# An introduction to R
# Sponsored by
# The Association of Psychological Science
# and
# Society of Multivariate Experimental Psychology

William Revelle

Department of Psychology
Northwestern University
Evanston, Illinois USA

## Outline

## 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.
    - 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."
- Q: My institute has been heavily dependent on SAS for the past while, and SAS is starting to charge us a very deep amount for license renewal.... The team is [condidering] switching to R, ... I am talking about the entire institute with considerable number of analysts using SAS their entire career. ... What kind of problems and challenges have you faced?
  A: One of your challenges will be that with the increased productivity of the team you will have time for more intellectually challenging problems. That frustrates some people.

## 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.”
- Q: Are R packages bug free?
- A: No. But bugs are fixed rapidly when identified.
- Q: How does function x work? May I adapt it for my functions.
- A: Look at the code. Borrow what you need.

## What is R?: Technically

- R is an open source implementation of S (The statistical language developed at Bell Labs). (S-Plus is a commercial implementation)
- R is a language and environment for statistical computing and graphics. R is available under GNU Copy-left
- R is a group project run by a core group of developers (with new releases semiannually). The current version of R is 3.1.0
- R is an integrated suite of software facilities for data manipulation, calculation and graphical display.

(Adapted from Robert Gentleman and the r-project.org web page)

## R is an integrated suite of software facilities for data manipulation, calculation and graphical display. It is:

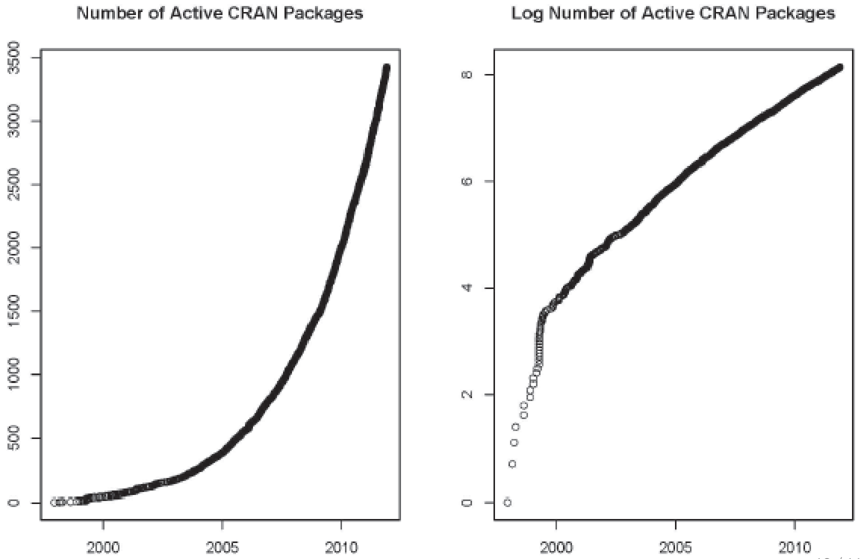
1. an effective data handling and storage facility,
2. a suite of operators for calculations on arrays, in particular matrices,
3. a large, coherent, integrated collection of intermediate tools for data analysis,
4. graphical facilities for data analysis and display either on-screen or on hardcopy, and
5. a well-developed, simple and effective programming language which includes conditionals, loops, user-defined recursive functions and input and output facilities.

Many users think of R as a statistics system. We prefer to think of it of an environment within which statistical techniques are implemented. R can be extended (easily) via packages ... available through the CRAN family of Internet sites covering a very wide range of modern statistics. (Adapted from r-project.org web page)

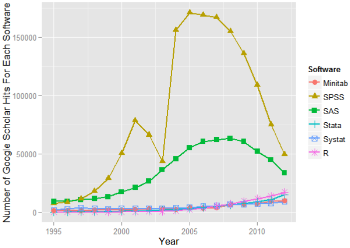
## R: A brief history

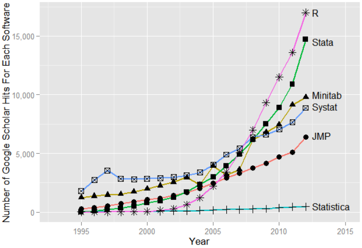
- 1991-93: Ross Dhaka and Robert Gentleman begin work on R project at U. Auckland
- 1995: R available by ftp under the GPL
- 96-97: mailing list and R core group is formed
- 2000: John Chambers, designer of S joins the Rcore (wins a prize for best software from ACM for S)
- 2001-2014: Core team continues 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
    - 2012-08-27: 4,000 packages
    - 2014-05-16: 5,547 packages (on CRAN) + 824 bioinformatic

# Rapid and consistent growth in packages contributed to R



# Popularity compared to other statistical packages




http://r4stats.com/articles/popularity/ considers various measures of popularity

1. discussion groups
2. blogs
3. Google Scholar citations ($> 14,000$ citations, $\approx 1,800/year$)
4. Google Page rank

## R as a way of facilitating replicable science

1. R scripts are published in our journals to show new methods
   - *Psychological Methods*
   - *Psychological Science*
   - *Journal of Research in Personality*
2. R based data sets are now accompanying journal articles
   - The *Journal of Research in Personality* now accepts R code and data sets.
   - JRP special issue in R is coming this fall.
3. By sharing our code and data the field can increase the possibility of doing replicable science.

## Reproducible Research: Sweave and KnitR

*Sweave is a tool that allows to embed the R code for complete data analyses in LaTeXdocuments. The purpose is to create dynamic reports, which can be updated automatically if data or analysis change. Instead of inserting a prefabricated graph or table into the report, the master document contains the R code necessary to obtain it. When run through R, all data analysis output (tables, graphs, etc.) is created on the fly and inserted into a final LaTeXdocument. The report can be automatically updated if data or analysis change, which allows for truly reproducible research.*

Friedrich Leisch (2002). Sweave: Dynamic generation of statistical reports using literate data analysis. I

## 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 – some interesting facts

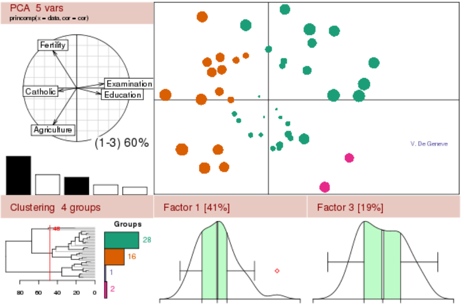
1. With a brief web based tutorial http://personality-project.org/r, 2nd and 3rd year undergraduates in psychological methods and personality research courses are using R for descriptive and inferential statistics and producing publication quality graphics.
2. More and more psychology departments are using it for graduate and undergraduate instruction.
3. R is easy to learn, hard to master
    - R-help newsgroup is very supportive
    - 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 for SPSS and SAS users trying to learn R (e.g., http://r4stats.com/) by Bob Muenchen (look for link to free version)).

## 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 3.1.0) (See a tutorial on how to install R and various packages at http://personality-project.org/r/psych)
- Start R
- Add useful packages (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



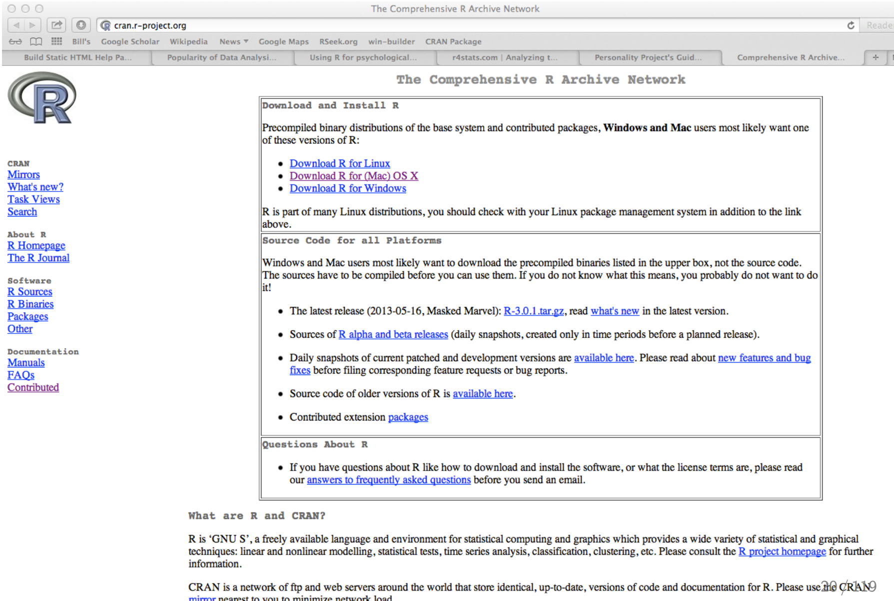

**Getting Started:**

- 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 3.1.0** (Spring Dance) has been released on 2014-04-10.
- **R version 3.0.3** (Warm Puppy) has been released on 2014-03-06.
- **The R Journal Vol.5/2** is available.
- **useR! 2013**, took place at the University of Castilla-La Mancha, Albacete, Spain, July 10-12 2013.
- **R version 2.15.3** (Security Blanket) has been released on 2013-03-01.

# Go to the Comprehensive R Archive Network (CRAN)



The Comprehensive R Archive Network

cran.r-project.org

Bill's Google Scholar Wikipedia News Google Maps RSeek.org win-builder CRAN Package

Build Static HTML Help Pa... | Popularity of Data Analysi... | Using R for psychological... | r4stats.com | Analyzing t... | Personality Project's Guid... | Comprehensive R Archive...

**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

## The Comprehensive R Archive Network

**Download and Install R**

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- Download R for Linux
- Download R for (Mac) OS X
- Download R for Windows

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

**Source Code for all Platforms**

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (2013-05-16, Masked Marvel): R-3.0.1.tar.gz, read what's new in the latest version.
- Sources of R alpha and beta releases (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are available here. Please read about new features and bug fixes before filing corresponding feature requests or bug reports.
- Source code of older versions of R is available here.
- Contributed extension packages

**Questions About R**

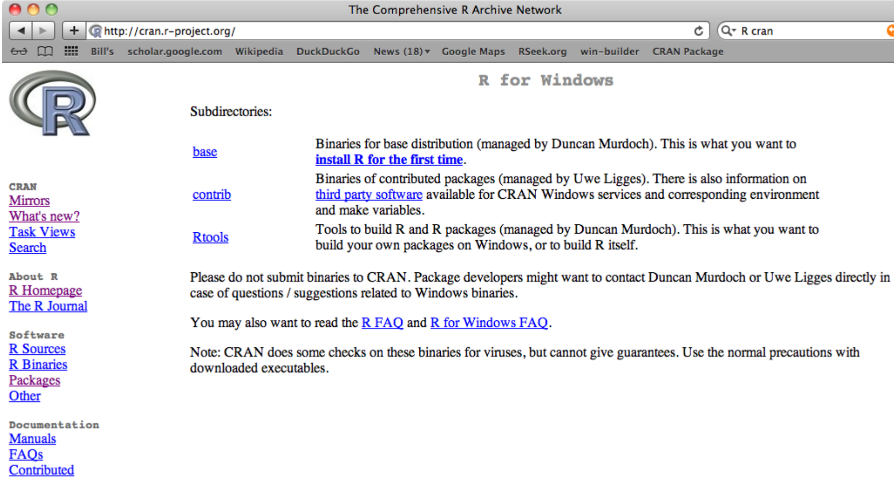
- If you have questions about R like how to download and install the software, or what the license terms are, please read our answers to frequently asked questions before you send an email.

**What are R and CRAN?**

R is 'GNU S', a freely available language and environment for statistical computing and graphics which provides a wide variety of statistical and graphical techniques: linear and nonlinear modelling, statistical tests, time series analysis, classification, clustering, etc. Please consult the R project homepage for further information.

CRAN is a network of ftp and web servers around the world that store identical, up-to-date, versions of code and documentation for R. Please use the CRAN mirror nearest to you to minimize network load.

# Download and install the appropriate version – PC



# Download and install the appropriate version – PC

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-3.1.0 for Windows (32/64 bit)

Download R 3.1.0 for Windows (54 megabytes, 32/64 bit)

Installation and other instructions
New features in this version

If you want to double-check that the package you have downloaded exactly matches the package distributed by R, you can compare the md5sum of the .exe to the true fingerprint. You will need a version of md5sum for windows: both graphical and command line versions are available.

### Frequently asked questions

- How do I install R when using Windows Vista?
- How do I update packages in my previous version of R?
- Should I run 32-bit or 64-bit R?

Please see the R FAQ for general information about R and the R Windows FAQ for Windows-specific information.

### Other builds

- Patches to this release are incorporated in the r-patched snapshot build.
- A build of the development version (which will eventually become the next major release of R) is available in the r-devel snapshot build.
- Previous releases

Note to webmasters: A stable link which will redirect to the current Windows binary release is
<CRAN MIRROR>/bin/windows/base/release.htm.

Last change: 2014-04-11, by Duncan Murdoch

# Download and install the appropriate version – Mac

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 10.6 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 10.5) and PowerPC Macs can be found in the old directory.

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

### R 3.1.0 "Spring Dance" released on 2014/04/10

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

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-3.1.0-snowleopard.pkg`
in the *Terminal* application to print the MD5 checksum for the R-3.1.0-snowleopard.pkg image. On Mac OS X 10.7 and later you can also validate the signature using `pkgutil --check-signature R-3.1.0-snowleopard.pkg`

### Files:

| | |
|---|---|
| R-3.1.0-snowleopard.pkg<br>MD5-hash: 60f9713fbb4c2868771de60ed3882009<br>SHA1-hash: 7943746b409897fc5c9ee5fb21e103c766b4ecfe<br>(ca. 68MB) | **R 3.1.0** binary for Mac OS X 10.6 (Snow Leopard) and higher, signed package. Contains R 3.1.0 framework, R.app GUI 1.64 in 64-bit for Intel Macs. 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.<br><br>This package contains the R framework, 64-bit GUI (R.app) and Tcl/Tk 8.6.0 X11 libraries. The latter component is optional and can be ommitted when choosing "custom install", it is only needed if you want to use the `tcltk` R package. GNU Fortran is **NOT** included (needed if you want to compile packages from sources that contain FORTRAN code) please see the tools directory. |
| R-3.1.0-mavericks.pkg<br>MD5-hash: 06c5465ef290b64efe25ac32790e816e<br>SHA1-hash: 5d1b93ab3569b7f126fb9ec7c196ef52134f9105<br>(ca. 55MB) | **R 3.1.0** binary for Mac OS X 10.9 (Mavericks) and higher, signed package. It contains the same software versions as above, but this R build has been built with Xcode 5 to leverage new compilers and functionalities in Mavericks not available in earlier OS X versions. |
| Mac-GUI-1.64.tar.gz<br>MD5-hash: 3c33bcda08a00f8778cfc978c7f806d5 | Sources for the R.app GUI 1.64 for Mac OS X. This file is only needed if you want to join the development of the GUI, it is not intended for regular users. Read the INSTALL file for further instructions. |
| NEWS (for Mac GUI) | News features and changes in the R.app Mac GUI |

The new R.app Cocoa GUI has been written by Simon Urbanek and Stefano Iacus with contributions from many developers and translators world-wide, see "About R" in the GUI.

### Subdirectories:

| | |
|---|---|
| tools | Additional tools necessary for building R for Mac OS X:<br>Universal GNU Fortran compiler for Mac OS X (see R for Mac tools page for details). |
| contrib | Binaries of package builds for Mac OS X 10.6 or higher (Snow Leopard build) |
| mavericks | Binaries of package builds for Mac OS X 10.9 or higher (Mavericks build) |
| leopard | Legacy binaries of universal (32-bit and 64-bit) package builds for Mac OS X 10.5 or higher (Leopard build) |
| universal | Legacy binaries of universal (32-bit) package builds for Mac OS X 10.4 (Tiger build) |

# Starting R on a PC

File Edit View Misc Packages Windows Help

R Console

```
R version 3.1.0 (2014-04-10) -- "Spring Dance"
Copyright (C) 2014 The R Foundation for Statistical Computing
Platform: i386-w64-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.

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.

> sessionInfo()
R version 3.1.0 (2014-04-10)
Platform: i386-w64-mingw32/i386 (32-bit)

locale:
[1] LC_COLLATE=English_United States.1252
[2] LC_CTYPE=English_United States.1252
```

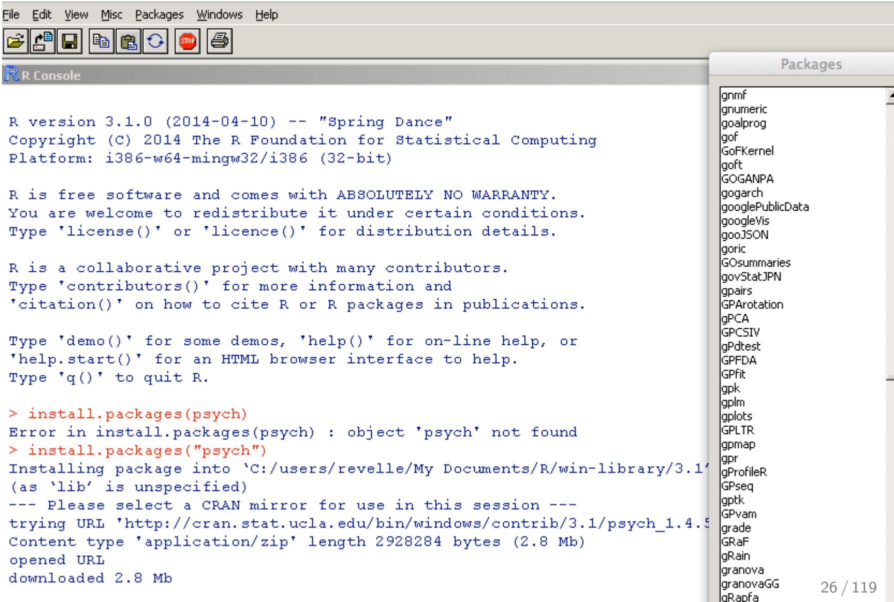
# Installing a package (psych) on a PC by hand – note error

File Edit View Misc Packages Windows Help

R Console

```
R version 3.1.0 (2014-04-10) -- "Spring Dance"
Copyright (C) 2014 The R Foundation for Statistical Computing
Platform: i386-w64-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.

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)
Error in install.packages(psych) : object 'psych' not found
> install.packages("psych")
Installing package into 'C:/users/revelle/My Documents/R/win-library/3.1'
(as 'lib' is unspecified)
--- Please select a CRAN mirror for use in this session ---
trying URL 'http://cran.stat.ucla.edu/bin/windows/contrib/3.1/psych_1.4.5.zip'
```

# Installing packages using the menu



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

```
R Under development (unstable) (2014-04-17 r65403) -- "Unsuffered Consequences"
Copyright (C) 2014 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin13.1.0 (64-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 1.65 (6738) x86_64-apple-darwin13.1.0]

[Workspace restored from /Users/revelle/.RData]
[History restored from /Users/revelle/.Rapp.history]
```

## Annotated installation guide: don't type the >

```
> install.packages("ctv")
> library(ctv)
> install.views("Psychometrics")
#or just install a few packages
> install.packages("psych")
> install.packages("GPArotation")
> install.packages("MASS")
> install.packages("mvtnorm")
```

- Install the task view installer package. You might have to choose a "mirror" site.
- Make it active
- Install all the packages in the "Psychometrics" task view. This will take a few minutes.
- Or, just install one package (e.g., psych)
- as well as a few suggested packages that add functionality for factor rotation, multivariate normal distributions, etc.

# Check the version number for R (should be ≥ 3.1.0) and for psych (≥1.4.5)

```
> library(psych)   #make the psych package active
> sessionInfo()    #what packages are active

R Under development (unstable) (2014-04-17 r65403)
Platform: x86_64-apple-darwin13.1.0 (64-bit)

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

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

other attached packages:
[1] psych_1.4.5
>
```

# R is extensible: The use of "packages"

1. More than 5,549 packages are available for R (and growing daily)
2. 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`("principal components") #will return 2,061 matches and reports the top 400
        - `findFn`("Item Response Theory") # will return 324 matches
        - `findFn`("INDSCAL ") # will return 7 matches.
3. install.packages("X") will install a particular package (add it to your R library – you need to do this just once)
4. 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
  - $>$ 5,500 known
  - $+$ ?

- General applications
  - most descriptive and inferential stats
  - Modern Applied Statistics with S
  - Lattice or Trellis graphics
  - Linear mixed-effects models
  - Personality/psychometrics general purpose
  - 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

# Questions?

## 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)

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

# 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.)

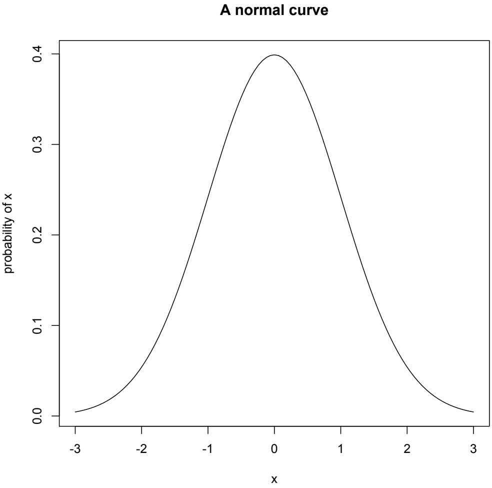
| Distribution | base name | P 1 | P 2 | P 3 | example application |
|---|---|---|---|---|---|
| *Normal* | norm | mean | sigma | | Most data |
| *Multivariate normal* | mvnorm | mean | r | sigma | Most data |
| *Log Normal* | lnorm | log mean | log sigma | | income or reaction time |
| *Uniform* | unif | 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 |
| $\chi^2$ | chisq | df | | nc | Testing for significance of $\chi^2$ |
| *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 |

## A very small list of the many data sets available

```
> data()

> data(package="psych")

> data(Titanic)
> ? Titanic

> data(cushny)
> ? cushney
```

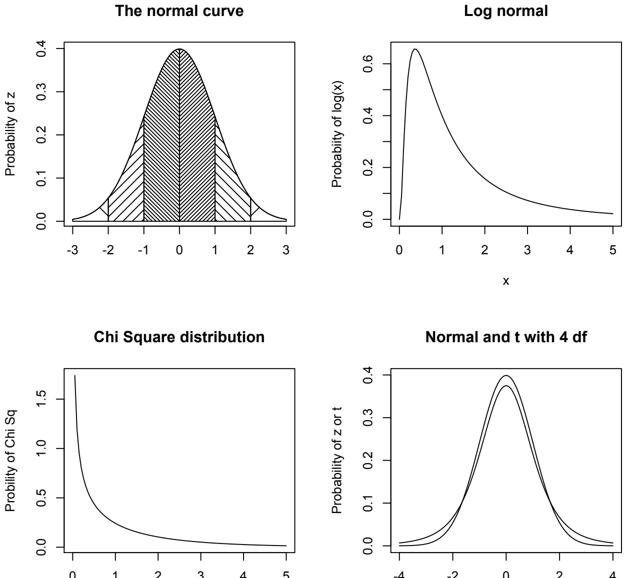
1. This opens up a separate text window and lists all of the data sets in the currently loaded packages.
2. Show the data sets available in a particular package (e.g., *psych*).
3. Gets the particular data set with its help file (e.g., the survival rates on the Titanic cross classified by age, gender and class).
4. Another original data set used by "student" (Gossett) for the t-test.

# R can draw distributions




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

# R can draw more interesting distributions

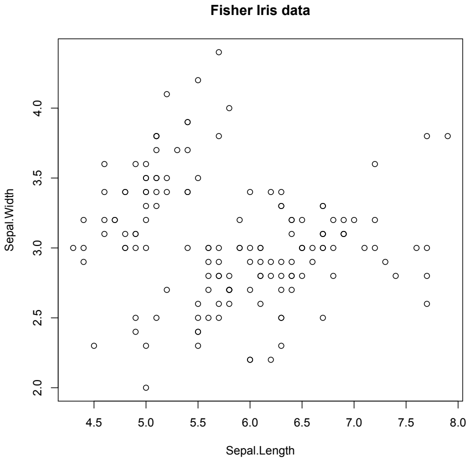


# R is also a graphics calculator

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

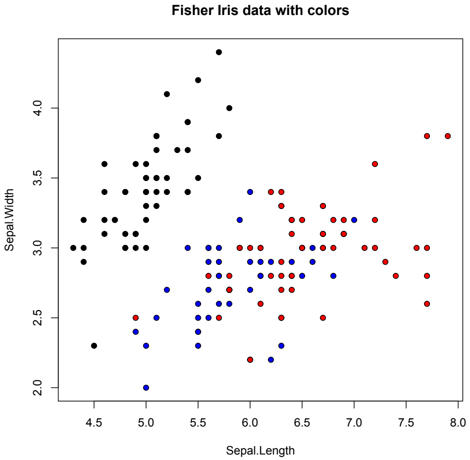
```
op <- par(mfrow=c(2,2))       #set up a 2 x 2 graph
curve(dnorm(x),-3,3,xlab="",ylab="Probability of z")
title(main="The normal curve",outer=FALSE)
xvals <-  seq(-3,-2,length=100)
dvals <- dnorm(xvals)
polygon(c(xvals,rev(xvals)),c(rep(0,100),rev(dvals)),density=2,angle=-45)
xvals <-  seq(-2,-1,length=100)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

# A simple scatter plot using `plot`




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

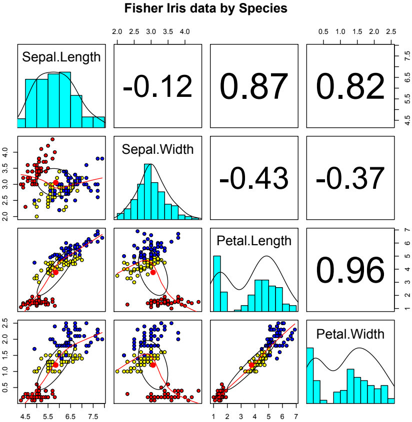
# A simple scatter plot using `plot` with some colors




1. Set parameters
2. bg for background colors
3. pch chooses the plot character

```
plot(iris[1:2],xlab="Sepal.Length",ylab="Sepal.Width"
+ ,main="Fisher Iris data with
colors",bg=c("black","blue","red")[iris[,5]],pch=21)
```

## 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")
```

# 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

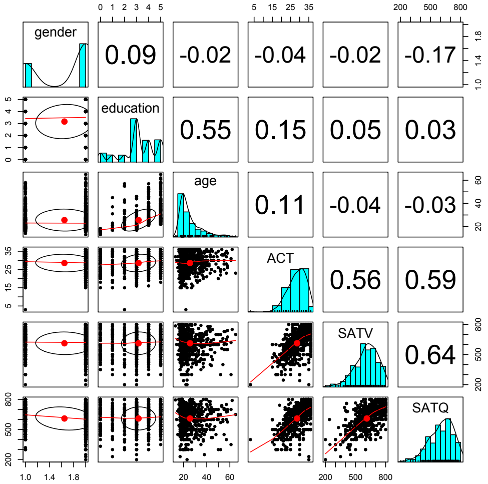
1. First read the data, either from a built in data set, a local file, a remote file, or from the clipboard.
2. Describe the data using the `describe` function from *psych*

```
> 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
> 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`. Use ?scrub for help on the parameters.

```
> cleaned <- scrub(my.data,"ACT",min=4)  #what data set, which variable, what value to fix
> describe(cleaned)   #look at the data again

          var   n   mean     sd median trimmed    mad min max range  skew kurtosis   se
gender      1 700   1.65   0.48      2    1.68   0.00   1   2     1 -0.61    -1.62 0.02
education   2 700   3.16   1.43      3    3.31   1.48   0   5     5 -0.68    -0.06 0.05
age         3 700  25.59   9.50     22   23.86   5.93  13  65    52  1.64     2.47 0.36
ACT         4 699  28.58   4.73     29   28.85   4.45  15  36    21 -0.50    -0.36 0.18
SATV        5 700 612.23 112.90    620  619.45 118.61 200 800   600 -0.64     0.35 4.27
SATQ        6 687 610.22 115.64    620  617.25 118.61 200 800   600 -0.59     0.00 4.41
```

## Find the pairwise correlations, round to 2 decimals

This also shows how two functions can be nested. We are rounding the output of the cor function.

```
 #specify all the parameters being passed
> round(cor(x=sat.act,use="pairwise"),digits=2)
 #the short way to specify the rounding parameter
> 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 |

## Display it differently using the lowerCor function

Operations that are done a lot may be made into your own functions. Thus, `lowerCor` finds the pairwise correlations, rounds to 2 decimals, displays the lower half of the correlation matrix, and then abbreviates the column labels to make them line up nicely

```
> lowerCor(sat.act)
```

```
          gendr edctn age   ACT   SATV  SATQ
gender     1.00
education  0.09  1.00
age       -0.02  0.55  1.00
ACT       -0.04  0.15  0.11  1.00
SATV      -0.02  0.05 -0.04  0.56  1.00
SATQ      -0.17  0.03 -0.03  0.59  0.64  1.00
```

# Testing the significance of one correlation using `cor.test`.

```
> cor.test(my.data$ACT,my.data$SATQ)


Pearson's product-moment correlation

data:  my.data$ACT and my.data$SATQ
t = 18.9822, df = 685, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.5358435 0.6340672
sample estimates:
      cor
0.5871122
```

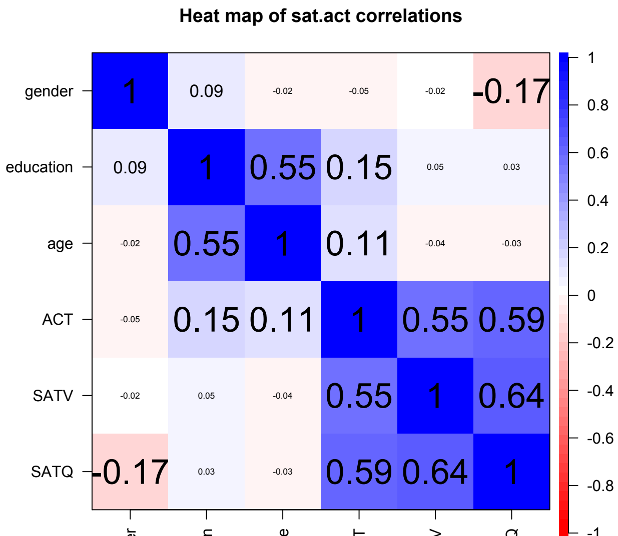
1. Specify the variables to correlate
2. Various statistics associated with the correlation.
3. But what if you want to do many tests? Use `corr.test`

# Test the correlations for significance using `corr.test`

```
> corr.test(cleaned)
Call:corr.test(x = sat.act)
Correlation matrix
          gender education   age   ACT  SATV  SATQ
gender      1.00      0.09 -0.02 -0.04 -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        -0.04      0.15  0.11  1.00  0.56  0.59
SATV       -0.02      0.05 -0.04  0.56  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       700 700 700  700  687
...
SATQ         687       687 687 687  687  687
Probability values (Entries above the diagonal are adjusted for multiple tests.
          gender education  age  ACT SATV SATQ
gender      0.00      0.17 1.00 1.00    1    0
education   0.02      0.00 0.00 0.00    1    1
age         0.58      0.00 0.00 0.03    1    1
ACT         0.33      0.00 0.00 0.00    0    0
SATV        0.62      0.22 0.26 0.00    0    0
SATQ        0.00      0.36 0.37 0.00    0    0
```

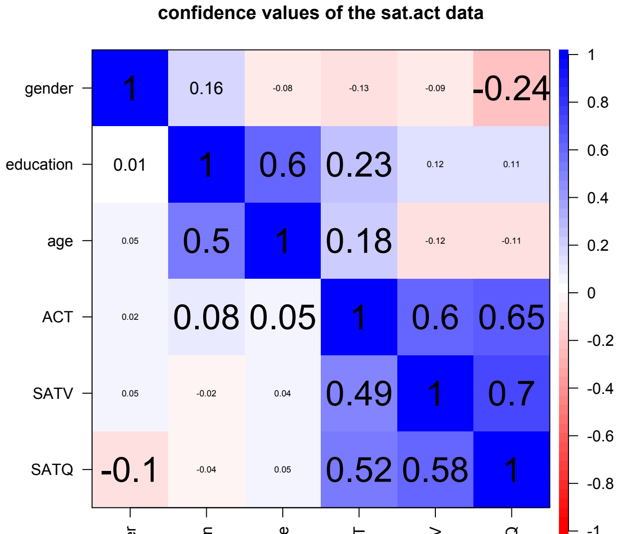
# The SAT.ACT correlations

```
ci <- cor.ci(cleaned,main='Heat map of sat.act')
```



## The SAT.ACT bootstrapped confidence intervals of correlation

cor.plot(ci,main='upper and lower confidence boundaries')



## Are education and gender independent? $\chi^2$ 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 $\chi^2$ 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.

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

```
          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
```

Note that we need to take the output of scale (which comes back as a matrix) and make it into a dataframe if we want to use the linear model on it.

# Zero center the data before examining interactions

```
> zsat <- data.frame(scale(my.data,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
```

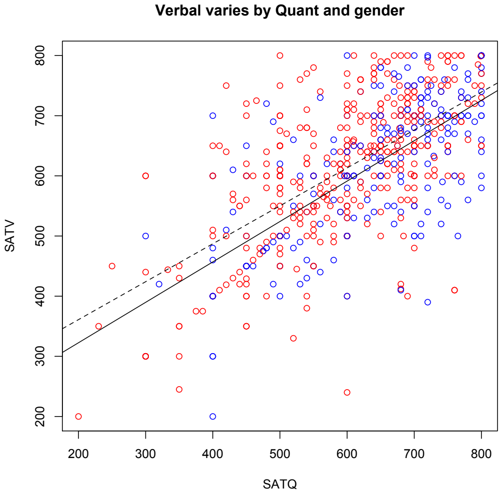
## Compare model 1 and model 2

Test the difference between the two linear models

```
> anova(mod1,mod2)
```

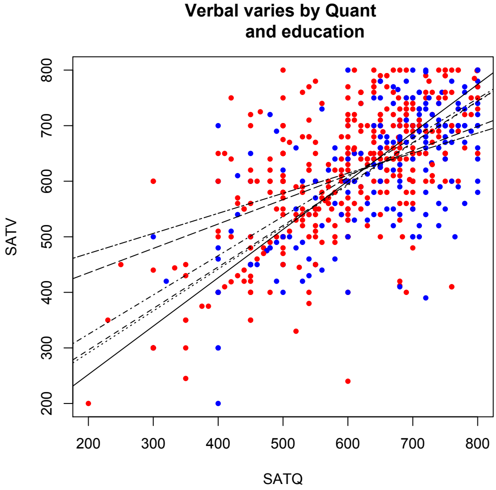
```
Analysis of Variance Table

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(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")
```

## Questions?

## 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 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 > 50 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 example.
Reise                             Seven data sets showing a bifactor solution.
affect                              Data sets of affect and arousal scores as a funct
                                            movie conditions (JPSP-12)
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
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
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.

## An brief diversion – https files

Although the next few examples work perfectly on http files, unfortunately, they do not work on https files. Some websites have switched to https and so we need to add a small fix. This did not make the psych version 1.4.5 release but if you copy the the following code into R it will allow us to read https files. You do not need to type in anything following the # : those are just comments. This is not necessary to do for http files.

```
"read.https" <-  function(filename,header=TRUE) { #define a new function
temp <- tempfile()   #create a temporary file
download.file(filename,destfile=temp,method="curl") #copy the https file to temp
result <- read.table(temp,header=header) #now, do the normal read.table command
unlink(temp) #get rid of the temporary file
return(result)}  #give us the result
```

Congratulations, you have just written your first R function.

# Reading from a local or remote file

1. Perhaps the standard way of reading in data is using the `read` command.
   - First must specify the location of the file
   - Can either type this in directly or use the `file.choose` function. This goes to your normal system file handler.
   - The file name/location can be a remote URL. (Note that read.file will not work on https files.)
2. Two examples of reading data

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

```
> 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
```

# Put it all together: read, show, describe

```
datafilename="http://personality-project.org/r/datasets/R.appendix1.data"
data.ex1<- read.table(datafilename,header=TRUE)    #unless it is https (see above)
dim(data.ex1)  #what are the dimensions of what we read?
data.ex1  #show the data
headTail(data.ex1) #just the top and bottom lines
describe(data.ex1) #descriptive stats
```

```
   Dosage Alertness
1       a        30
2       a        38
...  (rows deleted by hand)
17      c        20
18      c        19

> headTail(data.ex1) #just the top and bottom lines
    Dosage Alertness
1        a        30
2        a        38   'head' rows
3        a        35
4        a        41
...   <NA>       ... (rows automatically deleted)
15       c        17
16       c        21
17       c        20  'tail' rows
18       c        19
> describe(data.ex1) #descriptive stats
          vars  n  mean   sd median trimmed  mad min max range skew kurtosis   se
Dosage*      1 18  1.89 0.76      2    1.88 1.48   1   3     2 0.16    -1.35 0.18
Alertness    2 18 27.67 6.82     27   27.50 8.15  17  41    24 0.25    -1.06 1.61
```

1. Read the data from a remote file
2. Show all the cases (problematic if there are are 100s – 1000s)
3. Just show the first and last (4) lines
4. Find descriptive statistics

# Read a “foreign” file e.g., an SPSS sav file, using foreign package

`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?

# An example of reading from an SPSS file

```
> library(foreign)

> datafilename <- "http://personality-project.org/r/datasets/finkel.sav"

>  eli <- read.spss(datafilename,to.data.frame=TRUE,
                          use.value.labels=FALSE)
> headTail(eli,2,2)
> describe(eli,skew=FALSE)
```

| | USER | HAPPY | SOULMATE | ENJOYDEX | UPSET |
|---|---|---|---|---|---|
| 1 | "001" | 4 | 7 | 7 | 1 |
| 2 | "003" | 6 | 5 | 7 | 0 |
| ... | <NA> | ... | ... | ... | ... |
| 68 | "076" | 7 | 7 | 7 | 0 |
| 69 | "078" | 2 | 7 | 7 | 1 |

```
>
```

| | var | n | mean | sd | median | trimmed | mad | min | max | range | se |
|---|---|---|---|---|---|---|---|---|---|---|---|
| USER* | 1 | 69 | 35.00 | 20.06 | 35 | 35.00 | 25.20 | 1 | 69 | 68 | 2.42 |
| HAPPY | 2 | 69 | 5.71 | 1.04 | 6 | 5.82 | 0.00 | 2 | 7 | 5 | 0.13 |
| SOULMATE | 3 | 69 | 5.09 | 1.80 | 5 | 5.32 | 1.48 | 1 | 7 | 6 | 0.22 |
| ENJOYDEX | 4 | 68 | 6.47 | 1.01 | 7 | 6.70 | 0.00 | 2 | 7 | 5 | 0.12 |
| UPSET | 5 | 69 | 0.41 | 0.49 | 0 | 0.39 | 0.00 | 0 | 1 | 1 | 0.06 |

1. Make the *foreign* package active
2. Specify the name (and location) of the file to read
3. Read from a SPSS file
4. Show the top and bottom 2 cases
5. Describe it to make sure it is right

## Simulate data (Remember to always call them simulated!)

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 (using `headTail`), and then get basic descriptive statistics. For this example, we will use a built in data set.

```
> headTail(epi.bfi)
```

| | 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 |

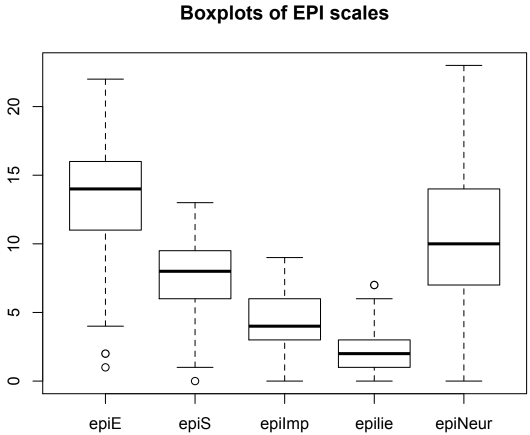
epi.bfi has 231 cases from two personality measures.

# Now find the descriptive statistics for this data set

```
> describe(epi.bfi)

         var   n   mean    sd median trimmed   mad min max range  skew kurtosis   se
epiE       1 231  13.33  4.14     14   13.49  4.45   1  22    21 -0.33    -0.01 0.27
epiS       2 231   7.58  2.69      8    7.77  2.97   0  13    13 -0.57     0.04 0.18
epiImp     3 231   4.37  1.88      4    4.36  1.48   0   9     9  0.06    -0.59 0.12
epilie     4 231   2.38  1.50      2    2.27  1.48   0   7     7  0.66     0.30 0.10
epiNeur    5 231  10.41  4.90     10   10.39  4.45   0  23    23  0.06    -0.46 0.32
bfagree    6 231 125.00 18.14    126  125.26 17.79  74 167    93 -0.21    -0.22 1.19
bfcon      7 231 113.25 21.88    114  113.42 22.24  53 178   125 -0.02     0.29 1.44
bfext      8 231 102.18 26.45    104  102.99 22.24   8 168   160 -0.41     0.58 1.74
bfneur     9 231  87.97 23.34     90   87.70 23.72  34 152   118  0.07    -0.51 1.54
bfopen    10 231 123.43 20.51    125  123.78 20.76  73 173   100 -0.16    -0.11 1.35
bdi       11 231   6.78  5.78      6    5.97  4.45   0  27    27  1.29     1.60 0.38
traitanx  12 231  39.01  9.52     38   38.36  8.90  22  71    49  0.67     0.54 0.63
stateanx  13 231  39.85 11.48     38   38.92 10.38  21  79    58  0.72     0.04 0.76
```

# Boxplots are a convenient descriptive device

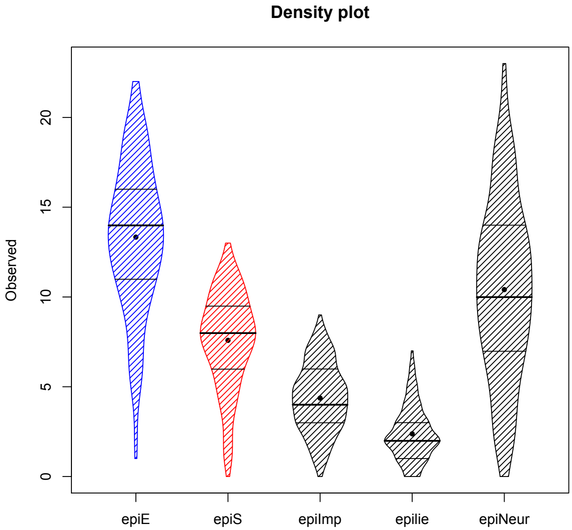
Show the Tukey “boxplot” for the Eysenck Personality Inventory




Use the box plot function

```
my.data <- epi.bfi
boxplot(my.data[1:5])
```

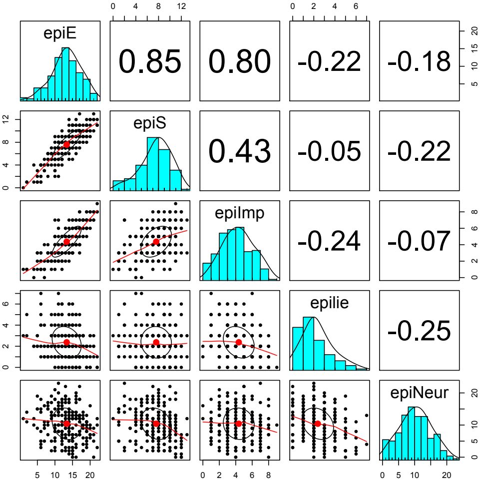
# An alternative display is a 'violin' plot (available as `violinBy`)




Use the `violinBy` function from *psych*

```
violinBy(my.data[1:5])
```

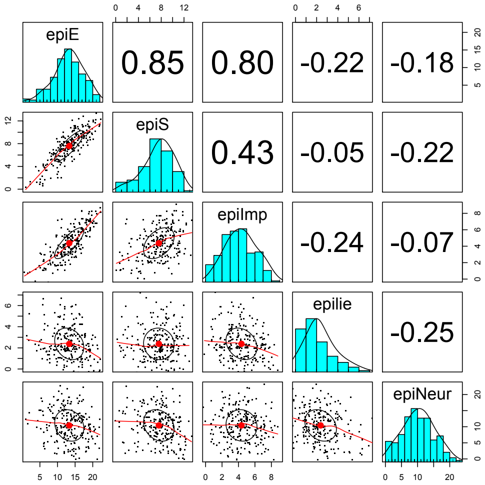
# Plot the scatter plot matrix (SPLOM) of the first 5 variables using the `pairs.panels` function




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

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

# Plot the scatter plot matrix (SPLOM) of the first 5 variables using the `pairs.panels` function but with smaller pch and jittering the points.




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

```
pairs.panels(my.data[1:5],pch='.',
     jiggle=TRUE)
```

# 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   bdi traitanx stateanx
epiE      1.00  0.85   0.80  -0.22   -0.18    0.18 -0.11  0.54  -0.09   0.14 -0.16    -0.23    -0.13
epiS      0.85  1.00   0.43  -0.05   -0.22    0.20  0.05  0.58  -0.07   0.15 -0.13    -0.26    -0.12
epiImp    0.80  0.43   1.00  -0.24   -0.07    0.08 -0.24  0.35  -0.09   0.07 -0.11    -0.12    -0.09
epilie   -0.22 -0.05  -0.24   1.00   -0.25    0.17  0.23 -0.04  -0.22  -0.03 -0.20    -0.23    -0.15
epiNeur  -0.18 -0.22  -0.07  -0.25    1.00   -0.08 -0.13 -0.17   0.63   0.09  0.58     0.73     0.49
bfagree   0.18  0.20   0.08   0.17   -0.08    1.00  0.45  0.48  -0.04   0.39 -0.14    -0.31    -0.19
bfcon    -0.11  0.05  -0.24   0.23   -0.13    0.45  1.00  0.27   0.04   0.31 -0.18    -0.29    -0.14
bfext     0.54  0.58   0.35  -0.04   -0.17    0.48  0.27  1.00   0.04   0.46 -0.14    -0.39    -0.15
bfneur   -0.09 -0.07  -0.09  -0.22    0.63   -0.04  0.04  0.04   1.00   0.29  0.47     0.59     0.49
bfopen    0.14  0.15   0.07  -0.03    0.09    0.39  0.31  0.46   0.29   1.00 -0.08    -0.11    -0.04
bdi      -0.16 -0.13  -0.11  -0.20    0.58   -0.14 -0.18 -0.14   0.47  -0.08  1.00     0.65     0.61
traitanx -0.23 -0.26  -0.12  -0.23    0.73   -0.31 -0.29 -0.39   0.59  -0.11  0.65     1.00     0.57
stateanx -0.13 -0.12  -0.09  -0.15    0.49   -0.19 -0.14 -0.15   0.49  -0.04  0.61     0.57     1.00
```

# Find the correlations for this data set, round off to 2 decimal places using `lowerCor`

```
> lowerCor(my.data)

         epiE  epiS  epImp epili epiNr bfagr bfcon bfext bfner bfopn bdi   trtnx sttnx
epiE      1.00
epiS      0.85  1.00
epiImp    0.80  0.43  1.00
epilie   -0.22 -0.05 -0.24  1.00
epiNeur  -0.18 -0.22 -0.07 -0.25  1.00
bfagree   0.18  0.20  0.08  0.17 -0.08  1.00
bfcon    -0.11  0.05 -0.24  0.23 -0.13  0.45  1.00
bfext     0.54  0.58  0.35 -0.04 -0.17  0.48  0.27  1.00
bfneur   -0.09 -0.07 -0.09 -0.22  0.63 -0.04  0.04  0.04  1.00
bfopen    0.14  0.15  0.07 -0.03  0.09  0.39  0.31  0.46  0.29  1.00
bdi      -0.16 -0.13 -0.11 -0.20  0.58 -0.14 -0.18 -0.14  0.47 -0.08  1.00
traitanx -0.23 -0.26 -0.12 -0.23  0.73 -0.31 -0.29 -0.39  0.59 -0.11  0.65  1.00
stateanx -0.13 -0.12 -0.09 -0.15  0.49 -0.19 -0.14 -0.15  0.49 -0.04  0.61  0.57  1.00
```

# Test the significance and use Holm correction for multiple tests

```
> corr.test(my.data)
Call:corr.test(x = my.data)
Correlation matrix
          epiE  epiS epiImp epilie epiNeur bfagree bfcon bfext bfneur bfopen   bdi traitanx stateanx
epiE      1.00  0.85   0.80  -0.22   -0.18    0.18 -0.11  0.54  -0.09   0.14 -0.16    -0.23    -0.13
epiS      0.85  1.00   0.43  -0.05   -0.22    0.20  0.05  0.58  -0.07   0.15 -0.13    -0.26    -0.12
epiImp    0.80  0.43   1.00  -0.24   -0.07    0.08 -0.24  0.35  -0.09   0.07 -0.11    -0.12    -0.09
..
stateanx -0.13 -0.12  -0.09  -0.15    0.49   -0.19 -0.14 -0.15   0.49  -0.04  0.61     0.57     1.00
Sample Size
         epiE epiS epiImp epilie epiNeur bfagree bfcon bfext bfneur bfopen bdi traitanx stateanx
epiE      231  231    231    231     231     231   231   231    231    231 231      231      231
..
stateanx  231  231    231    231     231     231   231   231    231    231 231      231      231
Probability values (Entries above the diagonal are adjusted for multiple tests.)
         epiE epiS epiImp epilie epiNeur bfagree bfcon bfext bfneur bfopen  bdi traitanx stateanx
epiE     0.00 0.00   0.00   0.03    0.27    0.27  1.00  0.00   1.00   1.00 0.59     0.02     1.00
epiS     0.00 0.00   0.00   1.00    0.04    0.08  1.00  0.00   1.00   0.62 1.00     0.00     1.00
epiImp   0.00 0.00   0.00   0.01    1.00    1.00  0.01  0.00   1.00   1.00 1.00     1.00     1.00
epilie   0.00 0.43   0.00   0.00    0.01    0.32  0.03  1.00   0.03   1.00 0.08     0.02     0.61
epiNeur  0.01 0.00   0.26   0.00    0.00    1.00  1.00  0.33   0.00   1.00 0.00     0.00     0.00
bfagree  0.01 0.00   0.23   0.01    0.21    0.00  0.00  0.00   1.00   0.00 0.95     0.00     0.12
bfcon    0.08 0.48   0.00   0.00    0.04    0.00  0.00  0.00   1.00   0.00 0.25     0.00     1.00
bfext    0.00 0.00   0.00   0.50    0.01    0.00  0.00  0.00   1.00   0.00 0.99     0.00     0.76
bfneur   0.15 0.30   0.18   0.00    0.00    0.50  0.50  0.57   0.00   0.00 0.00     0.00     0.00
bfopen   0.04 0.02   0.30   0.70    0.19    0.00  0.00  0.00   0.00   0.00 1.00     1.00     1.00
bdi      0.02 0.04   0.11   0.00    0.00    0.03  0.01  0.03   0.00   0.25 0.00     0.00     0.00
traitanx 0.00 0.00   0.07   0.00    0.00    0.00  0.00  0.00   0.00   0.11 0.00     0.00     0.00
stateanx 0.05 0.07   0.18   0.02    0.00    0.00  0.04  0.02   0.00   0.52 0.00     0.00     0.00
>
```

# t.test demonstration with Student's data (from the sleep dataset)

*sleep*

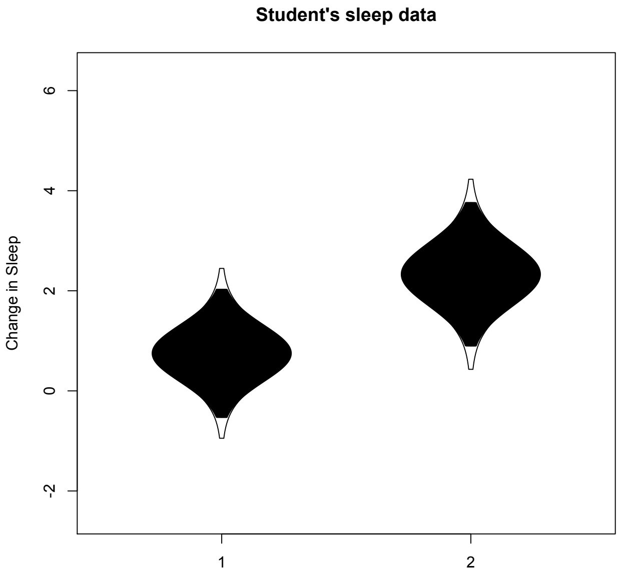
```
> sleep
   extra group ID
1    0.7     1  1
2   -1.6     1  2
3   -0.2     1  3
4   -1.2     1  4
5   -0.1     1  5
6    3.4     1  6
7    3.7     1  7
...
13   1.1     2  3
14   0.1     2  4
15  -0.1     2  5
16   4.4     2  6
17   5.5     2  7
18   1.6     2  8
19   4.6     2  9
20   3.4     2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

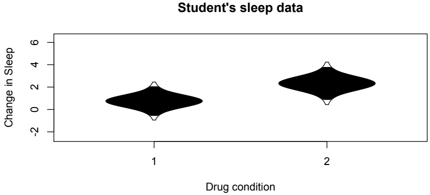
But the data were actually paired. Do it for a paired t-test

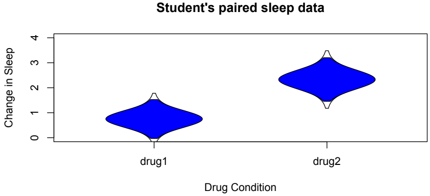
```
> with(sleep,t.test(extra~group,paired=TRUE))

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="https://personality-project.org/r/datasets/R.appendix2.data"
data.ex2=read.https(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 varian
summary(aov.ex2)                                     #show the summary table
```

```
 > aov.ex2 = aov(Alertness~Gender*Dosage,data=data.ex2)  #do the analysis of va
> 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
4. See also the tutorial by Jason French at http://gradstudents.wcas.northwestern.edu/~jaf502/tutorials/repeatedmeasures.html

# 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
```

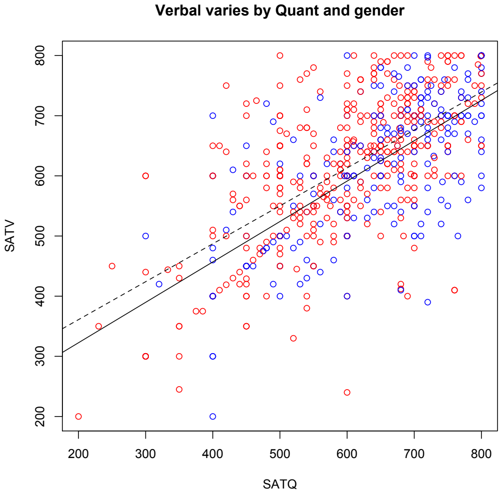
## Compare model 1 and model 2

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")
```

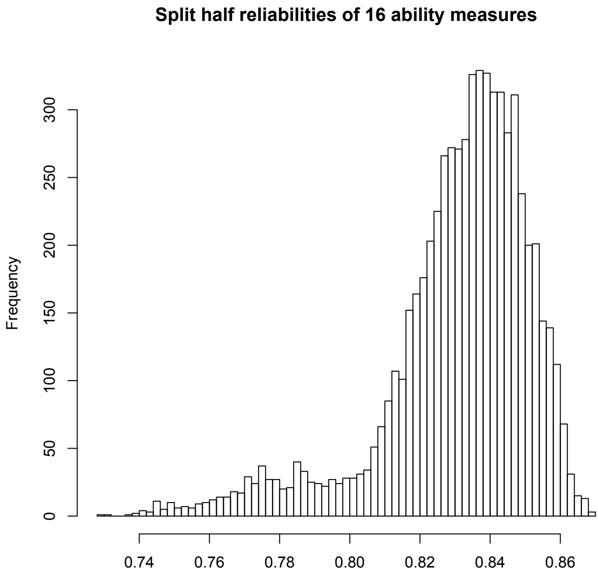
# Psychometrics

1. Classical test theory measures of reliability
   - Scoring tests
   - Reliability (alpha, beta, omega)
2. Multivariate Analysis
   - Factor Analysis
   - Components analysis
   - Multidimensional scaling
   - Structural Equation Modeling
3. Item Response Theory
   - One parameter (Rasch) models
   - 2PL and 2PN models

## Classic theory estimates of reliability

1. Scoring tests
   - `scoreItems` 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` $\alpha$ reliability of a single scale finds the average split half reliability. (some items may be reversed keyed).
   - `omega` $\omega_h$ reliability of a single scale estimates the general factor saturation of the test.
   - `guttman` Find the 6 Guttman reliability estimates
   - `splitHalf` Find the range of split half reliabilities

# 6,435 split half reliabilities of a 16 item ability test



# Finding coefficient $\alpha$ for a scale (see Revelle and Zinbarg, 2009, however, for why you should not)

```
Reliability analysis
Call: alpha(x = ability)

  raw_alpha std.alpha G6(smc) average_r S/N    ase mean   sd
       0.83      0.83    0.84      0.23 4.9 0.0086 0.51 0.25

 lower alpha upper     95% confidence boundaries
0.81 0.83 0.85

 Reliability if an item is dropped:
          raw_alpha std.alpha G6(smc) average_r S/N alpha se
reason.4       0.82      0.82    0.82      0.23 4.5   0.0093
reason.16      0.82      0.82    0.83      0.24 4.7   0.0091
...
rotate.6       0.82      0.82    0.82      0.23 4.5   0.0092
rotate.8       0.82      0.82    0.83      0.24 4.6   0.0091

 Item statistics
             n    r r.cor r.drop mean   sd
reason.4  1442 0.58  0.54   0.50 0.68 0.47
reason.16 1463 0.50  0.44   0.41 0.73 0.45
r...
```

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

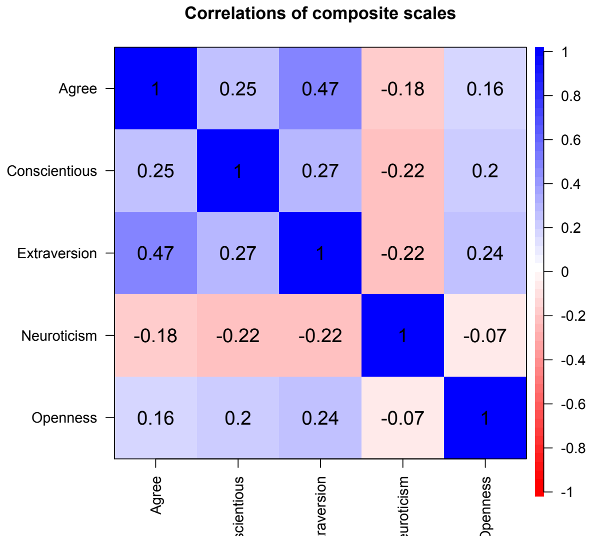
```
> keys.list <- list(Agree=c(-1,2:5),Conscientious=c(6:8,-9,-10),Extraversion=c(-11,-12,13:15),
                              Neuroticism=c(16:20),Openness = c(21,-22,23,24,-25))
> keys <- make.keys(bfi,keys.list)
> scores <- scoreItems(keys,bfi)

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
```

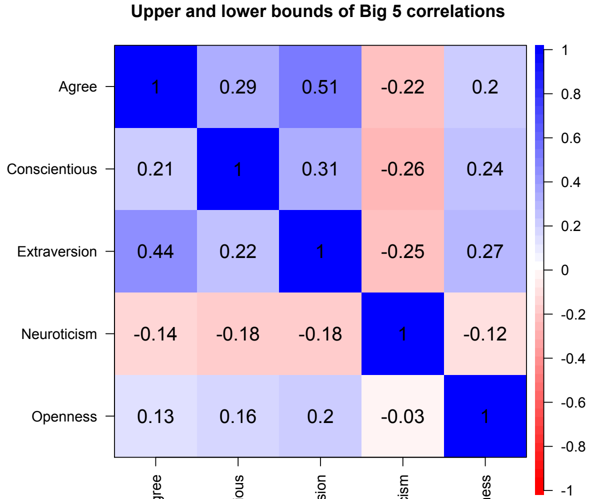
# Correlations of composite scores based upon item correlations

ci <- cor.ci(bfi,keys=keys,main='Correlations of composite scales')



# Upper and Lower bounds of Correlations of composite scores based upon item correlations

cor.plot(ci,main='Upper and lower bounds of Big 5 correlations')



# Factor analysis of Thurstone 9 variable problem

```
> f3 <- fa(Thurstone,3) #use this built in dataset
> 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

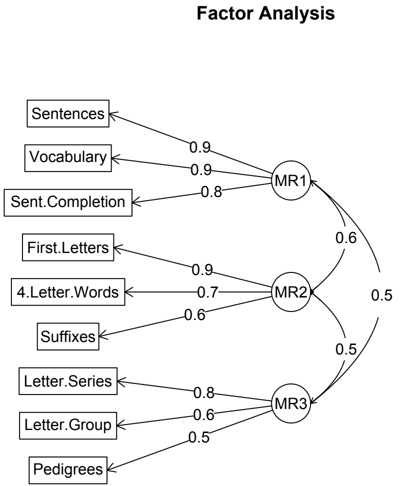
```
Test of the hypothesis that 3 factors are sufficient.

The degrees of freedom for the null model are  36  and the objective function was  5.2 with Chi Square of
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         0.77  0.91  0.96 -0.12 -0.04  0.07 -0.03  0.04  0.14
Vocabulary        0.85  0.89  0.95 -0.01  0.06  0.10 -0.12 -0.03  0.04
Sent.Completion   0.73  0.83  0.87 -0.04  0.04  0.13 -0.08  0.00  0.12
First.Letters    -0.06  0.00  0.10  0.68  0.86  0.93 -0.13  0.00  0.13
4.Letter.Words   -0.14 -0.01  0.07  0.58  0.74  0.86  0.01  0.10  0.25
Suffixes          0.07  0.18  0.27  0.46  0.63  0.76 -0.20 -0.08  0.06
Letter.Series    -0.04  0.03  0.13 -0.10 -0.01  0.10  0.56  0.84  0.93
Pedigrees         0.25  0.37  0.46 -0.16 -0.05  0.08  0.27  0.47  0.66
Letter.Group     -0.16 -0.06  0.06  0.09  0.21  0.31  0.44  0.64  0.79

 Interfactor correlations and bootstrapped confidence intervals
  lower estimate upper
1  0.40     0.59  0.64
2  0.29     0.54  0.63
3  0.29     0.52  0.61
```

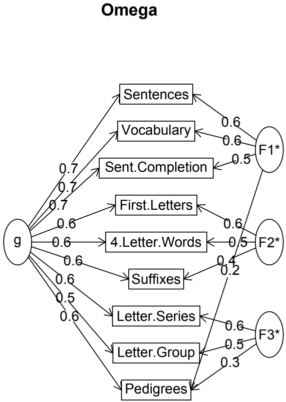
# The simple factor structure

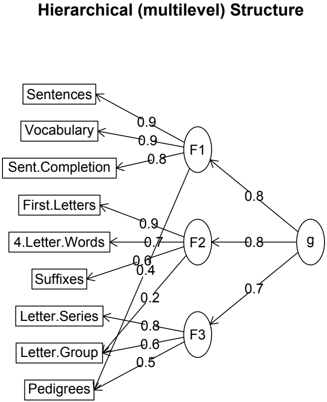
factor.diagram(f3) # show the diagram



# Two ways of viewing the higher order structure

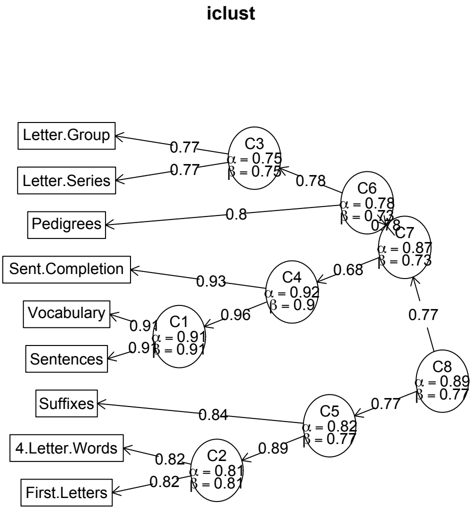
om <- omega(Thurstone) omega.diagram(om,sl=FALSE)



# A hierarchical cluster structure found by `iclust`

iclust(Thurstone)



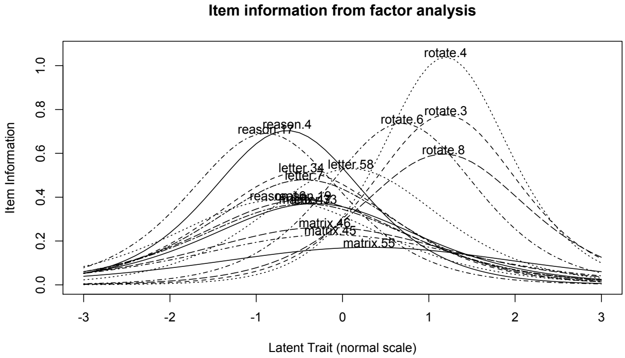
## Structural Equation modeling packages

1. sem (by John Fox and others)
    - uses RAM notation
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

# Mutiple packages to do Item Response Theory analysis

1. *psych* uses a factor analytic procedure to estimate item discriminations and locations
   - `irt.fa` finds either tetrachoric or polychoric correlation matrices
     - converts factor loadings to discriminations
   - `plot.irt` plots item information and item characteristic functions
   - look at examples for `irt.fa`
   - two example data sets: `ability` and `bfi`
2. Other packages include *ltm*, *eRm*, *mirt*, + others

# Item Response Information curves for 16 ability items from ICAR



## 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 |
| data.frame | (...) create a data frame |

| | |
|---|---|
| dim | (x) dimensions of x |
| str | (x) Structure of an object |
| list | (...) create a list |
| colnames | (x) set or find column names |
| rownames | (x) set or find row names |
| ncol(x), nrow(z) | number of row, columns |
| 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) |
| apply | (x,rc,FUNCTION) |
| 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,n) Factor analysis |
| principal | (x,n) Principal components |
| 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 |

# Questions?