots are also possibel with base graphics, for more features look at the lattice package. Here we use the plotting function pairs, which is very useful in data exploration prior to modelling.

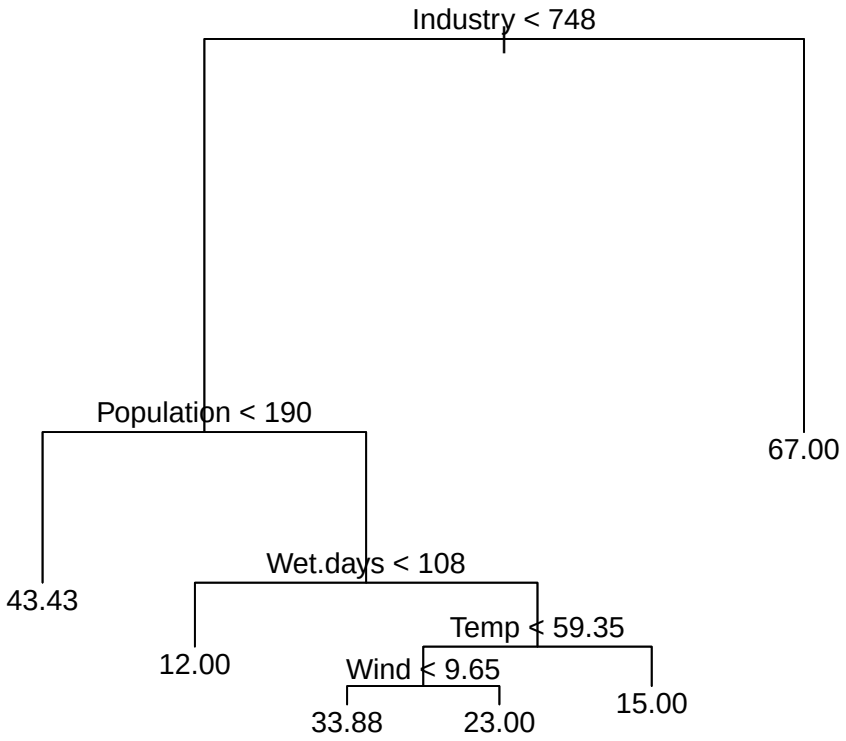
```
> pol <- read.table("Pollute.txt", header = T)
> pairs(pol, panel = panel.smooth)
```



Tree based modells

The tree package has useful objects to help model choice. Of course plots for these objects are very customized and useful.

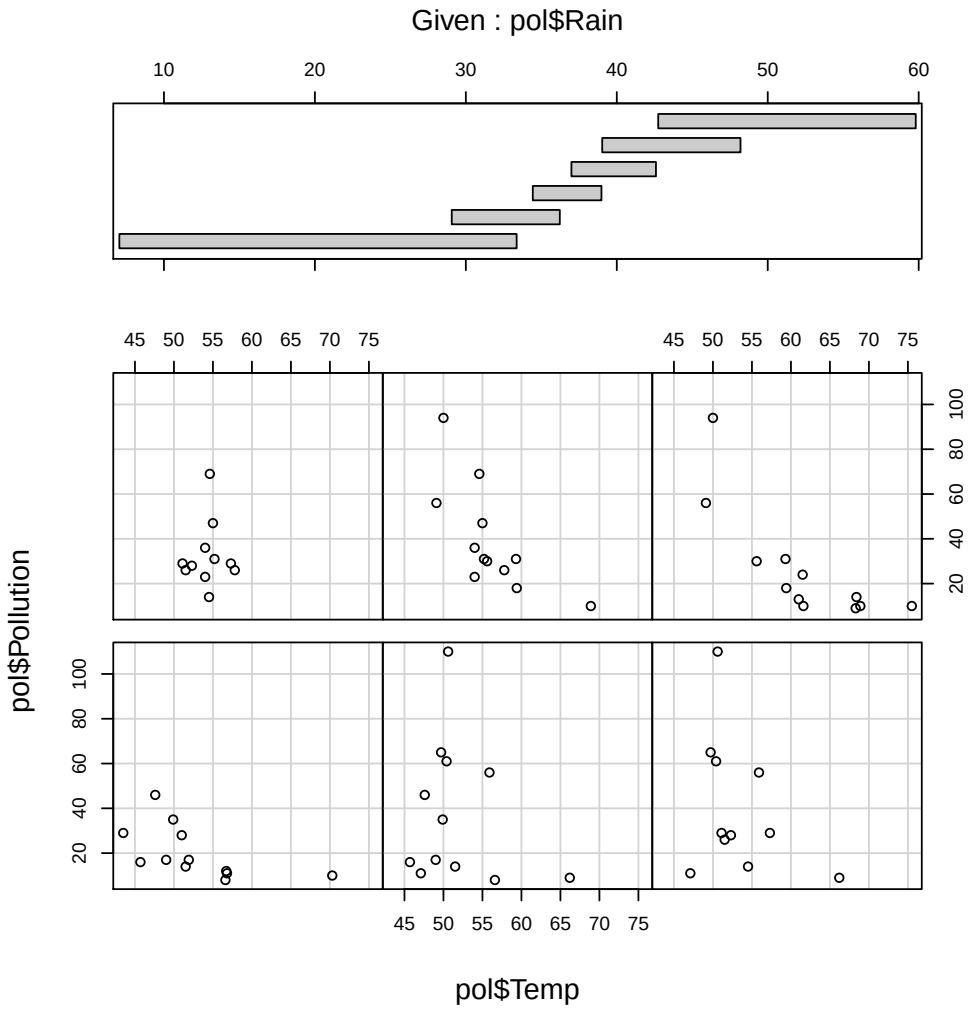
```
> library(tree)
> rtr <- tree(Pollution ~ ., data = pol)
> plot(rtr)
> text(rtr)
```



Conditioning plots

Conditioning plots can be very useful for multivariate data.

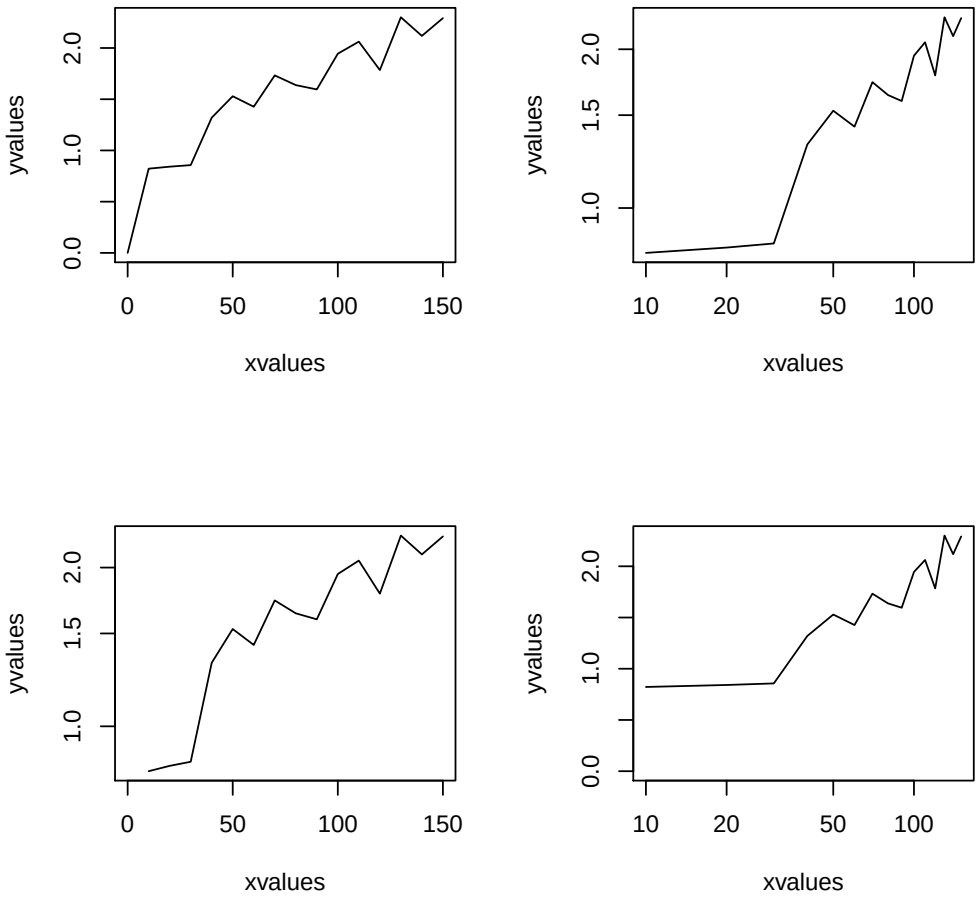
```
> coplot(pol$Pollution ~ pol$Temp | pol$Rain)
```



Logarithmic axis

Multiple plots and logarithmic axis.

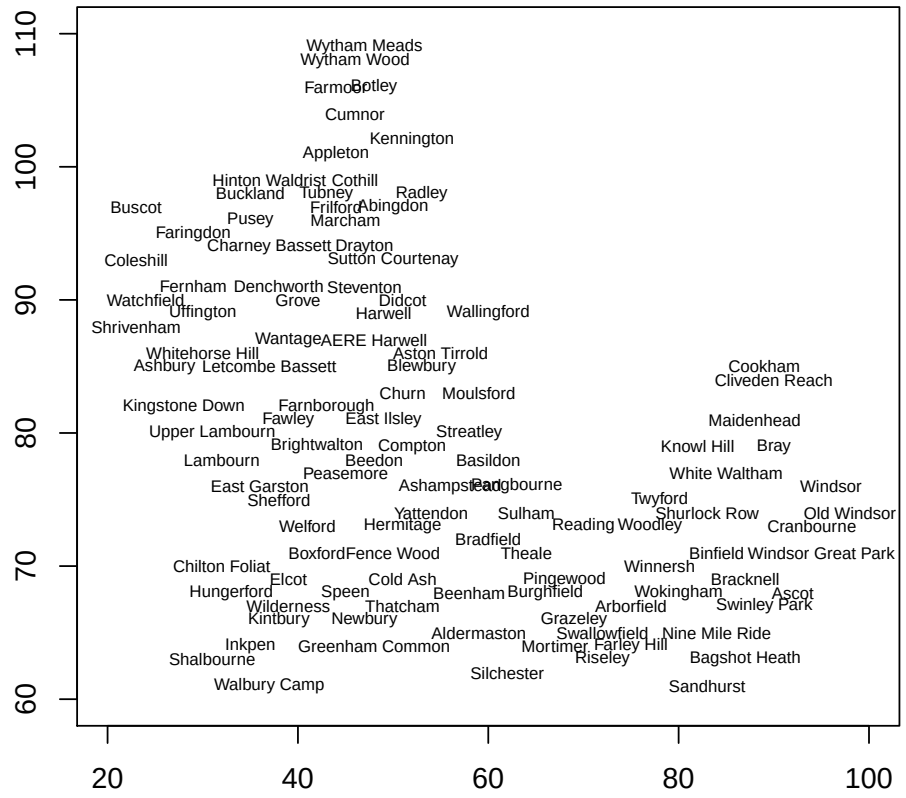
```
> plotdata <- read.table("logplots.txt", header = T)
> names(plotdata) <- c("xvalues", "yvalues")
> attach(plotdata)
> par(mfrow = c(2, 2))
> plot(xvalues, yvalues, type = "l")
> plot(xvalues, yvalues, log = "xy", type = "l")
> plot(xvalues, yvalues, log = "y", type = "l")
> plot(xvalues, yvalues, log = "x", type = "l")
> detach(plotdata)
```



Text in plots

We can use text in a loop to plot it (also on a more or less empty plot).

```
> map.places <- read.csv("map.places.csv", header = T)
> attach(map.places)
> map.data <- read.csv("bowens.csv", header = T)
> attach(map.data)
> nn <- ifelse(north < 60, north + 100, north)
> plot(c(20, 100), c(60, 110), type = "n", xlab = "", ylab = "")
> for (i in 1:length(wanted)) {
+     ii <- which(place == as.character(wanted[i]))
+     text(east[ii], nn[ii], as.character(place[ii]), cex = 0.6)
+ }
```



We can also play with letter alignment

```
> par(mfrow = c(1, 2))
> labels <- letters[1:10]
> plot(1:10, 1:10, type = "n")
> text(1:10, 1:10, labels, cex = 1.5)
> plot(1:10, 1:10, type = "n")
> text(1:10, 10:1, labels, cex = 1.5, srt = 180)
```