Appendix B
R commands

(Very rough summary of the most useful command)

B.1 Input and display

#read files with labels in first row
read.table(filename,header=TRUE) #read a tab or space delimited file
read.table(filename,header=TRUE,sep=',') #read csv files (comma separated)

x=c(1,2,4,8,16) #create a data vector with specified elements
y=c(1:8,1:4) #create a data vector with 12 entries
matr=rbind(1:8,1:4) #create two rows in a 2 * 8 matrix
matc=cbind(1:8,1:4) #create two columns in a 8 * 2 matrix
n=10
x1=c(rnorm(n)) #create a n item vector of random normal deviates
y1=c(runif(n))+n #create another n item vector that has n added to each random uniform distribution
z=rbinom(n,size,prob) #create n samples of size "size" with probability prob from the binomial item
sample(x, size, replace = FALSE, prob = NULL) #take a sample (with or without replacement) of size from x

vect=c(x,y) #combine them into one vector of length 2n
mat=cbind(x,y) #combine them into a n x 2 matrix (column wise)
mat[4,2] #display the 4th row and the 2nd column
mat[3,] #display the 3rd row
mat[,] #display the 2nd column
mat=cbind(rep(1:4,2),rep(4:1,2)) #create a 8 * 2 matrix with repeating elements
subset(data,logical) #those objects meeting a logical criterion
subset(data,df,select=variables,logical) #get those objects from a data frame that meet a criterion
B.2 moving around

ls()  #list the variables in the workspace
rm(x)  #remove x from the workspace
rm(list=ls())  #remove all the variables from the workspace
attach(mat)  #make the names of the variables in the matrix available
detach(mat)  #releases the names
new=old[-n]  #drop the nth column
new=old[n]  #drop the nth row
new=subset(old,logical)  #select those cases that meet the logical condition
complete = subset(data,complete.cases(data))  #find those cases with no missing values
new=old[n1:n2,n3:n4]  #select the n1 through n2 rows of variables n3 through n4)

B.3 data manipulation

x.df=data.frame(x1,x2,x3 ...)  #combine different kinds of data into a data frame
as.data.frame()
is.data.frame()
x=as.matrix()
scale()  #converts a data frame to standardized scores
factor()  #converts a numeric variable into a factor (essential for ANOVA)
gl(n,k,length)  #makes an n-level,k replicates, length long vector of factors
y <- edit(x)  #opens a screen editor and saves changes made to x into y
fix(x)  #opens a screen editor window and makes and saves changes to x

B.4 Statistics and transformations

max()
min()
mean()
median()
interp.median()  #for interpolated values
sum()
var()  #produces the variance covariance matrix
sd()  #standard deviation
mad()  #median absolute deviation
fivenum()  #Tukey fivenumbers min, lowerhinge, median, upper hinge, max
scale(data, scale=T)  #centers around the mean and scales by the sd
colSums(), rowSums(), colMeans(), rowMeans()  #see also apply(x,1,sum)
rowsum(x,group)  #sum by group
cor(x,y,use="pair")  #correlation matrix for pairwise complete data, use="complete" for complete cases
\texttt{t.test(x,y)} #x is a data vector, y is a grouping vector independent groups  
\texttt{t.test(x,y,pair=TRUE)} #x is a data vector, y is a grouping vector – paired groups  
\texttt{pairwise.t.test(x,g)} does multiple comparisons of all groups defined by g  
\texttt{aov(x,y,data=datafile)} #where x and y can be matrices  
\texttt{aov.ex1 = aov(Alertness Dosage, data=data.ex1)} #do the analysis of variance or  
\texttt{aov.ex2 = aov(Alertness Gender*Dosage, data=data.ex2)} #do a two way analysis of variance  
\texttt{summary(aov.ex1)} #show the summary table  
\texttt{print(model.tables(aov.ex1,"means"), digits=3)} #report the means and the number of subjects/cell  
\texttt{boxplot(Alertness Dosage, data=data.ex1)} #graphical summary appears in graphics window  

\texttt{lm(x,y, data=dataset)} #basic linear model where x and y can be matrices  
\texttt{lm(Y X)} #Y and X can be matrices  
\texttt{lm(Y X1+X2)}  
\texttt{lm(Y X|W)} #separate analyses for each level of W  
\texttt{solve(A,B)} #inverse of A * B - used for linear regression  
\texttt{solve(A)} #inverse of A

\section*{B.5 Useful additional commands}

\texttt{colSums (x, na.rm = FALSE, dims = 1)}  
\texttt{rowSums (x, na.rm = FALSE, dims = 1)}  
\texttt{colMeans(x, na.rm = FALSE, dims = 1)}  
\texttt{rowMeans(x, na.rm = FALSE, dims = 1)}  
\texttt{rowsum(x, group, reorder = TRUE, \ldots)} #finds row sums for each level of a grouping variable  
\texttt{apply(X, MARGIN, FUN, \ldots)} #applies the function (FUN) to either rows (1) or columns (2) on object X  
\texttt{apply(x,1,min)} #finds the minimum for each row  
\texttt{apply(x,2,max)} #finds the maximum for each column  
\texttt{col.max(x)} #another way to find which column has the maximum value for each row  
\texttt{which.min(x)}  
\texttt{which.max(x)}  
\texttt{z=apply(big5r,1,which.min)} #tells the row with the minimum value for every column

\section*{B.6 Graphics}

\texttt{stem()} #stem and leaf diagram  
\texttt{par(mfrow=c(2,1))} #number of rows and columns to graph  
\texttt{boxplot(x, notch=T, names= grouping, main="title")} #boxplot (box and whiskers)
hist() # histogram
plot()
plot(x,y,xlim=range(-1,1),ylim=range(-1,1),main=title)
par(mfrow=c(1,1)) # change the graph window back to one figure
symb=c(19,25,3,23)
colors=c("black","red","green","blue")
charact=c("S","T","N","H")
plot(x,y,pch=symb[group],col=colors[group],bg=colors[condit],cex=1.5,main="main title")
points(mPA,mNA,pch=symb[condit],cex=4.5,col=colors[condit],bg=colors[condit])

curve()
abline(a,b)
abline(a,b,untf = FALSE, ...)
abline(h=,untf = FALSE, ...)
abline(v=,untf = FALSE, ...)
abline(coef=,untf = FALSE, ...)
abline(reg=,untf = FALSE, ...)

identify()
plot(eatar,eanta,xlim=range(-1,1),ylim=range(-1,1),main=title)
identify(eatar,eanta,labels=labels(energy$sR[,1])) # dynamically puts names on the plots
locate()
pairs() # SPLOM (scatter plot Matrix)

matplot() # ordinate is row of the matrix
biplot() # factor loadings and factor scores on same graph
coplot(x,y|z) # x by y conditioned on z
symb=c(19,25,3,23) # choose some nice plotting symbols
colors=c("black","red","green","blue") # choose some nice colors

barplot() # simple bar plot
interaction.plot() # shows means for an ANOVA design

plot(degreedays,therms) # show the data points
by(heating,Location,function(x) abline(lm(therms$degreedays,data=x))) # show the best fitting regression for each group

x= recordPlot() # save the current plot device output in the object x
replayPlot(x) # replot object x
dev.control # various control functions for printing/saving graphic files
Table B.1 Functions used in this chapter. *are part of the psych package.

<table>
<thead>
<tr>
<th>Function</th>
<th>Use</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>?</td>
<td>Help about a function. Same as help.</td>
<td>?round or help(round)</td>
</tr>
<tr>
<td>%*%</td>
<td>Binary operator to do vector or matrix multiplication</td>
<td>A %*% B</td>
</tr>
<tr>
<td>%+%</td>
<td>*Binary operator to do vector or matrix like sums</td>
<td>A %+% B</td>
</tr>
<tr>
<td>==</td>
<td>Test of equality</td>
<td>A == B</td>
</tr>
<tr>
<td>&lt;-</td>
<td>Assignment A is replaced by B. This notation is preferred to =</td>
<td>A &lt;- B</td>
</tr>
<tr>
<td>=</td>
<td>Assignment A is replaced by B. (see &lt;-)</td>
<td>A = B</td>
</tr>
<tr>
<td>[ , ]</td>
<td>Evaluate an element of a matrix , array or data.frame. A[i,j] is in the i'th row, j' th column.</td>
<td>x_{ij} = X[i,j]</td>
</tr>
<tr>
<td>$</td>
<td>Evaluate an element of a list or data.frame. A$B is the element with name B in A.</td>
<td>a.b = A$B</td>
</tr>
<tr>
<td>as.vector()</td>
<td>Make a set of numbers into a vector</td>
<td>A &lt;- as.vector(A)</td>
</tr>
<tr>
<td>c()</td>
<td>Combine two or more items.</td>
<td>A &lt;- c(B,C)</td>
</tr>
<tr>
<td>colnames()</td>
<td>Find or make the column names (also see rownames)</td>
<td>A &lt;- colnames(B) finds colnames(A) &lt; B makes</td>
</tr>
<tr>
<td>rownames()</td>
<td>Find or make the row names (also see colnames)</td>
<td>A &lt;- rownames(A)</td>
</tr>
<tr>
<td>colMeans()</td>
<td>Find column or row means of each column or row in a data.frame or matrix</td>
<td>A &lt;- colMeans(A)</td>
</tr>
<tr>
<td>rowMeans()</td>
<td></td>
<td></td>
</tr>
<tr>
<td>curve()</td>
<td>Plot the curve for a specified function.</td>
<td>curve(1/(1+exp(-x)),-3,3)</td>
</tr>
<tr>
<td>data.frame()</td>
<td>Create a data.frame. (Similar to a matrix, but can have different types of elements)</td>
<td>A &lt;- data.frame(x,y,z)</td>
</tr>
<tr>
<td>describe()</td>
<td>*Report basic descriptive statistics for a vector, matrix, or dataframe</td>
<td>describe(X)</td>
</tr>
<tr>
<td>diag()</td>
<td>Create or find the diagonal of a square matrix</td>
<td>A &lt;- diag(B) finds diag(B) &lt;- A creates</td>
</tr>
<tr>
<td>dim</td>
<td>Report the dimensions (rows,cols) of a data.frame or matrix.</td>
<td>n_rows &lt;- dim(x,df)[1]</td>
</tr>
<tr>
<td>exp()</td>
<td>Raise e to the A power</td>
<td>exp(A)</td>
</tr>
<tr>
<td>for()</td>
<td>Execute a loop from start to finish</td>
<td>for (i in start:finish) {some operation using i}</td>
</tr>
<tr>
<td>function()</td>
<td>Create a new function to do something.</td>
<td>new.f &lt;- function(x,y) {new &lt;- x + y}</td>
</tr>
<tr>
<td>if() else</td>
<td>Do something if a logical condition holds. Do something else if it does not hold.</td>
<td>if(A &lt; B) {print(&quot;B&quot;) } else {print(&quot;A&quot;) }</td>
</tr>
<tr>
<td>length()</td>
<td>Report the number of elements in a vector</td>
<td>n &lt;- length(v)</td>
</tr>
<tr>
<td>list()</td>
<td>A general way of storing results.</td>
<td>A &lt;- list(a=,1,b=2,c=3.4)</td>
</tr>
<tr>
<td>log()</td>
<td>Find the natural logarithm of X</td>
<td>A &lt;- log(X)</td>
</tr>
<tr>
<td>lower.tri()</td>
<td>Logical function, true if an element is in lower triangular submatrix of a matrix (see upper.tri)</td>
<td>A &lt;- lower.tri(B)</td>
</tr>
<tr>
<td>matrix()</td>
<td>Create a matrix of m*n elements with m rows and n columns</td>
<td>A &lt;- matrix(B,ncol=n)</td>
</tr>
<tr>
<td>mean() median()</td>
<td>Find the mean, or median of a data.frame, vector, or matrix</td>
<td>A.m &lt;- mean(A)</td>
</tr>
<tr>
<td>model.fit()</td>
<td>*Calculate 3 alternative goodness of fit indices (see text)</td>
<td></td>
</tr>
<tr>
<td>paste()</td>
<td>Combine several numeric or text variable into a string</td>
<td>A &lt;- paste('B',a,'is',4)</td>
</tr>
<tr>
<td>pairs.panels()</td>
<td>*Plot the scatter plot matrices (SPLOM) and report the correlations for a data.frame or matrix</td>
<td>pairs.panels(x,df)</td>
</tr>
<tr>
<td>pnorm()</td>
<td>What is the probability of an observation, z, given the normal distribution $\approx -\infty &lt; z &lt; \infty$</td>
<td>p &lt;- pnorm(z)</td>
</tr>
<tr>
<td>qnorm()</td>
<td>What is the z score associated with a particular quantile (probability) in a normal distribution $0 &lt; x &lt; 1$</td>
<td>z &lt;- qnorm(x)</td>
</tr>
<tr>
<td>read.clipboard()</td>
<td>*Read a data matrix or data table from the clipboard</td>
<td>A &lt;- read.clipboard()</td>
</tr>
<tr>
<td>rep()</td>
<td>Repeat A N times</td>
<td>rep(A,N)</td>
</tr>
<tr>
<td>round()</td>
<td>Round off numbers to n digits</td>
<td>round(x,n)</td>
</tr>
<tr>
<td>runif()</td>
<td>Create n random numbers, uniformly distributed between a and b. (Defaults to a=0, b=1)</td>
<td>runif(n,a,b)</td>
</tr>
<tr>
<td>sample()</td>
<td>Draw n samples (with or without replacement) from x</td>
<td>sample(2,1000)</td>
</tr>
<tr>
<td>set.seed()</td>
<td>Supply a particular start value to the random number generator</td>
<td>set.seed(42)</td>
</tr>
<tr>
<td>seq()</td>
<td>Form the sequence from lower to upper by step size</td>
<td>x &lt;- seq(lower,upper,step)</td>
</tr>
<tr>
<td>t()</td>
<td>Transpose a vector or matrix</td>
<td>tA &lt;- t(A)</td>
</tr>
<tr>
<td>upper.tri()</td>
<td>Logical function, true if an element is in upper triangular submatrix of a matrix (see lower.tri)</td>
<td>A &lt;- lower.tri(B)</td>
</tr>
</tbody>
</table>