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

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

Partially supported by a grant from the National Science Foundation: SMA-1419324
Outline

Part I: What is R, where did it come from, why use it
  • Installing R and adding packages: the building blocks of R

Part II: A brief introduction – an overview
  • R is just a fancy (very fancy) calculator
  • Descriptive data analysis
  • Some inferential analysis

Part III R is a powerful statistical system
  • Data entry (detail and practice)
  • Descriptive (again)
  • Inferential (t and F with more practice)
  • Regression
  • Basic R commands

Part IV: Psychometrics
  • Reliability and its discontents
  • EFA, CFA, SEM

Part V: Help and More Help
  • List of useful commands

Part VI: The psych package and more practice
Outline of Part I

What is R?
  Where did it come from, why use it?
  Misconceptions
Installing R on your computer and adding packages
Installing for your operating system
  R-Applications
Installing and using packages
  What are packages
  Installing packages
Building Blocks
  Objects
  R is a language
Package Dependencies
  Objects act on objects
  Package dependencies
-> Part II
Where did it come from, why use it?

**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
6. Some basic programming concepts in R
7. An overview of the *psych* package
8. Extended practice on your data sets
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.
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$REL$
  - MPlu$
  - Your favorite program
R: A way of thinking

- “R is the lingua franca of statistical research. Work in all other languages should be discouraged.”
- “This is R. There is no if. Only how.”
- “Overall, SAS is about 11 years behind R and S-Plus in statistical capabilities (last year it was about 10 years behind) in my estimation.”
- Q: My institute has been heavily dependent on SAS for the past while, and SAS is starting to charge us a very deep amount for license renewal.... The team is [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.
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 3.5.0
- 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)
## 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)
- **2001-2018**: 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.

<table>
<thead>
<tr>
<th>Year</th>
<th>Packages on CRAN</th>
<th>Bioinformatic Packages on BioConductor</th>
<th>GitHub Repositories</th>
</tr>
</thead>
<tbody>
<tr>
<td>2003-04-01</td>
<td>250</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2004-10-01</td>
<td>500</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2007-04-12</td>
<td>1,000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2009-10-04</td>
<td>2,000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2011-05-12</td>
<td>3,000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2014-05-16</td>
<td>5,547</td>
<td>824</td>
<td></td>
</tr>
<tr>
<td>2015-05-20</td>
<td>6,678</td>
<td>1024</td>
<td></td>
</tr>
<tr>
<td>2016-03-31</td>
<td>8,427</td>
<td>1,104</td>
<td></td>
</tr>
<tr>
<td>2017-05-21</td>
<td>10,677</td>
<td>1,383</td>
<td></td>
</tr>
<tr>
<td>2018-05-20</td>
<td>12,583</td>
<td>1,560</td>
<td></td>
</tr>
</tbody>
</table>
Rapid and consistent growth in packages contributed to R
Rapid and consistent growth in packages contributed to R

Active CRAN Packages

Log of active packages
Where did it come from, why use it?

**Popularity compared to other statistical packages**

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

1. discussion groups
2. blogs
4. Google Page rank
5. Number of downloads (see http://www.rpackages.io/packages or...
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,
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/ RMarkdown
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). Go to March, 2017 APS Observer article by Sara Weston and Debbie Yee.

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 Comprehensive R Archive Network

Download and Install R

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

- [Download R for Linux](#)
- [Download R for (Mac) OS X](#)
- [Download R for Windows](#)

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

Source Code for all Platforms

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

- The latest release (2018-04-23, Joy in Playing) [R-3.5.0.tar.gz](#), read [what's new](#) in the latest version.
- Sources of [R alpha and beta releases](#) (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are [available here](#). Please read about [new features and bug fixes](#) before filing corresponding feature requests or bug reports.
- Source code of older versions of R is [available here](#).
- Contributed extension [packages](#)

Questions About R

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

What are R and CRAN?

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

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

- The latest release (Friday 2017-04-21, You Stupid Darkness) R-3.4.0.tar.gz, read what's new in the latest version.
- Sources of R alpha and beta releases (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are available here. Please read about new features and bug fixes before filing corresponding feature requests or bug reports.
- Source code of older versions of R is available here.
- Contributed extension packages

Questions About R

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

What are R and CRAN?

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

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

R for Windows

Subdirectories:

- **base**
  - Binaries for base distribution. This is what you want to [install R for the first time](https://cran.r-project.org/bin/windows/base).

- **contrib**
  - Binaries of contributed CRAN packages (for R >= 2.13.x; managed by Uwe Ligges). There is also information on [third party software](https://cran.r-project.org/bin/windows/contrib) available for CRAN Windows services and corresponding environment and make variables.

- **old contrib**
  - Binaries of contributed CRAN packages for outdated versions of R (for R < 2.13.x; managed by Uwe Ligges).

- **Rtools**
  - Tools to build R and R packages. 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 Uwe Ligges directly in case of questions / suggestions related to Windows binaries.

You may also want to read the [R FAQ](https://cran.r-project.org/bin/windows/FAQ) and [R for Windows FAQ](https://cran.r-project.org/bin/windows/FAQ).

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-3.5.0 for Windows (32/64 bit)

Download R 3.5.0 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 matches the package distributed by CRAN, you can compare the md5sum of the .exe to the fingerprint on the master server. You will need a version of md5sum for windows: both graphical and command line versions are available.

Frequently asked questions

- Does R run under my version of Windows?
- 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.
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](https://cran.r-project.org/src/base/MacOSX10.6-Bin/). Releases for old Mac OS X systems (through Mac OS X 10.5) and PowerPC Macs can be found in the [old](https://cran.r-project.org/src/base/MacOSX10.6-Bin/) 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.

As of 2016/03/01 package binaries for R versions older than 2.12.0 are only available from the [CRAN archive](https://cran.r-project.org/archive/) so users of such versions should adjust the CRAN mirror setting accordingly.

**R 3.5.0 "Joy in Playing" released on 2018/04/24**

**Important:** since R 3.4.0 release we are now providing binaries for OS X 10.11 (El Capitan) and higher using non-Apple toolkit to provide support for OpenMP and C++17 standard features. To compile packages you may have to download tools from the [tools](https://cran.r-project.org/tools/) directory and read the corresponding note below.

Please check the MD5 checksum of the downloaded image to ensure that it has not been tampered with or corrupted during the mirroring process. For example type

```
md5 R-3.5.0.pkg
```

in the *Terminal* application to print the MD5 checksum for the R-3.5.0.pkg image. On Mac OS X 10.7 and later you can also validate the signature using

```
pkgutil --check-signature R-3.5.0.pkg
```

**Lastest release:**

**R 3.5.0** binary for OS X 10.11 (El Capitan) and higher, signed package. Contains R 3.5.0 framework, R.app GUI 1.70 in 64-bit for Intel Macs, Tcl/Tk 8.6.6 X11 libraries and Texinfo 5.2. The latter two components are optional and can be omitted when choosing "custom install", they are only needed if you want to...
Starting R on a PC

R version 3.5.0 (2018-04-23) -- "Joy in Playing"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (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.

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

R version 3.5.0 (2018-04-23) -- "Joy in Playing"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin15.6.0 (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 1.70 (7521) x86_64-apple-darwin15.6.0]
[Workspace restored from /Users/wr/.RData]
[History restored from /Users/wr/.Rapp.history]

Good morning Bill.
Are you ready to have fun?
Check the version number for R ≥ 3.5.0) and for psych (≥1.8.4)

```r
sessionInfo()
```

R version 3.5.0 (2018-04-23)
Platform: x86_64-apple-darwin15.6.0 (64-bit)
Running under: macOS High Sierra 10.13.4

Matrix products: default
BLAS: /Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRlapack.dylib

locale:

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

other attached packages:
[1] psych_1.8.4

loaded via a namespace (and not attached):
[1] compiler_3.5.0 tools_3.5.0 parallel_3.5.0 foreign_0.8-70 nlme_3.1-137 mnormt_1.5
[7] grid_3.5.0 lattice_0.20-35
other attached packages:
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)

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) (recommended by David)
   - allows for multiple cores for parallel processing

4. From the web
   - allows remote R (but R = 3.4 and psych = 1.7.8)
   - Rdocumentation is helpful for package search
R Studio is a useful “Integrated Development Environment” (IDE)
R Studio may be run on a remote server
https://rdrr.io allows to run on a remote server (but R = 3.4.0 and psych = 1.7.8)
R is extensible: The use of “packages”

1. More than 12,583 packages are available for at CRAN (and growing daily. It was 10,677 last year and 8,427 two years ago).

2. Can search all packages that do a particular operation by using the sos package (probably disappearing soon).
   - `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 2,318 matches from 180 packages and reports the top 400
   - `findFn("Item Response Theory")` # will return 394 matches in 93 packages
   - `findFn("INDSCAL ")` # will return 18 matches in 6 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)
What is R? Installing R for your OS

Packages

Building Blocks

Dependencies

Part II

What are packages

A small subset of very useful packages

- General use
  - core R
  - MASS
  - lattice
  - lme4 (core)
  - psych
  - Zelig

- Special use
  - ltm/eRm/mirt
  - sem
  - lavaan/OpenMx
  - GPArotation
  - mvtnorm
  - > 15,180 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)
  - Jennrich rotations
  - Multivariate distributions
  - Thousands of more packages on CRAN
  - Code on GitHub/ webpages/journal articles
Even more very useful packages (see also Computer World list)

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

- **Special use**
  - plyr & dplyr
  - data.table
  - knitr
  - sweave
  - ggplot2
  - > 12,500
  - + ?

- **General applications**
  - Development tools from GitHub
  - input from excel
  - input from SPSS, etc. (part of Core)
  - 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.
  - 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 3.5.0) (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
# just install a few packages
> install.packages("psych", dependencies=TRUE)
#which installs psych and its required packages

> install.packages("GPArotation")
> install.packages("mnormt")

#or
> install.packages("ctv")

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

- Just install one package (e.g., psych) You might have to choose a “mirror” site.
- as well as a few suggested packages that add functionality for factor rotation, multivariate normal distributions, etc.
- Install the task view installer package.
- Make it active
- If you want to install all the packages in the “Psychometrics” task view.
Building Blocks
R is an object-oriented programming language.
What is R? Installing R
your OS Packages Building Blocks Dependencies

Part II

R is a language

- Think of R like having a conversation with a specific person.
- They (R) have their own language, and you need to learn how to speak it.
- R is not very forgiving of mistakes, so pay attention to grammar and punctuation.

R is an **object-oriented** programming **language**.
What is an object? // Everything!
Single-value objects

- The most basic object contains a single value.
  
  4

- **Objects** can be numbers, strings, or logical values.
  
  4
  "female"
  TRUE

- We can save objects to our environment by assigning them to names.

- Note, although better style is to use the “get” command, you can also use the = (which means replace) command.

  happy <- 4       #read as happy gets 4, or
  happy is given the value of 4
  gender = "female"
  #read as gender is given the value of 4

- The *only way* to create or change an object is to assign it to a name.
**Single-value objects (aka in some languages as scalers)**

You can call objects using their name. Writing the name of an object will print its value to your console.

```r
happy
[1] 4
```

You can also use the name of an object as a substitute for its value.

```r
happy + 8
[1] 12
```
Vectors

A vector is an ordered set of values. Some of us would call this an ordered n-tuplet.

```
genders <- c("male","male","female","male","male","male")
emotions <- c(4, 7, happy, 7, 3, 8)
(We use the c for the concatenate operator).
```

Important rules:

- Order matters
- Each element included in the vector is of the same class (numerical, logical, character) which will be the class of the object

```
class(emotions)
[1] "numeric"
class(genders)
[1] "character"
```
Vectors and character strings

A **vector** is an ordered set of values. Some of us would call this an ordered n-tuplet.

```r
genders <- c("male","male","female","male", "male","male")
```

#this uses the c() function for concatenation, #and we need to delimitate each element with " " #alternatively, use the cs() function which takes Character Strings:

```r
genders <- cs(male,male,female,male,male,male)
```

#show this

genders

"male" "male" "female" "male" "male" "male"

emotions <- c(4, 7, happy, 7, 3, 8)

(We use the c for the *concatenate* operator) or the cs for the *character string* operator.
Order matters

Values in a vector are given a specific position and they will always be printed in that position.
(Hence the term ordered n-tuplet.)

emotions
[1] 4 7 4 7 3 8
You cannot mix numbers and strings and logical values in a single vector.

bad.vector = c(7, 9, "2")
#by typing the name, we are asking for its contents

bad.vector
the numerical values have become characters!
[1] "7" "9" "2"
Indexing vectors

**Indexing** is when you want to refer to specific parts or values of a vector.

Usually we index with square brackets.

You can refer to the **positions** of the values by their number.

```r
> emotions[1:3]
[1] 4 7 4
emotions[c(1,5)]    #concatenate 1 and 5
[1] 4 3
```
## Indexing vectors

**Indexing** is when you want to refer to specific parts or values of a vector.

Usually we index with square brackets.

You can refer to the **names** of the values by their number, if they have names.

```r
names(emotions) <- cs(Bill, David, Sara, Dan, Josh, Pat)

emotions["Sara"]
Sara
  4

emotions[cs(Bill, "David")]
Bill  David
  4    7
```
Indexing vectors

**Indexing** is when you want to refer to specific parts or values of a vector.

Usually we index with square brackets.

You can use **logical statements** to select values that meet certain criteria.

```r
emotions[emotions > 6]
```

David    Dan    Pat
7        7      8
Data frames

Data frames are lists of vectors which are related to one another (Think “spreadsheets”)

Features:

- Data frames have two dimension: rows and columns.
- (Usually) Columns represent variables.
- Every value in a column is the same class (numeric, character, etc)
- (Usually) Rows represent observations (people, mice, time points, etc).
- Values in rows can be different classes.
- The length of each vector must be the same.
Data frames

Because data frames are simply collections of vectors, you can create a data frame using vectors.

```r
data.example = data.frame(GENDER = genders,
                          + EMOTIONS = emotions)
data.example

  GENDER EMOTIONS
Bill   male   4
David  male   7
Sara   female 4
Dan    male   7
Josh   male   3
Pat    male   8```
Indexing data frames

We can use the same methods to select specific parts of data frames. The trick is data frames have two dimensions, not one. So we have to separate selecting rows from selecting columns.

Using numbers

Indexing a vector

```r
emotions[1:3]
[1] 4 7 4
```

Indexing a data frame

```r
data.example[1:3, 1:2]
   GENDER  EMOTIONS
  1   male     4
  2   male     7
  3 female     4
```
Indexing data frames

We can use the same methods to select specific parts of data frames. The trick is data frames have two dimensions, not one. So we have to separate selecting rows from selecting columns. But, we can specify that we want all of either a row or column by leaving it blank.

Indexing a data frame

> data.example[,1]  #give me the entire first column (as a vector)
[1] male male female male male male
Levels: female male

> data.example[,1,drop=FALSE] #give me the entire first column neatly
GENDER
Bill  male
David male
Sara  female
Dan   male
Josh  male
Pat   male

Try it (example 2)
Indexing data frames

We can use the same methods to select specific parts of data frames. The trick is data frames have two dimensions, not one. So we have to separate selecting rows from selecting columns.

Using names

Indexing a vector

`emotions[c("Josh","Pat")]
Josh Pat
  3  8`

Indexing a data frame

`data.example[, "GENDER"]`  # refer to the column by name

[1] male  male  female male  male  male
Levels: female male
Indexing data frames

We can use the same methods to select specific parts of data frames. The trick is data frames have two dimensions, not one. So we have to separate selecting rows from selecting columns.

Using logical statements

Indexing a vector

emotions[emotions < 7]

Bill Sara Josh
4 4 3

Indexing a data frame

data.example[data.example$GENDER == "female", ]

GENDER EMOTIONS
Sara female 4

We looked for equality by using the \( == \) operator (read as equals)
Indexing data frames

Data frames can also be indexed using the dollar sign $.

```r
data.example$EMOTIONS
[1] 4 7 4 7 3 8
```

This is read as “from the data frame called data.example, give me the variable called EMOTIONS.”
What is R?

Installing R

your OS

Packages

Building Blocks

Dependencies

-> Part II

Objects

Other kinds of objects
Lists
• Like vectors, but each element can be anything (value, vector,

data frame, another list)
• Output of analysis functions
• Can index using $
• Can index by name
• or, can index by [ ] for the name and content of the vector or
[[ ]] for the contents
Matrices
• Like data frames but every value has to be the same class
(character, numeric, logical)
• Useful for matrix algebra (i.e., lots of correlation and
regression analyses)
• Operations are faster on matrices than data frames (for large
data sets)
56 / 279


R is a language

R is an object-oriented programming language.

- Think of R like having a conversation with a specific person.
- They (R) have their own language, and you need to learn how to speak it.
- R is not very forgiving of mistakes, so pay attention to grammar and punctuation.
Translating R

catch(x = ball)
Nouns

Subject: R is the subject of every sentence.

Object: Objects are objects of the sentence!
Verbs

- **Functions** are the verbs of sentences.
  ```r
catch(x = ball)
  ```

- Functions are always followed by parentheses.
  ```r
  mean(data.example$EMOTIONS)
  [1] 5.5
  ```

- Functions can be nested. This is like a run-on sentence.
  ```r
  round(mean(emotions))
  [1] 6
  ```

*Find the mean of the values in emotion, then round that number.*
Adverbs and other modifiers

To be more specific or change the default way of doing something, specify arguments. These are like adverbs or clauses.

```r
catch(x = ball, how = "smoothly", where = "beach", with = friends)
```

Arguments might be character values, numbers, more data, anything. The documentation (help) for a function will tell you what arguments are available to be changed and what values they can or should take.

```r
help(t.test)  # or
?t.test
```

```r
t.test(x = groupA, y = groupB, paired = T, mu = 5)
```
Punctuation

- **Spaces** – you can put as many spaces as you want between words and symbols, but not within them.

  ```r
  mean( data) #ok
  me an( data) #not ok
  ```

- **Parentheses** – It’s easy to forget one or put one in the wrong place when nesting.

  ```r
  round(x = mean(data, digits = 3) #this is wrong
  round(x = mean(data), digits = 3) #this is ok
  ```

- **Captialization** – MATTERS

  ```r
  data != DATA != Data
  ```
The power of R: Objects can act upon objects

1. Every function returns an object.
   - This object can contain objects.
   - To see what is in an object use the `str` command to see the structure of an object.

2. Other functions can then act upon those objects to create objects
   - `mean()`, `sd()`, `median()` each return objects as values
   - `describe()` then packages those objects to return a general set of useful statistics.

3. It is this ability to use the output object from one function as the input to the next function that makes R so powerful.
Functions act upon the output of other functions

R code

```r
m <- mean(sat.act$SATV, na.rm=TRUE)
s <- sd(sat.act$SATQ, na.rm=TRUE)
md <- median(sat.act[,3], na.rm=TRUE)
describe(sat.act)  #combines these prior three and more
```

describe(sat.act)  #combines these prior three functions and more

<table>
<thead>
<tr>
<th>vars</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>1.65</td>
<td>0.48</td>
<td>2</td>
<td>1.68</td>
<td>0</td>
<td>1</td>
<td>1</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>3.16</td>
<td>1.43</td>
<td>3</td>
<td>3.31</td>
<td>0</td>
<td>5</td>
<td>5 -0.68</td>
<td>-0.07</td>
<td>0.05</td>
<td></td>
</tr>
<tr>
<td>age</td>
<td>3</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>1.64</td>
<td>2.42</td>
<td>0.36</td>
</tr>
<tr>
<td>ACT</td>
<td>4</td>
<td>700</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>-0.66</td>
<td>0.53</td>
<td>0.18</td>
</tr>
<tr>
<td>SATV</td>
<td>5</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>-0.64</td>
<td>0.33</td>
<td>4.27</td>
</tr>
<tr>
<td>SATQ</td>
<td>6</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>-0.59</td>
<td>-0.02</td>
<td>4.41</td>
</tr>
</tbody>
</table>
Objects act on objects

### Use `str` to see the structure of an object

**R code**

```r
# form a new object
d <- describe(sat.act)

# just the names of the objects
names(d)

# the detailed structure of those objects
str(d)

# the objects organized in a pretty way for display
d
```

```r
# form a new object
d <- describe(sat.act)

# just the names of the objects
names(d)

# the detailed structure of those objects
str(d)

# the objects organized in a pretty way for display
d
```
Several ways to see the contents of an object

R code

headTail(sat.act)  #shows the first and last n rows of the data frame
quickView(sat.act)  #opens a window showing the first and last n rows of the data frame (scrollable)
view(sat.act)  #opens a window to show all the rows and columns of the data frame (scrollable)

<table>
<thead>
<tr>
<th>gender</th>
<th>education</th>
<th>age</th>
<th>ACT</th>
<th>SATV</th>
<th>SATQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>29442</td>
<td>2</td>
<td>3</td>
<td>19</td>
<td>24</td>
<td>500</td>
</tr>
<tr>
<td>29457</td>
<td>2</td>
<td>3</td>
<td>23</td>
<td>35</td>
<td>600</td>
</tr>
<tr>
<td>29498</td>
<td>2</td>
<td>3</td>
<td>20</td>
<td>21</td>
<td>480</td>
</tr>
<tr>
<td>29503</td>
<td>1</td>
<td>4</td>
<td>27</td>
<td>26</td>
<td>550</td>
</tr>
<tr>
<td>39937</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>39951</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>39961</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>39985</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

...
Objects act on objects
Packages extend the power of R

1. Just as functions can take the output from another function, so can packages build upon other packages.

2. Core packages come with the R installation
   - *base*-R includes 1220 different functions and then also loads in 5-8 other core packages:
   - e.g., *stats* includes 447 functions (commands) that do most of those basic statistics not done by *base*;
   - *foreign* handles different input and output formats from “foreign” languages (e.g., SPSS)

3. The Comprehensive R Archive Network (CRAN) is the repository for the other 12,560 packages that people have contributed

4. Most of these packages depend, in turn, on other packages. They all depend upon core-R.
Dependencies of the psych package

Packages imported by psych

- mnormt
- lattice
- nlme
- foreign
Dependencies of the psych package including base R

Packages imported by psych (including Base R)

- mnormt
- parallel
- methods
- grDevices
- graphics
- lattice
- utils
- stats
- nlme
- foreign
- grid

Packages imported by psych (including Base R)
Packages can “suggest” other useful packages which in turn “require” other packages
psych, lavaan and sem require other useful packages
psych and lavaan suggest other useful packages
psych, lavaan and sem suggest other useful packages
Some packages require many others to be helpful wrapper packages (e.g. userfriendlyscience)
apaTables require many others to be a helpful wrapper
Questions?
Outline

Part I: What is R, where did it come from, why use it
   • Installing R and adding packages: the building blocks of R

Part II: A brief introduction – an overview
   • R is just a fancy (very fancy) calculator
   • Descriptive data analysis
   • Some inferential analysis

Part III R is a powerful statistical system
   • Data entry (detail and practice)
   • Descriptive (again)
   • Inferential (t and F with more practice)
   • Regression
   • Basic R commands

Part IV: Psychometrics
   • Reliability and its discontents
   • EFA, CFA, SEM

Part V: Help and More Help
   • List of useful commands

Part VI: The psych package and more practice
Outline of Part II

-> Part I: What is R

Basic R: A brief example
  Basic R capabilities: Calculation, Statistical tables
  Basic Graphics

A brief example of exploratory and confirmatory data analysis
  Data preparation, descriptive statistics, data cleaning,
  correlation plots: (Examples part ii)
  Inferential statistics

Multiple regression modeling and graphics

-> Part III: Basic statistics and graphics
Basic R commands – remember don’t enter the > 
R is just a fancy calculator. Add, subtract, sum, products, group

```r
> 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 $\chi^2$ distribution, the xyz distribution)

```r
> 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 \textit{d}, probability with \textit{p}, quantiles with \textit{q} and to generate random values with \textit{r}. (e.g., the normal distribution may be chosen by using \texttt{dnorm}, \texttt{pnorm}, \texttt{qnorm}, or \texttt{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>P 3</th>
<th>example application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>norm</td>
<td>mean</td>
<td>sigma</td>
<td></td>
<td>Most data</td>
</tr>
<tr>
<td>Multivariate normal</td>
<td>mvnorm</td>
<td>mean</td>
<td>r</td>
<td>sigma</td>
<td>Most data</td>
</tr>
<tr>
<td>Log Normal</td>
<td>lnorm</td>
<td>log mean</td>
<td>log sigma</td>
<td></td>
<td>income or reaction time</td>
</tr>
<tr>
<td>Uniform</td>
<td>unif</td>
<td>min</td>
<td>max</td>
<td></td>
<td>rectangular distributions</td>
</tr>
<tr>
<td>Binomial</td>
<td>binom</td>
<td>size</td>
<td>prob</td>
<td></td>
<td>Bernoulli trials (e.g. coin flips)</td>
</tr>
<tr>
<td>Student’s $t$</td>
<td>t</td>
<td>df</td>
<td>prob</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>
<td>Multivariate applications</td>
</tr>
<tr>
<td>Fisher’s $F$</td>
<td>f</td>
<td>df1</td>
<td>df2</td>
<td>nc</td>
<td>Testing for significance of F test</td>
</tr>
<tr>
<td>$\chi^2$</td>
<td>chisq</td>
<td>df</td>
<td></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></td>
<td>Exponential decay</td>
</tr>
<tr>
<td>Gamma</td>
<td>gamma</td>
<td>shape</td>
<td>rate</td>
<td>scale</td>
<td>distribution theoryh</td>
</tr>
<tr>
<td>Hypergeometric</td>
<td>hyper</td>
<td>m</td>
<td>n</td>
<td>k</td>
<td></td>
</tr>
<tr>
<td>Logistic</td>
<td>logis</td>
<td>location</td>
<td>scale</td>
<td></td>
<td>Item Response Theory</td>
</tr>
<tr>
<td>Poisson</td>
<td>pois</td>
<td>lambda</td>
<td></td>
<td></td>
<td>Count data</td>
</tr>
<tr>
<td>Weibull</td>
<td>weibull</td>
<td>shape</td>
<td>scale</td>
<td></td>
<td>Reaction time distributions</td>
</tr>
</tbody>
</table>
An example of using r, p, and q for a distributions

R code

```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
```

See (Example 2)
R can draw distributions

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")
```

(Example 3)
R can draw more interesting distributions

Log normal

Chi Square distribution

Normal and t with 4 df

The normal curve
R is also a graphics calculator

R code

```r
op <- par(mfrow=c(2,2))  #set up a 2 x 2 graph
curve(dlnorm(x),0,5,ylab='Probability of log(x)',main='Log normal')
curve(dchisq(x,1),0,5,ylab='Probability of Chi Sq',xlab='Chi Sq',main='Chi Square distribution')
curve(dnorm(x),-4,4,ylab='Probability of z or t',xlab='z or t',main='Normal and t with 4 df')
curve(dt(x,4),add=TRUE)
#
#somewhat more complicated
#first draw the normal curve
curve(dnorm(x),-3,3,xlab="",ylab="Probability of z")  #the range of x
title(main="The normal curve",outer=FALSE)  #the title
#add the cross hatching by using polygons
xvals <- seq(-3,-2,length=100)  #From -3 to 2 with 100 points
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)
op <- par(mfrow=c(1,1))  #back to a normal 1 x 1 graph
```

```
R can help teach with 100s of example data sets.

```r
> data()

> data(package="psych")
#see the names of the 56 data sets

> data(Titanic)
> ? Titanic

> data(cushny)
> ? cushny

> 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.
**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")
```

Set parameters

1. `xlab` for x axis label
2. `ylab` for y axis label
3. `main` for title
4. (Example 4)
A simple scatter plot using `plot` with some colors and shapes

Fisher Iris data with colors and shapes

R code

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

Set parameters

1. `bg` for background colors of symbols
2. `pch` chooses the plot character
3. Note how these depend upon `iris[,5]` which is the species
Show the various graphic options for plot character (pch)

plot symbols : points (... pch = *, cex = 3 )
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 scatter plot matrix plot with loess regressions using `pairs.panels`

```r
pairs.panels(iris[1:4], bg=c("red", "yellow", "blue"))
[iris$Species], pch=21, main="Fisher Iris data by Species", stars=TRUE)
```
Data preparation, descriptive statistics, data cleaning, correlation plots: (Examples part ii)

A brief example with real data - example 5

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

1. First read the data, either from a built in data set, a local file, a remote file, or from the clipboard.

2. Describe the data using the `describe` function from `psych`

```R
my.data <- sat.act # an example data file that is part of psych
# or
# my.data <- read.file() # look for it on your hard drive
# or
file.name <- "http://personality-project.org/r/aps/sat.act.txt"
# now read it either locally or remotely
my.data <- read.file(file.name)
# or if you have copied the data to the clipboard
# my.data <- read.clipboard() # you can read it from there
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>1</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>2</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>3</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>4</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>5</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>6</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>
```
Data preparation, descriptive statistics, data cleaning, correlation plots: (Examples part ii)

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

`pairs.panels(my.data)  #Note the outlier for ACT`
Clean up the data using `scrub`. Use `?scrub` for help on the parameters.

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
pairs.panels(cleaned)
```

<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>1</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>2</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>3</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>4</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>5</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>6</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>
Data preparation, descriptive statistics, data cleaning, correlation plots: (Examples part ii)

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

![Correlation matrix and scatter plots](image)
Data preparation, descriptive statistics, data cleaning, correlation plots: (Examples part ii)

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.

R code

#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.

```r
lowerCor(cleaned)
```

<table>
<thead>
<tr>
<th></th>
<th>gendr</th>
<th>edctn</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></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>education</td>
<td>0.09</td>
<td>1.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>age</td>
<td>-0.02</td>
<td>0.55</td>
<td>1.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACT</td>
<td>-0.05</td>
<td>0.15</td>
<td>0.11</td>
<td>1.00</td>
<td></td>
<td></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></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>
Testing the significance of one correlation using `cor.test`.

```
R code

```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`
Inferential statistics

**Test many correlations for significance using** `corr.test`

```r
corr.test(cleaned)
```

### 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>-0.05</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>

### 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>699</td>
<td>700</td>
<td>687</td>
<td></td>
</tr>
<tr>
<td>SATQ</td>
<td>687</td>
<td>687</td>
<td>686</td>
<td>687</td>
<td>687</td>
<td></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>0.00</td>
</tr>
<tr>
<td>education</td>
<td>0.02</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
<td>1.00</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>1.00</td>
</tr>
<tr>
<td>ACT</td>
<td>0.21</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>
The **SAT.ACT** correlations. Confidence values from resampling

\[
\text{ci} \leftarrow \text{cor.ci( cleaned, main='Heat map of sat.act')}\right)
\]
The SAT.ACT bootstrapped confidence intervals of correlation

cor.plot.upperLowerCi(ci,main="Heat map of sat.act")

Confidence values of the sat.act data
Inferential statistics

Are education and gender independent? $\chi^2$ Test of association

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

> T

<table>
<thead>
<tr>
<th>gender</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<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>

> chisq.test(T)

Pearson's Chi-squared test

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

1. First create a table of associations
   - Do this on our data (my.data)
   - Use the “with” command to specify the data set

2. Show the table

3. Apply $\chi^2$ test
Finding $\chi^2$ from a table of data

- Consider the effect of a treatment on later arrest (From Ashley Kendall, 2016)

<table>
<thead>
<tr>
<th>Condition</th>
<th>Arrested</th>
<th>Not Arrested</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control</td>
<td>14</td>
<td>21</td>
</tr>
<tr>
<td>Treatment</td>
<td>3</td>
<td>23</td>
</tr>
</tbody>
</table>

R code

```r
ak.df <- data.frame(Control=c(14,21),Treated =c(3,23))
rownames(ak.df) <- c("Arrested","Not Arrested")
ak.df #show the data frame
chisq.test(ak.df) #Test it using the Yates continuity correction
```

> ak.df #show the data frame

Control Treated
Arrested 14 3
Not Arrested 21 23

> chisq.test(ak.df) #Test it using the Yates continuity correction

Pearson's Chi-squared test with Yates' continuity correction
data:  ak.df
X-squared = 4.6791, df = 1, p-value = 0.03053
Graph the tabled data showing confidence intervals of proportions

R code

```r
ak.df <- data.frame(Control=c(14,21), Treated = c(3,23))
ak.p <- t(t(ak.df)/colSums(ak.df)) # convert to probabilities
standard.error <- sqrt(ak.p[1,] * ak.p[2,]/colSums(ak.df))
stats <- data.frame(mean=as.vector(ak.p),
                    se=rep(standard.error, each=2))
rownames(stats) <- c("Control Arrested", "Control Not",
                    "Treatment Arrested", "Treatment Not")
error.bars(stats=stats, bars=TRUE, space=c(.1,.1,1,.1),
density=c(20,-10,20,-10), ylab="Probability",
xlab="Control vs Treatment",
main = "Effect of Treatment on subsequent arrest (95% confidence)"")
```

```
round(stats, 2)

<table>
<thead>
<tr>
<th></th>
<th>mean</th>
<th>se</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control Arrested</td>
<td>0.40</td>
<td>0.08</td>
</tr>
<tr>
<td>Control Not</td>
<td>0.60</td>
<td>0.08</td>
</tr>
<tr>
<td>Treatment Arrested</td>
<td>0.12</td>
<td>0.06</td>
</tr>
<tr>
<td>Treatment Not</td>
<td>0.88</td>
<td>0.06</td>
</tr>
</tbody>
</table>
```
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)                  180.873486   23.410190       7.726   3.96e-14 ***
education                    1.2404347   2.3236106      0.534    0.59363
gender                       20.692714    6.9965074     2.958    0.00321 **
SATQ                          0.6448893   0.02890928   22.309    < 2e-16 ***
```
Signif. codes:  ** 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

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
R code
csat <- data.frame(scale(my.data,scale=FALSE))
describe(csat) #centered not standardized data
```

<table>
<thead>
<tr>
<th>vars</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>-0.07</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>2.42</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>0.53</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>0.33</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>187.77</td>
<td>1.64</td>
<td>2.42</td>
<td>0.36</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 data.frame if we want to use the linear model on it.
Zero center the data before examining interactions

R code

csat <- data.frame(scale(my.data,scale=FALSE))
mod2 <- lm(SATV ~ education * gender * SATQ,data=csat)
summary(mod2)

Call:
  lm(formula = SATV ~ education * gender * SATQ, data = csat)

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

Residual standard error: 84.69 on 679 degrees of freedom
(13 observations deleted due to missingness)
Multiple R-squared:  0.4469,    Adjusted R-squared:  0.4412
F-statistic: 78.37 on 7 and 679 DF,  p-value: < 2.2e-16
### Compare model 1 and model 2 using `anova`

Test the difference between the two linear models:

```r
R code
anova(mod1, mod2)
```

<table>
<thead>
<tr>
<th>Analysis of Variance Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis of Variance Table</td>
</tr>
</tbody>
</table>

**Model 1:** \( \text{SATV} \sim \text{education} + \text{gender} + \text{SATQ} \)  
**Model 2:** \( \text{SATV} \sim \text{education} \times \text{gender} \times \text{SATQ} \)

<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

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

R code

```r
# first plot the data points
with(my.data, plot(SATV ~ SATQ, col=c("blue","red")[gender]))

# add the regression lines
by(my.data, my.data$gender, 
   function(x) abline(lm(SATV ~ SATQ, data=x),
                   lty=c("solid","dashed")[x$gender])

# add a title
title("Verbal varies by Quant and gender")

# label the lines
text(250,320,"male")
text(250,430,"female")
```
Show the regression lines by education

Verbal varies by Quant and education

R code

```r
with(my.data,plot(SATV~SATQ,
col=c("blue","red")[gender],
pch=20)) #plot character
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?
Outline

Part I: What is R, where did it come from, why use it
  • Installing R and adding packages: the building blocks of R
Part II: A brief introduction – an overview
  • R is just a fancy (very fancy) calculator
  • Descriptive data analysis
  • Some inferential analysis
Part III R is a powerful statistical system
  • Data entry (detail and practice)
  • Descriptive (again)
  • Inferential (t and F with more practice)
  • Regression (including mediation and moderation)
  • Basic R commands
Part IV: Psychometrics
  • Reliability and its discontents
  • EFA, CFA, SEM
Part V: Help and More Help
  • List of useful commands
Part VI: The psych package and more practice
Outline of Part III

-> Part II: Introduction and Overview

Basic statistics and graphics
  4 steps: read, explore, test, graph

Basic descriptive statistics and graphics
  Graphic displays
  Correlations

Inferential statistics
  The t-test
  ANOVA

Linear Regression, Moderation and Mediation
  Regression from the raw data
  Regression from covariance/correlation matrices

R structure
  Basic R
  Objects and Functions

-> Part IV: Psychometrics
4 steps: read, explore, test, graph

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 & 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 from the ICAR project (Condon &amp; Revelle, 2014)</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 (1937) 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 (Smillie, Cooper, Wilt &amp; Revelle, 2012)</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) (Bond, 1995)</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 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 (Costa &amp; McCrae, 1985)</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 (Guilford, 1954)</td>
</tr>
</tbody>
</table>
4 steps: read, explore, test, graph

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 delims
   my.data <- read.clipboard.csv()  # assumes headers and comma delimited
   my.data <- read.clipboard.tab()  # assumes headers and tab delimited
   (e.g., from Excel)
   my.data <- read.clipboard.lower()  # read in a matrix given the lower
   my.data <- read.clipboard.upper()  # or upper off diagonal
   my.data <- read.clipboard.fwf()  # read in data using a fixed format
   (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?
4 steps: read, explore, test, graph

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.

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.file(file.name)
# or
my.data = read.table(file.name, header=TRUE)  # the conventional way
dim(my.data)  # find the dimensionality of our data
describe(my.data)  # describe it to check the means, ranges, etc.
```

> 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

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

R code

```r
datafilename = "http://personality-project.org/r/datasets/R.appendix1.data"
data.ex1 <- read.file(datafilename)
dim(data.ex1) # what are the dimensions of what we read?
data.ex1 # show the data
headTail(data.ex1) # just the top and bottom lines
describe(data.ex1) # descriptive stats
```

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

> headTail(data.ex1) # just the top and bottom lines

<table>
<thead>
<tr>
<th>Dosage Alertness</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>15</td>
</tr>
<tr>
<td>16</td>
</tr>
<tr>
<td>17</td>
</tr>
<tr>
<td>18</td>
</tr>
</tbody>
</table>

> describe(data.ex1) # descriptive stats

<table>
<thead>
<tr>
<th>vars</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>Dosage</td>
<td>1</td>
<td>18</td>
<td>1.89</td>
<td>0.76</td>
<td>2</td>
<td>1.88</td>
<td>1.48</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>-1.35</td>
<td>0.18</td>
</tr>
<tr>
<td>Alertness</td>
<td>2</td>
<td>18</td>
<td>27.67</td>
<td>6.82</td>
<td>27</td>
<td>27.50</td>
<td>8.15</td>
<td>17</td>
<td>41</td>
<td>24</td>
<td>0.25</td>
<td>-1.06</td>
</tr>
</tbody>
</table>
However, some might want to Import SAS or SPSS files

The first thing to try is the `read.file` function. For more complicated data sets, there are several different packages that make importing SPSS, SAS, Systat, etc. files possible to do.

- **read.file** Function in psych to read .txt, .csv, .sav, .xpt, .r, .rda, .text (etc.)
- **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)
> describe(eli, skew=FALSE)

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

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

```r
> 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)
```

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

> library(haven)

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

> eli <- read_spss(datafilename)  #note that it figures out what to do
> headTail(eli,3,2) The first 3 and last 2
> describe(eli,skew=FALSE)

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
Part II <- Basics
          Descriptives      Inferential    Regression/Mediation R structure
          -----------      -----------      -----------
          0000000000      0000000000
          00

4 steps: read, explore, test, graph

read.file as a convenient solution to reading files

1. Combines file.choose and read.table
2. Also, based upon the suffix of the data, will choose the most likely way to read a SPSS, csv, text, rds or SAS export file.
3. Not as powerful as foreign or rio but easier.
4. Automatically reads SPSS .sav files as numeric values but can read them with the item information as well.

R code

eli <- read.file(). #goes off and searches for a local file
datafilename <- "http://personality-project.org/r/datasets/finkel.sav"
eli <- read.file(datafilename). #uses that remote address to get it
ashley <- read.file() #a file from Ashley Kendall on my computer
kendall <- read.file(read.file(use.value.labels=TRUE) #keep the labels
ashley[1:3,8:17]
kendall[1:3,8:17]

ashley[1:3,8:17]

<p>| | | | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>7</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

kendall[1:3,8:17]

<p>| | | | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8</td>
<td>3</td>
<td>0</td>
<td>a little</td>
<td>0</td>
<td>a little</td>
<td>somewhat</td>
<td>0</td>
<td>very much</td>
<td>somewhat</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>1</td>
<td>0</td>
<td>not at all</td>
<td>2</td>
<td>not at all</td>
<td>very much</td>
<td>0</td>
<td>somewhat</td>
<td>a little</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>7</td>
<td>0</td>
<td>very much</td>
<td>3</td>
<td>a little</td>
<td>not at all</td>
<td>0</td>
<td>not at all</td>
<td>not at all</td>
<td></td>
</tr>
</tbody>
</table>
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 them

Read in some data, look at the first and last few cases (using headTail), and then get basic descriptive statistics. For this example, we will use a built in data set.

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

```
epiE epiS epiImp epilie epiNeur bfagree bfcon bfext bfneur bfopen
1 18 10 7 3 9 138 96 141 51 1
2 16 8 5 1 12 101 99 107 116 132
3 6 1 3 2 5 143 118 38 68
4 12 6 4 3 15 104 106 64 114
228 12 7 4 3 15 155 129 127 88
229 19 10 7 2 11 162 152 163 104
230 4 1 1 2 10 95 111 75 123
231 8 6 3 2 15 85 62 90 131
```

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

```r
describe(epi.bfi)
```

<p>| | | | | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>var</td>
<td>n</td>
<td>mean</td>
<td>sd</td>
<td>median</td>
<td>trimmed</td>
<td>mad</td>
<td>min</td>
<td>max</td>
<td>range</td>
<td>skew</td>
<td>kurtosis</td>
<td>se</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>
</tr>
<tr>
<td>epiE</td>
<td>1</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>
</tr>
<tr>
<td>epiS</td>
<td>2</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>
</tr>
<tr>
<td>epiImp</td>
<td>3</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>0.06</td>
<td>-0.59</td>
</tr>
<tr>
<td>epilie</td>
<td>4</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>
</tr>
<tr>
<td>epiNeur</td>
<td>5</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>0.06</td>
<td>-0.46</td>
</tr>
<tr>
<td>bfagree</td>
<td>6</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>
</tr>
<tr>
<td>bfcon</td>
<td>7</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>
</tr>
<tr>
<td>bfext</td>
<td>8</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>
</tr>
<tr>
<td>bfnuer</td>
<td>9</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>
</tr>
<tr>
<td>bfopen</td>
<td>10</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>
</tr>
<tr>
<td>bdi</td>
<td>11</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>
</tr>
<tr>
<td>traitanx</td>
<td>12</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>
</tr>
<tr>
<td>stateanx</td>
<td>13</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>
</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`

`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.

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

```
pairs.panels(my.data[1:5])
```
Plot the scatter plot matrix (SPLOM) of the first 5 variables using the `pairs.panels` function but with smaller plot character (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 code**

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

<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>epiE</td>
<td>1.00</td>
<td>0.85</td>
<td>0.80</td>
<td>-0.22</td>
<td>-0.18</td>
<td>0.18</td>
<td>-0.11</td>
<td>0.54</td>
<td>-0.09</td>
<td>0.14</td>
<td>-0.16</td>
<td>-0.23</td>
<td></td>
</tr>
<tr>
<td>epiS</td>
<td>0.85</td>
<td>1.00</td>
<td>0.43</td>
<td>-0.05</td>
<td>-0.22</td>
<td>0.20</td>
<td>0.05</td>
<td>0.58</td>
<td>-0.07</td>
<td>0.15</td>
<td>-0.13</td>
<td>-0.26</td>
<td></td>
</tr>
<tr>
<td>epiImp</td>
<td>0.80</td>
<td>0.43</td>
<td>1.00</td>
<td>-0.24</td>
<td>-0.07</td>
<td>0.08</td>
<td>-0.24</td>
<td>0.35</td>
<td>-0.09</td>
<td>0.07</td>
<td>-0.11</td>
<td>-0.12</td>
<td></td>
</tr>
<tr>
<td>epilie</td>
<td>-0.22</td>
<td>-0.05</td>
<td>-0.24</td>
<td>1.00</td>
<td>-0.25</td>
<td>0.17</td>
<td>0.23</td>
<td>-0.04</td>
<td>-0.22</td>
<td>-0.03</td>
<td>-0.20</td>
<td>-0.23</td>
<td></td>
</tr>
<tr>
<td>epiNeur</td>
<td>-0.18</td>
<td>-0.22</td>
<td>-0.07</td>
<td>-0.25</td>
<td>1.00</td>
<td>-0.08</td>
<td>-0.13</td>
<td>-0.17</td>
<td>0.63</td>
<td>0.09</td>
<td>0.58</td>
<td>0.73</td>
<td></td>
</tr>
<tr>
<td>bfagree</td>
<td>0.18</td>
<td>0.20</td>
<td>0.08</td>
<td>0.17</td>
<td>-0.08</td>
<td>1.00</td>
<td>0.45</td>
<td>0.48</td>
<td>-0.04</td>
<td>0.39</td>
<td>-0.14</td>
<td>-0.31</td>
<td></td>
</tr>
<tr>
<td>bfcon</td>
<td>-0.11</td>
<td>0.05</td>
<td>-0.24</td>
<td>0.23</td>
<td>-0.13</td>
<td>0.45</td>
<td>1.00</td>
<td>0.27</td>
<td>0.04</td>
<td>0.31</td>
<td>-0.18</td>
<td>-0.29</td>
<td></td>
</tr>
<tr>
<td>bfext</td>
<td>0.54</td>
<td>0.58</td>
<td>0.35</td>
<td>-0.04</td>
<td>-0.17</td>
<td>0.48</td>
<td>0.27</td>
<td>1.00</td>
<td>0.04</td>
<td>0.46</td>
<td>-0.14</td>
<td>-0.39</td>
<td></td>
</tr>
<tr>
<td>bfneur</td>
<td>-0.09</td>
<td>-0.07</td>
<td>-0.09</td>
<td>-0.22</td>
<td>0.63</td>
<td>-0.04</td>
<td>0.04</td>
<td>0.04</td>
<td>1.00</td>
<td>0.29</td>
<td>0.47</td>
<td>0.59</td>
<td></td>
</tr>
<tr>
<td>bfopen</td>
<td>0.14</td>
<td>0.15</td>
<td>0.07</td>
<td>-0.03</td>
<td>0.09</td>
<td>0.39</td>
<td>0.31</td>
<td>0.46</td>
<td>0.29</td>
<td>1.00</td>
<td>-0.08</td>
<td>-0.11</td>
<td></td>
</tr>
<tr>
<td>bdi</td>
<td>-0.16</td>
<td>-0.13</td>
<td>-0.11</td>
<td>-0.20</td>
<td>0.58</td>
<td>-0.14</td>
<td>-0.18</td>
<td>-0.14</td>
<td>0.47</td>
<td>-0.08</td>
<td>1.00</td>
<td>0.65</td>
<td></td>
</tr>
<tr>
<td>traitanx</td>
<td>-0.23</td>
<td>-0.26</td>
<td>-0.12</td>
<td>-0.23</td>
<td>0.73</td>
<td>-0.31</td>
<td>-0.29</td>
<td>-0.39</td>
<td>0.59</td>
<td>-0.11</td>
<td>0.65</td>
<td>1.00</td>
<td></td>
</tr>
<tr>
<td>stateanx</td>
<td>-0.13</td>
<td>-0.12</td>
<td>-0.09</td>
<td>-0.15</td>
<td>0.49</td>
<td>-0.19</td>
<td>-0.14</td>
<td>-0.15</td>
<td>0.49</td>
<td>-0.04</td>
<td>0.61</td>
<td>0.57</td>
<td></td>
</tr>
</tbody>
</table>
**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 bfagr bfcon bfext bfner bfopen 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
bfagree  0.18  0.20  0.08  0.17 -0.08  1.00
bfcon   -0.11  0.05 -0.24  0.23 -0.13  0.45  1.00
bfext   0.54  0.58  0.35  0.04 -0.17  0.48  0.27  1.00
bfneur  -0.09 -0.07 -0.09 -0.22  0.63 -0.04  0.04  0.04  1.00
bfopen  0.14  0.15  0.07 -0.03  0.09  0.39  0.31  0.46  0.29  1.00
bdi     -0.16 -0.13 -0.11 -0.20  0.58 -0.14 -0.18 -0.14  0.47 -0.08  1.00
traitanx -0.23 -0.26 -0.12 -0.23  0.73 -0.31 -0.29 -0.39  0.59 -0.11  0.65  1.00
stateanx -0.13 -0.12 -0.09 -0.15  0.49 -0.19 -0.14 -0.15  0.49 -0.04  0.61  0.57  1.00
```
Test the significance and use Holm correction for multiple tests

```r
corr.test(my.data)
```

Call: `corr.test(x = my.data)`

Correlation matrix

<table>
<thead>
<tr>
<th></th>
<th>epiE</th>
<th>epiS</th>
<th>epiImp</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>epiE</td>
<td>1.00</td>
<td>0.85</td>
<td>0.80</td>
<td>-0.22</td>
<td>-0.18</td>
<td>0.18</td>
<td>-0.11</td>
<td>0.54</td>
<td>-0.09</td>
<td>0.14</td>
<td>-0.16</td>
<td>-0.23</td>
</tr>
<tr>
<td>epiS</td>
<td>0.85</td>
<td>1.00</td>
<td>0.43</td>
<td>-0.05</td>
<td>-0.22</td>
<td>0.20</td>
<td>0.05</td>
<td>0.58</td>
<td>-0.07</td>
<td>0.15</td>
<td>-0.13</td>
<td>-0.26</td>
</tr>
<tr>
<td>epiImp</td>
<td>0.80</td>
<td>0.43</td>
<td>1.00</td>
<td>-0.24</td>
<td>-0.07</td>
<td>0.08</td>
<td>-0.24</td>
<td>0.35</td>
<td>-0.09</td>
<td>0.07</td>
<td>-0.11</td>
<td>-0.12</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>
</tr>
<tr>
<td>stateanx</td>
<td>-0.13</td>
<td>-0.12</td>
<td>-0.09</td>
<td>-0.15</td>
<td>0.49</td>
<td>-0.19</td>
<td>-0.14</td>
<td>-0.15</td>
<td>0.49</td>
<td>-0.04</td>
<td>0.61</td>
<td>0.57</td>
</tr>
</tbody>
</table>

Sample Size

<table>
<thead>
<tr>
<th></th>
<th>epiE</th>
<th>epiS</th>
<th>epiImp</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>epiE</td>
<td>231</td>
<td>231</td>
<td>231</td>
<td>231</td>
<td>231</td>
<td>231</td>
<td>231</td>
<td>231</td>
<td>231</td>
<td>231</td>
<td>231</td>
<td>231</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>
</tr>
<tr>
<td>stateanx</td>
<td>231</td>
<td>231</td>
<td>231</td>
<td>231</td>
<td>231</td>
<td>231</td>
<td>231</td>
<td>231</td>
<td>231</td>
<td>231</td>
<td>231</td>
<td>231</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th></th>
<th>epiE</th>
<th>epiS</th>
<th>epiImp</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>epiE</td>
<td>0.00</td>
<td>0.00</td>
<td>0.03</td>
<td>0.27</td>
<td>0.27</td>
<td>1.00</td>
<td>0.00</td>
<td>1.00</td>
<td>1.00</td>
<td>0.59</td>
<td>0.02</td>
<td></td>
</tr>
<tr>
<td>epiS</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
<td>0.04</td>
<td>0.08</td>
<td>1.00</td>
<td>0.00</td>
<td>1.00</td>
<td>1.00</td>
<td>0.62</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>epiImp</td>
<td>0.00</td>
<td>0.00</td>
<td>0.01</td>
<td>1.00</td>
<td>1.00</td>
<td>0.01</td>
<td>0.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>0.00</td>
<td></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>
</tr>
<tr>
<td>stateanx</td>
<td>0.00</td>
<td>0.00</td>
<td>0.03</td>
<td>0.27</td>
<td>1.00</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>1.00</td>
<td>0.59</td>
<td>0.02</td>
<td></td>
</tr>
</tbody>
</table>
William Gossett, publishing under the name Student reported a small sample approximation (t) to the large sample z test. His first example was a data set on the different effect of optical isomers of hyoscyamine hydrobromide reported by Cushny & Peebles (1905). The sleep of 10 patients was measured without any drug and then following administration of D. and L isomers. The data from Cushny are available as the cushny data set.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Cntrl</th>
<th>drug1</th>
<th>drg2L</th>
<th>drg2R</th>
<th>delt1</th>
<th>dlt2L</th>
<th>dlt2R</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.60</td>
<td>1.3</td>
<td>2.50</td>
<td>2.10</td>
<td>0.70</td>
<td>1.90</td>
<td>1.50</td>
</tr>
<tr>
<td>2</td>
<td>3.00</td>
<td>1.4</td>
<td>3.80</td>
<td>4.40</td>
<td>-1.60</td>
<td>0.80</td>
<td>1.40</td>
</tr>
<tr>
<td>3</td>
<td>4.70</td>
<td>4.5</td>
<td>5.80</td>
<td>4.70</td>
<td>-0.20</td>
<td>1.10</td>
<td>0.00</td>
</tr>
<tr>
<td>4</td>
<td>5.50</td>
<td>4.3</td>
<td>5.60</td>
<td>4.80</td>
<td>-1.20</td>
<td>0.10</td>
<td>-0.70</td>
</tr>
<tr>
<td>5</td>
<td>6.20</td>
<td>6.1</td>
<td>6.10</td>
<td>6.70</td>
<td>-0.10</td>
<td>-0.10</td>
<td>0.50</td>
</tr>
<tr>
<td>6</td>
<td>3.20</td>
<td>6.6</td>
<td>7.60</td>
<td>8.30</td>
<td>3.40</td>
<td>4.40</td>
<td>5.10</td>
</tr>
<tr>
<td>7</td>
<td>2.50</td>
<td>6.2</td>
<td>8.00</td>
<td>8.20</td>
<td>3.70</td>
<td>5.50</td>
<td>5.70</td>
</tr>
<tr>
<td>8</td>
<td>2.80</td>
<td>3.6</td>
<td>4.40</td>
<td>4.30</td>
<td>0.80</td>
<td>1.60</td>
<td>1.50</td>
</tr>
<tr>
<td>9</td>
<td>1.10</td>
<td>1.1</td>
<td>5.70</td>
<td>5.80</td>
<td>0.00</td>
<td>4.60</td>
<td>4.70</td>
</tr>
<tr>
<td>10</td>
<td>2.90</td>
<td>4.9</td>
<td>6.30</td>
<td>6.40</td>
<td>2.00</td>
<td>3.40</td>
<td>3.50</td>
</tr>
<tr>
<td>Mean</td>
<td>3.25</td>
<td>4.0</td>
<td>5.58</td>
<td>5.57</td>
<td>0.75</td>
<td>2.33</td>
<td>2.32</td>
</tr>
<tr>
<td>Sd</td>
<td>1.78</td>
<td>2.1</td>
<td>1.66</td>
<td>1.91</td>
<td>1.79</td>
<td>2.00</td>
<td>2.27</td>
</tr>
</tbody>
</table>

R code

```R
error.bars(cushny,xlab="Group",ylab="hours of sleep",
main="The effect of drug upon sleep (95% confidence)")
```
The **cushny** data set with error bars (Cushny & Peebles, 1905)

**R code**

```r
error.bars(cushny, xlab="Group", ylab="hours of sleep", main="The effect of drug upon sleep (95% confidence)")
```

We can show these data graphically using the `error.bars` function. We pass labels to the x and y axis using the `xlab` and `ylab` parameters, and then supply an appropriate figure title.

We will use these data to show how to do t-tests as well as the generalization to Analysis of Variance.
The t-test

**Student’s t.test: As done by Student**

```r
with(cushny,t.test(delta1))  #control versus drug 1 difference scores
with(cushny,t.test(delta2L))  #control versus drug2L difference scores
with(cushny,t.test(delta1,delta2L,paired=TRUE))  #difference of differences
```

```r
> with(cushny,t.test(delta1))  #control versus drug 1 difference scores
   One Sample t-test
data:  delta1
t = 1.3257, df = 9, p-value = 0.2176
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
   -0.5297804  2.0297804
sample estimates:
mean of x
   0.75

> with(cushny,t.test(delta2L))  #control versus drug2L difference scores
   One Sample t-test
data:  delta2L
t = 3.6799, df = 9, p-value = 0.005076
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
   0.8976775  3.7623225
sample estimates:
mean of x
   2.33

> with(cushny,t.test(delta1,delta2L,paired=TRUE))  #difference of differences
   Paired t-test
data:  delta1 and delta2L
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
```
Two ways of organizing the data: Wide versus long

We can take the wide format of the cushny data set and make it long.

cushny[c("delta1","delta2L")]
delta1  delta2L
1     0.7      1.9
2    -1.6      0.8
3     -0.2      1.1
4    -1.2      0.1
5    -0.1      -0.1
6     3.4      4.4
7     3.7      5.5
8     0.8      1.6
9     0.0      4.6
10    2.0      3.4

R code

long.sleep <- stack(cushny[c("delta1","delta2L")])
long.sleep
The t-test

R code

```r
long.sleep <- +  stack(cushy[c("delta1", "delta2L")])
```

```r
t.test(values ~ ind, data=long.sleep)
```

Welch Two Sample t-test
data: values by ind
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 delta1 mean in group delta2L
0.75 2.33

But, the data were paired

R code

```r
t.test(values ~ ind, data=long.sleep, paired=TRUE)
```

data: values by ind
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
The t-test demonstration with Student’s data (from the sleep dataset)

Sleep data set is just 2 columns of cushny

R code

```r
with(sleep, t.test(extra~group))
with(sleep, t.test(extra~group, var.equal=TRUE))
```

Welch Two Sample t-test
data: extra by group
t = -1.8608, df = 17.776, p-value = 0.07939  
--- default value
t = -1.8608, df = 18, p-value = 0.07919.  
--- equal variances
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
0.75 2.33

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

R code

```r
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
ANOVA

**Analysis of Variance as special case of linear model**

1. `aov` provides a wrapper to `lm` for fitting linear models to balanced or unbalanced experimental designs.
2. The main difference from `lm` is in the way `print`, `summary` and so on handle the fit: this is expressed in the traditional language of the analysis of variance rather than that of linear models.
3. If the formula contains a single Error term, this is used to specify error strata, and appropriate models are fitted within each error stratum.
4. The formula can specify multiple responses.
5. `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.
6. 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`.

Several examples are available. One is found by `?aov` and is taken from Venables & Ripley (2002). This is an example of the effect of Nitrogen (N) Potassium (K) and Phosphate (P) on the growth of Peas. (Remember that the people who developed ANOVA were agronomists.). Another example is the effect of gender and drug on alertness. First, we will examine the `sleep` data set, using `aov` instead of the `t.test`.
ANOVA

*aov of the sleep data set: compare with the t.test results*

```
#independent subjects
summary(aov(extra ~ group, data=sleep))
```

```
> summary(aov(extra ~ group, data=sleep))

Df  Sum Sq Mean Sq   F value  Pr(>F)
group    1 12.482 12.482   3.4630   0.0792 .
Residuals 18 64.891  3.605
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

> t = -1.8608, df = 18, p-value = 0.07919. <- equal variances
```

```
#correlated subjects
summary(aov(extra~group + Error(ID),data=sleep))
```

```
> summary(aov(extra~group + Error(ID),data=sleep))

Error: ID
  Df  Sum Sq Mean Sq   F value  Pr(>F)
       9  58.083  6.453

Error: Within
  Df  Sum Sq Mean Sq   F value  Pr(>F)
group    1 12.482 12.482   16.50  0.00283 **
Residuals 9  6.808  0.756
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

> t = -4.0621, df = 9, p-value = 0.002833.
```
ANOVA

### aov: an example of chemicals upon the growth of peas.

#### R code

```
R code
npk #from Venables
```

<table>
<thead>
<tr>
<th>block</th>
<th>N</th>
<th>P</th>
<th>K</th>
<th>yield</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>13</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>15</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>17</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>18</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>19</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>20</td>
<td>5</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>21</td>
<td>6</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>22</td>
<td>6</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>23</td>
<td>6</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>24</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

#### Several models

```
R code
mod1 <- aov(yield ~ N, data=npk)
mod2 <- aov(yield ~ N + P + N*P, data=npk)
mod2a <- aov(yield ~ N*P, data=npk)
mod3 <- aov(yield ~ N*P*K, data=npk)
mod4 <- aov(yield ~ block + N*P*K, data=npk)
```

#### R code

```
> summary(mod1)
Df  Sum Sq  Mean Sq F value Pr(>F)
N    1 189.3  189.28  6.061 0.0221 *
Residuals 22 687.1  31.23
---
Signif. codes: 0 *** 0.001 ** 0.01 * 0.05 . 0.1 1

> summary(mod4)
Df  Sum Sq Mean Sq    F value Pr(>F)
block    5 343.3  68.66  4.447 0.01594 *
N        1 189.3 189.28 12.259 0.00437 **
P        1   8.4  8.40  0.544 0.47490
K        1  95.2  95.20  6.166 0.02880 *
N:P      1  21.3  21.28  1.378 0.26317
N:K      1  33.1  33.13  2.146 0.16865
P:K      1   0.5  0.48  0.031 0.86275
Residuals 12 185.3  15.44
---
Signif. codes: 0 *** 0.001 ** 0.01 * 0.05 . 0.1 1
```
Analysis of Variance: Another example

`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.

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

```R
# read the data into a data.frame
# show the data

data.ex2 = read.file(datafilename)  
data.ex2
```

<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>5</td>
<td>m</td>
<td>b</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>m</td>
<td>b</td>
<td>7</td>
</tr>
<tr>
<td>7</td>
<td>m</td>
<td>b</td>
<td>23</td>
</tr>
<tr>
<td>8</td>
<td>m</td>
<td>b</td>
<td>14</td>
</tr>
<tr>
<td>9</td>
<td>f</td>
<td>a</td>
<td>15</td>
</tr>
<tr>
<td>10</td>
<td>f</td>
<td>a</td>
<td>12</td>
</tr>
<tr>
<td>11</td>
<td>f</td>
<td>a</td>
<td>22</td>
</tr>
<tr>
<td>12</td>
<td>f</td>
<td>a</td>
<td>14</td>
</tr>
<tr>
<td>13</td>
<td>f</td>
<td>b</td>
<td>15</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

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

R code

```r
#do the analysis of variance
aov.ex2 <- aov(Alertness ~ Gender * Dosage, data=data.ex2)

summary(aov.ex2) #show the summary table
aov.ex2. #This shows the coefficients
```

```r
> aov.ex2 <- aov(Alertness ~ Gender * Dosage, data=data.ex2)
> summary(aov.ex2) #show the summary table

Df  Sum Sq Mean Sq  F value Pr(>F)
 Gender    1 76.562  76.562   2.952  0.111
 Dosage    1  5.062  5.062  0.195  0.666
 Gender:Dosage 1  0.062  0.062  0.002  0.962
 Residuals 12 311.250  25.94

aov(formula = Alertness ~ Gender * Dosage, data = data.ex2)

Terms:

<table>
<thead>
<tr>
<th></th>
<th>Gender</th>
<th>Dosage</th>
<th>Gender:Dosage</th>
<th>Residuals</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sum of Squares</td>
<td>76.5625</td>
<td>5.0625</td>
<td>0.0625</td>
<td>311.2500</td>
</tr>
<tr>
<td>Deg. of Freedom</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>12</td>
</tr>
</tbody>
</table>

Residual standard error: 5.092887
Estimated effects may be unbalanced
Show the results table

R code

```r
> print(model.tables(aov.ex2,"means"),digits=3)
Tables of means
Grand mean
14.0625

Gender
Gender
   f  m
16.25 11.88

Dosage
Dosage
   a  b
13.50 14.62

Gender:Dosage
   Dosage
Gender a  b
   f 15.75 16.75
   m 11.25 12.50
```
Analysis of Variance: Within subjects

1. Somewhat more complicated because we need to convert “wide” data.frames to “long” or “narrow” data.frames.
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
5. Many within subject designs can be treated as multi-level designs. For a discussion of analyzing multilevel data (particularly for personality dynamics), see http://personality-project.org/revelle/publications/rw.paid.17.final.pdf
Analysis of variance within subjects: Getting and showing the data

```r
filename="http://personality-project.org/r/datasets/R.appendix5.data"
data.ex5=read.file(filename)  #read the data into a data.frame
headTail(data.ex5,6,12)  #show the data (first 6, last 12)
```

<table>
<thead>
<tr>
<th>Obs</th>
<th>Subject</th>
<th>Gender</th>
<th>Dosage</th>
<th>Task</th>
<th>Valence</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A</td>
<td>M</td>
<td>A</td>
<td>F</td>
<td>Neg</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>A</td>
<td>M</td>
<td>A</td>
<td>F</td>
<td>Neu</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>A</td>
<td>M</td>
<td>A</td>
<td>F</td>
<td>Pos</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>A</td>
<td>M</td>
<td>A</td>
<td>C</td>
<td>Neg</td>
<td>7</td>
</tr>
<tr>
<td>5</td>
<td>A</td>
<td>M</td>
<td>A</td>
<td>C</td>
<td>Neu</td>
<td>9</td>
</tr>
<tr>
<td>6</td>
<td>A</td>
<td>M</td>
<td>A</td>
<td>C</td>
<td>Pos</td>
<td>10</td>
</tr>
<tr>
<td>97</td>
<td>Q</td>
<td>F</td>
<td>C</td>
<td>F</td>
<td>Neg</td>
<td>18</td>
</tr>
<tr>
<td>98</td>
<td>Q</td>
<td>F</td>
<td>C</td>
<td>F</td>
<td>Neu</td>
<td>17</td>
</tr>
<tr>
<td>99</td>
<td>Q</td>
<td>F</td>
<td>C</td>
<td>F</td>
<td>Pos</td>
<td>18</td>
</tr>
<tr>
<td>100</td>
<td>Q</td>
<td>F</td>
<td>C</td>
<td>C</td>
<td>Neg</td>
<td>17</td>
</tr>
<tr>
<td>101</td>
<td>Q</td>
<td>F</td>
<td>C</td>
<td>C</td>
<td>Neu</td>
<td>19</td>
</tr>
<tr>
<td>102</td>
<td>Q</td>
<td>F</td>
<td>C</td>
<td>C</td>
<td>Pos</td>
<td>19</td>
</tr>
<tr>
<td>103</td>
<td>R</td>
<td>F</td>
<td>C</td>
<td>F</td>
<td>Neg</td>
<td>19</td>
</tr>
<tr>
<td>104</td>
<td>R</td>
<td>F</td>
<td>C</td>
<td>F</td>
<td>Neu</td>
<td>17</td>
</tr>
<tr>
<td>105</td>
<td>R</td>
<td>F</td>
<td>C</td>
<td>F</td>
<td>Pos</td>
<td>19</td>
</tr>
<tr>
<td>106</td>
<td>R</td>
<td>F</td>
<td>C</td>
<td>C</td>
<td>Neg</td>
<td>22</td>
</tr>
<tr>
<td>107</td>
<td>R</td>
<td>F</td>
<td>C</td>
<td>C</td>
<td>Neu</td>
<td>21</td>
</tr>
<tr>
<td>108</td>
<td>R</td>
<td>F</td>
<td>C</td>
<td>C</td>
<td>Pos</td>
<td>20</td>
</tr>
</tbody>
</table>
Describe the data

describe(data.ex5)

<table>
<thead>
<tr>
<th>vars</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>Obs</td>
<td>1</td>
<td>108</td>
<td>54.50</td>
<td>31.32</td>
<td>54.5</td>
<td>54.5</td>
<td>1</td>
<td>108</td>
<td>107</td>
<td>0.00</td>
<td>-1.23</td>
<td>3.01</td>
</tr>
<tr>
<td>Subject*</td>
<td>2</td>
<td>108</td>
<td>9.50</td>
<td>5.21</td>
<td>9.5</td>
<td>9.5</td>
<td>1</td>
<td>18</td>
<td>17</td>
<td>0.00</td>
<td>-1.24</td>
<td>0.50</td>
</tr>
<tr>
<td>Gender*</td>
<td>3</td>
<td>108</td>
<td>1.50</td>
<td>0.50</td>
<td>1.5</td>
<td>1.5</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0.00</td>
<td>-2.02</td>
<td>0.05</td>
</tr>
<tr>
<td>Dosage*</td>
<td>4</td>
<td>108</td>
<td>2.00</td>
<td>0.82</td>
<td>2.0</td>
<td>2.0</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>0.00</td>
<td>-1.53</td>
<td>0.08</td>
</tr>
<tr>
<td>Task*</td>
<td>5</td>
<td>108</td>
<td>1.50</td>
<td>0.50</td>
<td>1.5</td>
<td>1.5</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0.00</td>
<td>-2.02</td>
<td>0.05</td>
</tr>
<tr>
<td>Valence*</td>
<td>6</td>
<td>108</td>
<td>2.00</td>
<td>0.82</td>
<td>2.0</td>
<td>2.0</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>0.00</td>
<td>-1.53</td>
<td>0.08</td>
</tr>
<tr>
<td>Recall</td>
<td>7</td>
<td>108</td>
<td>15.63</td>
<td>5.07</td>
<td>15.0</td>
<td>15.74</td>
<td>4</td>
<td>25</td>
<td>21</td>
<td>-0.13</td>
<td>-0.64</td>
<td>0.49</td>
</tr>
</tbody>
</table>

The * signify that the entries are not numerical, but rather categorical or logical.
Analysis of variance within subjects

R code

```r
filename="http://personality-project.org/r/datasets/R.appendix5.data"
data.ex5=read.table(filename,header=TRUE) #read the data into a table
#do the anova
aov.ex5 = aov(Recall~(Task*Valence*Gender*Dosage)+Error(Subject/(Task*Valence))+(Gender*Dosage),data.ex5)
#look at the output
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>542.3</td>
<td>542.3</td>
<td>5.685</td>
</tr>
<tr>
<td>Dosage</td>
<td>2</td>
<td>694.9</td>
<td>347.5</td>
<td>3.643</td>
</tr>
<tr>
<td>Gender:Dosage</td>
<td>2</td>
<td>70.8</td>
<td>35.4</td>
<td>0.371</td>
</tr>
<tr>
<td>Residuals</td>
<td>12</td>
<td>1144.6</td>
<td>95.4</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>96.33</td>
<td>96.33</td>
<td>39.862</td>
</tr>
<tr>
<td>Task:Gender</td>
<td>1</td>
<td>1.33</td>
<td>1.33</td>
<td>0.552</td>
</tr>
<tr>
<td>Task:Dosage</td>
<td>2</td>
<td>8.17</td>
<td>4.08</td>
<td>1.690</td>
</tr>
<tr>
<td>Task:Gender:Dosage</td>
<td>2</td>
<td>3.17</td>
<td>1.58</td>
<td>0.655</td>
</tr>
<tr>
<td>Residuals</td>
<td>12</td>
<td>29.00</td>
<td>2.42</td>
<td></td>
</tr>
</tbody>
</table>

---

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

+ lots more
### Analysis of variance within subjects output (continued)

#### Error: Subject:Valence

<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>Valence</td>
<td>2</td>
<td>14.69</td>
<td>7.343</td>
<td>2.998</td>
</tr>
<tr>
<td>Valence:Gender</td>
<td>2</td>
<td>3.91</td>
<td>1.954</td>
<td>0.798</td>
</tr>
<tr>
<td>Valence:Dosage</td>
<td>4</td>
<td>20.26</td>
<td>5.065</td>
<td>2.068</td>
</tr>
<tr>
<td>Valence:Gender:Dosage</td>
<td>4</td>
<td>1.04</td>
<td>0.259</td>
<td>0.106</td>
</tr>
<tr>
<td>Residuals</td>
<td>24</td>
<td>58.78</td>
<td>2.449</td>
<td></td>
</tr>
</tbody>
</table>

---

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

#### Error: Subject:Task:Valence

<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:Valence</td>
<td>2</td>
<td>5.39</td>
<td>2.6944</td>
<td>1.320</td>
</tr>
<tr>
<td>Task:Valence:Gender</td>
<td>2</td>
<td>2.17</td>
<td>1.0833</td>
<td>0.531</td>
</tr>
<tr>
<td>Task:Valence:Dosage</td>
<td>4</td>
<td>2.78</td>
<td>0.6944</td>
<td>0.340</td>
</tr>
<tr>
<td>Task:Valence:Gender:Dosage</td>
<td>4</td>
<td>2.67</td>
<td>0.6667</td>
<td>0.327</td>
</tr>
<tr>
<td>Residuals</td>
<td>24</td>
<td>49.00</td>
<td>2.0417</td>
<td></td>
</tr>
</tbody>
</table>
Multiple regression

1. Use the Garcia data set from psych (protest in Hayes (2013))
2. Do the linear model. (See the Garcia example)
3. Summarize the results

```r
mod1 <- lm(respappr ~ prot2 + sexism , data=Garcia)
summary(mod1)
```

Call:

```
  lm(formula = respappr ~ prot2 + sexism, data = Garcia)
```

Residuals:

```
    Min     1Q  Median     3Q    Max
-3.5636 -0.8091  0.1281  0.9028  2.3069
```

Coefficients:

```
                Estimate  Std. Error t value  Pr(>|t|)
(Intercept)  3.69100  0.69826  5.286 5.33e-07 ***
prot2       1.43715  0.22273  6.452 2.15e-09 ***
sexism       0.03809  0.13284  0.287   0.775
```

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

Residual standard error: 1.177 on 126 degrees of freedom

Multiple R-squared: 0.2497, Adjusted R-squared: 0.2378

F-statistic: 20.97 on 2 and 126 DF, p-value: 1.38e-08
However, zero center the data before examining interactions

```r
Garcia.centered <- data.frame(scale(Garcia, scale=FALSE))
mod2 <- lm(respappr ~ prot2 * sexism + sexism, data=Garcia.centered)
summary(mod2)
```

Call:
`lm(formula = respappr ~ prot2 * sexism + sexism, data = Garcia.centered)`

Residuals:
```
  Min  1Q  Median   3Q  Max
-3.4984 -0.7540  0.0801  0.8301  3.1853
```

Coefficients:
```
               Estimate   Std. Error  t value  Pr(>|t|)
(Intercept)  -0.011844    0.10085  -0.1170  0.906711
prot2         1.458033    0.216705  6.7280  5.52e-10 ***
sexism       0.023538     0.129264  0.1820  0.855792
prot2:sexism 0.809980     0.281910  2.8730  0.004780 **
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1 1
```

Residual standard error: 1.144 on 125 degrees of freedom
Multiple R-squared: 0.2962, Adjusted R-squared: 0.2793
Regression from the raw data

**Compare model 1 and model 2**

Test the difference between the two linear models

\[ \text{anova}(\text{mod1, mod2}) \]

Analysis of Variance Table

<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>126</td>
<td>1</td>
<td>174.54</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>125</td>
<td>1</td>
<td>10.813</td>
<td>8.2551</td>
<td>0.004776 **</td>
</tr>
</tbody>
</table>

---

Signif. codes: 0 ?***? 0.001 ?***? 0.01 ?**? 0.05 ?.
Show the regression lines by protest

Response to sexism varies as type of protest

plot(respappr ~ sexism, pch = 23, data=Garcia, main = "Response to sexism varies as type of protest", by(Garcia,Garcia$protest, function(x) abline(lm(respappr ~ sexism, data = x), lty=c("solid","dashed"), data = x), lty=c("solid","dashed")))
text(6.5,3.5,"No protest")
text(3,3.9,"Individual")
text(3,5.2,"Collective")

(See Examples: Garcia)
Regression from covariance/correlation matrices

1. Although most regression examples use the raw data, it is also possible to do this from the correlation/covariance matrices.

2. This is particularly useful for analyzing text book examples or data sets that come from synthetic covariance matrices (SAPA data).

3. Two functions do this
   3.1 setCor will find (and draw the paths) between a set of X variables and a set of Y variables from either the raw data or from a correlation matrix.

   3.2 mediate will show path diagrams in a way to highlight “mediated” (indirect) and direct effects. The significance of the indirect effect is found by bootstrapped confidence intervals

4. Both of these functions just use the standard matrix equation
   \[ \beta_{xy} = R^{-1}r_{xy} \]

5. The two examples are taken from the PMI example in Hayes (2013) which is saved as the Tal_0r dataset and used in the mediate help file.
Regression from covariance/correlation matrices

setCor finds regressions from covariances

**R code**

```r
C.pmi <- cov(Tal_Or) #just to show how we can do this
lowerMat(C.pmi) #show it
setCor(2:4,c(1,5,6),data=C.pmi)
```

Multiple Regression from matrix input

Beta weights

<table>
<thead>
<tr>
<th></th>
<th>pmi</th>
<th>import</th>
<th>reaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>cond</td>
<td>0.18</td>
<td>0.19</td>
<td>0.16</td>
</tr>
<tr>
<td>gender</td>
<td>0.00</td>
<td>-0.08</td>
<td>-0.01</td>
</tr>
<tr>
<td>age</td>
<td>-0.01</td>
<td>0.09</td>
<td>-0.09</td>
</tr>
</tbody>
</table>

Multiple R

<table>
<thead>
<tr>
<th></th>
<th>pmi</th>
<th>import</th>
<th>reaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>pmi</td>
<td>0.18</td>
<td>0.21</td>
<td>0.18</td>
</tr>
</tbody>
</table>

multiple R2

<table>
<thead>
<tr>
<th></th>
<th>pmi</th>
<th>import</th>
<th>reaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>pmi</td>
<td>0.033</td>
<td>0.043</td>
<td>0.033</td>
</tr>
</tbody>
</table>

(Specify n.obs if you want the standard errors, t, and probabilities of the estimates.)
Regressions from a covariance matrix

Regression Models

unweighted matrix correlation = 0.11
Regression from covariance/correlation matrices

A mediation example from TalOr (2010) included in Hayes (2013)

```r
mediate(y="reaction",x = "cond",m=c("pmi","import"),data=pmi,n.obs=123,n.iter=5000)
```

Call: mediate(y = "reaction", x = "cond", m = c("pmi", "import"), data = C.pmi, n.obs = 123, n.iter = 50)

The DV (Y) was reaction. The IV (X) was cond. The mediating variable(s) = pmi import.

Total Direct effect (c) of cond on reaction = 0.5 S.E. = 0.28 t direct = 1.79 with probability = 0.077

Direct effect (c') of cond on reaction removing pmi import = 0.1 S.E. = 0.24 t direct = 0.43 with probability = 0.67

Indirect effect (ab) of cond on reaction through pmi import = 0.39

Mean bootstrapped indirect effect = 0.4 with standard error = 0.13 Lower CI = 0.19 Upper CI = 0.63

R2 of model = 0.33

To see the longer output, specify short = FALSE in the print statement

Full output

Total effect estimates (c)

<table>
<thead>
<tr>
<th></th>
<th>reaction</th>
<th>se</th>
<th>t</th>
<th>Prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>cond</td>
<td>0.5</td>
<td>0.28</td>
<td>1.79</td>
<td>0.0766</td>
</tr>
</tbody>
</table>

Direct effect estimates (c')

<table>
<thead>
<tr>
<th></th>
<th>reaction</th>
<th>se</th>
<th>t</th>
<th>Prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>cond</td>
<td>0.10</td>
<td>0.24</td>
<td>0.43</td>
<td>6.66e-01</td>
</tr>
<tr>
<td>pmi</td>
<td>0.40</td>
<td>0.09</td>
<td>4.26</td>
<td>4.04e-05</td>
</tr>
<tr>
<td>import</td>
<td>0.32</td>
<td>0.07</td>
<td>4.59</td>
<td>1.13e-05</td>
</tr>
</tbody>
</table>

'a' effect estimates

<table>
<thead>
<tr>
<th></th>
<th>cond</th>
<th>se</th>
<th>t</th>
<th>Prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>pmi</td>
<td>0.48</td>
<td>0.24</td>
<td>2.02</td>
<td>0.0452</td>
</tr>
<tr>
<td>import</td>
<td>0.63</td>
<td>0.31</td>
<td>2.02</td>
<td>0.0452</td>
</tr>
</tbody>
</table>

'b' effect estimates

<table>
<thead>
<tr>
<th></th>
<th>reaction</th>
<th>se</th>
<th>t</th>
<th>Prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>pmi</td>
<td>0.40</td>
<td>0.09</td>
<td>4.26</td>
<td>4.04e-05</td>
</tr>
<tr>
<td>import</td>
<td>0.32</td>
<td>0.07</td>
<td>4.59</td>
<td>1.13e-05</td>
</tr>
</tbody>
</table>

'ab' effect estimates

<table>
<thead>
<tr>
<th></th>
<th>reaction boot</th>
<th>sd</th>
<th>lower</th>
<th>upper</th>
</tr>
</thead>
<tbody>
<tr>
<td>cond</td>
<td>0.39</td>
<td>0.4</td>
<td>0.13</td>
<td>0.63</td>
</tr>
</tbody>
</table>
A mediation example from Hayes (2013)

**Mediation model**

```
cond  reaction
pmi
0.48  c = 0.5   c' = 0.1
0.63  0.4
0.32
```

```
import
0.63  c = 0.5   c' = 0.1
0.32
```
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. Scalers (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 scalers, 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.
   
   ```
   > E.df <- data.frame(names = C, values = c(31, 28, 31, 30, 31))
   > dim(E.df)
   [1] 5 2
   ```

2. Lists are collections of what ever you want. They have length, but do not have dimensions.
   
   ```
   > F <- list(first = A, a.vector = C, a.matrix = E)
   > length(F)
   [1] 3
   ```
Show values by entering the variable name

> E.df

    names values
 1 January  31
 2 February  28
 3 March  31
 4 April  30
 5 May  31

> F

$first
[1] 1

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

$a.matrix

[1,]   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`

```r
> 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.

```r
> 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
```
Basic R

### 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 return “invisible” values
   - e.g., \(p \leftarrow \text{print}(x, \text{digits}=3)\) will print out \(x\) to 3 digits but also returns a value to \(p\).
   - Similarly, \(s \leftarrow \text{summary}(\text{some object})\) will return the value of the summary function.

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. Keep a list of the ones you use.
Objects and Functions

Getting help

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

- `file.choose()` find a file
- `file.choose(new=TRUE)` create a new file
- `read.table(filename)` reads a comma separated file
- `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,each)` 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(x)` 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

Selected functions from *psych* package

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

- **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
- **bestScales** empirical scale construction
Outline

Part I: What is R, where did it come from, why use it
  • Installing R and adding packages: the building blocks of R

Part II: A brief introduction – an overview
  • R is just a fancy (very fancy) calculator
  • Descriptive data analysis
  • Some inferential analysis

Part III R is a powerful statistical system
  • Data entry (detail and practice)
  • Descriptive (again)
  • Inferential (t and F with more practice)
  • Regression
  • Basic R commands

Part IV: Psychometrics
  • Reliability and its discontents
  • EFA, CFA, SEM

Part V: Help and More Help
  • List of useful commands

Part VI: The psych package and more practice
Outline of Part IV: Psychometrics

-> Part III: Basic Statistics

Classical Test Theory measures of reliability
  Split Half Reliability and $\alpha$
  Multiple Scales

Multivariate Analysis and Structural Equation Modeling
  Exploratory Factor Analysis
  Confirmatory Factor Analysis and Structural Equation Modeling

Item Response Theory
  Multiple programs
  IRT from factor analysis: the irt.fa function in psych

-> Part V: More help
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 estimates of reliability

1. Alternative estimates of reliability
   
   **alpha** \( \alpha \) reliability of a single scale finds the average split half reliability. (some items may be reversed keyed).

   **omega** \( \omega \) 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

2. Scoring tests with multiple scales
   
   **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.
6,435 split half reliabilities of a 16 item ability test

Split half reliabilities of 16 ability measures

R code

```r
sp <- splitHalf(ability, raw=TRUE, brute=TRUE)
hist(sp$raw, breaks=50)
```
Finding coefficient $\alpha$ for a scale (see Revelle and Zinbarg, 2009, however, for why you should not)

R code

```r
alpha(ability)
```

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>0.83</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>
<tr>
<td>rotate.4</td>
<td>1460</td>
<td>0.58</td>
<td>0.56</td>
<td>0.48</td>
<td>0.22</td>
</tr>
<tr>
<td>rotate.6</td>
<td>1456</td>
<td>0.56</td>
<td>0.53</td>
<td>0.46</td>
<td>0.31</td>
</tr>
<tr>
<td>rotate.8</td>
<td>1460</td>
<td>0.51</td>
<td>0.47</td>
<td>0.41</td>
<td>0.19</td>
</tr>
</tbody>
</table>
Using `scoreItems` to score 25 Big 5 items (see 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></th>
<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>
<td>0.6</td>
</tr>
</tbody>
</table>

Average item correlation:

<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>average.r</td>
<td>0.32</td>
<td>0.34</td>
<td>0.39</td>
<td>0.46</td>
<td>0.23</td>
</tr>
</tbody>
</table>

Guttman 6* reliability:

<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>Lambda.6</td>
<td>0.7</td>
<td>0.72</td>
<td>0.76</td>
<td>0.81</td>
<td>0.6</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></th>
<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>
<td>0.23</td>
</tr>
<tr>
<td>Conscientious</td>
<td>0.26</td>
<td>0.72</td>
<td>0.35</td>
<td>-0.305</td>
<td>0.30</td>
</tr>
<tr>
<td>Extraversion</td>
<td>0.46</td>
<td>0.26</td>
<td>0.76</td>
<td>-0.284</td>
<td>0.32</td>
</tr>
<tr>
<td>Neuroticism</td>
<td>-0.18</td>
<td>-0.23</td>
<td>-0.22</td>
<td>0.812</td>
<td>-0.12</td>
</tr>
<tr>
<td>Openness</td>
<td>0.15</td>
<td>0.19</td>
<td>0.22</td>
<td>-0.086</td>
<td>0.60</td>
</tr>
</tbody>
</table>
### score.items output, continued

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

<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>A1</td>
<td>-0.40</td>
<td>-0.06</td>
<td>-0.11</td>
<td>0.14</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>
<td>0.17</td>
</tr>
<tr>
<td>A3</td>
<td>0.70</td>
<td>0.22</td>
<td>0.48</td>
<td>-0.11</td>
<td>0.17</td>
</tr>
<tr>
<td>A4</td>
<td>0.49</td>
<td>0.29</td>
<td>0.30</td>
<td>-0.14</td>
<td>0.01</td>
</tr>
<tr>
<td>A5</td>
<td>0.62</td>
<td>0.23</td>
<td>0.55</td>
<td>-0.23</td>
<td>0.18</td>
</tr>
<tr>
<td>C1</td>
<td>0.13</td>
<td>0.53</td>
<td>0.19</td>
<td>-0.08</td>
<td>0.28</td>
</tr>
<tr>
<td>C2</td>
<td>0.21</td>
<td>0.61</td>
<td>0.17</td>
<td>0.00</td>
<td>0.20</td>
</tr>
<tr>
<td>C3</td>
<td>0.21</td>
<td>0.54</td>
<td>0.14</td>
<td>-0.09</td>
<td>0.08</td>
</tr>
<tr>
<td>C4</td>
<td>-0.24</td>
<td>-0.66</td>
<td>-0.23</td>
<td>0.31</td>
<td>-0.23</td>
</tr>
<tr>
<td>C5</td>
<td>-0.26</td>
<td>-0.59</td>
<td>-0.29</td>
<td>0.36</td>
<td>-0.10</td>
</tr>
<tr>
<td>E1</td>
<td>-0.30</td>
<td>-0.06</td>
<td>-0.59</td>
<td>0.11</td>
<td>-0.16</td>
</tr>
<tr>
<td>E2</td>
<td>-0.39</td>
<td>-0.25</td>
<td>-0.70</td>
<td>0.34</td>
<td>-0.15</td>
</tr>
<tr>
<td>E3</td>
<td>0.44</td>
<td>0.20</td>
<td>0.60</td>
<td>-0.10</td>
<td>0.37</td>
</tr>
<tr>
<td>E4</td>
<td>0.51</td>
<td>0.23</td>
<td>0.68</td>
<td>-0.22</td>
<td>0.04</td>
</tr>
<tr>
<td>E5</td>
<td>0.34</td>
<td>0.40</td>
<td>0.55</td>
<td>-0.10</td>
<td>0.31</td>
</tr>
<tr>
<td>N1</td>
<td>-0.22</td>
<td>-0.21</td>
<td>-0.11</td>
<td>0.76</td>
<td>-0.12</td>
</tr>
<tr>
<td>N2</td>
<td>-0.22</td>
<td>-0.19</td>
<td>-0.12</td>
<td>0.74</td>
<td>-0.06</td>
</tr>
<tr>
<td>N3</td>
<td>-0.14</td>
<td>-0.20</td>
<td>-0.14</td>
<td>0.74</td>
<td>-0.03</td>
</tr>
<tr>
<td>N4</td>
<td>-0.22</td>
<td>-0.30</td>
<td>-0.39</td>
<td>0.62</td>
<td>-0.02</td>
</tr>
<tr>
<td>N5</td>
<td>-0.04</td>
<td>-0.14</td>
<td>-0.19</td>
<td>0.55</td>
<td>-0.18</td>
</tr>
<tr>
<td>O1</td>
<td>0.16</td>
<td>0.20</td>
<td>0.31</td>
<td>-0.09</td>
<td>0.52</td>
</tr>
<tr>
<td>O2</td>
<td>-0.01</td>
<td>-0.18</td>
<td>-0.07</td>
<td>0.19</td>
<td>-0.45</td>
</tr>
<tr>
<td>O3</td>
<td>0.26</td>
<td>0.20</td>
<td>0.42</td>
<td>-0.07</td>
<td>0.61</td>
</tr>
<tr>
<td>O4</td>
<td>0.06</td>
<td>-0.02</td>
<td>-0.10</td>
<td>0.21</td>
<td>0.32</td>
</tr>
<tr>
<td>O5</td>
<td>-0.09</td>
<td>-0.14</td>
<td>-0.11</td>
<td>0.11</td>
<td>-0.53</td>
</tr>
<tr>
<td>gender</td>
<td>0.25</td>
<td>0.11</td>
<td>0.12</td>
<td>0.14</td>
<td>-0.07</td>
</tr>
<tr>
<td>education</td>
<td>0.06</td>
<td>0.03</td>
<td>0.01</td>
<td>-0.06</td>
<td>0.13</td>
</tr>
<tr>
<td>age</td>
<td>0.22</td>
<td>0.14</td>
<td>0.07</td>
<td>-0.13</td>
<td>0.10</td>
</tr>
</tbody>
</table>
Correlations of composite scores based upon item correlations

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

Correlations of composite scales

<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>0.25</td>
<td>0.47</td>
<td>-0.18</td>
<td>0.16</td>
</tr>
<tr>
<td>Conscientious</td>
<td>0.25</td>
<td>1</td>
<td>0.27</td>
<td>-0.22</td>
<td>0.2</td>
</tr>
<tr>
<td>Extraversion</td>
<td>0.47</td>
<td>0.27</td>
<td>1</td>
<td>-0.22</td>
<td>0.24</td>
</tr>
<tr>
<td>Neuroticism</td>
<td>-0.18</td>
<td>-0.22</td>
<td>-0.22</td>
<td>1</td>
<td>-0.07</td>
</tr>
<tr>
<td>Openness</td>
<td>0.16</td>
<td>0.2</td>
<td>0.24</td>
<td>-0.07</td>
<td>1</td>
</tr>
</tbody>
</table>
Multiple Scales

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

cor.plot.upperLowerCi(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>0.21</td>
<td>0.44</td>
<td>-0.14</td>
<td>0.13</td>
</tr>
<tr>
<td>Conscientious</td>
<td>0.29</td>
<td>1</td>
<td>0.22</td>
<td>-0.18</td>
<td>0.16</td>
</tr>
<tr>
<td>Extraversion</td>
<td>0.51</td>
<td>0.31</td>
<td>1</td>
<td>-0.18</td>
<td>0.2</td>
</tr>
<tr>
<td>Neuroticism</td>
<td>-0.22</td>
<td>-0.26</td>
<td>-0.25</td>
<td>1</td>
<td>-0.03</td>
</tr>
<tr>
<td>Openness</td>
<td>0.2</td>
<td>0.24</td>
<td>0.27</td>
<td>-0.12</td>
<td>1</td>
</tr>
</tbody>
</table>
Exploratory Factor Analysis

Factor analysis of Thurstone 9 variable problem

R code

f3 <- fa(Thurstone, nfactors=3)  # use this built in dataset
f3  # we keep the output as an object to use later

Factor Analysis using method = minres
Call: fa(r = Thurstone, nfactors = 3)
Standardized loadings (pattern matrix) based upon correlation matrix

<table>
<thead>
<tr>
<th></th>
<th>MR1</th>
<th>MR2</th>
<th>MR3</th>
<th>h2</th>
<th>u2</th>
<th>com</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sentences</td>
<td>0.91</td>
<td>-0.04</td>
<td>0.04</td>
<td>0.82</td>
<td>0.18</td>
<td>1.0</td>
</tr>
<tr>
<td>Vocabulary</td>
<td>0.89</td>
<td>0.06</td>
<td>-0.03</td>
<td>0.84</td>
<td>0.16</td>
<td>1.0</td>
</tr>
<tr>
<td>Sent.Completion</td>
<td>0.83</td>
<td>0.04</td>
<td>0.00</td>
<td>0.73</td>
<td>0.27</td>
<td>1.0</td>
</tr>
<tr>
<td>First.Letters</td>
<td>0.00</td>
<td>0.86</td>
<td>0.00</td>
<td>0.73</td>
<td>0.27</td>
<td>1.0</td>
</tr>
<tr>
<td>4.Letter.Words</td>
<td>-0.01</td>
<td>0.74</td>
<td>0.10</td>
<td>0.63</td>
<td>0.37</td>
<td>1.0</td>
</tr>
<tr>
<td>Suffixes</td>
<td>0.18</td>
<td>0.63</td>
<td>-0.08</td>
<td>0.50</td>
<td>0.50</td>
<td>1.2</td>
</tr>
<tr>
<td>Letter.Series</td>
<td>0.03</td>
<td>-0.01</td>
<td>0.84</td>
<td>0.72</td>
<td>0.28</td>
<td>1.0</td>
</tr>
<tr>
<td>Pedigrees</td>
<td>0.37</td>
<td>-0.05</td>
<td>0.47</td>
<td>0.50</td>
<td>0.50</td>
<td>1.9</td>
</tr>
<tr>
<td>Letter.Group</td>
<td>-0.06</td>
<td>0.21</td>
<td>0.64</td>
<td>0.53</td>
<td>0.47</td>
<td>1.2</td>
</tr>
</tbody>
</table>

SS loadings

<table>
<thead>
<tr>
<th></th>
<th>MR1</th>
<th>MR2</th>
<th>MR3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sentences</td>
<td>2.64</td>
<td>1.86</td>
<td>1.50</td>
</tr>
<tr>
<td>Vocabulary</td>
<td>0.29</td>
<td>0.21</td>
<td>0.17</td>
</tr>
<tr>
<td>Sent.Completion</td>
<td>0.29</td>
<td>0.50</td>
<td>0.67</td>
</tr>
<tr>
<td>First.Letters</td>
<td>0.44</td>
<td>0.31</td>
<td>0.25</td>
</tr>
<tr>
<td>4.Letter.Words</td>
<td>0.44</td>
<td>0.75</td>
<td>1.00</td>
</tr>
</tbody>
</table>

With factor correlations of

<table>
<thead>
<tr>
<th></th>
<th>MR1</th>
<th>MR2</th>
<th>MR3</th>
</tr>
</thead>
<tbody>
<tr>
<td>MR1</td>
<td>1.00</td>
<td>0.59</td>
<td>0.54</td>
</tr>
<tr>
<td>MR2</td>
<td>0.59</td>
<td>1.00</td>
<td>0.52</td>
</tr>
<tr>
<td>MR3</td>
<td>0.54</td>
<td>0.52</td>
<td>1.00</td>
</tr>
</tbody>
</table>
Exploratory Factor Analysis

Factor analysis output, continued

With factor correlations of

<table>
<thead>
<tr>
<th></th>
<th>MR1</th>
<th>MR2</th>
<th>MR3</th>
</tr>
</thead>
<tbody>
<tr>
<td>MR1</td>
<td>1.00</td>
<td>0.59</td>
<td>0.54</td>
</tr>
<tr>
<td>MR2</td>
<td>0.59</td>
<td>1.00</td>
<td>0.52</td>
</tr>
<tr>
<td>MR3</td>
<td>0.54</td>
<td>0.52</td>
<td>1.00</td>
</tr>
</tbody>
</table>

Mean item complexity = 1.2
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
The degrees of freedom for the model are 12 and the objective function was 0.01

The root mean square of the residuals (RMSR) is 0.01
The df corrected root mean square of the residuals is 0.01

Fit based upon off diagonal values = 1
Measures of factor score adequacy

Correlation of scores with factors

<table>
<thead>
<tr>
<th></th>
<th>MR1</th>
<th>MR2</th>
<th>MR3</th>
</tr>
</thead>
<tbody>
<tr>
<td>MR1</td>
<td>0.96</td>
<td>0.92</td>
<td>0.90</td>
</tr>
<tr>
<td>MR2</td>
<td>0.93</td>
<td>0.85</td>
<td>0.81</td>
</tr>
<tr>
<td>MR3</td>
<td>0.86</td>
<td>0.71</td>
<td>0.63</td>
</tr>
</tbody>
</table>
Part III — CTT \( \alpha \omega_h \omega_t \) — EFA, CFA, SEM — IRT — V

Exploratory Factor Analysis

**Bootstrapped confidence intervals**

R code

```r
da(Thurstone,3, n.obs=213, n.iter=20) #to do bootstrapping```

...  

Coefficients and bootstrapped confidence intervals

<table>
<thead>
<tr>
<th></th>
<th>low</th>
<th>MR1 upper</th>
<th>low</th>
<th>MR2 upper</th>
<th>low</th>
<th>MR3 upper</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sentences</td>
<td>0.83</td>
<td>0.91</td>
<td>0.97</td>
<td>-0.10</td>
<td>-0.04</td>
<td>0.06</td>
</tr>
<tr>
<td>Vocabulary</td>
<td>0.80</td>
<td>0.89</td>
<td>0.98</td>
<td>0.00</td>
<td>0.06</td>
<td>0.15</td>
</tr>
<tr>
<td>Sent.Completion</td>
<td>0.75</td>
<td>0.83</td>
<td>0.90</td>
<td>-0.05</td>
<td>0.04</td>
<td>0.11</td>
</tr>
<tr>
<td>First.Letters</td>
<td>-0.08</td>
<td>0.00</td>
<td>0.09</td>
<td>0.68</td>
<td>0.86</td>
<td>0.97</td>
</tr>
<tr>
<td>4.Letter.Words</td>
<td>-0.13</td>
<td>-0.01</td>
<td>0.12</td>
<td>0.57</td>
<td>0.74</td>
<td>0.90</td>
</tr>
<tr>
<td>Suffixes</td>
<td>0.07</td>
<td>0.18</td>
<td>0.26</td>
<td>0.50</td>
<td>0.63</td>
<td>0.76</td>
</tr>
<tr>
<td>Letter.Series</td>
<td>-0.09</td>
<td>0.03</td>
<td>0.13</td>
<td>-0.06</td>
<td>-0.01</td>
<td>0.08</td>
</tr>
<tr>
<td>Pedigrees</td>
<td>0.27</td>
<td>0.37</td>
<td>0.52</td>
<td>-0.17</td>
<td>-0.05</td>
<td>0.04</td>
</tr>
<tr>
<td>Letter.Group</td>
<td>-0.16</td>
<td>-0.06</td>
<td>0.08</td>
<td>0.12</td>
<td>0.21</td>
<td>0.29</td>
</tr>
</tbody>
</table>

Interfactor correlations and bootstrapped confidence intervals

<table>
<thead>
<tr>
<th></th>
<th>lower estimate</th>
<th>upper</th>
</tr>
</thead>
<tbody>
<tr>
<td>MR1—MR2</td>
<td>0.47</td>
<td>0.59</td>
</tr>
<tr>
<td>MR1—MR3</td>
<td>0.39</td>
<td>0.54</td>
</tr>
<tr>
<td>MR2—MR3</td>
<td>0.30</td>
<td>0.52</td>
</tr>
</tbody>
</table>
Exploratory Factor Analysis

The simple factor structure

\texttt{factor.diagami(f3)} # show the diagram

Factor Analysis

```
\begin{center}
\begin{tikzpicture}
\node (MR1) at (0,0) {MR1};
\node (Sentences) at (-2,-1) {Sentences};
\node (Vocabulary) at (-2,-2) {Vocabulary};
\node (Sent.Completion) at (-2,-3) {Sent.Completion};
\node (First.Leters) at (-2,-4) {First.Leters};
\node (4.Letter.Words) at (-2,-5) {4.Letter.Words};
\node (Suffixes) at (-2,-6) {Suffixes};
\node (Letter.Series) at (-2,-7) {Letter.Series};
\node (Letter.Group) at (-2,-8) {Letter.Group};
\node (Pedigrees) at (-2,-9) {Pedigrees};
\draw (MR1) -- (Sentences) node[midway, above] {0.9};
\draw (MR1) -- (Vocabulary) node[midway, above] {0.9};
\draw (MR1) -- (Sent.Completion) node[midway, above] {0.8};
\draw (Sent.Completion) -- (MR2) node[midway, above] {0.6};
\draw (MR2) -- (4.Letter.Words) node[midway, above] {0.7};
\draw (MR2) -- (Suffixes) node[midway, above] {0.6};
\draw (4.Letter.Words) -- (MR3) node[midway, above] {0.5};
\draw (Suffixes) -- (MR3) node[midway, above] {0.5};
\draw (MR3) -- (Letter.Series) node[midway, above] {0.8};
\draw (Letter.Series) -- (MR4) node[midway, above] {0.5};
\draw (MR4) -- (Letter.Group) node[midway, above] {0.6};
\end{tikzpicture}
\end{center}
```
Exploratory Factor Analysis

Two ways of viewing the higher order structure

\[ \text{om} \leftarrow \text{omega}(\text{Thurstone}) \]

\[ \text{omega}.\text{diagram}(\text{om}, \text{sl} = \text{FALSE}) \]
A hierarchical cluster structure found by iclust

iclust(Thurstone)
Confirmatory Factor Analysis and Structural Equation Modeling

Structural Equation modeling packages

1. **sem** (Fox, Nie & Byrnes, 2013)
   - uses RAM notation

2. **lavaan** (Rosseel, 2012)
   - Mimics as much as possible MPLUS output
   - Allows for multiple groups
   - Easy syntax

   - Open source and R version of Mx
   - Allows for multiple groups (and almost anything else)
   - Complicated syntax
Multiple programs

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

[Graph showing Item Response Information curves for 16 ability items from ICAR]
IRT from factor analysis: the irt.fa function in psych

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

- `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, each) 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><code>mean(x, na.rm=TRUE)</code></td>
<td>Mean with NA removal</td>
</tr>
<tr>
<td><code>is.na(x)</code></td>
<td>Checks for NA values</td>
</tr>
<tr>
<td><code>na.omit(x)</code></td>
<td>Removes NA values</td>
</tr>
<tr>
<td><code>sum(x)</code></td>
<td>Sum of elements</td>
</tr>
<tr>
<td><code>rowSums(x)</code></td>
<td>Row sums</td>
</tr>
<tr>
<td><code>colSums(x)</code></td>
<td>Column sums</td>
</tr>
<tr>
<td><code>min(x, na.rm=TRUE)</code></td>
<td>Minimum with NA removal</td>
</tr>
<tr>
<td><code>max(x)</code></td>
<td>Maximum ignores NA values</td>
</tr>
<tr>
<td><code>range(x)</code></td>
<td>Range of values</td>
</tr>
<tr>
<td><code>table(x)</code></td>
<td>Frequency table</td>
</tr>
<tr>
<td><code>summary(x)</code></td>
<td>Summary of data</td>
</tr>
<tr>
<td><code>sd(x)</code></td>
<td>Standard deviation</td>
</tr>
<tr>
<td><code>cor(x, use=&quot;pairwise&quot;)</code></td>
<td>Correlation</td>
</tr>
<tr>
<td><code>cov(x)</code></td>
<td>Covariance</td>
</tr>
<tr>
<td><code>solve(x)</code></td>
<td>Inverse of matrix</td>
</tr>
<tr>
<td><code>lm(y~x)</code></td>
<td>Linear model</td>
</tr>
<tr>
<td><code>aov(y~x)</code></td>
<td>ANOVA</td>
</tr>
<tr>
<td><code>describe(x)</code></td>
<td>Descriptive statistics</td>
</tr>
<tr>
<td><code>describeBy(x, y)</code></td>
<td>Descriptives by group</td>
</tr>
<tr>
<td><code>pairs.panels(x)</code></td>
<td>Scatterplot matrices</td>
</tr>
<tr>
<td><code>error.bars(x)</code></td>
<td>Error bars</td>
</tr>
<tr>
<td><code>error.bars.by(x)</code></td>
<td>Error bars by groups</td>
</tr>
<tr>
<td><code>fa(x, n)</code></td>
<td>Factor analysis</td>
</tr>
<tr>
<td><code>principal(x, n)</code></td>
<td>Principal components</td>
</tr>
<tr>
<td><code>iclust(x)</code></td>
<td>Item cluster analysis</td>
</tr>
<tr>
<td><code>scoreItems(x)</code></td>
<td>Score multiple scales</td>
</tr>
<tr>
<td><code>score.multiple.choice(x)</code></td>
<td>Score multiple choice scales</td>
</tr>
<tr>
<td><code>alpha(x)</code></td>
<td>Cronbach's alpha</td>
</tr>
<tr>
<td><code>omega(x)</code></td>
<td>MacDonald's omega</td>
</tr>
<tr>
<td><code>irt.fa(x)</code></td>
<td>Item response theory through factor analysis</td>
</tr>
<tr>
<td><code>mediate(y, x, m, data)</code></td>
<td>Mediation/moderation</td>
</tr>
</tbody>
</table>
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 listserv, StackOverflow, your students and colleagues
An introduction to the psych package
Outline

Part I: What is R, where did it come from, why use it
  • Installing R and adding packages: the building blocks of R

Part II: A brief introduction – an overview
  • R is just a fancy (very fancy) calculator
  • Descriptive data analysis
  • Some inferential analysis

Part III R is a powerful statistical system
  • Data entry (detail and practice)
  • Descriptive (again)
  • Inferential (t and F with more practice)
  • Regression
  • Basic R commands

Part IV: Psychometrics
  • Reliability and its discontents
  • EFA, CFA, SEM

Part V: Help and More Help
  • List of useful commands

Part VI: The psych package and more practice
The psych package

1. Developed at NU over the past 12 years to make using R easier for psychologists
2. Basically does the kind of statistics that my students and I find useful for personality, motivation and cognitive psychology
3. Available at CRAN for PCs and Macs
4. Development version (for Macs) is always available at the http://personality-project.org/r repository.
5. Bugs are fixed and new versions with new toys (functions) are released about every 4-6 months.
6. Version number reflects the year and month of release (1.8.4)
7. Has several vignettes to describe what it does:
   • http://personality-project.org/r/psych/vignettes/intro.pdf An introduction
   • http://personality-project.org/r/psych/vignettes/overview.pdf An overview
   • http://personality-project.org/r/psych/vignettes/psych_for_sem.pdf as a front end to doing sem
Show all the functions in the psych package

objects("package:psych")

objects("package:psych")
[1] "%+%"    "ability"     "affect"          "all.income"
[5] "alpha"   "anova.psych" "autoR"           "Bechtoldt"
[49] "cohen.kappa"  "comorbidity" "con2cat"      "congeneric.s"
[53] "cor.ci"      "cor.plot"    "cor.plot.upperLowerCi" "cor.smooth"
[81] "cushny"      "d2r"         "densityBy"    "describe"
[109] "epi.dictionary" "equamax"     "error.bars"    "error.bars.l"
[177] "ICC2latex"   "iclust"      "ICLUST"        "ICLUST.cluster"
[201] "irt.fa"      "irt.item.diff.rasch" "irt.person.rasch" "irt.responses"
[241] "mixed.cor"   "mixedCor"    "mlArrange"    "mlPlot"
[253] "omega"       "omega.diagram" "omega.graph"   "omega2latex"
[309] "read.clipboard.upper" "read.file"   "read.file.csv" "read.https"
[329] "score.alpha"  "score.irt"    "score.irt.2"   "score.irt.poly"
[333] "score.items"  "score.multiple.choice" "scoreFast"  "scoreIrt"
[405] "Thurstone"   "Thurstone.33" "topBottom"    "tr"
[409] "Tucker"      "unidim"      "varimin"       "veg"
Getting and cleaning data

**Get your data:** using `read.file` or `read.clipboard`

**From a website:** define the file name

```r
fn <- "http://personality-project.org/r/datasets/Maps.mixx.msQ1.epi.bf.txt"
fn # show it to check
[1] "http://personality-project.org/r/datasets/Maps.mixx.msQ1.epi.bf.txt"
mydata <- read.file(fn, header=TRUE)
```

**From a local file:** find the file using `read.file`

```r
> my.data <- read.file() # will open a search window, read the file
# depending upon the suffix, will read .sav, .csv, .txt, .rds, .rDa, etc.
```

**From the clipboard:** (first, go to the remote site, copy to the clipboard and then use the `read.clipboard` function).

```r
mydata <- read.clipboard() # or
mydata <- read.clipboard.tab() # if an excel file
my.data <- read.clipboard.csv() # if a tab delimited file
```
Getting and cleaning data

Checking the data using `describe`

```r
> dim(mydata)
[1] 231  86
> describe(mydata)

   vars  n  mean   sd median trimmed mad min max range  skew  kurtosis    se
id       1 231 66.82 45.13   58  64.14    50  1 160 159   0.45   -0.96  2.97
delighted 2 231  0.82  1.05    1   0.68    1.48  0  9   2.46     14.15  0.07
sociable  3 231  1.32  0.96    1  1.28    1.48  0  3   3.06    -1.03  0.06
jittery    4 231  0.55  0.78    0  0.39    0.00  0  3    3.37    1.23  0.05
hostile    5 231  0.35  0.85    0  0.17    0.00  0  9    9.34  45.21  0.06
sluggish   6 231  1.21  0.96    1  1.14    1.48  0  3    3.47    -0.70  0.06
depressed  7 231  0.56  0.83    0  0.39    0.00  0  3    3.15    1.37  0.05
  ... ashamed  71 231  0.32  1.15    0  0.06    0.00  0  9    9.52  40.25  0.08
anxious    72 231  0.75  1.26    0  0.53    0.00  0  9    3.85  21.39  0.08
idle       73 231  0.98  1.15    1  0.83    1.48  0  9    9.31  18.20  0.08
epiE       74 231 13.33  4.14    14 13.49    4.45  1 22    21-0.33  -0.06  0.27
epiS       75 231  7.58  2.69    8  7.77    2.97  0 13    13-0.57  -0.02  0.18
epiImp     76 231  4.37  1.88    4  4.36    1.48  0  9    9.06   -0.62  0.12
epile      77 231  2.38  1.50    2  2.27    1.48  0  7    7.66    0.24  0.10
  ... traitanx 85 231 39.01  9.52    38 38.36    8.90  22 71    49.67     0.47  0.63
stateanx   86 231 39.85  11.48   38 38.92    10.38  21 79    58.72    -0.01  0.76
```
Getting and cleaning data

**Cleaning the data using** `scrub`

We want to change 9s in variables 2 - 73 into NA

```r
> cleaned <- scrub(mydata,where=2:73,isvalue=9,newvalue=NA)
> describe(cleaned)

<table>
<thead>
<tr>
<th>vars</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>id</td>
<td>1</td>
<td>231</td>
<td>66.82</td>
<td>45.13</td>
<td>58.0</td>
<td>64.14</td>
<td>1</td>
<td>160</td>
<td>159</td>
<td>0.45</td>
<td>-0.96</td>
<td>2.97</td>
</tr>
<tr>
<td>delighted</td>
<td>2</td>
<td>230</td>
<td>0.78</td>
<td>0.90</td>
<td>1.0</td>
<td>0.67</td>
<td>1.48</td>
<td>0</td>
<td>3</td>
<td>3.079</td>
<td>-0.52</td>
<td>0.06</td>
</tr>
<tr>
<td>sociable</td>
<td>3</td>
<td>231</td>
<td>1.32</td>
<td>0.96</td>
<td>1.0</td>
<td>1.28</td>
<td>1.48</td>
<td>0</td>
<td>3</td>
<td>3.06</td>
<td>-1.03</td>
<td>0.06</td>
</tr>
<tr>
<td>jittery</td>
<td>4</td>
<td>231</td>
<td>0.55</td>
<td>0.78</td>
<td>0.0</td>
<td>0.39</td>
<td>0.00</td>
<td>0</td>
<td>3</td>
<td>3.137</td>
<td>1.23</td>
<td>0.05</td>
</tr>
<tr>
<td>hostile</td>
<td>5</td>
<td>230</td>
<td>0.31</td>
<td>0.63</td>
<td>0.0</td>
<td>0.17</td>
<td>0.00</td>
<td>0</td>
<td>3</td>
<td>3.212</td>
<td>4.19</td>
<td>0.04</td>
</tr>
<tr>
<td>sluggish</td>
<td>6</td>
<td>231</td>
<td>1.21</td>
<td>0.96</td>
<td>1.0</td>
<td>1.14</td>
<td>1.48</td>
<td>0</td>
<td>3</td>
<td>3.047</td>
<td>-0.70</td>
<td>0.06</td>
</tr>
<tr>
<td>sluggish</td>
<td>7</td>
<td>231</td>
<td>0.56</td>
<td>0.83</td>
<td>0.0</td>
<td>0.39</td>
<td>0.00</td>
<td>0</td>
<td>3</td>
<td>3.145</td>
<td>1.37</td>
<td>0.05</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>
</tr>
<tr>
<td>ashamed</td>
<td>71</td>
<td>228</td>
<td>0.21</td>
<td>0.57</td>
<td>0.0</td>
<td>0.05</td>
<td>0.00</td>
<td>0</td>
<td>3</td>
<td>3.00</td>
<td>8.95</td>
<td>0.04</td>
</tr>
<tr>
<td>anxious</td>
<td>72</td>
<td>228</td>
<td>0.64</td>
<td>0.84</td>
<td>0.0</td>
<td>0.51</td>
<td>0.00</td>
<td>0</td>
<td>3</td>
<td>3.111</td>
<td>0.32</td>
<td>0.06</td>
</tr>
<tr>
<td>idle</td>
<td>73</td>
<td>229</td>
<td>0.91</td>
<td>0.88</td>
<td>1.0</td>
<td>0.82</td>
<td>1.48</td>
<td>0</td>
<td>3</td>
<td>3.064</td>
<td>-0.42</td>
<td>0.06</td>
</tr>
<tr>
<td>epiE</td>
<td>74</td>
<td>231</td>
<td>13.33</td>
<td>4.14</td>
<td>14.0</td>
<td>13.49</td>
<td>4.45</td>
<td>1</td>
<td>22</td>
<td>21</td>
<td>-0.33</td>
<td>-0.06</td>
</tr>
<tr>
<td>epiS</td>
<td>75</td>
<td>231</td>
<td>7.58</td>
<td>2.69</td>
<td>8.0</td>
<td>7.77</td>
<td>2.97</td>
<td>0</td>
<td>13</td>
<td>13</td>
<td>-0.57</td>
<td>-0.02</td>
</tr>
<tr>
<td>epiImp</td>
<td>76</td>
<td>231</td>
<td>4.37</td>
<td>1.88</td>
<td>4.0</td>
<td>4.36</td>
<td>1.48</td>
<td>0</td>
<td>9</td>
<td>9</td>
<td>0.06</td>
<td>-0.62</td>
</tr>
<tr>
<td>epilie</td>
<td>77</td>
<td>231</td>
<td>2.38</td>
<td>1.50</td>
<td>2.0</td>
<td>2.27</td>
<td>1.48</td>
<td>0</td>
<td>7</td>
<td>7</td>
<td>0.66</td>
<td>0.24</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>
</tr>
<tr>
<td>traitanx</td>
<td>85</td>
<td>231</td>
<td>39.01</td>
<td>9.52</td>
<td>38.0</td>
<td>38.36</td>
<td>8.90</td>
<td>22</td>
<td>71</td>
<td>49</td>
<td>0.67</td>
<td>0.47</td>
</tr>
<tr>
<td>stateanx</td>
<td>86</td>
<td>231</td>
<td>39.85</td>
<td>11.48</td>
<td>38.0</td>
<td>38.92</td>
<td>10.38</td>
<td>21</td>
<td>79</td>
<td>58</td>
<td>0.72</td>
<td>-0.01</td>
</tr>
</tbody>
</table>
```
Multiple ways to graphically display data

1. boxplots (Core R)
2. Violin plots (violinBy in psych)
3. Scatter Plot Matrix (SPLOM) plots (pairs.panels in psych)

First, let's just make a smaller data.frame and then issue two different graphic commands.

```r
my.scales <- cleaned[74:86]
violinBy(my.scales[1:5])
pairs.panels(my.scales[c(1,4,5:10)], gap=0, pch=".")
```
Graphical displays

Violin Plot `violinBy(my.scales[1:5])`
Scatter Plot of Matrices (SPLOM) of select variables

```r
pairs.panels(my.scales[c(1,4,5:10)], gap=0, pch=".")
```
Descriptive and inferential statistics

Scores and Reliability $\beta, \omega$

FA and beyond

Mediation IRT multilevel

Graphical displays

**Scatter Plot of Matrices (SPLOM) of select variables**

```
pairs.panels(my.scales[,c(1,4,5:10)], gap=0, pch=".", smoother=TRUE)
```
Show a table of correlations

R code

```r
R <- lowerCor(my.scales[c(1,4,5:10)])
cor.plot(R,numbers=TRUE)
```

<table>
<thead>
<tr>
<th>epiE</th>
<th>epili</th>
<th>epiNr</th>
<th>bfagr</th>
<th>bfcon</th>
<th>bfext</th>
<th>bfner</th>
<th>bfopen</th>
</tr>
</thead>
<tbody>
<tr>
<td>epiE</td>
<td>1.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>epili</td>
<td>-0.22</td>
<td>1.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>epiNeur</td>
<td>-0.18</td>
<td>-0.25</td>
<td>1.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bfagree</td>
<td>0.18</td>
<td>0.17</td>
<td>-0.08</td>
<td>1.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bfcon</td>
<td>-0.11</td>
<td>0.23</td>
<td>-0.13</td>
<td>0.45</td>
<td>1.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bfext</td>
<td>0.54</td>
<td>-0.04</td>
<td>-0.17</td>
<td>0.48</td>
<td>0.27</td>
<td>1.00</td>
<td></td>
</tr>
<tr>
<td>bfneur</td>
<td>-0.09</td>
<td>-0.22</td>
<td>0.63</td>
<td>-0.04</td>
<td>0.04</td>
<td>0.04</td>
<td>1.00</td>
</tr>
<tr>
<td>bfopen</td>
<td>0.14</td>
<td>-0.03</td>
<td>0.09</td>
<td>0.39</td>
<td>0.31</td>
<td>0.46</td>
<td>0.29</td>
</tr>
</tbody>
</table>

Automatically calls the `cor` and `round` functions with default parameters and then does a pretty print out using `lowerMat`. Invisibly returns the full (square) matrix of unrounded values.
A simple heat map using `cor.plot`

Correlation plot

---

R code:
```
cor.plot(epi.bfi[c(1,4:9)], numbers=TRUE, upper=FALSE, scale=FALSE)
```
Graphical displays

Scale the correlations using `cor.plot`

R code:
```
cor.plot(epi.bfi[c(1,4:9)], numbers=TRUE, upper=FALSE)
```
Show the whole matrix \texttt{cor.plot}

```
R code
cor.plot(epi.bfi[c(1,4:9)], numbers=TRUE, diag=FALSE)
```
Some inferential statistics – testing correlations

**Testing and displaying the “significance” of a set of correlations**

1. Normal theory `corr.test`
   - raw probabilities as well as with a Holm adjusted for multiple correlations

2. Display these with `cor.plot`

3. Boot strapped confidence intervals based significance using `cor.ci`
   - Graphic displays correlations scaled by “significance”
   - Graphic displays of probability of correlation using `plot.cor.upperLowerCi`

**Code for the next slides**

```r
my.scales <- epi.bfi[c(1,4:10)]
corr.test(my.scales)
ci <- cor.ci(my.scales)
cor.plot.upperLowerCi(ci)
```
Normal theory test of correlations using `corr.test`

R code

```r
> corr.test(my.scales)
Call: corr.test(x = my.scales)
Correlation matrix
                  epiE  epilie epiNeur bfagree bfcon bfext bfneur bfopen
epiE     1.00 -0.22 -0.18  0.18 -0.11  0.54 -0.09  0.14
epilie  -0.22  1.00 -0.25  0.17  0.23 -0.04 -0.22 -0.03
epiNeur -0.18 -0.25  1.00 -0.08 -0.13 -0.17  0.63  0.09
bfagree  0.18  0.17 -0.08  1.00  0.45  0.48 -0.04  0.39
bfcon   -0.11  0.23 -0.13  0.45  1.00  0.27  0.04  0.31
bfext   0.54 -0.04 -0.17  0.48  0.27  1.00  0.04  0.46
bfneur  -0.09 -0.22  0.63 -0.04  0.04  0.04  1.00  0.29
bfopen  0.14 -0.03  0.09  0.39  0.31  0.46  0.29  1.00

Sample Size
[1] 231
Probability values (Entries above the diagonal are adjusted for multiple tests.)

```
epiE  epilie epiNeur bfagree bfcon bfext bfneur bfopen
epiE 0.00 0.01 0.11  0.11  0.75  0.00  1.00  0.4
epilie 0.00 0.00 0.00  0.12  0.01  1.00  0.01  1.0
epiNeur 0.01 0.00 0.00  1.00  0.43  0.12  0.00  1.0
bfagree 0.01 0.01 0.21  0.00  0.00  0.00  1.00  0.0
bfcon  0.08 0.00 0.04  0.00  0.00  0.00  1.00  0.0
bfext  0.00 0.50 0.01  0.00  0.00  0.00  1.00  0.0
bfneur 0.15 0.00 0.00  0.50  0.50  0.57  0.00  0.0
bfopen 0.04 0.70 0.19  0.00  0.00  0.00  0.00  0.0

To see confidence intervals of the correlations, print with the short=FALSE option
Some inferential statistics – testing correlations

Heat map scaled by “significance” using `cor.ci`

```
corPlot(ci,numbers=TRUE)
```

R code
Some inferential statistics – testing correlations

Heat map scaled by “significance” and showing magic asterisks

P values above the diagonal are Holm corrected

R code
```
corPlot(my.scales, stars=TRUE, numbers=TRUE, xlas=3, main="P values above the diagonal are Holm corrected")
```
Some inferential statistics – testing correlations

Heat map scaled by “significance” and showing confidence intervals

Upper and lower confidence intervals of correlations

<table>
<thead>
<tr>
<th></th>
<th>epiE</th>
<th>epilie</th>
<th>epiNeur</th>
<th>bfagree</th>
<th>bfcon</th>
<th>bfext</th>
<th>bfneur</th>
<th>bfopen</th>
</tr>
</thead>
<tbody>
<tr>
<td>epiE</td>
<td>1</td>
<td>-0.34</td>
<td>-0.27</td>
<td>0.29</td>
<td>-0.24</td>
<td>0.64</td>
<td>-0.22</td>
<td>0.27</td>
</tr>
<tr>
<td>epilie</td>
<td>-0.1</td>
<td>1</td>
<td>-0.37</td>
<td>0.29</td>
<td>0.35</td>
<td>-0.17</td>
<td>-0.33</td>
<td>-0.14</td>
</tr>
<tr>
<td>epiNeur</td>
<td>-0.06</td>
<td>-0.11</td>
<td>1</td>
<td>-0.19</td>
<td>-0.24</td>
<td>-0.28</td>
<td>0.69</td>
<td>0.24</td>
</tr>
<tr>
<td>bfagree</td>
<td>0.05</td>
<td>0.06</td>
<td>0.04</td>
<td>1</td>
<td>0.55</td>
<td>0.58</td>
<td>-0.18</td>
<td>0.48</td>
</tr>
<tr>
<td>bfcon</td>
<td>0</td>
<td>0.1</td>
<td>-0.01</td>
<td>0.33</td>
<td>1</td>
<td>0.38</td>
<td>0.17</td>
<td>0.41</td>
</tr>
<tr>
<td>bfext</td>
<td>0.44</td>
<td>0.09</td>
<td>-0.06</td>
<td>0.37</td>
<td>0.13</td>
<td>1</td>
<td>0.16</td>
<td>0.57</td>
</tr>
<tr>
<td>bfneur</td>
<td>0.04</td>
<td>-0.11</td>
<td>0.56</td>
<td>0.09</td>
<td>-0.09</td>
<td>0.08</td>
<td>1</td>
<td>0.42</td>
</tr>
<tr>
<td>bfopen</td>
<td>0</td>
<td>0.1</td>
<td>-0.05</td>
<td>0.29</td>
<td>0.17</td>
<td>0.33</td>
<td>0.16</td>
<td>1</td>
</tr>
</tbody>
</table>

R code: `cor.plot.upperLowerCi(ci)`
Multiple types of reliability

1. Internal consistency estimates
   - $\alpha, \lambda_6$, use the `alpha` function
   - $\omega_{hierarchical}$ and $\omega_{total}$ use the `omega` function
2. IntraClass coefficients
   - ICC
3. Rater agreement use `wkappa` function (finds Cohen’s kappa and weighted kappa)
4. Multilevel reliability and generalizability, use `mlr` or `multilevel.reliability`
For the next examples we will use a built in data set

1. `bfi` consists of 25 personality items measuring 5 factors as well as some demographics.
2. The data were collected as part of the SAPA project and have 2,800 subjects.
3. For help on this data set, `?bfi`
4. To see all of the `psych` data sets: `data(package="psych")`
First, we intentionally misspecify the data

\[
\text{alpha(bfi[1:5])} \quad \#\text{score the first five items}
\]

Some items (A1) were negatively correlated with the total scale and probably should be reversed.
To do this, run the function again with the 'check.keys=TRUE' option.

Reliability analysis
Call: alpha(x = bfi[1:5])

<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>
<th>median_r</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.43</td>
<td>0.46</td>
<td>0.53</td>
<td>0.15</td>
<td>0.85</td>
<td>0.016</td>
<td>4.2</td>
<td>0.74</td>
<td>0.32</td>
</tr>
</tbody>
</table>

lower alpha upper 95% confidence boundaries
0.4 0.43 0.46

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 se</th>
<th>var.r</th>
<th>med.r</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1 0.72</td>
<td>0.73</td>
<td>0.67</td>
<td>0.398</td>
<td>2.64</td>
<td>0.0087</td>
<td>0.376</td>
<td></td>
</tr>
<tr>
<td>A2 0.28</td>
<td>0.30</td>
<td>0.39</td>
<td>0.097</td>
<td>0.43</td>
<td>0.0219</td>
<td>0.081</td>
<td></td>
</tr>
<tr>
<td>A3 0.18</td>
<td>0.21</td>
<td>0.31</td>
<td>0.061</td>
<td>0.26</td>
<td>0.0249</td>
<td>0.081</td>
<td></td>
</tr>
<tr>
<td>A4 0.25</td>
<td>0.31</td>
<td>0.44</td>
<td>0.099</td>
<td>0.44</td>
<td>0.0229</td>
<td>0.105</td>
<td></td>
</tr>
<tr>
<td>A5 0.21</td>
<td>0.24</td>
<td>0.36</td>
<td>0.072</td>
<td>0.31</td>
<td>0.0238</td>
<td>0.095</td>
<td></td>
</tr>
</tbody>
</table>

Item statistics

<table>
<thead>
<tr>
<th>n</th>
<th>raw.r</th>
<th>std.r</th>
<th>r.cor</th>
<th>r.drop</th>
<th>mean</th>
<th>sd</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>2784</td>
<td>0.066</td>
<td>0.024</td>
<td>-0.39</td>
<td>-0.31</td>
<td>2.4</td>
</tr>
<tr>
<td>A2</td>
<td>2773</td>
<td>0.630</td>
<td>0.666</td>
<td>0.58</td>
<td>0.37</td>
<td>4.8</td>
</tr>
<tr>
<td>A3</td>
<td>2774</td>
<td>0.724</td>
<td>0.742</td>
<td>0.72</td>
<td>0.48</td>
<td>4.6</td>
</tr>
<tr>
<td>A4</td>
<td>2781</td>
<td>0.686</td>
<td>0.661</td>
<td>0.50</td>
<td>0.37</td>
<td>4.7</td>
</tr>
<tr>
<td>A5</td>
<td>2784</td>
<td>0.700</td>
<td>0.719</td>
<td>0.64</td>
<td>0.45</td>
<td>4.6</td>
</tr>
</tbody>
</table>
Try it again. Turn on automatic reversals. Get the scores

R code

```r
scores <- alpha(bfi[1:5], check.keys = TRUE)
```

Reliability analysis

Call: alpha(x = bfi[1:5], check.keys = TRUE)

```
raw_alpha std.alpha G6(smc) average_r S/N   ase mean   sd median_r
0.7     0.71   0.68  0.33  2.5  0.009 4.7  0.9  0.34
```

lower alpha upper 95% confidence boundaries
0.69 0.7 0.72

Reliability if an item is dropped:

```
raw_alpha std.alpha G6(smc) average_r S/N alpha se var.r med.r
A1- 0.72 0.73 0.67 0.40 2.6 0.0087 0.0065 0.38
A2 0.62 0.63 0.58 0.29 1.7 0.0119 0.0169 0.29
A3 0.60 0.61 0.56 0.28 1.6 0.0124 0.0094 0.32
A4 0.69 0.69 0.65 0.36 2.3 0.0098 0.0159 0.37
A5 0.64 0.66 0.61 0.32 1.9 0.0111 0.0126 0.34
```

Item statistics

```
n raw.r std.r r.cor r.drop mean   sd
A1- 2784 0.58 0.57 0.38 0.31 4.6 1.4
A2 2773 0.73 0.75 0.67 0.56 4.8 1.2
A3 2774 0.76 0.77 0.71 0.59 4.6 1.3
A4 2781 0.65 0.63 0.47 0.39 4.7 1.5
A5 2784 0.69 0.70 0.60 0.49 4.6 1.3
```

Warning message:

```
In alpha(bfi[1:5], check.keys = TRUE) :
Some items were negatively correlated with total scale and were automatically reversed. This is indicated by a negative sign for the variable name.
```
R functions will return objects without necessarily telling you

1. The basic logic of R is that you can do lots of calculations, but you might not want all the output.

2. The output is there, to be processed by other functions if you want, but you probably don’t want to see all of it unless you ask.

3. Thus, alpha returns the scores based upon the scales you asked for, but doesn’t show them, because they are so many.

4. The str command tells you the structure of an object. The names will just list the names of the objects.
```r
names(scores) str(scores)

names(scores)
[1] "total"    "alpha.drop"  "item.stats"  "response.freq" "keys"
    "scores"   "nvar"      "boot.ci"
[9] "boot"    "Unidim"     "Fit"       "call"     "title"

$ total : 'data.frame': 1 obs. of 8 variables:
  ..$ raw_alpha: num 0.703
  ..$ std.alpha: num 0.713
  ..$ G6(smc) : num 0.683
  ..$ average_r: num 0.332
  ..$ S/N     : num 2.48
  ..$ ase     : num 0.00895
  ..$ mean    : num 4.65
  ..$ sd      : num 0.898

$ alpha.drop : 'data.frame': 5 obs. of 6 variables:
  ..$ raw_alpha: num [1:5] 0.719 0.617 0.6 0.686 0.643
  ..$ std.alpha: num [1:5] 0.726 0.626 0.613 0.694 0.656
  ..$ G6(smc) : num [1:5] 0.673 0.579 0.558 0.65 0.605
  ..$ average_r: num [1:5] 0.398 0.295 0.284 0.361 0.322
  ..$ S/N     : num [1:5] 2.64 1.67 1.58 2.26 1.9
  ..$ alpha se : num [1:5] 0.00873 0.0119 0.01244 0.00983 0.01115

$ item.stats : 'data.frame': 5 obs. of 7 variables:
  ..$ n    : num [1:5] 2784 2773 2774 2781 2784
  ..$ raw.r: num [1:5] 0.581 0.728 0.76 0.654 0.687
  ..$ std.r: num [1:5] 0.566 0.748 0.767 0.631 0.699
  ..$ r.cor: num [1:5] 0.376 0.667 0.709 0.471 0.596
  ..$ r.drop: num [1:5] 0.308 0.564 0.587 0.394 0.489
  ..$ mean  : num [1:5] 1.59 1.9 1.6 1.7 1.56
```
One of the objects of \texttt{alpha} is the scores object

\begin{verbatim}
R code
describe(scores$scores)
\end{verbatim}

But, since there are scores for all subjects, but just one score, this is not very interesting.

\begin{verbatim}
describe(scores$scores)
\end{verbatim}
\begin{verbatim}
vars n mean sd median trimmed mad min max range skew kurtosis se
X1 1 2800 4.65 0.9 4.8 4.73 0.89 1 6 5 -0.76 0.4 0.02
\end{verbatim}

Note that \texttt{alpha} has the option of doing cumulative scores (adding up items, or scoring in the unit of the items (the default)).

\begin{verbatim}
R code
scores <- alpha(bfi[1:5], check.keys=TRUE, cumulative=TRUE)
#set the cumulative option to be true
describe(scores$scores)
\end{verbatim}
\begin{verbatim}
describe(scores$scores)
\end{verbatim}
\begin{verbatim}
vars n mean sd median trimmed mad min max range skew kurtosis se
X1 1 2800 23.08 4.54 24 23.43 4.45 5 30 25 -0.73 0.35 0.09
\end{verbatim}
### Scoring Scales

**Perhaps a more useful case: scoring multiple scales using `scoreItems`**

First, define the scoring keys, and then call `scoreitems`. Use the `msq` data set.

```r
keys <- make.keys(msq[1:75], list(
  EA = c("active", "energetic", "vigorous", "wakeful", "wide.awake", "full.of.pep", 
       "lively", 
       "-sleepy", "-tired", "-drowsy"),
  TA = c("intense", "jittery", "fearful", "tense", "clutched.up", "-quiet", "-still", 
       "-placid", "-calm", "-at.rest"),
  PA = c("active", "excited", "strong", "inspired", "determined", "attentive", 
       "interested", "enthusiastic", "proud", "alert"),
  NAf = c("jittery", "nervous", "scared", "afraid", "guilty", "ashamed", "distressed", 
       "upset", "hostile", "irritable" )
)

msq.scores <- scoreItems(keys, msq[1:75])
msq.scores
```

Call: `scoreItems(keys = keys, items = msq[1:75])`

(Unstandardized) Alpha:

```
   EA TA PA NAf
alpha  0.93  0.75  0.92  0.83
```

Standard errors of unstandardized Alpha:

```
   EA TA PA NAf
ASE  0.004  0.0082  0.0044  0.0064
```

Average item correlation:

```
   EA TA PA NAf
average.r  0.58  0.23  0.52  0.33
```
Score multiple scales (continued)

Standard errors of unstandardized Alpha:

<table>
<thead>
<tr>
<th></th>
<th>EA</th>
<th>TA</th>
<th>PA</th>
<th>NAf</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASE</td>
<td>0.004</td>
<td>0.0082</td>
<td>0.0044</td>
<td>0.0064</td>
</tr>
</tbody>
</table>

Average item correlation:

<table>
<thead>
<tr>
<th></th>
<th>EA</th>
<th>TA</th>
<th>PA</th>
<th>NAf</th>
</tr>
</thead>
<tbody>
<tr>
<td>average.r</td>
<td>0.58</td>
<td>0.23</td>
<td>0.52</td>
<td>0.33</td>
</tr>
</tbody>
</table>

Median item correlation:

<table>
<thead>
<tr>
<th></th>
<th>EA</th>
<th>TA</th>
<th>PA</th>
<th>NAf</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.59</td>
<td>0.24</td>
<td>0.52</td>
<td>0.40</td>
<td></td>
</tr>
</tbody>
</table>

Guttman 6* reliability:

<table>
<thead>
<tr>
<th></th>
<th>EA</th>
<th>TA</th>
<th>PA</th>
<th>NAf</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lambda.6</td>
<td>0.95</td>
<td>0.8</td>
<td>0.93</td>
<td>0.87</td>
</tr>
</tbody>
</table>

Signal/Noise based upon av.r:

<table>
<thead>
<tr>
<th></th>
<th>EA</th>
<th>TA</th>
<th>PA</th>
<th>NAf</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>3</td>
<td>11</td>
<td>4.9</td>
<td></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></th>
<th>EA</th>
<th>TA</th>
<th>PA</th>
<th>NAf</th>
</tr>
</thead>
<tbody>
<tr>
<td>EA</td>
<td>0.932</td>
<td>0.29</td>
<td>0.870</td>
<td>-0.069</td>
</tr>
<tr>
<td>TA</td>
<td>0.238</td>
<td>0.75</td>
<td>0.226</td>
<td>0.710</td>
</tr>
<tr>
<td>PA</td>
<td>0.804</td>
<td>0.19</td>
<td>0.915</td>
<td>0.044</td>
</tr>
<tr>
<td>NAf</td>
<td>-0.061</td>
<td>0.56</td>
<td>0.039</td>
<td>0.831</td>
</tr>
</tbody>
</table>
More detailed item statistics

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

<table>
<thead>
<tr>
<th></th>
<th>EA</th>
<th>TA</th>
<th>PA</th>
<th>NAf</th>
</tr>
</thead>
<tbody>
<tr>
<td>delighted</td>
<td>0.56</td>
<td>0.04</td>
<td>0.67</td>
<td>-0.17</td>
</tr>
<tr>
<td>sociable</td>
<td>0.57</td>
<td>0.06</td>
<td>0.64</td>
<td>-0.15</td>
</tr>
<tr>
<td>jittery</td>
<td>0.15</td>
<td>0.52</td>
<td>0.23</td>
<td>0.46</td>
</tr>
<tr>
<td>hostile</td>
<td>-0.21</td>
<td>0.37</td>
<td>-0.13</td>
<td>0.58</td>
</tr>
<tr>
<td>sluggish</td>
<td>-0.65</td>
<td>-0.02</td>
<td>-0.40</td>
<td>0.21</td>
</tr>
<tr>
<td>depressed</td>
<td>-0.30</td>
<td>0.44</td>
<td>-0.26</td>
<td>0.67</td>
</tr>
<tr>
<td>satisfied</td>
<td>0.54</td>
<td>-0.23</td>
<td>0.62</td>
<td>-0.35</td>
</tr>
<tr>
<td>relaxed</td>
<td>0.35</td>
<td>-0.56</td>
<td>0.40</td>
<td>-0.51</td>
</tr>
<tr>
<td>warmhearted</td>
<td>0.47</td>
<td>-0.09</td>
<td>0.66</td>
<td>-0.19</td>
</tr>
<tr>
<td>blue</td>
<td>-0.23</td>
<td>0.40</td>
<td>-0.19</td>
<td>0.64</td>
</tr>
<tr>
<td>intense</td>
<td>0.25</td>
<td>0.42</td>
<td>0.44</td>
<td>0.47</td>
</tr>
<tr>
<td>strong</td>
<td>0.55</td>
<td>0.00</td>
<td>0.69</td>
<td>-0.03</td>
</tr>
<tr>
<td>scared</td>
<td>-0.05</td>
<td>0.61</td>
<td>0.08</td>
<td>0.75</td>
</tr>
<tr>
<td>enthusiastic</td>
<td>0.69</td>
<td>0.16</td>
<td>0.83</td>
<td>-0.07</td>
</tr>
<tr>
<td>proud</td>
<td>0.57</td>
<td>0.05</td>
<td>0.73</td>
<td>-0.11</td>
</tr>
<tr>
<td>sad</td>
<td>-0.22</td>
<td>0.46</td>
<td>-0.18</td>
<td>0.73</td>
</tr>
<tr>
<td>active</td>
<td>0.71</td>
<td>0.18</td>
<td>0.83</td>
<td>-0.05</td>
</tr>
<tr>
<td>full.of.pep</td>
<td>0.80</td>
<td>0.05</td>
<td>0.80</td>
<td>-0.15</td>
</tr>
<tr>
<td>unhappy</td>
<td>-0.30</td>
<td>0.44</td>
<td>-0.26</td>
<td>0.70</td>
</tr>
<tr>
<td>lively</td>
<td>0.78</td>
<td>0.04</td>
<td>0.81</td>
<td>-0.13</td>
</tr>
</tbody>
</table>

Non missing response frequency for each item

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>miss</th>
</tr>
</thead>
<tbody>
<tr>
<td>delighted</td>
<td>0.50</td>
<td>0.27</td>
<td>0.19</td>
<td>0.04</td>
<td>0.00</td>
</tr>
<tr>
<td>sociable</td>
<td>0.24</td>
<td>0.30</td>
<td>0.35</td>
<td>0.11</td>
<td>0.00</td>
</tr>
<tr>
<td>jittery</td>
<td>0.60</td>
<td>0.28</td>
<td>0.09</td>
<td>0.03</td>
<td>0.00</td>
</tr>
<tr>
<td>hostile</td>
<td>0.77</td>
<td>0.17</td>
<td>0.05</td>
<td>0.01</td>
<td>0.00</td>
</tr>
<tr>
<td>sluggish</td>
<td>0.24</td>
<td>0.44</td>
<td>0.19</td>
<td>0.13</td>
<td>0.00</td>
</tr>
<tr>
<td>depressed</td>
<td>0.61</td>
<td>0.26</td>
<td>0.08</td>
<td>0.05</td>
<td>0.00</td>
</tr>
<tr>
<td>satisfied</td>
<td>0.22</td>
<td>0.29</td>
<td>0.38</td>
<td>0.12</td>
<td>0.00</td>
</tr>
<tr>
<td>relaxed</td>
<td>0.13</td>
<td>0.20</td>
<td>0.43</td>
<td>0.24</td>
<td>0.00</td>
</tr>
<tr>
<td>warmhearted</td>
<td>0.17</td>
<td>0.23</td>
<td>0.37</td>
<td>0.22</td>
<td>0.00</td>
</tr>
<tr>
<td>blue</td>
<td>0.60</td>
<td>0.30</td>
<td>0.08</td>
<td>0.02</td>
<td>0.00</td>
</tr>
<tr>
<td>intense</td>
<td>0.54</td>
<td>0.28</td>
<td>0.15</td>
<td>0.03</td>
<td>0.00</td>
</tr>
<tr>
<td>strong</td>
<td>0.31</td>
<td>0.26</td>
<td>0.32</td>
<td>0.11</td>
<td>0.00</td>
</tr>
<tr>
<td>scared</td>
<td>0.80</td>
<td>0.15</td>
<td>0.04</td>
<td>0.01</td>
<td>0.00</td>
</tr>
<tr>
<td>enthusiastic</td>
<td>0.43</td>
<td>0.29</td>
<td>0.19</td>
<td>0.08</td>
<td>0.00</td>
</tr>
<tr>
<td>proud</td>
<td>0.41</td>
<td>0.24</td>
<td>0.26</td>
<td>0.10</td>
<td>0.00</td>
</tr>
<tr>
<td>sad</td>
<td>0.66</td>
<td>0.22</td>
<td>0.09</td>
<td>0.03</td>
<td>0.00</td>
</tr>
<tr>
<td>active</td>
<td>0.38</td>
<td>0.32</td>
<td>0.22</td>
<td>0.08</td>
<td>0.00</td>
</tr>
<tr>
<td>full.of.pep</td>
<td>0.54</td>
<td>0.21</td>
<td>0.20</td>
<td>0.05</td>
<td>0.00</td>
</tr>
<tr>
<td>unhappy</td>
<td>0.65</td>
<td>0.24</td>
<td>0.07</td>
<td>0.04</td>
<td>0.00</td>
</tr>
<tr>
<td>lively</td>
<td>0.44</td>
<td>0.28</td>
<td>0.22</td>
<td>0.06</td>
<td>0.00</td>
</tr>
</tbody>
</table>
Show the SPLOM of the msq scales using `pairs.panels`
Scoring Scales

But what if we have overlapping scales?

1. Sometimes we are interested in how higher order scales relate to lower order scales.
2. The problem is, the items overlap.
3. Some people solve this problem by dropping the overlapping items. But this changes the meaning of the scales.
4. A fairly straightforward procedure is estimate the overlapping variances with the best estimate of shared (common) variance, similar to what is done when finding coefficient $\alpha$.
5. Need to do this on the correlation matrix of the items, not the raw data.
6. See ?scoreOverlap
Correcting for item overlap using `scoreOverlap`

```r
small.R <- cor(small.msq,use="pairwise")
keys.list <- list(
  EA = c("active", "energetic", "vigorous", "wakeful", "wide.awake", "full.of.pep", "lively", "-sleepy", "-tired", "-drowsy"),
  TA = c("intense", "jittery", "fearful", "tense", "clutched.up", "-quiet", "-still", "-placid", "-calm", "-at.rest"),
  high.EA = c("active", "energetic", "vigorous", "wakeful", "wide.awake", "full.of.pep", "lively"),
  low.EA = c("sleepy", "tired", "drowsy"),
  lowTA = c("quiet", "still", "placid", "calm", "at.rest"),
  highTA = c("intense", "jittery", "fearful", "tense", "clutched.up")
)

keys <- make.keys(small.R,keys.list)

adjusted.scales <- scoreOverlap(keys.list,small.R)
```
Scoring Scales

Correcting for item overlap using `scoreOverlap`. (continued)

Call: scoreOverlap(keys = keys.list, r = small.R)

(Standardized) Alpha:

<table>
<thead>
<tr>
<th></th>
<th>EA</th>
<th>TA</th>
<th>high.EA</th>
<th>low.EA</th>
<th>lowTA</th>
<th>highTA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.93</td>
<td>0.75</td>
<td>0.94</td>
<td>0.93</td>
<td>0.73</td>
<td>0.76</td>
</tr>
</tbody>
</table>

(Standardized) G6*:

<table>
<thead>
<tr>
<th></th>
<th>EA</th>
<th>TA</th>
<th>high.EA</th>
<th>low.EA</th>
<th>lowTA</th>
<th>highTA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.88</td>
<td>0.68</td>
<td>0.94</td>
<td>0.90</td>
<td>0.73</td>
<td>0.75</td>
</tr>
</tbody>
</table>

Average item correlation:

<table>
<thead>
<tr>
<th></th>
<th>EA</th>
<th>TA</th>
<th>high.EA</th>
<th>low.EA</th>
<th>lowTA</th>
<th>highTA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.59</td>
<td>0.23</td>
<td>0.68</td>
<td>0.81</td>
<td>0.35</td>
<td>0.38</td>
</tr>
</tbody>
</table>

Number of items:

<table>
<thead>
<tr>
<th></th>
<th>EA</th>
<th>TA</th>
<th>high.EA</th>
<th>low.EA</th>
<th>lowTA</th>
<th>highTA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>10</td>
<td>10</td>
<td>7</td>
<td>3</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

Signal to Noise ratio based upon average r and n

<table>
<thead>
<tr>
<th></th>
<th>EA</th>
<th>TA</th>
<th>high.EA</th>
<th>low.EA</th>
<th>lowTA</th>
<th>highTA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>14.1</td>
<td>3.0</td>
<td>14.8</td>
<td>12.9</td>
<td>2.7</td>
<td>3.1</td>
</tr>
</tbody>
</table>

Scale intercorrelations corrected for item overlap and attenuation adjusted for overlap correlations below the diagonal, alpha on the diagonal corrected correlations above the diagonal:

<table>
<thead>
<tr>
<th></th>
<th>EA</th>
<th>TA</th>
<th>high.EA</th>
<th>low.EA</th>
<th>lowTA</th>
<th>highTA</th>
</tr>
</thead>
<tbody>
<tr>
<td>EA</td>
<td>0.93</td>
<td>0.27</td>
<td>0.965</td>
<td>-0.803</td>
<td>-0.18</td>
<td>0.253</td>
</tr>
<tr>
<td>TA</td>
<td>0.23</td>
<td>0.75</td>
<td>0.282</td>
<td>-0.167</td>
<td>-0.81</td>
<td>0.821</td>
</tr>
<tr>
<td>high.EA</td>
<td>0.90</td>
<td>0.24</td>
<td>0.937</td>
<td>-0.620</td>
<td>-0.12</td>
<td>0.324</td>
</tr>
<tr>
<td>low.EA</td>
<td>-0.75</td>
<td>-0.14</td>
<td>-0.579</td>
<td>0.928</td>
<td>0.25</td>
<td>-0.023</td>
</tr>
<tr>
<td>lowTA</td>
<td>-0.15</td>
<td>-0.60</td>
<td>-0.098</td>
<td>0.204</td>
<td>0.73</td>
<td>-0.335</td>
</tr>
<tr>
<td>highTA</td>
<td>0.91</td>
<td>0.68</td>
<td>0.972</td>
<td>-0.018</td>
<td>-0.35</td>
<td>0.757</td>
</tr>
</tbody>
</table>
### Compare scoreOverlap with non-adjusted

```r
adjusted.scales <- scoreOverlap(keys.list, small.R)
raw <- scoreItems(keys.list, small.R)
```

#### Scale intercorrelations corrected for item overlap and attenuation
adjusted for overlap correlations below the diagonal, alpha on the diagonal corrected correlations above the diagonal:

<table>
<thead>
<tr>
<th></th>
<th>EA</th>
<th>TA</th>
<th>high.EA</th>
<th>low.EA</th>
<th>lowTA</th>
<th>highTA</th>
</tr>
</thead>
<tbody>
<tr>
<td>EA</td>
<td>0.93</td>
<td>0.27</td>
<td>0.965</td>
<td>-0.803</td>
<td>-0.18</td>
<td>0.253</td>
</tr>
<tr>
<td>TA</td>
<td>0.23</td>
<td>0.75</td>
<td>0.282</td>
<td>-0.167</td>
<td>-0.81</td>
<td>0.821</td>
</tr>
<tr>
<td>high.EA</td>
<td>0.90</td>
<td>0.24</td>
<td>0.937</td>
<td>-0.620</td>
<td>-0.12</td>
<td>0.324</td>
</tr>
<tr>
<td>low.EA</td>
<td>-0.75</td>
<td>-0.14</td>
<td>-0.579</td>
<td>0.928</td>
<td>0.25</td>
<td>-0.023</td>
</tr>
<tr>
<td>lowTA</td>
<td>-0.15</td>
<td>-0.60</td>
<td>-0.098</td>
<td>0.204</td>
<td>0.73</td>
<td>-0.335</td>
</tr>
<tr>
<td>highTA</td>
<td>0.21</td>
<td>0.62</td>
<td>0.273</td>
<td>-0.019</td>
<td>-0.25</td>
<td>0.757</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></th>
<th>EA</th>
<th>TA</th>
<th>high.EA</th>
<th>low.EA</th>
<th>lowTA</th>
<th>highTA</th>
</tr>
</thead>
<tbody>
<tr>
<td>EA</td>
<td>0.93</td>
<td>0.27</td>
<td>1.024</td>
<td>-0.848</td>
<td>-0.18</td>
<td>0.253</td>
</tr>
<tr>
<td>TA</td>
<td>0.23</td>
<td>0.75</td>
<td>0.282</td>
<td>-0.167</td>
<td>-1.06</td>
<td>1.056</td>
</tr>
<tr>
<td>high.EA</td>
<td>0.96</td>
<td>0.24</td>
<td>0.937</td>
<td>-0.620</td>
<td>-0.12</td>
<td>0.324</td>
</tr>
<tr>
<td>low.EA</td>
<td>-0.79</td>
<td>-0.14</td>
<td>-0.579</td>
<td>0.928</td>
<td>0.25</td>
<td>-0.023</td>
</tr>
<tr>
<td>lowTA</td>
<td>-0.15</td>
<td>-0.78</td>
<td>-0.098</td>
<td>0.204</td>
<td>0.73</td>
<td>-0.335</td>
</tr>
<tr>
<td>highTA</td>
<td>0.21</td>
<td>0.80</td>
<td>0.273</td>
<td>-0.019</td>
<td>-0.25</td>
<td>0.757</td>
</tr>
</tbody>
</table>
**α, ω_{hierarchical} and β** as alternative measures of internal consistency

1. **α** as the mean split half reliability
   - alpha to find \( \alpha \)
   - splitHalf to find all (if \( n \leq 16 \)) or 10,000 random possible split half reliabilities (\( n > 16 \))

2. **ω_{hierarchical} and ω_{total}** as factor based reliabilities
   - \( \omega_{hierarchical} \) estimates general factor saturation
   - Found using omega and omegaSem

3. **β** as worst split half reliability as an alternative estimate of the general factor saturation.
   - Found using a hierarchical clustering algorithm (iclust).
   - iclust is also useful for scale construction.
α from alpha and all split halves found using splitHalf

Find α and all split half reliabilities of 5 Agreeableness items and 5 Conscientiousness items from the bfi data set included in psych.

R code

```r
alpha(bfi[1:10], check.keys=TRUE) #find alpha, let it automatically reverse items
splitHalf(bfi[1:10], key=c(1,9,10)) #reverse 3 items
```

Reliability analysis
Call: alpha(x = bfi[1:10])

<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>0.73</td>
<td>0.74</td>
<td>0.76</td>
<td>0.22</td>
<td>2.8</td>
<td>0.01</td>
<td>4.5</td>
<td>0.73</td>
</tr>
</tbody>
</table>

lower alpha upper 95% confidence boundaries
0.71 0.73 0.75

Split half reliabilities
Call: splitHalf(r = bfi[1:10], key = c(1, 9, 10))

Maximum split half reliability (lambda 4) = 0.81
Guttman lambda 6 = 0.76
Average split half reliability = 0.73
Guttman lambda 3 (alpha) = 0.74
Minimum split half reliability (beta) = 0.41
Average interitem r = 0.22 with median = 0.17
All possible spit halves of 5 agreeableness and 5 conscientiousness items. Note the one worst one! This is not one construct.
Using the \texttt{omega} function

\begin{verbatim}
R code

\texttt{omega(ability,4)}
\end{verbatim}

\begin{verbatim}
Omega
Call: omega(m = ability, nfactors = 4)
Alpha: 0.83
G.6: 0.84
Omega Hierarchical: 0.65
Omega H asymptotic: 0.76
Omega Total 0.86

Schmid Leiman Factor loadings greater than 0.2
\begin{verbatim}
g F1* F2* F3* F4* h2 u2 p2
reason.4 0.50 0.27 0.34 0.66 0.73
reason.16 0.42 0.21 0.23 0.77 0.76
reason.17 0.55 0.47 0.52 0.48 0.57
reason.19 0.44 0.21 0.25 0.75 0.77
letter.7 0.52 0.35 0.39 0.61 0.69
letter.33 0.46 0.30 0.31 0.69 0.70
letter.34 0.54 0.38 0.43 0.57 0.67
letter.58 0.47 0.20 0.28 0.72 0.78
matrix.45 0.40 0.28 0.66 0.59 0.41 0.27
matrix.46 0.40 0.20 0.26 0.24 0.76 0.65
matrix.47 0.42 0.20 0.23 0.77 0.79
matrix.55 0.28 0.12 0.88 0.65
rotate.3 0.36 0.61 0.50 0.50 0.26
rotate.4 0.41 0.61 0.54 0.46 0.31
rotate.6 0.40 0.49 0.41 0.59 0.39
rotate.8 0.32 0.53 0.40 0.60 0.26
\end{verbatim}

With eigenvalues of:
\begin{verbatim}
g F1* F2* F3* F4*
3.04 1.32 0.46 0.42 0.55
\end{verbatim}

Explained Common Variance of the general factor = 0.53
The degrees of freedom are 62 and the fit is 0.05
The number of observations was 1525 with Chi Square = 70.19 with prob < 0.22
The root mean square of the residuals is 0.01
The df corrected root mean square of the residuals is 0.02
RMSEA index = 0.01 and the 90 % confidence intervals are NA 0.019
BIC = -384.25

Compare this with the adequacy of just a general factor and no group factors
The degrees of freedom for just the general factor are 104 and the fit is 0.78
The number of observations was 1525 with Chi Square = 1186.18 with prob < 5e-183
The root mean square of the residuals is 0.09
The df corrected root mean square of the residuals is 0.09
RMSEA index = 0.083 and the 90 % confidence intervals are 0.078 0.087
BIC = 423.88
Estimating $\omega_{hierarchical}$ and $\omega_{total}$ using omega

16 ability items from the International Cognitive Ability Resource

general ability and 4 subfactors of ICAR data
Estimating $\omega_{\text{hierarchical}}$ and $\omega_{\text{total}}$ using omega

general ability and 4 subfactors of ICAR data

\[
\begin{align*}
\text{rotate.3} & \rightarrow F1^* \\
\text{rotate.4} & \rightarrow F1^* \\
\text{rotate.8} & \rightarrow F2^* \\
\text{rotate.6} & \rightarrow F2^* \\
\text{letter.34} & \rightarrow F2^* \\
\text{letter.7} & \rightarrow F3^* \\
\text{letter.33} & \rightarrow F3^* \\
\text{letter.58} & \rightarrow F4^* \\
\text{matrix.47} & \rightarrow F4^* \\
\text{reason.17} & \rightarrow F4^* \\
\text{reason.4} & \rightarrow F4^* \\
\text{reason.16} & \rightarrow F4^* \\
\text{reason.19} & \rightarrow F4^* \\
\text{matrix.45} & \rightarrow F4^* \\
\text{matrix.46} & \rightarrow F4^* \\
\text{matrix.55} & \rightarrow F4^*
\end{align*}
\]
Hierarchical clustering of 16 ICAR ability items: iclust(ability)

Hierarchical clustering of 16 ability items using iclust

Estimating $\omega_{\text{hierarchical}}$ and $\omega_{\text{total}}$ using omega
Exploratory Factor Analysis

1. How many factors: an unsolved problem
   • Parallel analysis, MAPS, VSS, BIC, RMSEA, etc. available in `nfacors` and `fa.parallel`

2. Factor extraction algorithms available in the `fa` function
   • maximum likelihood, minimum residual, principal factor, ...

3. Factor rotation procedures are done using `GPArotation` package
   • orthogonal: varimax, quartimax, bifactor, ...
   • oblique: oblimin, geomin, biquartimin, ...

4. Displaying the solutions using `fa.plot`

Note, that EFA is not the same as Principal Components Analysis and the two should not be confused.

1. PCA done using `principal`
How many factors are in the mood data

The number of factors problem is easy and hard

No best rule, one worst rule

“Solving the number of factors problem is easy, I do it everyday before breakfast. But knowing the right solution is harder.” (Henry Kaiser)

1. Parallel analysis (Extract factors until the eigen values are less than those of a random matrix).
   - Although a good rule for 100-500 subjects, this will not do as well with many (>1000) subjects.

2. Velicer’s Minimum Average Partial (MAP) is pretty good

3. For items, the Very Simple Structure (VSS) criterion is pretty good.

4. Multiple statistical tests, many have problems with sample size.
   - If you want few factors, run few subjects
   - If you want many factors, run many subjects

5. One worst rule is the eigen value of 1.0 rule.
How many factors are in the mood data

**What about parallel analysis? Pearson R or polychoric ρ?**

Of the raw (Pearson) correlations compared to the polychoric correlations

```r
fa.parallel(cleaned[2:73])
Parallel analysis suggests that the number of factors = 5 and the number of components = 5

#use polychoric correlations
fa.parallel(cleaned[2:73], cor="poly")

> fa.parallel(cleaned[2:73], cor="poly")
some warnings are issued
The items do not have an equal number of response alternatives, global set to FALSE
Parallel analysis suggests that the number of factors = 4 and the number of components = 4
Warning message:
In cor.smooth(mat) : Matrix was not positive definite, smoothing was done
>
```
How many factors are in the mood data

**Parallel analysis with Pearson correlations**

Parallel Analysis Scree Plots

- **PC Actual Data**
- **PC Simulated Data**
- **FA Actual Data**
- **FA Simulated Data**

- Eigenvalues of principal components and factor analysis

Between 4 and 6 factors seem to be significant.
How many factors are in the mood data

**Parallel analysis with polychoric correlations (takes somewhat longer)**

![Parallel Analysis Scree Plots](image)

- **eigenvalues of principal components and factor analysis**
- **Factor/Component Number**
- **Parallel Analysis Scree Plots**
  - PC Actual Data
  - PC Simulated Data
  - PC Resampled Data
  - FA Actual Data
  - FA Simulated Data
  - FA Resampled Data
How many factors: what does `nfactors` tell us?

> nfactors(cleaned[2:73])

Number of factors
Call: vss(x = x, n = n, rotate = rotate, diagonal = diagonal, fm = fm,
       n.obs = n.obs, plot = FALSE, title = title, use = use, cor = cor)
VSS complexity 1 achieves a maximum of 0.74 with 2 factors
VSS complexity 2 achieves a maximum of 0.91 with 3 factors
The Velicer MAP achieves a minimum of 0.01 with 9 factors
Empirical BIC achieves a minimum of -10081.25 with 6 factors
Sample Size adjusted BIC achieves a minimum of -1843.62 with 11 factors

Statistics by number of factors

<table>
<thead>
<tr>
<th>vss1</th>
<th>vss2</th>
<th>map</th>
<th>dof</th>
<th>chisq</th>
<th>prob</th>
<th>sqresid</th>
<th>fit</th>
<th>RMSEA</th>
<th>BIC</th>
<th>SABIC</th>
<th>complex</th>
<th>eChisq</th>
<th>SRMR</th>
<th>eCRMS</th>
<th>eBIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.68</td>
<td>0.00</td>
<td>0.0552</td>
<td>2484</td>
<td>8907</td>
<td>0.0e+00</td>
<td>192.3</td>
<td>0.68</td>
<td>0.115</td>
<td>-4612</td>
<td>3261</td>
<td>1.0</td>
<td>35253</td>
<td>0.173</td>
<td>0.175</td>
</tr>
<tr>
<td>2</td>
<td>0.74</td>
<td>0.90</td>
<td>0.0179</td>
<td>2413</td>
<td>5827</td>
<td>2.9e-282</td>
<td>58.9</td>
<td>0.90</td>
<td>0.087</td>
<td>-7305</td>
<td>343</td>
<td>1.2</td>
<td>7619</td>
<td>0.080</td>
<td>0.083</td>
</tr>
<tr>
<td>3</td>
<td>0.69</td>
<td>0.91</td>
<td>0.0145</td>
<td>2343</td>
<td>4928</td>
<td>8.6e-186</td>
<td>40.3</td>
<td>0.93</td>
<td>0.077</td>
<td>-7823</td>
<td>-397</td>
<td>1.4</td>
<td>4442</td>
<td>0.061</td>
<td>0.064</td>
</tr>
<tr>
<td>4</td>
<td>0.66</td>
<td>0.86</td>
<td>0.0110</td>
<td>2274</td>
<td>4095</td>
<td>1.7e-107</td>
<td>29.0</td>
<td>0.95</td>
<td>0.067</td>
<td>-8281</td>
<td>-1074</td>
<td>1.5</td>
<td>2532</td>
<td>0.046</td>
<td>0.049</td>
</tr>
<tr>
<td>5</td>
<td>0.62</td>
<td>0.85</td>
<td>0.0101</td>
<td>2206</td>
<td>3587</td>
<td>1.9e-69</td>
<td>24.9</td>
<td>0.96</td>
<td>0.061</td>
<td>-8419</td>
<td>-1427</td>
<td>1.7</td>
<td>1933</td>
<td>0.040</td>
<td>0.043</td>
</tr>
<tr>
<td>6</td>
<td>0.63</td>
<td>0.84</td>
<td>0.0094</td>
<td>2139</td>
<td>3274</td>
<td>4.1e-51</td>
<td>22.0</td>
<td>0.96</td>
<td>0.057</td>
<td>-8367</td>
<td>-1588</td>
<td>1.8</td>
<td>1560</td>
<td>0.036</td>
<td>0.039</td>
</tr>
<tr>
<td>7</td>
<td>0.63</td>
<td>0.83</td>
<td>0.0094</td>
<td>2073</td>
<td>3047</td>
<td>2.1e-40</td>
<td>19.7</td>
<td>0.97</td>
<td>0.055</td>
<td>-8235</td>
<td>-1665</td>
<td>1.9</td>
<td>1314</td>
<td>0.033</td>
<td>0.036</td>
</tr>
<tr>
<td>8</td>
<td>0.63</td>
<td>0.82</td>
<td>0.0091</td>
<td>2008</td>
<td>2810</td>
<td>7.0e-30</td>
<td>17.9</td>
<td>0.97</td>
<td>0.052</td>
<td>-8118</td>
<td>-1754</td>
<td>2.0</td>
<td>1100</td>
<td>0.031</td>
<td>0.034</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>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.53</td>
<td>0.78</td>
<td>0.0105</td>
<td>1413</td>
<td>1467</td>
<td>1.5e-01</td>
<td>9.2</td>
<td>0.98</td>
<td>0.032</td>
<td>-6223</td>
<td>-1745</td>
<td>2.6</td>
<td>324</td>
<td>0.017</td>
<td>0.017</td>
</tr>
<tr>
<td>11</td>
<td>0.52</td>
<td>0.78</td>
<td>0.0108</td>
<td>1359</td>
<td>1371</td>
<td>4.0e-01</td>
<td>8.7</td>
<td>0.99</td>
<td>0.030</td>
<td>-6025</td>
<td>-1718</td>
<td>2.6</td>
<td>291</td>
<td>0.016</td>
<td>0.016</td>
</tr>
<tr>
<td>12</td>
<td>0.52</td>
<td>0.77</td>
<td>0.0111</td>
<td>1306</td>
<td>1284</td>
<td>6.6e-01</td>
<td>8.2</td>
<td>0.99</td>
<td>0.028</td>
<td>-5824</td>
<td>-1685</td>
<td>2.7</td>
<td>258</td>
<td>0.015</td>
<td>0.015</td>
</tr>
</tbody>
</table>
How many factors are in the mood data

The number of factors from nfactors

![Graphs showing the number of factors for Very Simple Structure, Complexity, Empirical BIC, and Root Mean Residual.](image-url)
How many factors are in the mood data

What if we use polychoric correlations

```r
> nfactors(cleaned[2:73], cor="poly")
The items do not have an equal number of response alternatives, global set to FALSE

Number of factors
Call: vss(x = x, n = n, rotate = rotate, diagonal = diagonal, fm = fm, 
    n.obs = n.obs, plot = FALSE, title = title, use = use, cor = cor)
VSS complexity 1 achieves a maximum of 0.76 with 2 factors
VSS complexity 2 achieves a maximum of 0.93 with 2 factors
The Velicer MAP achieves a minimum of 0.02 with 9 factors
Empirical BIC achieves a minimum of -9764.2 with 5 factors
Sample Size adjusted BIC achieves a minimum of 39801.49 with 20 factors

Statistics by number of factors

<table>
<thead>
<tr>
<th></th>
<th>vss1</th>
<th>vss2</th>
<th>map</th>
<th>dof</th>
<th>chisq</th>
<th>prob</th>
<th>sqresid</th>
<th>fit</th>
<th>RMSEA</th>
<th>BIC</th>
<th>SABIC</th>
<th>complex</th>
<th>eChisq</th>
<th>SRMR</th>
<th>eCRMS</th>
<th>eBIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.66</td>
<td>0.00</td>
<td>0.099</td>
<td>2484</td>
<td>56503</td>
<td>0</td>
<td>280.2</td>
<td>0.66</td>
<td>0.33</td>
<td>42984</td>
<td>50857</td>
<td>1.0</td>
<td>56472</td>
<td>0.219</td>
<td>0.222</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.76</td>
<td>0.93</td>
<td>0.027</td>
<td>2413</td>
<td>51500</td>
<td>0</td>
<td>58.1</td>
<td>0.93</td>
<td>0.32</td>
<td>38368</td>
<td>46016</td>
<td>1.3</td>
<td>9088</td>
<td>0.088</td>
<td>0.090</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.70</td>
<td>0.92</td>
<td>0.023</td>
<td>2343</td>
<td>50169</td>
<td>0</td>
<td>31.0</td>
<td>0.96</td>
<td>0.32</td>
<td>37417</td>
<td>44843</td>
<td>1.4</td>
<td>4295</td>
<td>0.060</td>
<td>0.063</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.67</td>
<td>0.89</td>
<td>0.018</td>
<td>2274</td>
<td>48889</td>
<td>0</td>
<td>21.6</td>
<td>0.97</td>
<td>0.32</td>
<td>36513</td>
<td>43721</td>
<td>1.5</td>
<td>2649</td>
<td>0.047</td>
<td>0.050</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.67</td>
<td>0.89</td>
<td>0.018</td>
<td>2206</td>
<td>48230</td>
<td>0</td>
<td>18.6</td>
<td>0.98</td>
<td>0.32</td>
<td>36224</td>
<td>43216</td>
<td>1.7</td>
<td>2242</td>
<td>0.044</td>
<td>0.047</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.66</td>
<td>0.87</td>
<td>0.018</td>
<td>2139</td>
<td>47695</td>
<td>0</td>
<td>16.0</td>
<td>0.98</td>
<td>0.33</td>
<td>36053</td>
<td>42833</td>
<td>1.8</td>
<td>1930</td>
<td>0.040</td>
<td>0.044</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0.64</td>
<td>0.86</td>
<td>0.017</td>
<td>2073</td>
<td>47098</td>
<td>0</td>
<td>13.7</td>
<td>0.98</td>
<td>0.33</td>
<td>35816</td>
<td>42386</td>
<td>1.9</td>
<td>1612</td>
<td>0.037</td>
<td>0.041</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.65</td>
<td>0.84</td>
<td>0.017</td>
<td>2008</td>
<td>46595</td>
<td>0</td>
<td>11.8</td>
<td>0.99</td>
<td>0.33</td>
<td>35666</td>
<td>42031</td>
<td>1.9</td>
<td>1391</td>
<td>0.034</td>
<td>0.039</td>
<td></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>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>0.59</td>
<td>0.82</td>
<td>0.022</td>
<td>1359</td>
<td>43055</td>
<td>0</td>
<td>3.5</td>
<td>1.00</td>
<td>0.40</td>
<td>35659</td>
<td>39966</td>
<td>2.6</td>
<td>462</td>
<td>0.020</td>
<td>0.027</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>0.56</td>
<td>0.82</td>
<td>0.022</td>
<td>1306</td>
<td>42770</td>
<td>0</td>
<td>3.1</td>
<td>1.00</td>
<td>0.41</td>
<td>35662</td>
<td>39801</td>
<td>2.6</td>
<td>419</td>
<td>0.019</td>
<td>0.026</td>
<td></td>
</tr>
</tbody>
</table>
```

Warning message:
In cor.smooth(mat) : Matrix was not positive definite, smoothing was done
How many factors are in the mood data

### The number of factors from \( n \)factors

#### Very Simple Structure

<table>
<thead>
<tr>
<th>Number of Factors</th>
<th>Very Simple Structure Fit</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>2 2 2 2 2 2 2 2 2 2 2 2 2</td>
</tr>
<tr>
<td>10</td>
<td>3 3 3 3 3 3 3 3 3 3 3 3 3</td>
</tr>
<tr>
<td>15</td>
<td>4 4 4 4 4 4 4 4 4 4 4 4 4</td>
</tr>
<tr>
<td>20</td>
<td>5 5 5 5 5 5 5 5 5 5 5 5 5</td>
</tr>
</tbody>
</table>

#### Complexity

<table>
<thead>
<tr>
<th>Number of factors</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1.0</td>
</tr>
<tr>
<td>10</td>
<td>1.5</td>
</tr>
<tr>
<td>15</td>
<td>2.0</td>
</tr>
<tr>
<td>20</td>
<td>2.5</td>
</tr>
</tbody>
</table>

#### Empirical BIC

<table>
<thead>
<tr>
<th>Number of factors</th>
<th>Empirical BIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>30000</td>
</tr>
<tr>
<td>10</td>
<td>-10000</td>
</tr>
<tr>
<td>15</td>
<td>-15000</td>
</tr>
<tr>
<td>20</td>
<td>-20000</td>
</tr>
</tbody>
</table>

#### Root Mean Residual

<table>
<thead>
<tr>
<th>Number of factors</th>
<th>SRMR</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.05</td>
</tr>
<tr>
<td>10</td>
<td>0.10</td>
</tr>
<tr>
<td>15</td>
<td>0.15</td>
</tr>
<tr>
<td>20</td>
<td>0.20</td>
</tr>
</tbody>
</table>
Factor extraction and graphical displays

### EFA of the Motivational State Questionnaire

```r
f2 <- fa(msq[1:72],2)
summary(f2)
```

```
Factor analysis with Call: fa(r = msq[1:72], nfactors = 2)

Test of the hypothesis that 2 factors are sufficient.
The degrees of freedom for the model is 2413 and the objective function was 17.51
The number of observations was 3896 with Chi Square = 67730.13

The root mean square of the residuals (RMSA) is 0.09
The df corrected root mean square of the residuals is 0.09

Tucker Lewis Index of factoring reliability = 0.637
RMSEA index = 0.083 and the 90 % confidence intervals are 0.083
BIC = 47780.16

With factor correlations of

<table>
<thead>
<tr>
<th></th>
<th>MR1</th>
<th>MR2</th>
</tr>
</thead>
<tbody>
<tr>
<td>MR1</td>
<td>1.00</td>
<td>-0.13</td>
</tr>
<tr>
<td>MR2</td>
<td>-0.13</td>
<td>1.00</td>
</tr>
</tbody>
</table>
```
Show the factors, sorted by factor loadings

```r
> print(f2, sort=TRUE)
Factor Analysis using method = minres
Call: fa(r = msq[1:72], nfactors = 2, cor = "poly")
Standardized loadings (pattern matrix) based upon correlation matrix

<table>
<thead>
<tr>
<th>item</th>
<th>MR1</th>
<th>MR2</th>
<th>h2</th>
<th>u2</th>
<th>com</th>
</tr>
</thead>
<tbody>
<tr>
<td>lively</td>
<td>0.89</td>
<td>-0.05</td>
<td>0.811</td>
<td>0.19</td>
<td>1.0</td>
</tr>
<tr>
<td>energetic</td>
<td>0.89</td>
<td>0.05</td>
<td>0.789</td>
<td>0.21</td>
<td>1.0</td>
</tr>
<tr>
<td>full.of.pep</td>
<td>0.89</td>
<td>-0.05</td>
<td>0.800</td>
<td>0.20</td>
<td>1.0</td>
</tr>
<tr>
<td>sluggish</td>
<td>-0.52</td>
<td>0.22</td>
<td>0.348</td>
<td>0.65</td>
<td>1.4</td>
</tr>
<tr>
<td>sleepy</td>
<td>-0.48</td>
<td>0.15</td>
<td>0.274</td>
<td>0.73</td>
<td>1.2</td>
</tr>
<tr>
<td>tired</td>
<td>-0.45</td>
<td>0.23</td>
<td>0.285</td>
<td>0.71</td>
<td>1.5</td>
</tr>
<tr>
<td>drowsy</td>
<td>-0.40</td>
<td>0.13</td>
<td>0.189</td>
<td>0.81</td>
<td>1.2</td>
</tr>
<tr>
<td>tense</td>
<td>0.14</td>
<td>0.85</td>
<td>0.714</td>
<td>0.29</td>
<td>1.1</td>
</tr>
<tr>
<td>frustrated</td>
<td>-0.10</td>
<td>0.83</td>
<td>0.718</td>
<td>0.28</td>
<td>1.0</td>
</tr>
<tr>
<td>ashamed</td>
<td>0.12</td>
<td>0.83</td>
<td>0.676</td>
<td>0.32</td>
<td>1.0</td>
</tr>
<tr>
<td>upset</td>
<td>-0.13</td>
<td>0.82</td>
<td>0.714</td>
<td>0.29</td>
<td>1.1</td>
</tr>
<tr>
<td>relaxed</td>
<td>0.44</td>
<td>-0.52</td>
<td>0.519</td>
<td>0.48</td>
<td>1.9</td>
</tr>
<tr>
<td>calm</td>
<td>0.26</td>
<td>-0.50</td>
<td>0.354</td>
<td>0.65</td>
<td>1.5</td>
</tr>
<tr>
<td>at.rest</td>
<td>0.38</td>
<td>-0.43</td>
<td>0.378</td>
<td>0.62</td>
<td>2.0</td>
</tr>
</tbody>
</table>

SS loadings 21.07 17.15
Proportion Var 0.29 0.24
Cumulative Var 0.29 0.53
Proportion Explained 0.55 0.45
Cumulative Proportion 0.55 1.00

With factor correlations of

<table>
<thead>
<tr>
<th>MR1</th>
<th>MR2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00</td>
<td>-0.13</td>
</tr>
<tr>
<td>-0.13</td>
<td>1.00</td>
</tr>
</tbody>
</table>
```

fa.plot(f2,labels=colnames(msq[1:72]),cex=.5,title="2 dimensions of the Motivational State Questionnaire")

2 dimensions of the Motivational State Questionnaire
Factor extraction and graphical displays

### Convert to and sort polar coordinates

\[
\text{round(polar(f2),2)}
\]

<table>
<thead>
<tr>
<th>Variable</th>
<th>Var</th>
<th>theta21</th>
<th>vec121</th>
</tr>
</thead>
<tbody>
<tr>
<td>strong</td>
<td>12</td>
<td>1.68</td>
<td>0.51</td>
</tr>
<tr>
<td>enthusiastic</td>
<td>14</td>
<td>1.89</td>
<td>0.75</td>
</tr>
<tr>
<td>anxious</td>
<td>71</td>
<td>69.09</td>
<td>0.63</td>
</tr>
<tr>
<td>nervous</td>
<td>45</td>
<td>71.26</td>
<td>0.69</td>
</tr>
<tr>
<td>angry</td>
<td>44</td>
<td>93.20</td>
<td>0.65</td>
</tr>
<tr>
<td>sorry</td>
<td>58</td>
<td>93.59</td>
<td>0.52</td>
</tr>
<tr>
<td>sad</td>
<td>16</td>
<td>101.25</td>
<td>0.68</td>
</tr>
<tr>
<td>blue</td>
<td>10</td>
<td>103.59</td>
<td>0.53</td>
</tr>
<tr>
<td>drowsy</td>
<td>51</td>
<td>162.17</td>
<td>0.18</td>
</tr>
<tr>
<td>sleepy</td>
<td>59</td>
<td>163.11</td>
<td>0.26</td>
</tr>
<tr>
<td>relaxed</td>
<td>8</td>
<td>310.03</td>
<td>0.46</td>
</tr>
<tr>
<td>at.rest</td>
<td>26</td>
<td>311.56</td>
<td>0.34</td>
</tr>
<tr>
<td>happy</td>
<td>61</td>
<td>347.14</td>
<td>0.73</td>
</tr>
<tr>
<td>pleased</td>
<td>60</td>
<td>349.61</td>
<td>0.64</td>
</tr>
<tr>
<td>alert</td>
<td>52</td>
<td>354.89</td>
<td>0.59</td>
</tr>
</tbody>
</table>
Graphical displays of hierarchical analysis

16 ability items from the International Cognitive Ability Resource

general ability and 4 subfactors of ICAR data
Schmid Leiman transformation of 16 ability items from ICAR

general ability and 4 subfactors of ICAR data
More types of reliability

1. $\alpha$ is a hodgepodge ratio of general factor and group factor reliability

2. $\omega_h$ (omega hierarchical) is an estimate of the general factor variance of a test

3. $\omega_t$ (omega total) is an estimate of the total reliable variance of a test

4. When do we use these?
   - When estimating how much of a test measures one thing.
     $\omega_h$
   - When estimating what is the total reliable variance in a test
     (when adjusting for test reliability in an SEM context)
Graphical displays of hierarchical analysis

\[ \omega_h \text{ and } \omega_t \text{ reliabilities} \]

```r
om <- omega(ability, nfactors=4)
```

```
om
Omega
Call: omega(m = ability, nfactors = 4)
Alpha: 0.83
G.6: 0.84
Omega Hierarchical: 0.65
Omega H asymptotic: 0.76
Omega Total 0.86

Schmid Leiman Factor loadings greater than 0.2

<table>
<thead>
<tr>
<th></th>
<th>F1*</th>
<th>F2*</th>
<th>F3*</th>
<th>F4*</th>
<th>h2</th>
<th>u2</th>
<th>p2</th>
</tr>
</thead>
<tbody>
<tr>
<td>reason.4</td>
<td>0.50</td>
<td>0.27</td>
<td>0.34</td>
<td>0.66</td>
<td>0.73</td>
<td></td>
<td></td>
</tr>
<tr>
<td>reason.16</td>
<td>0.42</td>
<td>0.21</td>
<td>0.23</td>
<td>0.77</td>
<td>0.76</td>
<td></td>
<td></td>
</tr>
<tr>
<td>reason.17</td>
<td>0.55</td>
<td>0.47</td>
<td>0.52</td>
<td>0.48</td>
<td>0.57</td>
<td></td>
<td></td>
</tr>
<tr>
<td>reason.19</td>
<td>0.44</td>
<td>0.21</td>
<td>0.25</td>
<td>0.75</td>
<td>0.77</td>
<td></td>
<td></td>
</tr>
<tr>
<td>letter.7</td>
<td>0.52</td>
<td>0.35</td>
<td>0.39</td>
<td>0.61</td>
<td>0.69</td>
<td></td>
<td></td>
</tr>
<tr>
<td>letter.33</td>
<td>0.46</td>
<td>0.30</td>
<td>0.31</td>
<td>0.69</td>
<td>0.70</td>
<td></td>
<td></td>
</tr>
<tr>
<td>letter.34</td>
<td>0.54</td>
<td>0.38</td>
<td>0.43</td>
<td>0.57</td>
<td>0.66</td>
<td></td>
<td></td>
</tr>
<tr>
<td>letter.58</td>
<td>0.47</td>
<td>0.20</td>
<td>0.28</td>
<td>0.72</td>
<td>0.78</td>
<td></td>
<td></td>
</tr>
<tr>
<td>matrix.45</td>
<td>0.40</td>
<td>0.20</td>
<td>0.66</td>
<td>0.59</td>
<td>0.41</td>
<td>0.27</td>
<td></td>
</tr>
<tr>
<td>matrix.46</td>
<td>0.40</td>
<td>0.26</td>
<td>0.24</td>
<td>0.76</td>
<td>0.65</td>
<td></td>
<td></td>
</tr>
<tr>
<td>matrix.47</td>
<td>0.42</td>
<td>0.23</td>
<td>0.77</td>
<td>0.79</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>matrix.55</td>
<td>0.28</td>
<td>0.12</td>
<td>0.88</td>
<td>0.65</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rotate.3</td>
<td>0.36</td>
<td>0.61</td>
<td>0.50</td>
<td>0.50</td>
<td>0.26</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rotate.4</td>
<td>0.41</td>
<td>0.61</td>
<td>0.54</td>
<td>0.46</td>
<td>0.31</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rotate.6</td>
<td>0.40</td>
<td>0.49</td>
<td>0.41</td>
<td>0.59</td>
<td>0.39</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rotate.8</td>
<td>0.32</td>
<td>0.53</td>
<td>0.40</td>
<td>0.60</td>
<td>0.26</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```
With eigenvalues of:
\[ g \quad F_1^* \quad F_2^* \quad F_3^* \quad F_4^* \]
\[ 3.04 \quad 1.32 \quad 0.46 \quad 0.42 \quad 0.55 \]

General/max = 2.3 \quad max/min = 3.17
Mean percent general = 0.58 \quad with sd = 0.2 \quad and \quad cv \quad of \quad 0.35

Explained Common Variance of the general factor = 0.53

The degrees of freedom are 62 \quad and \quad the \quad fit \quad is \quad 0.05
The number of observations was 1525 \quad with \quad Chi \quad Square = 70.19 \quad with \quad prob < 0.22
The root mean square of the residuals is 0.01
The df corrected root mean square of the residuals is 0.02
RMSEA index = 0.009 \quad and \quad the \quad 90% \quad confidence \quad intervals \quad are \quad 0.014
BIC = -384.25

Compare this with the adequacy of just a general factor and no group factors
The degrees of freedom for just the general factor are 104 \quad and \quad the \quad fit \quad is \quad 0.78
The number of observations was 1525 \quad with \quad Chi \quad Square = 1186.18 \quad with \quad prob < 5e-183
The root mean square of the residuals is 0.09
The df corrected root mean square of the residuals is 0.09
RMSEA index = 0.083 \quad and \quad the \quad 90% \quad confidence \quad intervals \quad are \quad 0.078 \quad 0.085
BIC = 423.88

Measures of factor score adequacy
\[ g \quad F_1^* \quad F_2^* \quad F_3^* \quad F_4^* \]
Correlation of scores with factors \quad 0.83 \quad 0.80 \quad 0.53 \quad 0.56 \quad 0.71
Multiple R square of scores with factors \quad 0.69 \quad 0.64 \quad 0.28 \quad 0.32 \quad 0.50
Minimum correlation of factor score estimates \quad 0.37 \quad 0.28 \quad -0.44 \quad -0.37 \quad 0.00

Total, General and Subset omega for each subset
\[ g \quad F_1^* \quad F_2^* \quad F_3^* \quad F_4^* \]
Data set from Preacher & Hayes (2004)

```r
# from Preacher and Hayes (2004)
sobel <- structure(list(SATIS = c(-0.59, 1.3, 0.02, 0.01, 0.79, -0.35, -0.03, 1.75, -0.8, -1.2, -1.27, 0.7, -1.59, 0.68, -0.39, 1.33,
  ...
  "Therapy", "Attributional Positivity"), .Names = c("SATIS", "THERAPY", "ATTRIB"))

# n.iter set to 50 (instead of default of 5000) for speed of example
mediate(1,2,3,sobel,n.iter=50)  # The example in Preacher and Hayes
```

The DV (Y) was SATIS. The IV (X) was THERAPY. The mediating variable(s) = ATTRIB.

Total Direct effect (c) of THERAPY on SATIS = 0.76  S.E. = 0.31  t direct = 2.5
Direct effect (c') of THERAPY on SATIS removing ATTRIB = 0.43  S.E. = 0.32  t direct = 1.35
Indirect effect (ab) of THERAPY on SATIS through ATTRIB = 0.33
Mean bootstrapped indirect effect = 0.31 with standard error = 0.16  Lower CI = 0.07  Upper CI = 0.68
R2 of model = 0.31
To see the longer output, specify short = FALSE in the print statement

Full output

Total effect estimates (c)

<table>
<thead>
<tr>
<th></th>
<th>SATIS</th>
<th>se</th>
<th>t</th>
<th>Prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>THERAPY</td>
<td>0.76</td>
<td>0.31</td>
<td>2.5</td>
<td>0.0186</td>
</tr>
</tbody>
</table>

Direct effect estimates (c')

<table>
<thead>
<tr>
<th></th>
<th>SATIS</th>
<th>se</th>
<th>t</th>
<th>Prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>THERAPY</td>
<td>0.43</td>
<td>0.32</td>
<td>1.35</td>
<td>0.190</td>
</tr>
<tr>
<td>ATTRIB</td>
<td>0.40</td>
<td>0.18</td>
<td>2.23</td>
<td>0.034</td>
</tr>
</tbody>
</table>

'a' effect estimates

<table>
<thead>
<tr>
<th></th>
<th>THERAPY</th>
<th>se</th>
<th>t</th>
<th>Prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>THERAPY</td>
<td>0.82</td>
<td>0.3</td>
<td>2.74</td>
<td>0.0106</td>
</tr>
</tbody>
</table>
The Preacher mediation example

Mediation model

\[ c = 0.76 \]
\[ c' = 0.43 \]
Take the data example from Hayes (2013)

```r
C.pmi <- cov(Tal_Or)
#n.iter set to 50 (instead of default of 5000) for speed of example
mediate(y="reaction",x = "cond",m=c("pmi","import"),data=C.pmi,n.obs=123,n.iter=50)
```

The DV (Y) was reaction. The IV (X) was cond. The mediating variable(s) = pmi import.

Total Direct effect (c) of cond on reaction = 0.5 S.E. = 0.28 t direct = 1.79
Direct effect (c') of cond on reaction removing pmi import = 0.1 S.E. = 0.24 t = 0.43
Indirect effect (ab) of cond on reaction through pmi import = 0.39
Mean bootstrapped indirect effect = 0.7 with standard error = 0.17 Lower CI = 0.39
R^2 of model = 0.33

To see the longer output, specify short = FALSE in the print statement

Full output

Total effect estimates (c)

<table>
<thead>
<tr>
<th>Reaction</th>
<th>se</th>
<th>t</th>
<th>Prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>cond</td>
<td>0.5</td>
<td>0.28</td>
<td>1.79</td>
</tr>
</tbody>
</table>

Direct effect estimates (c')

<table>
<thead>
<tr>
<th>Reaction</th>
<th>se</th>
<th>t</th>
<th>Prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>cond</td>
<td>0.10</td>
<td>0.24</td>
<td>0.43</td>
</tr>
<tr>
<td>pmi</td>
<td>0.40</td>
<td>0.09</td>
<td>4.26</td>
</tr>
<tr>
<td>import</td>
<td>0.32</td>
<td>0.07</td>
<td>4.59</td>
</tr>
</tbody>
</table>

'a' effect estimates

<table>
<thead>
<tr>
<th>Reaction</th>
<th>se</th>
<th>t</th>
<th>Prob</th>
</tr>
</thead>
<tbody>
<tr>
<td>cond</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The Hayes (2013) example mediation

Mediation model

\[ \text{cond} \rightarrow \text{pmi} \rightarrow \text{reaction} \]

- \( c = 0.5 \)
- \( c' = 0.1 \)
- \( \beta = 0.48 \)
- \( \omega = 0.4 \)
- \( \alpha = 0.63 \)
- \( \gamma = 0.32 \)
The “New” Psychometrics

1. Classical Test theory examines responses assuming items are equivalent, or at least congeneric equivalent.
2. Item Response Theory models item difficulty as well as item discrimination.
3. Although seemingly very different models, factor analysis of categorical items (using tetrachoric or polychoric correlations) is equivalent to IRT 2 PL models.
4. Rasch model is just a 1 PL model where items differ in difficulty, but not discrimination.
5. 2PL has difficulty and discrimination estimated from factor analysis of tetrachoric/polychoric items.
R code

```r
f1 <- irt.fa(ability)

> f1
Item Response Analysis using Factor Analysis

Call: irt.fa(x = ability)
Item Response Analysis using Factor Analysis

Summary information by factor and item
Factor = 1

<table>
<thead>
<tr>
<th>Item</th>
<th>-3</th>
<th>-2</th>
<th>-1</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>reason.4</td>
<td>0.05</td>
<td>0.24</td>
<td>0.64</td>
<td>0.53</td>
<td>0.16</td>
<td>0.03</td>
<td>0.01</td>
</tr>
<tr>
<td>reason.16</td>
<td>0.08</td>
<td>0.22</td>
<td>0.38</td>
<td>0.31</td>
<td>0.14</td>
<td>0.05</td>
<td>0.01</td>
</tr>
<tr>
<td>reason.17</td>
<td>0.08</td>
<td>0.33</td>
<td>0.69</td>
<td>0.42</td>
<td>0.11</td>
<td>0.02</td>
<td>0.00</td>
</tr>
<tr>
<td>reason.19</td>
<td>0.06</td>
<td>0.17</td>
<td>0.35</td>
<td>0.36</td>
<td>0.19</td>
<td>0.07</td>
<td>0.02</td>
</tr>
<tr>
<td>letter.7</td>
<td>0.05</td>
<td>0.18</td>
<td>0.41</td>
<td>0.44</td>
<td>0.20</td>
<td>0.06</td>
<td>0.02</td>
</tr>
<tr>
<td>letter.33</td>
<td>0.05</td>
<td>0.15</td>
<td>0.31</td>
<td>0.36</td>
<td>0.20</td>
<td>0.08</td>
<td>0.02</td>
</tr>
<tr>
<td>letter.34</td>
<td>0.05</td>
<td>0.19</td>
<td>0.45</td>
<td>0.46</td>
<td>0.20</td>
<td>0.06</td>
<td>0.01</td>
</tr>
<tr>
<td>letter.58</td>
<td>0.02</td>
<td>0.09</td>
<td>0.30</td>
<td>0.53</td>
<td>0.35</td>
<td>0.12</td>
<td>0.03</td>
</tr>
<tr>
<td>matrix.45</td>
<td>0.05</td>
<td>0.11</td>
<td>0.19</td>
<td>0.23</td>
<td>0.17</td>
<td>0.09</td>
<td>0.04</td>
</tr>
</tbody>
</table>

Test Info 0.67 2.11 4.73 5.83 5.28 2.55 0.69

SEM 1.22 0.69 0.46 0.41 0.44 0.63 1.20

Reliability -0.49 0.53 0.79 0.83 0.81 0.61 -0.45

Factor analysis with Call: fa(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate, fm = fm)

Test of the hypothesis that 1 factor is sufficient.
The degrees of freedom for the model is 104 and the objective function was 1.91
The number of observations was 1525 with Chi Square = 2893.45 with prob < 0
The root mean square of the residuals (RMSA) is 0.08
The df corrected root mean square of the residuals is 0.09
Tucker Lewis Index of factoring reliability = 0.723
RMSEA index = 0.133 and the 90% confidence intervals are 0.128 0.135
BIC = 2131.15
```
FA solution with tetrachoric correlations

Item information from factor analysis
IRT based scoring and Classical Test Theory based scoring

1. CTT and IRT based scores correlate almost perfectly without missing data
2. With lots of missing data, and different items for different people, IRT based scores provide more subtle distinctions.
3. `scoreIrt.2pl` and `scoreIrt.1pl` will do IRT based scores.
4. By default, will find the irt based parameters and then do the scoring.

```r
ability.irt <- irt.fa(ability)
ablety.scores <- scoreIrt(ability.irt,ability)
pairs.panels(ability.scores, smoother=TRUE)
```
CTT and IRT based scores are almost identical
Multilevel reliability

1. Classic reliability measures assess the variance of between person differences compared to error of the measurement.

\[ \rho_{xx} = \frac{1 - \sigma_e^2}{\sigma_x^2} \]  

(1)

2. Multilevel reliability is a series of generalizability coefficients, generalizing over items, over time, time x items,

\[ R_{kF} = \frac{\sigma_{id}^2 + (\sigma_{idxitems}^2/m)}{\sigma_{id}^2 + (\sigma_{idxitems}^2/m) + \sigma_{error}^2/(km)} \]  

(2)

From Equation 6 (Shrout & Lane, 2012, p 310). See Shrout & Lane (2012) for five other generalizability formula.

3. Implemented in psych as mlr or multilevel.reliability

4. Also simulations using sim.multi

5. I show the data from Fisher (2015) who reports 10 subjects measured over 60 (or more) days on 28 affect items.

6. (Download the R data files, minor rearrangement and reliability measurements.)
Table: The `multilevel.reliability` function estimates of the generalizability coefficients for the positively and negatively valenced items from Fisher (2015). RkF is the reliability of average of all ratings across all items and times (Fixed time effects), R1R is the generalizability of a single time point across all items (Random time effects), RkR is the generalizability of average time points across all items (Random time effects), Rc is the generalizability of change (fixed time points, fixed items), RkRn is the generalizability of between person differences averaged over time (time nested within people) and Rcn is the generalizability of within person variations averaged over items (time nested within people).

<table>
<thead>
<tr>
<th>Coefficient</th>
<th>Positive items</th>
<th>Negative items</th>
</tr>
</thead>
<tbody>
<tr>
<td>RkF</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>R1R</td>
<td>0.80</td>
<td>0.77</td>
</tr>
<tr>
<td>RkR</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>Rc</td>
<td>0.72</td>
<td>0.71</td>
</tr>
<tr>
<td>RkRn</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>Rcn</td>
<td>0.64</td>
<td>0.59</td>
</tr>
</tbody>
</table>
Assessing reliability of within subject differences in affect. Data from Fisher (2015)
psych includes some very old ideas

1. Schmid-Leiman (Schmid & Leiman, 1957) transformations from correlated factor structures to higher order structures.

2. Dwyer extension (Dwyer, 1937; Mosier, 1938; Horn, 1973) to extend a factor solution to more variables.

3. This can be used to extend other variables into a factor space, or to relate two domains to each other.
Extend a data set into another

First, create the data set

```r
set.seed(42)
d <- sim.item(12)  # two orthogonal factors
R <- cor(d)
Ro <- R[c(1,2,4,5,7,8,10,11),c(1,2,4,5,7,8,10,11)]
Roe <- R[c(1,2,4,5,7,8,10,11),c(3,6,9,12)]
fo <- fa(Ro, 2)
fe <- fa.extension(Roe, fo)
fa.diagram(fo, fe=fe)
```

fe

Call: fa.extension(Roe = Roe, fo = fo)
Standardized loadings (pattern matrix) based upon correlation matrix

<table>
<thead>
<tr>
<th></th>
<th>MR1</th>
<th>MR2</th>
<th>h2</th>
<th>u2</th>
</tr>
</thead>
<tbody>
<tr>
<td>V3</td>
<td>0.63</td>
<td>-0.02</td>
<td>0.39</td>
<td>0.61</td>
</tr>
<tr>
<td>V6</td>
<td>0.04</td>
<td>-0.61</td>
<td>0.37</td>
<td>0.63</td>
</tr>
<tr>
<td>V9</td>
<td>-0.61</td>
<td>0.01</td>
<td>0.38</td>
<td>0.62</td>
</tr>
<tr>
<td>V12</td>
<td>-0.06</td>
<td>0.58</td>
<td>0.33</td>
<td>0.67</td>
</tr>
</tbody>
</table>

SS loadings

<table>
<thead>
<tr>
<th></th>
<th>MR1</th>
<th>MR2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.77</td>
<td>0.69</td>
</tr>
</tbody>
</table>

Proportion Var

<table>
<thead>
<tr>
<th></th>
<th>0.19</th>
<th>0.17</th>
</tr>
</thead>
</table>

Cumulative Var

|     | 0.19 | 0.37|

Proportion Explained

|     | 0.53 | 0.47|

Cumulative Proportion

|     | 0.53 | 1.00|
**Factor extension**

*Factor analysis and extension*

![Diagram](image_url)
Hierarchical factor analysis

R code

```r
neo52 <- fa.multi(neo, 5, 2)
fa.multi.diagram(neo52)
```
Hierarchical (multilevel) Structure

\[ V \leftarrow \text{Descriptive and inferential statistics} \]

\[ \text{Scores and Reliability} \quad \beta, \omega \quad \text{EFA} \quad \omega \quad \text{Mediation} \quad \text{IRT} \quad \text{multilevel} \quad \text{FA and beyond} \]

\[ N_1 \]
\[ N_3 \]
\[ N_4 \]
\[ N_6 \]
\[ N_2 \]
\[ N_5 \]
\[ N_1 \]
\[ N_3 \]
\[ N_4 \]
\[ N_6 \]
\[ N_2 \]
\[ N_5 \]
\[ C_5 \]
\[ C_4 \]
\[ C_2 \]
\[ C_3 \]
\[ C_1 \]
\[ C_6 \]
\[ E_4 \]
\[ A_4 \]
\[ A_2 \]
\[ A_5 \]
\[ E_5 \]
\[ A_6 \]
\[ A_1 \]
\[ E_3 \]
\[ E_6 \]
\[ E_1 \]
\[ A_3 \]
\[ E_2 \]
\[ O_5 \]
\[ O_2 \]
\[ O_1 \]
\[ O_3 \]
\[ O_4 \]

\[ \text{References} \]

\[ MR1 \]
\[ MR2 \]
ESEM can be thought of as factor extension from A to B and B to A

1. If we have two sets of variables that show factor structures within each set
2. And then link the factor structures.
3. This can be done in SEM, but here show how to do exploratory SEM
4. We make up a toy data set
# make up a sem like problem using sim.structure

```r
fx <- matrix(c(.9, .8, .6, rep(0, 4), .6, .8, -.7), ncol=2)
fy <- matrix(c(.6, .5, .4), ncol=1)
rownames(fx) <- c("V", "Q", "A", "nach", "Anx")
rownames(fy) <- c("gpa", "Pre", "MA")
Phi <- matrix(c(1, 0, .7, .0, 1, .7, .7, .7, 1), ncol=3)
gre.gpa <- sim.structural(fx, Phi, fy)
print(gre.gpa)
```

**Call:** sim.structural(fx = fx, Phi = Phi, fy = fy)

### model (Population correlation matrix)

<table>
<thead>
<tr>
<th></th>
<th>V</th>
<th>Q</th>
<th>A</th>
<th>nach</th>
<th>Anx</th>
<th>gpa</th>
<th>Pre</th>
<th>MA</th>
</tr>
</thead>
<tbody>
<tr>
<td>V</td>
<td>1.00</td>
<td>0.72</td>
<td>0.54</td>
<td>0.00</td>
<td>0.00</td>
<td>0.38</td>
<td>0.32</td>
<td>0.25</td>
</tr>
<tr>
<td>Q</td>
<td>0.72</td>
<td>1.00</td>
<td>0.48</td>
<td>0.00</td>
<td>0.00</td>
<td>0.34</td>
<td>0.28</td>
<td>0.22</td>
</tr>
<tr>
<td>A</td>
<td>0.54</td>
<td>0.48</td>
<td>1.00</td>
<td>0.48</td>
<td>-0.42</td>
<td>0.50</td>
<td>0.42</td>
<td>0.34</td>
</tr>
<tr>
<td>nach</td>
<td>0.00</td>
<td>0.00</td>
<td>0.48</td>
<td>1.00</td>
<td>-0.56</td>
<td>0.34</td>
<td>0.28</td>
<td>0.22</td>
</tr>
<tr>
<td>Anx</td>
<td>0.00</td>
<td>0.00</td>
<td>-0.42</td>
<td>-0.56</td>
<td>1.00</td>
<td>-0.29</td>
<td>-0.24</td>
<td>-0.20</td>
</tr>
<tr>
<td>gpa</td>
<td>0.38</td>
<td>0.34</td>
<td>0.50</td>
<td>0.34</td>
<td>-0.29</td>
<td>1.00</td>
<td>0.30</td>
<td>0.24</td>
</tr>
<tr>
<td>Pre</td>
<td>0.32</td>
<td>0.28</td>
<td>0.42</td>
<td>0.28</td>
<td>-0.24</td>
<td>0.30</td>
<td>1.00</td>
<td>0.20</td>
</tr>
<tr>
<td>MA</td>
<td>0.25</td>
<td>0.22</td>
<td>0.34</td>
<td>0.22</td>
<td>-0.20</td>
<td>0.24</td>
<td>0.20</td>
<td>1.00</td>
</tr>
</tbody>
</table>

### reliability (population reliability)

<table>
<thead>
<tr>
<th></th>
<th>V</th>
<th>Q</th>
<th>A</th>
<th>nach</th>
<th>Anx</th>
<th>gpa</th>
<th>Pre</th>
<th>MA</th>
</tr>
</thead>
<tbody>
<tr>
<td>V</td>
<td>0.81</td>
<td>0.64</td>
<td>0.72</td>
<td>0.64</td>
<td>0.49</td>
<td>0.36</td>
<td>0.25</td>
<td></td>
</tr>
<tr>
<td>Q</td>
<td></td>
<td>0.64</td>
<td>0.72</td>
<td>0.64</td>
<td>0.49</td>
<td>0.36</td>
<td>0.25</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td></td>
<td></td>
<td>0.72</td>
<td>0.64</td>
<td>0.49</td>
<td>0.36</td>
<td>0.25</td>
<td>0.16</td>
</tr>
<tr>
<td>nach</td>
<td></td>
<td></td>
<td></td>
<td>0.64</td>
<td>0.49</td>
<td>0.36</td>
<td>0.25</td>
<td>0.16</td>
</tr>
<tr>
<td>Anx</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.49</td>
<td>0.36</td>
<td>0.25</td>
<td>0.16</td>
</tr>
<tr>
<td>gpa</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.36</td>
<td>0.25</td>
<td>0.16</td>
</tr>
<tr>
<td>Pre</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.25</td>
<td>0.16</td>
</tr>
<tr>
<td>MA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.16</td>
</tr>
</tbody>
</table>
Exploratory Structural Equation Modelling

```
example <- esem(gre.gpa$model, varsX=1:5, varsY=6:8, nfX=2, nfY=1, n.obs=1000, plot=FALSE)
```

> example

Exploratory Structural Equation Modeling Analysis using method = minres

Call: esem(r = gre.gpa$model, varsX = 1:5, varsY = 6:8, nfX = 2, nfY = 1, 
  n.obs = 1000, plot = FALSE)

For the 'X' set:

<table>
<thead>
<tr>
<th></th>
<th>MR1</th>
<th>MR2</th>
</tr>
</thead>
<tbody>
<tr>
<td>V</td>
<td>0.91</td>
<td>-0.06</td>
</tr>
<tr>
<td>Q</td>
<td>0.81</td>
<td>-0.05</td>
</tr>
<tr>
<td>A</td>
<td>0.53</td>
<td>0.57</td>
</tr>
<tr>
<td>nach</td>
<td>-0.10</td>
<td>0.81</td>
</tr>
<tr>
<td>Anx</td>
<td>0.08</td>
<td>-0.71</td>
</tr>
</tbody>
</table>

For the 'Y' set:

<table>
<thead>
<tr>
<th></th>
<th>MR1</th>
</tr>
</thead>
<tbody>
<tr>
<td>gpa</td>
<td>0.6</td>
</tr>
<tr>
<td>Pre</td>
<td>0.5</td>
</tr>
<tr>
<td>MA</td>
<td>0.4</td>
</tr>
</tbody>
</table>

Correlations between the X and Y sets.

<table>
<thead>
<tr>
<th></th>
<th>X1</th>
<th>X2</th>
<th>Y1</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1</td>
<td>1.00</td>
<td>0.19</td>
<td>0.68</td>
</tr>
<tr>
<td>X2</td>
<td>0.19</td>
<td>1.00</td>
<td>0.67</td>
</tr>
<tr>
<td>Y1</td>
<td>0.68</td>
<td>0.67</td>
<td>1.00</td>
</tr>
</tbody>
</table>

The degrees of freedom for the null model are 56 and the empirical chi square function was 6930.29 with p < 0.0027.

The degrees of freedom for the model are 7 and the empirical chi square function was 21.83 with p < 0.0027.

Empirical BIC = -26.53
ESABIC = -4.29
Fit based upon off diagonal values = 1

To see the item loadings for the X and Y sets combined, and the associated FA output, print with short = FALSE.
ESEM of our toy problem

Exploratory Structural Model

![Diagram of ESEM model with variables and paths and correlations labeled]
Outline

Part I: What is R, where did it come from, why use it
- Installing R and adding packages: the building blocks of R

Part II: A brief introduction – an overview
- R is just a fancy (very fancy) calculator
- Descriptive data analysis
- Some inferential analysis

Part III R is a powerful statistical system
- Data entry (detail and practice)
- Descriptive (again)
- Inferential (t and F with more practice)
- Regression
- Basic R commands

Part IV: Psychometrics
- Reliability and its discontents
- EFA, CFA, SEM

Part V: Help and More Help
- List of useful commands

Part VI: The psych package and more practice


