Lab 1: An Introduction to R and RStudio

Armin Hatefi

Saturday, January 15, 2022

Contents

Starting RStudio 1

Help Commands 1

Simple Tasks in R 2

Creating Variables 2

Creating Vectors 2

Operations on Vectors 3

Making a Matrix and Having Access to it 4

Creating Data Frame 5

Simple Random Sampling 7

This document shows you how to do simple tasks in RStudio.

Starting RStudio

RStudio is a free software and is available on all the machines in most of the university computer labs. But this semester, we have to work on computer labs remotely. First, log in to your computer and to start RStudio.

Note: Throughout this lab we work with `RStudio Console` and type the following commands in Console. When we hit return key, the Console runs the command immediately. Later, we learn to type the commands in `RStudio Editor` and then run the commands.

Help Commands

You can type either of

```
help(log)
?log
log(68)
```

```
## [1] 4.219508
```

To display the help file for the log (or any other) command. Type `help.start()` to start a help window. This is a way to list all the R commands and is **very useful**.

Simple Tasks in R

RStudio is an interactive computing environment which you will use for data analysis. It can also be used as a calculator to perform simple tasks:

```
5*3
```

```
## [1] 15
```

```
sqrt(25)
```

```
## [1] 5
```

```
4^2
```

```
## [1] 16
```

```
abs(-7)
```

```
## [1] 7
```

Creating Variables

There are two assignment operators in R `<-` and `=`. They can be used interchangeably.

```
x = 5
x
```

```
## [1] 5
```

```
x <- 5
x
```

```
## [1] 5
```

```
y = 3
z = 4
x + y + z
```

```
## [1] 12
```

```
y
```

```
## [1] 3
```

Creating Vectors

```
x = 4:12
x
```

```
## [1]  4  5  6  7  8  9 10 11 12
```

```r
y = seq(4,12,by=2)
y
```

```
## [1]  4  6  8 10 12
```

```r
y = seq(4,12,length=25)
y
```

```
##  [1]  4.000000  4.333333  4.666667  5.000000  5.333333  5.666667  6.000000
##  [8]  6.333333  6.666667  7.000000  7.333333  7.666667  8.000000  8.333333
## [15]  8.666667  9.000000  9.333333  9.666667 10.000000 10.333333 10.666667
## [22] 11.000000 11.333333 11.666667 12.000000
```

```r
height <- c(65,70, 66, 71, 66, 63)
height
```

```
## [1] 65 70 66 71 66 63
```

```r
height * 2.54
```

```
## [1] 165.10 177.80 167.64 180.34 167.64 160.02
```

```r
height[5]
```

```
## [1] 66
```

```r
height[c(2,5)]
```

```
## [1] 70 66
```

```r
height[-1]
```

```
## [1] 70 66 71 66 63
```

```r
length(height)
```

```
## [1] 6
```

```r
weight <- c(142,182,100,167,111,162)
weight
```

```
## [1] 142 182 100 167 111 162
```

```r
height/weight
```

```
## [1] 0.4577465 0.3846154 0.6600000 0.4251497 0.5945946 0.3888889
```

```r
name <- c('Marta',"John",'Doug','Sarah','Jen',"Jeff")
name
```

```
## [1] "Marta" "John"  "Doug"  "Sarah" "Jen"   "Jeff"
```

Operations on Vectors

Create two vectors x and y, **having the same length** and see what happens when you do each of the following operations. For example, we use height and weight from above.

```r
x = c(5,2,1,4)
y = c(15,12,10,13)
x-y
```

```
## [1] -10 -10  -9  -9
```

```
y / x
```

```
## [1]  3.00  6.00 10.00  3.25
```

```
x * y
```

```
## [1] 75 24 10 52
```

```
x^2
```

```
## [1] 25  4  1 16
```

```
log(x)
```

```
## [1] 1.6094379 0.6931472 0.0000000 1.3862944
```

```
cbind(x,y)
```

```
##      x  y
## [1,] 5 15
## [2,] 2 12
## [3,] 1 10
## [4,] 4 13
```

```
rbind(x,y)
```

```
##   [,1] [,2] [,3] [,4]
## x    5    2    1    4
## y   15   12   10   13
```

Making a Matrix and Having Access to it

```
A = matrix(1:10,nrow = 5,ncol = 2)
A
```

```
##      [,1] [,2]
## [1,]    1    6
## [2,]    2    7
## [3,]    3    8
## [4,]    4    9
## [5,]    5   10
```

```
B = matrix(1:10,nrow = 5,ncol = 2, byrow = T)
B
```

```
##      [,1] [,2]
## [1,]    1    2
## [2,]    3    4
## [3,]    5    6
## [4,]    7    8
## [5,]    9   10
```

```
B[4,1]
```

```
## [1] 7
```

```r
B[1,2]
```

```
## [1] 2
```

```r
B[,1]
```

```
## [1] 1 3 5 7 9
```

```r
B[3,]
```

```
## [1] 5 6
```

```r
B[c(1,4),]
```

```
##      [,1] [,2]
## [1,]    1    2
## [2,]    7    8
```

Creating Data Frame

```r
students <- data.frame(name,height,weight)
students
```

```
##    name height weight
## 1 Marta     65    142
## 2  John     70    182
## 3  Doug     66    100
## 4 Sarah     71    167
## 5   Jen     66    111
## 6  Jeff     63    162
```

```r
students[4,2]
```

```
## [1] 71
```

```r
students[4,]
```

```
##    name height weight
## 4 Sarah     71    167
```

```r
students[,2]
```

```
## [1] 65 70 66 71 66 63
```

```r
cars
```

```
##    speed dist
## 1      4    2
## 2      4   10
## 3      7    4
## 4      7   22
## 5      8   16
## 6      9   10
## 7     10   18
## 8     10   26
## 9     10   34
## 10    11   17
## 11    11   28
## 12    12   14
## 13    12   20
## 14    12   24
## 15    12   28
## 16    13   26
## 17    13   34
## 18    13   34
## 19    13   46
## 20    14   26
## 21    14   36
## 22    14   60
## 23    14   80
## 24    15   20
## 25    15   26
## 26    15   54
## 27    16   32
## 28    16   40
## 29    17   32
## 30    17   40
## 31    17   50
## 32    18   42
## 33    18   56
## 34    18   76
## 35    18   84
## 36    19   36
## 37    19   46
## 38    19   68
## 39    20   32
## 40    20   48
## 41    20   52
## 42    20   56
## 43    20   64
## 44    22   66
## 45    23   54
## 46    24   70
## 47    24   92
## 48    24   93
## 49    24  120
## 50    25   85
```

```
cars[,1]
```

```
##  [1]  4  4  7  7  8  9 10 10 10 11 11 12 12 12 12 13 13 13 13 14 14 14 14 15 15
## [26] 15 16 16 17 17 17 18 18 18 18 19 19 19 20 20 20 20 20 22 23 24 24 24 24 25
```

```
cars$speed
```

```
##  [1]  4  4  7  7  8  9 10 10 10 11 11 12 12 12 12 13 13 13 13 14 14 14 14 15 15
## [26] 15 16 16 17 17 17 18 18 18 18 19 19 19 20 20 20 20 20 22 23 24 24 24 24 25
```

```
cars$dist
```

```
##  [1]   2  10   4  22  16  10  18  26  34  17  28  14  20  24  28  26  34  34  46
## [20]  26  36  60  80  20  26  54  32  40  32  40  50  42  56  76  84  36  46  68
## [39]  32  48  52  56  64  66  54  70  92  93 120  85
```

Simple Random Sampling

In class, we learned how to take a simple random sample (SRS)

- Table of Random Digits
- Applets on Internet

We can use RStudio and take a simple random sample very easily. To take an SRS, we use command `sample`.

Let's first load a built-in data set called `rivers`.

```
? rivers
sample(rivers,size=20)
```

```
##  [1]  291  444  280  320  420  270  605  202  350  900  840  981  735  625  250
## [16]  524  390  260 1100  217
```

```
sample(rivers,20)
```

```
##  [1]  720  431 1000  350  314 2348  618  237  350  276  233  327  352  290  870
## [16]  529 1038  630  286  260
```

```
sample(rivers,20,replace=T)
```

```
##  [1]  524  215  981  735  630  314  330  210  360  255  410  301  720  600  375
## [16] 1171  720  500  281  981
```

```
sample(x=rivers,size=20,replace=T)
```

```
##  [1]  210  445  210  300  444  671  630  383  230  350  470  259  500 1270  500
## [16]  338  325  430  424  360
```

Let's now make the problem a little more complicated. This time, we load `trees` data set.

```
? trees
head(trees)
```

```
##   Girth Height Volume
## 1   8.3     70   10.3
## 2   8.6     65   10.3
## 3   8.8     63   10.2
## 4  10.5     72   16.4
## 5  10.7     81   18.8
## 6  10.8     83   19.7
```

```
 dim(trees)
```

```
## [1] 31  3
```

```
index <- sample(1:31,size=12)
index
```

```
##  [1]  5 29 15 22 23  8  4 12 17  9 31 24
```

```
trees[24,2]
```

```
## [1] 72
```

```
trees12 <- trees[index,]
trees12
```

```
##    Girth Height Volume
## 5   10.7     81   18.8
## 29  18.0     80   51.5
## 15  12.0     75   19.1
## 22  14.2     80   31.7
## 23  14.5     74   36.3
## 8   11.0     75   18.2
## 4   10.5     72   16.4
## 12  11.4     76   21.0
## 17  12.9     85   33.8
## 9   11.1     80   22.6
## 31  20.6     87   77.0
## 24  16.0     72   38.3
```