A short list of the most useful R commands

A summary of the most important commands with minimal examples. See the relevant part of the guide for better examples. For all of these commands, using the help(function) or ? function is the most useful source of information. Unfortunately, knowing what to ask for help about is the hardest problem.

See the R-reference card by Tom Short for a much more complete list.

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

x=c(1,2,4,8,16 )                          #create a data vector with specified elements
y=c(1:10)                                 #creat a data vector with elements 1-10
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
vect=c(x,y)                               #combine them into one vector of length 2n
mat=cbind(x,y)                            #combine them into a n x 2 matrix
mat[4,2]                                  #display the 4th row and the 2nd column
mat[3,]                                   #display the 3rd row
mat[,2]                                   #display the 2nd column
subset(dataset,logical)                   #those objects meeting a logical criterion
subset(data.df,select=variables,logical)  #get those objects from a data frame that meet
                                            a criterion
data.df[data.df=logical]                  #yet another way to get a subset
x[order(x$B),]                            #sort a dataframe by the order of the elements
                                            in B
x[rev(order(x$B)),]                       #sort the dataframe in reverse order
browse.workspace                          #a menu command that creates a window with
                                            information about all variables in the
                                            workspace
```

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
or data frame available in the workspace
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.df,complete.cases(data.df)) #find those cases with no missing values
new=old[n1:n2,n3:n4]                    #select the n1 through n2 rows of variables n3
                                          through n4)
```

distributions

```
beta(a, b)
gamma(x)
choose(n, k)
factorial(x)
dnorm(x, mean=0, sd=1, log = FALSE) #normal distribution
pnorm(q, mean=0, sd=1, lower.tail = TRUE, log.p = FALSE)
qnorm(p, mean=0, sd=1, lower.tail = TRUE, log.p = FALSE)
rnorm(n, mean=0, sd=1)
dunif(x, min=0, max=1, log = FALSE) #uniform distribution
punif(q, min=0, max=1, lower.tail = TRUE, log.p = FALSE)
qunif(p, min=0, max=1, lower.tail = TRUE, log.p = FALSE)
runif(n, min=0, max=1)
```

data manipulation

```
replace(x, list, values)                    #remember to assign this to some object i.e., x <-
                                              replace(x,x==-9,NA)
                                             #similar to the operation x[x==-9] <- NA


cut(x, breaks, labels = NULL,
      include.lowest = FALSE, right = TRUE, dig.lab = 3, ...)

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

round(x,n)                                   #rounds the values of x to n decimal places
ceiling(x)                                   #vector x of smallest integers > x
floor(x)                                     #vector x of largest interger < x
as.integer(x)                                #truncates real x to integers (compare to
                                               round(x,0)
as.integer(x < cutpoint)                     #vector x of 0 if less than cutpoint, 1 if greater
                                               than cutpoint)
factor(ifelse(a < cutpoint, "Neg", "Pos"))   #is another way to dichotomize and to make a
                                               factor for analysis
transform(data.df,variable names = some operation) #can be part of a set up for a data set

x%in%y                                       #tests each element of x for membership in y
y%in%x                                       #tests each element of y for membership in x
all(x%in%y)                                  #true if x is a proper subset of y
all(x)                                       # for a vector of logical values, are they all true?
any(x)                                       #for a vector of logical values, is at least one
true?
```

Statistics and transformations

```
max()
min()
mean()
```

```
median()
sum()
var()                    #produces the variance covariance matrix
sd()                     #standard deviation
mad()                    #(median absolute deviation)
fivenum()                #Tukey fivenumbers min, lowerhinge, median, upper hinge, max
table()                  #frequency counts of entries, ideally the entries are factors(although it
                           works with integers or even reals)
scale(data,scale=T)      #centers around the mean and scales by the sd)
cumsum(x)                #cumulative sum, etc.
cumprod(x)
cummax(x)
cummin(x)
rev(x)                     #reverse the order of values in x

cor(x,y,use="pair")      #correlation matrix for pairwise complete data, use="complete" for
                           complete cases

aov(x~y,data=datafile)   #where x and y can be matrices
   aov.ex1 = aov(DV~IV,data=data.ex1)          #do the analysis of variance or
   aov.ex2 = aov(DV~IV1*IV21,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(DV~IV,data=data.ex1)                   #graphical summary appears in graphics
                                                 window

lm(x~y,data=dataset)                           #basic linear model where x and y can be
                                                 matrices (see plot.lm for plotting options)
t.test(x,g)
pairwise.t.test(x,g)
power.anova.test(groups = NULL, n = NULL, between.var = NULL,
                        within.var = NULL, sig.level = 0.05, power = NULL)
power.t.test(n = NULL, delta = NULL, sd = 1, sig.level = 0.05,
                    power = NULL, type = c("two.sample", "one.sample", "paired"),
                    alternative = c("two.sided", "one.sided"),strict = FALSE)
```

More statistics: Regression and Linear model

```
lm(Y~X)                                  #Y and X can be matrices
lm(Y~X1+X2)
lm(Y~X|W)
solve(A,B)                               #inverse of A * B - used for linear regression
solve(A)                                 #inverse of A
factanal()
princomp()
```

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
```

Graphics

```
par(mfrow=c(nrow,mcol))                   #number of rows and columns to graph
par(ask=TRUE)                             #ask for user input before drawing a new graph
par(omi=c(0,0,1,0) )                       #set the size of the outer margins
```

```
mtext("some global title",3,outer=TRUE,line=1,cex=1.5)       #note that we seem to need to add the global title last
                                                            #cex = character expansion factor

boxplot(x,main="title")                                     #boxplot (box and whiskers)

title( "some title")                                        #add a title to the first graph

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(PA,NAF,pch=symb[group],col=colors[group],bg=colors[condit],cex=1.5,main="P
            ostive vs. Negative Affect by Film condition")
        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(energysR[,1])    )          #dynamically puts names on the plots
locate()

legend()
pairs()             #SPLOM (scatter plot Matrix)
pairs.panels ()     #SPLOM on lower off diagonal, histograms on diagonal, correlations on diagonal
                        #not standard R, but uses a function found in useful.r
```

```
matplot ()
biplot ())
plot(table(x))            #plot the frequencies of levels in x

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
pdf(height=6, width=6)        #create a pdf file for output
dev.of()                      #close the pdf file created with pdf
layout(mat)                   #specify where multiple graphs go on the page
                              #experiment with the magic code from Paul Murrell to do fancy
graphic location
layout(rbind(c(1, 1, 2, 2, 3, 3),c(0, 4, 4, 5, 5, 0)))
for (i in 1:5) {
   plot(i, type="n")
   text(1, i, paste("Plot", i), cex=4)
}
```

Distributions

To generate random samples from a variety of distributions

```
runif(n,lower,upper)
rnorm(n,mean,sd)
rbinom(n,size,p)
sample(x, size, replace = FALSE, prob = NULL)          #samples with or without replacement
```

Working with Dates

```
date <-strptime(as.character(date), "%m/%d/%y")     #change the date field to a internal form
for time
                                                    #see ?formats and ?POSIXlt
as.Date
month= months(date)                        #see also weekdays, Julian
```

Additional functions that I have created because I needed some specific operation may be included in the workspace by issuing the source command:

source(http://personality-project.org/r/useful.r)

These functions include:

#alpha.scale #find coefficient alpha for a scale and a dataframe of items
#describe give means, sd, skew, n, and se
#summ.stats #basic summary statistics by a grouping variable
#error.crosses (error bars in two space)
#skew find skew
#panel.cor taken from the examples for pairs
#pairs.panels adapted from panel.cor -- gives a splom, histogram, and correlation matrix
#multi.hist #plot multiple histograms
#correct.cor #given a correlation matrix and a vector of reliabilities, correct for reliability
#fisherz #convert pearson r to fisher z
#paired.r #test for difference of dependent correlations
#count.pairwise #count the number of good cases when doing pairwise analysis
#eigen.loadings #convert eigen vector vectors to factor loadings by unnormalizing them
#principal #yet another way to do a principal components analysis -- brute force eignvalue decomp
#factor.congruence #find the factor congruence coeffiecints
#factor.model #given a factor model, find the correlation matrix
#factor.residuals #how well does it fit?
#factor.rotate # rotate two columns of a factor matrix by theta (in degrees)
#phi2poly #convert a matrix of phi coefficients to polychoric correlations

---

part of a short guide to R
Version of February 20, 2005
William Revelle
Department of Psychology
Northwestern University