

# The R Environment

(Adapted from the Venables and Smith R Manual on [www.R-project.org](http://www.R-project.org) and from Andreas Buja's web site for Applied Statistics at

<http://www-stat.wharton.upenn.edu/~buja/STAT-541/Notes-Stat-541.R>)

R is an integrated suite of software facilities for data manipulation, calculation and graphical display. Among other things it has:

- a effective data handling and storage facility,
- a suite of operators for calculations on arrays, in particular matrices,

- a large, coherent, integrated collection of intermediate tools for data analysis,
- graphical facilities for data analysis and display,
- a well-developed, simple and effective programming language.

```
#          INTRO TO R/SPLUS
```

```
# The R website:
```

```
#  http://www.r-project.org
```

```
# Downloading R:
```

```
#
```

```
#  -> (Download) CRAN -> http://cran.us.r-project.org
```

```
#  -> Windows (95 and later) -> base -> rw1062.exe
```

```
#
```

```
#  rw1062.exe should self-install and create an icon on your desktop.
```

```
#  Clicking the icon should open up an R interpreter window.
```

```
#
```

```
#  The base is really just the base. There are many contributed
```

```
#  library packages whose binaries can be downloaded from
```

```
#  
# -> Package Sources  
#  
# You will not have to download them explicitly, though;  
# there are R functions that allow you to get them while running R.  
# These are accessed from the GUI Packages menu
```

## Starting and closing R

R can be started by double-clicking the RGui icon on the Windows screen.

Once started it can be stopped by typing

```
q()
```

in the commands window. R saves work, data and functions. When you quit it will ask whether you want to save the session. Mostly you want to do this.

## Recommended operation

- Go to the class web page and download the sample R code files. Open an editor (Word, Wordpad,...) on this file AND open up an R window. Reduce the size of both windows so both are accessible.
- Copy R code from this file into the R interpreter window; use shortcuts: highlight lines, hit `<Ctrl>-C` in the editor, then move to the R window and hit `<Ctrl>-V` repeat...
- Experiment with R code by editing the sample files in the editor window, or by editing the command line in the R window (if it's 1 line)

```
# commands for line editing in the R interpreter window:
# ^p get back the previous command line for editing and executing
# ^b step back one character in the command line
# ^f step forward ...
# ^a move to the beginning of the line
# ^e move to the end of the line
# ^d or Delete delete the character under the cursor
# ^h or Backspace to delete the previous character
# ^k kill the rest of the line starting from the cursor
# otherwise: you are in insert mode

# Note: "^p" means you hold down the modifier key <Ctrl> and hit "p",
#       just like the modifier key <Shift> used for capitalization.
```

## getting started

# Assignments:

```
x <- sqrt(2 * 3)      # root of product of 2 and 3
y <- c(1,3,10,1,1,1111,0,1,1,1,1,1,1)
# combine the values 1,3,10 into a vector
```

# Side effects:

```
x                # print or return x
```

# Syntax: largely scientific/math notation; 10

# considerable set of functions;

# comments run from a "#" to the end of the line.

## help and object listing

```
help(sqrt)
```

```
help(c)
```

```
help("c")    # same as help(c)
```

```
help(*)      # works in R, but not splus
```

```
help("*")    # works in both
```

```
# Listing R objects, including data and functions:
```

```
ls()         # your current data
```

```
ls(pos=1)
```

```
ls(pos=2)
```

```
ls(pos=3)
```

```
ls(pos=4)    # probably this one will list 1500 functions
```

## basic data structures

```
# manual entry: vector
```

```
x <- c(-1,2,5)
```

```
# equispaced sequences of numbers:
```

```
x <- 3:10
```

```
x <- seq(3, 10)
```

```
x <- seq(3, 10, by=1/3)
```

```
x <- seq(3, 10, len=8)
```

```
x <- rep(3, 10)
```

```
x <- rep(1:3, 5)
```

```
x <- rep(c(1,2,3,4), c(2,3,2,3))
```

```
x <- rep(1:3, rep(2,3))
```

```
# logical values T(=TRUE) and F(=FALSE):  
  x <- c(T,T,F,T,T,T,F)  
  x <- rep(T,5)  
  x <- ((1:10) > 5)  
  
# matrices  
  x <- matrix(c(1,2,3,4,5,6),ncol=2,byrow=T)  
  dimnames(x)<-list(c("Roe 1","Roe 2","Roe 3"),c("Var1","Var2"))  
  
# data frames  
  x<-data.frame(x)  
  attach(x)  
  Var1  
  Var2  
  detach(x)
```

```
# lists: ordered collections of objects
x <- list(name="Cox", wife="Mary", husband="Fred", no.child=3,
          child.ages=c(4,8,9))
x <- list(name=c("Cox","Wang"), wife=c("Mary","Pearl"),
          husband=c("Fred","Val"), no.child=c(3,2),
          child.ages=c(c(4,8,9),c(1,5)))

x$name
x$no.child
```

## reading/writing data from files

```
# for data of the form
# Name Price Floor Area Rooms Cent.Heat
# Coral 52.00 111 830 5 no
# Teal 54.75 128 710 5 no
# Ocean 57.50 101 1000 7 yes
# ...
x <- read.table("C:\\temp\\houses.asc",header=T, row.names=1)
x <- read.table("C:\\temp\\houses.dat")

# writing data
write.table(x,file="C:\\temp\\houses2.asc",
  row.names=T,col.names=T,sep=" ",quote=F,
  append=T)
```

```
x<-runif(100000)
cat(x,sep="\n",file="C:\\temp\\sp.dat")
```

```
# read vector from file: (first n values)
x <- scan("C:\\temp\\sp.dat", n=1000)
x <- scan("C:\\temp\\sp.dat", n=50000)
options(memory=1E10, object.size=1E10)
x <- scan("C:\\temp\\sp.dat")
```

## subsetting/selecting from a vector

```
# subsetting/selecting from a vector
x <- 1:100 * 10 # same as: seq(10,1000,by=10)
x[c(1,11,21)]
x[seq(1,21,by=10)]
x[1:10]
x[50:150]
x[x>500]

# simple statistics:
length(x); sum(x)
mean(x); var(x); sqrt(var(x))
min(x); max(x); range(x)
median(x)
```

## simple functions/transformations

```
x <- runif(10, -100, 100)
```

```
y <- round(x)
```

```
abs(x)
```

```
x*x; x^2
```

```
log(x^2); 2*log(abs(x))
```

```
x > 0
```

```
# ranking, sorting:
```

```
rank(x)
```

```
order(x)
```

```
sort(x); x[order(x)]
```

## matrices

```
m <- cbind(1:5, x[1:5])  
m <- rbind(1:5, x[1:5])  
m <- matrix(1:36, ncol=4)  
m <- matrix(1:12, ncol=3, byrow=T)
```

# subselecting rows and columns:

```
m  
m[1:2,]          # first 2 rows  
m[,1:2]          # first 2 columns  
m[1:2,c(1,3)]    # first 2 rows, 1st,3rd columns
```

```
# operating on rows of matrices:  
# return a vector with minima of the rows  
  apply(m, 1, min)  
# operating on columns of matrices:  
# return a vector with the means of the columns  
  apply(m, 2, mean)  
# return a vector with the sdevs of the columns  
  sqrt(apply(m, 2, var))
```

## looping

```
# looping
x <- runif(100); x <- sort(x); y <- NULL
for (i in 1:length(x))
  y <- c(y, x[i]*i)

y <- NULL
for (i in 1:length(x)) {
  z <- x[i]*i
  if (z < 10) y <- c(y, z)
}

y <- 0
while (y < 10)
  y <- y+1
```

## graphics

```
# graphics
x <- runif(100)
x<-sort(x)
y <- x^2+rnorm(100)*0.1
plot(x,y)
plot(x,y,pch=16)
plot(x,y,xlim=c(0,0.5))
plot(x,y,xlab="Cost",ylab="Number",main="Houses data")
par(mfrow=c(2,2))
for (i in 1:4) {
  plot(x[((i-1)*25+1):(i*25)],y[((i-1)*25+1):(i*25)],
       xlim=range(x),ylim=range(y),xlab="x",ylab="y")
  abline(lsfit(x[((i-1)*25+1):(i*25)],
              y[((i-1)*25+1):(i*25)]))$coef)
}
```

```
par(mar=c(0.2,0.2,0.2,0.2))
for (i in 1:4) {
  plot(x[((i-1)*25+1):(i*25)],y[((i-1)*25+1):(i*25)],
       xlim=range(x),ylim=range(y),xaxt="n",yaxt="n")
  abline(lsfrit(x[((i-1)*25+1):(i*25)],
              y[((i-1)*25+1):(i*25)])$coef,lwd=2*i)
}
```

## writing functions

```
# function computes standard deviation of a vector
sd <- function(x) {
  return(sqrt(var(x)))
}
x<-matrix(runif(100),ncol=5)
for (i in 1:5)
  x[,i]<-x[,i]*i
apply(x,2,sd)

# loading functions from files
source("mosaic.plot")
```