

# Introduction to computing in “R”

Practicum workshop February 6, 2008

## R Console

## Comments

*“R” is in many ways comparable with SAS. The software is predominately syntax driven and relies on its user to know the “R” language (which in many ways resembles the “S” and UNIX programming languages). “R” is comparable in structure and conceptual arrangement to other syntax based software packages. Similarities and dissimilarities with other software packages will be pointed out to facilitate an easier transition into “R.”*

R version 2.5.1 (2007-06-27)  
Copyright (C) 2007 The R Foundation for Statistical Computing  
ISBN 3-900051-07-0

R is free software and comes with ABSOLUTELY NO WARRANTY.

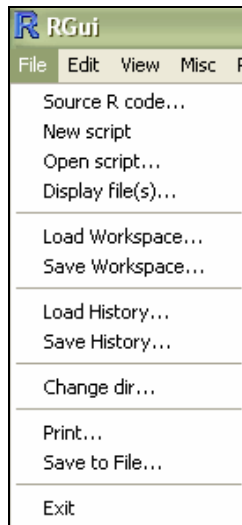
You are welcome to redistribute it under certain conditions.

Type 'license()' or 'licence()' for distribution details.  
[...]

```
> help( )  
> search( )  
> library( )
```

Upon running R the program will produce an introductory statement specifying the software version and basic helpful R commands.

Several global “R” functions provide additional assistance. Such functions ending in “( )” call on all elements available in the environment.



Source R code: Saved R code  
New Script: New syntax script (like .txt file)  
Open script: Open a saved script  
Display file(s): Open all saved “R” materials

Load Workspace: Load a previous workspace  
Save Workspace: Save all your current content

Load History: Load previously saved syntax  
Save History: Save all submitted syntax

Change dir: Changes the directory of temp. files, but resets to default after closing “R”

Print: Prints visible console content  
Save to File: Saves visible console content

Click “Packages,” then “Load package...” if the package is already installed or “Install package(s)...”. Find “rgl”, then click “OK”.

*or use the syntax*

```
> install.packages(“rgl”)  
> library(rgl)
```

“R” packages if they have already been downloaded from a CRAN mirror site can be loaded using this procedure. If the package has not been downloaded it can be installed using the “Install package(s)...” option. Also, an installed package can be loaded by specifying library(name of package).

```
> demo( )  
> demo(lollipop3d)  
> demo(bivar)
```

Most “R” packages will come with a demonstration of the included functions.

## R Console

## Comments

The "R" Environment contains the software's libraries with all the available datasets, expansion packages and macros. As compared to SAS the Log and Editor windows are consolidated into a single interface the "R" Console. "R" Environment controlling functions and options are not available over a separate window or through drop down menus as is the case in other software packages. Most, but not all, "R" Environment items must be called on using syntax.

```
> 2 + 2
[1] 4
> 2 +
+ 2
[1] 4
> 2 + (3xz
Error: syntax error, [...]
> z <- 2 # You can also type z = 2
> z
[1] 2
> 2 + (3*z)
[1] 8
```

"R" will provide intuitive error messages regarding the submitted syntax. Unlike in SPSS or SAS these comments are printed right in the console.

"R" treats all of its entered elements as matrices and vectors, consequently, these must be conformable in order for operations to work. Also, each operation result should be stored into a new "R" object.

```
> library(MASS)
> data( )
> ?CO2
> CO2 # You can also call on CO2 [ ]
  Plant      Type Treatment  conc uptake
1  Qn1      Quebec nonchilled  95  16.0
[...]
```

Once a library is loaded all datasets within the library are dropped into the "R" Workspace. The Workspace is the underlying environment for the "R" Console, which acts as the temporary library catalog (like the "Work" folder in SAS).

```
> summary(CO2)
  Plant      Type      Treatment
Qn1 : 7 Quebec  :42 nonchilled:42 [...]
> attach(CO2)
The following object(s) are masked...
> plot(CO2)
> fix (CO2) # Change "Treatment" to "treat"
> attach(CO2)
> plot(treat, uptake)
```

General "R" operations to preliminarily explore a loaded dataset. "R" will graph the loaded data intuitively or via the embedded plotting parameters.

```
> windows( )
> plot(uptake, pch=20); title("Grass Carbon Dioxide
Uptake", col.main="red")
> line(uptake)
Call: line(uptake) [...]
> l.up <- line(uptake)
> abline( l.up, col="blue")
> abline( 30, 0, lty=4, col="green"); abline( v=60, lty=2,
col="orange")
> identify(uptake)
```

Graphs, tables, charts and all other "R" devices can only be manipulated using syntax. Also, once a graph is created added elements cannot be removed.

## R Console

## Comments

*Objects and datasets can be imported into “R”. Various forms of data can be loaded, including data files from specific statistical software packages such as SPSS, SAS etc. For the following exercise create a fictitious dataset in Notepad on your Desktop and save it as a “.txt” file.*

```
> ?read.table
> example <- read.table(file="C:/Documents and
Settings/Desktop/example.txt", header=TRUE)
> example
  ID age  GPA
1  1  23  4.0
2  2  21  3.8
[...]
```

```
> example[1,2] = 28
> example
  ID age  GPA
1  1  28  4.0
2  2  21  3.8
[...]
```

Click “Packages,” then “Load package...” and select “foreign” Click “OK”.

```
> ?read.spss
```

```
> ?linear.model
No documentation for 'linear.model' [...]
```

```
> help.search("linear model")
> ? lm
> lm
function (formula, data, subset, weights,
na.action, [...])
```

```
> length(uptake)
[1] 84
> s.pre <- sort ( sample (1:400, 84, replace=TRUE) )
> plot( s.pre )
> s.up <- sort(uptake)
> plot( s.pre, s.up, pch=20)
```

```
> lm.uptake <- lm( s.up ~ s.pre )
> lm.uptake
Call:
lm(formula = s.up ~ s.pre) [...]
```

```
> summary (lm.uptake)
> lm.uptake [ ]
```

```
> abline(lm.uptake, lty=3, col="purple", lwd=3)
> plot (lm.uptake)
```

Reading in datasets can be problematic. Most issues can be resolved by knowing the dataset beforehand (i.e. how many variables are there, are variable labels included etc.).

Any function documentation can be read by preceding the function with a “?”. If the function is invalid, undefined or simply does not exist use the alternative `help.search(“”) instead.`

One of the most important steps in computing / programming in “R” is to ascribe the results of operations to new “R” objects. In return these “R” objects are used in other operations building more complex and useful functions.

Functions which come from “R” packages usually have built in graphic, analytic and output stipulations.

## R Console

## Comments

The "R" software packages deals quite differently with missing data than SPSS or SAS. In "R" a missing number is indicated with a "NA," Not Available, and missing non-numbers with "NaN," Not a Number. It is vital to remember that once missing values are coded in a dataset it is very problematic operating with that particular dataset or reversing the coding.

```
> up.res <- lm.uptake$residuals
> summary(up.res)
> plot(up.res)
> abline(0,0); abline(-2,0, col="red")
```

Individual variables from datasets or elements from output lists can be stored separately into new "R" objects using the assign procedures and specifying the object (using the "\$" sign).

```
> up.res
      1          2          3
-2.83362861 -1.33157849 -0.32747827 [...]
> res.trans <- function(x) { sqrt(x+2) }
> up.trans <- res.trans ( up.res )
Warning message:
NaNs produced in: sqrt(x+2)
> up.trans
      1          2          3
      NaN  0.8175705  1.2932601 [...]
> windows ( )
> plot(up.trans)
```

When a called-on function produces NaNs or NAs "R" will complete the function but replace those values which have caused the error. "R" will also inform you with a warning message imbeded in the console.

```
> is.nan( up.trans )
      1      2      3      83      84
TRUE FALSE FALSE [...] TRUE TRUE
> up.trans [1]
      1
      NaN
> up.trans [1] <- 0
> up.trans
      1          2          3          84
0.0000  0.8175  1.2932 [...]  NaN
> is.nan(up.trans)
      1      2      3      83      84
FALSE FALSE FALSE [...] TRUE TRUE
```

NaNs and NAs can be detected with logic functions. Individual NaNs or NAs can be replaced by assigning a new number (scalar) to the particular spot in the vector or dataset.

```
> ?ifelse
> up.fix <- ifelse ( is.nan(up.trans), 3, up.trans )
> up.fix
      1          2          3          84
0.0000  0.8175  1.2932 [...]  0.0000
> windows( )
> plot( up.fix )
> is.nan( up.fix )
      1      2      3      83      84
FALSE FALSE FALSE [...] FALSE FALSE
```

The replacement of NaNs and NAs in an entire vector/matrix has to be done by a conjunct work function (ifelse) and logic function (is.nan or is.na). As compared to individually exchanged integers "R" will remember in its registry (if the original dataset was a dataframe) that those NaNs and NAs replaced by such a function were once not a real number.

## R Console

## Comments

*"R" datasets and created objects are manipulated and managed through the "R" Workspace Image. These objects can be exported and saved into various target files.*

```
> length(up.fix)
[1] 84
> fix.data <- matrix (up.fix, 12, 7)
> fix.data
      [,1]      [,2]
[1,] 0.0000000 0.8709916
[2,] 0.8175705 0.8191014 [...]
```

Data can be transformed into any desirable form. Matrix, dataframe and general data functions make such transformations from single vectors (or arrays) fairly simple.

```
> ?write.table
> write.table (fix.data, file="sample_data.txt")

or

> write.table (fix.data, file="C:/Documents and
Settings/Desktop/sample_data.txt", col.names = TRUE)
```

Data objects can be exported (saved) as one of many desirable formats such as ".txt", ".csv", or ".dbf" files all of which can be either comma, space or character separated with or without variable and/or case labels included.

```
> ? ls
> ls ()
[1] "a"      "age"
[6] "Brand"  "calorie" [...]
```

```
> ? rm

> up.fix
  1  2  3
0.00 0.82 1.29 [...]
```

```
> rm (up.fix)
> up.fix
Error: object "up.fix" not found
> ls ()

> rm ( list = ls() )
> ls ()
character(0)
```

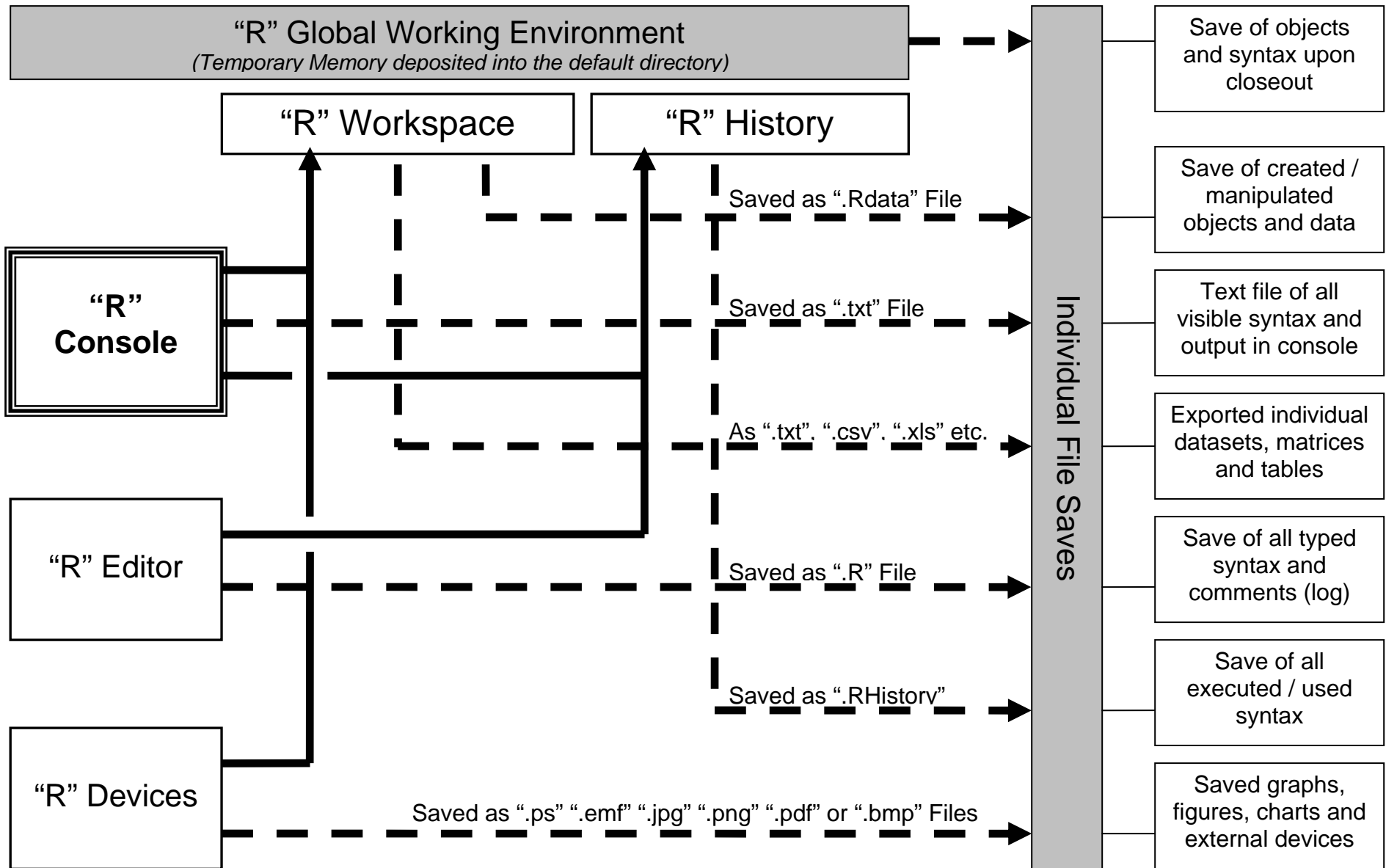
Objects in the Workspace Image can be monitored and managed using the *list objects* and *remove objects* function. However, objects removed from the Workspace Image will not be saved.

*Last but not least your should try to save some of the created objects and syntax so as to learn how to save your workspace for "R" but also to get a sense for the nature of the saved files (see Appendix A). Remember, that unless you change the directory, either manually by designating a new temporary directory from the File menu, or by specifying a specific path to your exported files, "R" will save everything to the default "R" system catalog.*

*Also, if you ever have a question regarding programming in "R" do not hesitate to use the "R" mailing list. Individuals contributing to this mailing list are not only highly knowledgeable but also exceptionally helpful. Often you will get new and improved ideas (approaches) by interacting with others. That said, do not always take every piece of advice as absolute truth (in particular regarding statistics). To err is to be human. Verify and double check.*

# Appendix A The "R" Environment

Practicum workshop February 6, 2008



---

## Appendix B

### Useful Free “R” Manuals

*Practicum workshop February 6, 2008*

---

- 1) **R Installation and Administration** (60 pages)  
By R Development Core Team  
  
Available: <http://cran.r-project.org/doc/manuals/R-admin.pdf>  
  
Content:
  - 1 Obtaining R
  - 2 Installing R under Unix-alikes
  - 3 Installing R under Windows
  - 4 Installing R under Mac OS X
  - 5 Running R
  - 6 Add-on packages
  - 7 Internationalization and Localization
  - 8 Choosing between 32- and 64-bit builds
  - 9 The standalone Rmath library
  
- 2) **R Internals** (34 Pages)  
By R Development Core Team  
  
Available: <http://cran.r-project.org/doc/manuals/R-ints.pdf>  
  
Content:
  - 1 R Internal Structures
  - 2 .Internal vs .Primitive
  - 3 Internationaliation in the R sources
  - 4 R coding standards
  - 5 Testing R code
  
- 3) **The R Reference Index** (2667 pages)  
By R Development Core Team  
  
Available: <http://cran.r-project.org/doc/manuals/fullrefman.pdf>  
  
Content:
  - 1 The base package
  - 2 The datasets package
  - 3 The grDevices package
  - 4 The graphics package
  - 5 The grid package
  - 6 The methods package
  - 7 The stats package
  - 8 The tools package
  - 9 The utils package
  - 10 The KernSmooth package
  - 11 The MASS package
  - 12 The boot package

- 13 The class package
- 14 The cluster package
- 15 The codetools package
- 16 The foreign package
- 17 The lattice package
- 18 The mgcv package
- 19 The nlme package
- 20 The nnet package
- 21 The rcompjen package
- 22 The rpart package
- 23 The spatial package
- 24 The splines package
- 25 The stats4 package
- 26 The survival package
- 27 The tcltk package

4) **An Introduction to R** (100 pages)  
By W. N. Venables, D. M. Smith and the R Development Core Team

Available: <http://cran.r-project.org/doc/manuals/R-intro.pdf>

- Content:
- 1 Introduction and preliminaries
  - 2 Simple manipulations; numbers and vectors
  - 3 Objects, their modes and attributes
  - 4 Ordered and unordered factors
  - 5 Arrays and matrices
  - 6 Lists and data frames
  - 7 Reading data from \_les
  - 8 Probability distributions
  - 9 Grouping, loops and conditional execution
  - 10 Writing your own functions
  - 11 Statistical models in R

5) **R Language Definition** (60 pages)  
By R Development Core Team

Available: <http://cran.r-project.org/doc/manuals/R-lang.pdf>

- Content:
- 1 Introduction
  - 2 Objects
  - 3 Evaluation of expressions
  - 4 Functions
  - 5 Object-oriented programming
  - 6 Computing on the language
  - 7 System and foreign language interfaces
  - 8 Exception handling
  - 9 Debugging
  - 10 Parser

- 6) **R Data Import / Export** (34 pages)  
By R Development Core Team
- Available: <http://cran.r-project.org/doc/manuals/R-data.pdf>
- Content:
- 1 Introduction
  - 2 Spreadsheet-like data
  - 3 Importing from other statistical systems
  - 4 Relational databases
  - 5 Binary \_les
  - 6 Connections
  - 7 Network interfaces
  - 8 Reading Excel spreadsheets
- 7) **simpleR – Using R for Introductory Statistics** (114 pages)  
By John Verzani
- Available: <http://cran.r-project.org/doc/contrib/Verzani-SimpleR.pdf>
- Content:
- 1 Introduction
  - 2 Data
  - 3 Univariate Data
  - 4 Bivariate Data
  - 5 Multivariate Data
  - 6 Random Data
  - 7 Simulations
  - 8 Exploratory Data Analysis
  - 9 Confidence Interval Estimation
  - 10 Hypothesis Testing
  - 11 Two-sample Test
  - 12 Chi-square Test
  - 13 Regression Analysis
  - 14 Multiple Regression Analysis
  - 15 Analysis of Variance
- 8) **Writing R Extensions** (125 pages)  
By R Development Core Team
- Available: <http://cran.r-project.org/doc/manuals/R-exts.pdf>
- Content:
- 1 Creating R packages
  - 2 Writing R documentation \_les
  - 3 Tidying and pro\_ling R code
  - 4 Debugging
  - 5 System and foreign language interfaces
  - 6 The R API: entry points for C code
  - 7 Generic functions and methods
  - 8 Linking GUIs and other front-ends to R

9) **Practical Regression and ANOVA using R**

(213 pages)

Julian J. Faraway

Available: <http://probability.ca/cran/doc/contrib/Faraway-PRA.pdf>

Content:

- 1 Introduction
- 2 Estimation
- 3 Inference
- 4 Errors in Predictors
- 5 Generalized Least Squares
- 6 Testing for Lack of Fit
- 7 Diagnostics
- 8 Transformation
- 9 Scale Changes, Principal Components and Collinearity
- 10 Variable Selection
- 11 Statistical Strategy and Model Uncertainty
- 12 Chicago Insurance Redlining - a complete example
- 13 Robust and Resistant Regression
- 14 Missing Data
- 15 Analysis of Covariance
- 16 ANOVA

10) **R Manual to Accompany Agresti's Categorical Data Analysis (2002)** (278 pages)

By Laura A. Thompson

Available: <https://home.comcast.net/~lthompson221/Splplusdiscrete2.pdf>

Content:

- Introduction and Changes from First Edition
- 1 Distributions and Inference for Categorical Data
- 2 Describing Contingency Tables
- 3 Inference for Contingency Tables
- 4 Generalized Linear Models
- 5 Logistic Regression
- 6 Building and Applying Logistic Regression Models
- 7 Logit Models for Multinomial Responses
- 8 Loglinear Models for Contingency Tables
- 9 Building and Extending Loglinear Models
- 10 Models for Matched Pairs
- 11 Analyzing Repeated Categorical Response Data
- 12 Random Effects
- 13 Other Mixture Models for Categorical Data

---

## Appendix C Additional “R” Web Resources

*Practicum workshop February 6, 2008*

---

- 1) **The R Project for Statistical Computing**  
R Development Core Team  
  
<http://www.r-project.org/>
- 2) **R-help – Main R Mailing List**  
Community Contributed  
  
<https://www.stat.math.ethz.ch/mailman/listinfo/r-help>
- 3) **R-Cookbook.com Delicious Statistical Recipes**  
Creative Commons  
  
<http://www.r-cookbook.com/>
- 4) **Using R for Psychological Research**  
William Revelle, Northwestern University  
  
<http://www.personality-project.org/r/>
- 5) **R Graphical Manuals**  
Osamu Ogasawara and IMS Lab Inc. (designed by CMG Technologies)  
  
<http://cged.genes.nig.ac.jp/RGM2/index.php>
- 6) **R Tutorials**  
Kelley Black, Department of Mathematics, Union College  
  
<http://www.cyclismo.org/tutorial/R/>
- 7) **Animated Statistics Using R**  
Yihui Xie  
  
<http://r.yihui.name/intro/documentations/index.htm>
- 8) **The Omega Project for Statistical Computing**  
Omega Project and Doug Bates  
  
<http://www.omegahat.org/>
- 9) **R Help @ MC**  
Center for Mathematical Sciences, Lund University  
  
<http://www1.maths.lth.se/help/R/>

---

# Appendix D

## Useful “R” Functions and Syntax

Practicum workshop February 6, 2008

---

### R reference card, by Jonathan Baron

Parentheses are for functions, brackets are for indicating the position of items in a vector or matrix. (Here, items with numbers like x1 are user-supplied variables.)

### Miscellaneous

q(): quit  
<-: assign  
INSTALL package1: install package1  
m1[,2]: column 2 of matrix m1  
m1[,2:5] or m1[,c(2,3,4,5)]: columns 2–5  
m1\$a1: variable a1 in data frame m1  
NA: missing data  
is.na: true if data missing  
library(mva): load (e.g.) the mva package

### Help

help(command1): get help with command1 (NOTE: USE THIS FOR MORE DETAIL THAN THIS CARD CAN PROVIDE.)  
help.start(): start browser help  
help(package=mva): help with (e.g.) package mva  
apropos("topic1") and help.search("topic1"): commands relevant to topic1  
example(command1): examples of command1

### Input and output

source("file1"): run the commands in file1.  
read.table("file1"): read in data from file1  
data.entry(): spreadsheet  
scan(x1): read a vector x1  
download.file("url1"): from internet  
url.show("url1"), read.table.url("url1"): remote input  
sink("file1"): output to file1, until sink()  
write(object1, "file1"): writes object1 to file1  
write.table(dataframe1, "file1"): writes a table

### Managing variables and objects

attach(x1) detach(x1): put (remove) x1 in search path  
ls(): lists all the active objects.  
str(object1): print useful information about object1  
rm(object1): remove object1  
dim(matrix1): dimensions of matrix1  
dimnames(x1): names of dimensions of x1  
length(vector1): length of vector1  
1:3: the vector 1,2,3  
c(1,2,3): creates the same vector  
rep(x1,n1): repeats the vector x1 n1 times  
cbind(a1,b1,c1), rbind(a1,b1,c1): binds columns or rows into a matrix  
merge(df1,df2): merge data frames  
matrix(vector1,r1,c1): make vector1 into a matrix with r1 rows and c1 columns

data.frame(v1,v2): make a data frame from vectors v1 and v2  
as.factor(), as.matrix(), as.vector(): conversion  
is.factor(), is.matrix(), is.vector(): what it is  
t(): switch rows and columns  
which(x1==a1): returns indices of x1 where x1==a1

### Control flow

for (i1 in vector1): repeat what follows  
if (condition1) ...else ...: conditional

### Arithmetic

%\*%: matrix multiplication  
%/%, ^, %%, sqrt(): integer division, power, modulus, square root

### Statistics

max(), min(), mean(), median(), sum(), var(): as named  
summary(data.frame): prints statistics  
rank(), sort() rank and sort  
ave(x1,y1): averages of x1 grouped by factor y1  
by(): apply function to data frame by factor  
apply(x1,n1,function1): apply function1 (e.g. mean) to x by rows (n1=1) or columns (n2=2)  
tapply(x1,list1,function1): apply function to x1 by list1  
table(): make a table  
tabulate(): tabulate a vector

### basic statistical analysis

aov(), anova(), lm(), glm(): (generalized) linear models, anova  
t.test(): t test  
prop.test(), binom.test(): sign test  
chisq.test(x1): chi-square test on matrix x1  
fisher.test(): Fisher exact test  
cor(a): show correlations  
cor.test(a,b): test correlation  
friedman.test(): Friedman test

### some statistics in mva package

prcomp(): principal components  
kmeans(): kmeans cluster analysis  
factanal(): factor analysis  
cancor(): canonical correlation

### Graphics

plot(), barplot(), boxplot(), stem(), hist(): basic plots  
matplot(): matrix plot  
pairs(matrix): scatterplots  
coplot(): conditional plot  
stripplot(): strip plot  
qqplot(): quantile-quantile plot  
qqnorm(), qqline(): fit normal distribution