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
mat=rbind(1:8,1:4) #create two rows in a 2 * 8 matrix
mat=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 binomialitem
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[2,] #display the 2nd column
mat=cbind(rep(1:4,2),rep(1:1,2)) #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()
names.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()
sum()
var()  # produces the variance covariance matrix
sd()  # standard deviation
mad()  # (median absolute deviation)
fivenum()  # Tukey five numbers min, lower hinge, 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
t.test(x,y)  # x is a data vector, y is a grouping vector independent groups
t.test(x,y,paired=TRUE)  # x is a data vector, y is a grouping vector – paired groups
pairwise.t.test(x,y) does multiple comparisons of all groups defined by y
aov(x ~ y, data=datafile)  # where x and y can be matrices
aov.ex1 = aov(Alertness Dosage,data=data.ex1)  # do the analysis of variance or
aov.ex2 = aov(Alertness Gender*Dosage,data=data.ex2)  # do a two way analysis of variance
summary(aov.ex1)  # show the summary table
print(model.tables(aov.ex1,"means"),digits=3)  # report the means and the number of subjects/cell
boxplot(Alertness Dosage,data=data.ex1)  # a graphical summary appears in graphics window
lm(x,y, data=dataset) # basic linear model where x and y can be matrices
lm(Y,X) # Y and X can be matrices
lm(Y,X1+X2)
lm(Y,X|W) # separate analyses for each level of W
solve(A,B) # inverse of A * B - used for linear regression
solve(A) # inverse of A

### B.5 Useful additional commands

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

### B.6 Graphics

stem() # stem and leaf diagram
par(mfrow=c(2,1)) # number of rows and columns to graph
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(energyS[,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(degreeSdays,therms) #show the data points
by(heating,Location,function(x) abline(lm(therms degreeSdays,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