Overview

I. Day 1: What is R: An introduction
II. Day 2: Graphical displays and EDA
III. Day 3: The general linear model and its special cases (ANOVA), multilevel models
IV. Day 4: Multivariate analysis
V. Day 5: R as a programming language
What is R: an introduction

I. What is it

II. How to get it

III. Entering or getting data

IV. Basic descriptive stats
I. What is it?

II. Why use it?

III. Common (mis)perceptions

IV. Examples for personality and individual differences research
R: What is it?

I.R: An international collaboration

II.R: the open source - public domain version of S/S+

III.R: written by statisticians (and all of us) for statisticians (and the rest of us)

IV.R: an extensible language
## Common statistical programs

<table>
<thead>
<tr>
<th>General</th>
<th>Specialized</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>AMOS</td>
</tr>
<tr>
<td>S+</td>
<td>EQS</td>
</tr>
<tr>
<td>SAS</td>
<td>LISREL</td>
</tr>
<tr>
<td>SPSS</td>
<td>MPlus</td>
</tr>
<tr>
<td>STATA</td>
<td>Mx</td>
</tr>
<tr>
<td>SYSTAT</td>
<td>your favorite program.</td>
</tr>
</tbody>
</table>
Common statistical programs most are costly

<table>
<thead>
<tr>
<th>General</th>
<th>Specialized</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>AMO$</td>
</tr>
<tr>
<td>$+$</td>
<td>EQ$</td>
</tr>
<tr>
<td>$A$</td>
<td>LI$REL</td>
</tr>
<tr>
<td>$P$$</td>
<td>MPlu$</td>
</tr>
<tr>
<td>$TATA$</td>
<td>Mx</td>
</tr>
<tr>
<td>$Y$TAT</td>
<td>your favorite program.</td>
</tr>
</tbody>
</table>
R: a way of thinking
(from the R point of view)

• “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)
But it is open source - how can you trust it?

I. Q: When you use it [R], since it is written by so many authors, how do you know that the results are trustable?

II. 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.

Taken from the R.-fortunes (selections from the R.-help list serve)
What is R? : Technically

I. R is an open source implementation of S (S-Plus is a commercial implementation)

II. R is available under GNU Copy-left

III. The current version of R is 2.8.1 (2.9.0 is at the alpha release, 2.10 is in development)

IV. R is group project run by a core group of developers (with new releases semiannually)

(Adapted from Robert Gentleman)
R: History

I. 1991-93: Ross Dhaka and Robert Gentleman begin work on R project at U. Auckland

II. 1995: R available by ftp under the GPL

III. 96-97: mailing list and R core group is formed

IV. 2000: John Chambers, designer of S joins the R core (wins a prize for best software from ACM for S)

V. 2001-2005: Core team continues to improve base package

VI. 2009: Becoming the standard for statistics, although other languages are relevant (Python)

VII. Many (>1455) other contributed “packages”
R in Psychology
Personality/Psychometrics

I. Some software packages tailored to psychology have been developed (e.g. those in the task view of psychometrics)

II. Some universities are now introducing R to graduate students

III. Some instructors are using it for undergraduates.
Why R?

I. Graphics for data exploration and interpretation
II. Data manipulation including statistics as data
III. Statistical analysis
   A. Standard univariate and multivariate generalizations of the linear model
   B. Multivariate-structural extensions
IV. Ease of programming for new applications
Ok, how do I get it

I. CRAN (Comprehensive R Archive Network)
   A. http://cran.r-project.org/
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!


- 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](http://cran.r-project.org/src/base/R-2.8.1/dev/). Please read about [new features and bug fixes](http://cran.r-project.org/src/base/R-2.8.1/CHANGES) before filing corresponding feature requests or bug reports.

- Source code of older versions of R is [available here](http://cran.r-project.org/src/base/R-2.8.1/old/).

- Contributed extension [packages](http://cran.r-project.org/src/base/R-2.8.1/CRAN/)

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](http://cran.r-project.org/src/base/R-2.8.1/doc/FAQ.pdf) before you send an email.
R for Mac OS X

This directory contains binaries for a base distribution and packages to run on Mac OS X (release 10.2 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.

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.

Universal R 2.8.1 for Mac OS X released on 2008/12/22

This binary distribution of R and the GUI supports both PowerPC and Intel based Macs. The corresponding binaries of R packages are available for both architectures as well. Starting with R 2.3.1, CRAN binaries support Mac OS X 10.4 (Tiger) and higher only. It is, however, possible to compile binaries for earlier OS X versions from sources.

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

```
md5 R-2.8.1.dmg
```

in the Terminal application to print the MD5 checksum for the R-2.8.1.dmg image.

Files:

- **R-2.8.1.dmg** (latest version)
  - MD5: e41f821d02b0a56f23d8455a7e1b582
  - (cn. 63MB)
  - Universal binary of **R 2.8.1** for Mac OS X 10.4.4 and higher. This is a disk image containing the installer of R for Mac OS X 10.4.4 or higher. This image also contains Tcl/Tk libraries (for X11) and GNU Fortran 4.2.3 for both PowerPC and Intel Macs. This binary was tested on both Mac OS X 10.4 (Tiger) and Mac OS X 10.5 (Leopard).
  - Depending on your browser, you may need to press the control key and click on this link to download the file. To install R simply double-click on icon of the multi-package "R.mpkg" contained in the R-2.8.0.dmg disk image.

- **R-2.8.1-mini.dmg**
  - MD5: 4cdff2d9d19a65f23d8455a7e1b582
  - (cn. 27MB)
  - Universal binary of **R 2.8.1** for Mac OS X, upgrade package without supplemental tools.
  - This is a subset of the above image. Unless a full R 2.8.0 installer was used before, it is not possible to compile packages from Fortran sources when this smaller subset is used. Only binary installs will work correctly. Also Tcl/Tk will not work unless installed separately. The supplemental tools are also available in the Tools directory.
Contributed Packages

Installation of Packages

Please type help("INSTALL") or help("install.packages") in R for information on how to install packages from this directory. The manual R Installation and Administration (also contained in the R base sources) explains the process in detail.

CRAN Task Views allow you to browse packages by topic and provide tools to automatically install all packages for special areas of interest. Currently, 23 views are available.

Daily Package Check Results

All packages are tested regularly on machines running Debian GNU/Linux. Packages are also checked under MacOS X and Windows, but only at the day the package appears on CRAN.

The results are summarized in the check summary (some timings are also available). Additional details for Windows checking and building can be found in the Windows check summary.

Writing Your Own Packages

The manual Writing R Extensions (also contained in the R base sources) explains how to write new packages and how to contribute them to CRAN.

Available Bundles and Packages

Currently, the CRAN package repository features 1733 objects including 1726 packages and 7 bundles containing 26 packages, for a total of 1752 available packages.

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

ADaCGH Analysis of data from aCGH experiments
AER Applied Econometrics with R
AIGIS Areal Interpolation for GIS data
AIS Tools to look at the data ("Ad Inidicia Spectata")
ALS multivariate curve resolution alternating least squares (MCR-ALS)
AMORE A MORE flexible neural network package
ARES All-Risk evaluation with a sequential model for analysis
CRAN Task Views

Bayesian
ChemPhys
Cluster
Distributions
Econometrics
Environmetrics
ExperimentalDesign
Finance
Genetics
Graphics
gR
HighPerformanceComputing
MachineLearning
Multivariate
NaturalLanguageProcessing
Optimization
Pharmacokinetics
Psychometrics
Robust
SocialSciences
Spatial
Survival
TimeSeries
Bayesian Inference
Chemometrics and Computational Physics
Cluster Analysis & Finite Mixture Models
Probability Distributions
Computational Econometrics
Analysis of Ecological and Environmental Data
Design of Experiments (DoE) & Analysis of Experimental Data
Empirical Finance
Statistical Genetics
Graphic Displays & Dynamic Graphics & Graphic Devices & Visualization
gRaphical Models in R
High Performance and Parallel Computing
Machine Learning & Statistical Learning
Multivariate Statistics
Natural Language Processing
Optimization and Mathematical Programming
Analysis of Pharmacokinetic Data
Psychometric Models and Methods
Robust Statistical Methods
Statistics for the Social Sciences
Analysis of Spatial Data
Survival Analysis
Time Series Analysis

To automatically install these views, the ctv package needs to be installed, e.g., via
install.packages("ctv")
library("ctv")
and then the views can be installed via install.views or update.views (which first assesses which of the
packages are already installed and up-to-date), e.g.,
install.views("Econometrics")
or
update.views("Econometrics")
R version 2.8.1 Patched (2008-12-31 r47414)
Copyright (C) 2008 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

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.

The R Core Team
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.

[Workspace restored from /Volumes/WR/bill/.Rdata]

>
Running in X11

bash-3.2# R

R version 2.9.0 alpha (2009-03-23 r49200)
Copyright (C) 2009 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

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.

[Previously saved workspace restored]

> library(psych)
> 

bash-3.2$ R

R version 2.3.0 alpha (2009-03-23 m48290)
Copyright (C) 2009 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

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.

[Previously saved workspace restored]

> library(psych)
> demo(psych)
Error in demo(psych) : No demo found for topic 'psych'
> example(psych)

psych> #See the separate man pages
psych> test.psych()

too many factors requested for this number of variables to use SMC, 1s used instead
Hit <Return> to see next plot;
Hit <Return> to see next plot;
Hit <Return> to see next plot;
Loading required package: GPArotation
too many factors requested for this number of variables to use SMC, 1s used instead
Loading required package:igraphiz
Loading required package: graph

Loading required package: grid
Hit <Return> to see next plot;
Hit <Return> to see next plot;
Hit <Return> to see next plot;
Hit <Return> to see next plot;
Hit <Return> to see next plot;
Hit <Return> to see next plot;
Hit <Return> to see next plot!
Getting help

psych {psych}

A package for personality, psychometric, and psychological research.

Description

Overview of the psych package.

The psych package has been developed at Northwestern University to include functions most useful for personality and psychological research. Some of the functions (e.g., `read.clipboard`, `describe`, `pairs.panels`, `error.bars`) are useful for basic data entry and analysis. Use `help(package="psych")` to see a list of all functions.

Psychometric applications include routines for Very Simple Structure (VSS), Minimum Average Partial correlation (MAP), (ICLUST) and principal axes factor analysis (factor.pa), as well as functions to do Schmid Leiman transformations (slc) and hierarchical factor structure into a bifactor solution, and to calculate reliability coefficients alpha (`score.items`, `score.r`) and McDonald's omega (`omega` and `omega.graph`). Guttman's six estimates of internal consistency reliability (measures of intraclass correlation coefficients (`icc`)) are also available.

The `score.items` and `score.multiple.choice` functions may be used to form single or multiple scales from sets of multiple choice items by specifying scoring keys.

Additional functions make for more convenient descriptions of item characteristics. Functions under development include Response measures.

A number of procedures have been developed as part of the Synthetic Aperture Personality Assessment (SAPA) project for forming and analyzing composite scales equivalent to using the raw data but doing so by adding within and between class clusters. These functions include extracting clusters from factor loading matrices (`factor2cluster`), synthetically forming clusters (`cluster.cor`), and finding multiple (`mat.regress`) and partial (`partial.r`) correlations from correlation matrices.

Functions to generate simulated data with particular structures include `sim.circ` (for circumplex structures), `sim.item` (`sim.congeneric` for a specific demonstration of congeneric measurement). The functions `sim.congeneric` and `sim.t` to create data sets with particular structural properties. A more general form for all of these is `sim.structural` for general models.

Functions to apply various standard statistical tests include `p.rep` and its variants for testing the probability of replication intervals of a correlation, `t.test` to test single, paired, or sets of correlations.

In order to study diurnal or circadian variations in mood, it is helpful to use circular statistics. Functions to find the circadian (phasic) correlations (`circadian.cor`) and the correlation between linear variables and circular variables (`circ.cor`) complement a function to find the best-fitting phase angles (`phase`) for measures taken with a fixed period (e.g., 24 hour cycles).
Getting help

R version 2.8.1 Patched (2008-12-31 r47414)
Copyright (C) 2008 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

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.

[Workspace restored from /Volumes/WR/bill/.RData]

> library(psych)
> ?psych
> ?c
>

c {base}

Combine Values into a Vector or List

Description

This is a generic function which combines its arguments.

The default method combines its arguments to form a vector. All arguments are coerced to a common type w

type of the returned value, and all attributes except names are removed.

Usage

c(..., recursive=FALSE)

Arguments

... objects to be concatenated.
recursive logical. If recursive = TRUE, the function recursively descends through lists (and pairlists) con
their elements into a vector.

Details

The output type is determined from the highest type of the components in the hierarchy NULL < raw < logic
real < complex < character < list < expression. Pairlists are treated as lists, but non-vector components (such r
calls) are treated as one-element lists which cannot be unlisted even if recursive = TRUE.

c is sometimes used for its side effect of removing attributes except names, for example to turn an array into a an
vector is a more intuitive way to do this, but also drops names. Note too that methods other than the def
required to do this (and they will almost certainly preserve a class attribute).

Value

NULL or an expression or a vector of an appropriate mode. (With no arguments the value is NULL.)
Package vignettes
R Site Search

Query: Rpad

Display: 20 | Description: normal | Sort: by score

Target:
- Functions
- R-help 2008
- R-help 2002-2007
- Rhelp 1997-2001
- R-devel

For problems WITH THIS PAGE (not with R) contact baron@psych.upenn.edu.

Results:

References:
- docs: [Rpad: 1]
- functions: [Rpad: 10]
- Rhelp02a: [Rpad: 32]

Total 43 documents matching your query.

1. [R] Rpad graphics from Paul Hiemstra on 2008-12-14 (stdin) (score: 51)
   Author: Paul Hiemstra (p.hiemstra)
   Date: Fri, 30 Jan 2009 16:22:44 -0500
   [R] Rpad graphics This message: [ Message body ] [ More options ] Related messages: [ (none) ]
   http://finzi.psych.upenn.edu/R/Rhelp08/archive/151942.html (6,378 bytes)
R/Rpad Reference Card

Help and basics
Most R functions have online documentation.
help(topic) documentation on topic
?topic id
help.search("topic") search the help system
apropos("topic") the names of all objects in the search list matching the regular expression "topic"
help.start() start the HTML version of help
str(a) display the internal structure of an R object
summary(a) gives a "summary" of a univariate statistical summary but it is
generic meaning it has different operations for different classes of a
ls() shows objects in the search path, specify pat="pat" to search on a pattern
ls.str() str() for each variable in the search path
dir() show files in the current directory
methods() show 3 methods of a
methods(class=class(a)) lists all the methods to handle objects of class a
options(... set or examine many global options; common ones: width, digits, error
library(x) load add-on packages; library(h=lp=x) lists datasets and functions in package x
attach(x) database x to the R search path; x can be a list, data frame, or
R data file created with save. use search() to see the search path.
detach(x) x from the R search path; x can be a name or character string of
an object previously attached or a package.

Input and output
load() load the datasets written with save
data(x) loads specified data sets
read.table(file) reads a file in table format and creates a data
frame from it, the default separator sep=" " is any whitespace; use
head=TRUE to read the first line as a header of column names; use
as.is=TRUE to prevent character vectors from being converted to factors;
use comment=FALSE to prevent # symbol from being interpreted as a comment; use
sep=to skip n lines before reading data; see the
help for options on row names, NA treatment, and others
read.csv("filename", header=TRUE) id. but with defaults set for
reading comma-delimited files
read.delim("filename", header=TRUE) id. but with defaults set

cat(... file="", sep=" ") prints the arguments according to
character; sep is the character separator between arguments
print(a, ...) prints its arguments; generic meaning it can have different
methods for different objects
format(x, ...) format an X object for pretty printing
write.table(x, file="", row.names=TRUE, col.names=TRUE,
sep=" ") prints x after converting to a data frame; if quote=TRUE,
character or factor columns are surrounded by quotes ("), sep is the
field separator, col is the end-of-line separator, na is the string for
missing values; use col.name=NA to add a blank column header to
get the column headers aligned correctly for spreadsheet input
sink(file) output to file, until sink()
Most of the I/O functions have a file argument. This can often be a character
string naming a file or a connection. file=" means the standard input or
output. Connections can include files, pipes, zipped files, and R variables.
On windows, the file connection can also be used with description
"clipboard". To read a table copied from Excel, use
x <- read.delim("clipboard")
To write a table to the clipboard for Excel, use
write.table(x,"clipboard",sep="",col.name=NA)
For database interaction, see packages DBI, DBI, RDSOL, and
ROracle. See packages XML, dBase, etCIF for reading other file formats.

Data creation
c(...) generic function to combine arguments with the default formatting a
vector: with recursive=TRUE descends through lists combining all
elements into one vector
from: to generate a sequence; "-" has operator priority, 1+4 i 12, 2, 3, 4, 5
seq(from,to) generates a sequence by specifying increment; length;
specifies desired length
seq_along(x) generator 1, 2, ..., length(x); useful for for loops
rep(x, times) replicate x times; use each=to repeat "each" element
of x each times; rep(c(1,2,3),2) is 1 2 3 1 2 3 2
rep(c(1,2,3),each=2) is 1 1 2 2 3 3

data.frame(...)
create a data frame of the named or unnamed argument:
data.frame(x=1:4,x=c("a","b","c","d"),y=10); shorter vectors are recycled to
the length of the longest
list(...)
create a list of the named or unnamed arguments;
list(list=1+c(1,2),b=3,3)
array(x,dim=) array with data x; specify dimensions like
dim=c(3,4,2), elements of x recycle if x is not long enough
matrix(:m,nrow=n,mcol=n) matrix; elements of x recycle
factor(x,levels=) encodes a vector x as factor
gl(n,k,length=k,labels=1:n) generate levels (factors) by specifying
the pattern of their levels, n is the number of levels, and
is the number of replications

expand.grid() a data frame from all combinations of the supplied vectors
rbind(...)
combine arguments by rows for matrices, data frames, and

Slicing and extracting data

Indexing list
x[1] list with elements
x[[1]] 1st element of the list
x["name"] element of the list named "name"
x["name""] id

Indexing vectors
x[1] all but the 1st element
x[-1] first elements
x[-c(1,4,2)] elements from 1 to the end
x["name"] all elements greater than
x[x>3] all elements between 3 and 5
x[x>3 & x<5] elements in the given set

Indexing matrices
x[i,j] element at row i, column j
x[i,"name"] row named "name"


Variable conversion
as.array(x), as.data.frame(x), as.numeric(x),
as.logical(x), as.complex(x), as.character
convert type, for a complete list, use methods(as)

Variable information
is.na(x), is.nul(x), is.sr(x), is.data.
is.numeric(x), is.complex(x), is.character
... test for type, for a complete list, use methods(is)

length(x) number of elements in
x

dim(x) Retrieve or set the dimension of an object; dim(x) < c
dimnames(x) Retrieve or set the dimension names of an object
trow(x) number of rows; dim(x) is the same but treats a vector
as row matrix
ncol(x) and ncol(x) id. for columns

class(x) or get the set of class of x; class(x) < - "myclass"
unclass(x) remove the class attribute of x
attr(x, which) get or set the name which of x
attributes(obj) get or set the list of attributes of obj

Data selection and manipulation
which.max(x) returns the index of the greatest element of x.
Data Manipulation

Data Entry

I. from console

II. from clipboard (copied from other programs)

III. from file (text files, csv, SPSS, Excel, MySQL)

IV. from the web
R is a desk calculator

> 2+2
[1] 4
> 3^4
[1] 81
> pi
[1] 3.141593
> x <- c(1,3,5,7)
> x
[1] 1 3 5 7
> m <- mean(x)
> m
[1] 4
> mean(x)
[1] 4
> sd(x)
[1] 2.581989
>
A very fancy desk calculator

```r
> set.seed(42)
> V <- seq(1:5)
> M <- matrix(sample(5,15),replace=TRUE,ncol=3,nrow=5)
Error in matrix(sample(5, 15), replace = TRUE, ncol = 3, nrow = 5) :
  unused argument(s) (replace = TRUE)
> M <- matrix(sample(5,15,replace=TRUE),ncol=3,nrow=5)
> V
[1] 1 2 3 4 5
> M
[,1] [,2] [,3]
[1,] 5 3 3
[2,] 5 4 4
[3,] 2 1 5
[4,] 5 4 2
[5,] 4 4 3
```
But a calculator none the less

```r
> V
[1] 1 2 3 4 5
> M
[,1] [,2] [,3]
[1,]  5  3  3
[2,]  5  4  4
[3,]  2  1  5
[4,]  5  4  2
[5,]  4  4  3
> V * M
[,1] [,2] [,3]
[1,]  95  73  67
[2,]  73  58  50
[3,]  67  50  63
> t(M) %*% M
[,1] [,2] [,3]
[1,]  95  73  67
[2,]  73  58  50
[3,]  67  50  63
```
A graphing calculator

> curve(dnorm(x,1,0.5),-3,3,ylab="probability of x",main="Comparing distributions")
Add the second line

```r
> curve(dnorm(x,1,0.5),-3,3,ylab="probability of x",main="Comparing distributions")
> curve(dnorm(x,0,1),add=TRUE)
```

Comparing distributions
R is a stats table

> pt(2.0,6) #probability (one tailed of a t > 2.0)
[1] 0.9537868
> pnorm(2.0) #probability of a normal distribution with value of 2.0)
[1] 0.9772499
> dnorm(-1) #normal with z value of -1.0
[1] 0.2419707
> pf(3.5,1,20) #probability of an F statistic
[1] 0.923926
> qf(.95,1,60) # the critical value of an F at the 95 percent value
[1] 4.001191
> qchisq(.95,1) # the critical value for a 1 df \chi^2
[1] 3.841459
A powerful stats table

```r
> z <- seq(0,3,.2)
> z
[1] 0.0 0.2 0.4 0.6 0.8 1.0 1.2 1.4 1.6 1.8 2.0 2.2 2.4 2.6 2.8 3.0
> pnorm(z)
[1] 0.5000000 0.5792597 0.6554217 0.7257469 0.7881446 0.8413447 0.8849303 0.9192433 0.9452007 0.9640697 0.9772499 0.9860966
[13] 0.9918025 0.9953388 0.9974449 0.9986501
> round(data.frame(z,p=pnorm(z)),3)
   z     p
1 0.0 0.500
2 0.2 0.579
3 0.4 0.655
4 0.6 0.726
5 0.8 0.788
6 1.0 0.841
7 1.2 0.885
8 1.4 0.919
```
## Random data

<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>Bernuilli trials (e.g. coin flips)</td>
</tr>
<tr>
<td>Student’s t</td>
<td>t</td>
<td>df</td>
<td></td>
<td></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>Beta</td>
<td>beta</td>
<td>shape1</td>
<td>shape2</td>
<td>nc</td>
<td>distribution theory</td>
</tr>
<tr>
<td>Cauchy</td>
<td>cauchy</td>
<td>location</td>
<td>scale</td>
<td></td>
<td>Infinite variance distribution</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 theory</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>
Graphics of random data

- Normal distribution
- Rectangular distribution
- Poisson distribution
- Log Normal distribution
Graphing commands

```r
op <- par(mfrow=c(2,2))

n <- 1000
x <- rnorm(n)
hist(x,main="Normal")
x <- runif(n)
hist(x,main="Rectangular")
x <- rpois(n,3)
hist(x,main="Poisson")
x <- rlnorm(n)
hist(x,main="Log Normal")

op <- par(mfrow=c(1,1))
```
Data Entry example

> 5/2  #it is a calculator
[1] 2.5

> 2^8  #still a calculator
[1] 256

> a <- c("A","short","list")  #store this value

> a  #now show it
[1] "A"  "short" "list"

>
Data entry

> A <- 1:25  # make a sequence
> A              # show it
[1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22
 23 24 25
> B <- matrix(A,ncol=5) # convert the vector to a matrix
> B                # show it
[1,]  1   6  11  16  21
[2,]  2   7  12  17  22
[3,]  3   8  13  18  23
[4,]  4   9  14  19  24
[5,]  5  10  15  20  25
> c <- seq(25,5,by=-5)  # seq can do more interesting stuff
> c
[1] 25 20 15 10   5
library(psych)

> A <- read.clipboard()

> A  #show it

V1 V2 V3 V4 V5
S1  1  6 11 16 21
S2  2  7 12 17 22
S3  3  8 13 18 23
S4  4  9 14 19 24
S5  5 10 15 20 25
From the clipboard (no headers)

```r
> C <- read.clipboard(header=FALSE)
> C  #automatically adds column names!
```

```plaintext
1  6 11 16 21  
2  7 12 17 22  
3  8 13 18 23  
4  9 14 19 24  
5 10 15 20 25  
```

<table>
<thead>
<tr>
<th>V1</th>
<th>V2</th>
<th>V3</th>
<th>V4</th>
<th>V5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>6</td>
<td>11</td>
<td>16</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>7</td>
<td>12</td>
<td>17</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>8</td>
<td>13</td>
<td>18</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>9</td>
<td>14</td>
<td>19</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>10</td>
<td>15</td>
<td>20</td>
</tr>
</tbody>
</table>
From built in datasets

> data() #show all data sets

> data(sat.act) # get this one

> head(sat.act) #show the first lines

> dim(sat.act) #show dimensions

[1] 700  6

<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>29504</td>
<td>1</td>
<td>2</td>
<td>33</td>
<td>31</td>
<td>600</td>
</tr>
<tr>
<td>29518</td>
<td>1</td>
<td>5</td>
<td>26</td>
<td>28</td>
<td>640</td>
</tr>
</tbody>
</table>
From files or the web

```r
Fn <- file.choose()  #uses the system commands to find it

My.data <- read.table(Fn)

> My.data <- read.table(Fn, header=TRUE)

> head(My.data)  #show the first 6 lines

<table>
<thead>
<tr>
<th></th>
<th>epiE</th>
<th>epiS</th>
<th>epiImp</th>
<th>epilie</th>
<th>epiNeur</th>
<th>bfagree</th>
<th>bfcon</th>
<th>bfext</th>
<th>bfneur</th>
<th>bfopen</th>
<th>bdi</th>
<th>traitanx</th>
<th>stateanx</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>18</td>
<td>10</td>
<td>7</td>
<td>3</td>
<td>9</td>
<td>138</td>
<td>96</td>
<td>141</td>
<td>51</td>
<td>138</td>
<td>1</td>
<td>24</td>
<td>22</td>
</tr>
<tr>
<td>2</td>
<td>16</td>
<td>8</td>
<td>5</td>
<td>1</td>
<td>12</td>
<td>101</td>
<td>99</td>
<td>107</td>
<td>116</td>
<td>132</td>
<td>7</td>
<td>41</td>
<td>40</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>143</td>
<td>118</td>
<td>38</td>
<td>68</td>
<td>90</td>
<td>4</td>
<td>37</td>
<td>44</td>
</tr>
<tr>
<td>4</td>
<td>12</td>
<td>6</td>
<td>4</td>
<td>3</td>
<td>15</td>
<td>104</td>
<td>106</td>
<td>64</td>
<td>114</td>
<td>101</td>
<td>8</td>
<td>54</td>
<td>40</td>
</tr>
<tr>
<td>5</td>
<td>14</td>
<td>6</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>115</td>
<td>102</td>
<td>103</td>
<td>86</td>
<td>118</td>
<td>8</td>
<td>39</td>
<td>67</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>4</td>
<td>2</td>
<td>5</td>
<td>15</td>
<td>110</td>
<td>113</td>
<td>61</td>
<td>54</td>
<td>149</td>
<td>5</td>
<td>51</td>
<td>38</td>
</tr>
</tbody>
</table>
```
Getting data from the web

> datafilename <- "http://personality-project.org/r/datasets/maps.mixx.epi.bfi.data"
> my.data(datafilename)
Error: could not find function "my.data"
> my.data <- read(datafilename)
Error: could not find function "read"
> my.data <- read.table(datafilename,header=TRUE) #read the data file
>
> dim(my.data)
[1] 231 13
> headtail(my.data)
Error in inherits(x, "data.frame") : object "mydata" not found
> 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
Looking at the data

> headtail(My.data) #a psych function
demonstrating the passing of parameters to functions

<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>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>18</td>
<td>10</td>
<td>7</td>
<td>3</td>
<td>9</td>
<td>138</td>
<td>96</td>
<td>141</td>
<td>51</td>
<td>138</td>
<td>1</td>
<td>24</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>16</td>
<td>8</td>
<td>5</td>
<td>1</td>
<td>12</td>
<td>101</td>
<td>99</td>
<td>107</td>
<td>116</td>
<td>132</td>
<td>7</td>
<td>41</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>143</td>
<td>118</td>
<td>38</td>
<td>68</td>
<td>90</td>
<td>4</td>
<td>37</td>
<td>44</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>12</td>
<td>6</td>
<td>4</td>
<td>3</td>
<td>15</td>
<td>104</td>
<td>106</td>
<td>64</td>
<td>114</td>
<td>101</td>
<td>8</td>
<td>54</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>14</td>
<td>6</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>115</td>
<td>102</td>
<td>103</td>
<td>86</td>
<td>118</td>
<td>8</td>
<td>39</td>
<td>67</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>
</tr>
<tr>
<td>229</td>
<td>19</td>
<td>10</td>
<td>7</td>
<td>2</td>
<td>11</td>
<td>162</td>
<td>152</td>
<td>163</td>
<td>104</td>
<td>164</td>
<td>1</td>
<td>29</td>
<td>47</td>
<td></td>
</tr>
<tr>
<td>230</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>10</td>
<td>95</td>
<td>111</td>
<td>75</td>
<td>123</td>
<td>138</td>
<td>5</td>
<td>39</td>
<td>58</td>
<td></td>
</tr>
<tr>
<td>231</td>
<td>8</td>
<td>6</td>
<td>3</td>
<td>2</td>
<td>15</td>
<td>85</td>
<td>62</td>
<td>90</td>
<td>131</td>
<td>96</td>
<td>24</td>
<td>58</td>
<td>58</td>
<td></td>
</tr>
</tbody>
</table>
Editing the data

Fix(My.data) # edits the object

A <- edit(My.data) # edits and then returns the new version

(Or just edit in your favorite editor)
Why R? Graphics

I. Sample graphics taken from
   A. http://personality-project.org/r/
      1. showing what can be done by an amateur
   B. http://addictedtor.free.fr/graphiques/
      1. showing some most impressive graphs
Standard Plots of factor loadings

Two dimensions of affect

- factor 1
- factor 2

Factors:
- sluggish
- happy
- depressed
- calm
- smiled
- etc.
Data points can be dynamically identified
Multi-panel graphs can be labeled separately and organized vertically or horizontally.

Simulated data can be generated to fit normal, rectangular, binomial, poisson, exponential, etc. distributions.
Scatter Plot Matrices can show smoothed fits
Can scale font size of correlations by absolute size of $r$
Error bars on two dimensions

Movie Study 1

Movie Study 2

Positive Affect

Negative Affect

Sad

Threat

Neutral

Happy
Presentation graphics scale to fit page and produce output for pdf presentations

Window or page size is controllable
Graphic output can be taken from multiple programs (e.g. Very Simple Structure, factor analysis)
Built-in data sets provide useful demonstrations of stats

Anscombe's 4 Regression data sets

1. $y_1 = 3 + 0.5x_1$
2. $y_2 = 3 + 0.5x_2$
3. $y_3 = 3 + 0.5x_3$
4. $y_4 = 3 + 0.5x_4$
Mapping data (GIS) may be combined with descriptive data.
Many GIS map files are available for download from ESRI
GIS maps detail regional boundaries
Combine income data from 2000 census with zipcode map of San Diego County

Median Household Income for 2000

2000 median household income for zip codes
GIS files of Borneo can show country boundaries (e.g., Malaysia, Brunei, Indonesia)
Public access GIS files include rivers and roads
Even more graphics

- Taken from a collection of R demonstrations and graphics
- http://addictedtor.free.fr/graphiques/
Principal components and clustering of sources of variance in USA arrest data

PCA 5 vars

princomp(x = data, cor = cor)

Fertility
Catholic
Examination
Education
Agriculture

Comp.1
Comp.2
Comp.3
Comp.4
Comp.5

Clustering 4 groups

80 60 40 20 0

Groups

Factor 1 [41%]

Factor 3 [19%]

V. De Genov
Mixture models
Notched Boxplots show confidence regions
Multipanel graphs

Three Varieties of Iris

Scatter Plot Matrix
Histograms and fitted distributions
Combine scatter plot with histograms
Why R?

I. Data manipulation including statistics as data
   A. the output of any function may be input for any other

II. Graphics for data exploration and interpretation

III. Statistical analysis
   A. Standard univariate and multivariate generalizations of the linear model
   B. Multivariate-structural extensions
Data Manipulation: Data Structures

I. Data types: integer, real, logical, character, string

II. Vectors of any data type

III. Matrices of any data type

IV. Data Frames (similar to matrix of mixed type)

V. Lists of any mixture of types

VI. All operations are functions and the returned values may be used in any data structure (e.g., as an element of a data frame or of a list)
Data structures

I. Elements (integers, reals, logicals, strings)

II. Vectors (ordered sets of similar elements)

III. Matrices (ordered sets of vectors of the same length)

IV. Data Frames (ordered sets of vectors where the vectors can be different, but all the same length)

V. Lists (ordered sets of anything, can be different lengths)
Structure examples

> x <- c(1,2,4)
> y <- c(letters[1:6],LETTERS[1:4])
> z <- seq(10,28,2)
> X <- matrix(1:20,ncol=4)
> Y <- matrix(c(11,22,44,4,15,42),ncol=3,byrow=TRUE)
> yz.df <- data.frame(A = y,b=z)
> L <- list(a=x,b=y,c=z,d=X,e=Y,f =yz.df)

> x
[1] 1 2 4
> y
[1] "a" "b" "c" "d" "e" "f" "A" "B" "C" "D"
> z
[1] 10 12 14 16 18 20 22 24 26 28
> X
[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
> Y
[,1] [,2] [,3]
[1,] 11  22  44
[2,]  4  15  42
> yz.df
   A  b
1 a 10
2 b 12
3 c 14
4 d 16
5 e 18
6 f 20
7 A 22
8 B 24
9 C 26
10 D 28
### Structure list

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<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>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b</td>
<td>c</td>
<td>d</td>
<td>e</td>
<td>f</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
</tr>
</tbody>
</table>

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>12</td>
<td>14</td>
<td>16</td>
<td>18</td>
<td>20</td>
</tr>
</tbody>
</table>

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>[1,1]</td>
<td>[1,2]</td>
<td>[1,3]</td>
<td>[1,4]</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>a</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>b</td>
<td>12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>c</td>
<td>14</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>d</td>
<td>16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>e</td>
<td>18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>f</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>A</td>
<td>22</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>B</td>
<td>24</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>C</td>
<td>26</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>D</td>
<td>28</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Structure of lists

> str(L)
List of 6

$ a: num [1:3] 1 2 4
$ b: chr [1:10] "a" "b" "c" "d" ...
$ c: num [1:10] 10 12 14 16 18 20 22 24 26 28
$ d: int [1:5, 1:4] 1 2 3 4 5 6 7 8 9 10 ...
$ e: num [1:2, 1:3] 11 4 22 15 44 42
$ f:'data.frame': 10 obs. of 2 variables:
  ..$ A: Factor w/ 10 levels "a","A","b","B",...
     : 1 3 5 7 9 10 2 4 6 8
  ..$ b: num [1:10] 10 12 14 16 18 20 22 24 26 28
Accessing elements

> L$d

[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

> L[[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
Manipulating data

I. Consider scoring a multiple choice test
   A. 10 items, 100 subjects
   B. create the scoring key
   C. score it
Scoring a multiple score test

```r
> set.seed(42)
> my.items <- matrix(sample(5,500,replace=TRUE),ncol=5)
> my.key <- c(1,2,3,2,4)
> my.scores <- t(t(my.items)==my.key[])  #these are TRUE or FALSE
> my.scores <- t(t(my.items)==my.key[]) +0 #these are 1s or 0s
> my.total <- rowSums(my.scores)           #total score
> describe(my.total)

  var   n mean   sd median trimmed  mad min max range skew kurtosis   se
  1   1 100 0.92 0.82      1    0.84 1.48   0   3     3 0.68    -0.03 0.08

> describe(my.scores)

  var   n mean   sd median trimmed  mad min max range skew kurtosis   se
  1   1 100 0.19 0.39      0    0.11 0.50  0   1     1 0.43    -0.71 0.04
  2   2 100 0.14 0.35      0    0.05 0.50  0   1     1 0.43    -0.71 0.03
  3   3 100 0.25 0.44      0    0.19 0.50  0   1     1 0.23    -0.71 0.04
  4   4 100 0.22 0.42      0    0.15 0.50  0   1     1 0.23    -0.71 0.03
  5   5 100 0.12 0.33      0    0.02 0.50  0   1     1 0.23    -0.71 0.03
```
> my.key
[1] 1 2 3 2 4
> dim(my.items)
[1] 100  5
> dim(t(my.items))
[1]  5 100
> t(my.items)[,1:10]
[1,]  5  5  2  5  4  3  4  1  4  4
[2,]  4  2  2  2  5  5  4  4  3  1
[3,]  5  3  5  3  1  3  5  3  2  2
[4,]  3  3  1  2  5  5  2  1  2  2
[5,]  1  3  4  3  5  1  5  2  1  2
> x <- t(my.items) == my.key
> x[,1:10]
[1,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
[2,] FALSE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
[3,] FALSE  TRUE FALSE  TRUE FALSE  TRUE FALSE  TRUE FALSE FALSE
[4,] FALSE FALSE FALSE  TRUE FALSE FALSE  TRUE FALSE  TRUE  TRUE
[5,] FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
> x0 <- x+0
> x0[,1:10]
[1,]    0    0    0    0    0    0    0    1    0     0
[2,]    0    1    1    1    0    0    0    0    0     0
[3,]    0    1    0    1    0    1    0    1    0     0
[4,]    0    0    0    1    0    0    1    0    1     1
[5,]    0    0    1    0    0    0    0    0    0     0
More on multiple choice

```r
> my.scores <- t(t(my.items)==my.key[])  #these are TRUE or FALSE
> head(my.scores)
[1,] FALSE FALSE FALSE FALSE FALSE
[2,] FALSE  TRUE  TRUE FALSE FALSE
[3,] FALSE  TRUE FALSE FALSE  TRUE
[4,] FALSE  TRUE  TRUE  TRUE FALSE
[5,] FALSE FALSE FALSE FALSE FALSE
[6,] FALSE FALSE  TRUE FALSE FALSE

> my.scores <- t(t(my.items)==my.key[]) +0  #these are 1s or 0s
> my.scores
[1,]    0    0    0    0    0
[2,]    0    1    1    0    0
[3,]    0    1    0    0    1
[4,]    0    1    1    1    0
[5,]    0    0    0    0    0
[6,]    0    0    1    0    0
...
> my.total <- rowSums(my.scores)
> head(my.total)
[1] 0 2 2 3 0 1
```
> data(iqitems)
> iq.keys <- c(4,4,3,1,4,3,2,3,1,4,1,3,4,3)
> score.multiple.choice(iq.keys,iqitems)

```
$item.stats
   key 0 1 2 3 4 5 6  r  n mean   sd skew kurtosis se
iq1  4 0.04 0.01 0.03 0.09 0.80 0.02 0.01 0.59 1000 0.80 0.40 -1.51 0.27 0.01
iq8  4 0.03 0.10 0.02 0.80 0.01 0.04 0.39 1000 0.80 0.40 -1.49 0.22 0.01
iq10 3 0.10 0.22 0.09 0.37 0.04 0.13 0.04 0.35 1000 0.37 0.48  0.53 -1.72 0.02
iq15 1 0.03 0.65 0.16 0.15 0.00 0.00 0.00 0.35 1000 0.65 0.48 -0.63 -1.40 0.01
iq20 4 0.03 0.02 0.03 0.85 0.02 0.01 0.02 0.42 1000 0.85 0.35 -2.00  2.01 0.02
iq44 3 0.03 0.10 0.06 0.64 0.02 0.14 0.01 0.42 1000 0.64 0.48 -0.61 -1.64 0.02
iq47 2 0.04 0.08 0.59 0.06 0.11 0.07 0.05 0.51 1000 0.59 0.49 -0.35 -1.88 0.02
iq2  3 0.07 0.08 0.31 0.32 0.15 0.05 0.02 0.26 1000 0.32 0.46  0.80 -1.37 0.01
iq11 1 0.04 0.87 0.03 0.01 0.01 0.01 0.04 0.54 1000 0.87 0.34 -2.15  2.61 0.01
iq16 4 0.05 0.05 0.08 0.74 0.01 0.00 0.56 1000 0.74 0.44 -1.11 -0.77 0.01
iq32 1 0.04 0.54 0.02 0.14 0.10 0.04 0.12 0.50 1000 0.54 0.50 -0.17 -1.97 0.02
iq37 3 0.07 0.10 0.09 0.26 0.13 0.02 0.34 0.23 1000 0.26 0.44  1.12 -0.74 0.01
iq43 4 0.04 0.07 0.04 0.02 0.78 0.03 0.00 0.50 1000 0.78 0.41 -1.35 -0.18 0.01
iq49 3 0.06 0.27 0.09 0.32 0.14 0.08 0.05 0.28 1000 0.32 0.47  0.79 -1.38 0.01

$alpha
   Averages
Averages 0.63

$av.r
   Averages
Averages 0.11
```
Examine the structure

```r
> iq.scores <- score.multiple.choice(iq.keys, iqitems, short=FALSE)
> str(iq.scores)
List of 4
$ scores    : num [1:1000, 1] 0.429 0.357 0.571 0.571 0.571 ... 
  ..- attr(*, "dimnames")=List of 2 
  .. ..$ : chr [1:1000] "72" "95" "100" "136" ... 
  .. ..$ : chr "Averages"
$ item.stats: 'data.frame': 14 obs. of 15 variables:
  ..$ key     : num [1:14] 4 4 3 1 4 3 2 3 1 4 ... 
  ..$ 0       : num [1:14] 0.035 0.028 0.1 0.032 0.028 0.029 0.045 0.071 0.036 0.047 ... 
  ..$ 1       : num [1:14] 0.01 0.104 0.223 0.651 0.024 0.097 0.08 0.078 0.866 0.054 ... 
  ..$ 2       : num [1:14] 0.034 0.006 0.088 0.163 0.027 0.055 0.586 0.308 0.027 0.079 ... 
  ..$ 3       : num [1:14] 0.088 0.016 0.371 0.153 0.034 0.645 0.063 0.315 0.011 0.07 ... 
  ..$ 4       : num [1:14] 0.801 0.799 0.044 0.001 0.854 0.019 0.106 0.154 0.009 0.743 ... 
  ..$ 5       : num [1:14] 0.024 0.009 0.133 0.019 0.145 0.067 0.053 0.008 0.007 ... 
  ..$ 6       : num [1:14] 0.008 0.038 0.041 0.014 0.01 0.053 0.021 0.043 0 ... 
  ..$ r       : num [1:14] 0.591 0.395 0.346 0.35 0.418 ... 
  ..$ n       : num [1:14] 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000 ... 
  ..$ mean    : num [1:14] 0.801 0.799 0.371 0.651 0.854 0.645 0.586 0.315 0.866 0.743 ... 
  ..$ sd      : num [1:14] 0.399 0.401 0.483 0.477 0.353 ... 
  ..$ skew    : num [1:14] -1.506 -1.49 0.533 -0.633 -2.002 ... 
  ..$ kurtosis: num [1:14] 0.267 0.22 -1.717 -1.601 2.01 ... 
  ..$ se      : num [1:14] 0.0126 0.0127 0.0153 0.0151 0.0112 ... 
$ alpha     : num [1, 1] 0.63 
  ..- attr(*, "dimnames")=List of 2 
  .. ..$ : chr "Averages"
  .. ..$ : chr "Averages"
$ av.r      : num [1, 1] 0.11 
  ..- attr(*, "dimnames")=List of 2 
  .. ..$ : chr "Averages"
  .. ..$ : chr "Averages"
```

Use the relevant part of a list

```r
> dim(iq.scores)
NULL
> length(iq.scores)
[1] 4
> dim(iq.scores$scores)
[1] 1000 1
> describe(iq.scores$scores)

           var    n mean   sd median trimmed  mad min  max range  skew kurtosis   se
Averages 1 1000 0.61 0.18   0.64    0.63 0.11   0 0.93 0.93  -1.07     1.44 0.01
```
Data Manipulation

I. standard arithmetic and logical operations

II. matrix operations including transpose, inner product, outer product, diagonal, trace, invert

III. searching, sorting, merging

IV. data cleaning by logical commands
### Basic data description

```r
summary(My.data)
epiE  epiS  epiImp  epilie  epiNeur  bfagree  bfcon
Min. :1.00  Min. : 0.000  Min. : 0.000  Min. :0.000  Min. : 0.00  Min. : 74.0  Min. :53.0
1st Qu.:11.00  1st Qu.: 6.000  1st Qu.: 3.000  1st Qu.: 1.000  1st Qu.: 7.00  1st Qu.:126.0  1st Qu.:114.0
Median :14.00  Median : 8.000  Median : 4.000  Median : 2.000  Median :10.00  Median :126.0  Median :114.0
Mean :13.68  Mean : 7.978  Mean : 4.784  Mean : 2.377  Mean :10.41  Mean :125.0  Mean :113.3
3rd Qu.:16.00  3rd Qu.:10.000  3rd Qu.: 6.000  3rd Qu.: 3.000  3rd Qu.:14.00  3rd Qu.:136.5  3rd Qu.:128.5
Max. :99.00  Max. :99.000  Max. :99.000  Max. : 7.000  Max. :23.00  Max. :167.0  Max. :178.0
bfext  bfneur  bfopen  bdi  traitanx  stateanx
Min. :  8.0  Min. : 34.00  Min. : 73.0  Min. : 0.000  Min. :22.00  Min. :21.00
1st Qu.: 87.5  1st Qu.: 70.00  1st Qu.:110.0  1st Qu.: 3.000  1st Qu.:32.00  1st Qu.:32.00
Median :104.0  Median : 90.00  Median :125.0  Median : 6.000  Median :38.00  Median :38.00
3rd Qu.:118.0  3rd Qu.:104.00  3rd Qu.:136.5  3rd Qu.: 9.000  3rd Qu.:44.00  3rd Qu.:46.50
Max. :168.0  Max. :152.00  Max. :173.0  Max. :27.000  Max. :71.00  Max. :79.00

> describe(My.data)

<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>epiE</td>
<td>1</td>
<td>13.33</td>
<td>4.14</td>
<td>13.49</td>
<td>4.45</td>
<td>1</td>
<td>22</td>
<td>21</td>
<td>-0.33</td>
<td>-0.06</td>
<td>0.27</td>
<td></td>
</tr>
<tr>
<td>epiS</td>
<td>2</td>
<td>7.58</td>
<td>2.69</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>
<td>0.18</td>
<td></td>
</tr>
<tr>
<td>epiImp</td>
<td>3</td>
<td>4.37</td>
<td>1.88</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>
<td>0.12</td>
<td></td>
</tr>
<tr>
<td>epilie</td>
<td>4</td>
<td>2.38</td>
<td>1.50</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>
<td>0.10</td>
<td></td>
</tr>
<tr>
<td>epiNeur</td>
<td>5</td>
<td>10.41</td>
<td>4.90</td>
<td>10.39</td>
<td>4.45</td>
<td>0</td>
<td>23</td>
<td>23</td>
<td>0.06</td>
<td>-0.50</td>
<td>0.32</td>
<td></td>
</tr>
<tr>
<td>bfagree</td>
<td>6</td>
<td>125.00</td>
<td>18.14</td>
<td>125.26</td>
<td>17.79</td>
<td>74</td>
<td>167</td>
<td>93</td>
<td>-0.21</td>
<td>-0.27</td>
<td>1.19</td>
<td></td>
</tr>
<tr>
<td>bfcon</td>
<td>7</td>
<td>113.25</td>
<td>21.88</td>
<td>113.78</td>
<td>22.24</td>
<td>53</td>
<td>178</td>
<td>125</td>
<td>-0.02</td>
<td>0.23</td>
<td>1.44</td>
<td></td>
</tr>
<tr>
<td>bfext</td>
<td>8</td>
<td>102.18</td>
<td>26.45</td>
<td>102.99</td>
<td>22.24</td>
<td>8</td>
<td>168</td>
<td>160</td>
<td>-0.41</td>
<td>0.51</td>
<td>1.74</td>
<td></td>
</tr>
<tr>
<td>bfneur</td>
<td>9</td>
<td>87.97</td>
<td>23.34</td>
<td>87.70</td>
<td>23.72</td>
<td>34</td>
<td>152</td>
<td>118</td>
<td>0.07</td>
<td>-0.55</td>
<td>1.54</td>
<td></td>
</tr>
<tr>
<td>bfopen</td>
<td>10</td>
<td>123.43</td>
<td>20.51</td>
<td>123.78</td>
<td>20.76</td>
<td>73</td>
<td>173</td>
<td>100</td>
<td>-0.16</td>
<td>-0.16</td>
<td>1.35</td>
<td></td>
</tr>
<tr>
<td>bdi</td>
<td>11</td>
<td>6.78</td>
<td>5.78</td>
<td>5.97</td>
<td>4.45</td>
<td>0</td>
<td>27</td>
<td>27</td>
<td>1.29</td>
<td>1.50</td>
<td>0.38</td>
<td></td>
</tr>
<tr>
<td>traitanx</td>
<td>12</td>
<td>39.01</td>
<td>9.52</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>
<td>0.63</td>
<td></td>
</tr>
<tr>
<td>stateanx</td>
<td>13</td>
<td>39.85</td>
<td>11.48</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>
<td>0.76</td>
<td></td>
</tr>
</tbody>
</table>
```
R graphics (base)

I. Multiple graphics packages
   A. base graphics
   B. lattice
   C. ggobi

II. Following examples from base graphics
> data(sat.act)
> pairs.panels(sat.act)
pairs.panels

pairs.panels(sat.act, scale=TRUE)
Do not draw bar graphs

A particularly uninformative graph

```r
> barplot(colMeans(na.omit(X.df)), ylim=c(0,14), main="A particularly uninformative graph")
> box()
```
Somewhat better

Somewhat more informative

```
error.bars(X.df,bars=TRUE,ylim=c(0,14),
           main="Somewhat more informative")
```
boxplot(X.df, main="Better yet")
stripchart(X.df, method="stack", vertical=TRUE, main="Perhaps better")
> error.bars(X.df, main="Just error bars")
notched boxplots

> boxplot(X.df, main="Better yet", notch=TRUE)
stripchart + error.bars

> stripchart(X.df, method="stack", vertical=TRUE, main="Add error bars")
> error.bars(X.df, add=TRUE)
Alternatives with larger data sets

I. The violin plot shows density distributions
II. Available in the vioplot package
mu <- 2
si <- 0.6
bimodal <- c(rnorm(1000, -mu, si), rnorm(1000, mu, si))
uniform <- runif(2000, -4, 4)
normal <- rnorm(2000, 0, 3)
vioplot(bimodal, uniform, normal)
boxplot(bimodal, uniform, normal)
An alternative

> bnu <- data.frame(bimodal, normal, uniform)
> pairs.panels(bnu)
plot(1:10)
specify labels and title

```r
> plot(1:10, xlab = "Label the x axis", ylab = "label the y axis",
+      main = "And add a title and new data", pch = 21, col = "blue")
> points(1:4, 3:6, bg = "red", pch = 22)
```
Change the ranges

> plot(1:10, xlab = "x is oversized", ylab = "y axis label",
+ main = "Change the axis sizes", pch = 23, bg = "blue",
+ xlim = c(-5, 15), ylim = c(0, 20))
> points(1:4, 13:16, bg = "red", pch = 24)
Plot the same data as a line graph

> plot(1:10,ylab="y axis label",main="Line graph",pch=23,bg="blue",type="l")
Show the data points, add a line

```r
> plot(1:10, 2:11, xlab="X axis", ylab="y axis label", + main="Line graphs with and without points", pch=23, bg="blue", type="b", ylim=c(0, 15))
> points(1:10, 12:3, type="l", lty="dotted")
```
Regression plots

\[ SATQ = 208 + 0.66 \times SATV \]

```r
> data(sat.act)
> with(sat.act, plot(SATQ~SATV, main="SAT Quantitative varies with SAT Verbal"))
> model = lm(SATQ~SATV, data=sat.act)
> abline(model)
> lab <- paste("SATQ = ", round(model$coef[1], 2)," + ", round(model$coef[2], 2)," \times SATV")
> text(600,200,lab)
```
Two groups

SATQ varies by SATV and gender

> data(sat.act)
> color <- c("blue","red")
> with(sat.act,plot(SATQ~SATV,col=color[gender],main="SATQ varies by SATV and gender"))
> by(sat.act,sat.act$gender,function(x) abline(lm(SATQ~SATV,data=x)))
The actual regression

```r
> by(sat.act, sat.act$gender, function(x) lm(SATQ ~ SATV, data = x))
sat.act$gender: 1

Call:
lm(formula = SATQ ~ SATV, data = x)

Coefficients:
(Intercept)         SATV
210.2046       0.6917

---

- sat.act$gender: 2

Call:
lm(formula = SATQ ~ SATV, data = x)

Coefficients:
(Intercept)         SATV
209.2093       0.6334
```
Error diagnostics

```r
> op <- par(mfrow=c(2,2))
> plot(lm(SATQ~SATV,data=sat.act))
> op <- par(mfrow=c(1,1))
```
Boxplots by group

SATV by gender

boxplot(sat.act$SATV~sat.act$gender,main="SATV by gender")
More complex graphs

I. Using the Rgraphviz package (installed from Bioconductor rather than CRAN)

II. Using the Social Network Analysis (sna) package.
A colorful demonstration...
gplot(rgraph(5, diag=TRUE), diag=TRUE, vertex.cex=1:5, vertex.sides=3:8,
vertex.col=1:5, vertex.border=2:6, vertex.rot=(0:4)*72,
displaylabels=TRUE, label.bg="gray90")
Cluster analysis with Rgraphviz output

The Holzinger-Harman 24 mental measurement problem

> data(Harman74.cor)
> ic <- ICLUST(Harman74.cor$cov, title="The Holzinger-Harman 24 mental measurement problem")
A bifactor solution

```
data(bifactor)
> om <- omega(Thurstone, main="A bifactor solution to a Thurstone data set")
```
Creating matrix input for a graph

```r
> fxs <- structure.list(9, list(X1=c(1,2,3), X2=c(4,5,6), X3=c(7,8,9)))
> phi <- phi.list(4, list(F1=c(4), F2=c(4), F3=c(4), F4=c(1,2,3)))
> fyx <- structure.list(3, list(Y=c(1,2,3)),"Y")
>
> fxs
X1 X2 X3
[1,] "a1" "0" "0"
[2,] "a2" "0" "0"
[3,] "a3" "0" "0"
[4,] "0" "b4" "0"
[5,] "0" "b5" "0"
[6,] "0" "b6" "0"
[7,] "0" "0" "c7"
[8,] "0" "0" "c8"
[9,] "0" "0" "c9"

> phi
F1 F2 F3 F4
F1 "1" "0" "0" "rda"
F2 "0" "1" "0" "rdb"
F3 "0" "0" "1" "rdc"
F4 "rad" "rbd" "rcd" "1"

> fyx
Y
[1,] "Ya1"
[2,] "Ya2"
[3,] "Ya3"
```
Showing the matrices

> sg3 <- structure.graph(fxs, phi, fyx)
Using the maps package

> library(maps)
> map("county")