Introduction to R and Rstudio

### Download and install R and Rstudio

1. Go to <http://www.r-project.org>. This is the home page for the R project.
2. On this page click on the link with the words Download R.
3. There is a list of many web servers from which you can download R. Choose one of these. The one in Iowa is faster than the one in Michigan.
4. Near the top of the page will be a section titled Download and install R. Choose whether it is for Windows or for a MAC.
5. Next install Rstudio. This is an interface for R that students find easier to use.
6. Go to <http://www.rstudio.com/>
7. Click on Download Rstudio
8. Click on the Desktop iccon
9. Under Open Source Edition Click on Download Rstudio Desktop.
10. Choose the version for MAC or Windows.
11. Follow the instruction in the pop-up windows.

### Before you start

Create a folder *Statwork* where you can save relevant documents. Create subfolders, e.g. *Rstudio* and *Documents* and *Data*, where you will save your computational work.

### R script

In Rstudio click on File > New File > R script. In this window you can type code and comments (with # sign for comments) that you can save to your folder. When you highlight a piece of code and click Run it will copy the code to the Console window and execute. Alternatively, you can type directly into the Console window, but it won't save your text.

## Important:

Click on Session > Set Working Directory > To Source File Location. Then Rstudio will know where to save your putout from your work. You can check your working directory anytime in the Console of Rstudio by typing

## Examples of calculations in R/Rstudio

2 + 3 \* 5 # Note the order of operations.

## [1] 17

log(10) # Natural logarithm with base e=2.718282

## [1] 2.302585

4^2 # 4 raised to the second power

## [1] 16

3/2 # Division

## [1] 1.5

sqrt(16) # Square root

## [1] 4

abs(3-7) # Absolute value of 3-7

## [1] 4

exp(2) # exponential function

## [1] 7.389056

15 %/%4 # This is the integer divide operation

## [1] 3

# This is a comment line

Let's type a data list into R. The letter "c" stands for "concatenate".

x = c(1, 3, 7, 9, 11, 13, 20)  
x

## [1] 1 3 7 9 11 13 20

Note that in the output the square bracket [1] refers to the first line of output. Sometimes you may have longer lines. They will be broken up and numbered this way.

Suppose you made an error and really wanted the first number in the data list called x to be a 10.

x[1] = 10  
x

## [1] 10 3 7 9 11 13 20

This is a quick way to change something without retyping the whole data list. You can check a number of things in your data list:

x[3] # the third element of x

## [1] 7

x[3:5] # the third to fifth element of x, inclusive

## [1] 7 9 11

x[x>3] # list of elements in x greater than 3

## [1] 10 7 9 11 13 20

You can also add a constant to each element in your data list or multiply by a constant:

x

## [1] 10 3 7 9 11 13 20

x+1

## [1] 11 4 8 10 12 14 21

x\*2

## [1] 20 6 14 18 22 26 40

R is case sensitive, so if you type upper case X instead of the lower case X, you will get an error message.

X

## Error in eval(expr, envir, enclos): object 'X' not found

Note: I prefer lower case names for variables. Then I don't have to remember whether a name was upper or lower case.

Now you have created a data list for which you can calculate data summaries, e.g.

mean(x)

## [1] 10.42857

sd(x)

## [1] 5.287001

IQR(x)

## [1] 4

summary(x)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 3.00 8.00 10.00 10.43 12.00 20.00

## More on data elements in R/Rstudio

Basics with data lists (vectors)

### basics with data vectors  
x<-c(1,3,2,10,5)  
x

## [1] 1 3 2 10 5

sum(x)

## [1] 21

summary(x)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1.0 2.0 3.0 4.2 5.0 10.0

sort(x, decreasing=T)

## [1] 10 5 3 2 1

y<-1:5  
y+2

## [1] 3 4 5 6 7

y\*2

## [1] 2 4 6 8 10

y^2

## [1] 1 4 9 16 25

length(x)

## [1] 5

length(y)

## [1] 5

x+y

## [1] 2 5 5 14 10

z<-cbind(x,y)  
z<-rbind(x,y)

Basics with data matrices

A<-matrix(c(1,2,3,4), nrow=2, ncol=2); A

## [,1] [,2]  
## [1,] 1 3  
## [2,] 2 4

rownames(A)<-c("row1", "row2")  
 colnames(A)<-c("col1", "col2")  
 dim(A)

## [1] 2 2

A<-matrix(c(1,2,3,4),byrow=TRUE);A

## [,1]  
## [1,] 1  
## [2,] 2  
## [3,] 3  
## [4,] 4

sqrt(A)

## [,1]  
## [1,] 1.000000  
## [2,] 1.414214  
## [3,] 1.732051  
## [4,] 2.000000

A^2

## [,1]  
## [1,] 1  
## [2,] 4  
## [3,] 9  
## [4,] 16

B<-matrix(c(1,2,3,4,5,6),nrow=2,ncol=3, byrow=T);B

## [,1] [,2] [,3]  
## [1,] 1 2 3  
## [2,] 4 5 6

B[order(B[,1], decreasing=T),]

## [,1] [,2] [,3]  
## [1,] 4 5 6  
## [2,] 1 2 3

B[1,2]

## [1] 2

B[2,] # second row

## [1] 4 5 6

t(B) # transpose of B

## [,1] [,2]  
## [1,] 1 4  
## [2,] 2 5  
## [3,] 3 6

B[-1,] # remove first row

## [1] 4 5 6

B[,-c(1,3)] # remove first and third column

## [1] 2 5

## Data frames in R

A data frame in R gathers together all the variables in a data set.

library(SemiPar) # if this generates and error message, do the following first: install.packages("SemiPar")  
data(fuel.frame)  
car<-fuel.frame  
names(car)

## [1] "car.name" "Weight" "Disp." "Mileage" "Fuel" "Type"

dim(car)

## [1] 60 6

car[1:3,]

## car.name Weight Disp. Mileage Fuel Type  
## 1 Eagle Summit 4 2560 97 33 3.030303 Small  
## 2 Ford Escort 4 2345 114 33 3.030303 Small  
## 3 Ford Festiva 4 1845 81 37 2.702703 Small

unique(car$car.name)[1:3] # specify the variable name after a dollar sign

## [1] Eagle Summit 4 Ford Escort 4 Ford Festiva 4  
## 60 Levels: Acura Legend V6 Audi 80 4 Buick Century 4 ... Volvo 240 4

mean(car$Mileage)

## [1] 24.58333

table(car$Type)

##   
## Compact Large Medium Small Sporty Van   
## 15 3 13 13 9 7

## Finish your R/Rstudio session

When you close Rstudio (e.g. click on the red button). It will ask you whether you want to save the R work space. I never do. If you have written all your code in a text file, script file or R markdown file, you can always pull it up and recreate what you have done.