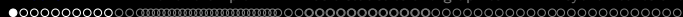


Outline I

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





Where did it come from, why use R?

R: Statistics for all us

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



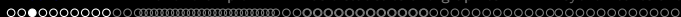


[Where did it come from, why use R?](#)

R: What is it?

- ➊ R: An international collaboration
- ➋ R: The open source - public domain version of S+
- ➌ R: Written by statistician (and all of us) for statisticians (and the rest of us)
- ➍ R: Not just a statistics system, also an extensible language.
 - This means that as new statistics are developed they tend to appear in R far sooner than elsewhere.
 - R facilitates asking questions that have not already been asked.





Where did it come from, why use R?

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 R?

Statistical Programs for Psychologists

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





Where did it come from, why use R?

R: A way of thinking

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

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



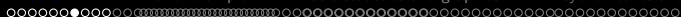


Where did it come from, why use R?

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.”





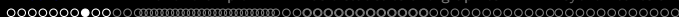
Where did it come from, why use R?

What is R?: Technically

- R is an open source implementation of S (S-Plus is a commercial implementation)
- R is available under GNU Copy-left
- The current version of R is 2.13.0
- R is a group project run by a core group of developers (with new releases semiannually)

(Adapted from Robert Gentleman)





Where did it come from, why use R?

R: A brief history

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



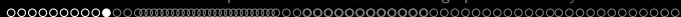


Where did it come from, why use R?

Misconception: R is hard to use

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



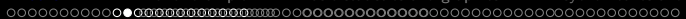


Where did it come from, why use R?

Misconceptions: R is hard to learn

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





Installing R on your computer and adding packages

Go to the R.project.org

The R Project for Statistical Computing

http://www.r-project.org

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

The R Project for Statistical Computing

PCA 5 vars
prcomp(x = data.co = cor)

(1-3) 60%

Clustering 4 groups

Factor 1 [41%]

Factor 3 [19%]

Groups
28
16
1
2

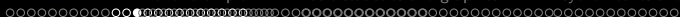
Getting Started:

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

News:

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





Installing R on your computer and adding packages

Go to the Comprehensive R Archive Network (CRAN)

The Comprehensive R Archive Network

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

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

The Comprehensive R Archive Network

Frequently used pages

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:

- [Linux](#)
- [MacOS X](#)
- [Windows](#)

Source Code for all Platforms

Windows and Mac users most likely want 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** (2011-04-13): [R-2.13.0.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](#)

CRAN

- [Mirrors](#)
- [What's new?](#)
- [Task Views](#)
- [Search](#)

About R

- [R Homepage](#)
- [The R Journal](#)

Software

- [R Sources](#)
- [R Binaries](#)
- [Packages](#)
- [Other](#)

Documentation

- [Manuals](#)
- [FAQs](#)
- [Contributed](#)






Installing R on your computer and adding packages

Download and install the appropriate version – PC

The Comprehensive R Archive Network

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

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



R for Windows

Subdirectories:

- [base](#) Binaries for base distribution (managed by Duncan Murdoch)
- [contrib](#) Binaries of contributed packages (managed by Uwe Ligges)

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

You may also want to read the [R FAQ](#) and [R for Windows FAQ](#).

Note: CRAN does some checks on these binaries for viruses, but cannot give guarantees. Use the normal precautions with downloaded executables.

CRAN

- [Mirrors](#)
- [What's new?](#)
- [Task Views](#)
- [Search](#)

About R

- [R Homepage](#)
- [The R Journal](#)

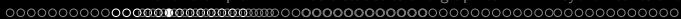
Software

- [R Sources](#)
- [R Binaries](#)
- [Packages](#)
- [Other](#)

Documentation

- [Manuals](#)
- [FAQs](#)
- [Contributed](#)





[Installing R on your computer and adding packages](#)

Starting R on a PC

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

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

Natural language support but running in an English locale

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

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

> |
```

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

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

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

Natural language support but running in an English locale

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

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

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

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





Installing R on your computer and adding packages

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

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

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

Natural language support but running in an English locale

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

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

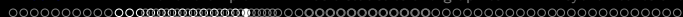
> install.packages("ctv")
--- Please select a CRAN mirror for use in this session ---
trying URL 'http://cran.stat.ucla.edu/bin/windows/contrib/2.13/ctv_0.7-2.zip'
Content type 'application/zip' length 298753 bytes (291 Kb)
opened URL
downloaded 291 Kb

package 'ctv' successfully unpacked and MD5 sums checked

The downloaded packages are in
  C:\users\revelle\Temp\RtmpwNzUtt\downloaded_packages
> library(ctv)
> |
```

Use the
package
menu to
select a
mirror





A small subset of very useful packages

- General use
 - core R
 - MASS
 - lattice
 - lme4 (core)
 - psych
 - Zelig
- Special use
 - ltm
 - sem
 - lavaan
 - OpenMx
 - GPArotation
 - mvtnorm
 - > 3000 known
 - + ?
- General applications
 - most descriptive and inferential stats
 - Modern Applied Statistics with S
 - Lattice or Trellis graphics
 - Linear mixed-effects models
 - Personality and psychometrics
 - General purpose toolkit
- More specialized packages
 - Latent Trait Model (IRT)
 - SEM and CFA (one group)
 - SEM and CFA (multiple groups)
 - SEM and CFA (multiple groups +)
 - Jennrich rotations
 - Multivariate distributions
 - Thousands of more packages on CRAN
 - Code on webpages/journal articles



More on distributions

We can find the probability of normal scores from -3 to 3

```
z <- seq(from=-3,to= 3, by = .5)
```

```
z
```

```
round(pnorm(z),digits=2)
```

```
z
```

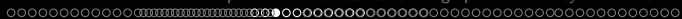
```
[1] -3.0 -2.5 -2.0 -1.5 -1.0 -0.5  0.0  0.5  1.0  1.5  2.0  2.5  3.0
```

```
> round(pnorm(z),digits=2)
```

```
[1] 0.00 0.01 0.02 0.07 0.16 0.31 0.50 0.69 0.84 0.93 0.98 0.99 1.00
```

Try this again with `by=.1`





Make a data frame out of the results

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

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





Add the ordinate of the normal curve to this data frame

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

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





Compare the z distribution with the t distribution with 10 df

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

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

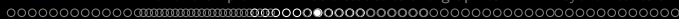


R is a set of distributions. Don't buy a stats book with tables!

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

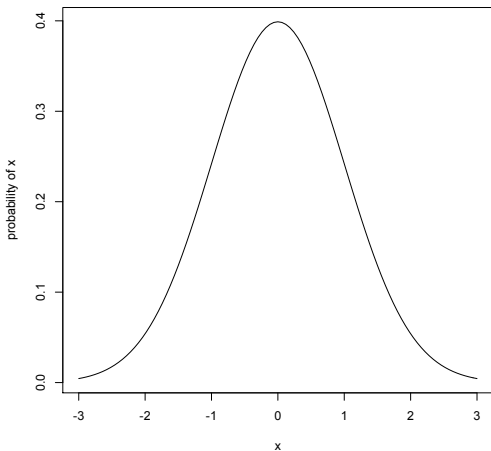
Distribution	base name	P 1	P 2	P 3	example application
<i>Normal</i>	norm	mean	sigma		Most data
<i>Multivariate normal</i>	mvnorm	mean	r	sigma	Most data
<i>Log Normal</i>	lnorm	log mean	log sigma		income or reaction time
<i>Uniform</i>	unif	min	max		rectangular distributions
<i>Binomial</i>	binom	size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





R can draw distributions

A normal curve



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

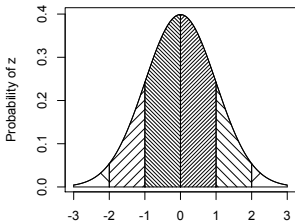




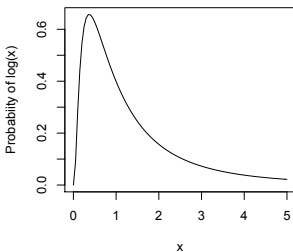
Basic R capabilities: Calculation, Statistical tables, Graphics

R can draw more interesting distributions

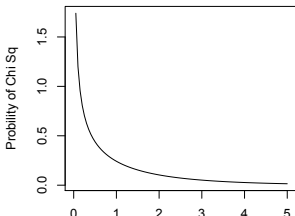
The normal curve



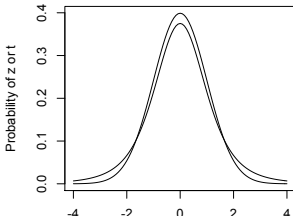
Log normal



Chi Square distribution



Normal and t with 4 df





Basic R capabilities: Calculation, Statistical tables, Graphics

R is also a graphics calculator

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

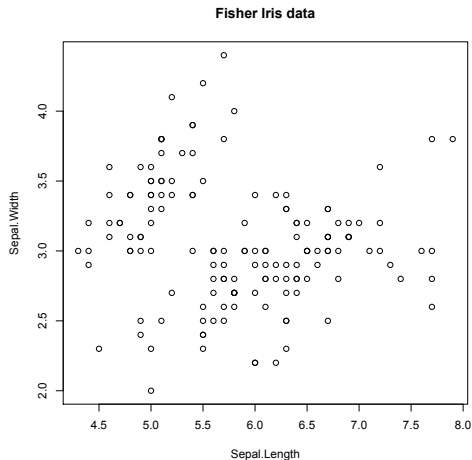
```
op <- par(mfrow=c(2,2))          #set up a 2 x 2 graph
curve(dnorm(x),-3,3,xlab="",ylab="Probability of z")
title(main="The normal curve",outer=FALSE)
xvals <- seq(-3,-2,length=100)
dvals <- dnorm(xvals)
polygon(c(xvals,rev(xvals)),c(rep(0,100),rev(dvals)),density=2,angle=-45)
xvals <- seq(-2,-1,length=100)
dvals <- dnorm(xvals)
polygon(c(xvals,rev(xvals)),c(rep(0,100),rev(dvals)),density=14,angle=45)
xvals <- seq(-1,-0,length=100)
dvals <- dnorm(xvals)
polygon(c(xvals,rev(xvals)),c(rep(0,100),rev(dvals)),density=34,angle=-45)
xvals <- seq(2,3,length=100)
dvals <- dnorm(xvals)
polygon(c(xvals,rev(xvals)),c(rep(0,100),rev(dvals)),density=2,angle=45)
xvals <- seq(1,2,length=100)
dvals <- dnorm(xvals)
polygon(c(xvals,rev(xvals)),c(rep(0,100),rev(dvals)),density=14,angle=-45)
xvals <- seq(0,1,length=100)
dvals <- dnorm(xvals)
polygon(c(xvals,rev(xvals)),c(rep(0,100),rev(dvals)),density=34,angle=45)
curve(dlnorm(x),0,5,ylab='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)
op <- par(mfrow=c(1,1))
```





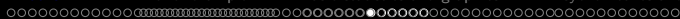
Basic R capabilities: [Calculation](#), [Statistical tables](#), [Graphics](#)

A simple scatter plot using plot



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





2 x 2 measures of association

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



Another way of looking at the data: Fisher exact test

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

Fisher's Exact Test for Count Data

data: Nach

p-value = 0.03808

alternative hypothesis: true odds ratio is not equal to 1

95 percent confidence interval:

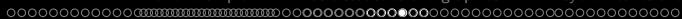
1.079216 32.685682

sample estimates:

odds ratio

5.433516





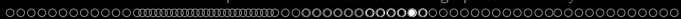
Some simple 2 x 2 data analysis

What about the phi measure of association?

```
Nach  
phi(Nach)
```

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





Some simple 2 x 2 data analysis

If we can assume normality, apply the tetrachoric coefficient

```
tetrachoric(Nach)
```

```
> Nach
```

```
      low high
quit   12   5
persist 5  12
```

```
> phi(Nach)
```

```
[1] 0.41
```

```
> tetrachoric(Nach)
```

```
Call: tetrachoric(x = Nach)
```

```
tetrachoric correlation
```

```
[1] 0.6
```

```
with tau of
```

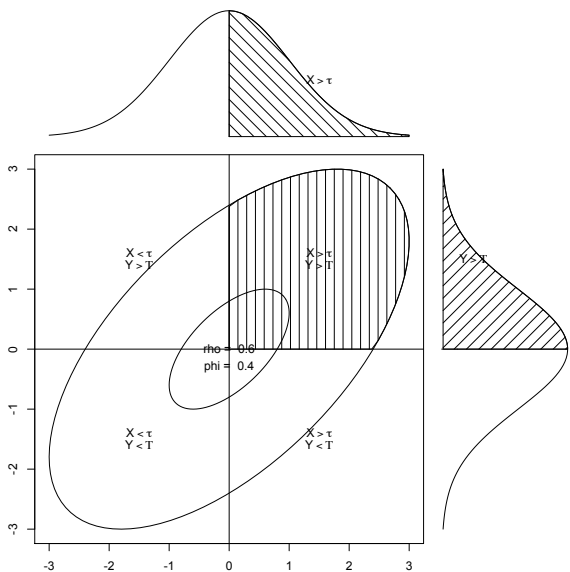
```
quit low
  0    0
```

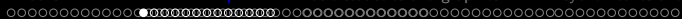




Some simple 2 x 2 data analysis

The tetrachoric correlation assumes normality with dichotomous cuts





A brief example with real data

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

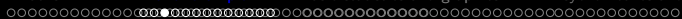
Get the data and describe it

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

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

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

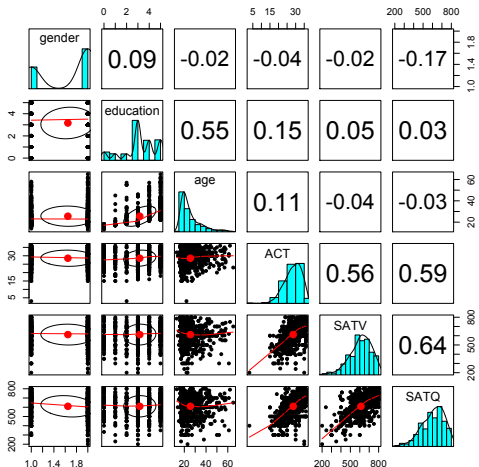


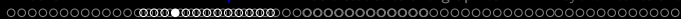


A brief example of exploratory and confirmatory data analysis

Graphic display of data using `pairs.panels`

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





A brief example of exploratory and confirmatory data analysis

Clean up the data using scrub

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

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



Test the correlations for significance using `corr.test`

```
> corr.test(cleaned)
```

```
Call:corr.test(x = cleaned)
```

```
Correlation matrix
```

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

```
Sample Size
```

	gender	education	age	ACT	SATV	SATQ
gender	700	700	700	699	700	687
...						
SATQ	687	687	687	686	687	687

```
Probability value
```

	gender	education	age	ACT	SATV	SATQ
gender	0.00	0.02	0.58	0.21	0.62	0.00
education	0.02	0.00	0.00	0.00	0.22	0.36
age	0.58	0.00	0.00	0.00	0.26	0.37
ACT	0.21	0.00	0.00	0.00	0.00	0.00
SATV	0.62	0.22	0.26	0.00	0.00	0.00
SATQ	0.00	0.36	0.37	0.00	0.00	0.00



Multiple regression

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

```
mod1 <- lm(SATV ~ education + gender + SATQ,data=my.data)
> summary(mod1,digits=2)
```

Call:

```
lm(formula = SATV ~ education + gender + SATQ, data = my.data)
```

Residuals:

Min	1Q	Median	3Q	Max
-372.91	-49.08	2.30	53.68	251.93

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	180.87348	23.41019	7.726	3.96e-14 ***
education	1.24043	2.32361	0.534	0.59363
gender	20.69271	6.99651	2.958	0.00321 **
SATQ	0.64489	0.02891	22.309	< 2e-16 ***

Signif. codes: 0 '***' 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



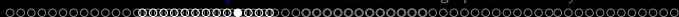
Zero center the data before examining interactions

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`.

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

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





Zero center the data before examining interactions

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

Call:

```
lm(formula = SATV ~ education * gender * SATQ, data = zsat)
```

Residuals:

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

Coefficients:

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

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



Compare model 1 and model 2

Test the difference between the two linear models

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

Analysis of Variance Table

```
Model 1: SATV ~ education + gender + SATQ
```

```
Model 2: SATV ~ education * gender * SATQ
```

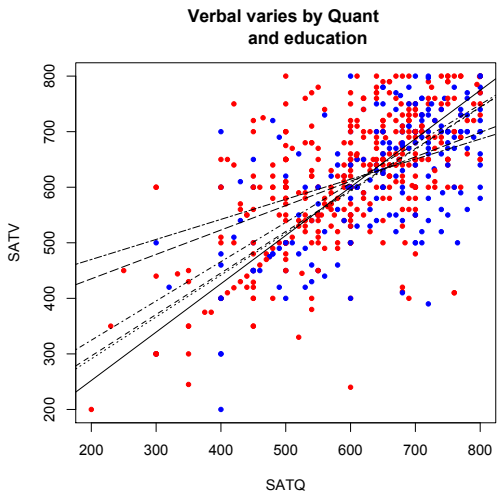
	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
1	683	5079984				
2	679	4870243	4	209742	7.3104	9.115e-06 ***

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




A brief example of exploratory and confirmatory data analysis

Show the regression lines by education



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

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





4 steps: read, explore, test, graph

Data entry overview

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



Examples of built in data sets from the psych package

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





Reading data from another program –using the clipboard

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

```
my.data <- read.clipboard()      #assumes headers and tab or space delimited
my.data <- read.clipboard.csv()  #assumes headers and comma delimited
my.data <- read.clipboard.tab()  #assumes headers and tab delimited
                                   (e.g., from Excel)
my.data <- read.clipboard.lower() #read in a matrix given the lower
my.data <- read.clipboard.upper() # or upper off diagonal
my.data <- read.clipboard.fwf()  #read in data using a fixed format width
                                   (see read.fwf for instructions)
```

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





4 steps: read, explore, test, graph

read a “foreign” file e.g., an SPSS sav file

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

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

file Character string: the name of the file or URL to read.

use.value.labels Convert variables with value labels into R factors with those levels?

to.data.frame return a data frame? Defaults to `FALSE`, probably should be `TRUE` in most cases.

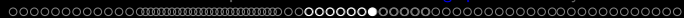
max.value.labels Only variables with value labels and at most this many unique values will be converted to factors if `use.value.labels = TRUE`.

trim.factor.names Logical: trim trailing spaces from factor levels?

trim_values logical: should values and value labels have trailing spaces ignored when matching for `use.value.labels = TRUE`?

use.missings logical: should information on user-defined missing values be used to set the corresponding values to `NA`?





4 steps: read, explore, test, graph

Simulate data

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

- 1 Simulate various item structures

`sim.congeneric` A one factor congeneric measure model

`sim.items` A two factor structure with either simple structure or a circumplex structure.

`sim.rasch` Generate items for a one parameter IRT model.

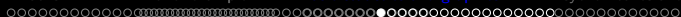
`sim.irt` Generate items for a one-four parameter IRT Model

- 2 Simulate various factor structures

`sim.simplex` Default is a four factor structure with a three time point simplex structure.

`sim.hierarchical` Default is 9 variables with three correlated factors.





Get the data and look at it

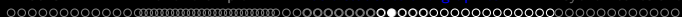
Read in some data, look at the first and last few cases, and then get basic descriptive statistics. For this example, we will use a built in data set.

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

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

epi.bfi has 231 cases from two personality measures



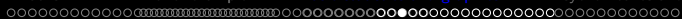


Now find the descriptive statistics for this data set

```
> describe(my.data)
```

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

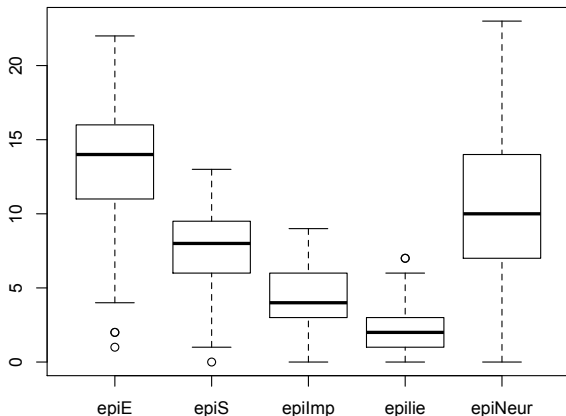




Boxplots are a convenient descriptive device

Show the Tukey “boxplot” for the Eysenck Personality Inventory

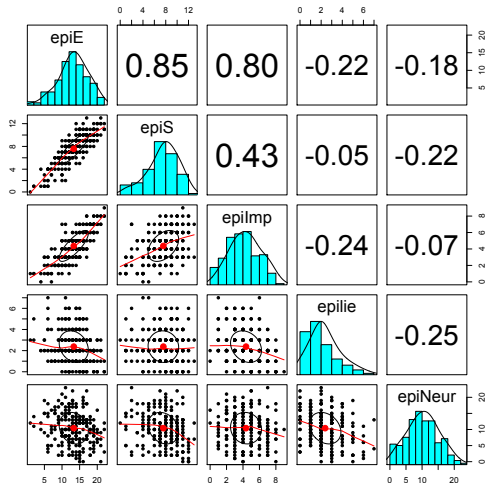
Boxplots of EPI scales





Basic descriptive and inferential statistics

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



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

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





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

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

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





t.test demonstration with Student's data

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

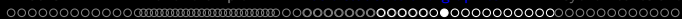
```
Welch Two Sample t-test
data:  extra by group
t = -1.8608, df = 17.776, p-value = 0.07939
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
-3.3654832  0.2054832
sample estimates:
mean in group 1 mean in group 2
       0.75           2.33

But the data were actually paired. Do it for a paired t-test
> with(sleep, t.test(extra~group, paired=TRUE))

Paired t-test
data:  extra by group
t = -4.0621, df = 9, p-value = 0.002833
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
-2.4598858 -0.7001142
sample estimates:
mean of the differences
      -1.58
```

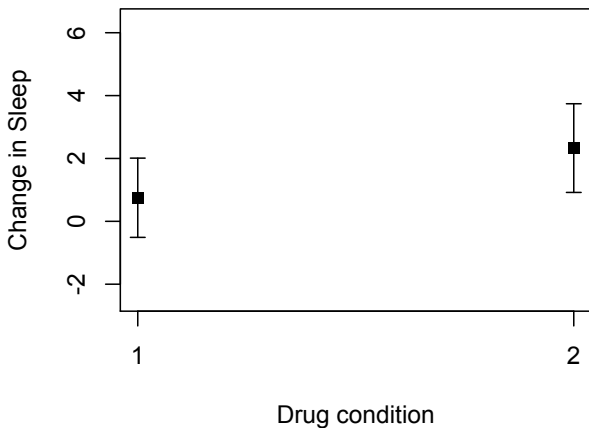
extra	group	ID
0.7	1	1
-1.6	1	2
-0.2	1	3
-1.2	1	4
-0.1	1	5
3.4	1	6
3.7	1	7
...
1.1	2	3
0.1	2	4
-0.1	2	5
4.4	2	6
5.5	2	7
1.6	2	8
4.6	2	9
3.4	2	10





Two ways of showing Student's t test data

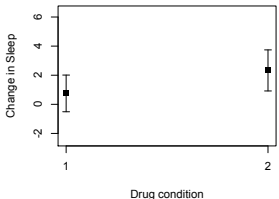
Student's sleep data



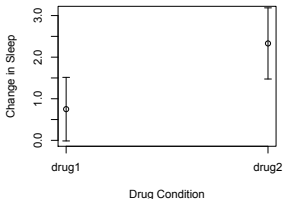


Two ways of showing Student's t test data

Student's sleep data



Student's paired sleep data



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

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





Analysis of Variance

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

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

```
data.ex2
```

```
#show the data
```

Observation	Gender	Dosage	Alertness	
1	1	m	a	8
2	2	m	a	12
3	3	m	a	13
4	4	m	a	12
...				
14	14	f	b	12
15	15	f	b	18
16	16	f	b	22



Analysis of Variance

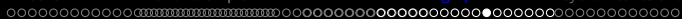
- do the analysis of variances and the show the table of results

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

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

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Gender	1	76.562	76.562	2.9518	0.1115
Dosage	1	5.062	5.062	0.1952	0.6665
Gender:Dosage	1	0.063	0.063	0.0024	0.9617





Show the results table

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

```
Residuals      12 311.250  25.938
```

```
Tables of means
```

```
Grand mean
```

```
14.0625
```

```
Gender
```

```
Gender
```

```
  f      m
```

```
16.25 11.88
```

```
Dosage
```

```
Dosage
```

```
  a      b
```

```
13.50 14.62
```

```
Gender: Dosage
```

```
  Dosage
```

```
Gender a      b
```

```
  f 15.75 16.75
```

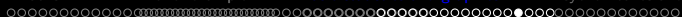
```
  m 11.25 12.50
```



Analysis of Variance: Within subjects

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





Multiple regression

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

```
mod1 <- lm(SATV ~ education + gender + SATQ, data=sat.act)
> summary(mod1, digits=2)
```

Call:

```
lm(formula = SATV ~ education + gender + SATQ, data = sat.act)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-372.91	-49.08	2.30	53.68	251.93

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	180.87348	23.41019	7.726	3.96e-14 ***
education	1.24043	2.32361	0.534	0.59363
gender	20.69271	6.99651	2.958	0.00321 **
SATQ	0.64489	0.02891	22.309	< 2e-16 ***

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

Residual standard error: 86.24 on 683 degrees of freedom

(13 observations deleted due to missingness)

Multiple R-squared: 0.4231, Adjusted R-squared: 0.4205

F-statistic: 167 on 3 and 683 DF, p-value: < 2.2e-16





Zero center the data before examining interactions

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

Call:

```
lm(formula = SATV ~ education * gender * SATQ, data = zsat)
```

Residuals:

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

Coefficients:

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

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



Compare model 1 and model 2

Test the difference between the two linear models

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

Analysis of Variance Table

```
Model 1: SATV ~ education + gender + SATQ
```

```
Model 2: SATV ~ education * gender * SATQ
```

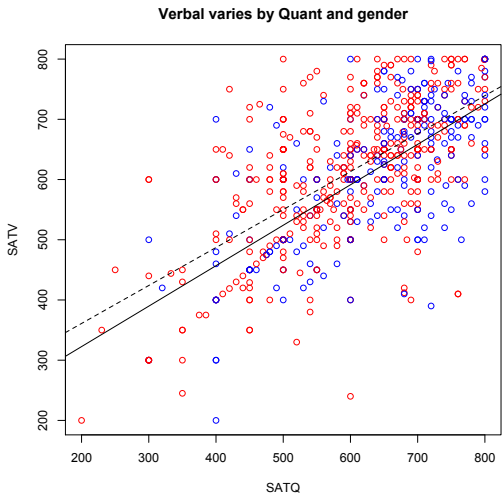
	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
1	683	5079984				
2	679	4870243	4	209742	7.3104	9.115e-06 ***

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





Show the regression lines by gender

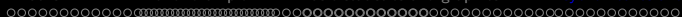


```

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

```





Psychometrics

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



Classic theory estimates of reliability

① Scoring tests

`score.items` Score 1-n scales using a set of keys and finding the simple sum or average of items. Reversed items are indicated by -1

`score.multiple.choice` : Score multiple choice items by first converting to 0 or 1 and then proceeding to score the items.

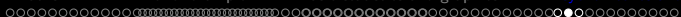
② Alternative estimates of reliability

`alpha` α reliability of a single scale finds the average split half reliability. (some items may be reversed keyed).

`omega` ω_h reliability of a single scale estimates the general factor saturation of the test.

`guttman` Find the 6 Guttman reliability estimates





Classical Test measures of reliability

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

```
> keys.list <- list(Agree=c(-1,2:5),Conscientious=c(6:8,-9,-10),Extraversion=c(-11,-12,13:15),Neuroticism=
> keys <- make.keys(28,keys.list,item.labels=colnames(bfi))
> score.items(keys,bfi)
```

```
Call: score.items(keys = keys, items = bfi)
```

```
(Unstandardized) Alpha:
```

	Agree	Conscientious	Extraversion	Neuroticism	Openness
alpha	0.7	0.72	0.76	0.81	0.6

```
Average item correlation:
```

	Agree	Conscientious	Extraversion	Neuroticism	Openness
average.r	0.32	0.34	0.39	0.46	0.23

```
Guttman 6* reliability:
```

	Agree	Conscientious	Extraversion	Neuroticism	Openness
Lambda.6	0.7	0.72	0.76	0.81	0.6

```
Scale intercorrelations corrected for attenuation
```

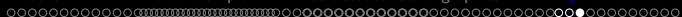
```
raw correlations below the diagonal, alpha on the diagonal
```

```
corrected correlations above the diagonal:
```

	Agree	Conscientious	Extraversion	Neuroticism	Openness
Agree	0.70	0.36	0.63	-0.245	0.23
Conscientious	0.26	0.72	0.35	-0.305	0.30
Extraversion	0.46	0.26	0.76	-0.284	0.32
Neuroticism	-0.18	-0.23	-0.22	0.812	-0.12
Openness	0.15	0.19	0.22	-0.086	0.60

```
...
```





Classical Test measures of reliability

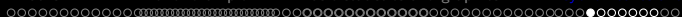
score.items output, continued

Item by scale correlations:

corrected for item overlap and scale reliability

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





Factor analysis of Thurstone 9 variable problem

```
> f3 <- fa(Thurstone,3)
> f3
```

Factor Analysis using method = minres

```
Call: fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate,
  scores = scores, residuals = residuals, SMC = SMC, missing = FALSE,
  impute = impute, min.err = min.err, max.iter = max.iter,
  symmetric = symmetric, warnings = warnings, fm = fm, alpha = alpha)
```

Standardized loadings based upon correlation matrix

	MR1	MR2	MR3	h2	u2
Sentences	0.91	-0.04	0.04	0.82	0.18
Vocabulary	0.89	0.06	-0.03	0.84	0.16
Sent.Completion	0.83	0.04	0.00	0.73	0.27
First.Letters	0.00	0.86	0.00	0.73	0.27
4.Letter.Words	-0.01	0.74	0.10	0.63	0.37
Suffixes	0.18	0.63	-0.08	0.50	0.50
Letter.Series	0.03	-0.01	0.84	0.72	0.28
Pedigrees	0.37	-0.05	0.47	0.50	0.50
Letter.Group	-0.06	0.21	0.64	0.53	0.47

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

With factor correlations of

	MR1	MR2	MR3
MR1	1.00	0.59	0.54
MR2	0.59	1.00	0.52
MR3	0.54	0.52	1.00

...



Factor analysis output, continued

Test of the hypothesis that 3 factors are sufficient.

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

The degrees of freedom for the model are 12 and the objective function was 0.01

The root mean square of the residuals is 0

The df corrected root mean square of the residuals is 0.01

The number of observations was 213 with Chi Square = 2.82 with prob < 1

Tucker Lewis Index of factoring reliability = 1.027

RMSEA index = 0 and the 90 % confidence intervals are 0 0.023

BIC = -61.51

Fit based upon off diagonal values = 1

Measures of factor score adequacy

	MR1	MR2	MR3
Correlation of scores with factors	0.96	0.92	0.90
Multiple R square of scores with factors	0.93	0.85	0.81
Minimum correlation of possible factor scores	0.86	0.71	0.63



Bootstrapped confidence intervals

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

```
Coefficients and bootstrapped confidence intervals
```

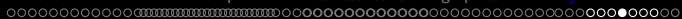
	low	MR1	upper	low	MR2	upper	low	MR3
Sentences	0.77	0.91	0.97	-0.08	-0.04	0.03	-0.07	0.04
Vocabulary	0.81	0.89	0.92	-0.01	0.06	0.14	-0.08	-0.03
Sent.Completion	0.75	0.83	0.89	-0.05	0.04	0.12	-0.11	0.00
First.Letters	-0.10	0.00	0.08	0.71	0.86	0.93	-0.05	0.00
4.Letter.Words	-0.12	-0.01	0.13	0.59	0.74	0.86	-0.01	0.10
Suffixes	0.08	0.18	0.27	0.49	0.63	0.75	-0.20	-0.08
Letter.Series	-0.09	0.03	0.15	-0.08	-0.01	0.08	0.38	0.84
Pedigrees	0.26	0.37	0.49	-0.19	-0.05	0.07	0.37	0.47
Letter.Group	-0.20	-0.06	0.11	0.13	0.21	0.29	0.44	0.64

	upper
Sentences	0.14
Vocabulary	0.05
Sent.Completion	0.11
First.Letters	0.09
4.Letter.Words	0.22
Suffixes	0.04
Letter.Series	0.98
Pedigrees	0.58
Letter.Group	0.75

```
Interfactor correlations and bootstrapped confidence intervals
```

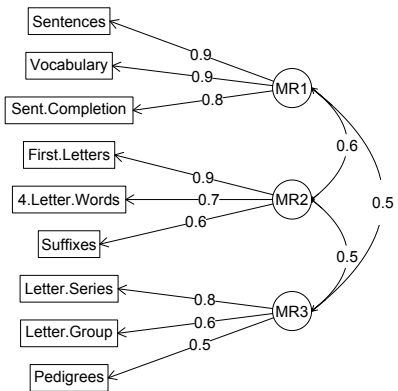
	lower	estimate	upper
1	0.39	0.59	0.61
2	0.33	0.54	0.59
3	0.28	0.52	0.60





The simple factor structure

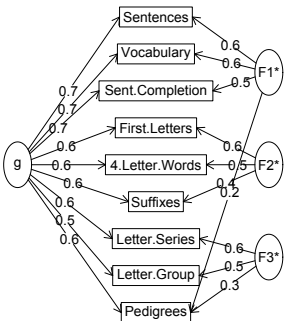
Factor Analysis



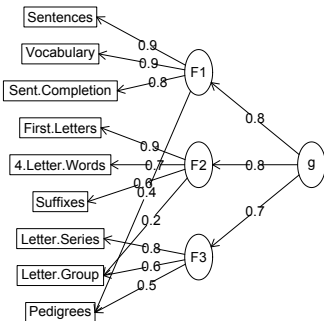


Two ways of viewing the higher order structure

Omega



Hierarchical (multilevel) Structure

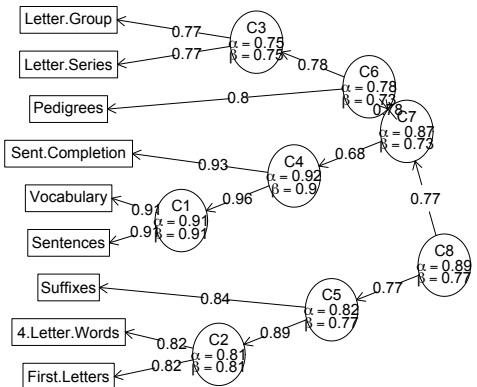




A hierarchical cluster structure found by iclust

iclust(Thurstone)

iclust

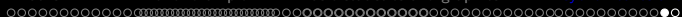




Structural Equation modeling packages

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





Multiple packages to do Item Response Theory analysis

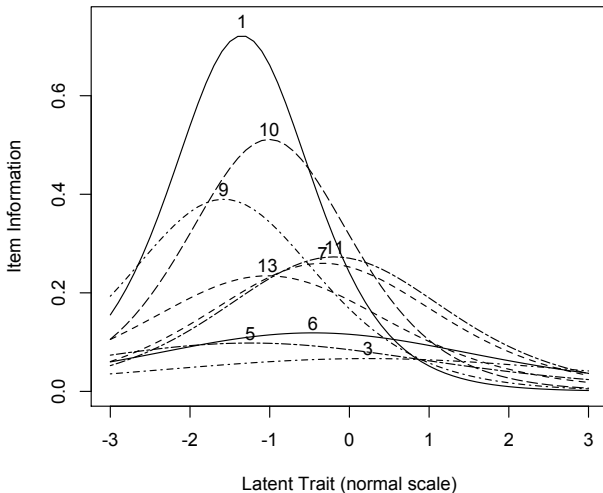
- 1 *psych* uses a factor analytic procedure to estimate item discriminations and locations
 - look at examples for `irt.fa`
 - two example data sets: `iqitems` and `bfi`
- 2 `irt.fa` finds either tetrachoric or polychoric correlation matrices
 - converts factor loadings to discriminations
- 3 `plot.irt` plots item information and item characteristic functions

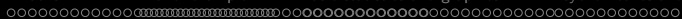




Item Response Information curves for 14 iq items

Item information from factor analysis

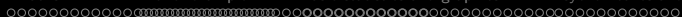




A brief technical interlude

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





The basic types of data structures

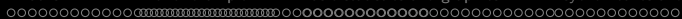
- 1 Scalars (characters, integers, reals, complex)

```
> A <- 1  
> B <- 2
```
- 2 Vectors (of scalars, all of one type) have length

```
> C <- month.name[1:5]  
> D <- 12:24  
> length(D)  
  
[1] 13
```
- 3 Matrices (all of one type) have dimensions

```
> E <- matrix(1:20, ncol = 4)  
> dim(E)  
  
[1] 5 4
```





Show values by entering the variable name

```
> A
```

```
[1] 1
```

```
> B
```

```
[1] 2
```

```
> C
```

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

```
> D
```

```
[1] 12 13 14 15 16 17 18 19 20 21 22 23 24
```

```
> E
```

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



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

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

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

```
> dim(E.df)
```

```
[1] 5 2
```

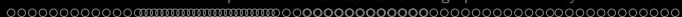
- 2 Lists are collections of what ever you want. They have length, but do not have dimensions.

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

```
> length(F)
```

```
[1] 3
```





Show values by entering the variable name

```
> E.df
```

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

```
> F
```

```
$first
```

```
[1] 1
```

```
$a.vector
```

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

```
$a.matrix
```

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



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

```
> str(F)
```

```
List of 3
```

```
$ first : num 1
```

```
$ a.vector: chr [1:5] "January" "February" "March" "April" ...
```

```
$ a.matrix: int [1:5, 1:4] 1 2 3 4 5 6 7 8 9 10 ...
```

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

```
> F[[2]]
```

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

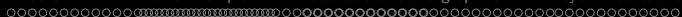
```
> F[["a.matrix"]][, 2]
```

```
[1] 6 7 8 9 10
```

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

```
[1] 2 7 12 17
```





Addressing the elements of a data.frame or matrix

Setting row and column names using paste

```
> E <- matrix(1:20, ncol = 4)
> colnames(E) <- paste("C", 1:ncol(E), sep = "")
> rownames(E) <- paste("R", 1:nrow(E), sep = "")
> E
```

```
      C1 C2 C3 C4
R1    1  6 11 16
R2    2  7 12 17
R3    3  8 13 18
R4    4  9 14 19
R5    5 10 15 20
```

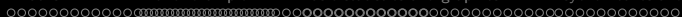
```
> E["R2", ]
```

```
 C1 C2 C3 C4
  2  7 12 17
```

```
> E[, 3:4]
```

```
      C3 C4
R1   11 16
R2   12 17
R3   13 18
R4   14 19
R5   15 20
```





Objects and Functions

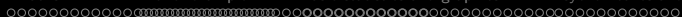
- 1 R is a collection of Functions that act upon and return Objects
- 2 Although most functions can act on an object and return an object ($a = f(b)$), some are binary operators
 - primitive arithmetic functions $+$, $-$, $*$, $/$, $\%*\%$,
 - logical functions $<$, $>$, $==$, $!=$
- 3 Some functions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



Getting help

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





Useful functions

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

`file.choose` () find a file

`file.choose` (new=TRUE) create a new file

`read.table` (filename)

`read.csv` (filename) reads a comma separated file

`read.delim` (filename) reads a tab delimited file

`c` (...) combine arguments

`from:to` e.g., 4:8

`seq` (from,to, by)

`rep` (x,times) repeat x

`gl` (n,k,...) generate factor levels

`matrix` (x,nrow=,ncol=) create a matrix

`dim` (x) dimensions of x

`data.frame` (...) create a data frame

`list` (...) create a list

`colnames` (x)

`rownames` (x)

`rbind` (...) combine by rows

`cbind` (...) combine by columns

`is.na` (x) also `is.null(x)`, `is...`

`na.omit` (x) ignore missing data

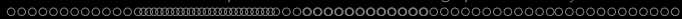
`table` (x)

`merge` (x,y)

`ls` () show workspace

`rm` () remove variables from workspace





Questions?

