Introduction to Computing Using R

Sponsored by
The Fordham Council on
Applied Psychometrics

Professional Workshop Series
October 3, 2008

Michael Chajewski

Table of Content

FCAP Workshop

1) Introduction and organization …….. p. 3

2) The R environment ……..…….... p. 4

3) Importing data ……..…….. p. 8

4) Exporting objects …….. p. 10

5) Algebraic operations

6) Matrix operations

7) Complex operations

8) Writing functions …….. p. 12

9) Statistical applications

10) R graphics

11) Advanced topics / writing R extensions

12) Appendix A: The R environment …… p. 16

13) Appendix B: Useful free R manuals ……... p. 17

14) Appendix C: Additional R web resources ……..……... p. 21

15) Appendix D: Useful R functions and syntax ……..……...p. 22

16) Appendix E: R base package commands ……... p. 23

Introduction

FCAP Workshop

R is a predominately syntax driven computational software package and relies on its user to known the R language (which is derived from the S, S-plus and UNIX programming languages). R is comparable in structure and conceptual arrangement to other syntax based software packages such as SAS. The operator manipulates syntax in an editor (*editor*) and observes the requested operations in another console (*log and output*).

Much like some of the more known packages, R is developing its own target market. Whereas SPSS has established itself as a basic exploratory/introductory statistical software package for the social sciences and SAS has been adopted by the professional business and research sector, R is gaining recognition as one of the primary venues for newly derived statistics. Many leaders in the statistical, mathematical, social and regular sciences utilize R to create custom models (both mathematically and graphically) otherwise not available (and often possible) in other environments.

Above all, R has several advantages over other programming statistical packages which sets this unique interface apart from the rest. Some (but certainly not all) of these benefits are:

- It is absolutely free!
- Its system operation needs are negligible
- R is an open source code software
- It has one of the largest professional communities
- It is being updated with unprecedented regularity
- It is intuitive and easy to navigate
- Countless books, articles, documentation, guides etc. are freely available (*and many, many more…)*

This workbook was designed in such a way so as to expose the workshop participant to a wide spectrum and variety of R applications including computational, statistical and graphical operations. Each spread in the workbook consists of four columns. The first is the actual R console syntax submitted along with any possible called on output. The second column comments on the demonstrated operations. The third provides R environment created devices, plots, and underlying output and the fourth column is reserved for the workshop attendees' personal comments on the demonstrations.

The R environment

FCAP Workshop

R Console

```
R version 2.7.0 (2008-04-22)
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.

> license( )

> citation( )

> help( )
> ?
> help.start( )

> ls( )
> search( )
> apropos( )

> library( )
> data( )
> help.search( )
> attach( )
```

(See Appendix A for a diagram of the environment)

Comments

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 function.

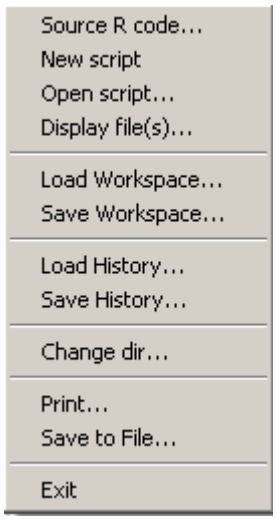
Giving credit where it is due is very important when using an open source software package. No one wants to be accused of plagiarism or intellectual theft.

The R environment provides a multitude of functions which allow easier navigation, self instruction and general exploration. Often specific knowledge is not needed in order to find and use a highly advanced operation (however interpretation is up to the user).

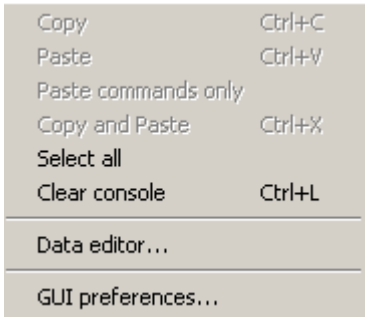
The R environment is not difficult to use. However, even advanced users can sometimes get bogged-down by the complexity of R’s environment and its subsequent navigation.

R Devices *(output)*

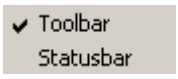
File:



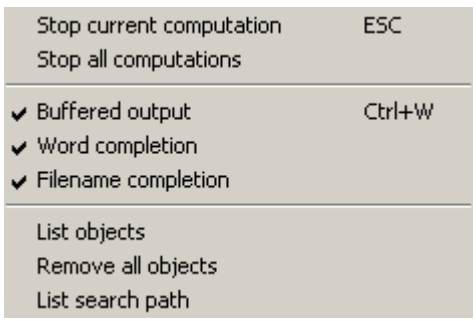
Edit



View



Misc

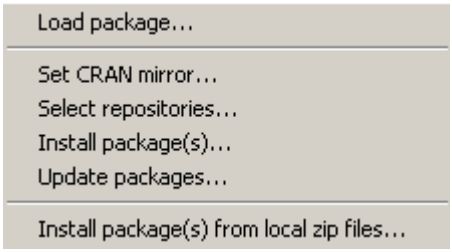


Notes

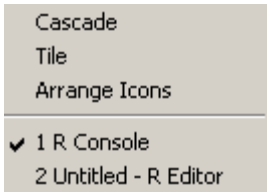
R Console	Comments
Click "Packages," then "Load package…" if the package is already installed or "Install package(s)…". Find "rgl", then click "OK". *or use the syntax* > demo( ) > install.packages("rgl") > library(rgl)	R packages if they have already been downloaded from a CRAN mirror site can be loaded 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).
(Download "rela" from http://www.chajewski.com)	Some unpublished R packages can be obtained privately as installable zip files.
> demo( )	
> demo(lollipop3d) > demo(bivar) > demo(glm.vr)	Some R packages will come with a demonstration of the included functions, plotting capabilities and statistical applications.
> example( )	Calls on a specific example from a package
> options( ) > windows( )	Two core global functions in R which allow you to manipulate devices.
> # Insert some comment here!	The pound sign is used to make non executable comments which will allow you to annotate your syntax.
> test <- 2 > test [1] 2 > test = 3 > test [1] 3 > ls( ) > rm(test) > test Error: object "test" not found	The entire functionality, workability and execution of R is completely founded in the manipulation of imported, created and/or derived objects within the environment.

R Devices *(output)*

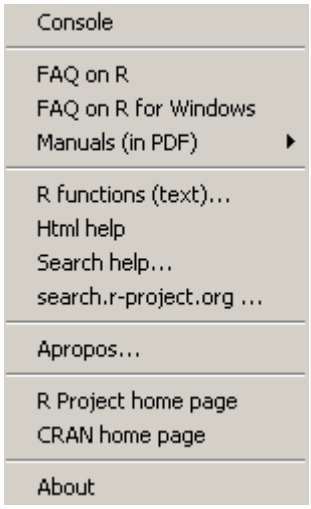
Packages:



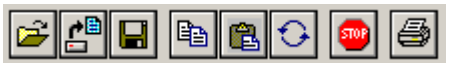
Windows



Help



Toolbar:



R Editor Toolbar:

Notes

Importing data

FCAP Workshop

R Console

```
> ?read.table
> ?scan

> bandflyer <- scan(file="E:/…/bandflyer.txt")
Read 52 items

> bandtxt <- read.table(file="E:/…/bandtxt.txt",
header=TRUE)

> bandcsv <- read.csv(file="E:/…/bandcsv.csv",
header=TRUE)
```

Click "Packages," then "Load package…" and select "foreign" Click "OK".

```
> ?read.spss
> ?read.ssd
> ?read.xport

> bandspss <- read.spss(file="E:/…/bandsav.sav",
to.data.frame=TRUE)

> bandspss2 <- read.spss(file="E:/…/bandsav.sav")

> bandspss2$FLYERS
[1] 129 142 127 140 […]

> attach(bandspss2)
> FLYER
[1] 129 142 127 140 […]

> mode(bandspss2)
> mode(bandspss)
> mode(bandtxt)
> mode(bandcsv)
[1] "list"

> ?vector
> ?array
> ?matrix
> ?data.frame
> bandtxt <- as.data.frame(bandtxt)
> mode(bandtxt)
[1] "list"
> bandtxt <- as.matrix(bandtxt)
> mode(bandtxt)
[1] "numeric"
```

Comments

Reading in data in R is very easy. As long as a uniform path is specified and the data are all deposited into that catalog, then importation and exportation are simple statements.

Virtually any format dataset/database can be imported in to R.

Particular attention should be given to the importation on non-standard data types because the translation may not be into a matrix like data frame.

Datasets and variables can be released into the environment for continuous use.

The function "mode" does not produce the measure of central tendency known as the mode. It checks for the storage mode of the dataset or variable.

Usually data need to be coerced into the format necessary for any one particular analysis.

R Devices *(output)*

	sales	performances	web	flyers
1	893.93	5	292	129
2	1091.3	5	252	142
3	1230.0	5	267	127
4	1045.8	5	379	140
5	997.24	5	318	146
6	1495.1	6	393	147
7	1200.6	5	331	143
8	747.24	4	204	111
9	866.43	5	266	115
10	603	5	253	152
11	343.52	5	315	96
12	472.1	6	271	116
13	171.79	4	166	81
14	135.79	4	204	123
15	925.95	5	335	131
16	1574	5	352	123
17	1405.3	5	274	152
18	971.27	4	333	134
19	1165.2	5	302	109
20	597.85	4	324	93
21	490.34	4	327	103
22	709.59	5	206	90
23	987.3	5	310	124
24	954.6	6	306	163
25	1216.9	6	350	141
26	1491.5	5	275	156
27	668.3	4	173	94
28	915.03	5	360	117
29	565.92	4	340	130

```
$SALES
 [1]  893.92833487 1091.26
 [6] 1495.13552886 1200.56
[11]  343.52127916  472.09
[16] 1574.00922579 1405.32
[21]  490.33983478  709.58
[26] 1491.51979533  668.30
[31]  930.23745371  379.37
[36]  722.27987391 1337.43
[41]  767.63724975 1020.02
[46] 1344.91018061 1361.77
[51]  803.16475939 1447.45

$PERFORMA
 [1] 5 5 5 5 5 6 5 4 5 5 5
[38] 4 6 5 5 5 5 6 4 5 5 6

$WEB
 [1] 292 252 267 379 318 39
[19] 302 324 327 206 310 30
[37] 322 231 368 357 260 29

$FLYERS
 [1] 129 142 127 140 146 14
[19] 109  93 103  90 124 16
[37] 108 136 172 130 106 10

attr(,"label.table")
attr(,"label.table")$SALES
NULL
```

Notes

Exporting objects

FCAP Workshop

R Console

```
> write.table(FLYERS, file="E:/…/flyers.txt")

> write.csv(bandsav, file="banddata.csv")
```

Saving out tables (as text files) or comma delimited datasets is the easiest way to export data out of R.

```
> save(bandtxt, file="bandtxt")
> rm(bandtxt)
> bandtxt
Error: object "bandtxt" not found

> load(file="bandtxt")
> bandtxt
     SALES          PERFORMA   WEB   FLYERS
1   893.92833487       5          292       129
2  1091.26961644       5          252       142  […]
```

When work continues on a project it is advised to save out R objects. They are easier imported and no transcription errors should occur.

```
> save(bandtxt, file="cheese")
> rm(bandtxt)
> bandtxt
Error: object "bandtxt" not found

> load(file="cheese")
> cheese
Error: object "cheese" not found
> bandtxt
     SALES          PERFORMA   WEB   FLYERS
1   893.92833487       5          292       129
2  1091.26961644       5          252       142  […]
```

Note that even the objects assigned name is saved out to the R file and therefore the file called should always match the object.

(dput( ) and dget( ) are the ASCI forms)

ASCI alternatived to the save and laod functions.

```
> itemlist <- c("bandtxt", "FLYERS", …)
> dump(itemlist, file="Band.R")
```

Created an R syntax file, readable as txt, containing the syntax on how to recreate all items which were contained in the list.

```
> capture.output(bandspss2, file="output.txt")
```

Captures all of your output as an arranged list viewable as a text file, just as seen in the R console.

R Devices *(output)*


flyers.txt


banddata.csv


bandtxt


cheese


Band.R


output.txt

Notes

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 enetred 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 preliminary 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.

```
> ?linear.model
No documentation for 'linear.model'  [...]
> help.search("linear model")
> ? lm
> lm
function (formula, data, subset, weights,
na.action, [...]
```

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

```
> 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)
```

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

```
> 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)
```

Functions which come from "R" packages usually have built in graphic, analtytic 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 Avaiable, 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 reveresing 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 seperately 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. Infividual 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.

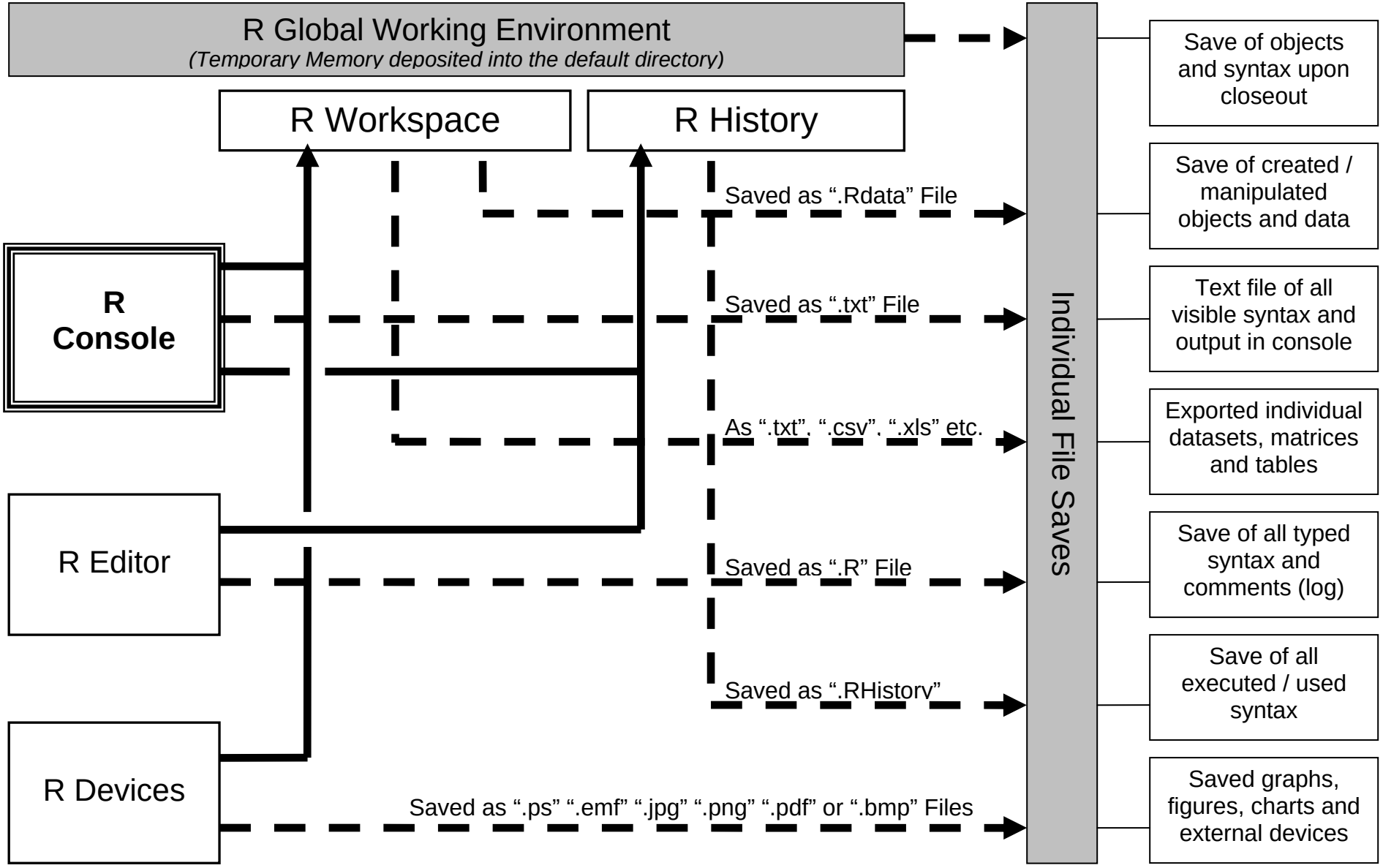
```
> length(up.fix)
[1] 84
> fix.data <- matrix (up.fix, 12, 7)
> fix.data
          [,1]          [,2]
 [1,] 0.0000000   0.8709916
 [2,] 0.8175705   0.8191014  [...]
> fix(fix.data) # Assign some variable names
```

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

Last but not least your shouold 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 menue, or by specifying a specific path to your exported files, "R" wil lsave everything to the defualt "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 knowledgable but also exceptionally helkpful. Often you will get new and imporved 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
FCAP FCAP Workshop



Appendix B
Useful Free R Manuals

FCAP FCAP Workshop

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 rcompgen 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/Splusdiscrete2.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

FCAP FCAP Workshop

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) **Quick R**
Guide to using R for SAS and SPSS users by Robert I. Kabacoff
http://www.statmethods.net/

4) **R-Cookbook.com Delicious Statistical Recipes**
Creative Commons
http://www.r-cookbook.com/

5) **Using R for Psychological Research**
William Revelle, Northwestern University
http://www.personality-project.org/r/

6) **R Graphical Manuals**
Osamu Ogasawara and IMS Lab Inc. (designed by CMG Technologies)
http://cged.genes.nig.ac.jp/RGM2/index.php

7) **R Tutorials**
Kelley Black, Department of Mathematics, Union College
http://www.cyclismo.org/tutorial/R/

8) **Rweb: Statistical Analysis on the Web**
R training site created by Jeff Banfield (University of Montana)
http://www.math.montana.edu/Rweb/

9) **Animated Statistics Using R**
Yihui Xie
http://r.yihui.name/intro/documentations/index.htm

10) **The Omega Project for Statistical Computing**
Omega Project and Doug Bates
http://www.omegahat.org/

11) **R Help @ MC**
Center for Mathematical Sciences, Lund University
http://www1.maths.lth.se/help/R/

Appendix D
Useful R Functions and Syntax

FCAP FCAP Workshop

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

Appendix E
R Base Package Commands

FCAP FCAP Workshop

%%	%/%	*	+	-
/	^	!=	<	<=
==	>	>=	$	[
[[	~	?	!	&
&&	|	||	abbreviate	abline
abs	acos	acosh	all	all.names
all.vars	anova	anova.glm	anova.lm	any
aperm	append	apply	approx	approxfun
apropos	Arg	args	Arithmetic	array
arrows	as.array	as.call	as.character	as.complex
as.data.frame	as.double	as.expression	as.factor	asin
asinh	as.integer	as.list	as.logical	as.matrix
as.na	as.name	as.null	as.numeric	as.ordered
as.qr	as.real	assign	as.ts	as.vector
atan	atan2	atanh	attach	attr
attributes	autoload	.AutoloadEnv	axis	backsolve
barplot	beta	binomial	box	boxplot
boxplot.stats	break	browser	bw.bcv	bw.sj
bw.ucv	bxp	c	.C	call
cat	cbind	ceiling	character	charmatch
chisq.test	chol	chol2inv	choose	class
class<-	codes	coef	coefficients	coefficients.glm
coefficients.lm	co.intervals	col	colnames	colors
colours	Comparison	complete.cases	complex	Conj
contour	contrasts	contr.helmert	contr.poly	contr.sum
contr.treatment	convolve	cooks.distance	coplot	cor
cos	cosh	count.fields	cov	covratio
crossprod	cummax	cummin	cumprod	cumsum
curve	cut	D	data	data.class
data.entry	dataentry	data.frame	data.matrix	dbeta
dbinom	dcauchy	dchisq	de	debug
delay	demo	de.ncols	density	deparse
de.restore	deriv	deriv.default	deriv.formula	de.setup

detach	deviance	deviance.glm	deviance.lm	device
Devices	dev.off	dexp	df	dfbetas
dffits	df.residual	df.residual.glm	df.residual.lm	dgamma
dgeom	dget	dhyper	diag	diff
digamma	dim	dim<-	dimnames	dimnames<-
dlnorm	dlogis	dnbinom	dnchisq	dnorm
do.call	dotplot	double	dpois	dput
drop	dt	dump	dunif	duplicated
dweibull	dyn.load	edit	effects.glm	effects.lm
eigen	else	emacs	end	environment
environment<-	eval	exists	exp	expression
Extract	factor	family	family.glm	fft
finite	fitted	fitted.values	fitted.values.glm	fitted.values.lm
fivenum	fix	floor	for	formals
format	formatC	format.default	formula.default	formula.formula
formula.terms	.Fortran	frame	frequency	function
Gamma	gamma	gaussian	gc	gcinfo
get	getenv	gl	glm	glm.control
glm.fit	.GlobalEnv	graphics.off	gray	grep
grid	gsub	hat	heat.colors	help
hist	hsv	identify	if	ifelse
Im	image	%in%	influence.measures	inherits
integer	interactive	.Internal	inverse.gaussian	invisible
invisible	IQR	is.array	is.atomic	is.call
is.character	is.complex	is.data.frame	is.double	is.environment
is.expression	is.factor	is.function	is.integer	is.language
is.list	is.loaded	is.logical	is.matrix	is.na
is.name	is.null	is.numeric	is.ordered	is.qr
is.real	is.recursive	is.single	is.ts	is.unordered
is.vector	lapply	lbeta	lchoose	legend
length	LETTERS	letters	levels	levels<-
lgamma	.lib.loc	.Library	library	library.dynam
license	lines	lines.default	list	lm
lm.fit	lm.influence	lm.wfit	load	locator
log	log10	log2	Logic	logical
lower.tri	lowess	ls	ls.diag	lsfit
lsf.str	ls.print	ls.str	.Machine	Machine

machine	macintosh	mad	match	match.arg
match.call	matlines	mat.or.vec	matplot	matpoints
matrix	max	mean	median	menu
methods	min	missing	Mod	mode
mode<-	model.frame	model.frame.default	model.matrix	model.matrix.default
month.abb	month.name	mtext	mvfft	NA
na.action	na.action.default	na.fail	names	na.omit
nargs	nchar	NCOL	ncol	next
NextMethod	nextn	nlevels	nlm	[.noquote
noquote	NROW	nrow	NULL	numeric
objects	on.exit	optimize	options	order
ordered	outer	pairs	palette	par
parse	paste	pbeta	pbinom	pcauchy
pchisq	pentagamma	pexp	pf	pgamma
pgeom	phyper	pi	pictex	piechart
plnorm	plogis	plot	plot.default	plot.density
plot.ts	plot.xy	pmatch	pmax	pmin
pnbinom	pnchisq	pnorm	points	points.default
poisson	polygon	polyroot	postscript	ppoints
ppois	pretty	print	print.anova.glm	print.anova.lm
print.data.frame	print.default	print.density	print.formula	print.glm
print.lm	print.noquote	print.plot	print.summary.glm	print.summary.lm
print.terms	print.ts	proc.time	prod	prompt
prompt.default	prop.test	provide	.Provided	ps.options
pt	punif	pweibull	q	qbeta
qbinom	qcauchy	qchisq	qexp	qf
qgamma	qgeom	qhyper	qlnorm	qlogis
qnbinom	qnchisq	qnorm	qpois	qqline
qqnorm	qqplot	qr	qr.coef	qr.fitted
qr.Q	qr.qty	qr.qy	qr.R	qr.resid
qr.solve	qr.X	qt	quantile	quasi
quit	qunif	quote	qweibull	rainbow
.Random.seed	range	rank	rbeta	rbind
rbinom	rcauchy	rchisq	Re	readline
read.table	real	rect	remove	rep
repeat	replace	require	resid	residuals
residuals.glm	residuals.lm	return	rev	rexp

rf	rgamma	rgb	rgeom	rhyper
RLIBS	rlnorm	rlogis	rm	rnbinom
rnchisq	rnorm	round	row	row.names
rownames	rpois	rstudent	rt	runif
rweibull	sample	sapply	save	save.plot
scale	scan	sd	segments	seq
sequence	sign	signif	sin	sinh
sink	solve	solve.qr	sort	source
spline	splinefun	split	sqrt	start
stem	stop	storage.mode	storage.mode<-	str
str.data.frame	str.default	strheight	stripplot	strsplit
structure	strwidth	sub	Subscript	substitute
substr	substring	sum	summary	summary.glm
summary.lm	svd	sweep	switch	symbol.C
symbol.For	symnum	sys.call	sys.calls	sys.frame
sys.frames	sys.function	sys.nframe	sys.on.exit	sys.parent
sys.parents	system	system.date	system.time	t
table	tabulate	tan	tanh	tapply
tempfile	terms	terms.default	terms.formula	terms.terms
terrain.colors	tetragamma	text	time	title
topo.colors	trace	traceback	trigamma	trunc
ts	tsp	t.test	typeof	unclass
undebug	unique	uniroot	unlink	unlist
untrace	update	update.formula	update.glm	update.lm
upper.tri	UseMethod	var	vector	Version
version	vi	warning	weighted.mean	weights.lm
while	window	windows	write	x11
xedit	xemacs	xinch	xor	xy.coords
yinch				

Notes

Fordham Council on Applied Psychometrics
Professional Workshop Series

Graduate School of Arts and Sciences

Psychology Department / Psychometrics Program

2008