EPID674 Epidemiologic Data Analysis using R

Getting Started with R

Kelly Bakulski

Last compiled on January 18, 2021

## R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

## Chapter 1, Getting Started with R

## Explaining commands/output

An R command (we’ll also call it code or a code chunk) will be grey and look like this

print("I'm code")

## [1] "I'm code"

And then directly after it, will be the output of the code.  
So, print("I'm code") is the code chunk and [1] “I’m code” is the output.

## R as a calculator

The R console is a full calculator Try to play around with it: +, -, /, \* are add, subtract, divide and multiply ^ or \*\* is power parentheses – ( and ) – work with order of operations

2 + 2

## [1] 4

2 \* 4

## [1] 8

2 ^ 3

## [1] 8

2 + (2 \* 3)^2

## [1] 38

(1 + 3) / 2 + 45

## [1] 47

Note, when you type your command, R inherently thinks you want to print the result.

Test your comprehension: Try evaluating the following: 2 + 2 \* 3 / 4 -3 2 \* 3 / 4 \* 2 2^4 - 1

## Commenting in Scripts

# is the comment symbol in R

# Comments in R follow the hashtag symbol  
# Anything from the # to the end of the line (on the right) will be ignored by R. Note the color of your code changes after you use a hashtag  
  
# This # is still a comment  
### You can use many #'s as you want  
1 + 2 # Can be the right of code

## [1] 3

# Best practice is to take a ton of notes to help your future self and anyone who comes later to re-run your code

## Make your first R objects

* You can create variables from within the R environment and from files on your computer
* R uses “<-”or “=” to assign values to a variable name
* Variable names are case-sensitive, i.e. X and x are different

x <- 4 # x is assigned the value of 4. # Same as: x = 2  
x # See what x is

## [1] 4

y <- 7 # y is assigned the value of 7  
x + y # See what x+y is

## [1] 11

z <- x \* (x + y) # z is assigned the value of x\*(x+y)  
z # See what z is

## [1] 44

# Perform some calculations with ‘x’ and observe the results

You can perform functions to entire vectors of numbers very easily.

x + 4

## [1] 8

x - 3

## [1] 1

x \* 7

## [1] 28

x / 10

## [1] 0.4

x \* x

## [1] 16

x^2

## [1] 16

x / x

## [1] 1

x == x

## [1] TRUE

# Assign a vector of integers from 1 to 4 to the object x, using multiple different functions

* Here we introduce “1 dimensional” classes; often referred to as ‘vectors’
* The function c() collects/combines/joins single R objects into a vector of R objects. It is mostly used for creating vectors of numbers, character strings, and other data types.
* The function length(): Get or set the length of vectors (including lists) and factors, and of any other R object for which a method has been defined.

x <- 1:4 # This writes over the previous x object. No warning or error messages!  
x

## [1] 1 2 3 4

length(x) # length is a function for checking how long an object is

## [1] 4

x <- seq(from=1, to=4, by=1) # Create a vector counting from 1 to 4 by 1  
x

## [1] 1 2 3 4

x <- c(1, 2, 3, 4) # Combine elements into a vector  
x

## [1] 1 2 3 4

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

## [1] 2 3 4 1

length(x)

## [1] 4

y <- rep(7, times=4) # Create a vector with four 7's by rep() and assign it to y  
y

## [1] 7 7 7 7

# Make a character vector

z <- c("UM", "SPH", "EPI", "Kelly")  
z <- c("University of Michigan", "School of Public Health", "Epidemiology", "John Snow")  
z

## [1] "University of Michigan" "School of Public Health"  
## [3] "Epidemiology" "John Snow"

## R variable classes

* Vectors can have multiple sets of observations, but each observation has to be the same class.

class(x)

## [1] "numeric"

class(z)

## [1] "character"

# What are the differences between ( ) and [ ] ?

length(z) # Rounded parentheses are for functions

## [1] 4

z[3] # Square brackets are for looking in objects, this is also called "indexing"

## [1] "Epidemiology"

z[2] # Find value based on position

## [1] "School of Public Health"

z[1:3]

## [1] "University of Michigan" "School of Public Health"  
## [3] "Epidemiology"

#Find positions meeting criteria  
x<3 #Provides True/False for whether meets the criteria

## [1] TRUE FALSE FALSE TRUE

which(x<3) #Finds the positions of the Trues

## [1] 1 4

#Find values meeting criteria  
x[x<3]

## [1] 2 1

# Specify options in a function

seq(from = 10, to = 23, by = 1) # count from 10 to 23 by 1

## [1] 10 11 12 13 14 15 16 17 18 19 20 21 22 23

seq(from = 10, to = 23) # equivalent

## [1] 10 11 12 13 14 15 16 17 18 19 20 21 22 23

seq(10, 23) # equivalent

## [1] 10 11 12 13 14 15 16 17 18 19 20 21 22 23

seq(10, -3) # count from 10 to -3

## [1] 10 9 8 7 6 5 4 3 2 1 0 -1 -2 -3

seq(from = -1, to = 11, by = 3) # count from -1 to 11 by 3

## [1] -1 2 5 8 11

#Note, you have all of the tools to complete homework 1

# Make your first data frame

* The most comfortable and familiar class/data type for many of you will be data.frame
* You can think of these as essentially Excel spreadsheets with rows (usually subjects or observations) and columns (usually variables) data.frames are somewhat advanced objects in R; we will start with simpler objects;

# Combine x and y vectors by column into a data frame and assign it to an object called 'df'  
df <- data.frame(x, y, z)  
df # Look at df

## x y z  
## 1 2 7 University of Michigan  
## 2 3 7 School of Public Health  
## 3 4 7 Epidemiology  
## 4 1 7 John Snow

df$x # Use $ to call up columns within data frames

## [1] 2 3 4 1

df$y

## [1] 7 7 7 7

# Index (look around) inside the data frame based on the position

#Index based on position  
df[3, 1] # structure: df[rows,columns]

## [1] 4

df[4, ]

## x y z  
## 4 1 7 John Snow

df[, 1]

## [1] 2 3 4 1

df[c(1, 2, 3), ] # Read your R code from the inside out. Start with the innermost set of parentheses.

## x y z  
## 1 2 7 University of Michigan  
## 2 3 7 School of Public Health  
## 3 4 7 Epidemiology

df[c(3, 1, 3), ]

## x y z  
## 3 4 7 Epidemiology  
## 1 2 7 University of Michigan  
## 3.1 4 7 Epidemiology

df[c(FALSE, TRUE, TRUE, FALSE), ]

## x y z  
## 2 3 7 School of Public Health  
## 3 4 7 Epidemiology

df[c(T, F, F, T), ]

## x y z  
## 1 2 7 University of Michigan  
## 4 1 7 John Snow

#Index based on value  
df$x>2 #Find a logical vector (True/False) of the rows that meet the value of interest (in this case x>2)

## [1] FALSE TRUE TRUE FALSE

df[, 1] > 2 #Same as above

## [1] FALSE TRUE TRUE FALSE

which(df[,1]>2) #Find rows that meet the criteria, similar to line above

## [1] 2 3

df[df$x>2,] #Now show the rows that are True

## x y z  
## 2 3 7 School of Public Health  
## 3 4 7 Epidemiology

df[df[, 1] > 2, ] #Same as above

## x y z  
## 2 3 7 School of Public Health  
## 3 4 7 Epidemiology

# Perform calculations on the data frame

sum(df[, 1])

## [1] 10

sum(x)

## [1] 10

# Recode a value in the data frame based on the position

df

## x y z  
## 1 2 7 University of Michigan  
## 2 3 7 School of Public Health  
## 3 4 7 Epidemiology  
## 4 1 7 John Snow

df[3, 2] <- 5 # recode a value  
#What do you expect?  
df #What do you get?

## x y z  
## 1 2 7 University of Michigan  
## 2 3 7 School of Public Health  
## 3 4 5 Epidemiology  
## 4 1 7 John Snow

#Do they match?

## Common new R users frustrations

1. Different versions of software
   * RStudio Cloud solves this
2. Working directory problems: trying to read files that R “can’t find”
   * RStudio Cloud solves this and so does RStudio Projects
3. Data type problems (is that a string or a number?)
   * discussed throughout
4. Typos (R is **case sensitive**, x and X are different)
   * RStudio helps with “tab completion”
   * discussed throughout
5. R often does not include any error/warning messages. Need to train yourself in 1) What do I expect?, 2) What do I get?, and 3) Do they match?

# Remember to save your R script!

# To exit R

q()