Graphics in R

Johannes Karreth
Department of Political Science
University of Colorado at Boulder
johannes.karreth@colorado.edu

MOPS Workshop
September 30, 2011

Contents

1 Why you want to use R for graphics (and most of your quantitative work)

R provides a wide range of highly customizable options to produce descriptive and statistical graphics. The three most common ways to generate graphics are the built-in base plotting device, the lattice and

ggplot2 packages. All three options are widely documented on the web. Some resources that I have found tremendously helpful are listed below. While not all of them might have the immediate solution for a specific question, searching the web has almost never failed me when I was looking for help with even the most specific problem! The **R** user community is huge.

The base plotting device is highly versatile for a variety of applications. The lattice package is particularly useful for displaying grouped data in panels, such as scatterplots for different subsets of one dataset. The ggplot2 package has become popular because it has a number of automated features that maximize the information displayed in a graph, adhering to Edward Tufte's suggestion to always use graphs with a high data-to-ink ratio.

Aside from graphing data and results, **R** makes it easier than most other software to extract quantities of interest from statistical models and work with these, e.g. for plotting, calculate predicted values, etc.

This document only shows a very small selection of graphics that can be made with **R**. Much more can be found at the links below.

2 Helpful resources

- Paul Murray's *R Graphics* book (Chapman & Hall/CRC, August 2011) has a companion website with lots of code: http://www.stat.auckland.ac.nz/~paul/RG2e/.
- Quick-R is a website with many examples, from simple to advanced: http://www.statmethods.net/.
- Fox and Weisberg's *R Companion to Applied Regression*: http://bit.ly/q20IbM and http://tinyurl.com/carbook. John Fox has written the car and effect packages. They provide very useful functions for data analysis, regression diagnostics, and the effect displays.
- William G. Jacoby's course materials on the lattice package are detailed and also present some data visualization theory at http://polisci.msu.edu/jacoby/icpsr/graphics/.
- Hadley Wickham's ggplot2 website: http://had.co.nz/ggplot2/.
- Christopher DeSante's *very brief* guide on ggplot2: http://bit.ly/n8lKuf.
- The R-Bloggers blog aggregator contains some interesting code for graphics (and other applications) every now and then: http://www.r-bloggers.com/. For instance, see this blog post on graphs in lattice vs. ggplot2: http://wp.me/psR36-ez.
- Text editors: See this list of editors, https://github.com/RatRiceEEB/RIntroCode/wiki/R-Resources, and have a look at RStudio – an open-source integrated development environment (IDE), http://rstudio.org/.

3 Packages used in the code below

Start by installing the following packages:

```
> install.packages(c("car", "lattice", "ggplot2", "coefplot2", "effects",
+     "arm", "rworldmap"))
```

4 Working directory

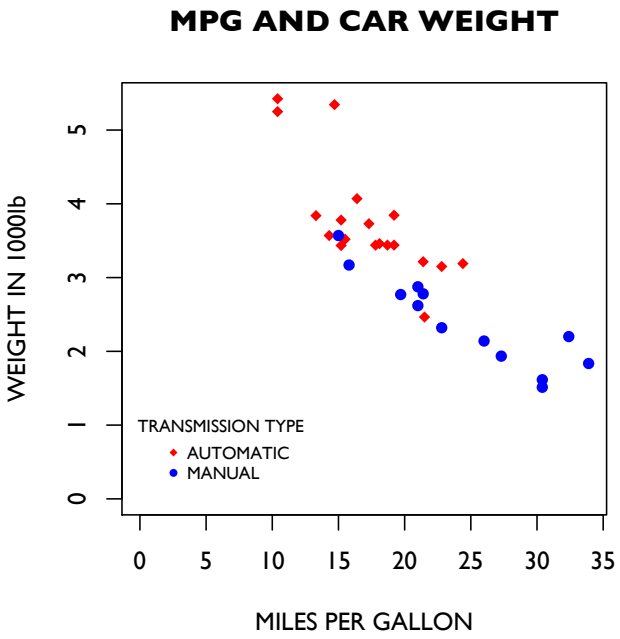
In most cases, it is useful to set a project-specific working directory – especially if you work with graphics that you want to have printed to .pdf or .eps files.

```
> setwd("~/Documents/Uni/Teaching/R")
```

5 Customizing plots

R offers a multitude of options to customize plots. Here are some: font (`family="Gill Sans"`), symbols (`pch=16`), and color (`col="blue"`). Much more information on those parameters can be optioned via `?plot`, `?par`, and for instance at Quick-R: http://www.statmethods.net/advgraphs/parameters.html.

```
> library(car)
> data(mtcars)
> cars.dat <- mtcars
> cars.aut <- cars.dat[cars.dat$am==0, ]
> cars.man <- cars.dat[cars.dat$am==1, ]
> par(family="Gill Sans")
> plot(cars.aut$mpg, cars.aut$wt, main="MPG AND CAR WEIGHT", pch=18, col="red",
+     xlab="MILES PER GALLON", ylab="WEIGHT IN 1000lb", ylim=c(0,
+     max(cars.dat$wt)), xlim=c(0, max(cars.dat$mpg)))
> points(cars.man$mpg, cars.man$wt, pch=16, col="blue")
> legend("bottomleft", pch=c(18, 16), col=c("red", "blue"),
+     title="TRANSMISSION TYPE", c("AUTOMATIC", "MANUAL"), cex=0.7, bty='n',
+     inset=0.05)
> par(family="sans")
```



6 Inspecting your data

6.1 Correlation plots

Generate fake data:

```
> x1 <- rnorm(1000, 50, 2)
> x2 <- rbinom(1000, 1, prob = 0.63)
> x3 <- rpois(1000, 2)
> x4 <- runif(1000, 40, 100)
> x5 <- rnorm(1000, 100, 30)
> x6 <- rbeta(1000, 2, 2)
> x7 <- rpois(1000, 10)
> x8 <- rbinom(1000, 1, prob = 0.4)
> x9 <- rbeta(1000, 5, 4)
> corr.dat <- cbind(x1, x2, x3, x4, x5, x6, x7, x8, x9)
> corr.cor <- cor(corr.dat)
```

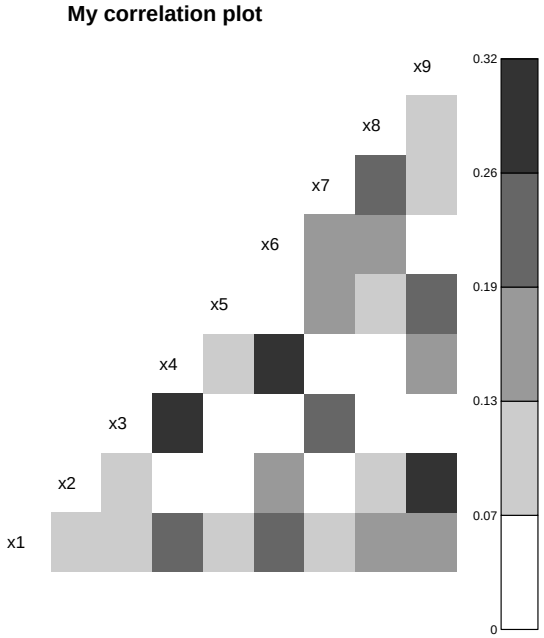
Or use an existing dataset:

```
> library(car)
> data(mtcars)
> cars.dat <- mtcars
> cars.cor <- cor(cars.dat)
```

6.1.1 Base plots

Use `corrplot` from the arm package to produce a correlation plot:

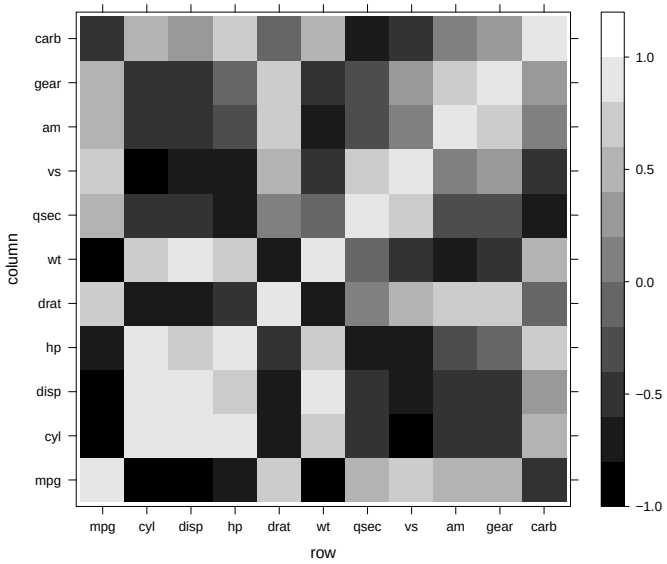
```
> corrplot(corr.cor)
> title(main = "My correlation plot")
> corrplot(cars.cor)
> title(main = "Correlation plot for the cars data")
```



6.1.2 Lattice

Or use `levelplot` in the lattice package:

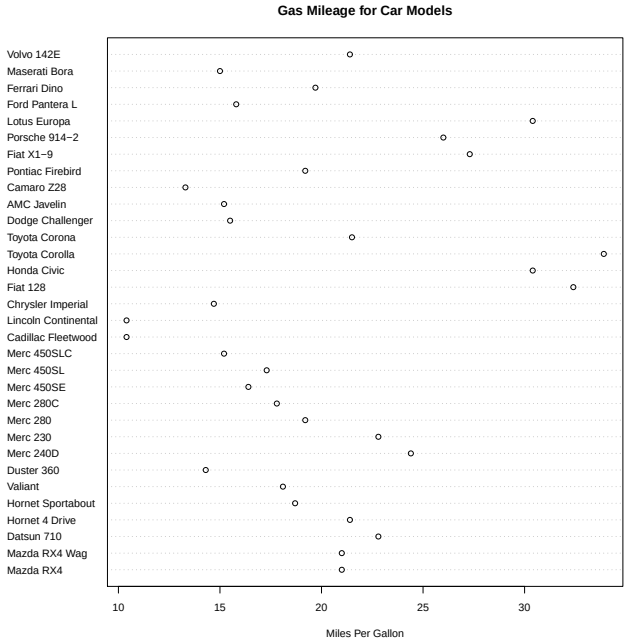
```
> library(lattice)
> levelplot(cars.cor, pretty = T, col.regions = gray(0:100/100))
```



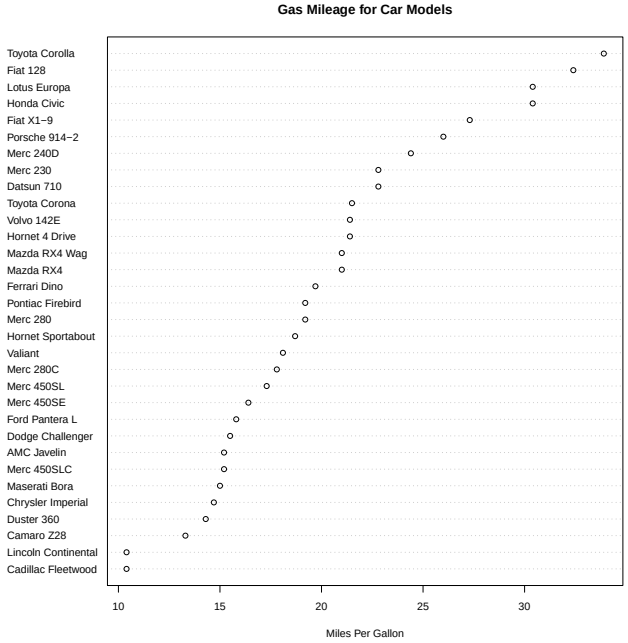
6.2 Dot plots

Dot plots are very efficient plots to present a variety of quantities, such as values on one variable over all observations of a dataset, or regression (see further below).

```
> data(mtcars)
> mtcars$carname <- row.names(mtcars)
> dotchart(mtcars$mpg, labels = mtcars$carname, cex = 0.7, main = "Gas
+     Mileage for Car Models", xlab = "Miles Per Gallon")
> mtcars.ord <- mtcars[order(mtcars$mpg), ]
> dotchart(mtcars.ord$mpg, labels = mtcars.ord$carname, cex = 0.7,
+     main = "Gas Mileage for Car Models", xlab = "Miles Per Gallon")
```

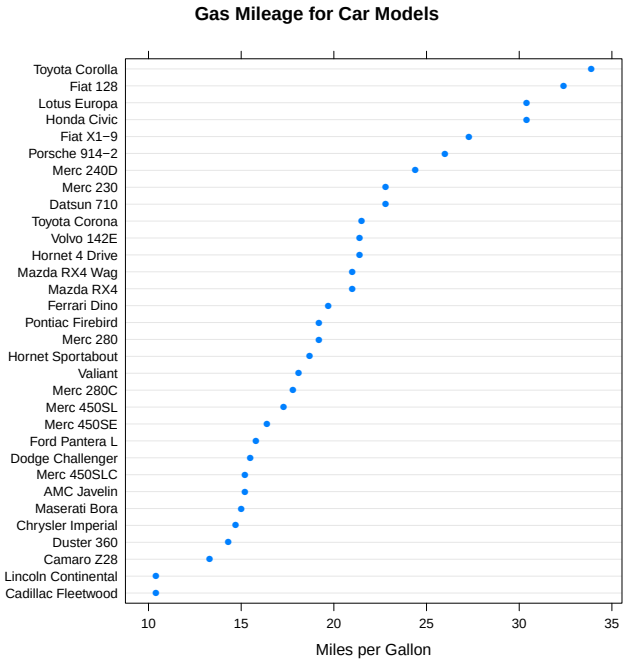


After sorting the data, the dot plot presents additional information about the shape of the distribution of the data:

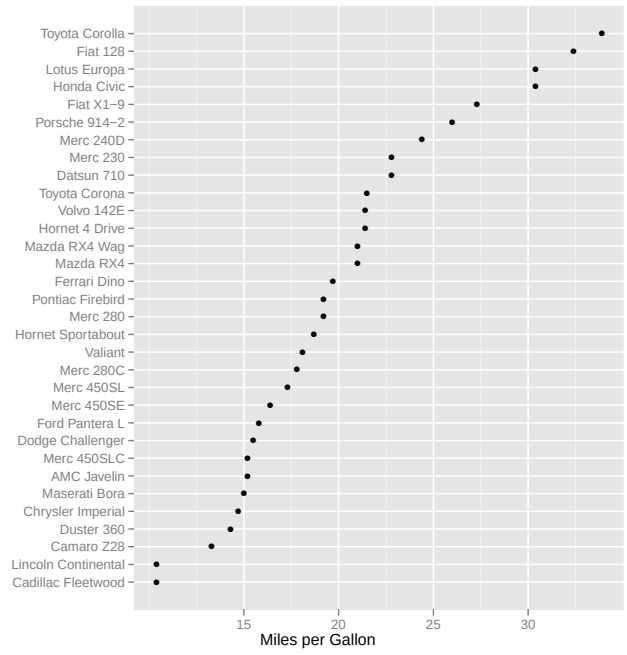


The lattice package has its own `dotplot` command. But note that it requires an extra step to sort the data (because the default would be to order the plot by the factor variable, which would be the car names):

```
> mtcars.ord$carname <- reorder(mtcars.ord$carname, mtcars.ord$mpg)
> dotplot(carname ~ mpg, data = mtcars.ord, xlab = "Miles per Gallon",
+     main = "Gas Mileage for Car Models")
```

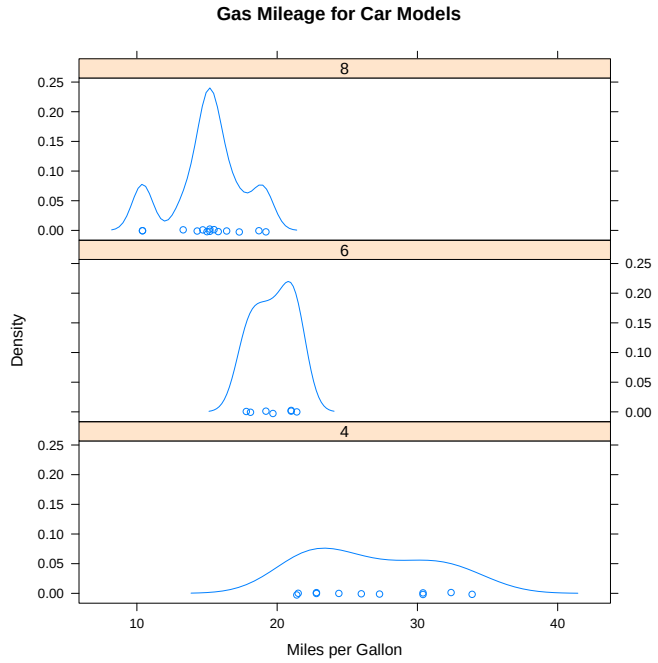



```
> library(ggplot2)
> ggdp <- ggplot(mtcars.ord, aes(mpg, carname)) + geom_point() + ylab("") + xlab("Miles per
> ggdp
```



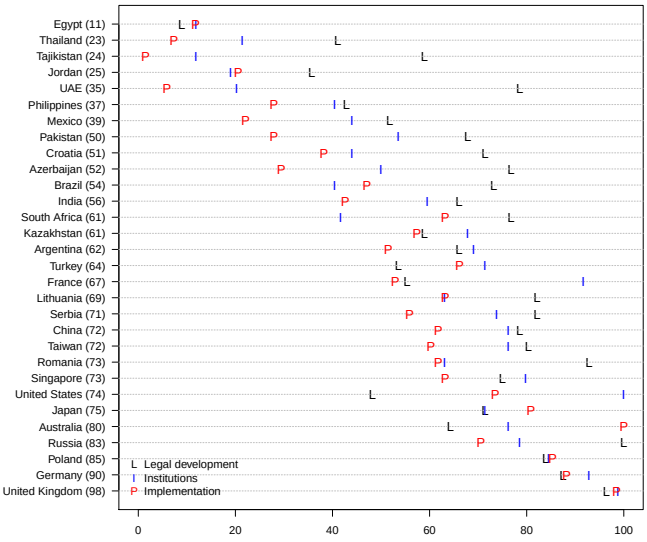
Or a density plot, with different panels for types:

```
> densityplot(~mpg | as.factor(cyl), data = mtcars.ord, xlab = "Miles per Gallon",
+     main = "Gas Mileage for Car Models", layout = c(1, 3))
```



Dot plots are also useful to present multidimensional data: here, I show countries' scores on three dimensions of a variable, using the base plot and some adjustments by hand. The advantage is that one can see how the relationship between the three variables changes – countries that generally score lower have higher values on L, whereas countries with higher average values have higher values on P and I:

```
> nonpro.dat <- read.csv("http://jkarreth.myweb.uga.edu/rstuff/nonpro.scores.csv")
> y.axis <- length(nonpro.dat$cnamept):1
> y2.axis <- length(nonpro.dat$nonproint):1
> layout(matrix(c(2, 1), 1, 2), widths = c(0.5, 3))
> par(mar = c(2, 9, 0.5, 2), lheight = 0.8)
> plot(nonpro.dat$legal, y.axis, type = "p", axes = F, xlab = "", ylab = "", pch = "L",
+     cex = 1.2, xlim = c(0, 100), ylim = c(min(y.axis), max(y.axis)), main = "",
+     frame = TRUE)
> axis(1, at = seq(0, 100, by = 20), label = seq(0, 100, by = 20), cex.axis = 1)
> axis(2, at = y.axis, label = nonpro.dat$cnamept, las = 1, tick = T, cex.axis = 1)
> abline(h = y.axis, lty = 2, lwd = 0.5, col = "gray")
> points(nonpro.dat$inst, y.axis, pch = "I", cex = 1.2, col = "blue")
> points(nonpro.dat$imp, y.axis, pch = "P", cex = 1.2, col = "red")
> legend("bottomleft", c("Legal development", "Institutions", "Implementation"),
+     col = c("black", "blue", "red"), pch = c("L", "I", "P"), inset = 0.01, bty = "n")
```



7 Regression models

R offers very convenient ways of plotting information from regression outputs, with a minimal need for copying-and-pasting by hand. This makes reproducing our workflow much easier. Of course, one can also use elements of regression outputs for other purposes – calculating predicted probabilities, plotting coefficients for different groups, etc.

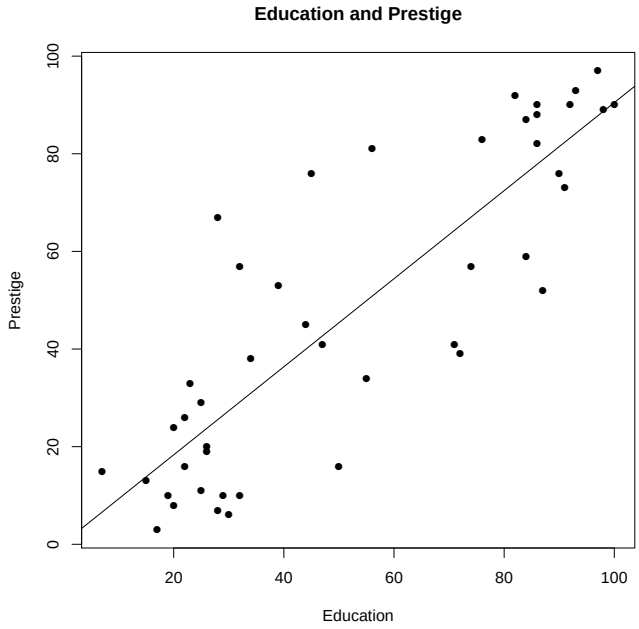
7.1 Linear models and bivariate relationships

Note the differences in how the three packages take the output from a linear regression model.

```
> Duncan <- read.table("http://jkarreth.myweb.uga.edu/rstuff/Duncan.txt")
> Duncan$typef <- as.factor(Duncan$type)
> D1.mod <- lm(prestige ~ education, data = Duncan)
```

7.1.1 Base plot

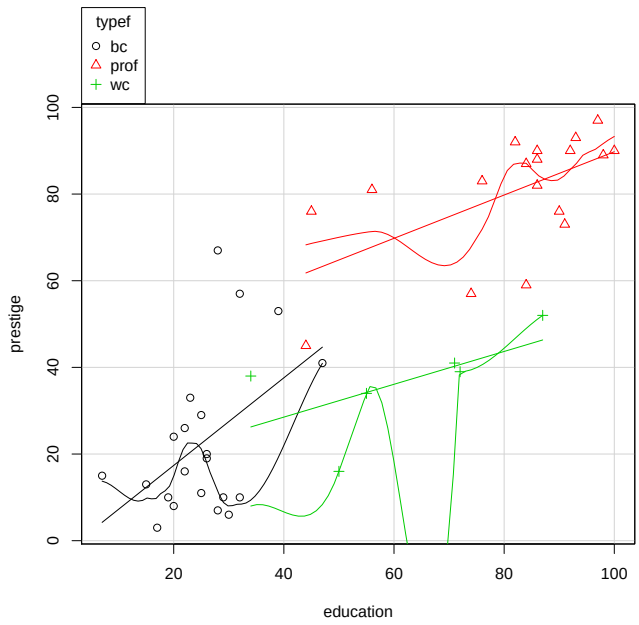
```
> par(mfrow = c(1, 1), mar = c(5, 4, 4, 2) + 0.1)
> plot(Duncan$education, Duncan$prestige, main = "Education and Prestige",
+      xlab = "Education", ylab = "Prestige", pch = 16)
> abline(D1.mod)
```



7.1.2 Lattice

In this example, I specify | `typef` to obtain separate plots for each value of the factor.

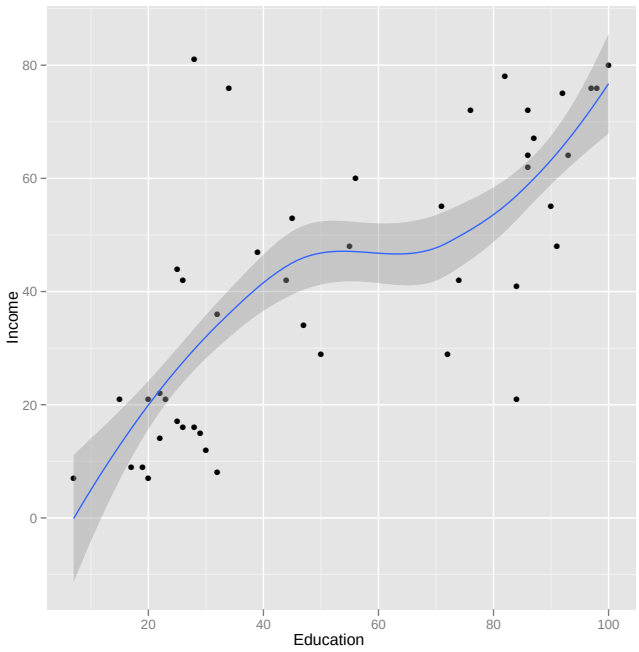
```
> library(lattice)
> scatterplot(prestige ~ education | typef, data = Duncan)
```



7.1.3 ggplot2

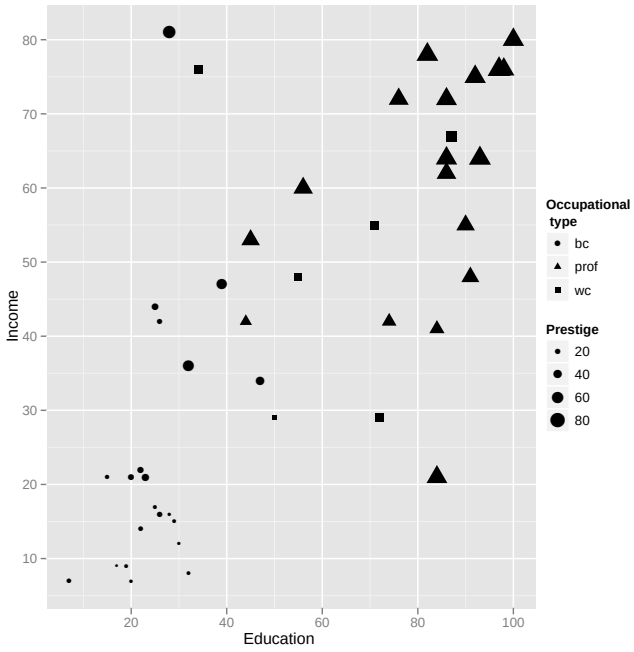
Bivariate scatterplot with a LOESS regression fit line:

```
> library(ggplot2)
> D.plot1 <- qplot(education, income, data = Duncan, geom = c("point", "smooth"),
+     method = loess, xlab = "Education", ylab = "Income")
> print(D.plot1)
```



Bivariate scatterplot with point sizes and icons conveying information about values of a factor (here, type) and a continuous variable (here, prestige):

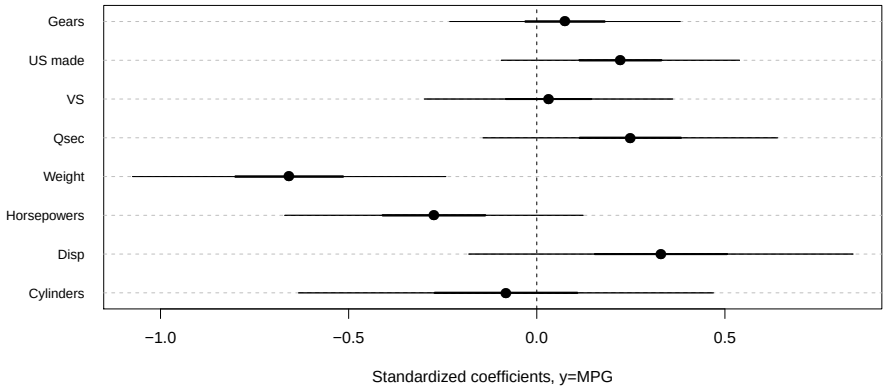
```
> D.plot2 <- qplot(education, income, data = Duncan, geom = c("point"),
+     shape = typef, size = prestige)
> D.plot2 <- D.plot2 + labs(x = "Education", y = "Income",
+     shape = "Occupational \n type", size = "Prestige")
> print(D.plot2)
```



7.2 Regression results: Dot plots

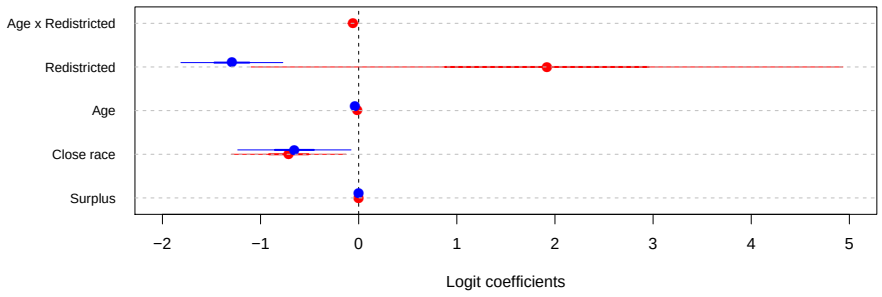
Simple OLS, using standardized coefficients:

```
> cars.std <- scale(mtcars[, 1:11])
> cars.std <- as.data.frame(cars.std)
> cars.mod <- lm(mpg ~ cyl + disp + hp + wt + qsec + vs + am + gear, data = cars.std)
> summary(cars.mod)
> coefplot2(cars.mod)
> coefplot2(cars.mod, varnames = c("Cylinders", "Disp", "Horsepowers", "Weight",
+     "Qsec", "VS", "US made", "Gears"), top.axis = F, xlab = "Standardized
+     coefficients, y=MPG", main = "", frame = T, cex.pts=1.4)
> abline(h=seq(1,8, by=1), lty=2, col="grey")
> coefplot2(cars.mod, varnames=c("Cylinders", "Disp", "Horsepowers", "Weight",
+     "Qsec", "VS", "US made", "Gears"), top.axis=F, xlab="Standardized coefficients,
+     y=MPG", main="", frame=T, cex.pts=1.4, add=T)
```



Logit (and comparing two models)

```
> library(foreign)
> logit.dat <- read.dta("http://jkarreth.myweb.uga.edu/rstuff/hw4.logit.dta")
> logit.dat$redistm <- as.factor(logit.dat$redistm)
> logit.dat$marginal <- as.factor(logit.dat$marginal)
> l.mod <- glm(returned ~ surplus + marginal + age + redistm, family = binomial,
+     data = logit.dat)
> l.mod$coefficients
> l.mod$residuals
> l.mod$fitted.values
> li.mod <- glm(returned ~ surplus + marginal + age * redistm, family = binomial,
+     data = logit.dat)
> coefplot2(li.mod)
> coefplot2(li.mod, varnames = c("Surplus", "Close race", "Age", "Redistricted",
+     "Age x Redistricted"), main = "", top.axis = F, intercept = FALSE, add = FALSE,
+     col.pts = "red", , cex.pts=1.4xlim = c(-2, 5), xlab = "Logit coefficients")
> coefplot2(l.mod, intercept = FALSE, add = TRUE, offset = 0.1,
+     col.pts = "blue", cex.pts=1.4)
```

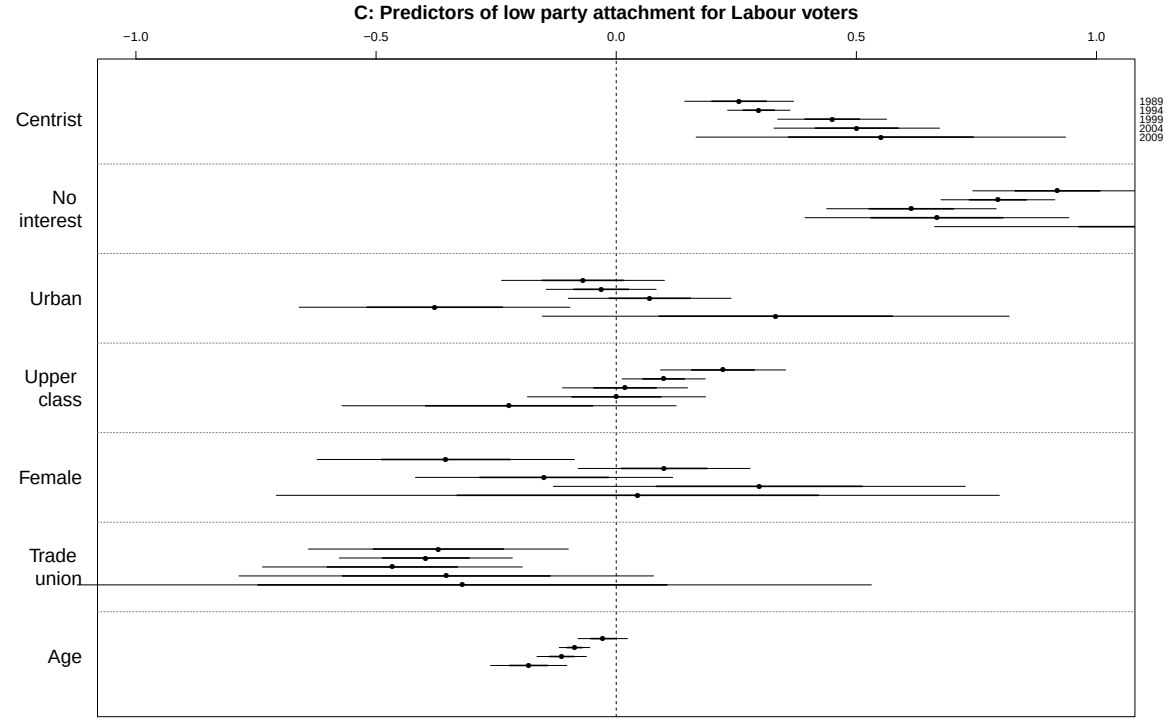


The dot plot also allows to compare several models, e.g., after subsetting. In this case, I estimated models for each of five waves of a survey, and presented separate coefficients on each variable for each

survey wave:

```
> EES <- read.dta("http://jkarreth.myweb.uga.edu/rstuff/EES84-09.SD3.dta")
> EES$t_var212 <- as.factor(EES$t_var212)
> library(gtools)
> EES$agedec <- quantcut(EES$age, seq(0, 1, by = 0.1))
> EES$agedn <- as.numeric(EES$agedec)
> EESlab <- EES[EES$t_var113 == 51320, ]
> EESlab1989 <- EES[EESlab$t_ees == 1989, ]
> EESlab1994 <- EES[EESlab$t_ees == 1994, ]
> EESlab1999 <- EES[EESlab$t_ees == 1999, ]
> EESlab2004 <- EES[EESlab$t_ees == 2004, ]
> EESlab2009 <- EES[EESlab$t_ees == 2009, ]
> library(MASS)
> labm89 <- polr(t_var212 ~ agedn + tradeunion + female + upperclass + urban +
+     nointerest + lrsd, data = EESlab1989, method = c("logistic"))
> labm94 <- polr(t_var212 ~ agedn + tradeunion + female + upperclass + urban +
+     nointerest + lrsd, data = EESlab1994, method = c("logistic"))
> labm99 <- polr(t_var212 ~ agedn + tradeunion + female + upperclass + urban +
+     nointerest + lrsd, data = EESlab1999, method = c("logistic"))
> labm04 <- polr(t_var212 ~ agedn + tradeunion + female + upperclass + urban +
+     nointerest + lrsd, data = EESlab2004, method = c("logistic"))
> labm09 <- polr(t_var212 ~ agedn + tradeunion + female + upperclass + urban +
+     nointerest + lrsd, data = EESlab2009, method = c("logistic"))
> labm89sum <- summary(labm89)
> labm89c <- labm89sum$coefficients[1:7, 1]
> labm89se <- labm89sum$coefficients[1:7, 2]
> labm94sum <- summary(labm94)
> labm94c <- labm94sum$coefficients[1:7, 1]
> labm94se <- labm94sum$coefficients[1:7, 2]
> labm99sum <- summary(labm99)
> labm99c <- labm99sum$coefficients[1:7, 1]
> labm99se <- labm99sum$coefficients[1:7, 2]
> labm04sum <- summary(labm04)
> labm04c <- labm04sum$coefficients[1:7, 1]
> labm04se <- labm04sum$coefficients[1:7, 2]
> labm09sum <- summary(labm09)
> labm09c <- labm09sum$coefficients[1:7, 1]
> labm09se <- labm09sum$coefficients[1:7, 2]
> coefplot2(labm99c, labm99se, xlim = c(-1, 1), ylim = c(0.6, 7.4), main = "",
+     varnames = c("Age", "Trade \nunion", "Female", "Upper \nclass", "Urban",
+         "No \ninterest", "Centrist"), cex.var = 1, frame = TRUE)
> title(main = list("C: Predictors of low party attachment for Labour voters",
+     cex = 1.2))
> coefplot2(labm94c, labm94se, xlim = c(-1, 1), add = TRUE, offset = 0.1)
> coefplot2(labm89c, labm99se, xlim = c(-1, 1), add = TRUE, offset = 0.2)
> coefplot2(labm04c, labm04se, xlim = c(-1, 1), add = TRUE, offset = -0.1)
```

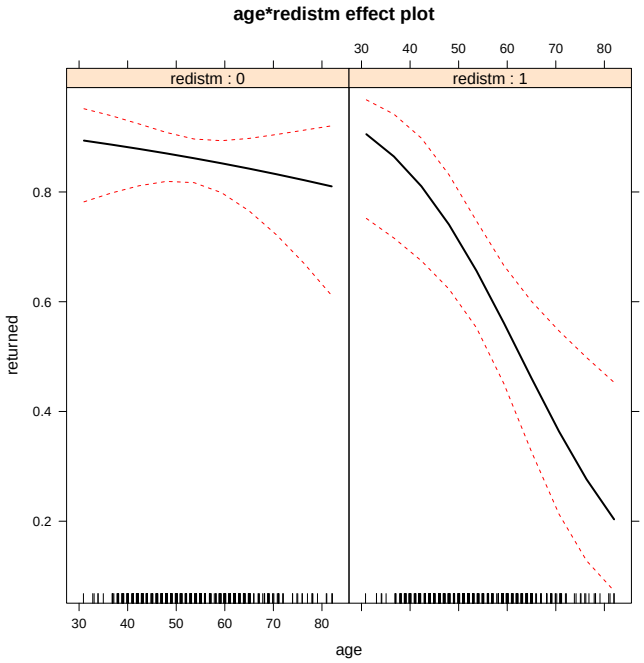
```
> coefplot2(labm09c, labm09se, xlim = c(-1, 1), add = TRUE, offset = -0.2)
> abline(h = 1.5, lty = 2, lwd = 0.5, col = "gray48")
> abline(h = 2.5, lty = 2, lwd = 0.5, col = "gray48")
> abline(h = 3.5, lty = 2, lwd = 0.5, col = "gray48")
> abline(h = 4.5, lty = 2, lwd = 0.5, col = "gray48")
> abline(h = 5.5, lty = 2, lwd = 0.5, col = "gray48")
> abline(h = 6.5, lty = 2, lwd = 0.5, col = "gray48")
> axis(4, at = c(6.8, 6.9, 7, 7.1, 7.2), labels = c("2009", "2004", "1999", "1994",
+     "1989"), tick = FALSE, las = 1, cex.axis = 0.8, line = c(-0.7))
```



7.3 Regression results: Predicted probabilities

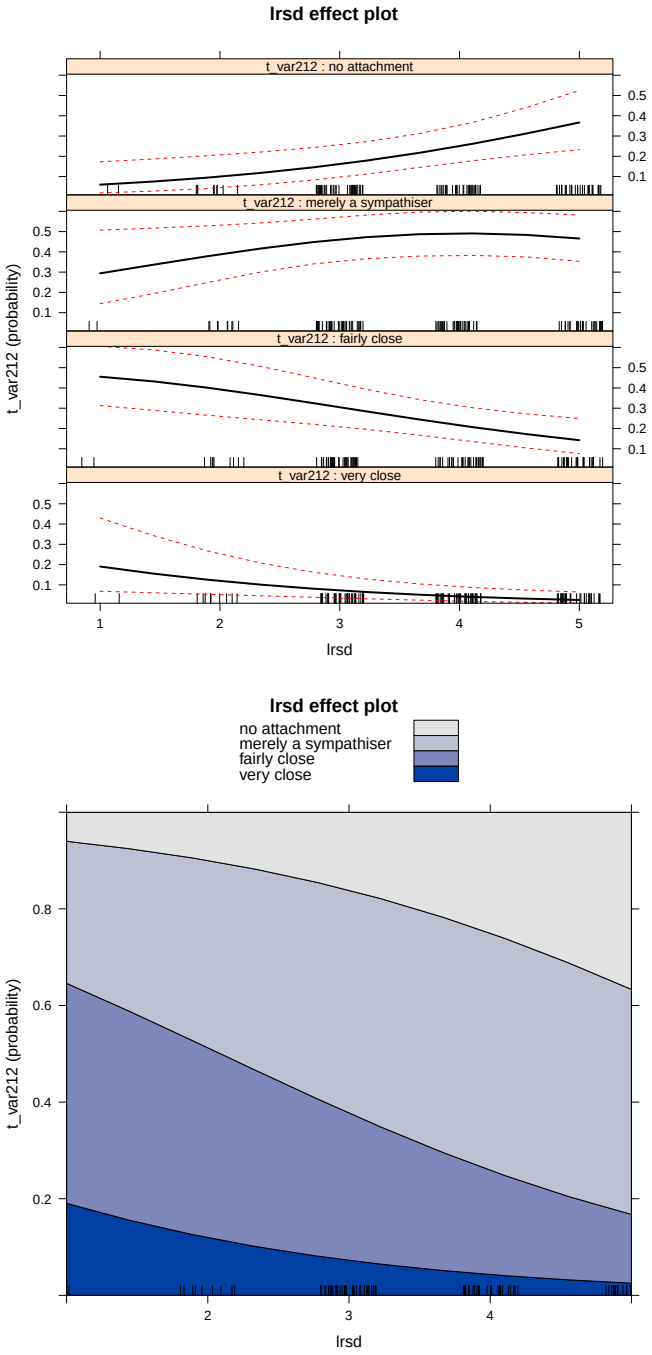
The `effects` package makes obtaining and plotting predicted probabilities easy:

```
> library(effects)
> plot(effect("age", l.mod), rescale.axis = F)
> plot(effect("age", l.mod, given.value = c(surplus = 0, marginal1 = 0,
+     redistm1 = 0)), rescale.axis = F)
> plot(effect("age", l.mod, given.value = c(surplus = 10000, marginal1 = 1,
+     redistm1 = 1)), rescale.axis = F)
> plot(effect("age*redistm", li.mod), rescale.axis = F)
```



After an ordered logit models, where predicted probabilites are even more helpful:

```
> labm09.eff <- effect("lrsd", labm09)
> labm09.eff$prob
> labm09.eff$lower.prob
> labm09.eff$upper.prob
> plot
> plot(effect("lrsd", labm09))
> plot(effect("lrsd", labm09), style = "stacked")
```



8 Maps

The rworldmap package provides an easy option to plot values of variables onto world maps. There are also many other options that are more customizable, for instance for plotting data on at U.S. county level. See http://dsparks.wordpress.com/2011/02/21 for an example...

```
> library(rworldmap)
> library(countrycode)
```

```
> igo.dat <- read.dta("http://jkarreth.myweb.uga.edu/rstuff/igo.cy.dta")
> cow <- igo.dat$ccode
> igo.dat$iso3c <- countrycode(cow, "cown", "iso3c")
> igo.dat[1:10, ]
> igo.dat.1985 <- igo.dat[igo.dat$year == 1985, ]
> sPDF <- joinCountryData2Map(igo.dat.1985, joinCode = "ISO3", nameJoinColumn = "iso3c")
> sPDF <- joinCountryData2Map(igo.dat.1985, joinCode = "NAME", nameJoinColumn = "country")
> mapCountryData(sPDF, nameColumnToPlot = "gle_gdp")
> mapCountryData(sPDF, nameColumnToPlot = "igo3_use", colourPalette = "white2Black",
+    catMethod = "pretty", mapTitle = "Memberships in HSIGOs, 1995",
+    oceanCol = "white", missingCountryCol = "white")
```