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

William Revelle, David M. Condon & Sara Weston*
Northwestern University
Evanston, Illinois USA
*Washington University, St. Louis, USA

https://personality-project.org/r/aps/aps-short.pdf
Outline

What is R?
Where did it come from, why use it?
Installing R on your computer and adding packages
R-Applications and Packages
Installing and using packages
A brief example
Basic R capabilities: Calculation, Statistical tables, Graphics
A brief example of exploratory and confirmatory data analysis
Multiple regression modeling and graphics
Basic statistics and graphics
4 steps: read, explore, test, graph
Basic descriptive and inferential statistics
t-test, ANOVA, $\chi^2$ and regression
Linear Regression
Basic R commands
Basic R
Help
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
5. List of major commands and packages

Although programming is easy in R, that is beyond the scope of today
Where did it come from, why use it?

R: What is it?

1. R: An international collaboration
2. R: The open source - public domain version of S+
3. R: Written by statisticians (and some 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.
Where did it come from, why use it?

**Statistical Programs for Psychologists**

- **General purpose programs**
  - R
  - S+
  - SAS
  - SPSS
  - STATA
  - Systat

- **Specialized programs**
  - Mx
  - EQS
  - AMOS
  - LISREL
  - MPlus
  - Your favorite program
Where did it come from, why use it?

Statistical Programs for Psychologists

- General purpose programs
  - R
  - $+$
  - $A$
  - $P$
  - $TATA$
  - $y$tat

- Specialized programs
  - Mx (OpenMx is part of R)
  - EQ$
  - AMO$
  - LI$REL$
  - MPI$u$
  - 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 [considering] 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.
Where did it come from, why use it?

**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 3.2.0
- 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 as 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)
Where did it come from, why use it?

**R: A brief history**

- 1991-93: Ross Dhaka and Robert Gentleman begin work on R project for Macs at U. Auckland (S for Macs).
- 1995: R available by ftp under the General Public License.
- 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)
- 2001-2015: Core team continues to improve base package with a new release every 6 months (now more like yearly).
- 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 packages on BioConductor
  - 2015-05-20 6,678 packages (on CRAN) + 1024 bioinformatic packages + ?,000s on GitHub
Rapid and consistent growth in packages contributed to R
http://r4stats.com/articles/popularity/ considers various measures of popularity

1. discussion groups
2. blogs
3. Google Scholar citations (> 27,000 citations, ≈ 1,800/year)
4. Google Page rank
R as a way of facilitating replicable science

1. R is not just for statisticians, it is for all research oriented psychologists.

2. R scripts are published in psychology journals to show new methods:
   - *Psychological Methods*
   - *Psychological Science*
   - *Journal of Research in Personality*

3. 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 last fall.

4. 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 \LaTeX{} documents. 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 \LaTeX{} document. The report can be automatically updated if data or analysis change, which allows for truly reproducible research.


Supplementary material for journals can be written in Sweave/KnitR.
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 menu after 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 (usually)
   - Multiple web based and pdf tutorials see (e.g., http://www.r-project.org/)
   - Short courses using R for many applications. (Look at APS program).

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).
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 3.2.0 (Full of Ingredients) has been released on 2015-04-16.
- R version 3.1.3 (Smooth Sidewalk) has been released on 2015-03-09.
- The R Journal Volume 6/2 is available.
- useR! 2015, will take place at the University of Aalborg, Denmark, June 30 - July 3, 2015.
- useR! 2014, took place at the University of California, Los Angeles, USA June 30 - July 3, 2014.
Installing R on your computer and adding packages

Go to the Comprehensive R Archive Network (CRAN)

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!

- 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.
Installing R on your computer and adding packages

Download and install the appropriate version – PC

![R for Windows](http://cran.r-project.org/)

**Subdirectories:**

- **base**
  - Binaries for base distribution (managed by Duncan Murdoch). This is what you want to install R for the first time.

- **contrib**
  - Binaries of contributed packages (managed by Uwe Ligges). There is also information on third party software available for CRAN Windows services and corresponding environment and make variables.

- **Rtools**
  - Tools to build R and R packages (managed by Duncan Murdoch). This is what you want to build your own packages on Windows, or to build R itself.

Please do not submit binaries to CRAN. Package developers might want to contact Duncan Murdoch or Uwe Ligges directly in case of questions / suggestions related to Windows binaries.

You may also want to read the [R FAQ](http://cran.r-project.org/FAQ.html) and [R for Windows FAQ](http://cran.r-project.org/doc/manuals/R-FAQ-Windows.html).

Note: CRAN does some checks on these binaries for viruses, but cannot give guarantees. Use the normal precautions with downloaded executables.
Download and install the appropriate version – PC

R-3.2.0 for Windows (32/64 bit)

Download R 3.2.0 for Windows (62 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: 2015-04-17, by Duncan Murdoch
Installing R on your computer and adding packages

Download and install the appropriate version – Mac

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 3.2.0 "Full of Ingredients" released on 2015/04/18

This binary distribution of R and the GUI supports 64-bit Intel based Macs on Mac OS X 10.9 (Mavericks) 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

```bash
md5 R-3.2.0.pkg
```

in the Terminal application to print the MD5 checksum for the R-3.2.0.pkg image. On Mac OS X 10.7 and later you can also validate the signature using `pkgutil --check-signature R-3.2.0.pkg`

Files:

**R-3.2.0.pkg**

MD5 hash: `dbd66637f3bb4e39ad21d0877b24`

SHA1 hash: `671f64f8a33c8fd295e87f194`

Contains R 3.2.0 framework, R.app GUI 1.65 in 64-bit for Intel Macs, Tcl/Tk 8.6.0 X11 libraries and Textino 5.2. The latter two components are optional and can be omitted when choosing "custom install", it is only needed if you want to use the tcltk R package or build package documentation from sources.

Note: the use of X11 (including tcltk) requires XQuartz to be installed since it is no longer part of OS X. Always re-install XQuartz when upgrading your OS X to a new major version.

(If you are using legacy OS X 10.6 through 10.8 and are interested in R 3.2.0, please see the [R for Mac development page](#).)

**R-3.1.3-snowleopard.pkg**

MD5 hash: `6231f2b525c099b4d9b15c7`

**R 3.1.3** binary for Mac OS X 10.6 (Snow Leopard) and higher, signed package. Contains R 3.1.3 framework, R.app GUI 1.65 64-bit for Intel
Starting R on a PC

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

R version 3.2.0 (2015-04-16) -- "Full of Ingredients"
Copyright (C) 2015 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin13.4.0 (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 (6931) x86_64-apple-darwin13.4.0]

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

# >  is the prompt for all commands ... #is for comments
Check the version number for R (should be $\geq 3.2.0$) and for psych ($\geq 1.5.4$)

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

R version 3.2.0 (2015-04-16)
Platform: x86_64-apple-darwin13.4.0 (64-bit)
Running under: OS X 10.10.3 (Yosemite)

locale:

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

other attached packages:
[1] psych_1.5.4

loaded via a namespace (and not attached):
[1] parallel_3.2.0 mnormt_1.5-2
Various ways to run R

1. UNIX (and *NIX like) environments
   - Can be scripted for use on remote servers
   - Particularly fast if on remote processors with many cores
   - RStudio Server as “Integrated Development Environment” (IDE)
   - RStudio can be run remotely with a browser (e.g., even from an IPad)

2. PC
   - quasi GUI + text editor of choice
   - RStudio as “Integrated Development Environment” (IDE) (recommended by Sara)

3. Mac
   - R.app + text editor of choice (preferred by Bill)
   - RStudio as “Integrated Development Environment” (IDE) (preferred by David)
   - allows for multiple cores for parallel processing
R Studio is a useful “Integrated Development Environment” (IDE)

```{r}
library(psych)
myData <- sat.act
pairs.panels(myData)
describe(myData)
cleaned <- scrub(myData, "ACT", min=5)
describe(cleaned)
```
R Studio may be run on a remote server
R is extensible: The use of “packages”

1. More than 6,652 packages are available for R (and growing daily. It was 5,549 last year).
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 2,374 matches from 159 packages and reports the top 400
     - `findFn("Item Response Theory")` # will return 499 matches in 73 packages
     - `findFn("INDSCAL ")` # will return 13 matches in 7 packages.
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 (RAM path notation)
  - 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
A small subset of very useful packages (see also Computer World list)

- General use
  - devtools
  - readxl
  - foreign
  - RMySQL
  - readr
  - rio

- Special use
  - dplyr
  - plyr
  - data.table
  - knitr
  - sweave
  - ggplot2
  - > 5,500 known
  - + ?

- General applications
  - Get packages from GitHub
  - input from excel
  - input from SPSS, etc.
  - input from MySQL
  - fast input for very large csv files
  - simple to use integrated input/output

- More specialized packages
  - reshape from wide to long etc.
  - reshape
  - faster data handling for large data sets
  - integrate markdown documentation with R
  - integrate \LaTeX documentation with R
  - powerful grammar of graphics
  - Thousands of more packages on CRAN
  - Code on webpages/journal articles
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 3.2.0) (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
Annotated installation guide: don’t type the `>`

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

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

```r
> library(ctv)
```

- Make it active

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

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

```r
#or just install a few packages
> install.packages("psych", dependencies=TRUE)
```

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

```r
#which installs psych and its required packages
> install.packages("GPArotation")
> install.packages("mnormt")
```

- as well as a few suggested packages that add functionality for factor rotation, multivariate normal distributions, etc.
Questions?
Basic R commands – remember don’t enter the >

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

> 2 + 2    #sum two numbers
[1] 4    #show the output

> 3^4    #3 raised to the 4th
[1] 81    #that was easy

> sum(1:10)    #find the sum of the first 10 numbers
[1] 55    #the answer

> prod(c(1, 2, 3, 5, 7))    #the product of the concatenated (c) numbers
[1] 210    #Note how we combined product with concatenate

It is also a statistics table (the normal distribution, the t, the F, the χ² distribution, the xyz distribution)

> pnorm(q = 1)    #the probability of a normal with value of 1 sd
[1] 0.8413447    #

> pt(q = 2, df = 20)    #what about the probability of a t-test value of
[1] 0.9703672    #this is the upper tail
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.) Each function can be modified with various parameters.

<table>
<thead>
<tr>
<th>Distribution</th>
<th>base name</th>
<th>P 1</th>
<th>P 2</th>
<th>example application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>norm</td>
<td>mean</td>
<td>sigma</td>
<td>Most data</td>
</tr>
<tr>
<td>Multivariate normal</td>
<td>mvnorm</td>
<td>mean</td>
<td>r</td>
<td>sigma</td>
</tr>
<tr>
<td>Log Normal</td>
<td>lnorm</td>
<td>log mean</td>
<td>log sigma</td>
<td>income or reaction time</td>
</tr>
<tr>
<td>Uniform</td>
<td>unif</td>
<td>min</td>
<td>max</td>
<td>rectangular distributions</td>
</tr>
<tr>
<td>Binomial</td>
<td>binom</td>
<td>size</td>
<td>prob</td>
<td>Bernoulli trials (e.g. coin flips)</td>
</tr>
<tr>
<td>Student's t</td>
<td>t</td>
<td>df</td>
<td>nc</td>
<td>Finding significance of a t-test</td>
</tr>
<tr>
<td>Multivariate t</td>
<td>mvt</td>
<td>df</td>
<td>corr</td>
<td>nc</td>
</tr>
<tr>
<td>Fisher's F</td>
<td>f</td>
<td>df1</td>
<td>df2</td>
<td>Testing for significance of F test</td>
</tr>
<tr>
<td>$\chi^2$</td>
<td>chisq</td>
<td>df</td>
<td>nc</td>
<td>Testing for significance of $\chi^2$</td>
</tr>
<tr>
<td>Exponential</td>
<td>exp</td>
<td>rate</td>
<td></td>
<td>Exponential decay</td>
</tr>
<tr>
<td>Gamma</td>
<td>gamma</td>
<td>shape</td>
<td>rate</td>
<td>scale</td>
</tr>
<tr>
<td>Hypergeometric</td>
<td>hyper</td>
<td>m</td>
<td>n</td>
<td>k</td>
</tr>
<tr>
<td>Logistic</td>
<td>logis</td>
<td>location</td>
<td>scale</td>
<td>Item Response Theory</td>
</tr>
<tr>
<td>Poisson</td>
<td>pois</td>
<td>lambda</td>
<td></td>
<td>Count data</td>
</tr>
<tr>
<td>Weibull</td>
<td>weibull</td>
<td>shape</td>
<td>scale</td>
<td>Reaction time distributions</td>
</tr>
</tbody>
</table>
An example of using r, p, and q for a distributions

```r
set.seed(42) #set the random seed to get the same sequence
x <- rnorm(5) #find 5 randomly distributed normals
round(x,2) #show them, rounded to 2 decimals
round(pnorm(x),2) #show their probabilities to 2 decimals
round(qnorm(pnorm(x)),2) #find the quantiles of the normal
```

Produces this output

```r
> set.seed(42) #set the random seed to get the same sequence
> x <- rnorm(5) #find 5 randomly distributed normals
> round(x,2) #show them, rounded to 2 decimals
[1] 1.37 -0.56 0.36 0.63 0.40
> round(pnorm(x),2) #show their probabilities to 2 decimals
[1] 0.91 0.29 0.64 0.74 0.66
> round(qnorm(pnorm(x)),2) #find the quantiles of the normal
[1] 1.37 -0.56 0.36 0.63 0.40
```
A very small list of the many data sets available

> `data()`

> `data(package="psych")`

> `data(Titanic)`
> `? Titanic`

> `data(cushny)`
> `? cushney`

> `data(UCBAdmissions)`
> `? UCBAdmissions`

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.

5. The UC Berkeley example of "sex discrimination" as a Simpson paradox.
We do this by using the `curve` function to which we pass the values of the `dnorm` function.

```r
curve(dnormal(x), -3, 3, ylab = "probability of x", main = "A normal curve")
```
R can draw more interesting distributions

The normal curve

Log normal

Chi Square distribution

Normal and t with 4 df
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.

```r
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='Probiabiity 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='Proobility of z or t',xlab='z or t',main='Normal and t with
curve(dt(x,4),add=TRUE)

op <- par(mfrow=c(1,1)) #back to a normal 1 x 1 graph
```

Basic R capabilities: Calculation, Statistical tables, Graphics
R can show current statistical concepts:

**Type I Errors:** It is not the power, it is the prior likelihood

dashed/dotted lines reflect alpha = .05, .01, .001 with power = 1

1. Extreme claims require extreme probabilities

2. Given that a finding is “significant”, what is the likelihood that it is a Type I error?

3. Depends upon the prior likelihood (the 'sexiness') of the claim.
A simple scatter plot using `plot` with Fisher’s Iris data set.

```r
plot(iris[1:2], xlab="Sepal.Length", ylab="Sepal.Width", main="Fisher Iris data")
```
A simple scatter plot using `plot` with some colors and shapes

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

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

```r
pairs.panels(iris[1:4], bg=c("red","yellow","blue") [iris$Species], pch=21, main="Fisher Iris data by Species")
```
A brief example of exploratory and confirmatory data analysis

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
A brief example of exploratory and confirmatory data analysis

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`

```r
> 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
```

<table>
<thead>
<tr>
<th>var</th>
<th>n</th>
<th>mean</th>
<th>sd</th>
<th>median</th>
<th>trimmed</th>
<th>mad</th>
<th>min</th>
<th>max</th>
<th>range</th>
<th>skew</th>
<th>kurtosis</th>
<th>se</th>
</tr>
</thead>
<tbody>
<tr>
<td>gender</td>
<td>1700</td>
<td>1.65</td>
<td>0.48</td>
<td>2</td>
<td>1.68</td>
<td>0.00</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>-0.61</td>
<td>-1.62</td>
<td>0.02</td>
</tr>
<tr>
<td>education</td>
<td>2700</td>
<td>3.16</td>
<td>1.43</td>
<td>3</td>
<td>3.31</td>
<td>1.48</td>
<td>0</td>
<td>5</td>
<td>5</td>
<td>-0.68</td>
<td>-0.06</td>
<td>0.05</td>
</tr>
<tr>
<td>age</td>
<td>3700</td>
<td>25.59</td>
<td>9.50</td>
<td>22</td>
<td>23.86</td>
<td>5.93</td>
<td>13</td>
<td>65</td>
<td>52</td>
<td>1.64</td>
<td>2.47</td>
<td>0.36</td>
</tr>
<tr>
<td>ACT</td>
<td>4700</td>
<td>28.55</td>
<td>4.82</td>
<td>29</td>
<td>28.84</td>
<td>4.45</td>
<td>3</td>
<td>36</td>
<td>33</td>
<td>-0.66</td>
<td>0.56</td>
<td>0.18</td>
</tr>
<tr>
<td>SATV</td>
<td>5700</td>
<td>612.23</td>
<td>112.90</td>
<td>620</td>
<td>619.45</td>
<td>118.61</td>
<td>200</td>
<td>800</td>
<td>600</td>
<td>-0.64</td>
<td>0.35</td>
<td>4.27</td>
</tr>
<tr>
<td>SATQ</td>
<td>6687</td>
<td>610.22</td>
<td>115.64</td>
<td>620</td>
<td>617.25</td>
<td>118.61</td>
<td>200</td>
<td>800</td>
<td>600</td>
<td>-0.59</td>
<td>0.00</td>
<td>4.41</td>
</tr>
</tbody>
</table>
A brief example of exploratory and confirmatory data analysis.

**Graphic display of data using** `pairs.panels`

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

![Graph showing data visualization with `pairs.panels` function in R](image-url)
Clean up the data using `scrub`. Use `?scrub` for help on the parameters.

We noticed an outlier in the ACT data in the previous graph (you always graph your data, don’t you).
We also noticed that the minimum value for ACT was unlikely (of course, you always describe your data).
So we change any case below 4 on the ACT to be missing (NA).

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

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

<table>
<thead>
<tr>
<th></th>
<th>gender</th>
<th>education</th>
<th>age</th>
<th>ACT</th>
<th>SATV</th>
<th>SATQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>gender</td>
<td>1.00</td>
<td>0.09</td>
<td>-0.02</td>
<td>-0.05</td>
<td>-0.02</td>
<td>-0.17</td>
</tr>
<tr>
<td>education</td>
<td>0.09</td>
<td>1.00</td>
<td>0.55</td>
<td>0.15</td>
<td>0.05</td>
<td>0.03</td>
</tr>
<tr>
<td>age</td>
<td>-0.02</td>
<td>0.55</td>
<td>1.00</td>
<td>0.11</td>
<td>-0.04</td>
<td>-0.03</td>
</tr>
<tr>
<td>ACT</td>
<td>-0.05</td>
<td>0.15</td>
<td>0.11</td>
<td>1.00</td>
<td>0.55</td>
<td>0.59</td>
</tr>
<tr>
<td>SATV</td>
<td>-0.02</td>
<td>0.05</td>
<td>-0.04</td>
<td>0.55</td>
<td>1.00</td>
<td>0.64</td>
</tr>
<tr>
<td>SATQ</td>
<td>-0.17</td>
<td>0.03</td>
<td>-0.03</td>
<td>0.59</td>
<td>0.64</td>
<td>1.00</td>
</tr>
</tbody>
</table>
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)

gender 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
A brief example of exploratory and confirmatory data analysis

**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** Normal theory

\[
> \text{corr.test(cleaned)}
\]

Call: corr.test(x = sat.act)

Correlation matrix

<table>
<thead>
<tr>
<th></th>
<th>gender</th>
<th>education</th>
<th>age</th>
<th>ACT</th>
<th>SATV</th>
<th>SATQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>gender</td>
<td>1.00</td>
<td>0.09</td>
<td>-0.02</td>
<td>-0.04</td>
<td>-0.02</td>
<td>-0.17</td>
</tr>
<tr>
<td>education</td>
<td>0.09</td>
<td>1.00</td>
<td>0.55</td>
<td>0.15</td>
<td>0.05</td>
<td>0.03</td>
</tr>
<tr>
<td>age</td>
<td>-0.02</td>
<td>0.55</td>
<td>1.00</td>
<td>0.11</td>
<td>-0.04</td>
<td>-0.03</td>
</tr>
<tr>
<td>ACT</td>
<td>-0.04</td>
<td>0.15</td>
<td>0.11</td>
<td>1.00</td>
<td>0.56</td>
<td>0.59</td>
</tr>
<tr>
<td>SATV</td>
<td>-0.02</td>
<td>0.05</td>
<td>-0.04</td>
<td>0.56</td>
<td>1.00</td>
<td>0.64</td>
</tr>
<tr>
<td>SATQ</td>
<td>-0.17</td>
<td>0.03</td>
<td>-0.03</td>
<td>0.59</td>
<td>0.64</td>
<td>1.00</td>
</tr>
</tbody>
</table>

Sample Size

<table>
<thead>
<tr>
<th></th>
<th>gender</th>
<th>education</th>
<th>age</th>
<th>ACT</th>
<th>SATV</th>
<th>SATQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>gender</td>
<td>700</td>
<td>700</td>
<td>700</td>
<td>700</td>
<td>700</td>
<td>687</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>SATQ</td>
<td>687</td>
<td>687</td>
<td>687</td>
<td>687</td>
<td>687</td>
<td>687</td>
</tr>
</tbody>
</table>

Probability values (Entries above the diagonal are adjusted for multiple tests.)

<table>
<thead>
<tr>
<th></th>
<th>gender</th>
<th>education</th>
<th>age</th>
<th>ACT</th>
<th>SATV</th>
<th>SATQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>gender</td>
<td>0.00</td>
<td>0.17</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>0</td>
</tr>
<tr>
<td>education</td>
<td>0.02</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>1</td>
</tr>
<tr>
<td>age</td>
<td>0.58</td>
<td>0.00</td>
<td>0.00</td>
<td>0.03</td>
<td>1.00</td>
<td>1</td>
</tr>
<tr>
<td>ACT</td>
<td>0.33</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0</td>
</tr>
<tr>
<td>SATV</td>
<td>0.62</td>
<td>0.22</td>
<td>0.26</td>
<td>0.00</td>
<td>0.00</td>
<td>0</td>
</tr>
<tr>
<td>SATQ</td>
<td>0.19</td>
<td>0.36</td>
<td>0.27</td>
<td>0.00</td>
<td>0.00</td>
<td>0</td>
</tr>
</tbody>
</table>
The SAT.ACT correlations. Confidence values from resampling

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

Heat map of sat.act correlations

- SATQ    SATV   ACT    age    education    gender
- SATQ   1
- SATV 0.09 1
- ACT -0.02 0.55 1
- age -0.05 0.11 0.15 1
- education -0.02 0.11 0.55 0.59 1
- gender -0.17 0.03 -0.04 0.55 0.64 1

The heatmap visualizes the correlation matrix with confidence intervals, showing the relationships between SATQ, SATV, and ACT scores across different demographic variables such as age, education, and gender, with the colors ranging from red to blue indicating the strength and direction of the correlations.
A brief example of exploratory and confirmatory data analysis

The SAT.ACT bootstrapped confidence intervals of correlation

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

certainty values of the sat.act data
Are education and gender independent? \( \chi^2 \) Test of association

\[
T \leftarrow \text{with(my.data,table(gender,education))}
\]

> T

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>gender</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>27</td>
<td>20</td>
<td>23</td>
<td>80</td>
<td>51</td>
<td>46</td>
</tr>
<tr>
<td>2</td>
<td>30</td>
<td>25</td>
<td>21</td>
<td>195</td>
<td>87</td>
<td>95</td>
</tr>
</tbody>
</table>

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

> chisq.test(T)

Pearson's Chi-squared test

data:  T
X-squared = 16.0851, df = 5, p-value = 0.006605
Multiple regression and the general linear model

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

```r
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)               180.8735   23.4102   7.726   3.96e-14 ***
education                 1.2404    2.3236   0.534    0.59363
gender                   20.6927    6.9965   2.958    0.00321 **
SATQ                     0.6449    0.0289  22.309  < 2.2e-16 ***
```

Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 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
Multiple regression modeling and graphics

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.

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

<table>
<thead>
<tr>
<th></th>
<th>n</th>
<th>mean</th>
<th>sd</th>
<th>median</th>
<th>trimmed</th>
<th>mad</th>
<th>min</th>
<th>max</th>
<th>range</th>
<th>skew</th>
<th>kurtosis</th>
<th>se</th>
</tr>
</thead>
<tbody>
<tr>
<td>gender</td>
<td>1</td>
<td>700</td>
<td>0</td>
<td>0.48</td>
<td>0.35</td>
<td>0.04</td>
<td>0.00</td>
<td>-0.65</td>
<td>0.35</td>
<td>-0.61</td>
<td>-1.62</td>
<td>0.02</td>
</tr>
<tr>
<td>education</td>
<td>2</td>
<td>700</td>
<td>0</td>
<td>1.43</td>
<td>-0.16</td>
<td>0.14</td>
<td>1.48</td>
<td>-3.16</td>
<td>1.84</td>
<td>-0.68</td>
<td>-0.06</td>
<td>0.05</td>
</tr>
<tr>
<td>age</td>
<td>3</td>
<td>700</td>
<td>0</td>
<td>9.50</td>
<td>-3.59</td>
<td>-1.73</td>
<td>5.93</td>
<td>-12.59</td>
<td>52</td>
<td>1.64</td>
<td>2.47</td>
<td>0.36</td>
</tr>
<tr>
<td>ACT</td>
<td>4</td>
<td>700</td>
<td>0</td>
<td>4.82</td>
<td>0.45</td>
<td>0.30</td>
<td>4.45</td>
<td>-25.55</td>
<td>7.45</td>
<td>-0.66</td>
<td>0.56</td>
<td>0.18</td>
</tr>
<tr>
<td>SATV</td>
<td>5</td>
<td>700</td>
<td>0</td>
<td>112.90</td>
<td>7.77</td>
<td>7.22</td>
<td>118.61</td>
<td>-412.23</td>
<td>600</td>
<td>-0.64</td>
<td>0.35</td>
<td>4.27</td>
</tr>
<tr>
<td>SATQ</td>
<td>6</td>
<td>687</td>
<td>0</td>
<td>115.64</td>
<td>9.78</td>
<td>7.04</td>
<td>118.61</td>
<td>-410.22</td>
<td>189.78</td>
<td>6</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

```r
> 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 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Significance codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Multiple regression modeling and graphics
Compare model 1 and model 2

Test the difference between the two linear models

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

Analysis of Variance Table

<table>
<thead>
<tr>
<th></th>
<th>Res.Df</th>
<th>RSS</th>
<th>Df</th>
<th>Sum of Sq</th>
<th>F</th>
<th>Pr(&gt;F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model 1: SATV ~ education + gender + SATQ</td>
<td>683</td>
<td>5079984</td>
<td>4</td>
<td>209742</td>
<td>7.3104</td>
<td>9.115e-06 ***</td>
</tr>
<tr>
<td>Model 2: SATV ~ education * gender * SATQ</td>
<td>679</td>
<td>4870243</td>
<td>209742</td>
<td>7.3104</td>
<td>9.115e-06 ***</td>
<td></td>
</tr>
</tbody>
</table>

---

Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 1
Show the regression lines by gender

First plot all the data.
Then add the regression lines.
Then put a title on the whole thing.

```r
> 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

Verbal varies by Quant and education

Do this again, but for levels of education as the moderator.

```r
> 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")
```
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
   - Use either the `foreign`, `haven` or `rio` packages

5. Simulate it (using various simulation routines)

6. Model it using simulations (e.g., cta Revelle and Condon, 2015)
### Examples of built in data sets from the psych package

```r
> data(package="psych")
```

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ability</td>
<td>16 multiple choice IQ items (N=1525)</td>
</tr>
<tr>
<td>Bechtoldt</td>
<td>Seven data sets showing a bifactor solution.</td>
</tr>
<tr>
<td>Dwyer</td>
<td>8 cognitive variables used by Dwyer for an example.</td>
</tr>
<tr>
<td>Reise</td>
<td>Seven data sets showing a bifactor solution.</td>
</tr>
<tr>
<td>affect</td>
<td>Data sets of affect and arousal scores as a function of personality and movie conditions (JPSP-12)</td>
</tr>
<tr>
<td>income</td>
<td>US family income from US census 2008</td>
</tr>
<tr>
<td>bfi</td>
<td>25 Personality items representing 5 factors (N=2800)</td>
</tr>
<tr>
<td>blot</td>
<td>Bond’s Logical Operations Test - BLOT (N=150)</td>
</tr>
<tr>
<td>burt</td>
<td>11 emotional variables from Burt (1915)</td>
</tr>
<tr>
<td>cities</td>
<td>Distances between 11 US cities</td>
</tr>
<tr>
<td>epi.bfi</td>
<td>13 personality scales from the Eysenck Personality Inventory and Big 5 inventory</td>
</tr>
<tr>
<td>income</td>
<td>US family income from US census 2008</td>
</tr>
<tr>
<td>msq</td>
<td>75 mood items from the Motivational State Questionnaire for N=3896</td>
</tr>
<tr>
<td>neo</td>
<td>NEO correlation matrix from the NEOPI-R manual</td>
</tr>
<tr>
<td>sat.act</td>
<td>3 Measures of ability: SATV, SATQ, ACT (N=700)</td>
</tr>
<tr>
<td>Thurstone</td>
<td>Seven data sets showing a bifactor solution.</td>
</tr>
<tr>
<td>veg (vegetables)</td>
<td>Paired comparison of preferences for 9 vegetables</td>
</tr>
</tbody>
</table>
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

   ```r
   my.data <- read.clipboard() # assumes headers and tab or space delimit (e.g., from Excel)
   my.data <- read.clipboard.csv() # assumes headers and comma delimited
   my.data <- read.clipboard.tab() # assumes headers and tab delimited
   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
   (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. In particular, are headers provided for each column of input?
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` might not work on https files.)

2. Two examples of reading data

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

   ```r
   > dim(my.data ) #what are the dimensions of what we read?
   [1] 18 2
   > describe(my.data ) #do the data look right?
   var  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  -1.12  0.18
   Alertness   2 18    27.67 6.82    27  27.50 8.15    17    41    24 0.25  -0.68  1.61
   ```
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

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
However, some might want to Import SAS or SPSS files

There are several different packages that make importing SPSS, SAS, Systat, etc. files easy to do.

**foreign** Read data stored by Minitab, S, SAS, SPSS, Stata, Systat, Weka, dBase. Comes installed with R. Somewhat complicated syntax.

**haven** Reads/writes SPSS and Stata files. Handles SPSS labels nicely (keeps the item labels, but converts the data to factors).

**rio** A general purpose package that requires installation of many of the other packages used for data import. Easiest to use, but overkill if just reading in one type of file. Basically a front end to many import/export packages. It determines which package to use based upon the file name suffix (e.g., csv, txt, sav, ...)

4 steps: read, explore, test, graph
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. (The defaults lead to problems, make sure to specify that you want use.value.labels = FALSE, to.data.frame = TRUE)

```
read.spss(file, use.value.labels = FALSE, to.data.frame = TRUE,
          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? Should be FALSE
- **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 using foreign

> 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)

<table>
<thead>
<tr>
<th>USER</th>
<th>HAPPY</th>
<th>SOULMATE</th>
<th>ENJOYDEX</th>
<th>UPSET</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>&quot;001&quot;</td>
<td>4</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>&quot;003&quot;</td>
<td>6</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>&lt;NA&gt;</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>68</td>
<td>&quot;076&quot;</td>
<td>7</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>69</td>
<td>&quot;078&quot;</td>
<td>2</td>
<td>7</td>
<td>7</td>
</tr>
</tbody>
</table>

> describe(eli, skew=FALSE)

<table>
<thead>
<tr>
<th>var</th>
<th>n</th>
<th>mean</th>
<th>sd</th>
<th>median</th>
<th>trimmed</th>
<th>mad</th>
<th>min</th>
<th>max</th>
<th>range</th>
<th>se</th>
</tr>
</thead>
<tbody>
<tr>
<td>USER*</td>
<td>1</td>
<td>69</td>
<td>35.00</td>
<td>35.00</td>
<td>25.20</td>
<td>1</td>
<td>69</td>
<td>68</td>
<td>2</td>
<td>2.42</td>
</tr>
<tr>
<td>HAPPY</td>
<td>2</td>
<td>69</td>
<td>5.71</td>
<td>1.04</td>
<td>5.82</td>
<td>0</td>
<td>2</td>
<td>7</td>
<td>5</td>
<td>0.13</td>
</tr>
<tr>
<td>SOULMATE</td>
<td>3</td>
<td>69</td>
<td>5.09</td>
<td>1.80</td>
<td>5.32</td>
<td>1.48</td>
<td>1</td>
<td>7</td>
<td>6</td>
<td>0.22</td>
</tr>
<tr>
<td>ENJOYDEX</td>
<td>4</td>
<td>68</td>
<td>6.47</td>
<td>1.01</td>
<td>6.70</td>
<td>0.00</td>
<td>2</td>
<td>7</td>
<td>5</td>
<td>0.12</td>
</tr>
<tr>
<td>UPSET</td>
<td>5</td>
<td>69</td>
<td>0.41</td>
<td>0.49</td>
<td>0.39</td>
<td>0.00</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0.06</td>
</tr>
</tbody>
</table>

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
An example of reading from an SPSS file using rio

> library(rio)

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

> eli <- import(datafilename)  #note that it figures out what to do
> headTail(eli,2,2)  #The first and last 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

<table>
<thead>
<tr>
<th></th>
<th>n</th>
<th>mean</th>
<th>sd</th>
<th>median</th>
<th>trimmed</th>
<th>mad</th>
<th>min</th>
<th>max</th>
<th>range</th>
<th>se</th>
</tr>
</thead>
<tbody>
<tr>
<td>USER*</td>
<td>1</td>
<td>69</td>
<td>35</td>
<td>35.00</td>
<td>25.20</td>
<td>1</td>
<td>69</td>
<td>68</td>
<td>2</td>
<td>2.42</td>
</tr>
<tr>
<td>HAPPY</td>
<td>2</td>
<td>69</td>
<td>5.71</td>
<td>1.04</td>
<td>0.00</td>
<td>2</td>
<td>7</td>
<td>5</td>
<td>0</td>
<td>0.13</td>
</tr>
<tr>
<td>SOULMATE</td>
<td>3</td>
<td>69</td>
<td>5.09</td>
<td>1.80</td>
<td>1.48</td>
<td>1</td>
<td>7</td>
<td>6</td>
<td>5</td>
<td>0.22</td>
</tr>
<tr>
<td>ENJOYDEX</td>
<td>4</td>
<td>68</td>
<td>6.47</td>
<td>1.01</td>
<td>0.00</td>
<td>2</td>
<td>7</td>
<td>5</td>
<td>0</td>
<td>0.12</td>
</tr>
<tr>
<td>UPSET</td>
<td>5</td>
<td>69</td>
<td>0.41</td>
<td>0.49</td>
<td>0.39</td>
<td>0.00</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.06</td>
</tr>
</tbody>
</table>

1. Make the `rio` package active
2. Specify the name (and location) of the file to read
3. Import from a SPSS file
4. Show the top and bottom 2 cases
5. Describe it to make sure it is right
An example of reading from an SPSS file using haven

1. Make the `haven` package active
2. Specify the name (and location) of the file to read
3. Import from a SPSS file
4. Show the top 3 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.

```r
> headTail(epi.bfi)

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

epi.bfi has 231 cases from two personality measures.
Now find the descriptive statistics for this data set

```r
> describe(epi.bfi)

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

Boxplots of EPI scales

Use the `boxplot` function and select the first five variables.

```r
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`

```r
violinBy(my.data[1:5])
```
Plot the scatter plot matrix (SPLOM) of the first 5 variables using the `pairs.panels` function. Note that the plotting points overlap because of the polytomous nature of the data.

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

Use the `pairs.panels` function from `psych`
Plot the scatter plot matrix (SPLOM) of the first 5 variables using the `pairs.panels` function but with smaller pch and jittering the points in order to better show the distributions.

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

```r
pairs.panels(my.data[1:5], pch='.', jiggle=TRUE)
```
Find the correlations for this data set, round off to 2 decimal places.

Because we have some missing data, we use “pairwise complete” correlations. For the purists amongst us, it is irritating that the columns are not equally spaced.

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

This is just a wrapper for `round(cor(x, use='pairwise'), 2)` that has been prettied up with `lowerMat`.

```r
> lowerCor(my.data)

   epiE  epiS  epImp epili epiNr bfgagr bfcon bfext bfner bfoxn bdi    trtnx sttnx
epiE    1.00
epiS    0.85 1.00
epImp   0.80 0.43 1.00
epili   -0.22 -0.05 -0.24 1.00
epiNeur -0.18 -0.22 -0.07 -0.25 1.00
bfgagr  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
bfner   -0.09 -0.07 -0.09 -0.22 0.63 -0.04 0.04 0.04 1.00
bfoxn   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

```r
> corr.test(my.data)
Call: corr.test(x = my.data)
Correlation matrix
  epiE epiS epiImp epilie epiNeur bfagree bfcon bfext bfnur bfnur bfnur bfnur bfnur
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
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
Sample Size
  epiE epiS epiImp epilie epiNeur bfagree bfcon bfext bfnur bfnur bfnur bfnur bfnur
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 bfnur bfnur bfnur bfnur bfnur
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 0.00 0.03 0.03 1.00 0.08 0.02 1.00
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 1.00
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 0.00 1.00
bfcon 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
bfext 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
bfnur 0.15 0.30 0.18 0.00 0.00 0.00 0.50 0.50 0.57 0.00 0.00 0.00 0.00 0.00 1.00
bfnur 0.04 0.02 0.30 0.70 0.19 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 1.00
bdi 0.02 0.04 0.11 0.00 0.00 0.00 0.03 0.01 0.03 0.00 0.25 0.00 0.00 0.00 1.00
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 0.00 0.00
stateanx 0.05 0.07 0.18 0.02 0.00 0.00 0.00 0.04 0.02 0.00 0.52 0.00 0.00 0.00 0.00
```

Basic descriptive and inferential statistics

**t.test demonstration with Student’s data (from the sleep dataset)**

```
> 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 to 0
  95 percent confidence interval:
    -3.3654832  0.2054832
  sample estimates:
    mean in group 1 mean in group 2
      0.750          2.330
```

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 to 0
  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

Student's sleep 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.

```r
> 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`.

```r
datafilename="https://personality-project.org/r/datasets/R.appendix2.data"
data.ex2=read.https(datafilename,header=T) #read the data into a tabular format
head(data.ex2) #show the data
```

<table>
<thead>
<tr>
<th>Observation</th>
<th>Gender</th>
<th>Dosage</th>
<th>Alertness</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>m</td>
<td>a</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>m</td>
<td>a</td>
<td>12</td>
</tr>
<tr>
<td>3</td>
<td>m</td>
<td>a</td>
<td>13</td>
</tr>
<tr>
<td>4</td>
<td>m</td>
<td>a</td>
<td>12</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>f</td>
<td>b</td>
<td>12</td>
</tr>
<tr>
<td>15</td>
<td>f</td>
<td>b</td>
<td>18</td>
</tr>
<tr>
<td>16</td>
<td>f</td>
<td>b</td>
<td>22</td>
</tr>
</tbody>
</table>
Analysis of Variance

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

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

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

Df  Sum Sq Mean Sq F value  Pr(>F)
Gender    1    76.56  76.56  2.9518 0.1115
Dosage    1     5.06  5.06  0.1952 0.6665
Gender:Dosage 1     0.06  0.06  0.0024 0.9617
```
Show the results table

```r
> 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 16.50 16.50
```
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://jason-french.com/tutorials/repeatedmeasures.html
Analysis of variance within subjects

```r
> 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**

<table>
<thead>
<tr>
<th>Df</th>
<th>Sum Sq</th>
<th>Mean Sq</th>
<th>F value</th>
<th>Pr(&gt;F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gender</td>
<td>1</td>
<td>542.26</td>
<td>542.26</td>
<td>5.6853</td>
</tr>
<tr>
<td>Dosage</td>
<td>2</td>
<td>694.91</td>
<td>347.45</td>
<td>3.6429</td>
</tr>
<tr>
<td>Gender:Dosage</td>
<td>2</td>
<td>70.80</td>
<td>35.40</td>
<td>0.3711</td>
</tr>
<tr>
<td>Residuals</td>
<td>12</td>
<td>1144.56</td>
<td>95.38</td>
<td></td>
</tr>
</tbody>
</table>

Signif. codes: 0 *** 0.001 ** 0.01 * 0.05 . 0.1 1  

**Error: Subject:Task**

<table>
<thead>
<tr>
<th>Df</th>
<th>Sum Sq</th>
<th>Mean Sq</th>
<th>F value</th>
<th>Pr(&gt;F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task</td>
<td>1</td>
<td>96.333</td>
<td>96.333</td>
<td>39.8621</td>
</tr>
<tr>
<td>Task:Gender</td>
<td>1</td>
<td>1.333</td>
<td>1.333</td>
<td>0.5517</td>
</tr>
<tr>
<td>Task:Dosage</td>
<td>2</td>
<td>8.167</td>
<td>4.083</td>
<td>1.6897</td>
</tr>
<tr>
<td>Task:Gender:Dosage</td>
<td>2</td>
<td>3.167</td>
<td>1.583</td>
<td>0.6552</td>
</tr>
<tr>
<td>Residuals</td>
<td>12</td>
<td>29.000</td>
<td>2.417</td>
<td></td>
</tr>
</tbody>
</table>
Multiple regression

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

```r
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
-372.910  -49.082    2.300   53.678   251.928
```

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 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 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 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 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

<table>
<thead>
<tr>
<th>Res.Df</th>
<th>RSS</th>
<th>Df</th>
<th>Sum of Sq</th>
<th>F</th>
<th>Pr(&gt;F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>683</td>
<td>5079984</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>679</td>
<td>4870243</td>
<td>4</td>
<td>209742</td>
<td>7.3104</td>
</tr>
</tbody>
</table>

---

Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
Show the regression lines by gender

Verbal varies by Quant and gender

```r
> 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")
```
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 (?function) or ?? function

4. Vignettes for help on the entire package (available either as part of the help file, or as a web page supplement to the package).
The basic types of data structures

1. **Scalars (characters, integers, reals, complex)**
   ```r
   > A <- 1  #Assign the value 1 to the object A
   > B <- 2  #Assign the value 2 to the object B
   ```

2. **Vectors (of scalars, all of one type) have length**
   ```r
   > C <- month.name[1:5]  #Assign the names of the first 5 months to C
   > D <- 12:24  #assign the numbers 12 to 24 to D
   > length(D)  #how many numbers are in D?
   [1] 13
   ```

3. **Matrices (all of one type) have dimensions**
   ```r
   > E <- matrix(1:20, ncol = 4)
   > dim(E)  #number of rows and columns of E
   [1] 5 4
   ```
Show values by entering the variable name

> A  #what is the value of A?
[1] 1

> B  #and of B?
[1] 2

> C  #and C
[1] "January" "February" "March" "April" "May"

> D

> E

[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.

   ```r
   > E.df <- data.frame(names = C, values = c(31, 28, 31, 30, 31))
   > dim(E.df)
   [1] 5 2
   ```

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

   ```r
   > F <- list(first = A, a.vector = C, a.matrix = E)
   > length(F)
   [1] 3
   ```
Show values by entering the variable name

```r
> 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,]    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"
   > 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`

```r
> 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](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](http://Rpad.org) for a list of important commands, [http://personality-project.org/r](http://personality-project.org/r)) for a tutorial for psychologists.
Useful functions

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

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mean</td>
<td>(x) standard deviation</td>
</tr>
<tr>
<td>is.na</td>
<td>(x) also is.null(x), is...</td>
</tr>
<tr>
<td>na.omit</td>
<td>(x) ignore missing data</td>
</tr>
<tr>
<td>sum</td>
<td>(x)</td>
</tr>
<tr>
<td>rowSums</td>
<td>(x) see also colSums(x)</td>
</tr>
<tr>
<td>min</td>
<td>(x)</td>
</tr>
<tr>
<td>max</td>
<td>(x)</td>
</tr>
<tr>
<td>range</td>
<td>(x)</td>
</tr>
<tr>
<td>table</td>
<td>(x)</td>
</tr>
<tr>
<td>summary</td>
<td>(x) depends upon x</td>
</tr>
<tr>
<td>sd</td>
<td>(x)</td>
</tr>
<tr>
<td>cor</td>
<td>(x)</td>
</tr>
<tr>
<td>cov</td>
<td>(x)</td>
</tr>
<tr>
<td>solve</td>
<td>(x) inverse of x</td>
</tr>
<tr>
<td>lm</td>
<td>(y~x) linear model</td>
</tr>
<tr>
<td>aov</td>
<td>(y~x) ANOVA</td>
</tr>
<tr>
<td>describe</td>
<td>(x) descriptive stats</td>
</tr>
<tr>
<td>describeBy</td>
<td>(x,y) descriptives by group</td>
</tr>
<tr>
<td>pairs.panels</td>
<td>(x) SPLOM</td>
</tr>
<tr>
<td>error.bars</td>
<td>(x) means + error bars</td>
</tr>
<tr>
<td>error.bars.by</td>
<td>(x) Error bars by groups</td>
</tr>
<tr>
<td>fa</td>
<td>(x,n) Factor analysis</td>
</tr>
<tr>
<td>principal</td>
<td>(x,n) Principal components</td>
</tr>
<tr>
<td>iclust</td>
<td>(x) Item cluster analysis</td>
</tr>
<tr>
<td>scoreItems</td>
<td>(x) score multiple scales</td>
</tr>
<tr>
<td>score.multiple.choice</td>
<td>(x) score multiple choice scales</td>
</tr>
<tr>
<td>alpha</td>
<td>(x) Cronbach’s alpha</td>
</tr>
<tr>
<td>omega</td>
<td>(x) MacDonald’s omega</td>
</tr>
<tr>
<td>irt.fa</td>
<td>(x) Item response theory through factor analysis</td>
</tr>
</tbody>
</table>
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
Classical Test Theory measures 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.multiple.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

Split half reliabilities of 16 ability measures

```r
sp <- splitHalf(ability, raw=TRUE, brute=TRUE)
hist(sp$raw, breaks=50)
```
Classical Test Theory measures of reliability

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

Reliability analysis
Call: alpha(x = ability)

<table>
<thead>
<tr>
<th>raw_alpha</th>
<th>std.alpha</th>
<th>G6(smc)</th>
<th>average_r</th>
<th>S/N</th>
<th>ase</th>
<th>mean</th>
<th>sd</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.83</td>
<td>0.83</td>
<td>0.84</td>
<td>0.23</td>
<td>4.9</td>
<td>0.0086</td>
<td>0.51</td>
<td>0.25</td>
</tr>
</tbody>
</table>

lower alpha upper 95% confidence boundaries
0.81 0.83 0.85

Reliability if an item is dropped:

<table>
<thead>
<tr>
<th>raw_alpha</th>
<th>std.alpha</th>
<th>G6(smc)</th>
<th>average_r</th>
<th>S/N</th>
<th>alpha</th>
<th>se</th>
</tr>
</thead>
<tbody>
<tr>
<td>reason.4</td>
<td>0.82</td>
<td>0.82</td>
<td>0.82</td>
<td>0.23</td>
<td>4.5</td>
<td>0.0093</td>
</tr>
<tr>
<td>reason.16</td>
<td>0.82</td>
<td>0.82</td>
<td>0.83</td>
<td>0.24</td>
<td>4.7</td>
<td>0.0091</td>
</tr>
<tr>
<td>rotate.6</td>
<td>0.82</td>
<td>0.82</td>
<td>0.82</td>
<td>0.23</td>
<td>4.5</td>
<td>0.0092</td>
</tr>
<tr>
<td>rotate.8</td>
<td>0.82</td>
<td>0.82</td>
<td>0.83</td>
<td>0.24</td>
<td>4.6</td>
<td>0.0091</td>
</tr>
</tbody>
</table>

Item statistics

<table>
<thead>
<tr>
<th>n</th>
<th>r</th>
<th>r.cor</th>
<th>r.drop</th>
<th>mean</th>
<th>sd</th>
</tr>
</thead>
<tbody>
<tr>
<td>reason.4</td>
<td>1442</td>
<td>0.58</td>
<td>0.54</td>
<td>0.50</td>
<td>0.68</td>
</tr>
<tr>
<td>reason.16</td>
<td>1463</td>
<td>0.50</td>
<td>0.44</td>
<td>0.41</td>
<td>0.73</td>
</tr>
</tbody>
</table>
Classical Test Theory measures of reliability

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

```r
> 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:

<table>
<thead>
<tr>
<th>Agree</th>
<th>Conscientious</th>
<th>Extraversion</th>
<th>Neuroticism</th>
<th>Openness</th>
</tr>
</thead>
<tbody>
<tr>
<td>alpha</td>
<td>0.7</td>
<td>0.72</td>
<td>0.76</td>
<td>0.81</td>
</tr>
</tbody>
</table>

Average item correlation:

<table>
<thead>
<tr>
<th>Agree</th>
<th>Conscientious</th>
<th>Extraversion</th>
<th>Neuroticism</th>
<th>Openness</th>
</tr>
</thead>
<tbody>
<tr>
<td>average.r</td>
<td>0.32</td>
<td>0.34</td>
<td>0.39</td>
<td>0.46</td>
</tr>
</tbody>
</table>

Guttman 6* reliability:

<table>
<thead>
<tr>
<th>Agree</th>
<th>Conscientious</th>
<th>Extraversion</th>
<th>Neuroticism</th>
<th>Openness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lambda.6</td>
<td>0.7</td>
<td>0.72</td>
<td>0.76</td>
<td>0.81</td>
</tr>
</tbody>
</table>

Scale intercorrelations corrected for attenuation
raw correlations below the diagonal, alpha on the diagonal corrected correlations above the diagonal:

<table>
<thead>
<tr>
<th>Agree</th>
<th>Conscientious</th>
<th>Extraversion</th>
<th>Neuroticism</th>
<th>Openness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agree</td>
<td>0.70</td>
<td>0.36</td>
<td>0.63</td>
<td>-0.245</td>
</tr>
<tr>
<td>Conscientious</td>
<td>0.26</td>
<td>0.72</td>
<td>0.35</td>
<td>-0.305</td>
</tr>
<tr>
<td>Extraversion</td>
<td>0.46</td>
<td>0.26</td>
<td>0.76</td>
<td>-0.284</td>
</tr>
<tr>
<td>Neuroticism</td>
<td>-0.18</td>
<td>-0.23</td>
<td>-0.22</td>
<td>0.812</td>
</tr>
<tr>
<td>Openness</td>
<td>0.15</td>
<td>0.19</td>
<td>0.22</td>
<td>-0.086</td>
</tr>
</tbody>
</table>
score.items output, continued

Item by scale correlations:
corrected for item overlap and scale reliability

<table>
<thead>
<tr>
<th>Agree</th>
<th>Conscientious</th>
<th>Extraversion</th>
<th>Neuroticism</th>
<th>Openness</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>-0.40</td>
<td>-0.06</td>
<td>-0.11</td>
<td>0.14</td>
</tr>
<tr>
<td>A2</td>
<td>0.67</td>
<td>0.23</td>
<td>0.40</td>
<td>-0.07</td>
</tr>
<tr>
<td>A3</td>
<td>0.70</td>
<td>0.22</td>
<td>0.48</td>
<td>-0.11</td>
</tr>
<tr>
<td>A4</td>
<td>0.49</td>
<td>0.29</td>
<td>0.30</td>
<td>-0.14</td>
</tr>
<tr>
<td>A5</td>
<td>0.62</td>
<td>0.23</td>
<td>0.55</td>
<td>-0.23</td>
</tr>
<tr>
<td>C1</td>
<td>0.13</td>
<td>0.53</td>
<td>0.19</td>
<td>-0.08</td>
</tr>
<tr>
<td>C2</td>
<td>0.21</td>
<td>0.61</td>
<td>0.17</td>
<td>0.00</td>
</tr>
<tr>
<td>C3</td>
<td>0.21</td>
<td>0.54</td>
<td>0.14</td>
<td>-0.09</td>
</tr>
<tr>
<td>C4</td>
<td>-0.24</td>
<td>-0.66</td>
<td>-0.23</td>
<td>0.31</td>
</tr>
<tr>
<td>C5</td>
<td>-0.26</td>
<td>-0.59</td>
<td>-0.29</td>
<td>0.36</td>
</tr>
<tr>
<td>E1</td>
<td>-0.30</td>
<td>-0.06</td>
<td>-0.59</td>
<td>0.11</td>
</tr>
<tr>
<td>E2</td>
<td>-0.39</td>
<td>-0.25</td>
<td>-0.70</td>
<td>0.34</td>
</tr>
<tr>
<td>E3</td>
<td>0.44</td>
<td>0.20</td>
<td>0.60</td>
<td>-0.10</td>
</tr>
<tr>
<td>E4</td>
<td>0.51</td>
<td>0.23</td>
<td>0.68</td>
<td>-0.22</td>
</tr>
<tr>
<td>E5</td>
<td>0.34</td>
<td>0.40</td>
<td>0.55</td>
<td>-0.10</td>
</tr>
<tr>
<td>N1</td>
<td>-0.22</td>
<td>-0.21</td>
<td>-0.11</td>
<td>0.76</td>
</tr>
<tr>
<td>N2</td>
<td>-0.22</td>
<td>-0.19</td>
<td>-0.12</td>
<td>0.74</td>
</tr>
<tr>
<td>N3</td>
<td>-0.14</td>
<td>-0.20</td>
<td>-0.14</td>
<td>0.74</td>
</tr>
<tr>
<td>N4</td>
<td>-0.22</td>
<td>-0.30</td>
<td>-0.39</td>
<td>0.62</td>
</tr>
<tr>
<td>N5</td>
<td>-0.04</td>
<td>-0.14</td>
<td>-0.19</td>
<td>0.55</td>
</tr>
<tr>
<td>O1</td>
<td>0.16</td>
<td>0.20</td>
<td>0.31</td>
<td>-0.09</td>
</tr>
<tr>
<td>O2</td>
<td>-0.01</td>
<td>-0.18</td>
<td>-0.07</td>
<td>0.19</td>
</tr>
<tr>
<td>O3</td>
<td>0.26</td>
<td>0.20</td>
<td>0.42</td>
<td>-0.07</td>
</tr>
<tr>
<td>O4</td>
<td>0.06</td>
<td>-0.02</td>
<td>-0.10</td>
<td>0.21</td>
</tr>
<tr>
<td>O5</td>
<td>-0.09</td>
<td>-0.14</td>
<td>-0.11</td>
<td>0.11</td>
</tr>
<tr>
<td>gender</td>
<td>0.25</td>
<td>0.11</td>
<td>0.12</td>
<td>0.14</td>
</tr>
<tr>
<td>education</td>
<td>0.06</td>
<td>0.03</td>
<td>0.01</td>
<td>-0.06</td>
</tr>
<tr>
<td>age</td>
<td>0.22</td>
<td>0.14</td>
<td>0.07</td>
<td>-0.13</td>
</tr>
</tbody>
</table>
Classical Test Theory measures of reliability

Correlations of composite scores based upon item correlations

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

**Correlations of composite scales**

![Correlation Matrix]

- **Agree**
  - 1
  - 0.25
  - 0.47
  - -0.18
  - 0.16

- **Conscientious**
  - 0.25
  - 1
  - 0.27
  - -0.22
  - 0.2

- **Extraversion**
  - 0.47
  - 0.27
  - 1
  - -0.22
  - 0.24

- **Neuroticism**
  - -0.18
  - -0.22
  - -0.22
  - 1
  - -0.07

- **Openness**
  - 0.16
  - 0.2
  - 0.24
  - -0.07
  - 1
**Upper and Lower bounds of Correlations of composite scores based upon item correlations and bootstrap resampling**

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

### Upper and lower bounds of Big 5 correlations

<table>
<thead>
<tr>
<th></th>
<th>Agree</th>
<th>Conscientious</th>
<th>Extraversion</th>
<th>Neuroticism</th>
<th>Openness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agree</td>
<td>1</td>
<td>0.29</td>
<td>0.51</td>
<td>-0.22</td>
<td>0.2</td>
</tr>
<tr>
<td>Conscientious</td>
<td>0.21</td>
<td>1</td>
<td>0.31</td>
<td>-0.26</td>
<td>0.24</td>
</tr>
<tr>
<td>Extraversion</td>
<td>0.44</td>
<td>0.22</td>
<td>1</td>
<td>-0.25</td>
<td>0.27</td>
</tr>
<tr>
<td>Neuroticism</td>
<td>-0.14</td>
<td>-0.18</td>
<td>-0.18</td>
<td>1</td>
<td>-0.12</td>
</tr>
<tr>
<td>Openness</td>
<td>0.13</td>
<td>0.16</td>
<td>0.2</td>
<td>-0.03</td>
<td>1</td>
</tr>
</tbody>
</table>
Factor analysis of Thurstone 9 variable problem

```r
> fa(Thurstone, nfactors=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
```

```r
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
```
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

<table>
<thead>
<tr>
<th>MR1</th>
<th>MR2</th>
<th>MR3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.96</td>
<td>0.92</td>
<td>0.90</td>
</tr>
</tbody>
</table>

Correlation of scores with factors
Multiple R square of scores with factors
Minimum correlation of possible factor scores
> f3 <- fa(Thurstone,3,n.obs=213,n.iter=20)  #to do bootstrapping

Coefficients and bootstrapped confidence intervals

<table>
<thead>
<tr>
<th>Sentences</th>
<th>low</th>
<th>MR1</th>
<th>upper</th>
<th>low</th>
<th>MR2</th>
<th>upper</th>
<th>low</th>
<th>MR3</th>
<th>upper</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sentences</td>
<td>0.77</td>
<td>0.91</td>
<td>0.96</td>
<td>-0.12</td>
<td>-0.04</td>
<td>0.07</td>
<td>-0.03</td>
<td>0.04</td>
<td>0.14</td>
</tr>
<tr>
<td>Vocabulary</td>
<td>0.85</td>
<td>0.89</td>
<td>0.95</td>
<td>-0.01</td>
<td>0.06</td>
<td>0.10</td>
<td>-0.12</td>
<td>-0.03</td>
<td>0.04</td>
</tr>
<tr>
<td>Sent.Completion</td>
<td>0.73</td>
<td>0.83</td>
<td>0.87</td>
<td>-0.04</td>
<td>0.04</td>
<td>0.13</td>
<td>-0.08</td>
<td>0.00</td>
<td>0.12</td>
</tr>
<tr>
<td>First.Letters</td>
<td>-0.06</td>
<td>0.00</td>
<td>0.10</td>
<td>0.68</td>
<td>0.86</td>
<td>0.93</td>
<td>-0.13</td>
<td>0.00</td>
<td>0.13</td>
</tr>
<tr>
<td>4.Letter.Words</td>
<td>-0.14</td>
<td>-0.01</td>
<td>0.07</td>
<td>0.58</td>
<td>0.74</td>
<td>0.86</td>
<td>0.01</td>
<td>0.10</td>
<td>0.25</td>
</tr>
<tr>
<td>Suffixes</td>
<td>0.07</td>
<td>0.18</td>
<td>0.27</td>
<td>0.46</td>
<td>0.63</td>
<td>0.76</td>
<td>-0.20</td>
<td>-0.08</td>
<td>0.06</td>
</tr>
<tr>
<td>Letter.Series</td>
<td>-0.04</td>
<td>0.03</td>
<td>0.13</td>
<td>-0.10</td>
<td>-0.01</td>
<td>0.10</td>
<td>0.56</td>
<td>0.84</td>
<td>0.93</td>
</tr>
<tr>
<td>Pedigrees</td>
<td>0.25</td>
<td>0.37</td>
<td>0.46</td>
<td>-0.16</td>
<td>-0.05</td>
<td>0.08</td>
<td>0.27</td>
<td>0.47</td>
<td>0.66</td>
</tr>
<tr>
<td>Letter.Group</td>
<td>-0.16</td>
<td>-0.06</td>
<td>0.06</td>
<td>0.09</td>
<td>0.21</td>
<td>0.31</td>
<td>0.44</td>
<td>0.64</td>
<td>0.79</td>
</tr>
</tbody>
</table>

Interfactor correlations and bootstrapped confidence intervals

<table>
<thead>
<tr>
<th>lower estimate upper</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 0.40 0.59 0.64</td>
</tr>
<tr>
<td>2 0.29 0.54 0.63</td>
</tr>
<tr>
<td>3 0.29 0.52 0.61</td>
</tr>
</tbody>
</table>
The simple factor structure

factor.diagram(f3) # show the diagram

Factor Analysis

```
    Sentences
    |          0.9
    |          0.9
Sent.Completion
    |          0.8
    |          0.6
    MR1
    |          0.6
    | 0.5
First.Leters
    |          0.9
    |          0.7
4.Letter.Words
    |          0.6
    | 0.5
    MR2
    |          0.5
    | 0.5
Suffixes
    | 0.9
    | 0.6
    MR3
    | 0.8
    | 0.5
    Pedigrees
    | 0.6
    | 0.5
    Letter.Group
    | 0.6
    | 0.5
    Letter.Series
    | 0.8
    | 0.5
    MR1
```
Two ways of viewing the higher order structure

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

Omega

Hierarchical (multilevel) Structure

```
F1*: 0.6 0.6 0.2
F2*: 0.6 0.5 0.4
F3*: 0.6 0.5 0.3
```

```
F1: 0.9 0.9 0.4
F2: 0.9 0.7 0.6
F3: 0.8 0.6 0.5
```

```
g: 0.8 0.8 0.7
```
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 to do more conventional IRT include *ltm*, *eRm*, *mirt*, + others
Item Response Information curves for 16 ability items from ICAR

Item information from factor analysis
<table>
<thead>
<tr>
<th>What is R?</th>
<th>A brief example</th>
<th>Basic statistics and graphics</th>
<th>Basic R commands</th>
<th>Psychometrics</th>
<th>More Help</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Item Response Theory

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

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>file.choose()</code></td>
<td>( ) find a file</td>
</tr>
<tr>
<td><code>file.choose()</code></td>
<td>(new=TRUE) create a new file</td>
</tr>
<tr>
<td><code>read.table()</code></td>
<td>(filename)</td>
</tr>
<tr>
<td><code>read.csv()</code></td>
<td>(filename) reads a comma separated file</td>
</tr>
<tr>
<td><code>read.delim()</code></td>
<td>(filename) reads a tab delimited file</td>
</tr>
<tr>
<td><code>c()</code></td>
<td>( ... ) combine arguments</td>
</tr>
<tr>
<td><code>from:to</code></td>
<td>e.g., 4:8</td>
</tr>
<tr>
<td><code>seq()</code></td>
<td>(from,to, by)</td>
</tr>
<tr>
<td><code>rep()</code></td>
<td>(x,times) repeat x</td>
</tr>
<tr>
<td><code>gl()</code></td>
<td>(n,k,...) generate factor levels</td>
</tr>
<tr>
<td><code>matrix()</code></td>
<td>(x,nrow=,ncol= ) create a matrix</td>
</tr>
<tr>
<td><code>data.frame()</code></td>
<td>( ... ) create a data frame</td>
</tr>
<tr>
<td><code>dim()</code></td>
<td>(x) dimensions of x</td>
</tr>
<tr>
<td><code>str()</code></td>
<td>(x) Structure of an object</td>
</tr>
<tr>
<td><code>list()</code></td>
<td>( ... ) create a list</td>
</tr>
<tr>
<td><code>colnames()</code></td>
<td>(x) set or find column names</td>
</tr>
<tr>
<td><code>rownames()</code></td>
<td>(x) set or find row names</td>
</tr>
<tr>
<td><code>ncol(x), nrow(z)</code></td>
<td>number of row, columns</td>
</tr>
<tr>
<td><code>rbind()</code></td>
<td>( ... ) combine by rows</td>
</tr>
<tr>
<td><code>cbind()</code></td>
<td>( ... ) combine by columns</td>
</tr>
<tr>
<td><code>is.na()</code></td>
<td>(x) also is.null(x), is...</td>
</tr>
<tr>
<td><code>na.omit()</code></td>
<td>(x) ignore missing data</td>
</tr>
<tr>
<td><code>table()</code></td>
<td>(x)</td>
</tr>
<tr>
<td><code>merge()</code></td>
<td>(x,y)</td>
</tr>
<tr>
<td><code>apply()</code></td>
<td>(x,rc,FUNCTION)</td>
</tr>
<tr>
<td><code>ls()</code></td>
<td>() show workspace</td>
</tr>
<tr>
<td><code>rm()</code></td>
<td>() remove variables from workspace</td>
</tr>
</tbody>
</table>
# More useful statistical functions, Use ? for details

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>mean</code></td>
<td>(x) standard deviation</td>
</tr>
<tr>
<td><code>is.na</code></td>
<td>(x) also is.null(x), is...</td>
</tr>
<tr>
<td><code>na.omit</code></td>
<td>(x) ignore missing data</td>
</tr>
<tr>
<td><code>sum</code></td>
<td>(x)</td>
</tr>
<tr>
<td><code>rowSums</code></td>
<td>(x) see also colSums(x)</td>
</tr>
<tr>
<td><code>min</code></td>
<td>(x)</td>
</tr>
<tr>
<td><code>max</code></td>
<td>(x)</td>
</tr>
<tr>
<td><code>range</code></td>
<td>(x)</td>
</tr>
<tr>
<td><code>table</code></td>
<td>(x)</td>
</tr>
<tr>
<td><code>summary</code></td>
<td>(x) depends upon x</td>
</tr>
<tr>
<td><code>sd</code></td>
<td>(x) standard deviation</td>
</tr>
<tr>
<td><code>cor</code></td>
<td>(x) correlation</td>
</tr>
<tr>
<td><code>cov</code></td>
<td>(x) covariance</td>
</tr>
<tr>
<td><code>solve</code></td>
<td>(x) inverse of x</td>
</tr>
<tr>
<td><code>lm</code></td>
<td>(y~x) linear model</td>
</tr>
<tr>
<td><code>aov</code></td>
<td>(y~x) ANOVA</td>
</tr>
</tbody>
</table>

## Selected functions from `psych` package

- `describe` (x) descriptive stats
- `describeBy` (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
- `scoreItems` (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 help

1. An introduction to R as HTML, PDF or EPUB from http://cran.r-project.org/manuals.html (many different links on this page)
2. FAQ General and then Mac and PC specific
4. Various “cheat sheets” from RStudio http://www.rstudio.com/resources/cheatsheets/
5. Using R for psychology http://personality-project.org/r/
6. Package vignettes (e.g., http://personality-project.org/r/psych/vignettes/overview.pdf)
7. R listserve, StackOverflow, your students and colleagues