A

Appendix: R: Getting started

Note: This is (obviously) currently a stub. A more complete reference for psychologists on which this appendix is based is at http://personality-project.org/r

There are many possible statistical programs that can be used in psychological research. They differ in multiple ways, at least some of which are ease of use, generality, and cost. Some of the more common packages used are Systat, SPSS, and SAS. These programs have GUIs (Graphical User Interfaces) that are relatively easy to use but that are unique to each package. These programs are also very expensive and limited in what they can do. Although convenient to use, GUI based operations are difficult to discuss in written form. When teaching statistics or communicating results to others, it is helpful to use examples that may be used by others using different computing environments and perhaps using different software. This book as well as other introductions describes an alternative approach that is widely used by practicing statisticians, the statistical environment R. This appendix is not meant as a user's guide to R, but merely the first step in using R for psychometrics in particular and psychological research in general.

Throughout the text, examples of analyses are given in R. But what is R and how to get it to work on your computer is perhaps the first question the reader faces.

A.1 R:A statistical programming environment

The R project, based upon the S and S+ stats programs, has developed an extremely powerful set of"packages" that operate within one program. Although described as merely "an effective data handling and storage facility [with] a suite of operators for calculations on arrays, in particular, matrices" R is, in fact, a very useful interactive package for data analysis. When compared to most other stats packages used by psychologists, R has at least three compelling advantages: it is free, it runs on multiple platforms (e.g., Windows, Unix, Linux, and Mac OS X and Classic), and combines many of the most useful statistical programs into one quasi integrated program. (R is free software as part of the GNU Project. That is, users are free to use, modify, and distribute the program, within the limits of the GNU non-license). The program itself and detailed installation instructions for Linux, Unix, Windows, and Macs are available through CRAN (Comprehensive R Archive Network) at `http://www.r-project.org`.

Although many run R as a language and text oriented programming environment, there are GUIs available for PCs, Linux and Macs. See for example, R Commander by John Fox. Stefano Iacus and Simon Urbanek have done wonders in converting R to the Mac, and there is now a semi-GUI available as a Mac release of R 2.4.1. Compared to the basic PC environment, the Mac GUI is to be preferred.

(A note on the numbering system: The R-development core team releases an updated version of R about every six months. That is, the current version of 2.4.1 will be replaced with 2.5.0 sometime in the spring of 2007. Bug fixes are then added with a sub version number (e.g. 2.4.1 fixed minor problems with 2.4.0).

R is an integrated, interactive package for data manipulation and analysis that includes functions for standard descriptive statistics (means, variances, ranges) and also includes useful tools for Exploratory Data Analysis. In terms of inferential statistics it has many varieties of the General Linear Model including the conventional special cases such as Analysis of Variance and MANOVA. Advanced features include correlational packages for multivariate analyses including Factor and Principal Components Analysis, and cluster analysis. Advanced multivariate analyses packages that have been contributed to the R project include Structural Equation Modeling, Hierarchical Linear Modeling (referred to as non linear mixed effects) and taxometric analysis. All of these are available in the free "packages" distributed by the R group. Statisticians and statistically minded people around the world have contributed packages to the R Group and maintain a very active news group offering suggestions and help. In addition to be a package of routines, R is a interpreted programming language that allows one to create specific functions when needed.

R is also an amazing program for producing statistical graphics. A collection of some of the best graphics is available at addictedtoR `http://addictedtor.free.fr/graphiques/` with a complete gallery of thumbnail of figures.

A.2 General comments

R is not overly user friendly (at first). Its error messages are at best cryptic. It is, however, very powerful and once partially mastered, easy to use. As additional modules are added, it becomes even more useful. Modules included allow for multilevel (hierarchical) linear modeling, confirmatory factor analysis, etc.

Commands are entered into the "Console" window. You can add a comment to any line by using a #. The Mac version has a text editor window that allows you to write, edit and save your commands. Alternatively, if you use a normal text editor (As a Mac user, I use BBEDIT, PC users can use Notepad), you can write out the commands you want to run, comment them so that you can remember what they do the next time you run a similar analysis, and then copy and paste into the R console. The R code throughout this text is meant to be copied and pasted into R.

Although being syntax driven seems a throwback to an old, pre Graphical User Interface type command structure, it is very powerful for doing production statistics. Once you get a particular set of commands to work on one data file, you can change the name of the data file and run the entire sequence again on the new data set. This is is also very helpful when doing professional graphics for papers. In addition, for teaching, it is possible to prepare a web page of instructional commands that students can then cut and paste into R to see for themselves how things work. That is what may be done with the instructions throughout this book. It is also possible to write text in LaTeX with embedded R commands. Then executing the *Sweave* function on that text file will add the R output to the LaTeX file. This almost magical feature allows rapid integration of content with statistical techniques. More importantly, it allows for "reproducible research" in that the actual data files and instructions may be specified for all to see.

A.3 Using R in 8 simple steps

(These steps are not meant to limit what can be done with R, but merely to describe how to do the analysis for the most basic of research projects and to give a first experience with R).

1. Install R on your computer or go to a machine that has it.
2. Enter your data using a text editor and save as a text file (perhaps comma delimited if using Excel)
3. Start R, read the data file or paste from the clipboard.
4. Find basic descriptive statistics (e.g., means, standard deviations, minimum and maxima)
5. Prepare a simple descriptive graph (e.g, a box plot) of your variables

6. Find the correlation matrix to give an overview of relationships
7. If you have an experimental variable, do the appropriate multiple regression using standardized scores.
8. Graph the results

A.4 Getting started

A.4.1 Installing R on your computer

Although it is possible that your local computer lab already has R, it is useful to do analyses on your own machine. In this case you will need to download the R program from the R project and install it yourself. Go to the R home page at `http://www.r-project.org` and then choose the Download from CRAN (Comprehensive R Archive Network) option. This will take you to list of mirror sites around the world. You may download the Windows, Linux, or Mac versions at this site.

A.4.2 Useful additions to R

One of the advantages of R is that it can be supplemented with additional programs that are included as "packages" using the *package manager.* (e.g., *sem* does structural equation modeling) or that can be added using the "source" command. Most packages are directly available through the CRAN repository. Others are available at the BioConductor repository. Yet others are available "other" repositories. As of this writing, the "psych" package may be downloaded from the personality-project.org/r repository.

- (For Macs)
  - From the "Packages and Data" menu, select "Package Installer"
  - Select "other repository" and enter the url http://personality-project.org/r
  - Uncheck the binary format box
  - A window will report your installed version and what is available at the repository. Select *psych*.
  - "Install selected" (and wait while it does so).
- (For PCs only)
  - Find the psych.zip file at: `http://www.personality-project.org/r/src/contrib` and download it ("psych.zip") to your PC.
  - Go to the "Packages" menu in RGui, and select, "Install package(s) from local zip files".
  - Select the psych.zip file.
  - R shoud now add to the R console:
    > utils:::menuInstallLocal() updating HTML package descriptions
- (Macs and PCs) For this, or any other package to work, you must activate it by either using the Package Manager or the "library" command:
  - e.g., library(psych) or library(sem)
  - If loading the psych package works, function such as "describe" and "pairs.panels" should work (or at least give an error message that is NOT "could not find function").
  - entering ?psych will give a list of the functions available in the psych package as well as an overview of their funtionality.
  - objects("package:psych") will list the functions available in a package (in this case, psych).

R is case sensitive and does not give overly useful diagnostic messages. If you get an error message, don't be flustered but rather be patient and try the command again using the correct spelling for the command.

A.4.3 Help and Guidance

When in doubt, use the help() function. This is identical to the ? function where function is what you want to know about. e.g.,

?read.table #ask for help in using the read.table function – see the answer in its own window, or
help(read.table) #another way of asking for help. - see the window

A.5 Basic R commands and syntax

At the abstract level, almost all operations in R consists of executing a function on an object. The result is a new object. This very simple idea allows the output of any operation to be operated on by another function.

Command syntax tends to be of the form:
variable = function (parameters) or
variable <- function (parameters)
The = and the <- symbol imply replacement, not equality. The preferred style is to use the <- symbol to avoid confusion. Elements of arrays are indicated within brackets [ ]. Thus, the 4th row, 5th column of a matrix X is X[4,5].

The result of an operation will not necessarily appear unless you ask for it. The command
m <- mean(x)
will find the mean of x but will not print anything on the console with the additional request m

A suggestion about programming style that is not strictly R related: It is useful to label variables with names that will make sense to you later when you look at an analysis several months later. Thus, rather than calling variables x, y, and z, giving names that reflect that they are in fact impulsivity, anxiety, and performance tends to be more useful.

For a more complete list of R commands, see Appendix B. A limited number of examples are shown below. Examples of using R to do simple matrix operations are discussed in Appendix D.

A.6 Entering or getting the data

For most data analysis, rather than manually enter the data into R, it is probably more convenient to use a spreadsheet (e.g., Excel or OpenOffice) as a data editor, save as a tab or comma delimited file, and then read the data. Most of the examples in this tutorial assume that the data have been entered this way. Many of the examples in the help menus have small data sets entered using the c() command or created on the fly. It is also possible to read data in from a remote file server.

Using the copy.clipboard() function from the psych package, it is also possible to have a data file open in a text editor or spreadsheet program, copy the relevant lines to the clipboard, and then read the clipboard directly into R.

For the first example, we read data from a remote file server for several hundred subjects on 13 personality scales (5 from the Eysenck Personality Inventory (EPI), 5 from a Big Five Inventory (BFI) , 1 Beck Depression, and two anxiety scales). The data are taken from a study in the Personality, Motivation, and Cognition Laboratory at Northwestern University. The file is structured normally, i.e. rows represent different subjects, columns different variables, and the first row gives subject labels. Had we saved this file as comma delimited, we would add the separation (sep=",") parameter.

To read a file from your local machine, change the datafilename to specify the path to the data. Using the *file.choose* command, you can set a local file name to the data file name anywhere on your computer.

```
#specify the name and address of the remote file
>datafilename <- "http://personality-project.org/r/datasets/maps.mixx.epi.bfi.data"
#Or, If I want to read a datafile from my desktop
#datafilename <- file.choose()  #where you dynamically can go find the file

#now read the data file
>person.data <-  read.table(datafilename,header=TRUE)  #read the data file

>names(person.data)  #list the names of the variables

> names(person.data)  #list the names of the variables
 [1] "epiE"     "epiS"     "epiImp"   "epilie"   "epiNeur"  "bfagree"  "bfcon"
 [8] "bfext"    "bfneur"   "bfopen"   "bdi"      "traitanx" "stateanx"
```

The data are now in the data.frame "person.data". Data.frames allow one to have columns that are either numeric or alphanumeric. They are conceptually a generalization of a matrix in that they have rows and columns, but unlike a matrix, the columns can be of different "types" (integers, reals, characters, strings).

A.7 Basic descriptive statistics

Basic descriptive statistics are most easily reported by using the summary, means and Standard Deviations (sd) commands. Using the describe function available in the supplement is also convenient. Graphical displays that also capture this are available as a boxplot.

```
> summary(person.data)  #print out the min, max, range, mean, median, etc. of the data
> round(mean(person.data),2)   #means of all variables, rounded to 2 decimals
> round(sd(person.data),2)      #standard deviations, rounded to 2 decimals
```

```
      epiE            epiS            epiImp          epilie         epiNeur
 Min.   : 1.00   Min.   : 0.000   Min.   :0.000   Min.   :0.000   Min.   : 0.00
 1st Qu.:11.00   1st Qu.: 6.000   1st Qu.:3.000   1st Qu.:1.000   1st Qu.: 7.00
 Median :14.00   Median : 8.000   Median :4.000   Median :2.000   Median :10.00
 Mean   :13.33   Mean   : 7.584   Mean   :4.368   Mean   :2.377   Mean   :10.41
 3rd Qu.:16.00   3rd Qu.: 9.500   3rd Qu.:6.000   3rd Qu.:3.000   3rd Qu.:14.00
 Max.   :22.00   Max.   :13.000   Max.   :9.000   Max.   :7.000   Max.   :23.00
    bfagree          bfcon           bfext           bfneur          bfopen
 Min.   : 74.0   Min.   : 53.0   Min.   :  8.0   Min.   : 34.00   Min.   : 73.0
 1st Qu.:112.0   1st Qu.: 99.0   1st Qu.: 87.5   1st Qu.: 70.00   1st Qu.:110.0
 Median :126.0   Median :114.0   Median :104.0   Median : 90.00   Median :125.0
 Mean   :125.0   Mean   :113.3   Mean   :102.2   Mean   : 87.97   Mean   :123.4
 3rd Qu.:136.5   3rd Qu.:128.5   3rd Qu.:118.0   3rd Qu.:104.00   3rd Qu.:136.5
 Max.   :167.0   Max.   :178.0   Max.   :168.0   Max.   :152.00   Max.   :173.0
      bdi            traitanx        stateanx
 Min.   : 0.000   Min.   :22.00   Min.   :21.00
 1st Qu.: 3.000   1st Qu.:32.00   1st Qu.:32.00
 Median : 6.000   Median :38.00   Median :38.00
 Mean   : 6.779   Mean   :39.01   Mean   :39.85
 3rd Qu.: 9.000   3rd Qu.:44.00   3rd Qu.:46.50
 Max.   :27.000   Max.   :71.00   Max.   :79.00
```

```
    epiE     epiS   epiImp   epilie  epiNeur  bfagree    bfcon    bfext   bfneur
   13.33     7.58     4.37     2.38    10.41   125.00   113.25   102.18    87.97
  bfopen      bdi traitanx stateanx
  123.43     6.78    39.01    39.85

    epiE     epiS   epiImp   epilie  epiNeur  bfagree    bfcon    bfext   bfneur
    4.14     2.69     1.88     1.50     4.90    18.14    21.88    26.45    23.34
  bfopen      bdi traitanx stateanx
   20.51     5.78     9.52    11.48
```

A.7.1 Using functions in the psych package

The psych package has been developed particularly for simple psychometrics and exploratory data of psychological data. It may be downloaded using the package installer from the `http:personality-project.org/r` personality project.

Once downloaded and installed, it needs to be loaded before it can be used. The library command does this.

Among the functions within the psych package are describe and pairs.panels.

```
> library(psych)
> describe(person.data)
```

```
          var   n   mean    sd median min max range   se
epiE        1 231  13.33  4.14     14   1  22    21 0.27
epiS        2 231   7.58  2.69      8   0  13    13 0.18
epiImp      3 231   4.37  1.88      4   0   9     9 0.12
epilie      4 231   2.38  1.50      2   0   7     7 0.10
epiNeur     5 231  10.41  4.90     10   0  23    23 0.32
bfagree     6 231 125.00 18.14    126  74 167    93 1.19
bfcon       7 231 113.25 21.88    114  53 178   125 1.44
bfext       8 231 102.18 26.45    104   8 168   160 1.74
bfneur      9 231  87.97 23.34     90  34 152   118 1.54
bfopen     10 231 123.43 20.51    125  73 173   100 1.35
bdi        11 231   6.78  5.78      6   0  27    27 0.38
traitanx   12 231  39.01  9.52     38  22  71    49 0.63
stateanx   13 231  39.85 11.48     38  21  79    58 0.76
```

The *describe* function can be combined with the *by* function to provide even more detailed tables. This example reports descriptive statistics for subjects with lie scores < 3 and those $>= 3$. The second element in the by command could be a categorical variable (e.g., sex).

```
 by(person.data,epilie<3,describe)
```

```
epilie < 3: FALSE

          var  n   mean    sd median min max range   se
epiE        1 90  12.64  4.00   13.0   1  21    20 0.42
epiS        2 90   7.61  2.81    8.0   0  13    13 0.30
epiImp      3 90   3.97  1.67    4.0   1   8     7 0.18
epilie      4 90   3.89  1.08    4.0   3   7     4 0.11
epiNeur     5 90   9.33  5.20    9.0   0  20    20 0.55
bfagree     6 90 128.12 16.55  129.0  87 167    80 1.74
```

```
bfcon      7 90 117.56 20.46  118.0  58 178   120 2.16
bfext      8 90 100.88 25.24  101.0  24 151   127 2.66
bfneur     9 90  82.22 22.80   81.5  35 144   109 2.40
bfopen    10 90 121.97 20.55  121.0  75 172    97 2.17
bdi       11 90   5.77  4.71    5.0   0  24    24 0.50
traitanx  12 90  37.01  9.06   36.0  22  71    49 0.95
stateanx  13 90  38.41 11.36   36.5  21  69    48 1.20
------------------------------------------------------------
epilie < 3: TRUE
         var   n   mean    sd median min max range   se
epiE       1 141  13.77  4.17     14   4  22    18 0.35
epiS       2 141   7.57  2.62      8   1  13    12 0.22
epiImp     3 141   4.62  1.97      5   0   9     9 0.17
epilie     4 141   1.41  0.73      2   0   2     2 0.06
epiNeur    5 141  11.10  4.59     10   0  23    23 0.39
bfagree    6 141 123.00 18.88    124  74 165    91 1.59
bfcon      7 141 110.50 22.38    111  53 176   123 1.88
bfext      8 141 103.01 27.25    105   8 168   160 2.29
bfneur     9 141  91.64 23.01     94  34 152   118 1.94
bfopen    10 141 124.36 20.50    126  73 173   100 1.73
bdi       11 141   7.43  6.29      6   0  27    27 0.53
traitanx  12 141  40.28  9.62     39  23  71    48 0.81
stateanx  13 141  40.77 11.51     39  23  79    56 0.97
```

A.8 Simple Graphics

There are a variety of ways of graphically reporting the data. One is the box plot to show the Tukey 5 numbers (upper and lower hinges, upper and lower quartiles, median). Another way to grasp the distribution of the data is to overlay the actual data points with a stripchart.

```
boxplot(person.data[,1:5])
stripchart(person.data[,1:5],vertical=T,add=T,method="jitter",jitter=.2)  #add in the points
```

Another way of describing the data is to graph them. *boxplot* show the top and bottom quartiles, medians, and the "hinges". histograms show the distribution in more detail. The pairs.panels command draws a matrix of scatter plots. (Note that just the first five variables are shown in the SPLOM to make it more readable).

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

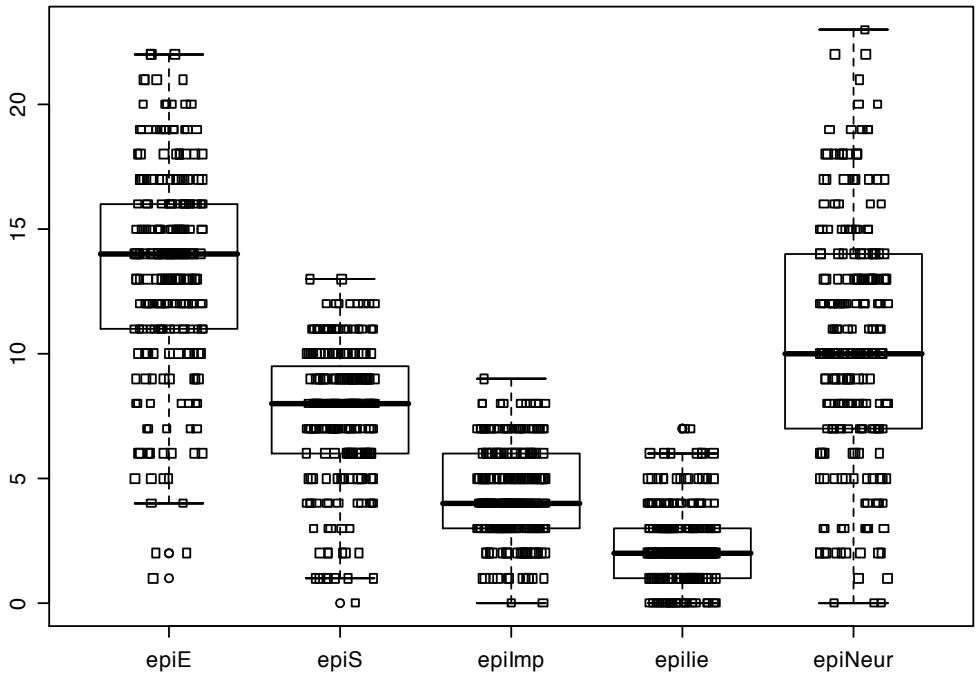

Fig. A.1. A boxplot with an added stripchart summarizes basic distributional properties of the data.

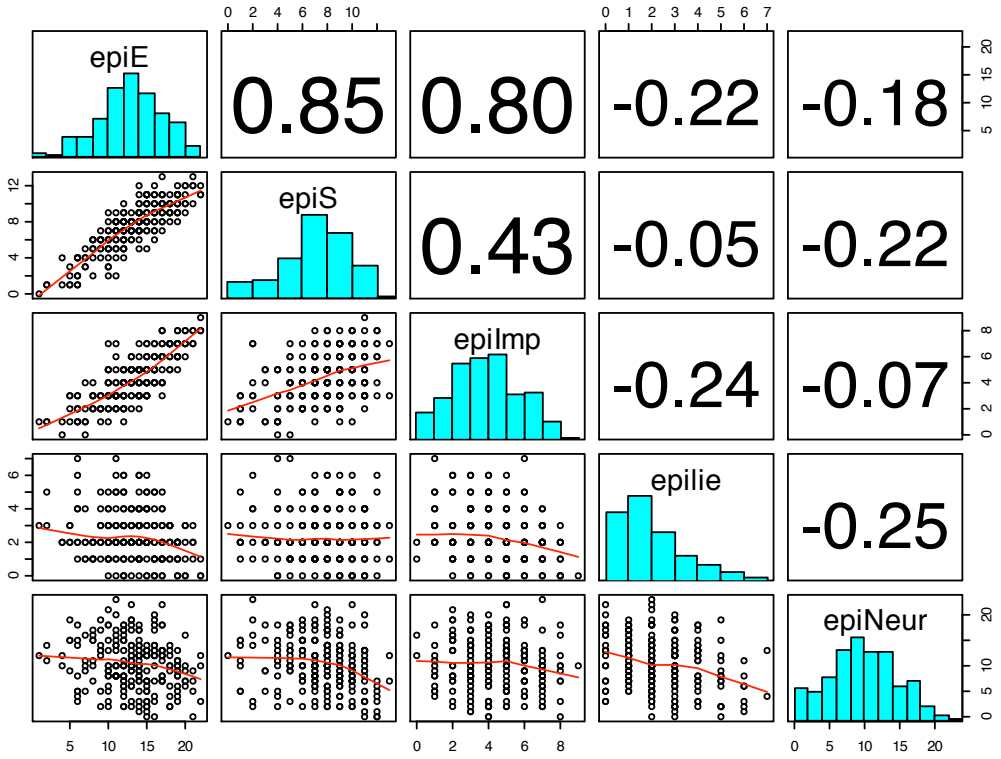

Fig. A.2. A scatter plot matrix of the data can be modified to give histograms as well as the correlations.