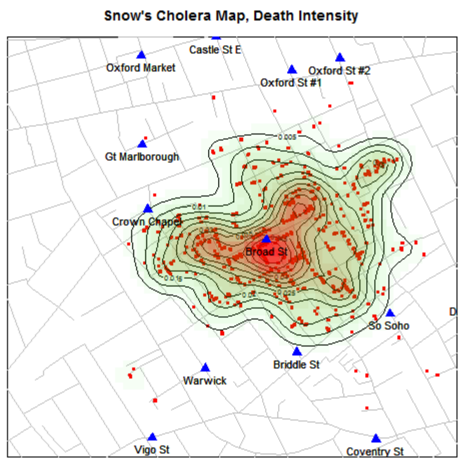
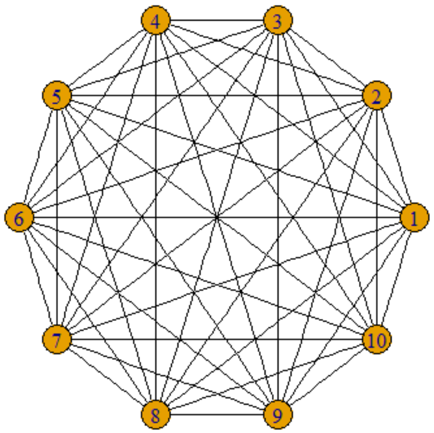
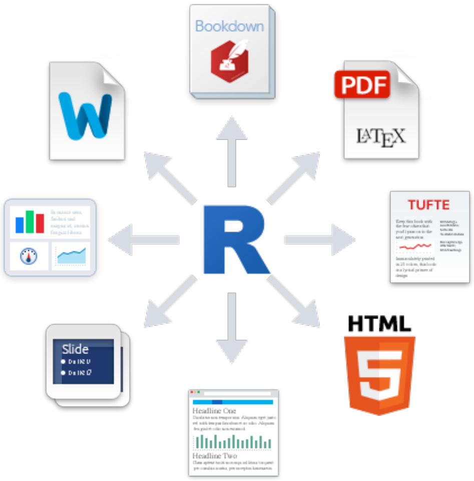






Data Visualization in R

1. Overview

Michael Friendly
SCS Short Course
Sep/Oct, 2018

http://datavis.ca/courses/RGraphics/

Statistical Consulting Service

YORK UNIVERSITÉ UNIVERSITY

Course outline

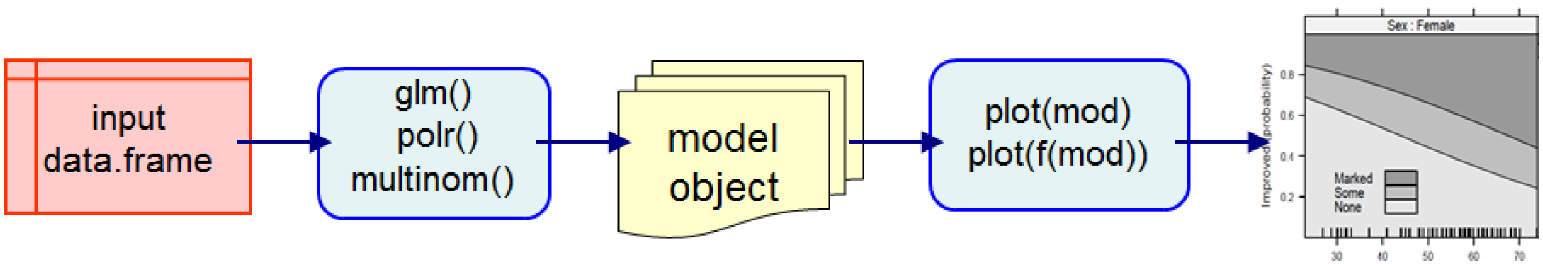
1. Overview of R graphics
2. Standard graphics in R
3. ~~Grid & lattice graphics~~
4. ggplot2

Outline: Session 1

- Session 1: Overview of R graphics, the big picture
  - Getting started: R, R Studio, R package tools
  - Roles of graphics in data analysis
    - Exploration, analysis, presentation
  - What can I do with R graphics?
    - Anything you can think of!
    - Standard data graphs, maps, dynamic, interactive graphics – we'll see a sampler of these
    - R packages: many application-specific graphs
  - Reproducible analysis and reporting
    - knitr, R markdown
    - R Studio

Outline: Session 2

- Session 2: Standard graphics in R
  - R object-oriented design



  - Tweaking graphs: control graphic parameters
    - Colors, point symbols, line styles
    - Labels and titles
  - Annotating graphs
    - Add fitted lines, confidence envelopes

Outline: Session 3

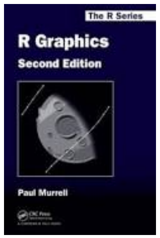
- Session 3: Grid & lattice graphics
  - Another, more powerful “graphics engine”
  - All standard plots, with more pleasing defaults
  - Easily compose collections (“small multiples”) from subsets of data
  - vcd and vcdExtra packages: mosaic plots and others for categorical data

Lecture notes for this session are available on the web page

Outline: Session 4

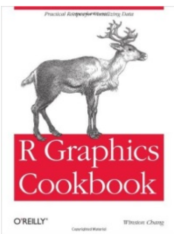
- Session 4: ggplot2
  - Most powerful approach to statistical graphs, based on the “Grammar of Graphics”
  - A graphics language, composed of layers, “geoms” (points, lines, regions), each with graphical “aesthetics” (color, size, shape)
  - part of a workflow for “tidy” data manipulation and graphics

Resources: Books



Paul Murrell, *R Graphics*, 2nd Ed.

Covers everything: traditional (base) graphics, lattice, ggplot2, grid graphics, maps, network diagrams, …
R code for all figures: https://www.stat.auckland.ac.nz/~paul/RG2e/

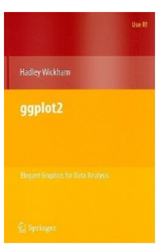


Winston Chang, *R Graphics Cookbook: Practical Recipes for Visualizing Data*

Cookbook format, covering common graphing tasks; the main focus is on ggplot2
R code from book: http://www.cookbook-r.com/Graphs/
Download from: http://ase.tufts.edu/bugs/guide/assets/R%20Graphics%20Cookbook.pdf

Deepayan Sarkar, *Lattice: Multivariate Visualization with R*

R code for all figures: http://lmdvr.r-forge.r-project.org/

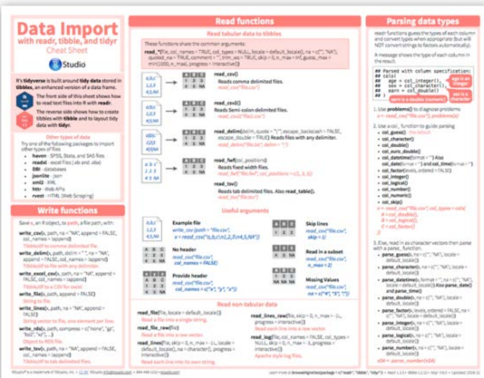


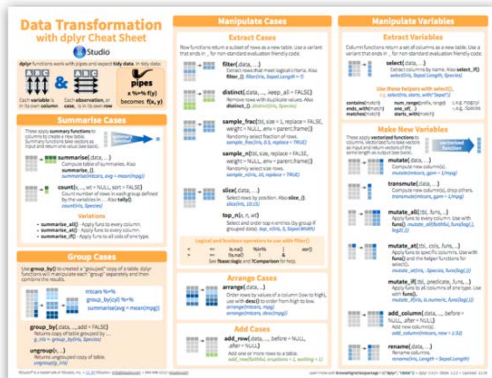
Hadley Wickham, *ggplot2: Elegant graphics for data analysis*, 2nd Ed.

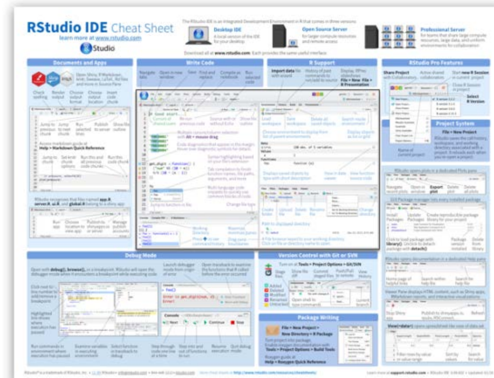
1st Ed: Online, http://ggplot2.org/book/
ggplot2 Quick Reference: http://sape.inf.usi.ch/quick-reference/ggplot2/
Complete ggplot2 documentation: http://docs.ggplot2.org/current/

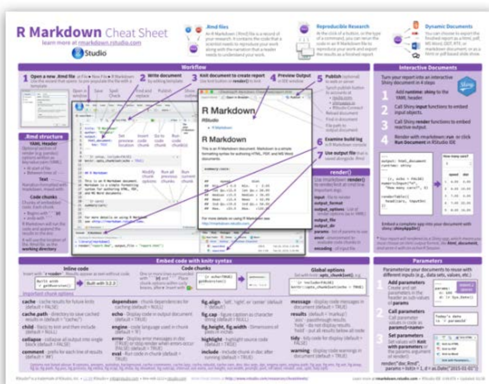
Resources: cheat sheets

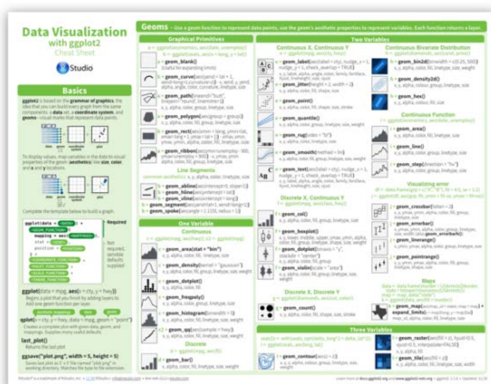
R Studio provides a variety of handy cheat sheets for aspects of data analysis & graphics See: https://www.rstudio.com/resources/cheatsheets/











Download, laminate, paste them on your fridge

Getting started: Tools

- To profit best from this course, you need to install both R and R Studio on your computer

The basic R system: R console (GUI) & packages
Download: http://cran.us.r-project.org/
Add my recommended packages:
source("http://datavis.ca/courses/RGraphics/R/install-pkgs.R")

The R Studio IDE: analyze, write, publish
Download:
https://www.rstudio.com/products/rstudio/download/
Add: R Studio-related packages, as useful

R package tools

Data prep: Tidy data makes analysis and graphing much easier.
Packages: tidyverse, comprised of: tidyr, dplyr, lubridate, …

The tidyverse

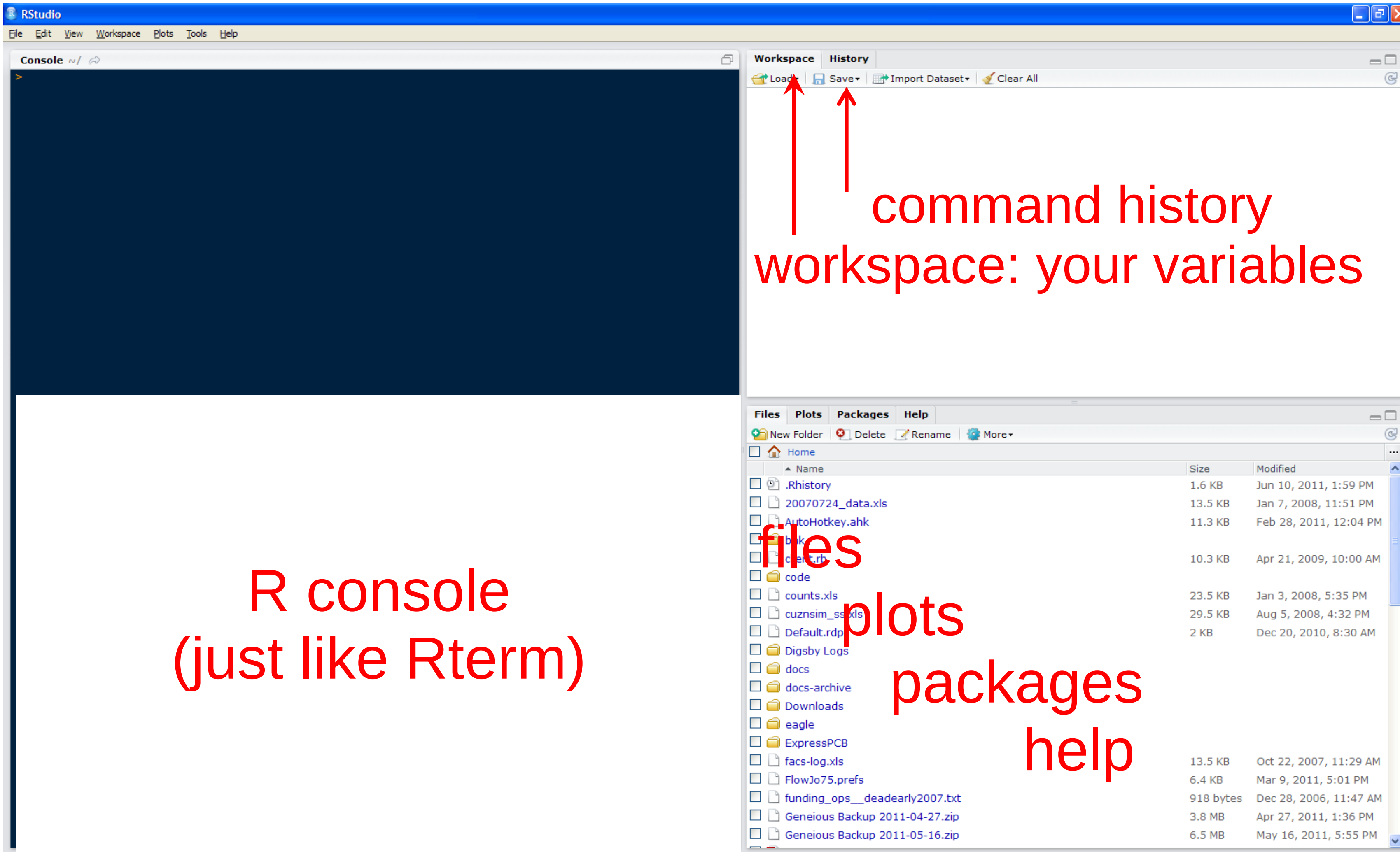
Components

R graphics: general frameworks for making standard and custom graphics
Graphics frameworks: base graphics, lattice, ggplot2, rgl (3D)
Application packages: car (linear models), vcd (categorical data analysis), heplots (multivariate linear models)

Publish: A variety of R packages make it easy to write and publish research reports and slide presentations in various formats (HTML, Word, LaTeX, …), all within R Studio

Web apps: R now has several powerful connections to preparing dynamic, web-based data display and analysis applications.

Getting started: R Studio



R Studio navigation

R folder navigation commands:

- Where am I?

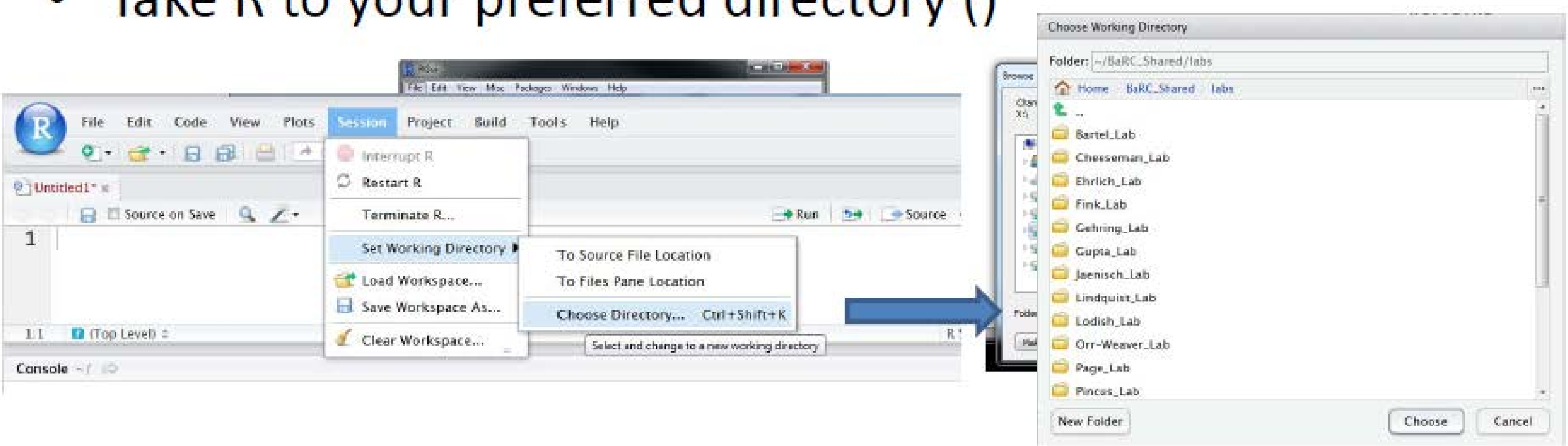
```
> getwd()
[1] "C:/Dropbox/Documents/6135"
```

- Go somewhere:

```
> setwd("C:/Dropbox")
> setwd(file.choose())
```

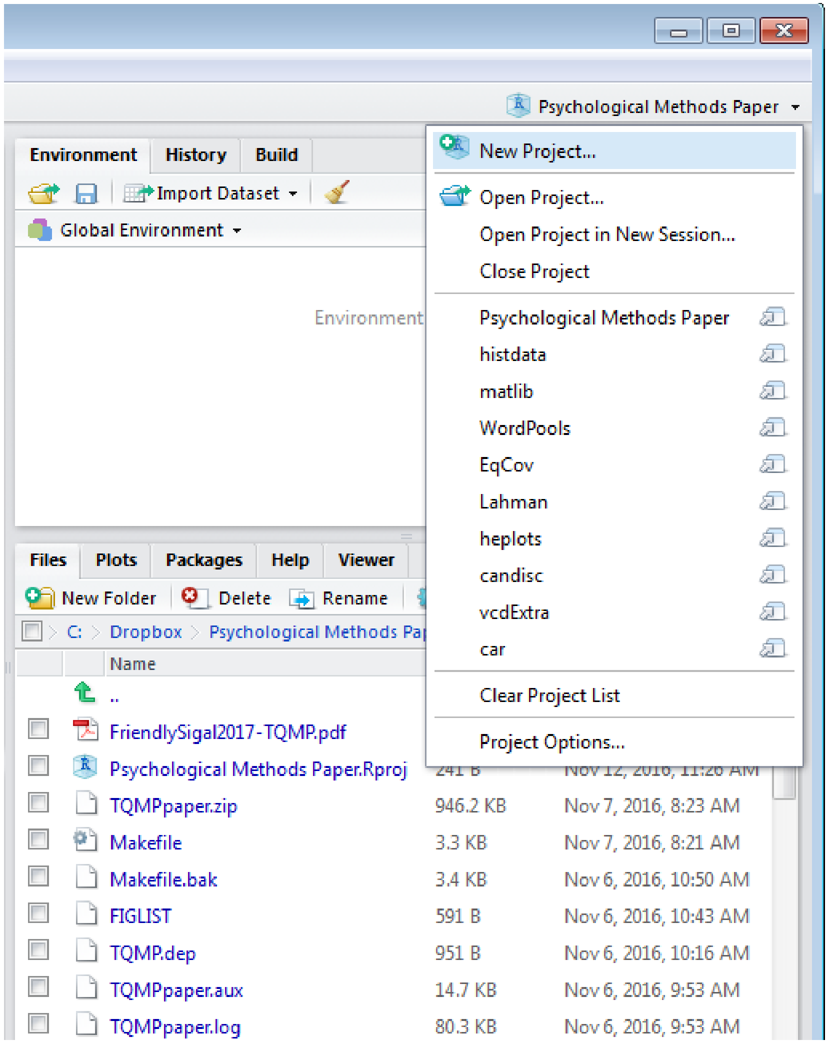
R Studio GUI

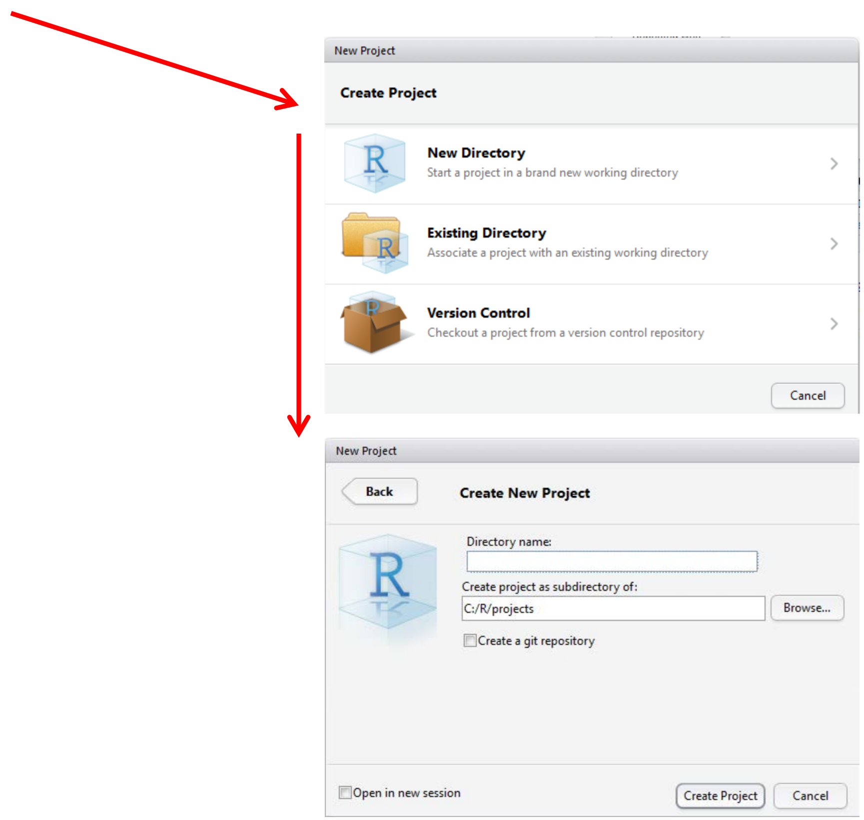
- Take R to your preferred directory ()



R Studio projects

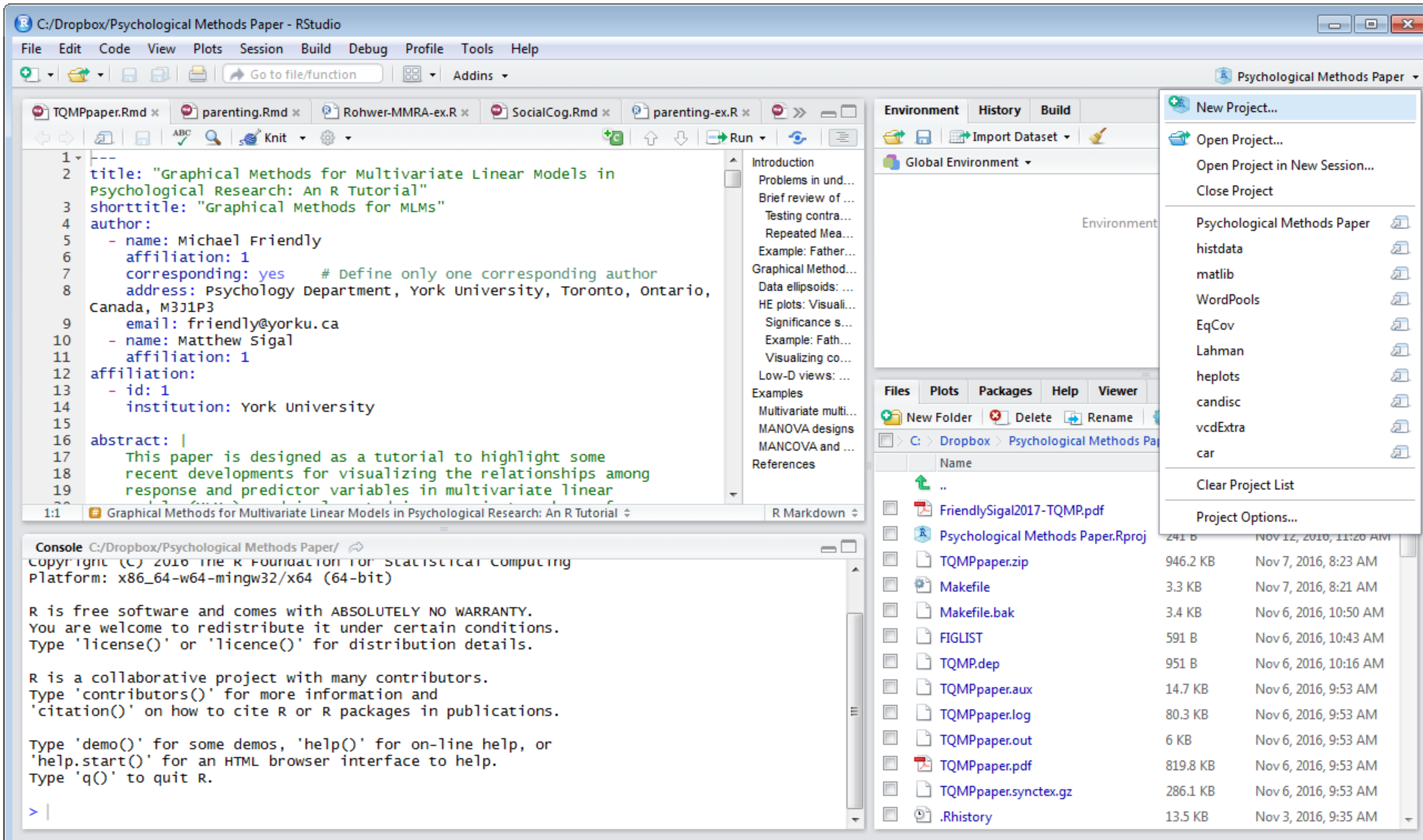
R Studio projects are a handy way to organize your work





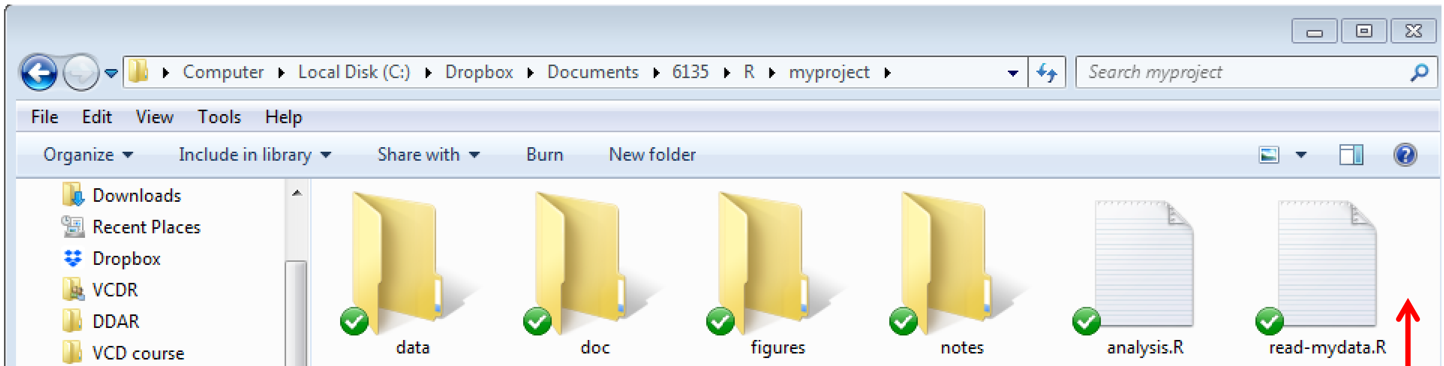
R Studio projects

An R Studio project for a research paper: R files (scripts), Rmd files (text, R “chunks”)



Organizing an R project

- Use a separate folder for each project
- Use sub-folders for various parts



data files:

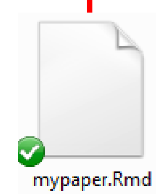
- raw data (.csv)
- saved R data (.Rdata)

figures:

- diagrams
- analysis plots

R files:

- data import
- analysis



Write up files will go here (.Rmd, .docx, .pdf)

Organizing an R project

- Use separate R files for different steps:
  - Data import, data cleaning, … → save as an RData file
  - Analysis: load RData, …

read-mydata.R

```
 # read the data; better yet: use RStudio File -> Import Dataset ...
mydata <- read.csv("data/mydata.csv")

# data cleaning ....

# save the current state
save("data/mydata.RData")
```

Organizing an R project

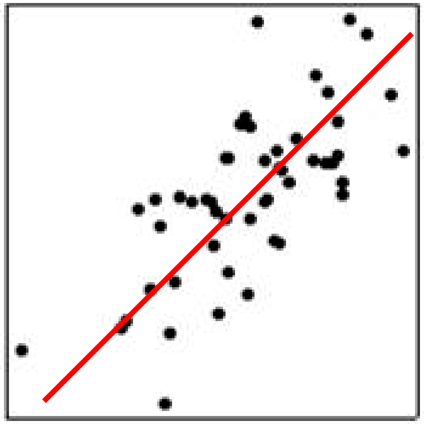
- Use separate R files for different steps:
  - Data import, data cleaning, … → save as an RData file
  - Analysis: load RData, …

analyse.R

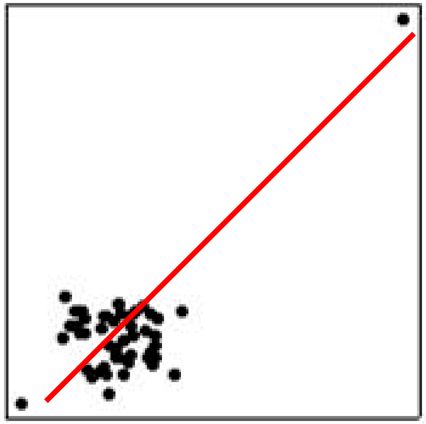
```
 # analysis
load("data/mydata.RData")

# do the analysis – exploratory plots
plot(mydata)

# fit models
mymod.1 <- lm(y ~ X1 + X2 + X3, data=mydata)

# plot models, extract model summaries
plot(mymod.1)
summary(mymod.1)
```

Graphics: Why plot your data?

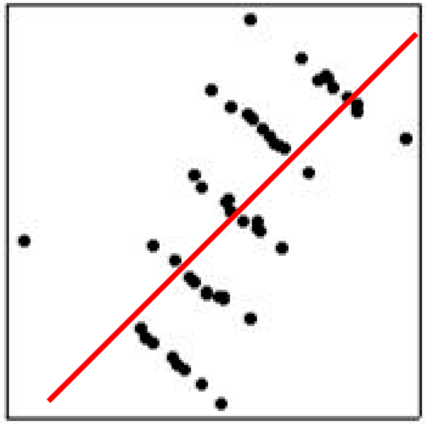
- Three data sets with exactly the same bivariate summary statistics:
  - Same correlations, linear regression lines, etc
  - Indistinguishable from standard printed output



Standard data



r=0 but + 2 outliers



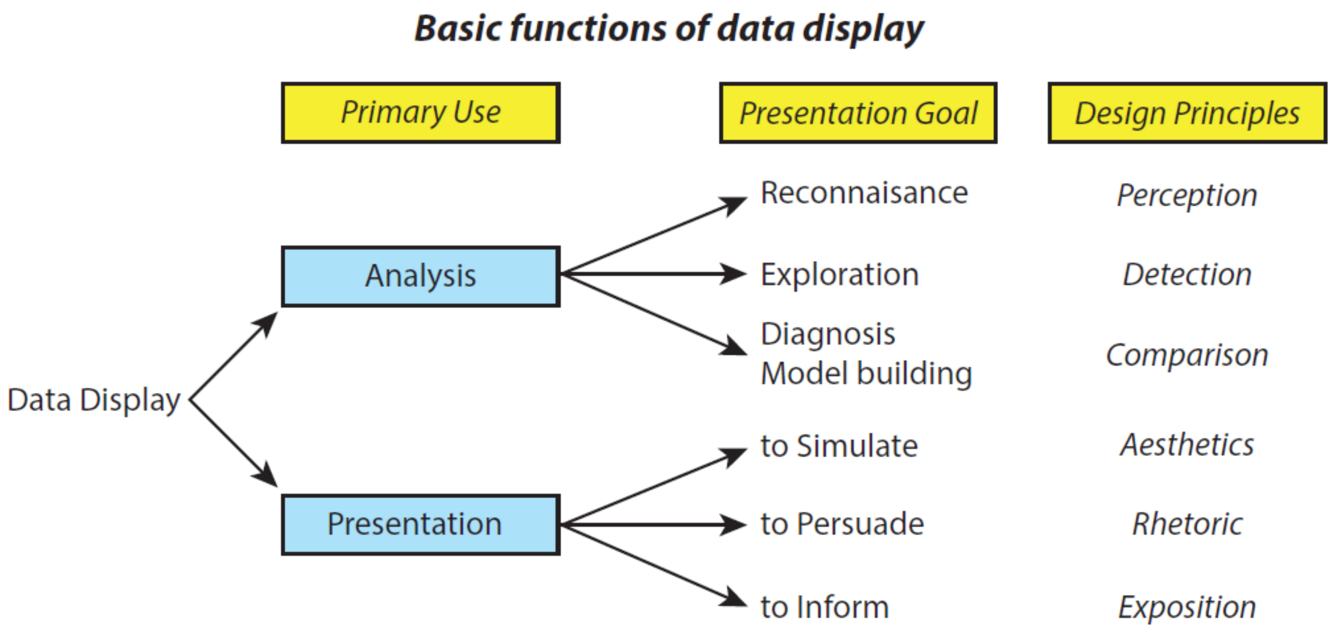
Lurking variable?

Roles of graphics in data analysis

- Graphs (& tables) are forms of communication:
  - What is the audience?
  - What is the message?

Analysis graphs: design to see patterns, trends, aid the process of data description, interpretation

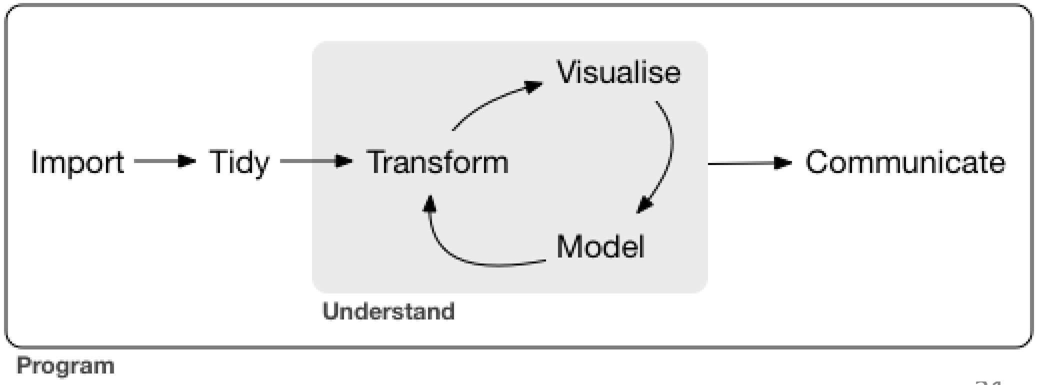
Presentation graphs: design to attract attention, make a point, illustrate a conclusion



The 80-20 rule: Data analysis

- Often ~80% of data analysis time is spent on data preparation and data cleaning
  1. data entry, importing data set to R, assigning factor labels,
  2. data screening: checking for errors, outliers, …
  3. Fitting models & diagnostics: whoops! Something wrong, go back to step 1
- Whatever you can do to reduce this, gives more time for:
  - Thoughtful analysis,
  - Comparing models,
  - Insightful graphics,
  - Telling the story of your results and conclusions

This view of data analysis, statistics and data vis is now rebranded as "data science"

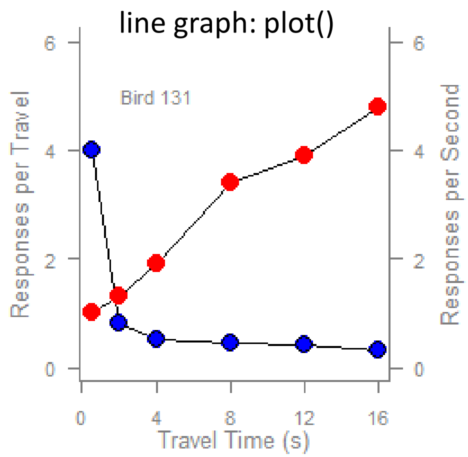


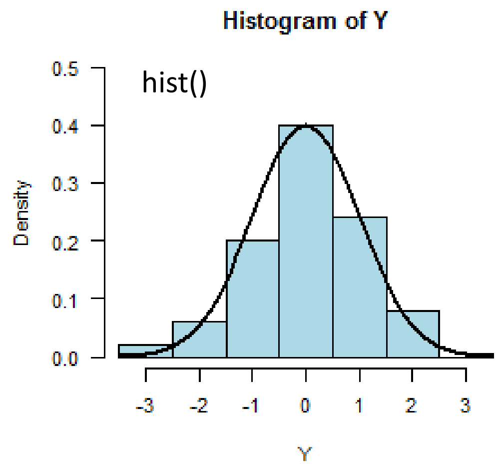
The 80-20 rule: Graphics

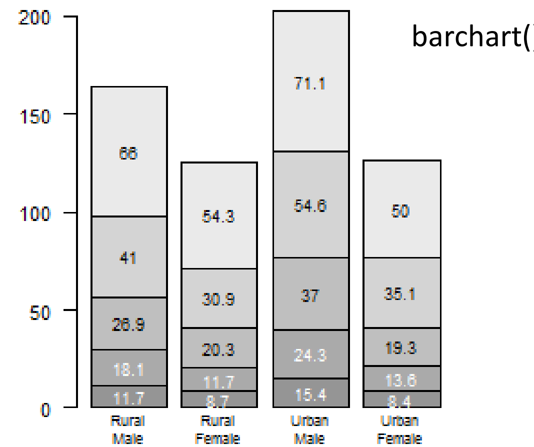
- **Analysis graphs**: Happily, 20% of effort can give 80% of a desired result
  - Default settings for plots often give something reasonable
  - 90-10 rule: Plot annotations (regression lines, smoothed curves, data ellipses, …) add additional information to help understand patterns, trends and unusual features, with only 10% more effort
- **Presentation graphs**: Sadly, 80% of total effort may be required to give the remaining 20% of your final graph
  - Graph title, axis and value labels: should be directly readable
  - Grouping attributes: visually distinct, allowing for BW vs color
    - color, shape, size of point symbols;
    - color, line style, line width of lines
  - Legends: Connect the data in the graph to interpretation
  - Aspect ratio: need to consider the H x V size and shape

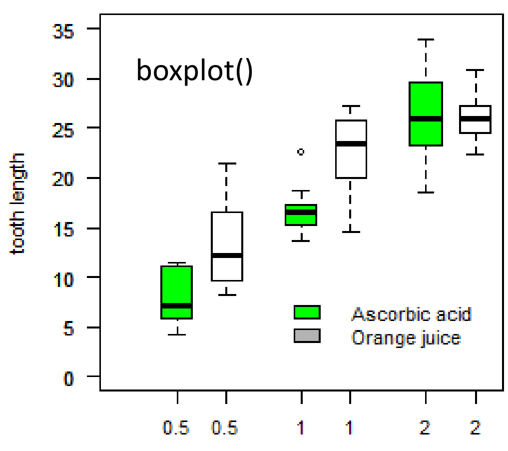
What can I do with R graphics?

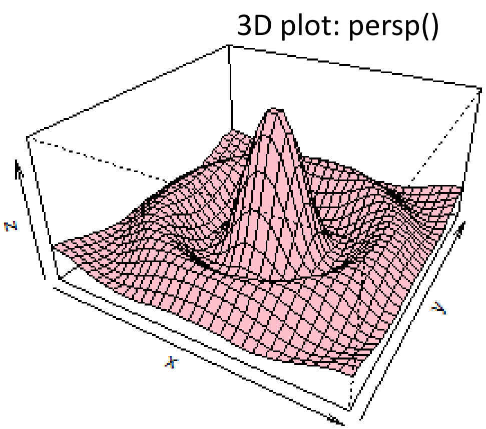
A wide variety of standard plots (customized)







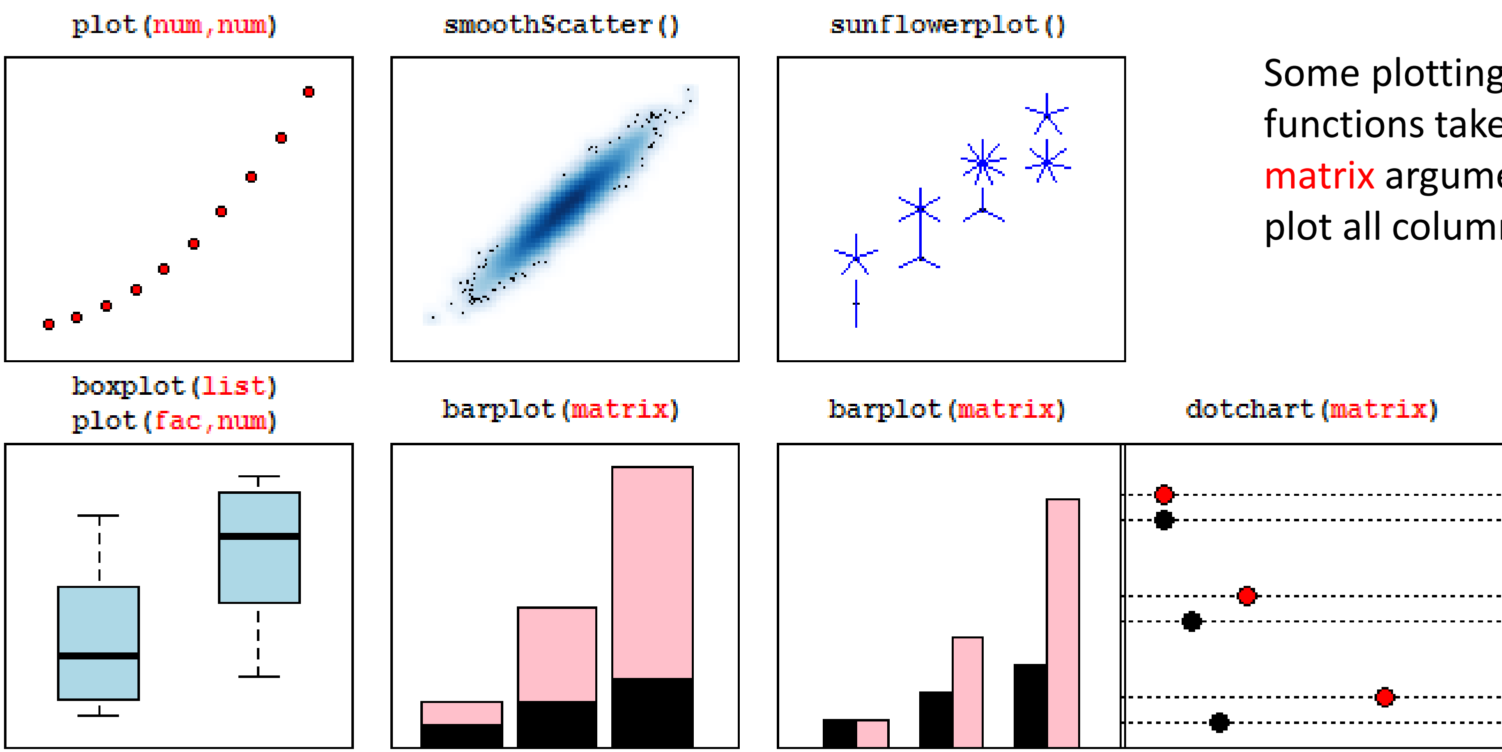




Bivariate plots

R base graphics provide a wide variety of different plot types for bivariate data

The function **`plot(x, y)`** is generic. It produces different kinds of plots depending on whether x and y are numeric or factors.

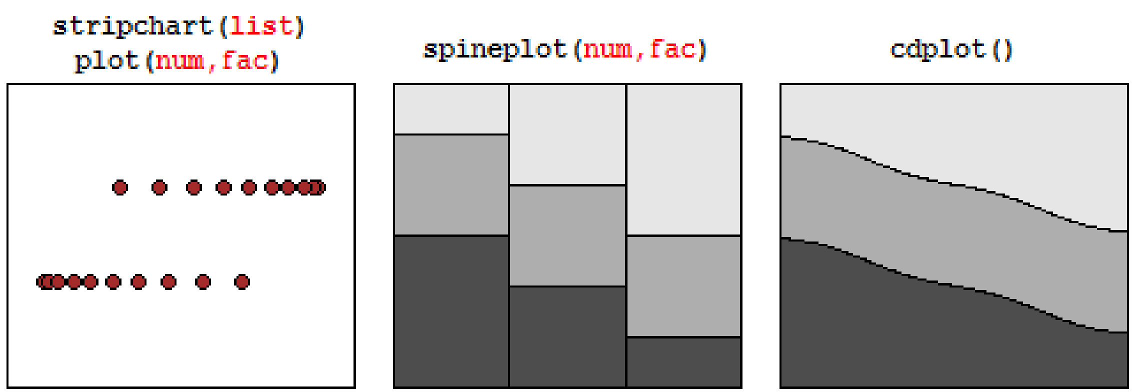


Some plotting functions take a matrix argument & plot all columns

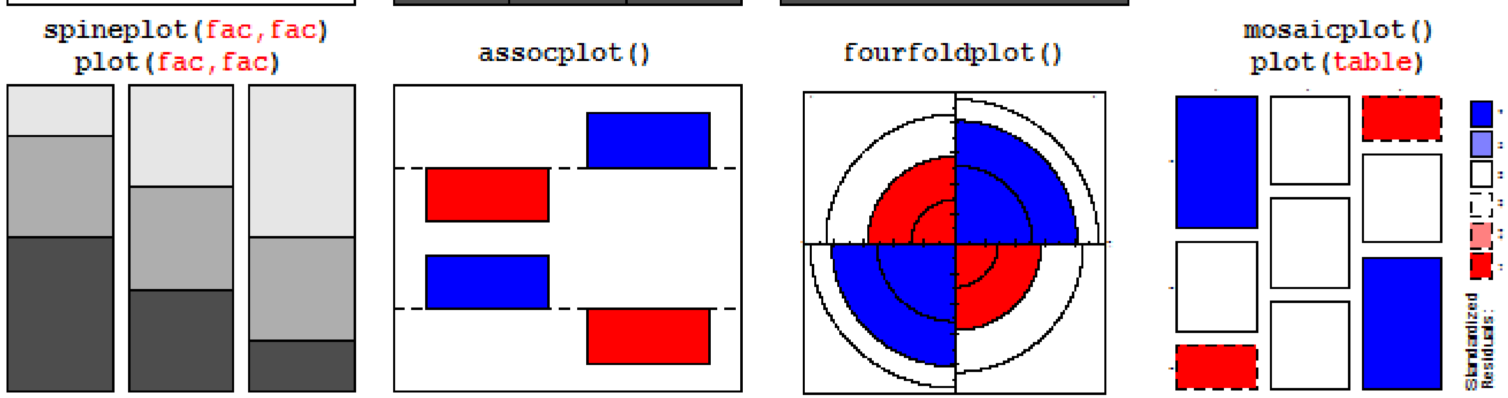
Bivariate plots

A number of specialized plot types are also available in base R graphics

Plot methods for factors and tables are designed to show the association between categorical variables



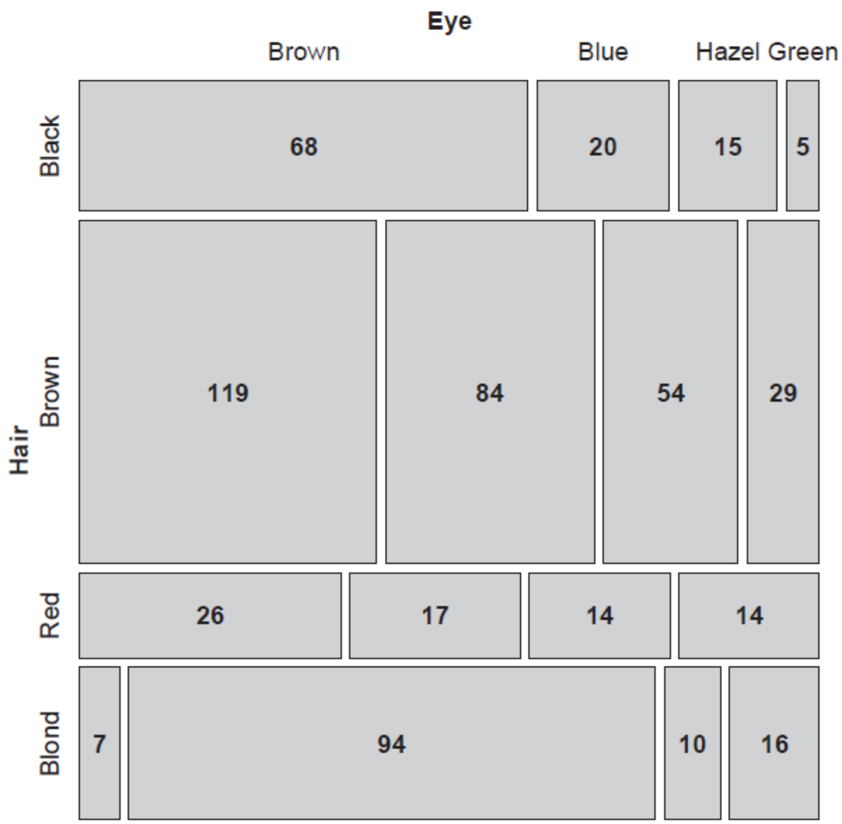
The vcd & vcdExtra packages provide more and better plots for categorical data



Mosaic plots

Similar to a grouped bar chart
Shows a frequency table with tiles,
area ~ frequency

```
> data(HairEyeColor)
> HEC <- margin.table(HairEyeColor, 1:2)
> HEC
       Eye
Hair     Brown Blue Hazel Green
  Black     68   20    15     5
  Brown    119   84    54    29
  Red       26   17    14    14
  Blond      7   94    10    16
> chisq.test(HEC)

        Pearson's Chi-squared test

data:  HEC
X-squared = 140, df = 9, p-value <2e-16
```



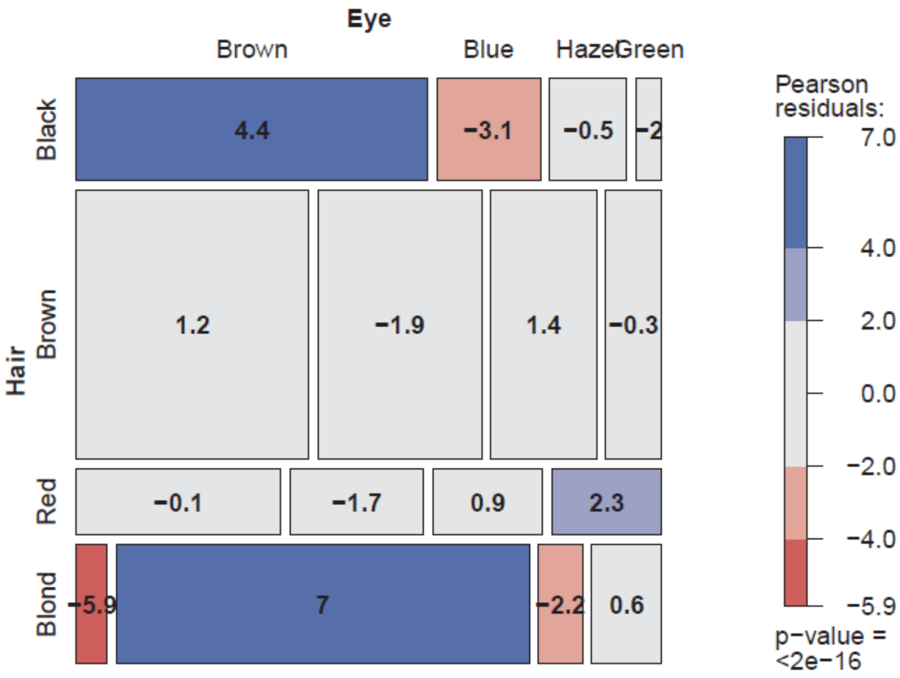
How to understand the association between hair color and eye color?

Mosaic plots

Shade each tile in relation to the contribution to the Pearson χ^2 statistic

$$\chi^2 = \sum r_{ij}^2 = \sum \frac{(o_{ij} - e_{ij})^2}{e_{ij}}$$

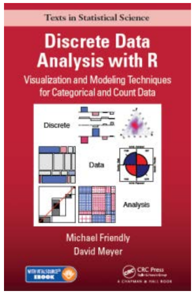
```
> round(residuals(chisq.test(HEC)),2)
       Eye
Hair     Brown  Blue Hazel Green
  Black   4.40 -3.07 -0.48 -1.95
  Brown   1.23 -1.95  1.35 -0.35
  Red    -0.07 -1.73  0.85  2.28
  Blond  -5.85  7.05 -2.23  0.61
```



Mosaic plots extend readily to 3-way + tables

They are intimately connected with loglinear models

See: Friendly & Meyer (2016), Discrete Data Analysis with R, http://ddar.datavis.ca/



Follow along

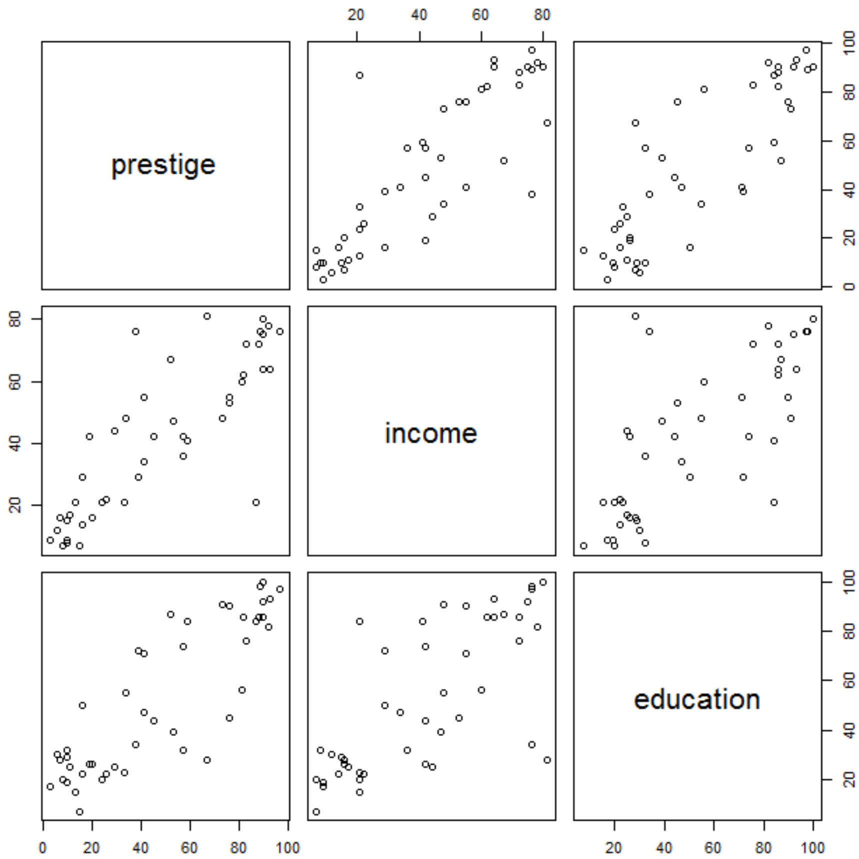
- From the course web page, click on the script duncan-plots.R, http://www.datavis.ca/courses/RGraphics/R/duncan-plots.R
- Select all (ctrl+A) and copy (ctrl+C) to the clipboard
- In R Studio, open a new R script file (ctrl+shift+N)
- Paste the contents (ctrl+V)
- Run the lines (ctrl+Enter) along with me

Multivariate plots

The simplest case of multivariate plots is a **scatterplot matrix** – all pairs of bivariate plots

In R, the generic functions **`plot()`** and **`pairs()`** have specific methods for data frames

```
data(Duncan, package="car")
plot(~ prestige + income + education,
        data=Duncan)
pairs(~ prestige + income + education,
        data=Duncan)
```

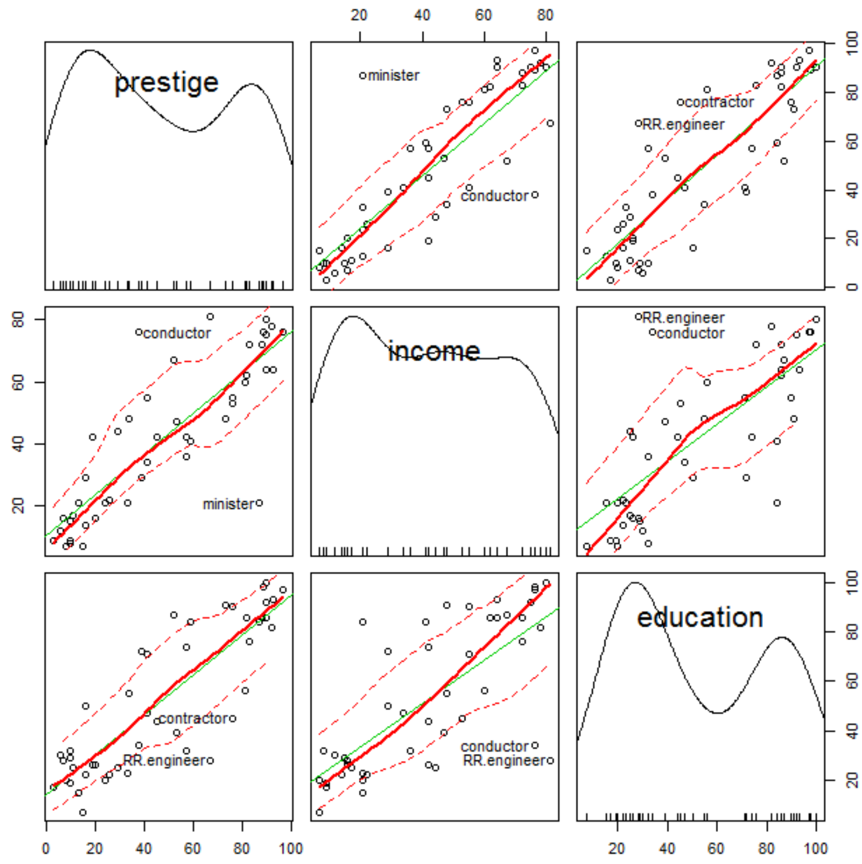


Multivariate plots

These basic plots can be enhanced in many ways to be more informative.

The function scatterplotMatrix() in the car package provides

- univariate plots for each variable
- linear regression lines and loess smoothed curves for each pair
- automatic labeling of noteworthy observations (id.n=)



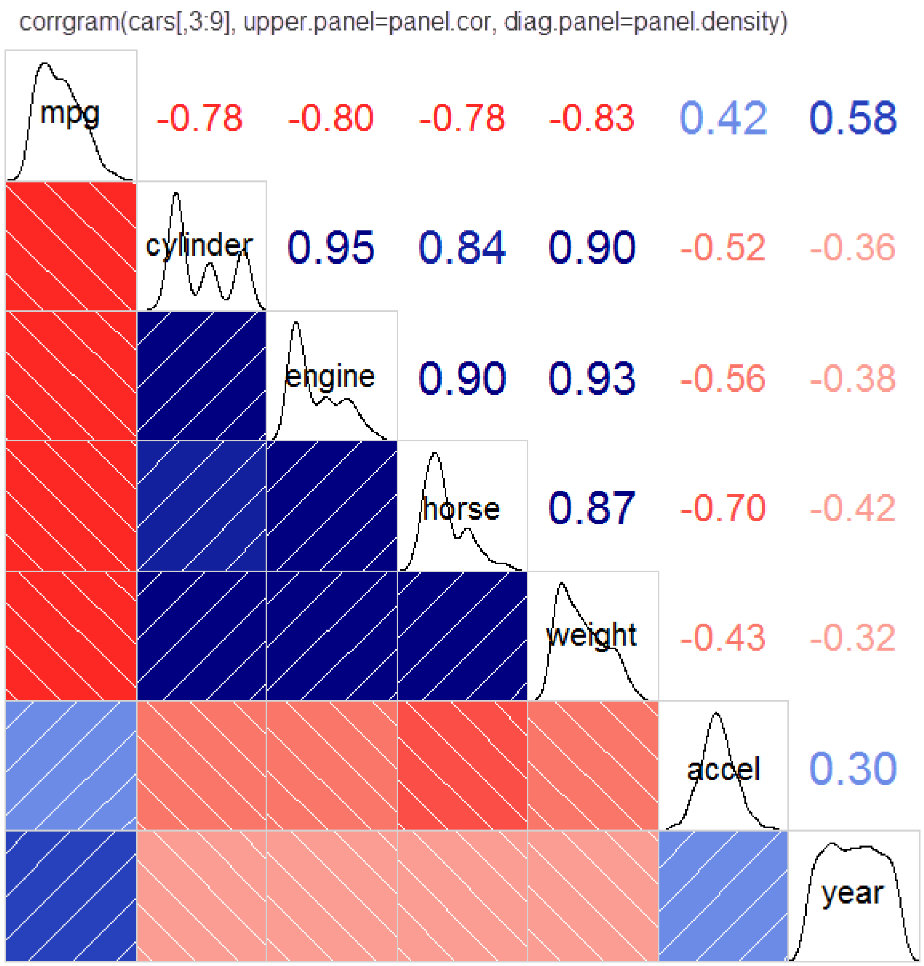

```
library(car)
scatterplotMatrix(~prestige + income + education,  data=Duncan, id.n=2)
```

Multivariate plots: corrgrams

For larger data sets, visual summaries are often more useful than direct plots of the raw data

A corrgram (“correlation diagram”) allows the data to be rendered in a variety of ways, specified by panel functions.

Here the main goal is to see how mpg is related to the other variables



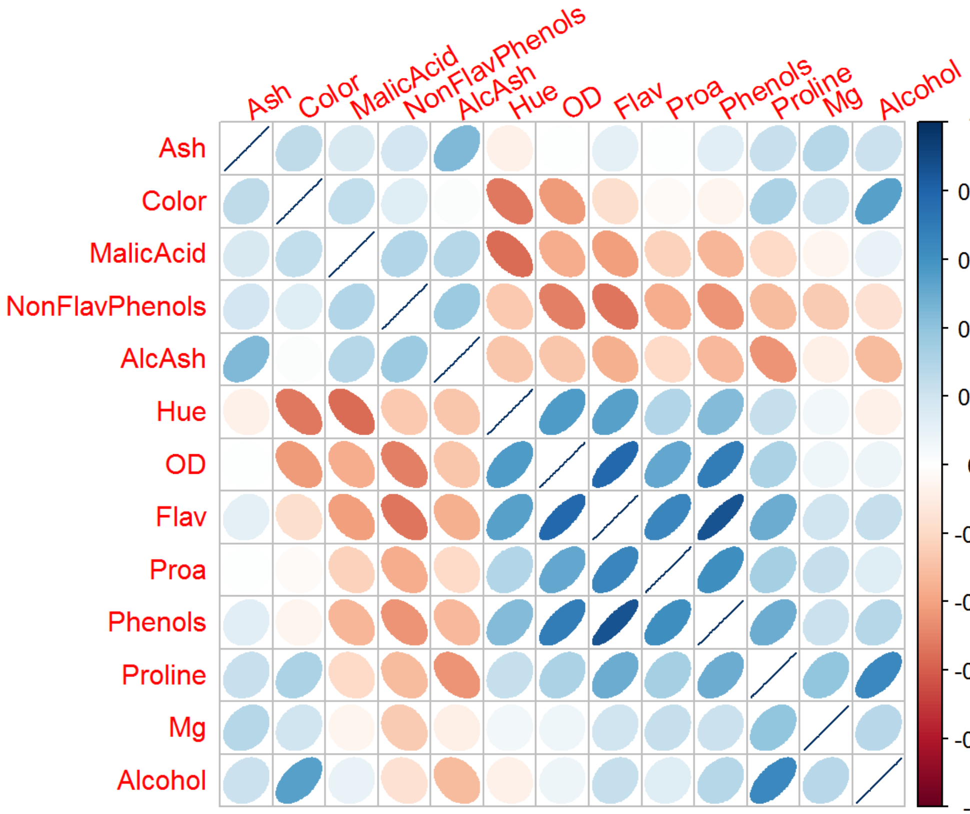
See: Friendly, M. Corrgrams: Exploratory displays for correlation matrices. *The American Statistician*, **2002**, *56*, 316-324

Multivariate plots: corrgrams

For even larger data sets, more abstract visual summaries are necessary to see the patterns of relationships.

This example uses schematic ellipses to show the strength and direction of correlations among variables on a large collection of Italian wines.

Here the main goal is to see how the variables are related to each other.



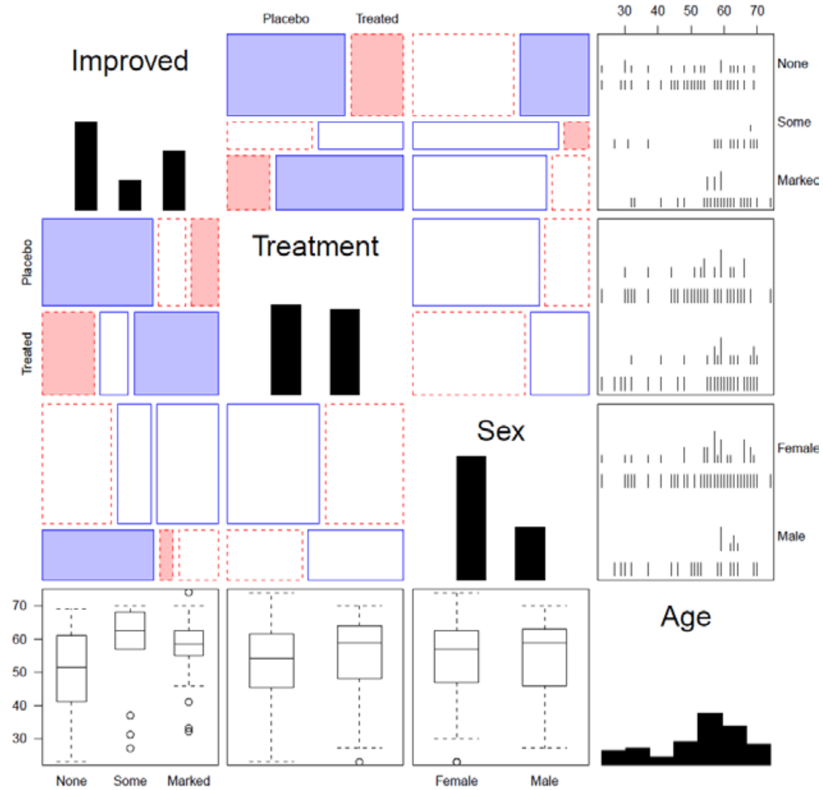

```
library(corrplot)
corrplot(cor(wine), tl.srt=30, method="ellipse", order="AOE")
```

See: Friendly, M. Corrgrams: Exploratory displays for correlation matrices. *The American Statistician,* **2002**, *56*, 316-324

Generalized pairs plots

Generalized pairs plots from the gpairs package handle both categorical (**C**) and quantitative (**Q**) variables in sensible ways

x	y	plot
Q	Q	scatterplot
C	Q	boxplot
Q	C	barcode
C	C	mosaic



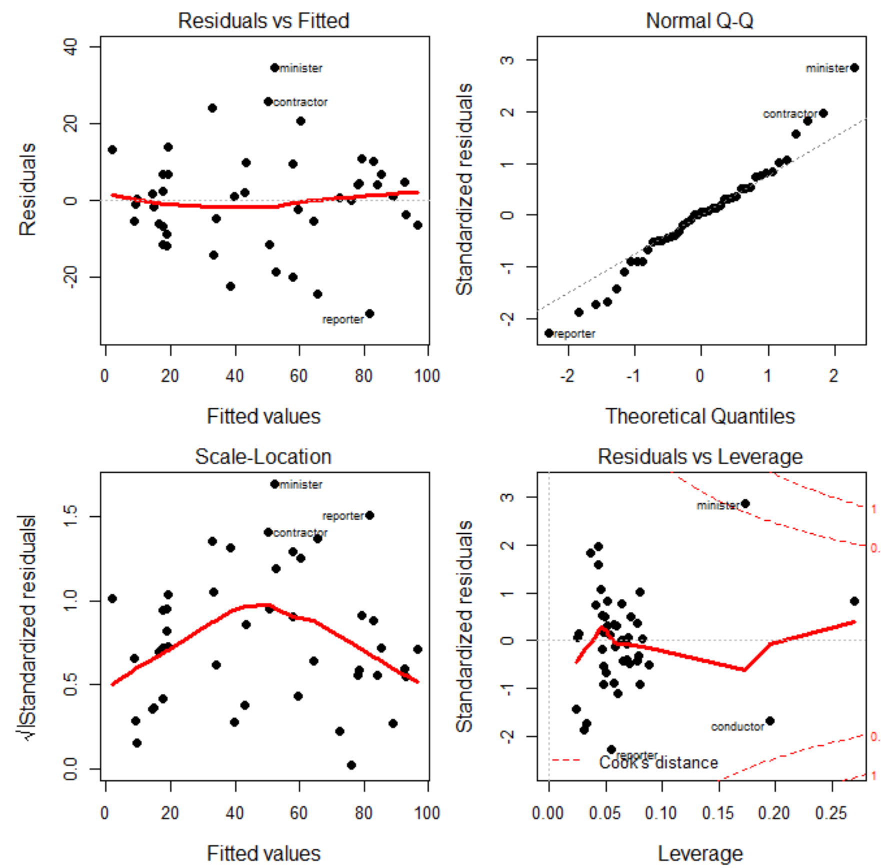

```
library(gpairs)
data(Arthritis)
gpairs(Arthritis[, c(5, 2:5)], …)
```

Models: diagnostic plots

Linear statistical models (ANOVA, regression), $\mathbf{y} = \mathbf{X}\,\boldsymbol{\beta} + \boldsymbol{\varepsilon}$, require some assumptions: $\boldsymbol{\varepsilon} \sim N(\mathbf{0}, \boldsymbol{\sigma}^2)$

For a fitted model object, the plot() method gives some useful diagnostic plots:

- residuals vs. fitted: any pattern?
- Normal QQ: are residuals normal?
- scale-location: constant variance?
- residual-leverage: outliers?

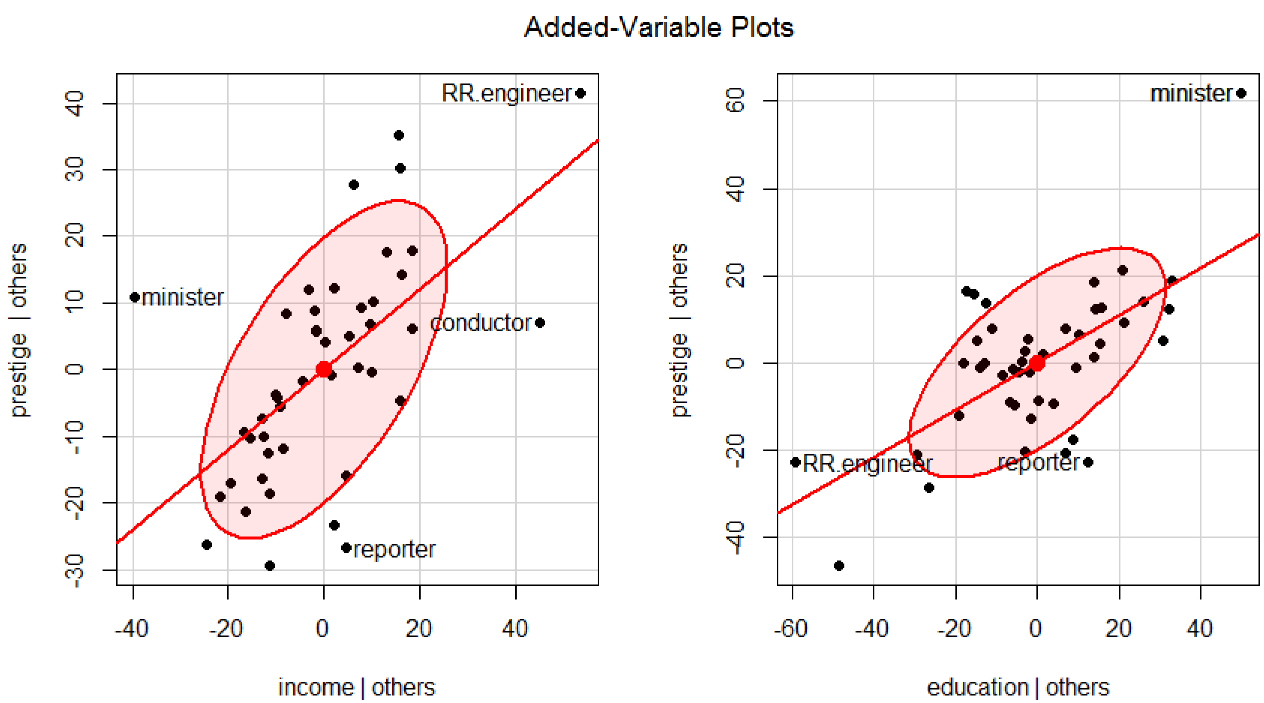



```
duncan.mod <- lm(prestige ~ income + education, data=Duncan)
plot(duncan.mod)
```

Models: Added variable plots

The car package has many more functions for plotting linear model objects
Among these, added variable plots show the partial relations of y to each x, holding all other predictors constant.

```
library(car)
avPlots(duncan.mod, id.n=2,ellipse=TRUE, …)
```



Each plot shows:
partial slope, β_j
influential obs.

Models: Interpretation

Fitted models are often difficult to interpret from tables of coefficients

```
# add term for type of job
duncan.mod1 <- update(duncan.mod, . ~ . + type)
summary(duncan.mod1)
```

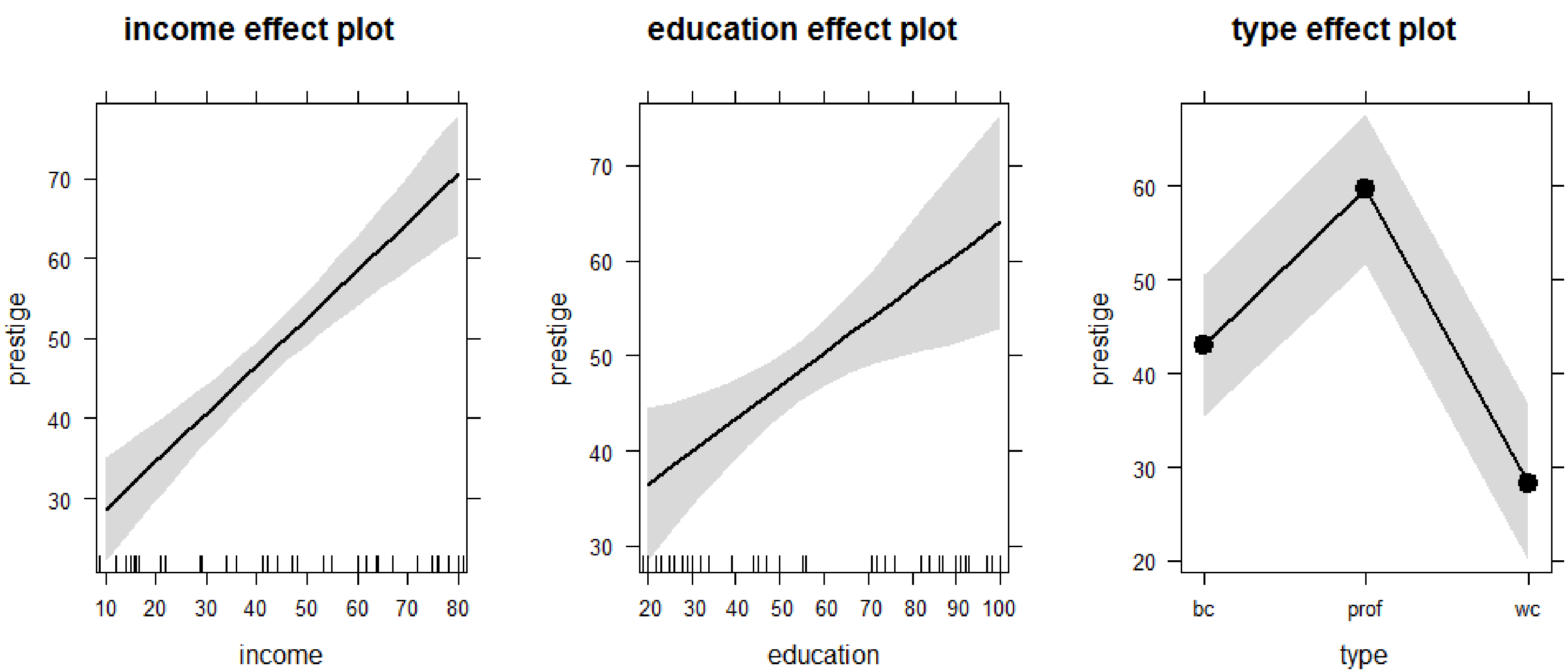
```
Call:
lm(formula = prestige ~ income + education + type, data = Duncan)


Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept)  -0.18503    3.71377  -0.050  0.96051    
income        0.59755    0.08936   6.687 5.12e-08 ***
education     0.34532    0.11361   3.040  0.00416 ** 
typeprof     16.65751    6.99301   2.382  0.02206 *  
typewc      -14.66113    6.10877  -2.400  0.02114 *  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 9.744 on 40 degrees of freedom
Multiple R-squared:  0.9131,    Adjusted R-squared:  0.9044 
F-statistic:   105 on 4 and 40 DF,  p-value: < 2.2e-16
```

How to understand effect of each predictor?

Models: Effect plots

Fitted models are more easily interpreted by plotting the predicted values.
Effect plots do this nicely, making plots for each high-order term, controlling for others

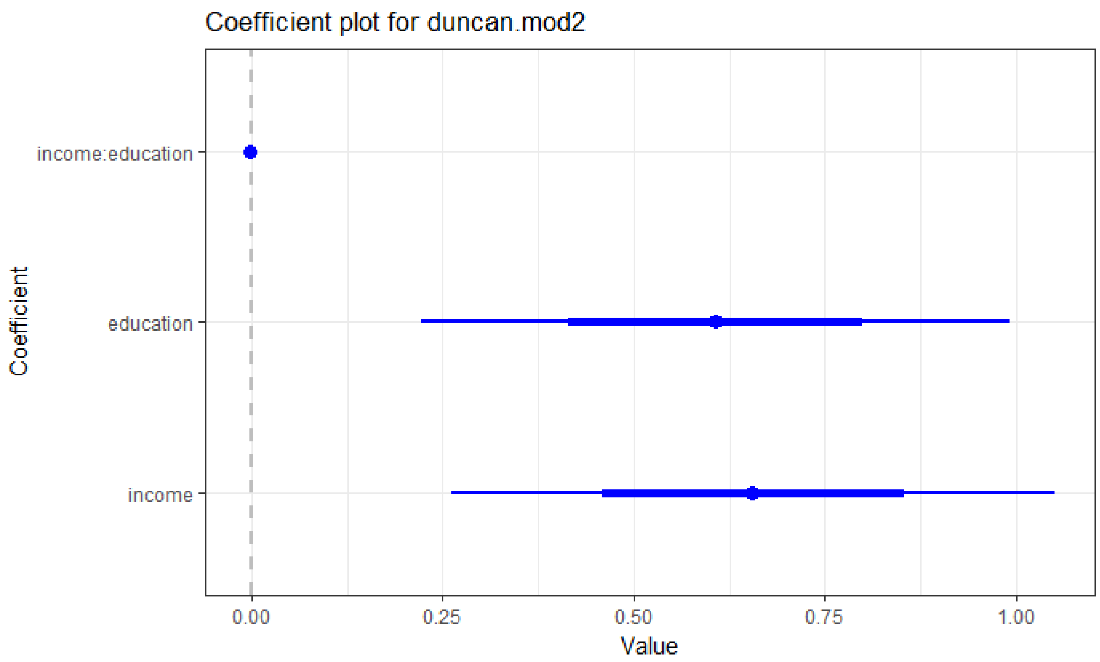
```
library(effects)
duncan.eff1 <- allEffects(duncan.mod1)
plot(duncan.eff1)
```



Models: Coefficient plots

Sometimes you need to report or display the coefficients from a fitted model.
A plot of coefficients with CIs is sometimes more effective than a table.

```
library(coefplot)
duncan.mod2 <- lm(prestige ~ income * education, data=Duncan)
coefplot(duncan.mod2, intercept=FALSE, lwdInner=2, lwdOuter=1,
     title="Coefficient plot for duncan.mod2")
```



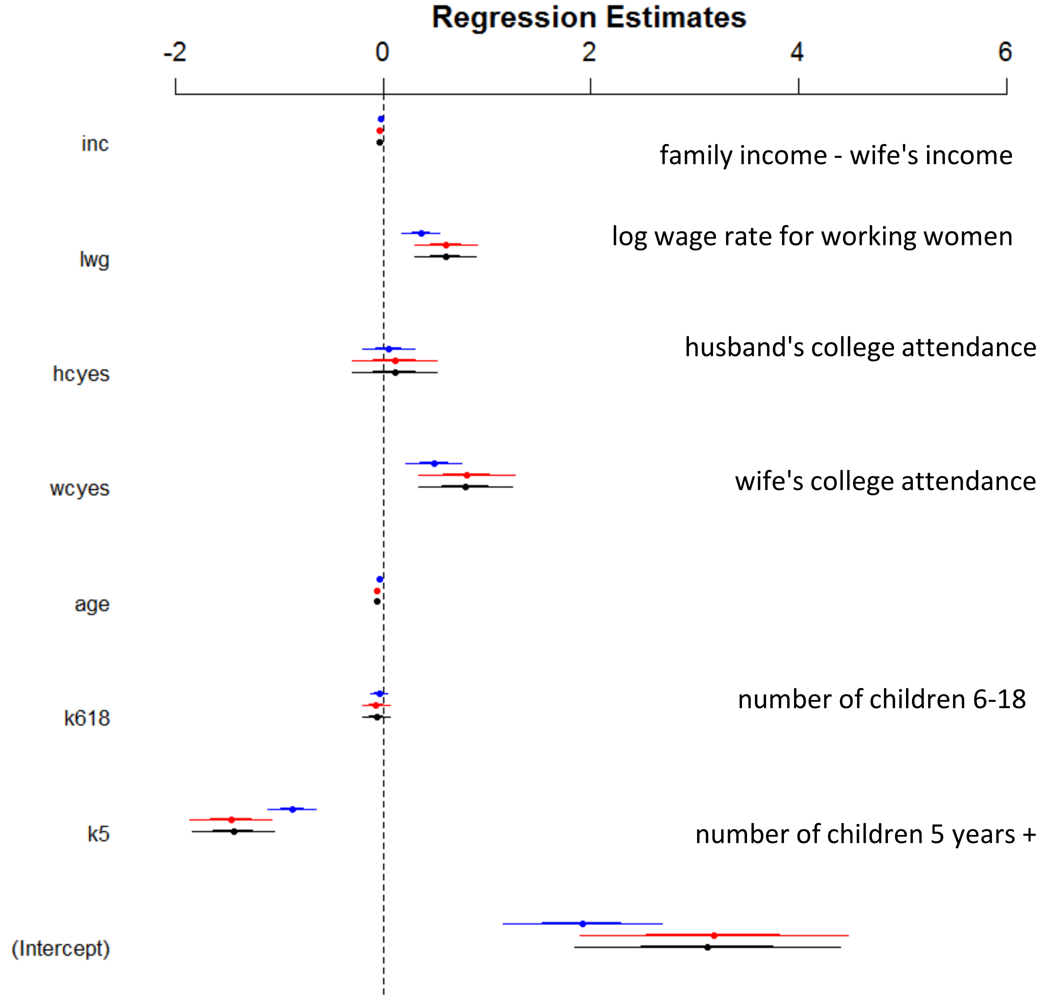
Coefficient plots become increasingly useful as:

(a) models become more complex
(b) we have several models to compare

This plot compares three different models for women's labor force participation fit to data from Mroz (1987) in the car package

This makes it relatively easy to see

(a) which terms are important
(b) how models differ



This example from: https://www.r-statistics.com/2010/07/visualization-of-regression-coefficients-in-r/

3D graphics

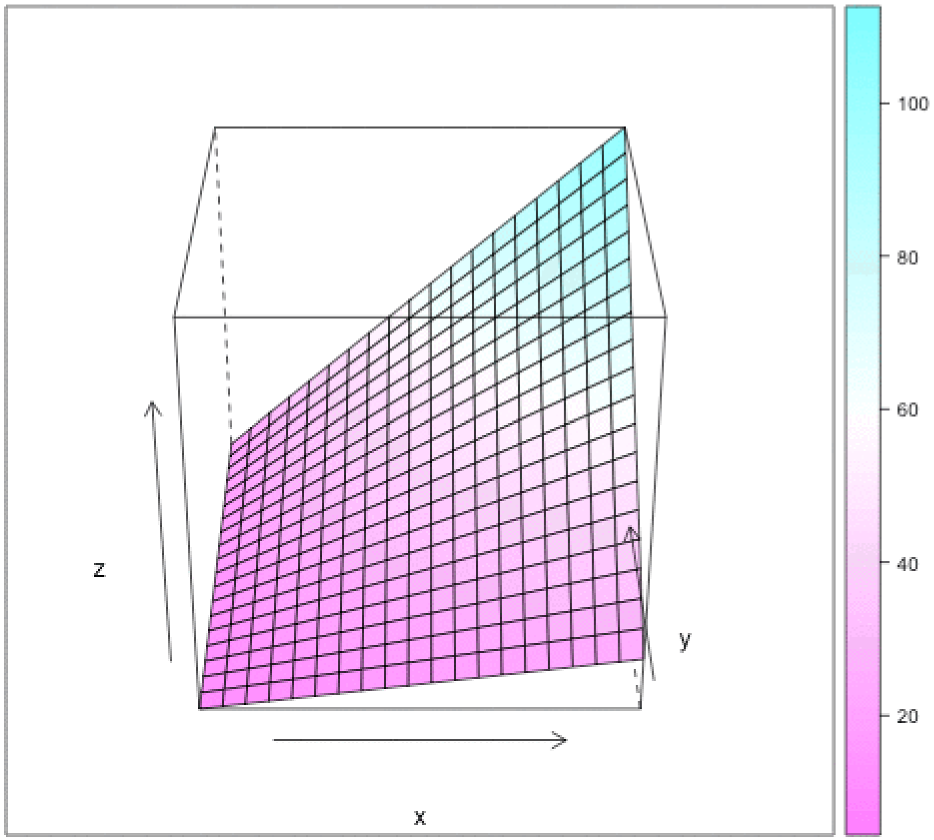
R has a wide variety of features and packages that support 3D graphics

This example illustrates the concept of an interaction between predictors in a linear regression model

It uses:
lattice::wireframe(z ~ x + y, …)

The basic plot is "printed" 36 times rotated 10^o about the z axis to produce 36 PNG images.

The ImageMagick utility is used to convert these to an animated GIF graphic



z = 10 + .5x +.3y + .2 x*y

3D graphics: code

1. Generate data for the model z = 10 + .5x +.3y + .2 x*y

```
b0 <- 10      # intercept
b1 <- .5      # x coefficient
b2 <- .3      # y coefficient
int12 <- .2   # x*y coefficient
g <- expand.grid(x = 1:20, y = 1:20)
g$z <- b0 + b1*g$x + b2*g$y + int12*g$x*g$y
```

2. Make one 3D plot

```
library(lattice)
wireframe(z ~ x * y, data = g)
```

3. Create a set of PNG images, rotating around the z axis

```
png(file="example%03d.png", width=480, height=480)
for (i in seq(0, 350 ,10)){
    print(wireframe(z ~ x * y, data = g,
              screen = list(z = i, x = -60), drape=TRUE))}
dev.off()
```

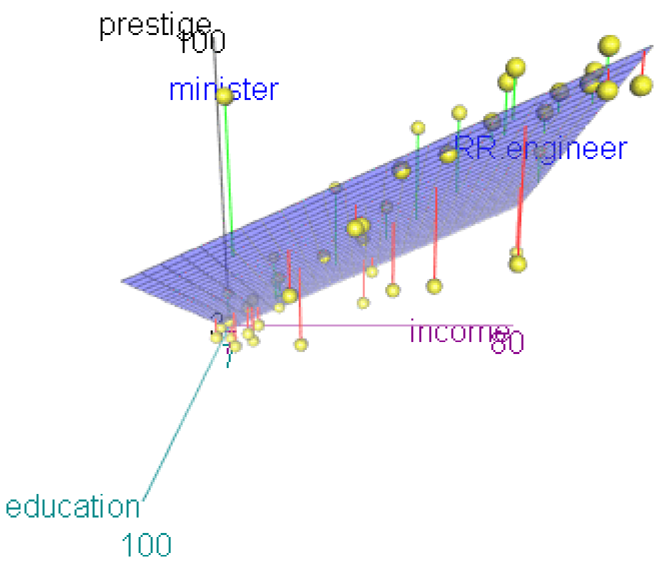
4. Convert PNGs to GIF using ImageMagik

```
system("convert -delay 40 example*.png animated_3D_plot.gif")
```

3D graphics

The rgl package is the most general for drawing 3D graphs in R.
Other R packages use this for 3D statistical graphs

This example uses car::scatter3d() to show the data and fitted response surface for the multiple regression model for the Duncan data




```
scatter3d(prestige ~ income + education,
  data=Duncan, id.n=2, revolutions=2)
```

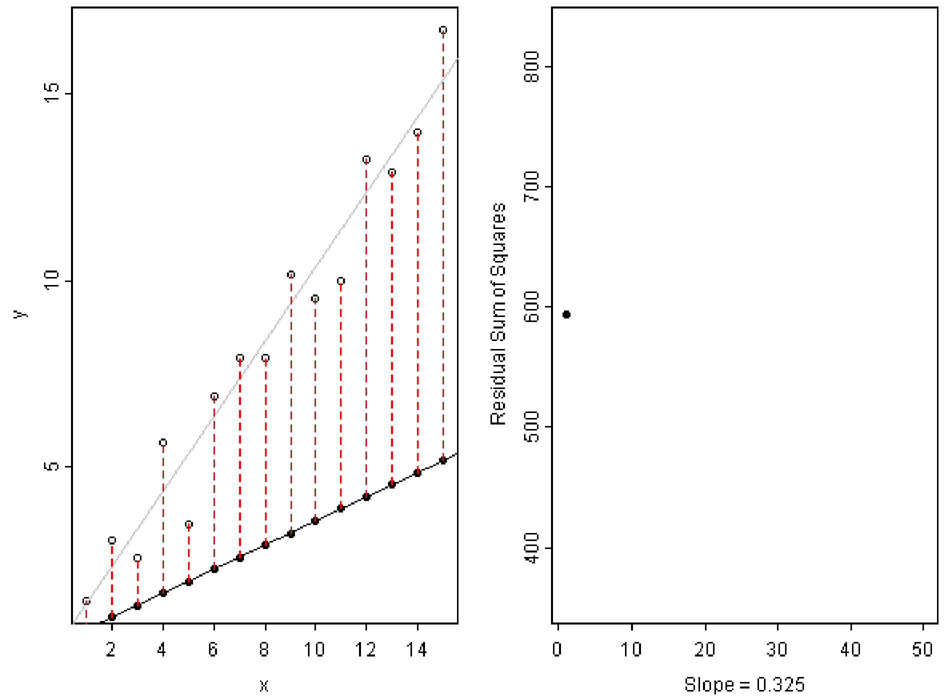
Statistical animations

Statistical concepts can often be illustrated in a dynamic plot of some process.

This example illustrates the idea of least squares fitting of a regression line.

As the slope of the line is varied, the right panel shows the residual sum of squares.

This plot was done using the animate package

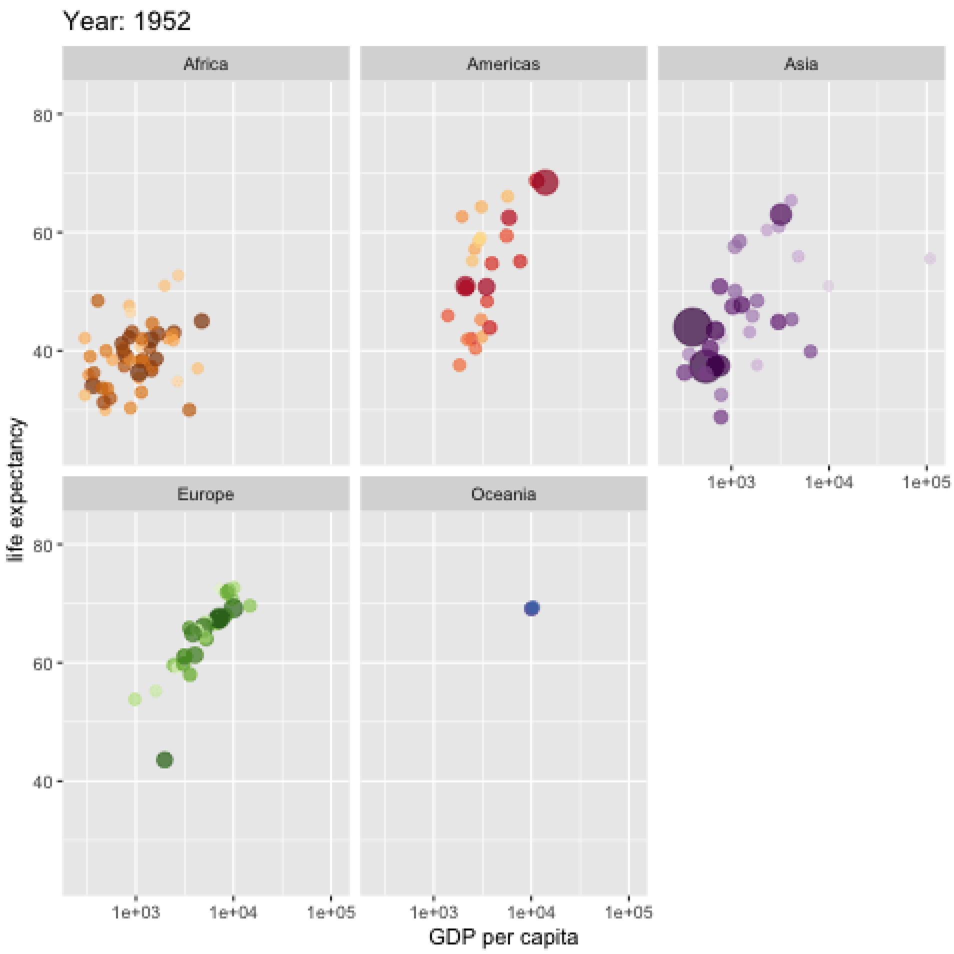


Data animations

Time-series data are often plotted against time on an X axis.

Complex relations over time can often be made simpler by animating change – liberating the X axis to show something else

This example from the tweenr package (using gganimate)



See: https://github.com/thomasp85/tweenr for some simple examples

Maps and spatial visualizations

Spatial visualization in R, combines map data sets, statistical models for spatial data, and a growing number of R packages for map-based display

This example, from Paul Murrell's *R Graphics* book shows a basic map of Brazil, with provinces and their capitals, shaded by region of the country.

Data-based maps can show spatial variation of some variable of interest

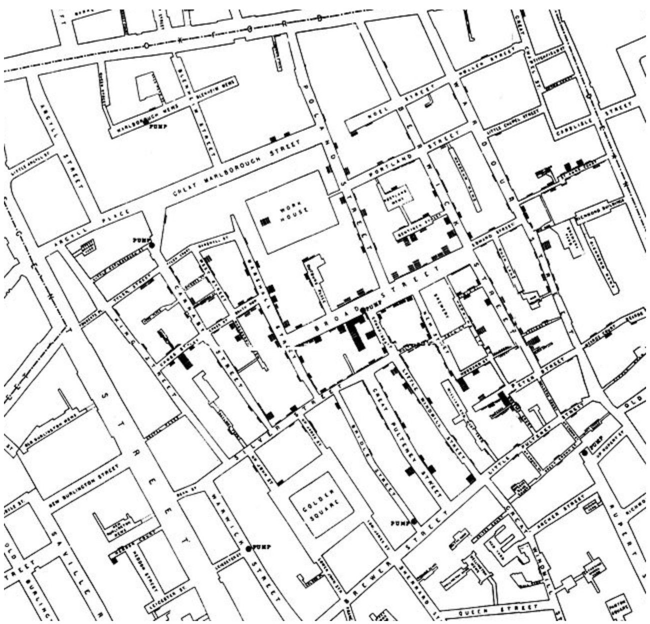
Murrell, Fig. 14.5

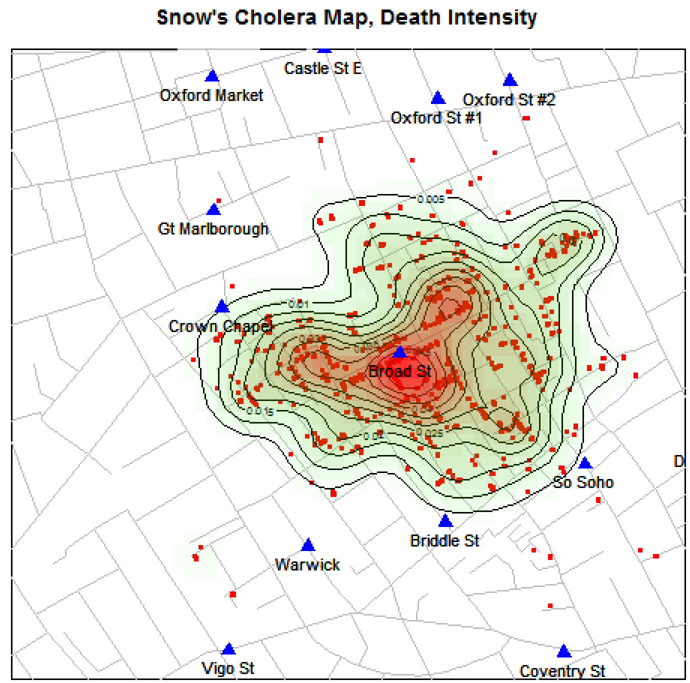
Maps and spatial visualizations

Dr. John Snow's map of cholera in London, 1854

Enhanced in R in the HistData package to make Snow's point

Portion of Snow's map:






```
library(HistData)
SnowMap(density=TRUE,
   main="Snow's Cholera Map, Death Intensity")
```

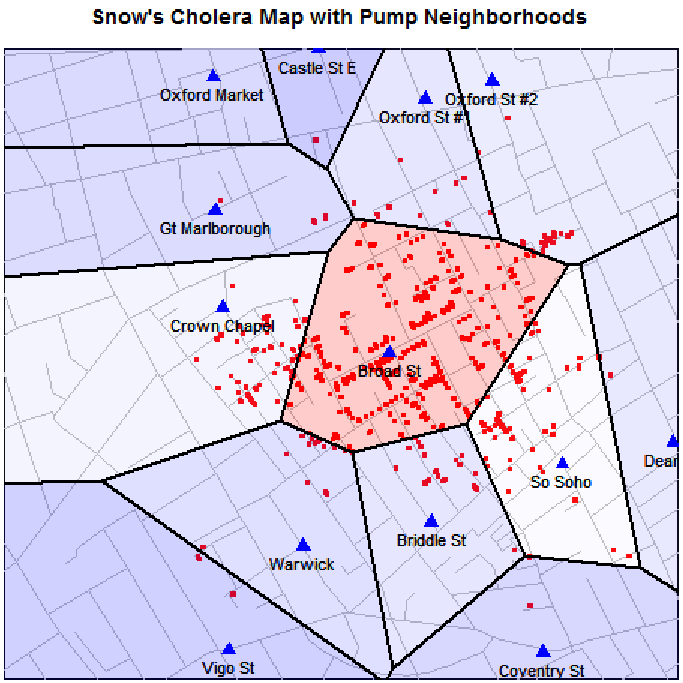
Contours of death densities are calculated using a 2d binned kernel density estimate, bkde2D() from the KernSmooth package

Maps and spatial visualizations

Dr. John Snow's map of cholera in London, 1854

Enhanced in R in the HistData package to make Snow's point

These and other historical examples come from Friendly & Wainer, *The Origin of Graphical Species*, Harvard Univ. Press, in progress.




```
SnowMap(density=TRUE,
  main="Snow's Cholera Map with Pump Neighborhoods“)
```

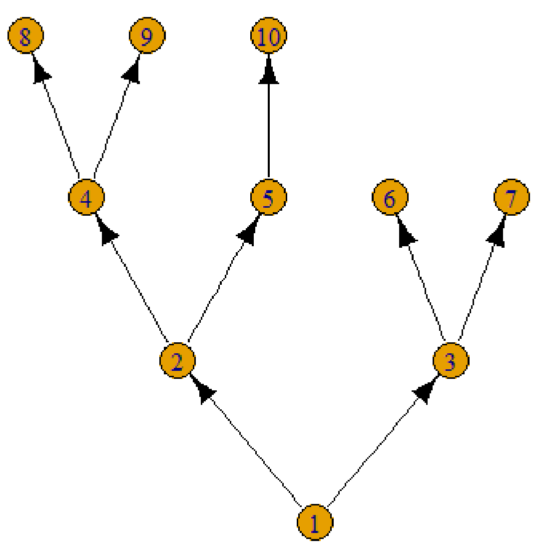
Neighborhoods are the Voronoi polygons of the map closest to each pump, calculated using the deldir package.

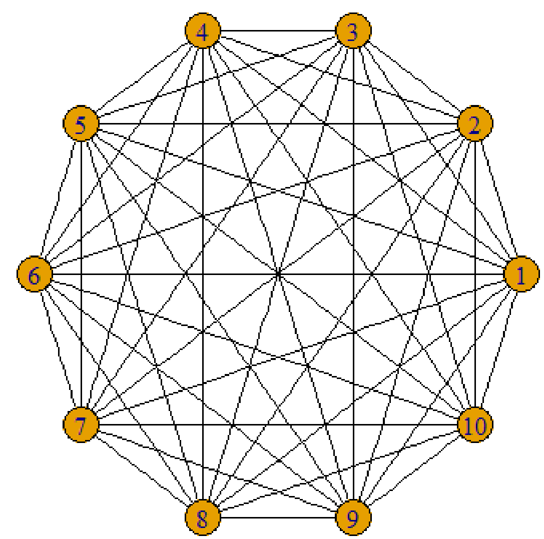
Diagrams: Trees & Graphs

A number of R packages are specialized to draw particular types of diagrams.
igraph is designed for network diagrams of nodes and edges

```
library(igraph)
tree <- graph.tree(10)
tree <- set.edge.attribute(tree, "color", value="black")
plot(treeIgraph,
    layout=layout.reingold.tilford(tree,
    root=1, flip.y=FALSE))
```

```
full <- graph.full(10)
fullIgraph <- set.edge.attribute(full, "color",
value="black")
plot(full, layout=layout.circle)
```



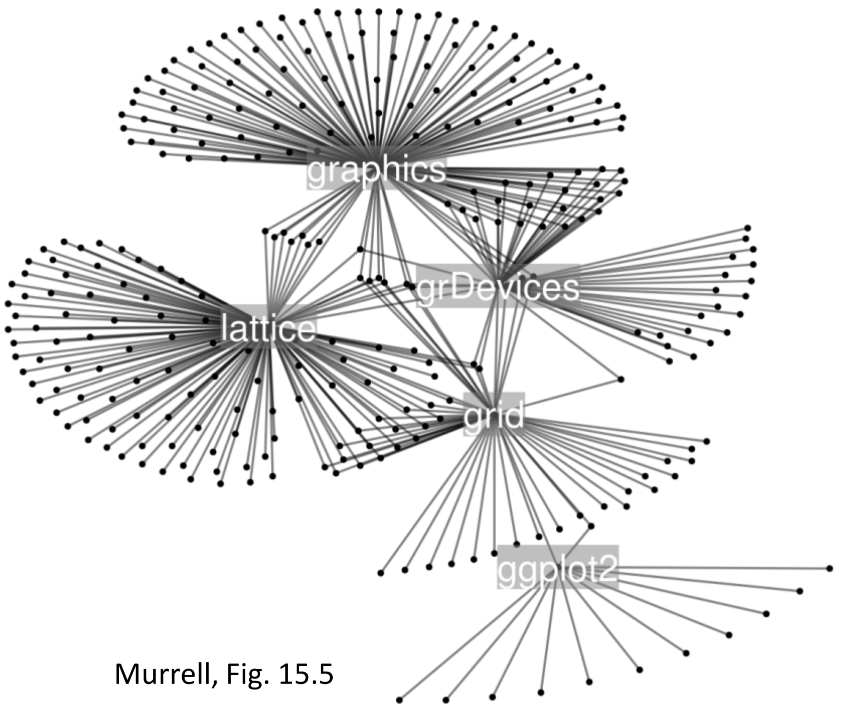


Diagrams: Network diagrams

graphvis (http://www.graphviz.org/) is a comprehensive program for drawing network diagrams and abstract graphs. It uses a simple notation to describe nodes and edges.
The Rgraphviz package (from Bioconductor) provides an R interface

This example, from Murrell's *R Graphics* book, shows a node for each package that directly depends on the main R graphics packages.

An interactive version could provide "tool tips", allowing exploring the relationships among packages

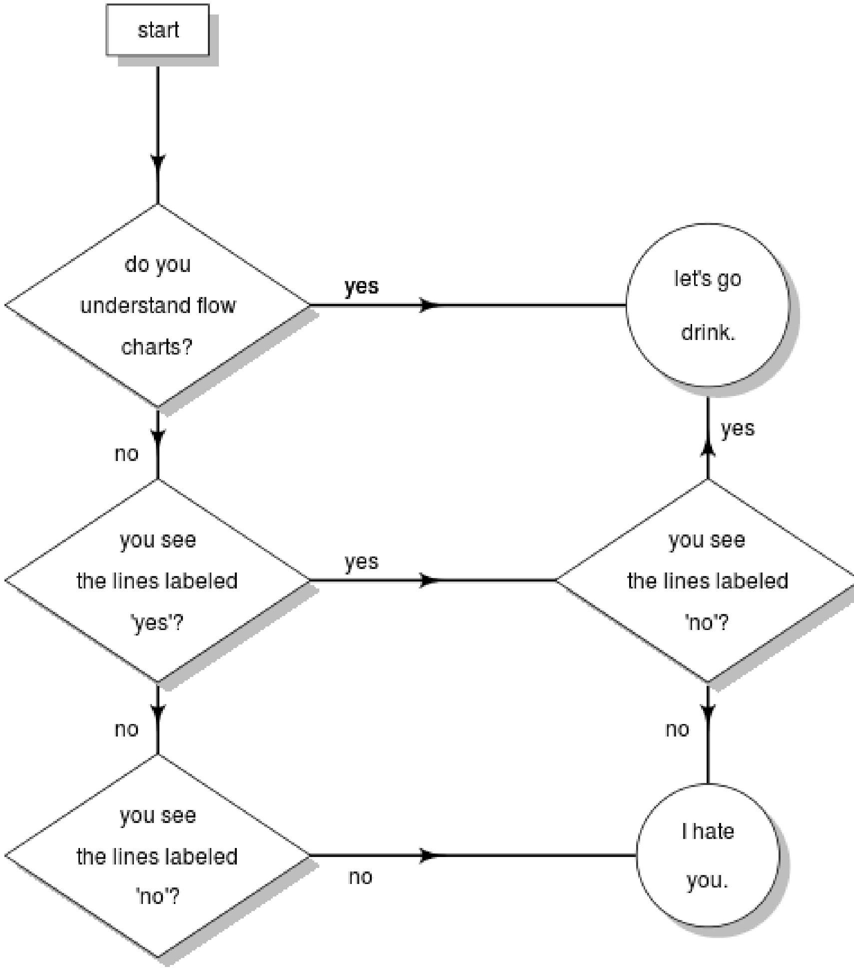


Murrell, Fig. 15.5

Diagrams: Flow charts

The diagram package:

Functions for drawing diagrams with various shapes, lines/arrows, text boxes, etc.

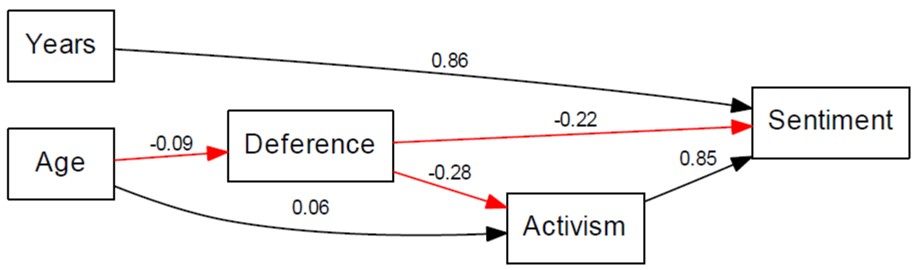


Flow chart about understanding flow charts (after http://xkcd.com/518). From: Murrell, Fig 15.10

Path diagrams: structural equation models

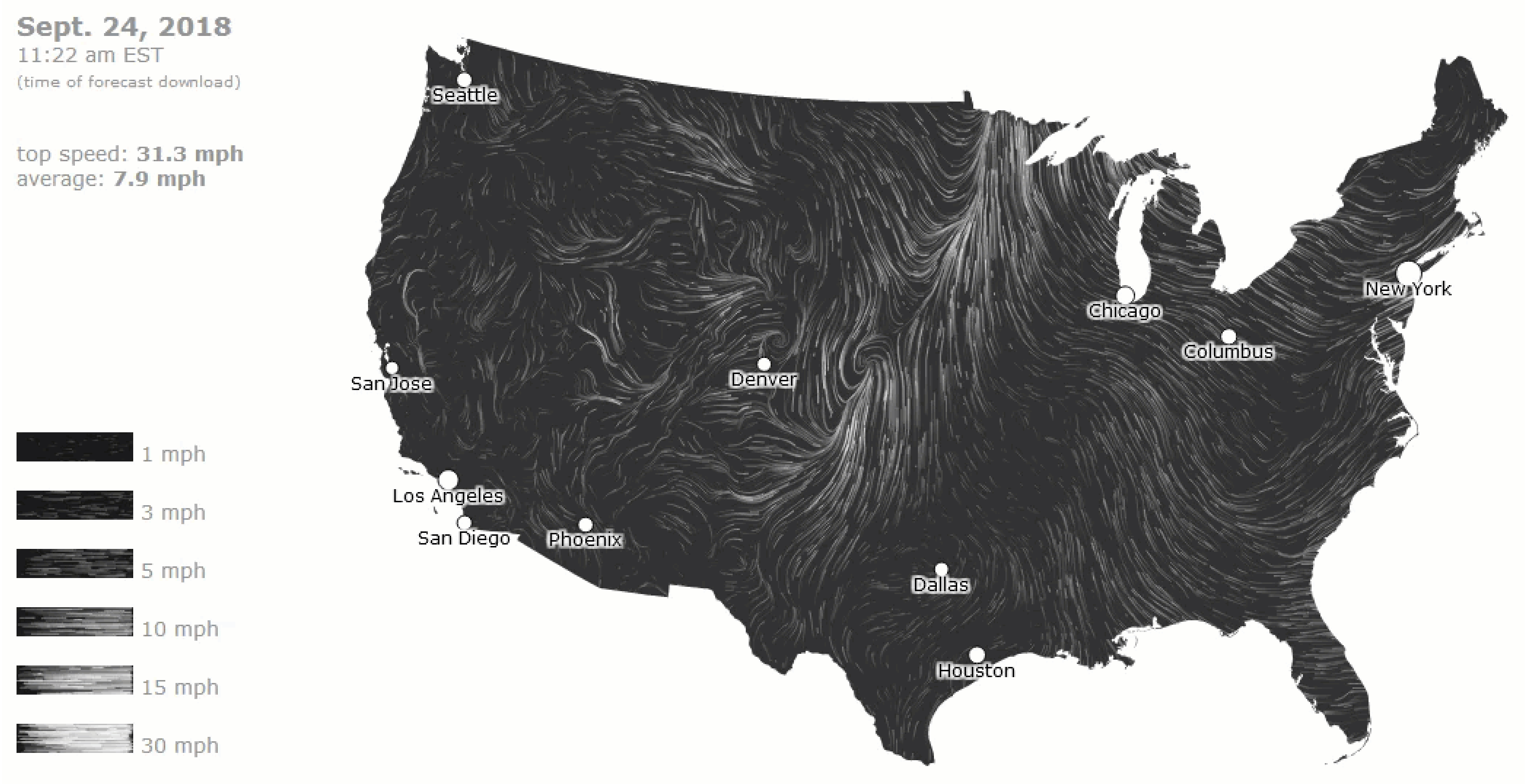
Similar diagrams are used to display structural equation models as "path diagrams"
The sem and laavan packages have pathDiagram() functions to draw a proposed or fitted model.
They use the DiagrammeR package to do the drawing.

```
library(sem)
union.mod <- specifyEquations(covs="x1, x2", text="
  y1 = gam12*x2
  y2 = beta21*y1 + gam22*x2
  y3 = beta31*y1 + beta32*y2 + gam31*x1
")
union.sem <- sem(union.mod, union, N=173)
pathDiagram(union.sem,
  edge.labels="values",
  file="union-sem1",
  min.rank=c("x1", "x2"))
```



Dynamically updated data visualizations

The wind map app, http://hint.fm/wind/ is one of a growing number of R-based applications that harvests data from standard sources, and presents a visualization



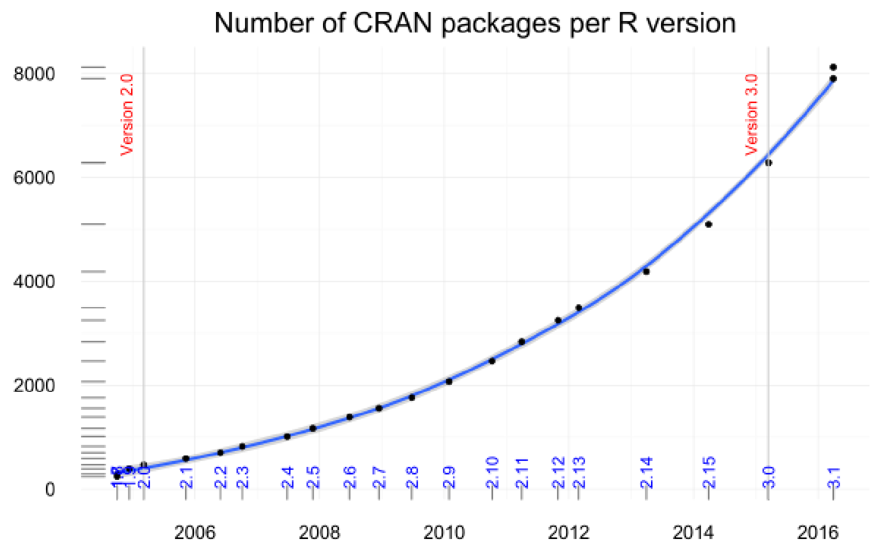
Web scraping: CRAN package history

R has extensive facilities for extracting and processing information obtained from web pages. The XML package is one useful tool for this purpose.

This example:

- downloads information about all R packages from the CRAN web site,
- finds & counts all of those available for each R version,
- plots the counts with ggplot2, adding a smoothed curve, and plot annotations

On Jan. 27, 2017, the number of R packages on CRAN reached 10,000



Code from: https://git.io/vy4wS

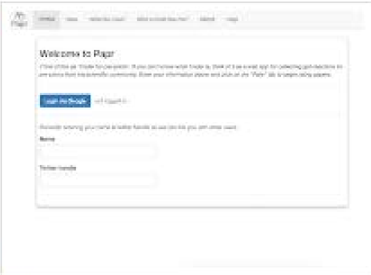
shiny: Interactive R applications

shiny, from R Studio, makes it easier to develop interactive applications

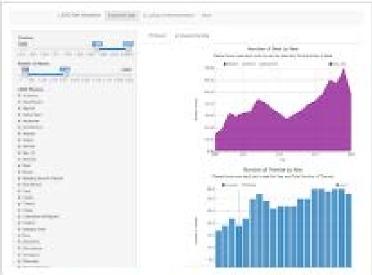
Shiny User Showcase

The Shiny User Showcase contains an inspiring set of sophisticated apps developed and contributed by Shiny users.

Genome browser



Papr

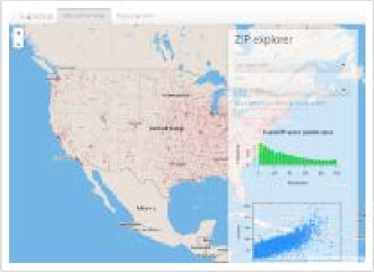


Lego Set Database Explorer

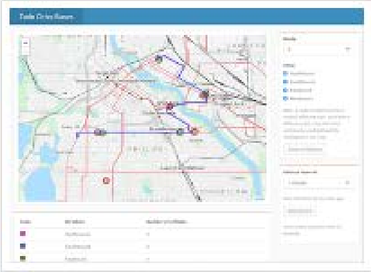
See more

Interactive visualizations

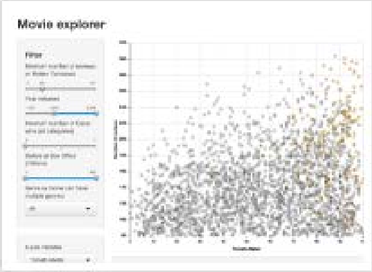
Shiny is designed for fully interactive visualization, using JavaScript libraries like d3, Leaflet, and Google Charts.



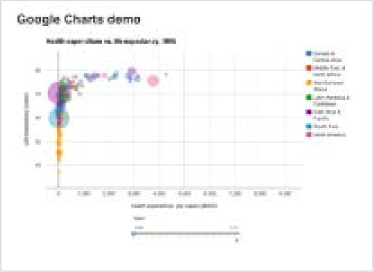
SuperZip example



Bus dashboard



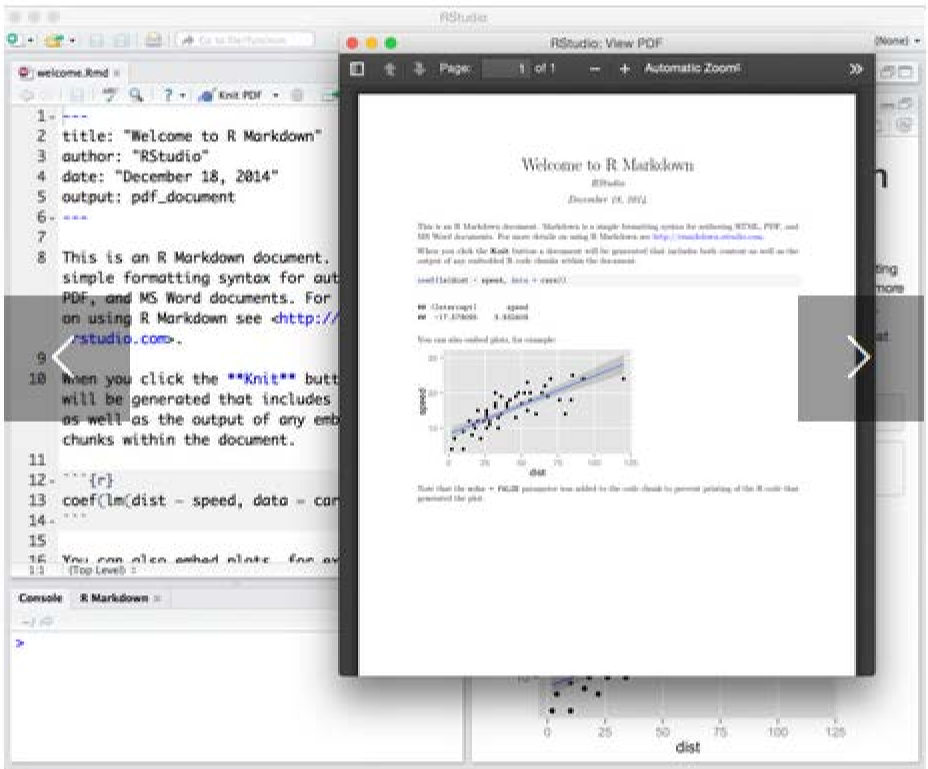
Movie explorer



Google Charts

Many examples at https://shiny.rstudio.com/gallery/

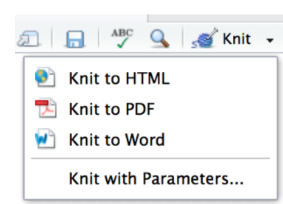
Reproducible analysis & reporting



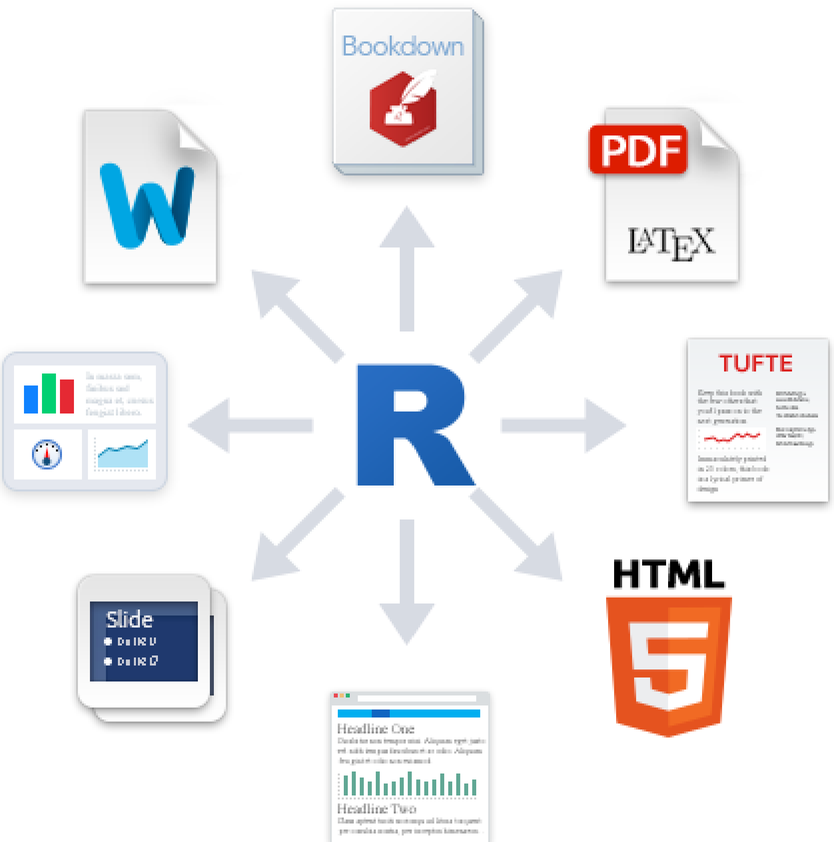
R Studio, together with the knitr and rmarkdown packages provide an easy way to combine writing, analysis, and R output into complete documents

.Rmd files are just text files, using rmarkdown markup and knitr to run R on "code chunks"

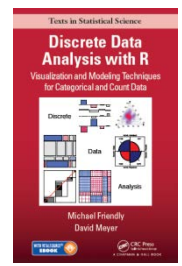
A given document can be rendered in different output formats:



Output formats and templates



The integration of R, R Studio, knitr, rmarkdown and other tools is now highly advanced.



My last book was written entirely in R Studio, using .Rnw syntax ⟶ LaTeX ⟶ PDF ⟶ camera ready copy

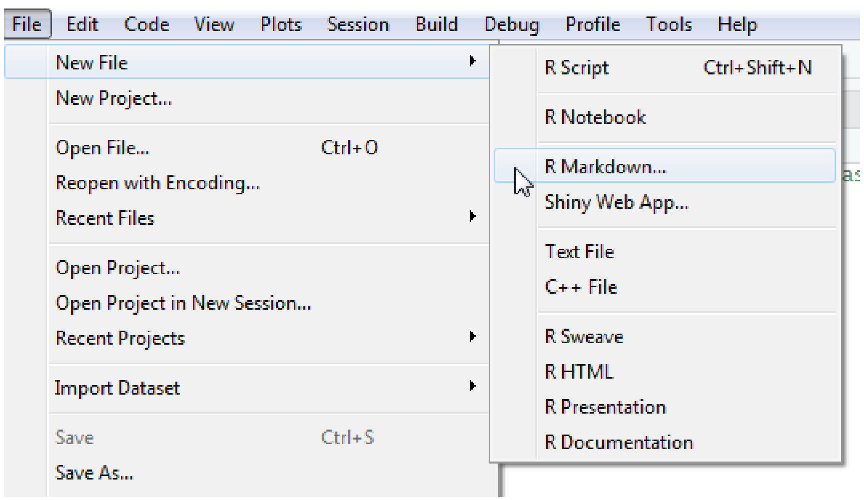
The ggplot2 book was written using .Rmd format.

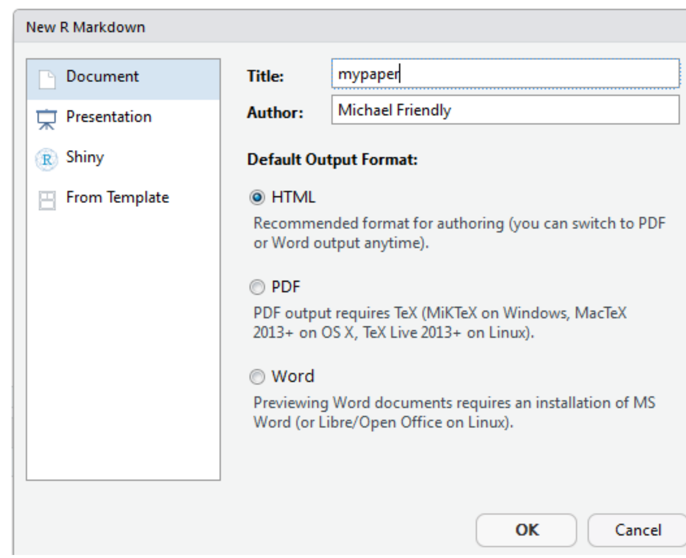
The bookdown package makes it easier to manage a book-length project – TOC, fig/table #s, cross-references, etc.

Templates are available for APA papers, slides, handouts, entire web sites, etc.

Writing it up

- In R Studio, create a .Rmd file to use R Markdown for your write-up
  - lots of options: HTML, Word, PDF (needs LaTeX)



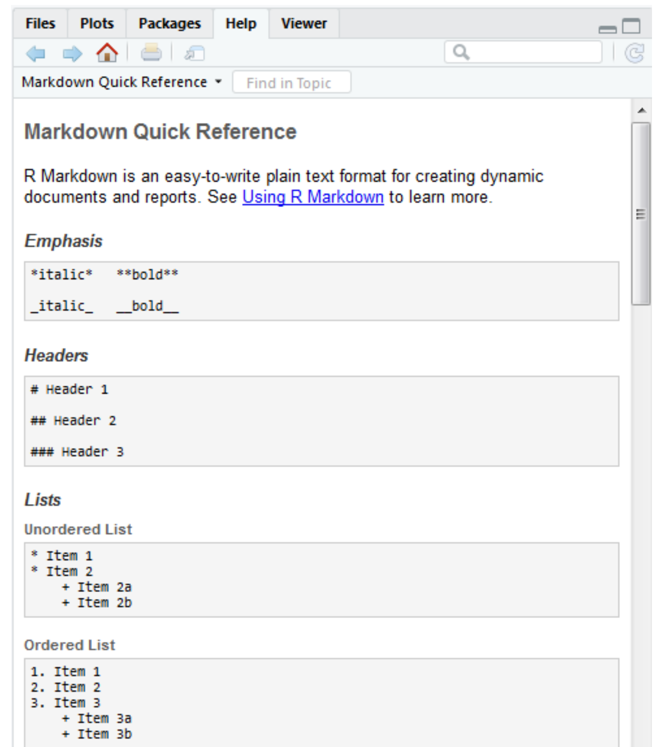


Writing it up

- Use simple Markdown to write text
- Include code chunks for analysis & graphs

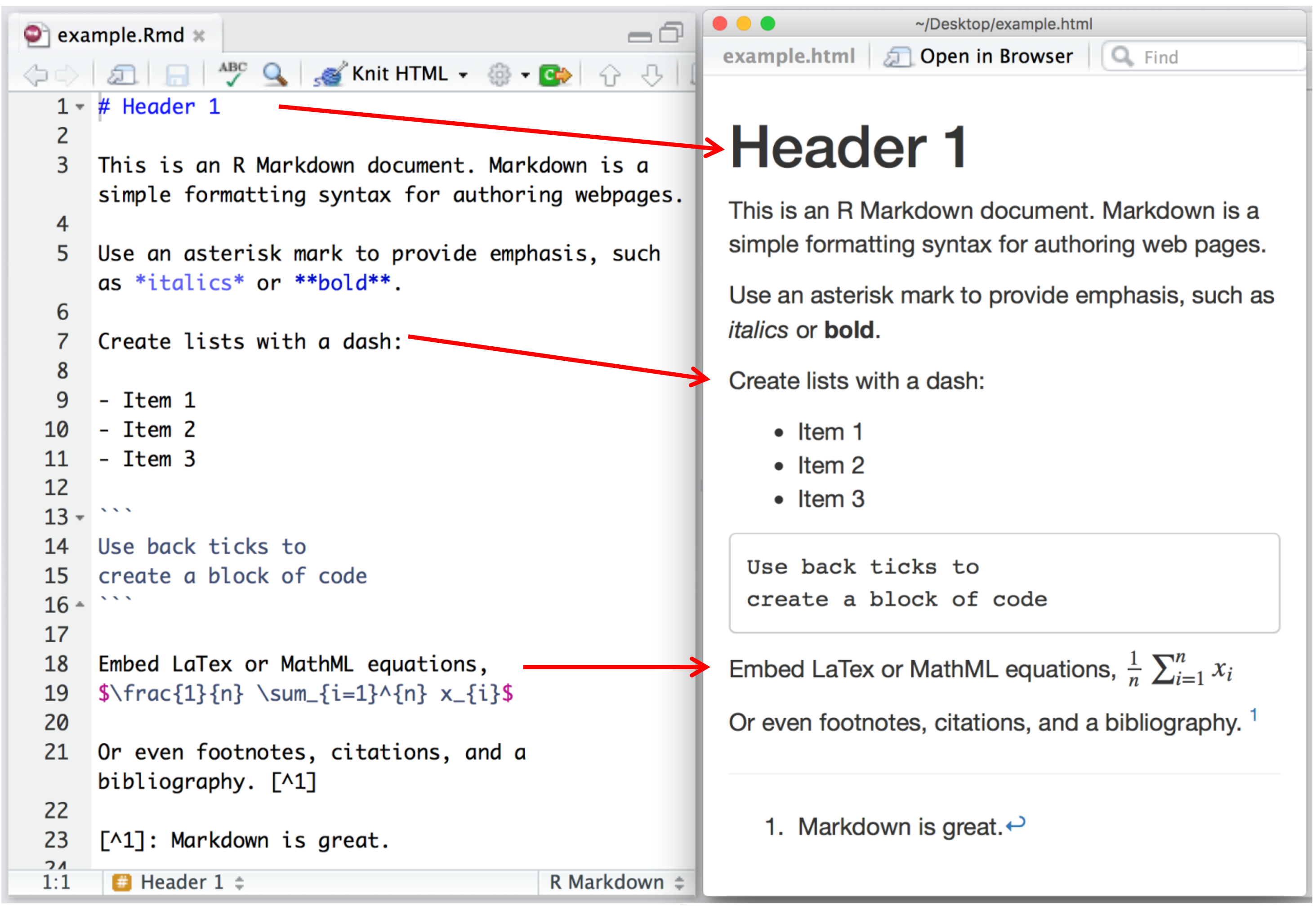
mypaper.Rmd, created from a template

Help -> Markdown quick reference



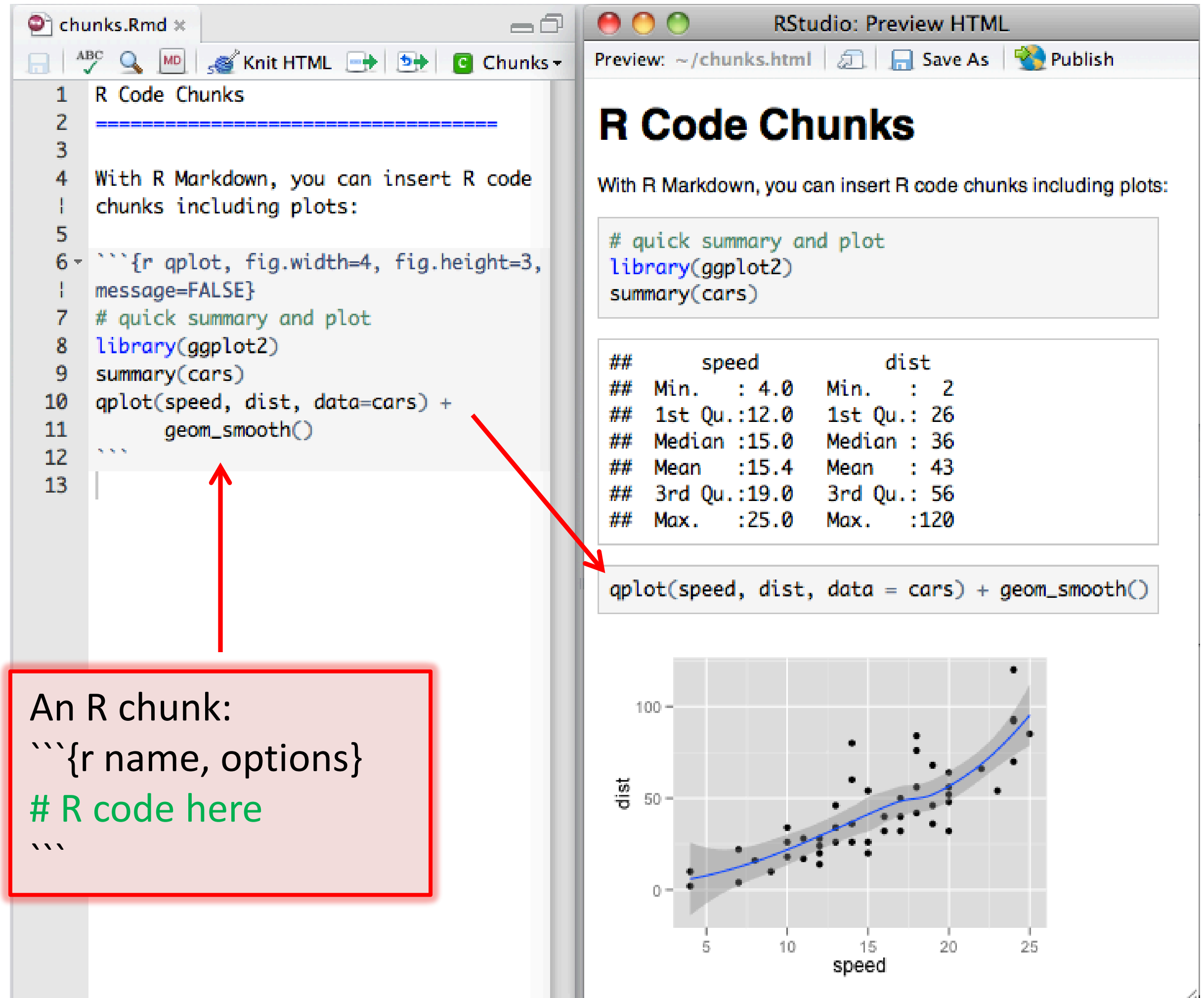
rmarkdown basics

rmarkdown uses simple markdown formatting for all standard document elements



R code chunks

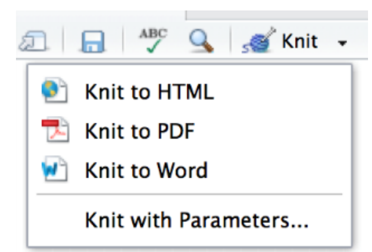
R code chunks are run by knitr, and the results are inserted in the output document



An R chunk:
```{r name, options}
# R code here
```

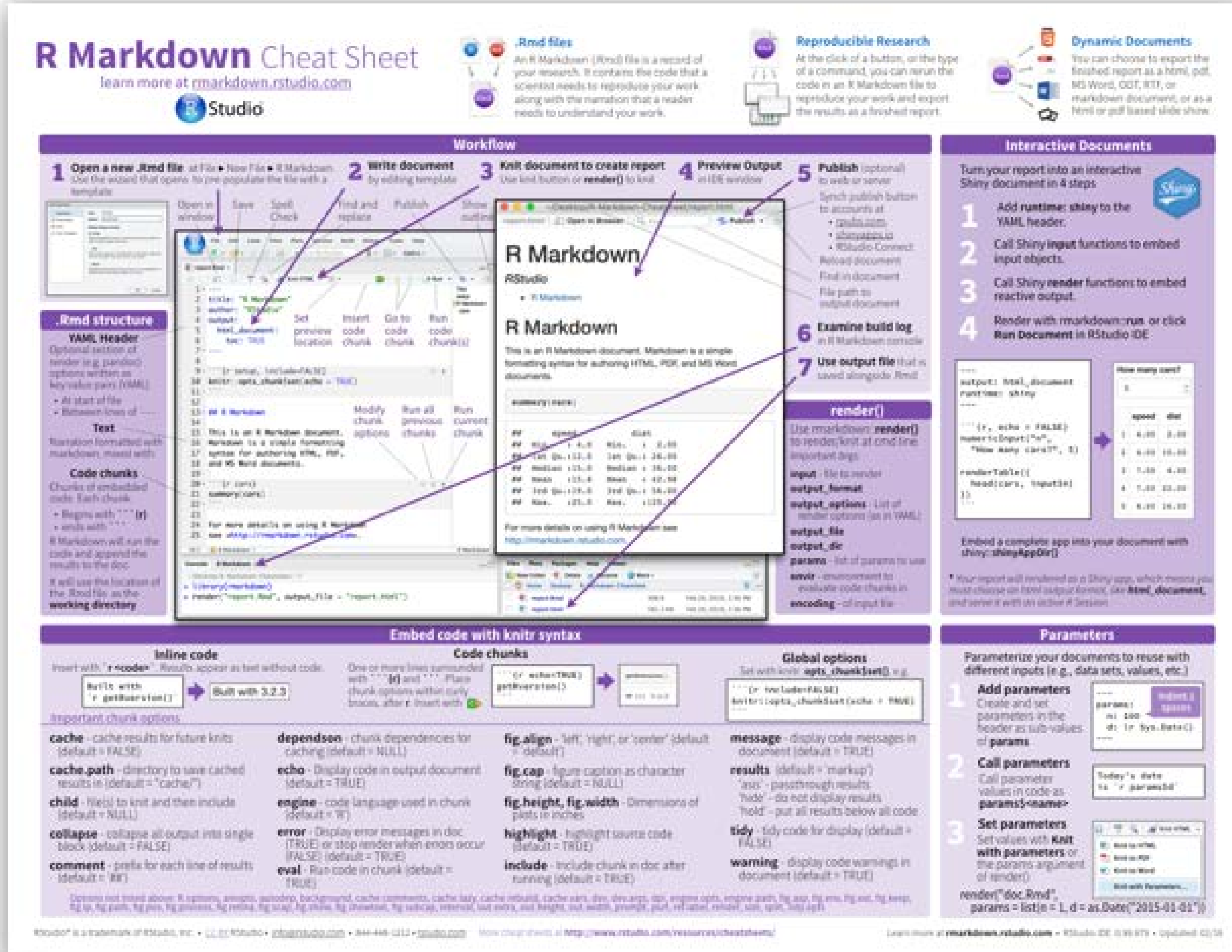
There are many options for controlling the details of chunk output – numbers, tables, graphs

Choose the output format:



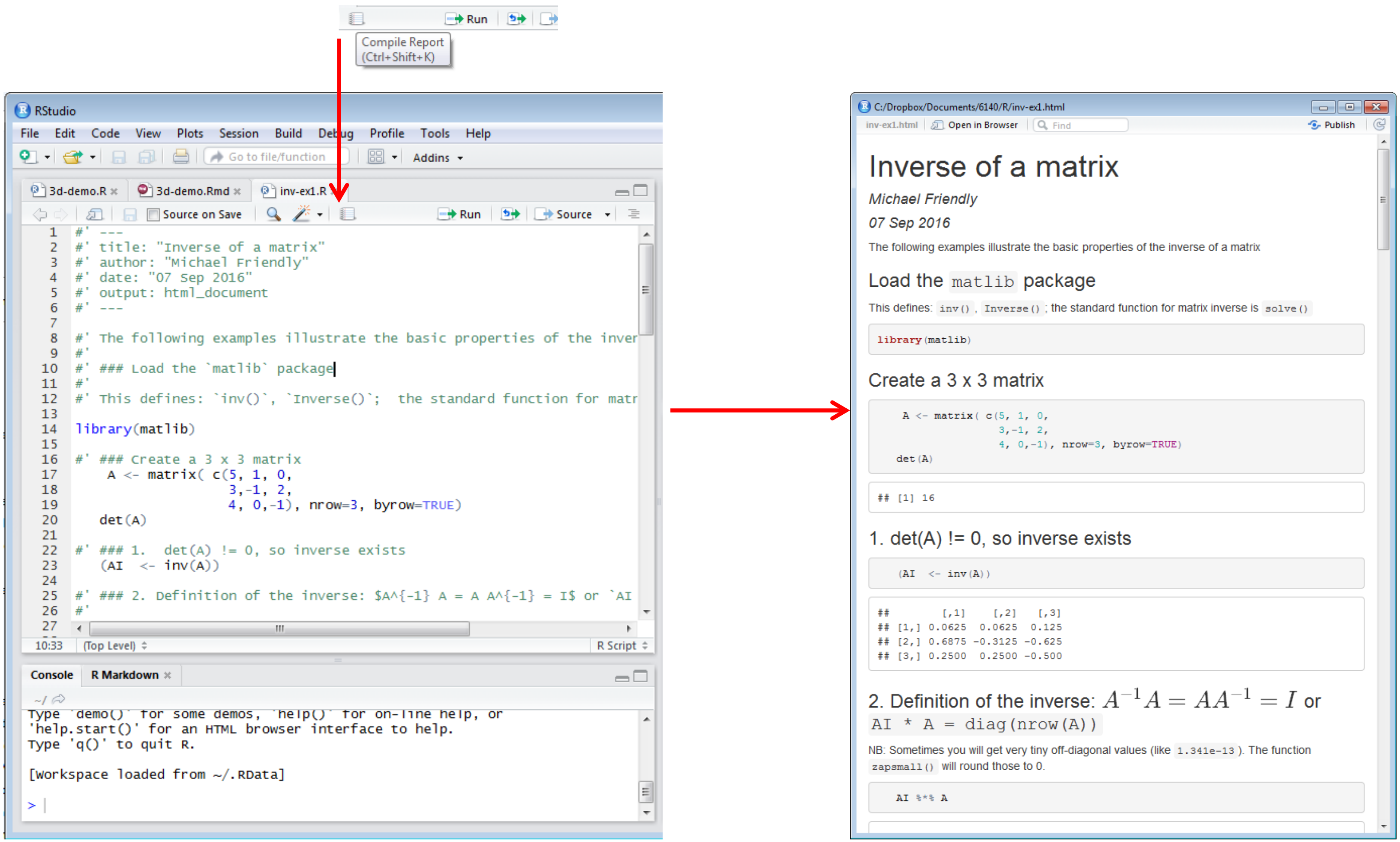
The R Markdown Cheat Sheet provides most of the details

https://www.rstudio.com/wp-content/uploads/2016/03/rmarkdown-cheatsheet-2.0.pdf



R notebooks

Often, you just want to “compile” an R script, and get the output embedded in the result, in HTML, Word, or PDF. Just type Ctrl-Shift-K or tap the Compile Report button



Summary & Homework

- Today has been mostly about an overview of R graphics, but with emphasis on:
  - R, R Studio, R package tools
  - Roles of graphics in data analysis,
  - A small gallery of examples of different kinds of graphic applications in R; only small samples of R code
  - Work flow: How to use R productively in analysis & reporting
- Next week: start on skills with traditional graphics
- Homework:
  - Install R & R Studio
  - Find one or more examples of data graphs from your research area
    - What are the graphic elements: points, lines, areas, regions, text, labels, ???
    - How could they be “described” to software such as R?
    - How could they be improved?