An introduction to R
Presented at The 2nd biennial meeting of the Association of Research in Personality

William Revelle

Department of Psychology
Northwestern University
Evanston, Illinois USA

June 16, 2011

Overview

Outline of Part 1

1. What is R?
   - Where did it come from, why use it?
   - Installing R on your computer and adding packages
   - Basic R capabilities: Calculation, Statistical tables, Graphics
     - Basic Graphics
   - Some simple 2 x 2 data analysis
2. A brief example
   - A brief example of exploratory and confirmatory data analysis
3. Basic statistics and graphics
   - 4 steps: read, explore, test, graph
   - Basic descriptive and inferential statistics
     - t-test, ANOVA, χ^2
     - Linear Regression

R: Statistics for all us

1. What is it?
2. Why use it?
3. Common (mis)perceptions of R
4. Examples for psychologists
   - graphical displays
   - basic statistics
   - advanced statistics
   - Although programming is easy in R, that is beyond the scope of today

R: What is it?

1. R: An international collaboration
2. R: The open source - public domain version of S+
3. R: Written by statistician (and all of us) for statisticians (and the rest of us)
4. R: Not just a statistics system, also an extensible language.
    - This means that as new statistics are developed they tend to appear in R far sooner than elsewhere.
    - For example, the most recent issue of *Pschological Methods* had at least three articles with examples or supplementary work done in R
    - R facilitates asking questions that have not already been asked.

Statistical Programs for Psychologists

- General purpose programs
  - R
  - S+
  - SAS
  - SPSS
  - STATA
  - Systat
- Specialized programs
  - Mx
  - EQS
  - AMOS
  - LISREL
  - MPlus
  - Your favorite program

Statistical Programs for Psychologists

- General purpose programs
    - R
    - $+
    - A
    - $P$$
    - $TATA
    - $y$tat
- Specialized programs
    - Mx (OpenMx is part of R)
    - EQ$
    - AMO$
    - LI$REL
    - MPlu$
    - Your favorite program

R: A way of thinking

- "R is the lingua franca of statistical research. Work in all other languages should be discouraged."
- "This is R. There is no if. Only how."
- "Overall, SAS is about 11 years behind R and S-Plus in statistical capabilities (last year it was about 10 years behind) in my estimation."

Taken from the R.-fortunes (selections from the R.-help list serve)

R is open source, how can you trust it?

- Q: "When you use it [R], since it is written by so many authors, how do you know that the results are trustable?"
- A: "The R engine [...] is pretty well uniformly excellent code but you have to take my word for that. Actually, you don't. The whole engine is open source so, if you wish, you can check every line of it. If people were out to push dodgy software, this is not the way they'd go about it."

What is R?: Technically

- R is an open source implementation of S (S-Plus is a commercial implementation)
- R is available under GNU Copy-left
- The current version of R is 2.13.0
- The development version of R 2.14.0 is available to test and will be released this fall
- R is a group project run by a core group of developers (with new releases semiannually)

(Adapted from Robert Gentleman)

R: A brief history

- 1991-93: Ross Dhaka and Robert Gentleman begin work on R project at U. Auckland
- 1995: R available by ftp under the GPL
- 96-97: mailing list and R core group is formed
- 2000: John Chambers, designer of S joins the Rcore (wins a prize for best software from ACM for S)
- 2001-2011: Core team continues to improve base package with a new release every 6 months.
- Many others contribute "packages" to supplement the functionality for particular problems
  - 2003-04-01: 250 packages
  - 2004-10-01: 500 packages
  - 2007-04-12: 1,000 packages
  - 2009-10-04: 2,000 packages
  - 2011-05-12 3,000 packages

Misconception: R is hard to use

1. R doesn't have a GUI (Graphical User Interface)
    - Partly true, many use syntax
    - Partly not true, GUIs exist (e.g., R Commander, R-Studio)
    - Quasi GUIs for Mac and PCs make syntax writing easier
2. R syntax is hard to use
    - Not really, unless you think an iPhone is hard to use
    - Easier to give instructions of 1-4 lines of syntax rather than pictures of what menu to pull down.
    - Keep a copy of your syntax, modify it for the next analysis.
3. R is not user friendly: A personological description of R
    - R is introverted: it will tell you what you want to know if you ask, but not if you don't ask.
    - R is conscientious: it wants commands to be correct.
    - R is not agreeable: its error messages are at best cryptic.
    - R is stable: it does not break down under stress.
    - R is open: new ideas about statistics are easily developed.

Misconceptions: R is hard to learn

1. With a brief web based tutorial at http://personality-project.org/r, 2nd and 3rd year undergraduates in psychological methods and personality research courses are using R for descriptive and inferential statistics and producing publication quality graphics.
2. More and more psychology departments are using it for graduate and undergraduate instruction.
3. R is easy to learn, hard to master
   - R-help newsgroup is very supportive
   - There are multiple web based and pdf tutorials see (e.g., http://www.r-project.org/)
   - Short courses using R for many applications
4. Books and websites are available for for SPSS and SAS users trying to learn R (e.g.,http://oit.utk.edu/scc/RforSAS&SPSSusers.pdf by Bob Muenchen).

Ok, how do I get it? Getting started with R

- Download from R Cran (http://cran.r-project.org/)
  - Choose appropriate operating system and download compiled R
- Install R (current version is 2.13.0)
- Start R
- Add useful packages (you just need to do this once)
  - install.packages("ctv") #this downloads the task view package
  - library(ctv) #this activates the ctv package
  - install.views("Psychometrics") #among others
  - Take a 5 minute break
- Activate the package(s) you want to use today (e.g., *psych*)
  - library(psych) #necessary for most of today's examples
- Use R

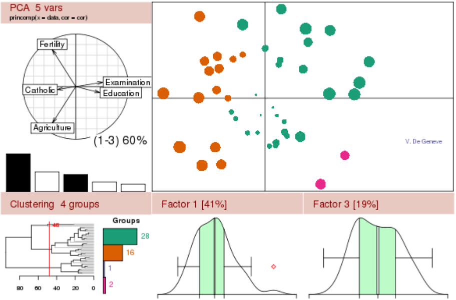
Go to the R.project.org

The R Project for Statistical Computing

http://www.r-project.org/

Bill's Apple Yahoo! scholar.google.com Google Maps Wikipedia YouTube News (609) Popular CRAN Package

The R Project for Statistical Computing



About R
- What is R?
- Contributors
- Screenshots
- What's new?

Download, Packages
- CRAN

R Project
- Foundation
- Members & Donors
- Mailing Lists
- Bug Tracking
- Developer Page
- Conferences
- Search

Documentation
- Manuals
- FAQs
- The R Journal
- Wiki
- Books
- Certification
- Other

Misc
- Bioconductor
- Related Projects
- User Groups
- Links

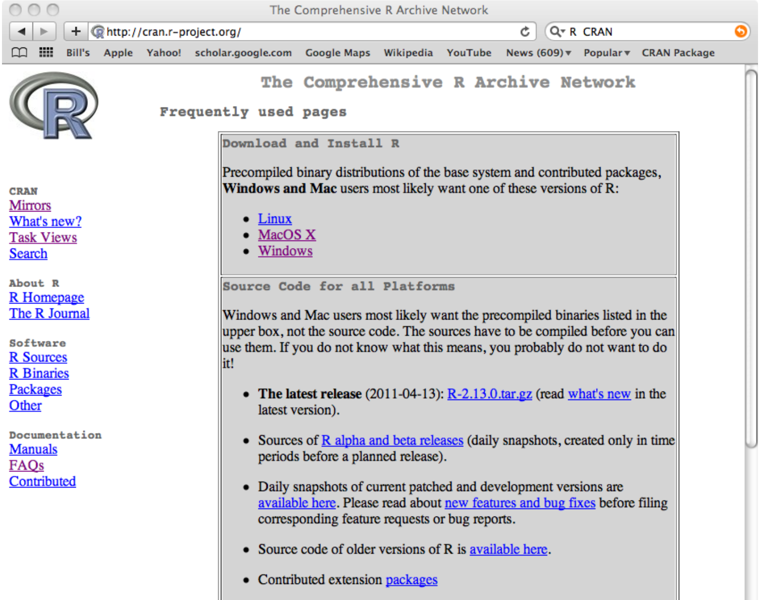
Getting Started:

- R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS. To **download R**, please choose your preferred CRAN mirror.
- If you have questions about R like how to download and install the software, or what the license terms are, please read our answers to frequently asked questions before you send an email.

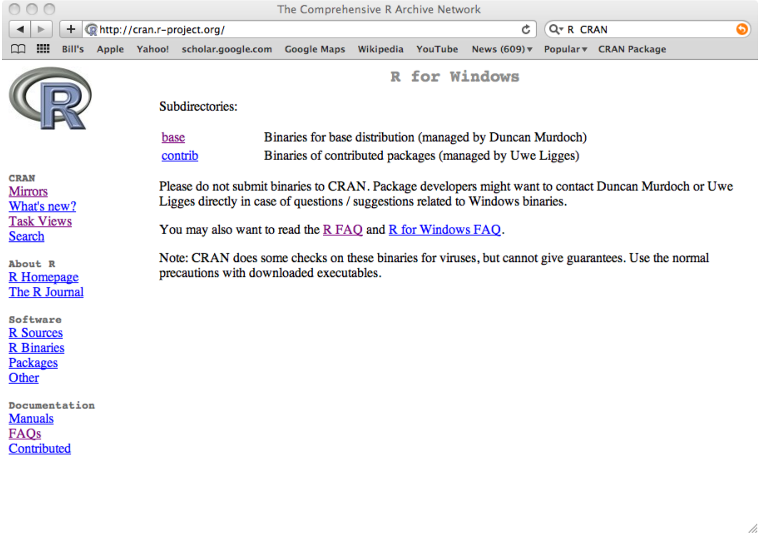
News:

- **R version 2.13.0** has been released on 2011-04-13. The source code is first available in this directory, and eventually via all of CRAN. Binaries will arrive in due course (see download instructions above).
- **The R Journal Vol.2/2** is available
- R has participated with 5 project in the Google Summer of Code 2010.

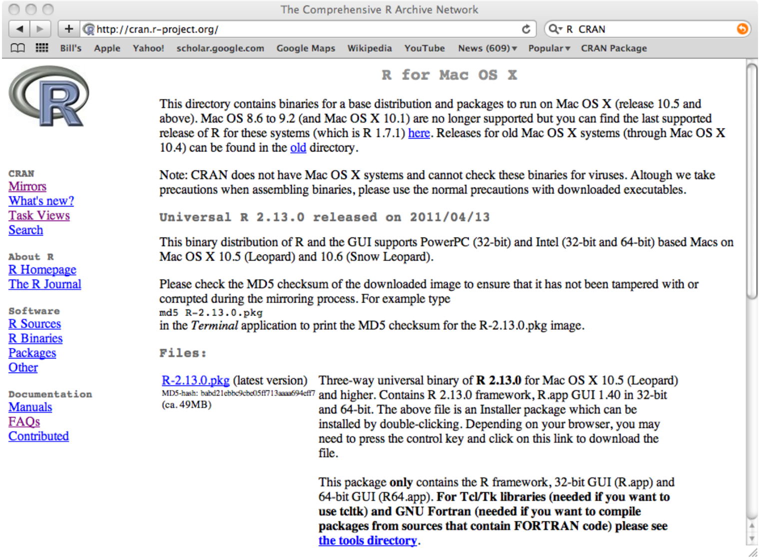
Go to the Comprehensive R Archive Network (CRAN)



Download and install the appropriate version – PC



Download and install the appropriate version – Mac



The Comprehensive R Archive Network

http://cran.r-project.org/

R CRAN

Bill's Apple Yahoo! scholar.google.com Google Maps Wikipedia YouTube News (609) Popular CRAN Package

CRAN
Mirrors
What's new?
Task Views
Search

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
Other

Documentation
Manuals
FAQs
Contributed

R for Mac OS X

This directory contains binaries for a base distribution and packages to run on Mac OS X (release 10.5 and above). Mac OS 8.6 to 9.2 (and Mac OS X 10.1) are no longer supported but you can find the last supported release of R for these systems (which is R 1.7.1) here. Releases for old Mac OS X systems (through Mac OS X 10.4) can be found in the old directory.

Note: CRAN does not have Mac OS X systems and cannot check these binaries for viruses. Altough we take precautions when assembling binaries, please use the normal precautions with downloaded executables.

Universal R 2.13.0 released on 2011/04/13

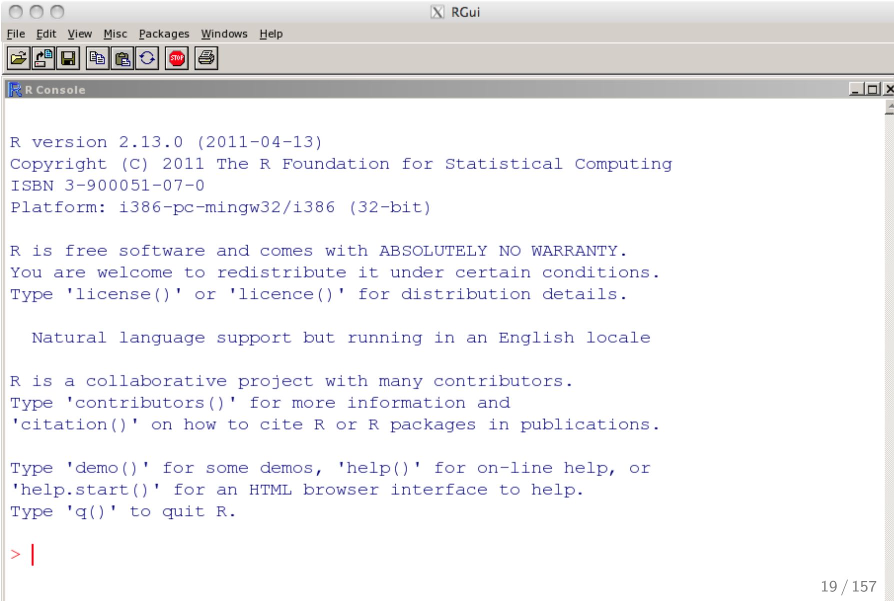
This binary distribution of R and the GUI supports PowerPC (32-bit) and Intel (32-bit and 64-bit) based Macs on Mac OS X 10.5 (Leopard) and 10.6 (Snow Leopard).

Please check the MD5 checksum of the downloaded image to ensure that it has not been tampered with or corrupted during the mirroring process. For example type
`md5 R-2.13.0.pkg`
in the *Terminal* application to print the MD5 checksum for the R-2.13.0.pkg image.

Files:

R-2.13.0.pkg (latest version) MD5-hash: babd21ebbc9cbe05ff713aaaa694eff7 (ca. 49MB)	Three-way universal binary of **R 2.13.0** for Mac OS X 10.5 (Leopard) and higher. Contains R 2.13.0 framework, R.app GUI 1.40 in 32-bit and 64-bit. The above file is an Installer package which can be installed by double-clicking. Depending on your browser, you may need to press the control key and click on this link to download the file. This package **only** contains the R framework, 32-bit GUI (R.app) and 64-bit GUI (R64.app). **For Tcl/Tk libraries (needed if you want to use tcltk) and GNU Fortran (needed if you want to compile packages from sources that contain FORTRAN code) please see the tools directory.**

Starting R on a PC



Start up R and get ready to play (Mac version)

```
R version 2.13.0 (2011-04-13)
Copyright (C) 2011 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
Platform: i386-apple-darwin9.8.0/i386 (32-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[R.app GUI 1.40 (5751) i386-apple-darwin9.8.0]

>    # >  is the prompt for all commands    #is for comments
```

Annotated installation guide: don't type the >

```
> install.packages("ctv")
```

- Install the task view installer package. You might have to choose a "mirror" site.

```
> library(ctv)
```

- Make it active

```
> install.views("Psychometrics")
```

- Install all the packages in the "Psychometrics" task view. This will take a few minutes.

```
#or just install a few packages
> install.packages("psych")
```

- Or, just install one package (e.g., psych)

```
> install.packages("GPArotation")
> install.packages("MASS")
> install.packages("mvtnorm")
> install.packages("lavaan")
```

- as well as a few suggested packages that add functionality for factor rotation, multivariate normal distributions, etc.

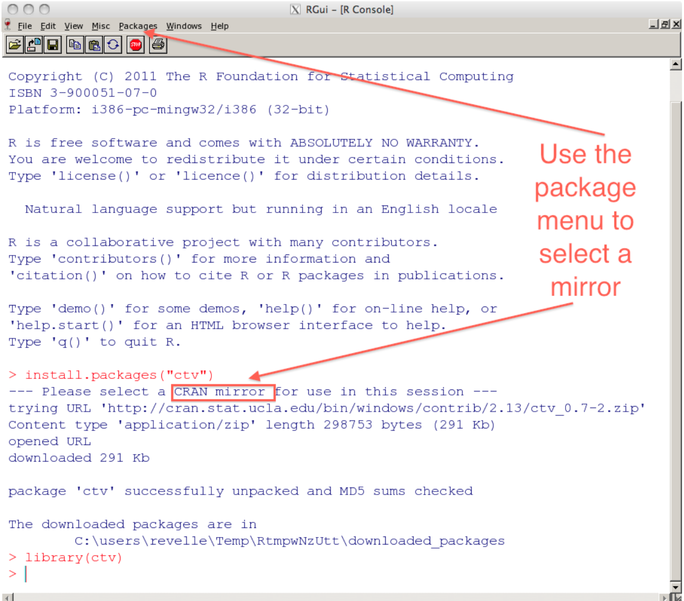
Installing just the psych package

RGui - [R Console]

File Edit View Misc Packages Windows Help

```
R version 2.13.0 (2011-04-13)
Copyright (C) 2011 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
Platform: i386-pc-mingw32/i386 (32-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> install.packages("psych")
--- Please select a CRAN mirror for use in this session ---
trying URL 'http://cran.stat.ucla.edu/bin/windows/contrib/2.13/psych_1.0-97.zip'
Content type 'application/zip' length 1952216 bytes (1.9 Mb)
opened URL
downloaded 1.9 Mb
```

Or, install and use ctv package to load a task view on a PC



Check the version number for R (should be $\geq$ 2.13) and for psych ($\geq$1.0-.97)

```
> library(psych)
> sessionInfo()

R version 2.13.0 (2011-04-13)
Platform: x86_64-apple-darwin9.8.0/x86_64 (64-bit)

locale:
[1] C/en_US.UTF-8/C/C/C/C

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] MASS_7.3-13     mvtnorm_0.9-999 psych_1.0-97

loaded via a namespace (and not attached):
[1] tools_2.13.0
```

R is extensible: The use of “packages”

- More than 3000 packages are available for R (and growing daily)
- Can search all packages that do a particular operation by using the sos package
  - `install.packages("sos")` #if you haven't already
  - library(sos) # make it active once you have it
    - `findFn`("X") #will search a web data base for all packages/functions that have "X"
    - `findFn`("factor analysis") #will return 8293 matches and reports the top 400
    - `findFn`("Item Response Theory") # will return 161 matches
    - `findFn`("INDSCAL ") # will return 8 matches.
- install.packages("X") will install a particular package (add it to your R library – you need to do this just once)
- library(X) #will make the package X available to use if it has been installed (and thus in your library)

A small subset of very useful packages

- General use
  - core R
  - MASS
  - lattice
  - lme4 (core)
  - psych
  - Zelig
- Special use
  - ltm
  - sem
  - lavaan
  - OpenMx
  - GPArotation
  - mvtnorm
  - $>$ 3000 known
  - $+$?

- General applications
  - most descriptive and inferential stats
  - Modern Applied Statistics with S
  - Lattice or Trellis graphics
  - Linear mixed-effects models
  - Personality and psychometrics
  - General purpose toolkit
- More specialized packages
  - Latent Trait Model (IRT)
  - SEM and CFA (one group)
  - SEM and CFA (multiple groups)
  - SEM and CFA (multiple groups $+$)
  - Jennrich rotations
  - Multivariate distributions
  - Thousands of more packages on CRAN
  - Code on webpages/journal articles

Basic R commands – remember don't enter the >

R is just a fancy calculator. Add, subtract, sum, products, group

```
> 2 + 2
[1] 4
> 3^4
[1] 81
> sum(1:10)
[1] 55
> prod(c(1, 2, 3, 5, 7))
[1] 210
```

It is also a statistics table (the normal distribution, the t distribution + many more)

```
> pnorm(q = 1)
[1] 0.8413447
> pt(q = 2, df = 20)
[1] 0.9703672
```

More on distributions

We can find the probability of normal scores from -3 to 3 by chaining together several commands.

```
z <- seq(from=-3,to= 3, by = .5)
z
round(pnorm(z),digits=2)
```

```
 z
 [1] -3.0 -2.5 -2.0 -1.5 -1.0 -0.5  0.0  0.5  1.0  1.5  2.0  2.5  3.0
> round(pnorm(z),digits=2)
 [1] 0.00 0.01 0.02 0.07 0.16 0.31 0.50 0.69 0.84 0.93 0.98 0.99 1.00
```

Try this again with by =.1

Make a “data frame” out of the results to provide a useful table

```
z <- seq(from=-3,to= 3, by = .5)
p <- pnorm(z)
norm.df <- data.frame(z,p)
print(norm.df,digits=2)
```

```
      z    p
1  -3.0 0.00
2  -2.5 0.01
3  -2.0 0.02
4  -1.5 0.07
5  -1.0 0.16
6  -0.5 0.31
7   0.0 0.50
8   0.5 0.69
9   1.0 0.84
10  1.5 0.93
11  2.0 0.98
12  2.5 0.99
13  3.0 1.00
```

Add the ordinate of the normal curve to this data frame

```
z <- seq(from=-3,to= 3, by = .5)
p <- pnorm(z)
d <- dnorm(z)
norm.df <- data.frame(z,p,d)
print(norm.df,digits=2)
```

```
      z      p      d
1  -3.0 0.0013 0.0044
2  -2.5 0.0062 0.0175
3  -2.0 0.0228 0.0540
4  -1.5 0.0668 0.1295
5  -1.0 0.1587 0.2420
6  -0.5 0.3085 0.3521
7   0.0 0.5000 0.3989
8   0.5 0.6915 0.3521
9   1.0 0.8413 0.2420
10  1.5 0.9332 0.1295
11  2.0 0.9772 0.0540
12  2.5 0.9938 0.0175
13  3.0 0.9987 0.0044
```

Compare the z distribution with the t distribution with 10 df

```
z <- seq(from=-3,to= 3, by = .5)
p <- pnorm(z)
d <- dnorm(z)
t <- pt(z,df=10)
norm.df <- data.frame(z,p,d,t)
print(norm.df,digits=2)
```

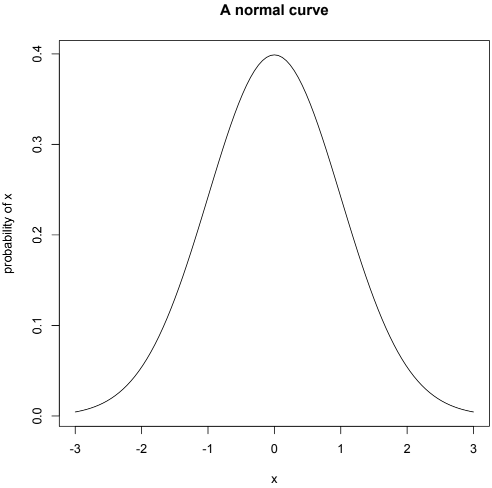
```
      z      p      d      t
1  -3.0 0.0013 0.0044 0.0067
2  -2.5 0.0062 0.0175 0.0157
3  -2.0 0.0228 0.0540 0.0367
4  -1.5 0.0668 0.1295 0.0823
5  -1.0 0.1587 0.2420 0.1704
6  -0.5 0.3085 0.3521 0.3139
7   0.0 0.5000 0.3989 0.5000
8   0.5 0.6915 0.3521 0.6861
9   1.0 0.8413 0.2420 0.8296
10  1.5 0.9332 0.1295 0.9177
11  2.0 0.9772 0.0540 0.9633
12  2.5 0.9938 0.0175 0.9843
13  3.0 0.9987 0.0044 0.9933
```

R is a set of distributions. Don't buy a stats book with tables!

Table: To obtain the density, prefix with *d*, probability with *p*, quantiles with *q* and to generate random values with *r*. (e.g., the normal distribution may be chosen by using dnorm, pnorm, qnorm, or rnorm.)

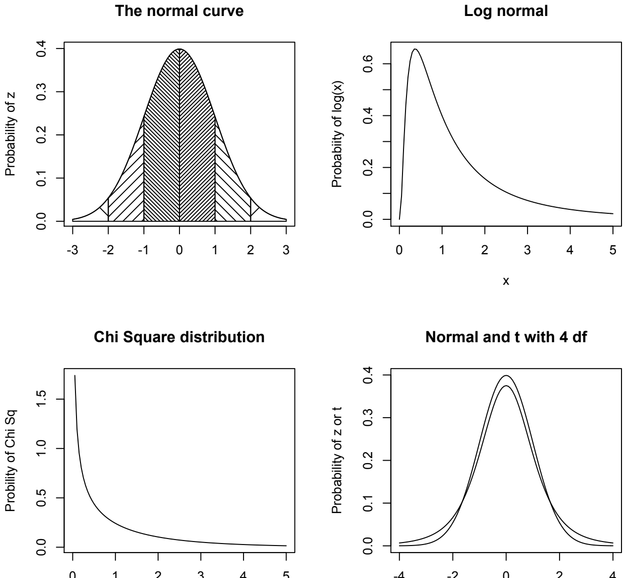
Distribution	base name	P 1	P 2	P 3	example application
Normal	norm	mean	sigma		Most data
Multivariate normal	mvnorm	mean	r	sigma	Most data
Log Normal	lnorm	log mean	log sigma		income or reaction time
Uniform	unif	min	max		rectangular distributions
Binomial	binom	size	prob		Bernuilli trials (e.g. coin flips)
Student's t	t	df		nc	Finding significance of a t-test
Multivariate t	mvt	df	corr	nc	Multivariate applications
Fisher's F	f	df1	df2	nc	Testing for significance of F test
χ^2	chisq	df		nc	Testing for significance of χ^2
Exponential	exp	rate			Exponential decay
Gamma	gamma	shape	rate	scale	distribution theoryh
Hypergeometric	hyper	m	n	k	
Logistic	logis	location	scale		Item Response Theory
Poisson	pois	lambda			Count data
Weibull	weibull	shape	scale		Reaction time distributions

R can draw distributions



curve(dnormal(x),-3,3, ylab=”probability of x”,main=”A normal curve”)

R can draw more interesting distributions

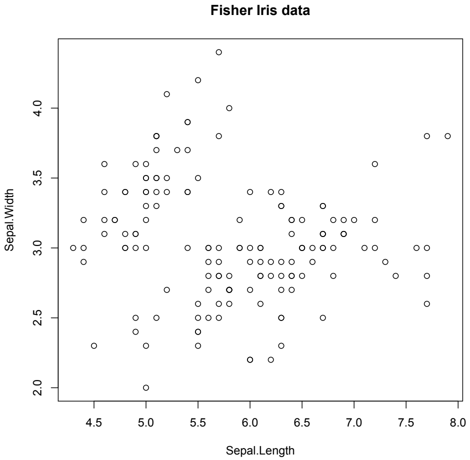


R is also a graphics calculator

The first line draws the normal curve, the second prints the title, the next lines draw the cross hatching.

```
op <- par(mfrow=c(2,2))       #set up a 2 x 2 graph
curve(dnorm(x),-3,3,xlab="",ylab="Probability of z")
title(main="The normal curve",outer=FALSE)
xvals <-  seq(-3,-2,length=100)
dvals <- dnorm(xvals)
polygon(c(xvals,rev(xvals)),c(rep(0,100),rev(dvals)),density=2,angle=-45)
xvals <-  seq(-2,-1,length=100)
dvals <- dnorm(xvals)
polygon(c(xvals,rev(xvals)),c(rep(0,100),rev(dvals)),density=14,angle=45)
xvals <-  seq(-1,-0,length=100)
dvals <- dnorm(xvals)
polygon(c(xvals,rev(xvals)),c(rep(0,100),rev(dvals)),density=34,angle=-45)
xvals <-  seq(2,3,length=100)
dvals <- dnorm(xvals)
polygon(c(xvals,rev(xvals)),c(rep(0,100),rev(dvals)),density=2,angle=45)
xvals <-  seq(1,2,length=100)
dvals <- dnorm(xvals)
polygon(c(xvals,rev(xvals)),c(rep(0,100),rev(dvals)),density=14,angle=-45)
xvals <-  seq(0,1,length=100)
dvals <- dnorm(xvals)
polygon(c(xvals,rev(xvals)),c(rep(0,100),rev(dvals)),density=34,angle=45)

curve(dlnorm(x),0,5,ylab='Probabiity of log(x)',main='Log normal')
curve(dchisq(x,1),0,5,ylab='Probility of Chi Sq',xlab='Chi Sq',main='Chi Square distribution')
curve(dnorm(x),-4,4,ylab='Probability of z or t',xlab='z or t',main='Normal and t with 4 df')
curve(dt(x,4),add=TRUE)
op <- par(mfrow=c(1,1))  #change back to a 1 panel graph
```

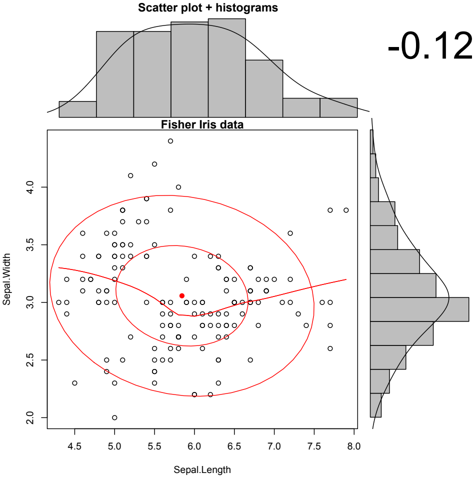
A simple scatter plot using `plot`



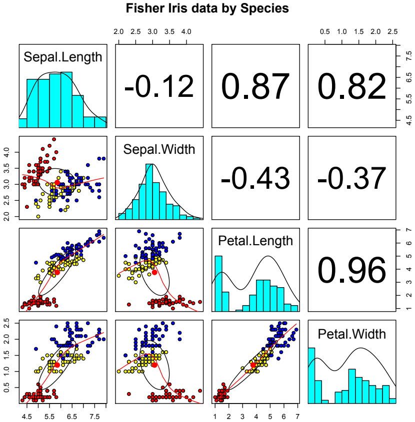

```
plot(iris[1:2],xlab="Sepal.Length",ylab="Sepal.Width"
,main="Fisher Iris data")
```

A somewhat more complex plot

```
scatter.hist(iris[1:2],xlab="Sepal.Length",
ylab="Sepal.Width",main="Fisher Iris data")
```



A scatter plot matrix plot with loess regressions using `pairs.panels`



1. Correlations above the diagonal
2. Diagonal shows histograms and densities
3. scatter plots below the diagonal with correlation ellipse
4. locally smoothed (loess) regressions for each pair
5. optional color coding of grouping variables.

```
pairs.panels(iris[1:4],bg=c("red","yellow","blue")
[iris$Species],pch=21,main="Fisher Iris data by
Species")
```

2 x 2 measures of association

1. Directly enter the data
2. Can test for association using χ^2 or Fisher Exact test
3. Can also measure association using ϕ coefficient
4. With assumption of normality, can apply tetrachoric coefficient

Some basic statistics: measures of association for 2 x 2 tables

Consider the most simple data table: 2 levels of X and 2 levels of Y. Are they associated?

```
Nach <- matrix(c(12,5,5,12),ncol=2,byrow=TRUE) #the data
colnames(Nach) <- c("low","high")
rownames(Nach) <- c("quit","persist")
Nach
chisq.test(Nach)  #the normal Chi Square test
```

```
> Nach
        low high
quit     12    5
persist   5   12


Pearson's Chi-squared test with Yates' continuity correction

data:  Nach
X-squared = 4.2353, df = 1, p-value = 0.03959
```

Another way of looking at the data: Fisher exact test

```
fisher.test(Nach)   #The Fisher exact test

Fisher's Exact Test for Count Data

data:  Nach
p-value = 0.03808
alternative hypothesis: true odds ratio is not equal to 1
95 percent confidence interval:
  1.079216 32.685682
sample estimates:
odds ratio
  5.433516
```

What about the phi measure of association?

Nach
phi(Nach)

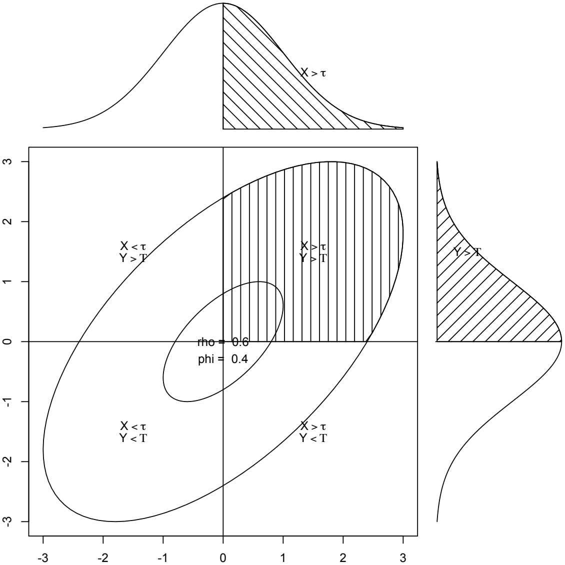
```
> Nach
        low high
quit     12    5
persist   5   12
> phi(Nach)
[1] 0.41
```

If we can assume normality, apply the tetrachoric coefficient

tetrachoric(Nach)

```
> Nach
        low high
quit     12    5
persist   5   12
> phi(Nach)
[1] 0.41
> tetrachoric(Nach)
Call: tetrachoric(x = Nach)
tetrachoric correlation
[1] 0.6

 with tau of
quit  low
   0    0
```

The tetrachoric correlation assumes normality with dichotomous cuts



A brief example with real data

1. Get the data
2. Descriptive statistics
   - Graphic
   - Numerical
3. Inferential statistics using the linear model
   - regressions
4. More graphic displays

Get the data and describe it

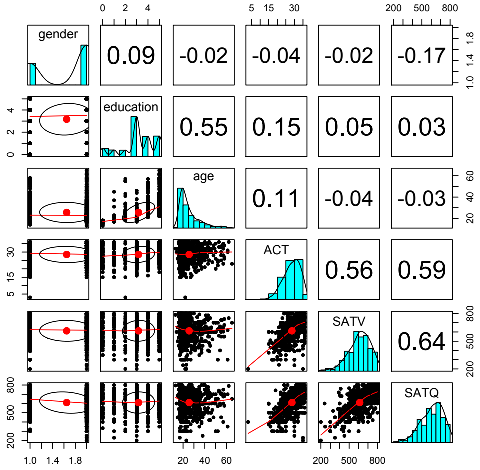
1. First read the data, either from a built in data set, a local file, a remote file, or from the clipboard.
2. Describe the data using the `describe` function from *psych*

```
> my.data <- sat.act  #an example data file that is part of psych
#or
> file.name <- file.choose()      #look for it on your hard drive
#or
> file.name <-"http://personality-project.org/r/aps/sat.act.txt"
#now read it  from this remote site
> my.data <- read.table(file.name,header=TRUE)
#or
> my.data <- read.clipboard()  #if you have copied the data to the clipboard
> describe(my.data) #report basic descriptive statistics
```

	var	n	mean	sd	median	trimmed	mad	min	max	range	skew	kurto
gender	1	700	1.65	0.48	2	1.68	0.00	1	2	1	-0.61	-1
education	2	700	3.16	1.43	3	3.31	1.48	0	5	5	-0.68	-0
age	3	700	25.59	9.50	22	23.86	5.93	13	65	52	1.64	2
ACT	4	700	28.55	4.82	29	28.84	4.45	3	36	33	-0.66	0
SATV	5	700	612.23	112.90	620	619.45	118.61	200	800	600	-0.64	0
SATQ	6	687	610.22	115.64	620	617.25	118.61	200	800	600	-0.59	0

Graphic display of data using pairs.panels

pairs.panels(my.data) #Note the outlier for ACT



Clean up the data using `scrub`

`scrub` allows you to recode and/or delete cases that meet certain criteria.

```
> cleaned <- scrub(my.data,"ACT",min=4)
> describe(cleaned)
```

	var	n	mean	sd	median	trimmed	mad	min	max	range	skew	kurto
gender	1	700	1.65	0.48	2	1.68	0.00	1	2	1	-0.61	-1
education	2	700	3.16	1.43	3	3.31	1.48	0	5	5	-0.68	-0
age	3	700	25.59	9.50	22	23.86	5.93	13	65	52	1.64	2
ACT	4	699	28.58	4.73	29	28.85	4.45	15	36	21	-0.50	-0
SATV	5	700	612.23	112.90	620	619.45	118.61	200	800	600	-0.64	0
SATQ	6	687	610.22	115.64	620	617.25	118.61	200	800	600	-0.59	0

By making that one data point NA, we have changed the range of ACT significantly.

Find the pairwise correlations, round to 2 decimals

```
> round(cor(cleaned,use="pairwise"),2)

          gender education   age   ACT  SATV  SATQ
gender      1.00      0.09 -0.02 -0.05 -0.02 -0.17
education   0.09      1.00  0.55  0.15  0.05  0.03
age        -0.02      0.55  1.00  0.11 -0.04 -0.03
ACT        -0.05      0.15  0.11  1.00  0.55  0.59
SATV       -0.02      0.05 -0.04  0.55  1.00  0.64
SATQ       -0.17      0.03 -0.03  0.59  0.64  1.00
```

Test the correlations for significance using `corr.test`

```
> corr.test(cleaned)
Call:corr.test(x = cleaned)
Correlation matrix
          gender education   age   ACT  SATV  SATQ
gender      1.00      0.09 -0.02 -0.05 -0.02 -0.17
education   0.09      1.00  0.55  0.15  0.05  0.03
age        -0.02      0.55  1.00  0.11 -0.04 -0.03
ACT        -0.05      0.15  0.11  1.00  0.55  0.59
SATV       -0.02      0.05 -0.04  0.55  1.00  0.64
SATQ       -0.17      0.03 -0.03  0.59  0.64  1.00
Sample Size
          gender education age ACT SATV SATQ
gender       700       700 700 699  700  687
...
SATQ         687       687 687 686  687  687
Probability value
          gender education  age  ACT SATV SATQ
gender      0.00      0.02 0.58 0.21 0.62 0.00
education   0.02      0.00 0.00 0.00 0.22 0.36
age         0.58      0.00 0.00 0.00 0.26 0.37
ACT         0.21      0.00 0.00 0.00 0.00 0.00
SATV        0.62      0.22 0.26 0.00 0.00 0.00
SATQ        0.00      0.36 0.37 0.00 0.00 0.00
```

Are education and gender independent? χ^2 Test of association

$$T <- with(my.data,table(gender,education))$$

```
> T
       education
gender   0   1   2   3   4   5
     1  27  20  23  80  51  46
     2  30  25  21 195  87  95

> chisq.test(T)
Pearson's Chi-squared test

data:  T
X-squared = 16.0851, df = 5, p-value = 0.006605
```

1. First create a table of associations
   - Do this on our data (my.data)
   - Use the "with" command to specify the data set
2. Show the table
3. Apply χ^2 test

Multiple regression

1. Use the sat.act data example
2. Do the linear model
3. Summarize the results

```
 mod1 <- lm(SATV ~ education + gender + SATQ,data=my.data)
> summary(mod1,digits=2)
Call:
lm(formula = SATV ~ education + gender + SATQ, data = my.data)
Residuals:
    Min      1Q  Median      3Q     Max
-372.91  -49.08    2.30   53.68  251.93
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 180.87348   23.41019   7.726 3.96e-14 ***
education     1.24043    2.32361   0.534  0.59363
gender       20.69271    6.99651   2.958  0.00321 **
SATQ          0.64489    0.02891  22.309  < 2e-16 ***
Signif. codes:  0 Ô***Õ 0.001 Ô**Õ 0.01 Ô*Õ 0.05 Ô.Õ 0.1 Ô Õ 1
Residual standard error: 86.24 on 683 degrees of freedom
  (13 observations deleted due to missingness)
Multiple R-squared: 0.4231, Adjusted R-squared: 0.4205
F-statistic:   167 on 3 and 683 DF,  p-value: < 2.2e-16
```

Zero center the data before examining interactions

In order to examine interactions using multiple regression, we must first "zero center" the data. This may be done using the `scale` function. By default, `scale` will standardize the variables. So to keep the original metric, we make the scaling parameter FALSE.

```
censat <- data.frame(scale(my.data,scale=FALSE))
describe(censat)
```

```
          var   n mean     sd median trimmed    mad     min    max range  skew 
gender      1 700    0   0.48   0.35    0.04   0.00   -0.65   0.35     1 -0.61
education   2 700    0   1.43  -0.16    0.14   1.48   -3.16   1.84     5 -0.68
age         3 700    0   9.50  -3.59   -1.73   5.93  -12.59  39.41    52  1.64
ACT         4 700    0   4.82   0.45    0.30   4.45  -25.55   7.45    33 -0.66
SATV        5 700    0 112.90   7.77    7.22 118.61 -412.23 187.77   600 -0.64
SATQ        6 687    0 115.64   9.78    7.04 118.61 -410.22 189.78   600 -0.59
```

Variable names are arbitrary but it is useful to give them some mnemonic value.

Zero center the data before examining interactions

```
> censat <- data.frame(scale(my.data,scale=FALSE))
> mod2 <-  lm(SATV  ~ education * gender * SATQ,data=censat)
> summary(mod2)
Call:
lm(formula = SATV ~ education * gender * SATQ, data =censat)

Residuals:
    Min      1Q  Median      3Q     Max
-372.53  -48.76    3.33   51.24  238.50

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)            0.773576   3.304938   0.234  0.81500
education              2.517314   2.337889   1.077  0.28198
gender                18.485906   6.964694   2.654  0.00814 **
SATQ                   0.620527   0.028925  21.453  < 2e-16 ***
education:gender       1.249926   4.759374   0.263  0.79292
education:SATQ        -0.101444   0.020100  -5.047 5.77e-07 ***
gender:SATQ            0.007339   0.060850   0.121  0.90404
education:gender:SATQ  0.035822   0.041192   0.870  0.38481
---
Signif. codes:  0 Ô***Õ 0.001 Ô**Õ 0.01 Ô*Õ 0.05 Ô.Õ 0.1 Ô Õ 1
```

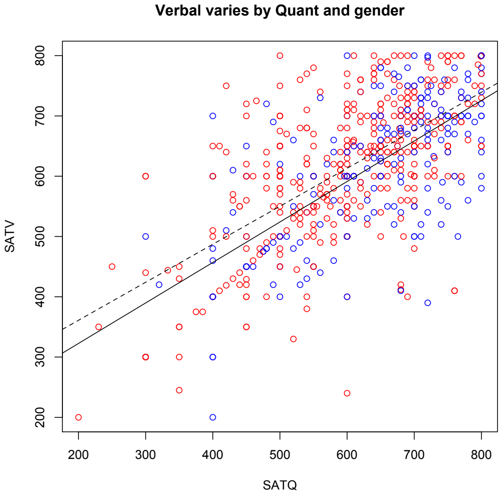
Compare model 1 and model 2

Test the difference between the two linear models

```
> anova(mod1,mod2)

Analysis of Variance Table

Model 1: SATV ~ education + gender + SATQ
Model 2: SATV ~ education * gender * SATQ
  Res.Df     RSS Df Sum of Sq      F    Pr(>F)
1    683 5079984
2    679 4870243  4    209742 7.3104 9.115e-06 ***
---
Signif. codes:  0 Ô***Õ 0.001 Ô**Õ 0.01 Ô*Õ 0.05 Ô.Õ 0.1 Ô
```

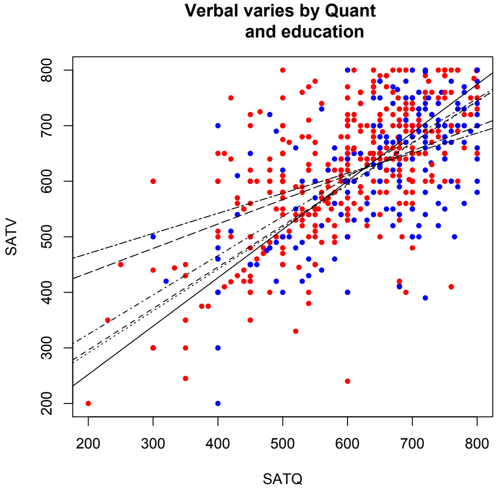
The model, although more complicated is significantly better when the change of degrees of freedom is considered.

Show the regression lines by gender




```
> with(my.data,plot(SATV~SATQ,
    col=c("blue","red")[gender]))
> by(my.data,my.data$gender,
     function(x) abline
          (lm(SATV~SATQ,data=x),
          lty=c("solid","dashed")[
> title("Verbal varies by Quant
         and gender")
```

Show the regression lines by education




```
> with(my.data,plot(SATV~SATQ,
    col=c("blue","red")[gender]))
by(my.data,my.data$education,
   function(x) abline   (lm(SATV~S
    lty=c("solid", "dashed", "dott
    "dotdash", "longdash",
      "twodash")[(x$education+1)]

> title("Verbal varies by Quant
          and education")
```

Using R for psychological statistics: Basic statistics

1. Writing syntax
   - For a single line, just type it
   - Mistakes can be redone by using the up arrow key
   - For longer code, use a text editor (built into some GUIs)
2. Data entry
   - Using built in data sets for examples
   - Copying and pasting from another program
   - Reading a text or csv file
   - Importing from SPSS or SAS
   - Simulate it (using various simulation routines)
3. Descriptives
   - Graphical displays
   - Descriptive statistics
   - Correlation
4. Inferential
   - the t test
   - the F test
   - the linear model

Data entry overview

1. Using built in data sets for examples
   - `data()` will list > 100 data sets in the `datasets` package as well as all sets in loaded packages.
   - Most packages have associated data sets used as examples
   - *psych* has > 40 example data sets
2. Copying from another program
   - use copy and paste into R using `read.clipboard` and its variations
3. Reading a text or csv file
   - read a local or remote file
4. Importing from SPSS or SAS
5. Simulate it (using various simulation routines)

Examples of built in data sets from the psych package

```
> data(package="psych")
Bechtoldt                Seven data sets showing a bifactor solution.
Dwyer                    8 cognitive variables used by Dwyer for an examp
Reise                    Seven data sets showing a bifactor solution.
all.income (income)      US family income from US census 2008
bfi                      25 Personality items representing 5 factors
blot                     Bond's Logical Operations Test - BLOT
burt                     11 emotional variables from Burt (1915)
cities                   Distances between 11 US cities
epi.bfi                  13 personality scales from the Eysenck Personali
                         and Big 5 inventory
flat (affect)            Two data sets of affect and arousal scores as a
                         personality and movie conditions
galton                   Galton's Mid parent child height data
income                   US family income from US census 2008
iqitems                  14 multiple choice IQ items
msq                      75 mood items from the Motivational State Questi
                         3896 participants
neo                      NEO correlation matrix from the NEO_PI_R manual
sat.act                  3 Measures of ability: SATV, SATQ, ACT
Thurstone                Seven data sets showing a bifactor solution.
veg (vegetables)         Paired comparison of preferences for 9 vegetable
```

Reading data from another program –using the clipboard

1. Read the data in your favorite spreadsheet or text editor
2. Copy to the clipboard
3. Execute the appropriate `read.clipboard` function with or without various options specified

```
my.data <- read.clipboard()   #assumes headers and tab or space delimited
                                          but with no missing cells
my.data <- read.clipboard.csv()   #assumes headers and comma delimited
my.data <- read.clipboard.tab()   #assumes headers and tab delimited
                                            (e.g., from Excel)
my.data <- read.clipboard.lower()  #read in a matrix given the lower
my.data <- read.clipboard.upper() #  or upper  off diagonal
my.data <- read.clipboard.fwf()  #read in data using a fixed format width
                                              (see read.fwf for instructions)
```

4. `read.clipboard()` has default values for the most common cases and these do not need to be specified. Consult ?read.clipboard for details.

Reading from a local or remote file

1. Perhaps the standard way of reading in data is using the `read.table` command.
   - First must specify the location of the file
   - Can either type this in directly or use the `file.choose` function
   - The file name/location can be a remote URL
2. Two examples of reading data

```
file.name <- file.choose() #this opens a window to allow you find the file
my.data <- read.table(file.name)
datafilename="http://personality-project.org/r/datasets/R.appendix1.data"
data.ex1=read.table(datafilename,header=TRUE)   #read the data into a table
```

```
> dim(data.ex1)  #what are the dimensions of what we read?
[1] 18  2
> describe(data.ex1) #do the data look right?
          var  n  mean   sd median trimmed  mad min max range skew kurtosi
Dosage*     1 18  1.89 0.76      2    1.88 1.48   1   3     2 0.16    -1.1
Alertness   2 18 27.67 6.82     27   27.50 8.15  17  41    24 0.25    -0.6
```

read a "foreign" file e.g., an SPSS sav file

`read.spss` reads a file stored by the SPSS save or export commands.

```
read.spss(file, use.value.labels = TRUE, to.data.frame = FALSE,
          max.value.labels = Inf, trim.factor.names = FALSE,
          trim_values = TRUE, reencode = NA, use.missings = to.data.frame)
```

file	Character string: the name of the file or URL to read.
use.value.labels	Convert variables with value labels into R factors with those levels?
to.data.frame	return a data frame? Defaults to FALSE, probably should be TRUE in most cases.
max.value.labels	Only variables with value labels and at most this many unique values will be converted to factors if use.value.labels = *TRUE*.
trim.factor.names	Logical: trim trailing spaces from factor levels?
trim_values	logical: should values and value labels have trailing spaces ignored when matching for use.value.labels = *TRUE*?
use.missings	logical: should information on user-defined missing values be used to set the corresponding values to NA?

Simulate data

For many demonstration purposes, it is convenient to generate simulated data with a certain defined structure. the *psych* package has a number of built in simulation functions. Here are a few of them.

1. Simulate various item structures
   - sim.congeneric A one factor congeneric measure model
   - sim.items A two factor structure with either simple structure or a circumplex structure.
   - sim.rasch Generate items for a one parameter IRT model.
   - sim.irt Generate items for a one-four parameter IRT Model
2. Simulate various factor structures
   - sim.simplex Default is a four factor structure with a three time point simplex structure.
   - sim.hierarchical Default is 9 variables with three correlated factors.

Get the data and look at it

Read in some data, look at the first and last few cases, and then get basic descriptive statistics. For this example, we will use a built in data set (EPI and Big 5 inventory data).
The `headtail` function shows the `head` and the `tail` of the data.

```
> my.data <- epi.bfi
> headtail(my.data)
```

	epiE	epiS	epiImp	epilie	epiNeur	bfagree	bfcon	bfext	bfneur	bfopen	bdi	traitanx	stateanx
1	18	10	7	3	9	138	96	141	51	138	1	24	22
2	16	8	5	1	12	101	99	107	116	132	7	41	40
3	6	1	3	2	5	143	118	38	68	90	4	37	44
4	12	6	4	3	15	104	106	64	114	101	8	54	40
...	...	...	...	...	...	...	...	...	...	...	...	...	...
228	12	7	4	3	15	155	129	127	88	110	9	35	34
229	19	10	7	2	11	162	152	163	104	164	1	29	47
230	4	1	1	2	10	95	111	75	123	138	5	39	58
231	8	6	3	2	15	85	62	90	131	96	24	58	58

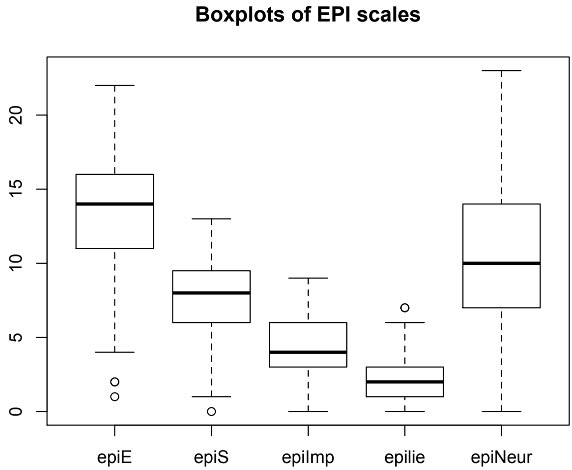
epi.bfi has 231 cases from two personality measures

Now find the descriptive statistics for this data set

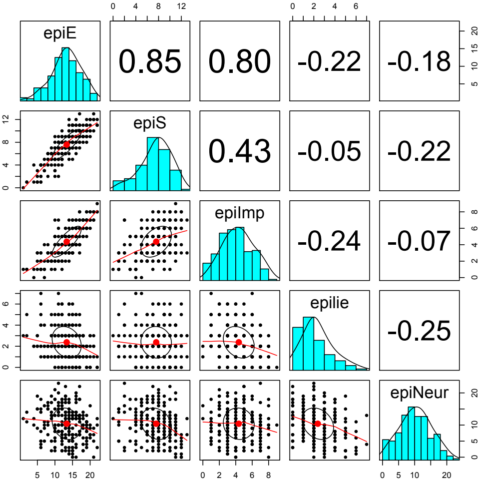
```
> describe(my.data)

         var   n   mean    sd median trimmed   mad min max range  skew kurtosis
epiE       1 231  13.33  4.14     14   13.49  4.45   1  22    21 -0.33    -0.01
epiS       2 231   7.58  2.69      8    7.77  2.97   0  13    13 -0.57     0.04
epiImp     3 231   4.37  1.88      4    4.36  1.48   0   9     9  0.06    -0.59
epilie     4 231   2.38  1.50      2    2.27  1.48   0   7     7  0.66     0.30
epiNeur    5 231  10.41  4.90     10   10.39  4.45   0  23    23  0.06    -0.46
bfagree    6 231 125.00 18.14    126  125.26 17.79  74 167    93 -0.21    -0.22
bfcon      7 231 113.25 21.88    114  113.42 22.24  53 178   125 -0.02     0.29
bfext      8 231 102.18 26.45    104  102.99 22.24   8 168   160 -0.41     0.58
bfneur     9 231  87.97 23.34     90   87.70 23.72  34 152   118  0.07    -0.51
bfopen    10 231 123.43 20.51    125  123.78 20.76  73 173   100 -0.16    -0.11
bdi       11 231   6.78  5.78      6    5.97  4.45   0  27    27  1.29     1.60
traitanx  12 231  39.01  9.52     38   38.36  8.90  22  71    49  0.67     0.54
stateanx  13 231  39.85 11.48     38   38.92 10.38  21  79    58  0.72     0.04
```

Boxplots are a convenient descriptive device

Show the Tukey "boxplot" for the Eysenck Personality Inventory



Plot the scatter plot matrix (SPLOM) of the first 5 variables using the `pairs.panels`function



Use the `pairs.panels` function from *psych*

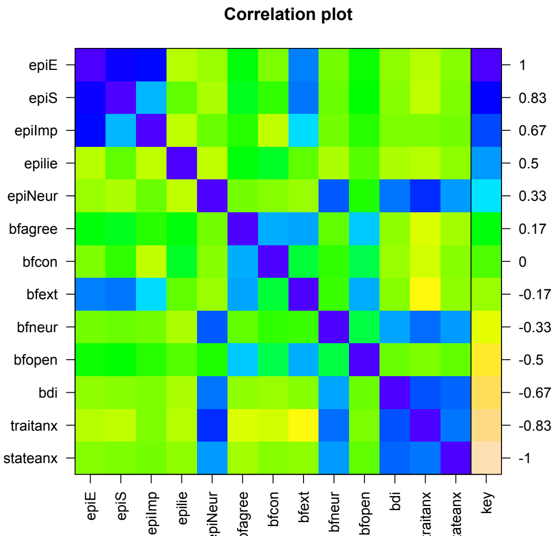
```
pairs.panels(my.data[1:5])
```

Find the correlations for this data set, round off to 2 decimal places

```
> round(cor(my.data, use = "pairwise"), 2)

          epiE  epiS epiImp epilie epiNeur bfagree bfcon bfext bfneur bfopen
epiE      1.00  0.85   0.80  -0.22   -0.18    0.18 -0.11  0.54  -0.09   0.14 -0
epiS      0.85  1.00   0.43  -0.05   -0.22    0.20  0.05  0.58  -0.07   0.15 -0
epiImp    0.80  0.43   1.00  -0.24   -0.07    0.08 -0.24  0.35  -0.09   0.07 -0
epilie   -0.22 -0.05  -0.24   1.00   -0.25    0.17  0.23 -0.04  -0.22  -0.03 -0
epiNeur  -0.18 -0.22  -0.07  -0.25    1.00   -0.08 -0.13 -0.17   0.63   0.09  0
bfagree   0.18  0.20   0.08   0.17   -0.08    1.00  0.45  0.48  -0.04   0.39 -0
bfcon    -0.11  0.05  -0.24   0.23   -0.13    0.45  1.00  0.27   0.04   0.31 -0
bfext     0.54  0.58   0.35  -0.04   -0.17    0.48  0.27  1.00   0.04   0.46 -0
bfneur   -0.09 -0.07  -0.09  -0.22    0.63   -0.04  0.04  0.04   1.00   0.29  0
bfopen    0.14  0.15   0.07  -0.03    0.09    0.39  0.31  0.46   0.29   1.00 -0
bdi      -0.16 -0.13  -0.11  -0.20    0.58   -0.14 -0.18 -0.14   0.47  -0.08  1
traitanx -0.23 -0.26  -0.12  -0.23    0.73   -0.31 -0.29 -0.39   0.59  -0.11  0
stateanx -0.13 -0.12  -0.09  -0.15    0.49   -0.19 -0.14 -0.15   0.49  -0.04  0
```

cor.plot the correlation matrix to visually detect patterns

```
cor.plot(cor(my.data,use="complete"),colors=TRUE,n=100)
```



t.test demonstration with Student's original data

sleep

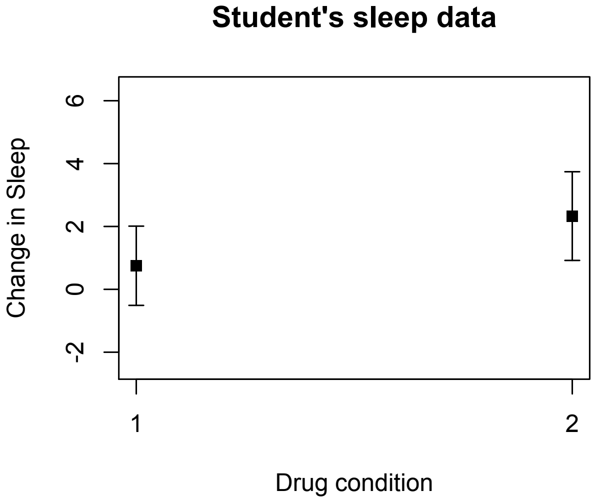
```
> sleep
   extra group ID
1    0.7     1  1
2   -1.6     1  2
3   -0.2     1  3
4   -1.2     1  4
5   -0.1     1  5
6    3.4     1  6
7    3.7     1  7
...
13   1.1     2  3
14   0.1     2  4
15  -0.1     2  5
16   4.4     2  6
17   5.5     2  7
18   1.6     2  8
19   4.6     2  9
20   3.4     2 10
```

```
> with(sleep,t.test(extra~group))

Welch Two Sample t-test
data:  extra by group
t = -1.8608, df = 17.776, p-value = 0.07939
alternative hypothesis: true difference in means is not equal t
95 percent confidence interval:
 -3.3654832  0.2054832
sample estimates:
mean in group 1 mean in group 2
           0.75            2.33
```

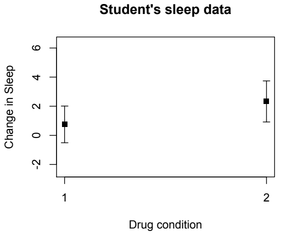
But the data were actually paired. Do it for a paired t-test

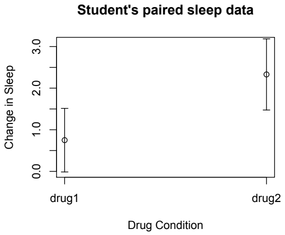
```
> with(sleep,t.test(extra~group,paired=TRUE))

Paired t-test
data:  extra by group
t = -4.0621, df = 9, p-value = 0.002833
alternative hypothesis: true difference in means is not equal t
95 percent confidence interval:
 -2.4598858 -0.7001142
sample estimates:
mean of the differences
                  -1.58
```

Two ways of showing Student’s t test data



Two ways of showing Student's t test data





Use the `error.bars.by` and `error.bars` functions. Note that we need to change the data structure a little bit to get the within subject error bars.

```
> error.bars.by(sleep$extra,sleep$group,
     by.var=TRUE,  lines=FALSE,
     ylab="Change in Sleep", xlab="Drug
     condition",main="Student's sleep data")
```

```
> error.bars(data.frame(drug1=sleep[1:10,1],
       drug2=sleep[11:20,1]), within=TRUE,
        ylab="Change in Sleep"
        ,xlab="Drug Condition",
         main="Student's paired sleep data")
```

Analysis of Variance

1. aov is designed for balanced designs, and the results can be hard to interpret without balance: beware that missing values in the response(s) will likely lose the balance.
2. If there are two or more error strata, the methods used are statistically inefficient without balance, and it may be better to use `lme` in package *nlme*.

```
datafilename="http://personality-project.org/R/datasets/R.appendix2.data"
data.ex2=read.table(datafilename,header=T)   #read the data into a table
data.ex2                                      #show the data
```

```
 data.ex2                                      #show the data
   Observation Gender Dosage Alertness
1            1      m      a         8
2            2      m      a        12
3            3      m      a        13
4            4      m      a        12
...
14          14      f      b        12
15          15      f      b        18
16          16      f      b        22
```

Analysis of Variance

1. do the analysis of variances and the show the table of results

```
aov.ex2 = aov(Alertness~Gender*Dosage,data=data.ex2)          #do the analysis of
summary(aov.ex2)                                             #show the summary table
```

```
 > aov.ex2 = aov(Alertness~Gender*Dosage,data=data.ex2)         #do the analysi
> summary(aov.ex2)                                            #show the summary table
              Df  Sum Sq Mean Sq F value Pr(>F)
Gender         1  76.562  76.562  2.9518 0.1115
Dosage         1   5.062   5.062  0.1952 0.6665
Gender:Dosage  1   0.063   0.063  0.0024 0.9617
```

Show the results table

```
> print(model.tables(aov.ex2,"means"),digits=3)
Residuals      12 311.250   25.938

Tables of means
Grand mean
14.0625
 Gender
Gender
    f     m
16.25 11.88

 Dosage
Dosage
    a     b
13.50 14.62

 Gender:Dosage
      Dosage
Gender a     b
     f 15.75 16.75
     m 11.25 12.50
```

Analysis of Variance: Within subjects

1. Somewhat more complicated because we need to convert “wide” data.frames to “long” or “narrow” data.frame.
2. This can be done by using the `stack` function. Some data sets are already in the long format.
3. A detailed discussion of how to work with repeated measures designs is at http://personality-project.org/r/r.anova.html and at http://personality-project.org/r

Analysis of variance within subjects

First get the data from a remote file

```
> datafilename="http://personality-project.org/r/datasets/R.appendix5.data"
> data.ex5=read.table(datafilename,header=T)    #read the data into a table
> data.ex5[c(1:3,20:24,60:64,105:108),]                                           #sho
```

```
> data.ex5[c(1:3,22:24,62:64,105:108),]
    Obs Subject Gender Dosage Task Valence Recall
    Obs Subject Gender Dosage Task Valence Recall
1     1       A      M      A    F     Neg      8
2     2       A      M      A    F     Neu      9
3     3       A      M      A    F     Pos      5
22   22       D      M      B    C     Neg     17
23   23       D      M      B    C     Neu     18
24   24       D      M      B    C     Pos     20
62   62       K      F      A    F     Neu     20
63   63       K      F      A    F     Pos     23
64   64       K      F      A    C     Neg     25
105 105       R      F      C    F     Pos     19
106 106       R      F      C    C     Neg     22
107 107       R      F      C    C     Neu     21
108 108       R      F      C    C     Pos     20
```

Analysis of variance within subjects

```
> datafilename="http://personality-project.org/r/datasets/R.appendix5.data"
> data.ex5=read.table(datafilename,header=T)   #read the data into a table
> #data.ex5                                     #show the data
> aov.ex5 =
+ aov(Recall~(Task*Valence*Gender*Dosage)+Error(Subject/(Task*Valence))+
+ (Gender*Dosage),data.ex5)
> summary(aov.ex5)
Error: Subject
              Df  Sum Sq Mean Sq F value  Pr(>F)
Gender         1  542.26  542.26  5.6853 0.03449 *
Dosage         2  694.91  347.45  3.6429 0.05803 .
Gender:Dosage  2   70.80   35.40  0.3711 0.69760
Residuals     12 1144.56   95.38
Signif. codes:  0 Ô***Õ 0.001 Ô**Õ 0.01 Ô*Õ 0.05 Ô.Õ 0.1 Ô Õ 1

Error: Subject:Task
                   Df Sum Sq Mean Sq F value    Pr(>F)
Task                1 96.333  96.333 39.8621 3.868e-05 ***
Task:Gender         1  1.333   1.333  0.5517    0.4719
Task:Dosage         2  8.167   4.083  1.6897    0.2257
Task:Gender:Dosage  2  3.167   1.583  0.6552    0.5370
Residuals          12 29.000   2.417
... (lots more)
```

Multiple regression

1. Use the sat.act data set from *psych*
2. Do the linear model
3. Summarize the results

```
 mod1 <- lm(SATV ~ education + gender + SATQ,data=sat.act)
> summary(mod1,digits=2)
Call:
lm(formula = SATV ~ education + gender + SATQ, data = sat.act)
Residuals:
    Min      1Q  Median      3Q     Max
-372.91  -49.08    2.30   53.68  251.93
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 180.87348   23.41019   7.726 3.96e-14 ***
education     1.24043    2.32361   0.534  0.59363
gender       20.69271    6.99651   2.958  0.00321 **
SATQ          0.64489    0.02891  22.309  < 2e-16 ***
Signif. codes:  0 Ô***Õ 0.001 Ô**Õ 0.01 Ô*Õ 0.05 Ô.Õ 0.1 Ô Õ 1
Residual standard error: 86.24 on 683 degrees of freedom
  (13 observations deleted due to missingness)
Multiple R-squared: 0.4231, Adjusted R-squared: 0.4205
F-statistic:   167 on 3 and 683 DF,  p-value: < 2.2e-16
```

Zero center the data before examining interactions

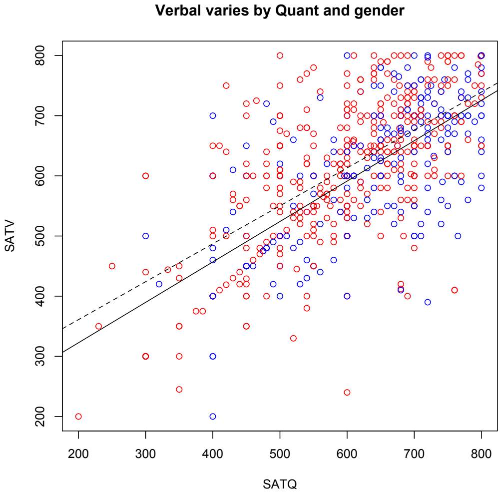
```
> zsat <- data.frame(scale(sat.act,scale=FALSE))
> mod2 <-  lm(SATV  ~ education * gender * SATQ,data=zsat)
> summary(mod2)
Call:
lm(formula = SATV ~ education * gender * SATQ, data = zsat)

Residuals:
    Min      1Q  Median      3Q     Max
-372.53  -48.76    3.33   51.24  238.50

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)            0.773576   3.304938   0.234  0.81500
education              2.517314   2.337889   1.077  0.28198
gender                18.485906   6.964694   2.654  0.00814 **
SATQ                   0.620527   0.028925  21.453  < 2e-16 ***
education:gender       1.249926   4.759374   0.263  0.79292
education:SATQ        -0.101444   0.020100  -5.047 5.77e-07 ***
gender:SATQ            0.007339   0.060850   0.121  0.90404
education:gender:SATQ  0.035822   0.041192   0.870  0.38481
---
Signif. codes:  0 Ô***Õ 0.001 Ô**Õ 0.01 Ô*Õ 0.05 Ô.Õ 0.1 Ô Õ 1
```

Compare model 1 and model 2

Test the difference between the two linear models

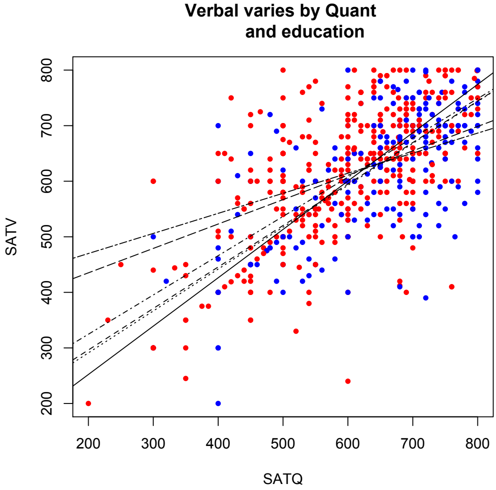
```
> anova(mod1,mod2)

Analysis of Variance Table

Model 1: SATV ~ education + gender + SATQ
Model 2: SATV ~ education * gender * SATQ
  Res.Df     RSS Df Sum of Sq      F    Pr(>F)
1    683 5079984
2    679 4870243  4    209742 7.3104 9.115e-06 ***
---
Signif. codes:  0 Ô***Õ 0.001 Ô**Õ 0.01 Ô*Õ 0.05 Ô.Õ 0.1 Ô
```

Show the regression lines by gender




```
> with(sat.act,plot(SATV~SATQ,
    col=c("blue","red")[gender]))
> by(sat.act,sat.act$gender,
     function(x) abline
           (lm(SATV~SATQ,data=x),
           lty=c("solid","dashed")
> title("Verbal varies by Quant
          and gender")
```

Show the regression lines by education




```
# Show an interaction

> with(my.data,plot(SATV~SATQ,
    col=c("blue","red")[gender]))
by(my.data,my.data$education,
 function(x) abline   (lm(SATV~SATQ,data=x),
 lty=c("solid", "dashed", "dotted",
  "dotdash", "longdash",
   "twodash")[(x$education+1)]))

> title("Verbal varies by Quant
        and education")
```

- Part I: an introduction to R
- Part II: Using R for psychometrics
- Part III: Structures, Objects, Functions

Outline of Part II: Psychometrics and beyond

Classic theory estimates of reliability

1. Scoring tests
   - score.items Score 1-n scales using a set of keys and finding the simple sum or average of items. Reversed items are indicated by -1
   - score.multipe.choice : Score multiple choice items by first converting to 0 or 1 and then proceeding to score the items.
2. Alternative estimates of reliability
   - alpha α reliability of a single scale finds the average split half reliability. (some items may be reversed keyed).
   - omega ω_h reliability of a single scale estimates the general factor saturation of the test.
   - guttman Find the 6 Guttman reliability estimates

Using `score.items` to score 25 Big 5 items (taken from the bfi example

```
#first create a list of items to score
>  keys.list <- list(Agree=c(-1,2:5),Conscientious=c(6:8,-9,-10),Extraversion=c(-11,-12,13:15),
        Neuroticism=c(16:20),Openness = c(21,-22,23,24,-25))
>  keys <- make.keys(28,keys.list,item.labels=colnames(bfi)) #create the keys list
>  bfi.scores <- score.items(keys,bfi)  #use this list to score the items
> bfi.scores   #show the statistics
Call: score.items(keys = keys, items = bfi)

(Unstandardized) Alpha:
      Agree Conscientious Extraversion Neuroticism Openness
alpha   0.7          0.72         0.76        0.81      0.6

Average item correlation:
          Agree Conscientious Extraversion Neuroticism Openness
average.r  0.32          0.34         0.39        0.46     0.23

 Guttman 6* reliability:
         Agree Conscientious Extraversion Neuroticism Openness
Lambda.6   0.7          0.72         0.76        0.81      0.6

Scale intercorrelations corrected for attenuation
 raw correlations below the diagonal, alpha on the diagonal
 corrected correlations above the diagonal:
              Agree Conscientious Extraversion Neuroticism Openness
Agree          0.70          0.36         0.63      -0.245     0.23
Conscientious  0.26          0.72         0.35      -0.305     0.30
Extraversion   0.46          0.26         0.76      -0.284     0.32
Neuroticism   -0.18         -0.23        -0.22       0.812    -0.12
Openness       0.15          0.19         0.22      -0.086     0.60
```

score.items output, continued

```
Item by scale correlations:
 corrected for item overlap and scale reliability
         Agree Conscientious Extraversion Neuroticism Openness
A1       -0.40         -0.06        -0.11        0.14    -0.14
A2        0.67          0.23         0.40       -0.07     0.17
A3        0.70          0.22         0.48       -0.11     0.17
A4        0.49          0.29         0.30       -0.14     0.01
A5        0.62          0.23         0.55       -0.23     0.18
C1        0.13          0.53         0.19       -0.08     0.28
C2        0.21          0.61         0.17        0.00     0.20
C3        0.21          0.54         0.14       -0.09     0.08
C4       -0.24         -0.66        -0.23        0.31    -0.23
C5       -0.26         -0.59        -0.29        0.36    -0.10
E1       -0.30         -0.06        -0.59        0.11    -0.16
E2       -0.39         -0.25        -0.70        0.34    -0.15
E3        0.44          0.20         0.60       -0.10     0.37
E4        0.51          0.23         0.68       -0.22     0.04
E5        0.34          0.40         0.55       -0.10     0.31
N1       -0.22         -0.21        -0.11        0.76    -0.12
N2       -0.22         -0.19        -0.12        0.74    -0.06
N3       -0.14         -0.20        -0.14        0.74    -0.03
N4       -0.22         -0.30        -0.39        0.62    -0.02
N5       -0.04         -0.14        -0.19        0.55    -0.18
O1        0.16          0.20         0.31       -0.09     0.52
O2       -0.01         -0.18        -0.07        0.19    -0.45
O3        0.26          0.20         0.42       -0.07     0.61
O4        0.06         -0.02        -0.10        0.21     0.32
O5       -0.09         -0.14        -0.11        0.11    -0.53
gender    0.25          0.11         0.12        0.14    -0.07
education 0.06          0.03         0.01       -0.06     0.13
age       0.22          0.14         0.07       -0.13     0.10
```

Comment about the output from a function

1. Many functions produce far more output than you normally want
   - The package developer typically has decided what is most informative
   - This will be shown by displaying the function's output
   - Sometimes you will want to `summary`(x) the function output
2. To access the other elements of the results you can
   - Look at the help file for the function where the output is discussed.
   - Or, just look at the `str` of the result.
3. For instance, the score.items function reports frequently used statistics, but hides the actual scores.
   - To access an element of an object (e.g. the scores of bfi.scores), use the $ or the [[name]] feature.

The structure of an object

```
> str(bfi.scores)
List of 13
 $ scores       : num [1:2800, 1:5] 20 20.2 19.8 20.6 20 20.6 20.6 18.6 19.6 21.4 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:2800] "61617" "61618" "61620" "61621" ...
  .. ..$ : chr [1:5] "Agree" "Conscientious" "Extraversion" "Neuroticism" ...
 $ missing      : num [1:2800, 1:5] 0 0 0 0 0 0 0 0 0 0 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:2800] "61617" "61618" "61620" "61621" ...
  .. ..$ : chr [1:5] "Agree" "Conscientious" "Extraversion" "Neuroticism" ...
 $ alpha        : num [1, 1:5] 0.701 0.725 0.76 0.812 0.597
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr "alpha"
  .. ..$ : chr [1:5] "Agree" "Conscientious" "Extraversion" "Neuroticism" ...
 $ av.r         : num [1, 1:5] 0.319 0.345 0.387 0.463 0.228
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr "average.r"
  .. ..$ : chr [1:5] "Agree" "Conscientious" "Extraversion" "Neuroticism" ...
 $ n.items      : Named num [1:5] 5 5 5 5 5
  ..- attr(*, "names")= chr [1:5] "Agree" "Conscientious" "Extraversion" "Neuroticism" ...
 $ item.cor     : num [1:28, 1:5] -0.577 0.726 0.759 0.654 0.686 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:28] "A1" "A2" "A3" "A4" ...
  .. ..$ : chr [1:5] "Agree" "Conscientious" "Extraversion" "Neuroticism" ...
 $ cor          : num [1:5, 1:5] 1 0.258 0.462 -0.185 0.147 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:5] "Agree" "Conscientious" "Extraversion" "Neuroticism" ...
  .. ..$ : chr [1:5] "Agree" "Conscientious" "Extraversion" "Neuroticism" ...
 $ corrected    : num [1:5, 1:5] 0.701 0.258 0.462 -0.185 0.147 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:5] "Agree" "Conscientious" "Extraversion" "Neuroticism" ...
  .. ..$ : chr [1:5] "Agree" "Conscientious" "Extraversion" "Neuroticism" ...
```

Statistics on the scores from score.items

```
scores <- bfi.scores$scores     #one way to address them
scores <- bfi.scores[["scores"]]      #another way to address the elements of a
describe(scores)
```

```
                 var    n  mean   sd median trimmed  mad  min max range  skew kurt
Agree              1 2800 20.65 0.89   20.8   20.73 0.89 17.0  22   5.0 -0.77
Conscientious      2 2800 36.27 0.95   36.4   36.31 0.89 33.0  38   5.0 -0.41    -
Extraversion       3 2800 36.15 1.05   36.2   36.20 1.19 33.0  38   5.0 -0.48    -
Neuroticism        4 2800  3.16 1.19    3.0    3.13 1.19  1.0   6   5.0  0.22    -
Openness           5 2800 36.59 0.80   36.6   36.62 0.89 33.2  38   4.8 -0.34    -
```

Something is wrong with the scores!

1. `score.items` reverses items
   - to reverse, it subtracts item from (max - min) + 1
   - but for the bfi, the data include age and thus the max and min are incorrect.
2. Can specify the maximum and minimum for the items to be used when reversing
   - (This is a reason to read the help file for each function!)
3. Reversing with the wrong minimum and maximum just affects the mean scores, not the scale reliabilities or intercorrelations

Score the items again, setting the min to 1, max to 6

```
bfi.scores <- score.items(keys,bfi,min=1,max=6)
describe(bfi.scores$scores)
```

```
                 var    n mean   sd median trimmed  mad min max range  skew kurtos
Agree              1 2800 4.65 0.89    4.8    4.73 0.89 1.0   6   5.0 -0.77      0.
Conscientious      2 2800 4.27 0.95    4.4    4.31 0.89 1.0   6   5.0 -0.41     -0.
Extraversion       3 2800 4.15 1.05    4.2    4.20 1.19 1.0   6   5.0 -0.48     -0.
Neuroticism        4 2800 3.16 1.19    3.0    3.13 1.19 1.0   6   5.0  0.22     -0.
Openness           5 2800 4.59 0.80    4.6    4.62 0.89 1.2   6   4.8 -0.34     -0.
```

Unts of the scale

1. Some people like to report scores as sum scores, others as mean scores
    - Sum scores are simple to find, but reflect the number of items on the scale. This can be confusing when comparing scores from alternative versions of a scale.
    - Mean scores are in the metric of the items.
2. Different subfields of psychology seem to prefer one or the other
    - Beck Depression scores range from 0 - 60+
    - STAI Anxiety scores from 20-80
    - EPI extraversion from 0-24
3. But mean scores are more informative
4. score.items defaults to means, but will report totals if desired.
    - This is just one more example of the flexibility of functions.
    - As well as the need to read the help files!

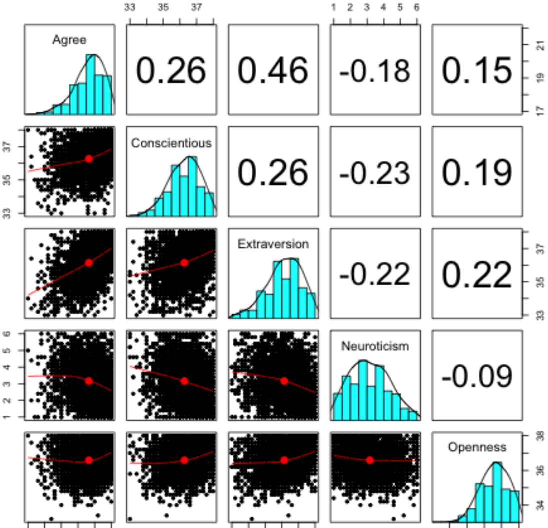
Score for total scores

```
> bfi.totals <- score.items(keys,bfi,min=1,max=6,totals=TRUE)
> describe(bfi.totals[["scores"]])
```

	var	n	mean	sd	median	trimmed	mad	min	max	range	skew	kurto
Agree	1	2800	23.27	4.47	24	23.63	4.45	5	30	25	-0.77	0
Conscientious	2	2800	21.35	4.74	22	21.55	4.45	5	30	25	-0.41	-0
Extraversion	3	2800	20.73	5.27	21	21.02	5.93	5	30	25	-0.48	-0
Neuroticism	4	2800	15.81	5.93	15	15.64	5.93	5	30	25	0.22	-0
Openness	5	2800	22.95	4.02	23	23.10	4.45	6	30	24	-0.34	-0

Show the pairs.panels result of the big 5 scores

```
pairs.panels(scores)
```



Score a multiple score test

Using `score.multiple.choice` we can either just find item and scale statistics, or convert the items to correct/incorrect and then use other functions for further analysis.

```
data(iqitems)
 iq.keys <- c(4,4,3,1,4,3,2,3,1,4,1,3,4,3)  #what are the right answers
 score.multiple.choice(iq.keys,iqitems) #get the item responses and alpha reliab
```

```
(Unstandardized) Alpha:
[1] 0.63
Average item correlation:
[1] 0.11
item statistics
     key    0    1    2    3    4    5    6 miss    r    n mean   sd  skew kurt
iq1    4 0.04 0.01 0.03 0.09 0.80 0.02 0.01    0 0.59 1000 0.80 0.40 -1.51
iq8    4 0.03 0.10 0.01 0.02 0.80 0.01 0.04    0 0.39 1000 0.80 0.40 -1.49
iq10   3 0.10 0.22 0.09 0.37 0.04 0.13 0.04    0 0.35 1000 0.37 0.48  0.53    -
iq15   1 0.03 0.65 0.16 0.15 0.00 0.00 0.00    0 0.35 1000 0.65 0.48 -0.63    -
iq20   4 0.03 0.02 0.03 0.03 0.85 0.02 0.01    0 0.42 1000 0.85 0.35 -2.00
iq44   3 0.03 0.10 0.06 0.64 0.02 0.14 0.01    0 0.42 1000 0.64 0.48 -0.61    -
iq47   2 0.04 0.08 0.59 0.06 0.11 0.07 0.05    0 0.51 1000 0.59 0.49 -0.35    -
iq2    3 0.07 0.08 0.31 0.32 0.15 0.05 0.02    0 0.26 1000 0.32 0.46  0.80    -
iq11   1 0.04 0.87 0.03 0.01 0.01 0.01 0.04    0 0.54 1000 0.87 0.34 -2.15
iq16   4 0.05 0.05 0.08 0.07 0.74 0.01 0.00    0 0.56 1000 0.74 0.44 -1.11    -
iq32   1 0.04 0.54 0.02 0.14 0.10 0.04 0.12    0 0.50 1000 0.54 0.50 -0.17    -
```

Convert the items to correct and incorrect

```
iq.tf <- score.multiple.choice(iq.keys,iqitems,score=FALSE)
 describe(iq.tf)  #compare to previous results
```

	var	n	mean	sd	median	trimmed	mad	min	max	range	skew	kurtosis	se
iq1	1	1000	0.80	0.40	1	0.88	0	0	1	1	-1.51	0.28	0.01
iq8	2	1000	0.80	0.40	1	0.87	0	0	1	1	-1.49	0.23	0.01
iq10	3	1000	0.37	0.48	0	0.34	0	0	1	1	0.53	-1.72	0.02
iq15	4	1000	0.65	0.48	1	0.69	0	0	1	1	-0.63	-1.60	0.02
iq20	5	1000	0.85	0.35	1	0.94	0	0	1	1	-2.00	2.04	0.01
iq44	6	1000	0.64	0.48	1	0.68	0	0	1	1	-0.61	-1.63	0.02
iq47	7	1000	0.59	0.49	1	0.61	0	0	1	1	-0.35	-1.88	0.02
iq2	8	1000	0.32	0.46	0	0.27	0	0	1	1	0.80	-1.37	0.01
iq11	9	1000	0.87	0.34	1	0.96	0	0	1	1	-2.15	2.64	0.01
iq16	10	1000	0.74	0.44	1	0.80	0	0	1	1	-1.11	-0.76	0.01
iq32	11	1000	0.54	0.50	1	0.55	0	0	1	1	-0.17	-1.97	0.02
iq37	12	1000	0.26	0.44	0	0.19	0	0	1	1	1.12	-0.73	0.01
iq43	13	1000	0.78	0.41	1	0.85	0	0	1	1	-1.35	-0.17	0.01
iq49	14	1000	0.32	0.47	0	0.27	0	0	1	1	0.79	-1.38	0.01

Just give me alpha, damn it!

For the user who wants to know just the alpha of a set of items and is used to SPSS output, the `alpha` function is provided. Better alternatives include the `guttman` function which provides more information.

```
alpha(iq.tf)

Reliability analysis
Call: alpha(x = iq.tf)

  raw_alpha std.alpha G6(smc) average_r mean   sd
      0.63      0.65    0.65      0.12 0.61 0.18

 Reliability if an item is dropped:
     raw_alpha std.alpha G6(smc) average_r
iq1       0.58      0.60    0.60      0.10
iq8       0.61      0.63    0.63      0.12
iq10      0.63      0.64    0.65      0.12
iq15      0.63      0.64    0.64      0.12
iq20      0.61      0.62    0.63      0.11
iq44      0.61      0.63    0.63      0.12
iq47      0.60      0.62    0.62      0.11
iq2       0.64      0.66    0.66      0.13
iq11      0.59      0.60    0.60      0.10
iq16      0.59      0.60    0.60      0.10
iq32      0.60      0.62    0.62      0.11
iq37      0.64      0.66    0.66      0.13
iq43      0.60      0.61    0.62      0.11
iq49      0.64      0.65    0.66      0.13
```

```
alpha(iq.tf)

Item statistics
        n    r r.cor r.drop mean   sd
iq1  1000 0.61 0.594  0.475 0.80 0.40
iq8  1000 0.41 0.318  0.251 0.80 0.40
iq10 1000 0.33 0.211  0.166 0.37 0.48
iq15 1000 0.34 0.227  0.173 0.65 0.48
iq20 1000 0.45 0.379  0.295 0.85 0.35
iq44 1000 0.41 0.318  0.254 0.65 0.48
iq47 1000 0.49 0.434  0.345 0.59 0.49
iq2  1000 0.25 0.111  0.085 0.32 0.46
iq11 1000 0.58 0.555  0.440 0.87 0.34
iq16 1000 0.56 0.541  0.426 0.74 0.44
iq32 1000 0.48 0.418  0.330 0.54 0.50
iq37 1000 0.23 0.081  0.066 0.26 0.44
iq43 1000 0.50 0.454  0.359 0.78 0.41
iq49 1000 0.26 0.124  0.098 0.32 0.47
```

Multivariate data reduction and description

A recurring theme in personality research is the description of personality items (be they adjectives or short questions), in terms of a limited number of higher order dimensions. These are typically identified through factor analysis, principal components analysis, or cluster analysis. All of these procedures are straightforward in R.

1. Exploratory factor analysis: a latent trait model
    - Items are assumed to represent the influence of unobserved (latent) variables.
    - Issues are the means of extraction, the number of factors to extract, the rotations to use, the estimation of factor scores.
    - Factor scores are *estimated*
2. Confirmatory factor analysis: a latent trait model
    - (discussed under structural equation modeling) the typical model is one of a cluster structure with items loading on one and only one factor.
    - This assumption is probably not appropriate, and rotational techniques for complexity > 1 are available.

Multivariate data reduction and description: 2

1. Principal Components analysis: an observed variable model
   - Components are defined as sums of observed variables.
   - Component scores may be calculated as weighted sums, not *estimated* as is necessary for factor scores.
   - Components include measurement error as part of the score.
2. Cluster analysis, although usually applied to clustering of objects (people), may be applied to clustering of items.
   - Some algorithms take reliability into account (correct for attenuation), and thus implicitly become latent variable models.

There are several ways to do factor analysis in R

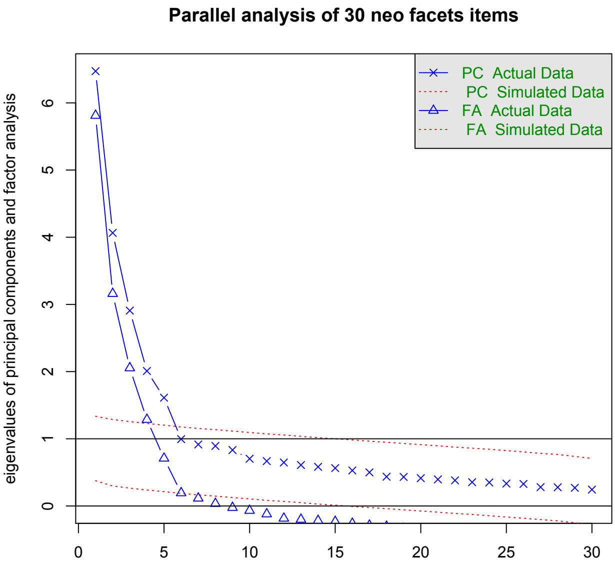
1. `factanal` from core R
   - Maximum likelihood factor analysis
2. `fa` and `fa.poly` from *psych* (replacing `factor.pa`, `fa.wls`)
   - data input = A correlation matrix or a raw data matrix. If raw data, the correlation matrix will be found using pairwise deletion.
   - factor method = factoring method fm="minres" will do a minimum residual (OLS), fm="wls" will do a weighted least squares (WLS) solution, fm="gls" does a generalized weighted least squares (GLS), fm="pa" will do the principal factor solution, fm="ml" will do a maximum likelihood factor analysis
   - rotation method = "none", "varimax", "quartimax", "bentlerT", and "geominT" are orthogonal rotations. "promax", "oblimin", "simplimax", "bentlerQ", and "geominQ" or "cluster" are possible rotations or transformations of the solution. The default is to do a oblimin transformation.
   - Confidence intervals may be found by bootstrapping multiple solutions.

The number of factors problem

"It is easy to solve the number of factors problem, I do it everyday before breakfast. The problem is what is the right answer " (attributed to Henry Kaiser)

1. χ^2 tests (either of n factor solution or of change from n-1 to n factors)
    - Sensitive to sample size.
    - Larger samples have more significant factors
2. Scree test
    - Generally good, sometimes hard to identify break in scree
3. Parallel analysis (compare to random data)
    - Factors and components give different solutions
4. Very Simple Structure
    - Works well with items of complexity 1 or 2
5. Minimum Average Partial
6. Eigen values > 1
    - Perhaps the uniformly agreed worst test

Parallel analysis of 30 NEO facets

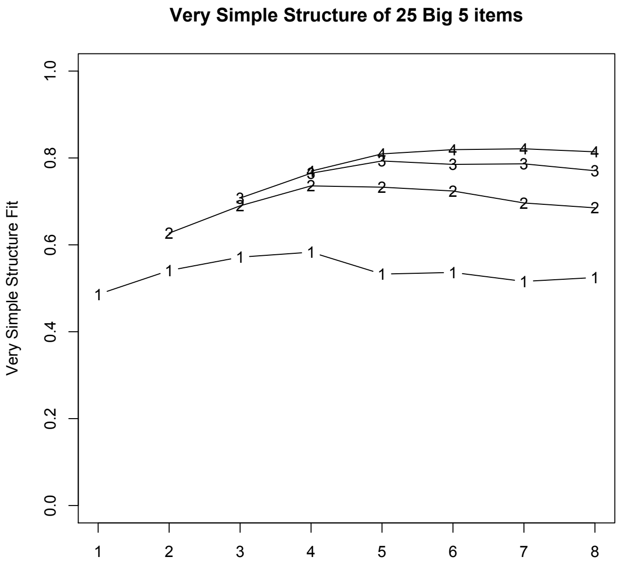


Very Simple Structure and Velicer's Map criterion

```
> VSS(bfi[1:25],title="Very Simple Structure of 25 Big 5 items")
```

```
Very Simple Structure of  Very Simple Structure of 25 Big 5 items
Call: VSS(x = bfi[1:25], title = "Very Simple Structure of 25 Big 5 items")
VSS complexity 1 achieves a maximimum of 0.58  with  4  factors
VSS complexity 2 achieves a maximimum of 0.74  with  4  factors

The Velicer MAP criterion achieves a minimum of 0.01  with  5  factors

Velicer MAP
[1] 0.02 0.02 0.02 0.02 0.01 0.02 0.02 0.02

Very Simple Structure Complexity 1
[1] 0.49 0.54 0.57 0.58 0.53 0.54 0.52 0.52

Very Simple Structure Complexity 2
[1] 0.00 0.63 0.69 0.74 0.73 0.72 0.70 0.69
```

Very Simple Structure and Velicer's Map



Factor analysis of Thurstone 9 variable problem

```
> f3 <- fa(Thurstone,3) #we want a 3 factor solution, otherwise, use the defaults
> f3

Factor Analysis using method =  minres
Call: fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate,
    scores = scores, residuals = residuals, SMC = SMC, missing = FALSE,
    impute = impute, min.err = min.err, max.iter = max.iter,
    symmetric = symmetric, warnings = warnings, fm = fm, alpha = alpha)
Standardized loadings based upon correlation matrix
                  MR1   MR2   MR3   h2   u2
Sentences        0.91 -0.04  0.04 0.82 0.18
Vocabulary       0.89  0.06 -0.03 0.84 0.16
Sent.Completion  0.83  0.04  0.00 0.73 0.27
First.Letters    0.00  0.86  0.00 0.73 0.27
4.Letter.Words  -0.01  0.74  0.10 0.63 0.37
Suffixes         0.18  0.63 -0.08 0.50 0.50
Letter.Series    0.03 -0.01  0.84 0.72 0.28
Pedigrees        0.37 -0.05  0.47 0.50 0.50
Letter.Group    -0.06  0.21  0.64 0.53 0.47

                 MR1  MR2  MR3
SS loadings     2.64 1.86 1.50
Proportion Var 0.29 0.21 0.17
Cumulative Var 0.29 0.50 0.67

 With factor correlations of
     MR1  MR2  MR3
MR1 1.00 0.59 0.54
MR2 0.59 1.00 0.52
MR3 0.54 0.52 1.00
...
```

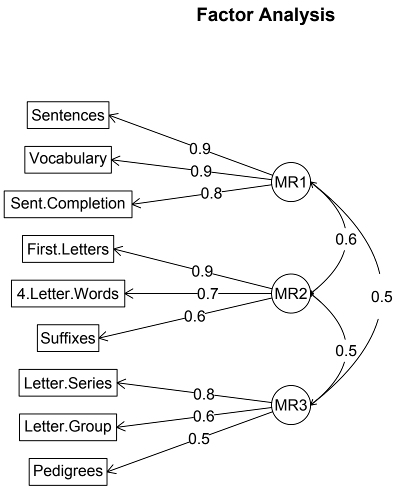
Factor analysis output, continued

```
Test of the hypothesis that 3 factors are sufficient.

The degrees of freedom for the null model are  36  and the objective function
             was  5.2  with Chi Square of  1081.97
The degrees of freedom for the model are 12  and
           the objective function was  0.01

The root mean square of the residuals is  0
The df corrected root mean square of the residuals is  0.01
The number of observations was  213  with Chi Square =  2.82  with prob <  1

Tucker Lewis Index of factoring reliability =  1.027
RMSEA index =  0  and the 90 % confidence intervals are  0 0.023
BIC =  -61.51
Fit based upon off diagonal values = 1
Measures of factor score adequacy
                                                MR1  MR2  MR3
Correlation of scores with factors             0.96 0.92 0.90
Multiple R square of scores with factors       0.93 0.85 0.81
Minimum correlation of possible factor scores  0.86 0.71 0.63
```

Bootstrapped confidence intervals

```
> f3 <- fa(Thurstone,3,n.obs=213,n.iter=20)   #to do bootstrapping

Coefficients and bootstrapped confidence intervals
                   low   MR1 upper   low   MR2 upper   low   MR3 upper
Sentences         0.80  0.91  0.96 -0.10 -0.04  0.04 -0.02  0.04  0.13
Vocabulary        0.77  0.89  0.94  0.01  0.06  0.16 -0.10 -0.03  0.07
Sent.Completion   0.73  0.83  0.92 -0.06  0.04  0.11 -0.09  0.00  0.09
First.Letters    -0.06  0.00  0.10  0.68  0.86  0.93 -0.08  0.00  0.10
4.Letter.Words   -0.13 -0.01  0.10  0.58  0.74  0.84  0.03  0.10  0.21
Suffixes          0.00  0.18  0.34  0.49  0.63  0.76 -0.19 -0.08  0.03
Letter.Series    -0.04  0.03  0.12 -0.12 -0.01  0.11  0.53  0.84  0.96
Pedigrees         0.26  0.37  0.52 -0.17 -0.05  0.07  0.26  0.47  0.61
Letter.Group     -0.19 -0.06  0.05  0.07  0.21  0.35  0.43  0.64  0.79

 Interfactor correlations and bootstrapped confidence intervals
  lower estimate upper
1  0.39     0.59  0.63
2  0.34     0.54  0.59
3  0.32     0.52  0.56
```

The simple factor structure (pattern) may be shown graphically



Analyzing the higher order structure: the ω coefficients

1. If items or scales intercorrelate, they may be in turn factored.
   - The effect of these higher order factors may be found on the lowest level variables and then removed from the first level factors.
   - The debate about the "general factor of personality" hinges on this method.
   - Higher order factors may be found using exploratory or confirmatory procedures.
2. `omega` is an exploratory hierarchical factoring function to find
   - ω_h (hierarchical), an estimate of the general factor of a test
   - ω_t, an estimate of the reliable variance in a test
3. `omega.sem` will do a confirmatory analysis based upon the simple cluster structure found by `omega`
   - CFA solutions based upon a simple cluster structure will overestimate the general factor by not identifying all the cross loadings.

omega analysis of the Thurstone problem.

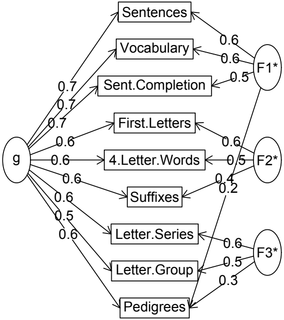
```
> omega(Thurstone,n.obs=213) #defaults to 3 factors
Omega
Call: omegah(m = m, nfactors = nfactors, fm = fm, key = key, flip = flip,
    digits = digits, title = title, sl = sl, labels = labels,
    plot = plot, n.obs = n.obs, rotate = rotate, Phi = Phi, option = option)
Alpha:                 0.89
G.6:                   0.91
Omega Hierarchical:    0.74
Omega H asymptotic:    0.79
Omega Total            0.93

Schmid Leiman Factor loadings greater than  0.2
                   g   F1*   F2*   F3*   h2   u2   p2
Sentences       0.71  0.57              0.82 0.18 0.61
Vocabulary      0.73  0.55              0.84 0.16 0.63
Sent.Completion 0.68  0.52              0.73 0.27 0.63
First.Letters   0.65        0.56        0.73 0.27 0.57
4.Letter.Words  0.62        0.49        0.63 0.37 0.61
Suffixes        0.56        0.41        0.50 0.50 0.63
Letter.Series   0.59              0.61 0.72 0.28 0.48
Pedigrees       0.58  0.23        0.34 0.50 0.50 0.66
Letter.Group    0.54              0.46 0.53 0.47 0.56
```

omega output continued

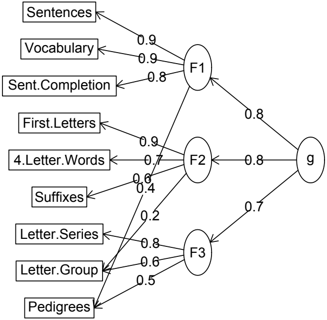
```
With eigenvalues of:
   g  F1*  F2*  F3*
3.58 0.96 0.74 0.71

general/max  3.71   max/min =   1.35
mean percent general =  0.6    with sd =  0.05 and cv of  0.09

The degrees of freedom are 12  and the fit is  0.01
The number of observations was  213  with Chi Square =  2.82  with prob <  1
The root mean square of the residuals is  0
The df corrected root mean square of the residuals is  0.01
RMSEA index =  0  and the 90 % confidence intervals are  0 0.023
BIC =  -61.51

Compare this with the adequacy of just a general factor and no group factors
The degrees of freedom for just the general factor are 27  and the fit is  1.48
The number of observations was  213  with Chi Square =  307.1  with prob <  2.8e-49
The root mean square of the residuals is  0.1
The df corrected root mean square of the residuals is  0.16

RMSEA index =  0.224  and the 90 % confidence intervals are  0.223 0.226
BIC =  162.35

Measures of factor score adequacy
                                                g  F1*  F2*  F3*
Correlation of scores with factors           0.86 0.73 0.72 0.75
Multiple R square of scores with factors     0.74 0.54 0.52 0.56
Minimum correlation of factor score estimates 0.49 0.08 0.03 0.11
```

Two ways of viewing the higher order structure

Omega



Hierarchical (multilevel) Structure



Principal Components Analysis is an observed data model

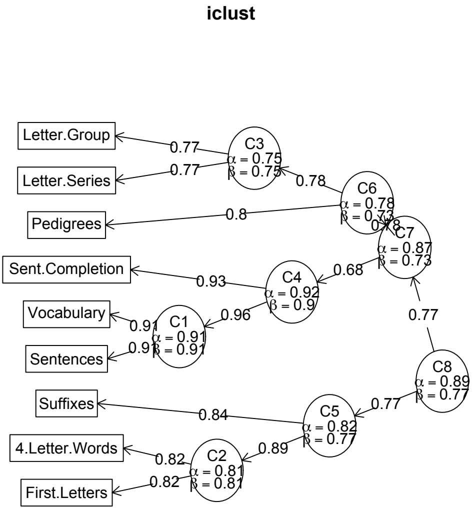
```
> principal(Thurstone,3,n.obs=213) #ask for 3 components
Principal Components Analysis
Call: principal(r = Thurstone, nfactors = 3, n.obs = 213)
Standardized loadings based upon correlation matrix
                 RC1  RC2  RC3   h2   u2
Sentences       0.86 0.24 0.23 0.86 0.14
Vocabulary      0.85 0.31 0.19 0.86 0.14
Sent.Completion 0.85 0.26 0.19 0.83 0.17
First.Letters   0.23 0.82 0.23 0.78 0.22
4.Letter.Words  0.18 0.79 0.30 0.75 0.25
Suffixes        0.31 0.77 0.06 0.70 0.30
Letter.Series   0.25 0.16 0.83 0.78 0.22
Pedigrees       0.53 0.08 0.61 0.67 0.33
Letter.Group    0.10 0.31 0.80 0.75 0.25
                RC1  RC2  RC3
SS loadings    2.73 2.25 1.99
Proportion Var 0.30 0.25 0.22
Cumulative Var 0.30 0.55 0.78
Test of the hypothesis that 3 factors are sufficient.
The degrees of freedom for the null model are  36  and the objective function w
The degrees of freedom for the model are 12  and the objective function was 0.
The number of observations was  213  with Chi Square =  127.9  with prob <  1.6
Fit based upon off diagonal values = 0.98
```

Cluster analysis as an alternative to factor analysis and principal components analysis

1. An alternative to factor analysis for dimensional reduction is cluster analysis
   - The `iclust` algorithm was developed for clustering items based upon basic psychometric principals
2. Procedure
   1. Find the correlation matrix
   2. Identify the most similar pair of items (correcting for attenuation)
   3. Combine them.
   4. Repeat steps 1-3 until β (the worst split half reliability) fails to increase.
   5. As an alternative, a specified number of clusters may be extracted.

A hierarchical cluster structure found by `iclust`

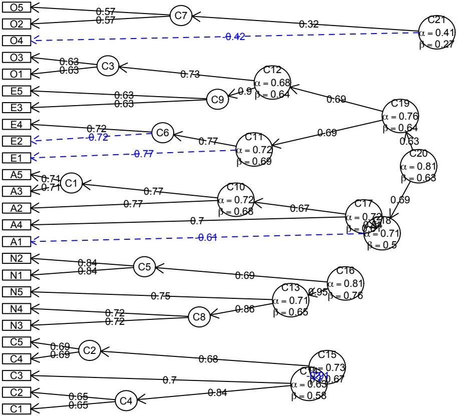
iclust(Thurstone)



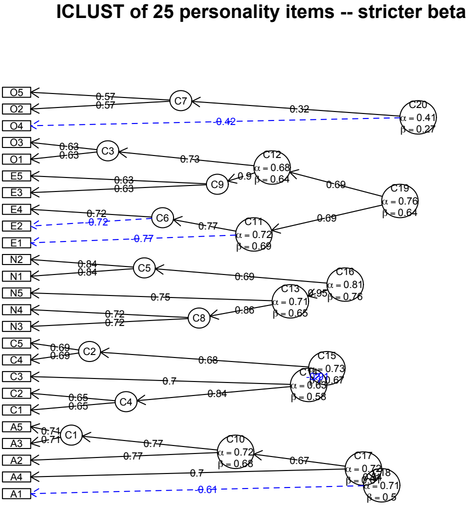
A hierarchical cluster structure of 25 Big 5 items found by `iclust`

iclust(bfi[1:25])

ICLUST of 25 personality items



A hierarchical cluster structure of 25 Big 5 items found by `iclust` with a more strict criterion



ICLUST produces basic scale reliability information

```
> iclust(bfi[1:25],beta=2,title="ICLUST of 25 personality items -- stricter beta

ICLUST (Item Cluster Analysis)
Call: ICLUST(r.mat = r.mat, nclusters = nclusters, alpha = alpha, beta = beta,
    beta.size = beta.size, alpha.size = alpha.size, correct = correct,
    correct.cluster = correct.cluster, reverse = reverse, beta.min = beta.min,
    output = output, digits = digits, labels = labels, cut = cut,
    n.iterations = n.iterations, title = title, plot = plot,
    weighted = weighted, cor.gen = cor.gen, SMC = SMC)
Purified Alpha:
 C19  C18  C16  C15  C20
0.76 0.71 0.81 0.73 0.61

G6* reliability:
 C19  C18  C16  C15  C20
0.77 0.71 0.81 0.72 0.61

Original Beta:
 C19  C18  C16  C15  C20
0.64 0.50 0.76 0.67 0.27
Cluster size:
C19 C18 C16 C15 C20
  5   5   5   5   5
```

ICLUST output (continued) shows item by cluster loadings and cluster intercorrelations

```
Item by Cluster Structure matrix:
   C19   C18   C16   C15   C20
A1 -0.10 -0.39  0.14  0.05  0.13
A2  0.40  0.67 -0.07 -0.23 -0.19
....

O4 -0.10  0.06  0.21  0.00 -0.33
O5 -0.11 -0.10  0.11  0.15  0.53

With eigenvalues of:
C19 C18 C16 C15 C20
3.6 3.1 3.0 2.6 1.9

Purified scale intercorrelations
 reliabilities on diagonal
 correlations corrected for attenuation above diagonal:
      C19   C18   C16   C15   C20
C19  0.76  0.64 -0.28 -0.36 -0.35
C18  0.47  0.71 -0.24 -0.35 -0.25
C16 -0.22 -0.18  0.81  0.29  0.11
C15 -0.27 -0.25  0.22  0.73  0.30
C20 -0.24 -0.16  0.07  0.20  0.61
```

Factor Extension and Set Correlation

1. Originally developed by Dwyer for the case of having completed a factor analysis and then a new variable is introduced.
   - At the time, factoring was hard and time consuming
2. May now be used to extend the factors from one domain into another domain.
   - Differs from SEM in that the factors are estimated in the first domain and are not changed with the addition of the second domain
3. Another technique for relating two domains is "Set Correlation" as discussed by Cohen, Cohen, Aiken and West.

Consider the case of the NEO

Split the NEO facets into odds and evens. Factor the odds, extend to the evens.

```
> neo <- as.matrix(neo)
> odd <- seq(1,29,2)
> f5 <- fa(neo[odd,odd],5)
> fe <- fa.extension(neo[odd,-odd],f5)
> fe <- fa.extension(neo[odd,-odd],f5)
```

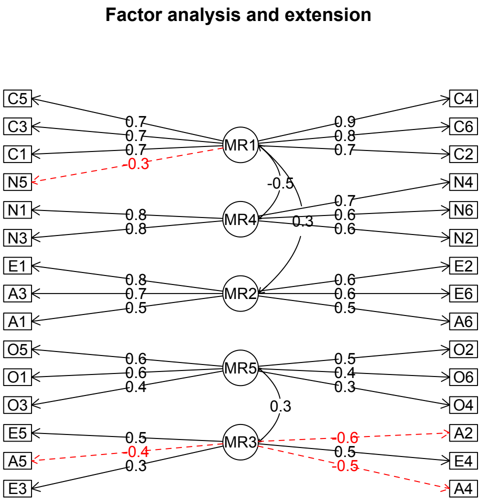
```
Call: fa.extension(Roe = neo[ss, -ss], fo = f5)
Standardized loadings based upon correlation matrix
     MR1   MR4   MR3   MR2   MR5   h2   u2
N5  0.44 -0.18 -0.28  0.15  0.09 0.37 0.63
N6  0.75 -0.33  0.09 -0.06  0.01 0.86 0.14
E5 -0.01 -0.02 -0.49  0.25  0.12 0.33 0.67
E6 -0.02  0.09 -0.14  0.61  0.22 0.57 0.43
O5 -0.26  0.16 -0.08 -0.08  0.65 0.52 0.48
O6 -0.10 -0.11 -0.11  0.07  0.26 0.11 0.89
A5  0.23 -0.10  0.56  0.07 -0.06 0.37 0.63
A6  0.08 -0.05  0.39  0.44  0.11 0.38 0.62
C5 -0.31  0.75  0.07  0.06 -0.09 0.85 0.15
C6 -0.25  0.52  0.34 -0.10 -0.15 0.55 0.45

                 MR1  MR4  MR3  MR2  MR5
SS loadings     1.26 1.26 0.99 0.71 0.71
Proportion Var 0.13 0.13 0.10 0.07 0.07
Cumulative Var 0.13 0.25 0.35 0.42 0.49

 With factor correlations of
      MR1   MR4   MR3   MR2   MR5
MR1  1.00 -0.32 -0.13 -0.26  0.05
MR4 -0.32  1.00  0.00  0.32  0.08
```

Set correlation is a generalized R^2 between two sets of variables

$R^2 = 1 - \prod(1 - \lambda_i^2)$ where λ_i^2 is the is ith squared canonical correlation. Unfortunately, the R^2 is sensitive to one of the canonical correlations being very high. An alternative, T^2, is the proportion of additive variance and is the average of the squared canonicals.

```
> set.cor(even,odd,data=neo)

Multiple Regression from matrix input
Beta weights
      N2    N4    N6    E2    E4    E6    O2    O4    O6    A2    A4    A6    C2    C4    C6
N1  0.19  0.23  0.30  0.07  0.06 -0.03 -0.05 -0.04  0.02  0.02 -0.01  0.00  0.06  0.07  0.10
N3  0.26  0.30  0.20 -0.11  0.06 -0.11  0.05 -0.10 -0.02  0.02 -0.04  0.09  0.01  0.03 -0.08
...
C3 -0.02  0.03  0.00 -0.01  0.00 -0.12  0.04 -0.09 -0.12  0.15  0.03 -0.04  0.10  0.16  0.19
C5 -0.01 -0.04 -0.13 -0.02  0.28  0.08  0.04  0.10 -0.05  0.04  0.00  0.00  0.47  0.42  0.03
Multiple R
  N2   N4   N6   E2   E4   E6   O2   O4   O6   A2   A4   A6   C2   C4   C6
0.69 0.69 0.77 0.61 0.61 0.68 0.58 0.45 0.38 0.63 0.65 0.54 0.60 0.69 0.63
Multiple R2
  N2   N4   N6   E2   E4   E6   O2   O4   O6   A2   A4   A6   C2   C4   C6
0.48 0.47 0.60 0.37 0.37 0.46 0.33 0.20 0.14 0.40 0.42 0.30 0.35 0.48 0.39
Various estimates of between set correlations
Squared Canonical Correlations
 [1] 8.0e-01 6.5e-01 5.2e-01 4.3e-01 3.5e-01 1.5e-01 1.1e-01 5.9e-02 4.7e-02
                     3.8e-02 1.7e-02 1.2e-02 8.6e-03 4.6e-03 2.4e-05
  Average squared canonical correlation =  0.21
 Cohen's Set Correlation R2 =  0.99
```

Factor extension of the odd NEO facets to the even



Structural Equation modeling packages

SEM packages allow for Confirmatory Factor Analysis as well as Structural modeling.

1. sem (by John Fox and others)
   - uses RAM notation
   - does not handle multiple groups
   - does not seem to be actively developed
2. lavaan (by Yves Rosseel and others)
   - Mimics as much as possible MPLUS output
   - Allows for multiple groups
   - Easy syntax
3. OpenMx
   - Open source and R version of Mx
   - Allows for multiple groups (and almost anything else)
   - Complicated syntax

lavaan analysis – from the example – output mimics MPlus

```
#The Holzinger and Swineford (1939) example
HS.model <- ' visual  =~ x1 + x2 + x3
              textual =~ x4 + x5 + x6
              speed   =~ x7 + x8 + x9 '

fit <- lavaan(HS.model, data=HolzingerSwineford1939,
              auto.var=TRUE, auto.fix.first=TRUE,
              auto.cov.lv.x=TRUE)
summary(fit, fit.measures=TRUE)
```

```
lavaan (0.4-7) converged normally after 35 iterations

  Number of observations                           301

  Estimator                                         ML
  Minimum Function Chi-square                   85.306
  Degrees of freedom                                24
  P-value                                        0.000

Chi-square test baseline model:

  Minimum Function Chi-square                  918.852
  Degrees of freedom                                36
  P-value                                       0.00
```

lavaan example – continued

```
Full model versus baseline model:
  Comparative Fit Index (CFI)                    0.931
  Tucker-Lewis Index (TLI)                       0.896
Loglikelihood and Information Criteria:
  Loglikelihood user model (H0)              -3737.745
  Loglikelihood unrestricted model (H1)      -3695.092

  Number of free parameters                         21
  Akaike (AIC)                                7517.490
  Bayesian (BIC)                              7595.339
  Sample-size adjusted Bayesian (BIC)         7528.739

Root Mean Square Error of Approximation:
  RMSEA                                          0.092
  90 Percent Confidence Interval          0.071  0.114
  P-value RMSEA <= 0.05                          0.001

Standardized Root Mean Square Residual:
  SRMR                                           0.065
Parameter estimates:
  Information                                 Expected
  Standard Errors                             Standard
                 Estimate  Std.err  Z-value  P(>|z|)
Latent variables:
  visual =~
    x1              1.000
    x2              0.554    0.100    5.554    0.000
    x3              0.729    0.109    6.685    0.000
  textual =~
    x4              1.000
    x5              1.113    0.065   17.014    0.000
    x6              0.926    0.055   16.703    0.000
```

Using lavaan to examine measurement invariance – from the example

```
HW.model <- ' visual  =~ x1 + x2 + x3
              textual =~ x4 + x5 + x6
              speed   =~ x7 + x8 + x9 '
measurementInvariance(HW.model, data=HolzingerSwineford1939, group="school")
```

```
Measurement invariance tests:
Model 1: configural invariance:
   chisq       df    pvalue      cfi     rmsea      bic
 115.851   48.000    0.000    0.923    0.097 7604.094
Model 2: weak invariance (equal loadings):
   chisq       df    pvalue      cfi     rmsea      bic
 124.044   54.000    0.000    0.921    0.093 7578.043
[Model 1 versus model 2]
  delta.chisq      delta.df delta.p.value     delta.cfi
        8.192         6.000         0.224         0.002
Model 3: strong invariance (equal loadings + intercepts):
   chisq       df    pvalue      cfi     rmsea      bic
 164.103   60.000    0.000    0.882    0.107 7686.588
[Model 1 versus model 3]
  delta.chisq      delta.df delta.p.value     delta.cfi
       48.251        12.000         0.000         0.041
[Model 2 versus model 3]
  delta.chisq      delta.df delta.p.value     delta.cfi
       40.059         6.000         0.000         0.038
Model 4: equal loadings + intercepts + means:
   chisq       df    pvalue      cfi     rmsea      bic
 204.605   63.000    0.000    0.854    0.122 7709.969
[Model 1 versus model 4]
  delta.chisq      delta.df delta.p.value     delta.cfi
       88.754        15.000         0.000         0.069
[Model 3 versus model 4]
  delta.chisq      delta.df delta.p.value     delta.cfi
       40.502         3.000         0.000         0.028
```

Item Response Theory

1. Said to be the "new psychometrics", IRT combines item and person information
   - Several packages for IRT, including 1 parameter (Rasch) as well as 2 and 3 parameter models
   - These estimate the parameters using standard IRT approaches
2. An alternative is to recognize that 2 parameter IRT models are just factor models applied to the *tetrachoric* or *polychoric* correlations.
   - That is, find the factor analysis loadings (λ_i) and the item endorsement frequencies expressed as normal deviates (τ_i and then convert to IRT parameters
   - discrimination $\alpha = \frac{\lambda_i}{\sqrt{1-\lambda_i^2}}$
   - location (difficulty) $\delta = \frac{\tau_i}{\sqrt{1-\lambda_i^2}}$

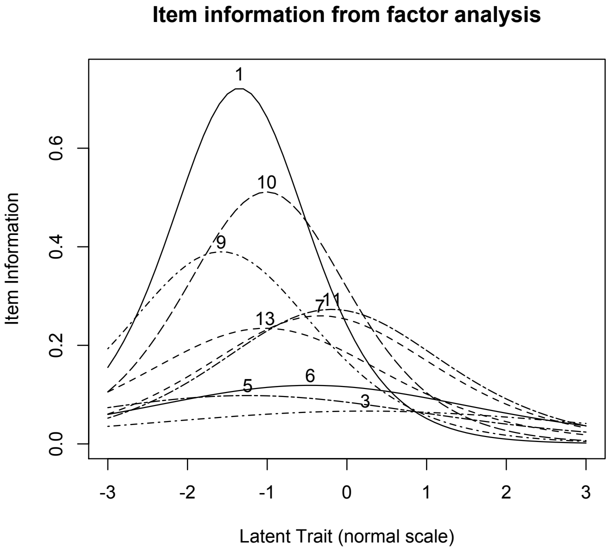
Multiple packages to do Item Response Theory analysis

1. *psych* uses a factor analytic procedure to estimate item discriminations and locations
   - look at examples for irt.fa
   - two example data sets: iqitems and bfi
2. `irt.fa` finds either tetrachoric or polychoric correlation matrices
   - Returns normal factor analysis output as well as IRT parameters
   - Converts factor loadings to disciminations
   - Saves the tetrachoric/polychoric correlation matrix for faster reanalyses
3. `plot.irt` plots item information and item characteristic functions
4. Other packages include *ltm*, *MCMCpack* (for Markov chain Monte Carlo k-dimensional IRT models), and *irtoys* for interfacing with different packages.

IRT analysis of 14 iq items – dichotomous items

```
> iq.keys <- c(4,4,3,1,4,3,2,3,1,4,1,3,4,3)
> iq.tf <- score.multiple.choice(iq.keys,iqitems,score=FALSE)  #just the respons
> iq.irt <- irt.fa(iq.tf)
> plot(iq.irt)
> iq.irt

Item Response Analysis using Factor Analysis
Call: irt.fa(x = iq.tf)
Item discrimination and location for factor  MR1
     discrimination location
iq1            1.15    -1.29
iq8            0.50    -0.94
iq10           0.34     0.35
iq15           0.30    -0.41
iq20           0.70    -1.29
iq44           0.46    -0.41
iq47           0.64    -0.26
iq2            0.19     0.49
iq11           1.23    -1.76
iq16           1.01    -0.93
iq32           0.69    -0.13
iq37           0.12     0.66
iq43           0.75    -0.97
iq49           0.18     0.48
```

Item Response Information curves for 14 iq items



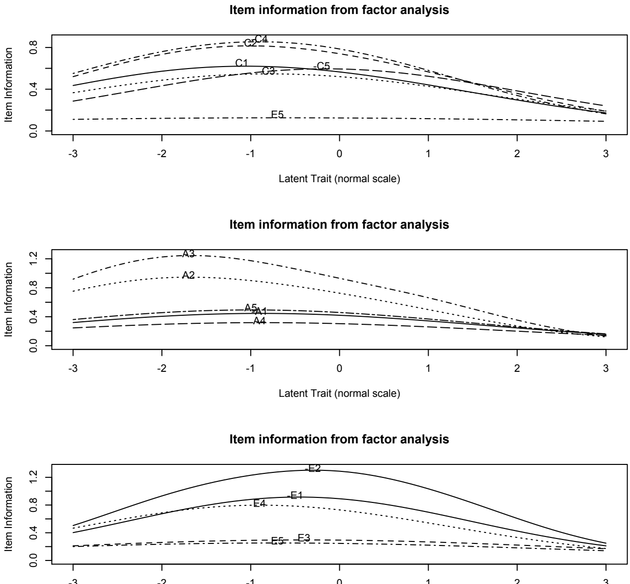
Extending IRT to the multidimensional case

1. By using a factor analytic approach, we can find IRT parameters for multiple factors
   - `irt.fa` will find multiple factors and then convert the highest loadings on each factor to IRT parameters
2. One powerful advantage of IRT is that by displaying item information statistics, we can choose items that provide maximal information.
   - Area under the curve is reported for each item information curve.
   - Can also plot item characteristic curves, or test information curves.

IRT analysis of the first 15 bfi items – Polytomous items – this is time consuming the first time

```
> irt.bfi <- irt.fa(bfi[1:15],3)  #save the results for a faster reanalysis
> irt.bfi

Item Response Analysis using Factor Analysis
Call: irt.fa(x = bfi[1:15], 3)
Item discrimination and location for factor  MR2
   discrimination location.1 location.2 location.3 location.4 location.5
A1           0.06      -0.44       0.32       0.74       1.23       1.89
...
C1           0.77      -2.45      -1.74      -1.14      -0.26       1.00
C2           0.92      -2.52      -1.62      -1.03      -0.15       1.15
C3           0.72      -2.31      -1.45      -0.93      -0.03       1.18
C4          -0.95      -0.81       0.22       0.86       1.73       2.75
C5          -0.73      -1.13      -0.36       0.03       0.76       1.57
E1           0.11      -0.71      -0.07       0.30       0.78       1.37
...
Item discrimination and location for factor  MR3
   discrimination location.1 location.2 location.3 location.4 location.5
A1          -0.62      -0.51       0.38       0.87       1.45       2.22
A2           1.02      -3.02      -2.19      -1.70      -0.68       0.69
A3           1.23      -2.93      -2.09      -1.52      -0.52       0.96
A4           0.51      -1.89      -1.30      -0.99      -0.43       0.25
A5           0.67      -2.44      -1.63      -1.11      -0.30       0.81
...
E5           0.05      -1.82      -1.21      -0.78      -0.15       0.77
Item discrimination and location for factor  MR1
   discrimination location.1 location.2 location.3 location.4 location.5
...
C5          -0.14      -0.92      -0.30       0.02       0.62       1.28
E1          -0.94      -0.97      -0.09       0.41       1.06       1.86
E2          -1.25      -1.40      -0.27       0.22       1.18       2.13
```

Plot the item information functions for the three factors



Part I: an introduction to R
Part II: Using R for psychometrics
Part III: Structures, Objects, Functions

Outline of Part III: Basic R Commands

8 Data Structures

9 Objects and functions

10 Getting help

11 Frequently used functions

12 More on Functions
- Writing your own function

A brief technical interlude

1. Data structures
   - The basic: scalers, vectors, matrices
   - More advanced data frames and lists
   - Showing the data
2. Getting the length, dimensions and structure of a data structure
   - length(x), dim(x), str(x)
3. Objects and Functions
   - Functions act upon objects
   - Functions actually are objects themselves
   - Getting help for a function or a package

The basic types of data structures

1. Scalers (characters, integers, reals, complex)

```
> A <- 1
> B <- 2
```

2. Vectors (of scalers, all of one type) have `length`

```
> C <- month.name[1:5]
> D <- 12:24
> length(D)

[1] 13
```

3. Matrices (all of one type) have `dimensions`

```
> E <- matrix(1:20, ncol = 4)
> dim(E)

[1] 5 4
```

Show values by entering the variable name

```
> A
[1] 1
> B
[1] 2
> C
[1] "January"  "February" "March"    "April"    "May"
> D
 [1] 12 13 14 15 16 17 18 19 20 21 22 23 24
> E
     [,1] [,2] [,3] [,4]
[1,]    1    6   11   16
[2,]    2    7   12   17
[3,]    3    8   13   18
[4,]    4    9   14   19
[5,]    5   10   15   20
```

More complicated (and useful) types: Data frames and Lists

1. Data frames are collections of vectors and may be of different type. They have two dimensions.

```
> E.df <- data.frame(names = C, values = c(31, 28, 31, 30, 31))
> dim(E.df)

[1] 5 2
```

2. Lists are collections of what ever you want. They have length, but do not have dimensions.

```
> F <- list(first = A, a.vector = C, a.matrix = E)
> length(F)

[1] 3
```

Show values by entering the variable name

```
> E.df
      names values
1   January     31
2  February     28
3     March     31
4     April     30
5       May     31
> F
$first
[1] 1

$a.vector
[1] "January"  "February" "March"    "April"    "May"

$a.matrix
     [,1] [,2] [,3] [,4]
[1,]    1    6   11   16
[2,]    2    7   12   17
[3,]    3    8   13   18
[4,]    4    9   14   19
[5,]    5   10   15   20
```

1. To show the structure of a list, use `str`

```
> str(F)

List of 3
 $ first   : num 1
 $ a.vector: chr [1:5] "January" "February" "March" "April" ...
 $ a.matrix: int [1:5, 1:4] 1 2 3 4 5 6 7 8 9 10 ...
```

2. to address an element of a list, call it by name or number, to get a row or column of a matrix specify the row, column or both.

```
> F[[2]]

[1] "January"  "February" "March"    "April"    "May"

> F[["a.matrix"]][, 2]

[1]  6  7  8  9 10

> F[["a.matrix"]][2, ]

[1]  2  7 12 17
```

Addressing the elements of a data.frame or matrix

Setting row and column names using paste

```
> E <- matrix(1:20, ncol = 4)
> colnames(E) <- paste("C", 1:ncol(E), sep = "")
> rownames(E) <- paste("R", 1:nrow(E), sep = "")
> E
   C1 C2 C3 C4
R1  1  6 11 16
R2  2  7 12 17
R3  3  8 13 18
R4  4  9 14 19
R5  5 10 15 20
> E["R2", ]
C1 C2 C3 C4
 2  7 12 17
> E[, 3:4]
   C3 C4
R1 11 16
R2 12 17
R3 13 18
R4 14 19
R5 15 20
```

Objects and Functions

1. R is a collection of Functions that act upon and return Objects
2. Although most functions can act on an object and return an object (a =f(b)), some are binary operators
   - primitive arithmetic functions +, -, * , /, %*%,
   - logical functions <, > ,==, !=
3. Some functions do not return values
   - `print(x,digits=3)`
   - `summary(some object)`
4. But most useful functions act on an object and return a resulting object
   - this allows for extraordinary power because you can combine functions by making the output of one the input of the next.
   - The number of R functions is very large, for each package has introduced more functions, but for any one task, not many functions need to be learned.

Getting help

1. All functions have a help menu
   - `help(the function)`
   - ? the function
   - most function help pages have examples to show how to use the function
2. Most packages have "vignettes" that give overviews of all the functions in the package and are somewhat more readable than the help for a specific function.
   - The examples are longer, somewhat more readable. (e.g., the vignette for *psych* is available either from the menu (Mac) or from http://cran.r-project.org/web/packages/psych/vignettes/overview.pdf
3. To find a function in the entire R space, use `findFn` in the *sos* package.
4. Online tutorials (e.g.,http://Rpad.org for a list of important commands, http://personality-project.org/r) for a tutorial for psychologists.
5. Online and hard copy books

A few of the most useful data manipulations functions (adapted from Rpad-refcard). Use ? for details

file.choose	() find a file
file.choose	(new=TRUE) create a new file
read.table	(filename)
read.csv	(filename) reads a comma separated file
read.delim	(filename) reads a tab delimited file
c	(...) combine arguments
from:to	e.g., 4:8
seq	(from,to, by)
rep	(x,times) repeat x
gl	(n,k,...) generate factor levels
matrix	(x,nrow=,ncol=) create a matrix
dim	(x) dimensions of x
data.frame	(...) create a data frame
list	(...) create a list
colnames	(x)
rownames	(x)
rbind	(...) combine by rows
cbind	(...) combine by columns
is.na	(x) also is.null(x), is...
na.omit	(x) ignore missing data
table	(x)
merge	(x,y)
as.matrix	(x) convert to a matrix,
as.data.frame	(x) convert to a data.frame
ls	() show workspace
rm	() remove variables from workspace

More useful statistical functions, Use ? for details

mean	(x)
is.na	(x) also is.null(x), is...
na.omit	(x) ignore missing data
sum	(x)
rowSums	(x) see also colSums(x)
min	(x)
max	(x)
range	(x)
table	(x)
summary	(x) depends upon x
sd	(x) standard deviation
cor	(x) correlation
cov	(x) covariance
solve	(x) inverse of x
lm	`(y~x) linear model`
aov	`(y~x) ANOVA`

Selected functions from *psych* package

describe	(x) descriptive stats
describe.by	(x,y) descriptives by group
pairs.panels	(x) SPLOM
error.bars	(x) means + error bars
error.bars.by	(x) Error bars by groups
fa	(x) Factor analysis
iclust	(x) Item cluster analysis
score.items	(x) score multiple scales
score.multiple.choice	(x) score multiple choice scales
alpha	(x) Cronbach's alpha
omega	(x) MacDonald's omega
irt.fa	(x) Item response theory through factor analysis

More psych commands

Simulation functions

sim	a factor simplex
sim.simplex	an item simplex
sim.item	items with 2 dimensional simple structure
sim.circ	items in a circumplex structure
sim.congeneric	items for a congeneric measurement model
sim.hierarchical	items with a hierarchical factor structure
sim.rasch	Rasch items
sim.irt	1-4 parameter IRT items
sim.structural	a general structural model
sim.anova	for ANOVA and lm problems

Graphical displays of structure

diagram	a generic set of diagram tools
fa.diagram	Show a factor structure
omega.diagram	Show Schmid Leiman structures
ICLUST.diagram	draw a cluster tree
plot.psych	a generic call for various plots additional data displays
error.crosses	two way error bars
biplot.psych	Plot factors and scores on same graph
draw.tetra	Show a tetrachoric correlation
scatter.hist	scatter plot with histogram

Writing your own function

1. At first, one just has a few lines of syntax that are repeatedly used
   - This could be any routine operation that you do
   - Probably hard coded and needing minor modifications each time.
2. Think of making this into a short function
   - Specify the input parameters
   - Return either a single value, vector or matrix or return a list
3. Test the function
   - Modify it a little to be more general
   - Perhaps specify a few default values
4. Add this to your file of frequently used operations.
5. To see how other functions work, just type in their name
   - Copy it to you text editor
   - Change a few lines
   - Paste it back into R (you must say the name <- function(....)

Writing functions is more typically “adapting” a function

1. Many functions do almost what you want to do, but not quite.
    - Their defaults are not what you like
    - You might see a way of adding something
2. Learn by reading other people's code
    - Either directly from the console
    - Download the source from CRAN
3. Try to understand what the person is doing
    - Styles differ
    - Use a style you like
    - Document your work
4. If you find a bug
    - Write the package maintainer
    - Say what you did, what you expected, what you got
    - R is a community, be helpful

Getting information about a package and its contents

CRAN – Package psych

http://cran.r-project.org/web/packages/psych/index.html

psych CRAN

Bill's Apple Yahoo! scholar.google.com Google Maps Wikipedia YouTube News (542) Popular CRAN Package

psych: Procedures for Psychological, Psychometric, and Personality Research

A number of routines for personality, psychometrics and experimental psychology. Functions are primarily for scale construction using factor analysis, cluster analysis and reliability analysis, although others provide basic descriptive statistics. Item Response Theory is done using factor analysis of tetrachoric and polychoric correlations. Functions for simulating particular item and test structures are included. Several functions serve as a useful front end for structural equation modeling. Graphical displays of path diagrams, factor analysis and structural equation models are created using basic graphics. Some of the functions are written to support a book on psychometrics as well as publications in personality research. For more information, see the personality-project.org/r webpage.

Version:	1.0-98
Suggests:	MASS, GPArotation, graph, Rgraphviz, mvtnorm, polycor, sem, Rcsdp, lavaan
Published:	2011-06-09
Author:	William Revelle
Maintainer:	William Revelle <revelle at northwestern.edu>
License:	GPL (≥ 2)
URL:	http://personality-project.org/r, http://personality-project.org/r/psych.manual.pdf
Citation:	psych citation info
In views:	Psychometrics
CRAN checks:	psych results

Downloads:

Package source:	psych_1.0-98.tar.gz
MacOS X binary:	psych_1.0-98.tgz
Windows binary:	psych_1.0-98.zip
Reference manual:	psych.pdf
Vignettes:	Overview of the psych package input for sem
Old sources:	psych archive

Reverse dependencies:

Reverse depends:	DeducerPlugInScaling, HDMD, lmSupport, nFactors, nonparaeff, random.polychor.pa
Reverse imports:	qgraph

A few final thoughts

1. Topics not discussed
   - Multilevel modeling is done in *multilevel*, *nlme*
   - Graphics can be done in *lattice* (implementation of Trellis), or *ggobi*
   - Network analysis in *sna* and *qgraph*
   - `Sweave` allows for automatic report generation embedded in LaTeXor OpenOffice.
2. R is a journey, you learn by doing but never master it
   - R is merely a tool for helping us do better research
   - R allows us to ask questions that we want to ask, not those that others have asked already
3. Warning: R can be addictive and lead to proselytizing.