R02: R Basics

Meike Niederhausen and Nicky Wakim

2024-10-07

We will open RStudio on our computer (not R!)

1.1.2 Using R via RStudio

Recall our car analogy from earlier. Much as we don't drive a car by interacting directly with the engine but rather by interacting with elements on the car's dashboard, we won't be using R directly but rather we will use RStudio's interface. After you install R and RStudio on your computer, you'll have two new *programs* (also called *applications*) you can open. We'll always work in RStudio and not in the R application. Figure 1.2 shows what icon you should be clicking on your computer.

R: Do not open this	RStudio: Open this
	

FIGURE 1.2: Icons of R versus RStudio on your computer.

Modern Dive

RStudio anatomy

BuzzR

RStudio anatomy

https://buzzrbeeline.blog/

Emma Rand

Script file

Write code here

To run code put your cursor on the line and click the run button

Edit to correct errors

⇒ record of commands that worked

Save scripts with the **.R** extension

⇒ syntax will be highlighted

⇒ good practice

`<-` is the assignment operator

⇒ puts what is on the right in to the object on the left

⇒ Assign results if you want to use them again

Console

When you click run, code is sent to the console and executed

`>` is the prompt

⇒ do not type it

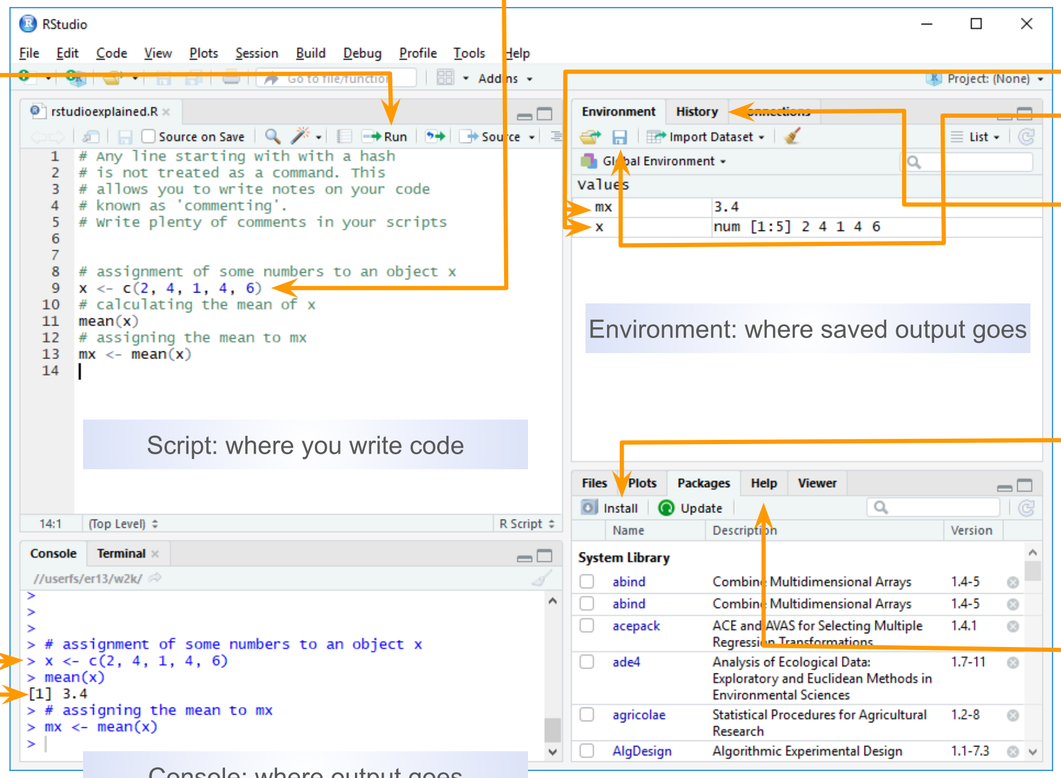
⇒ appears when R is ready for next command

Command output goes here by default

⇒ output is in a different colour

⇒ `[1]` indicates 3.4 is the first element of the output

⇒ many commands will not have output, the prompt just reappears



Environment

Name objects by assignment to use them again

All the objects you created in your session

Saving the environment saves all the objects, but not the code with a **.RData** extension

History

A history of every command you sent to the console, mistakes included.

File can be saved but usually you just need the script

Packages

Many functions come with R

A huge amount of extra functionality is available in packages

Packages can be installed by clicking the Install button

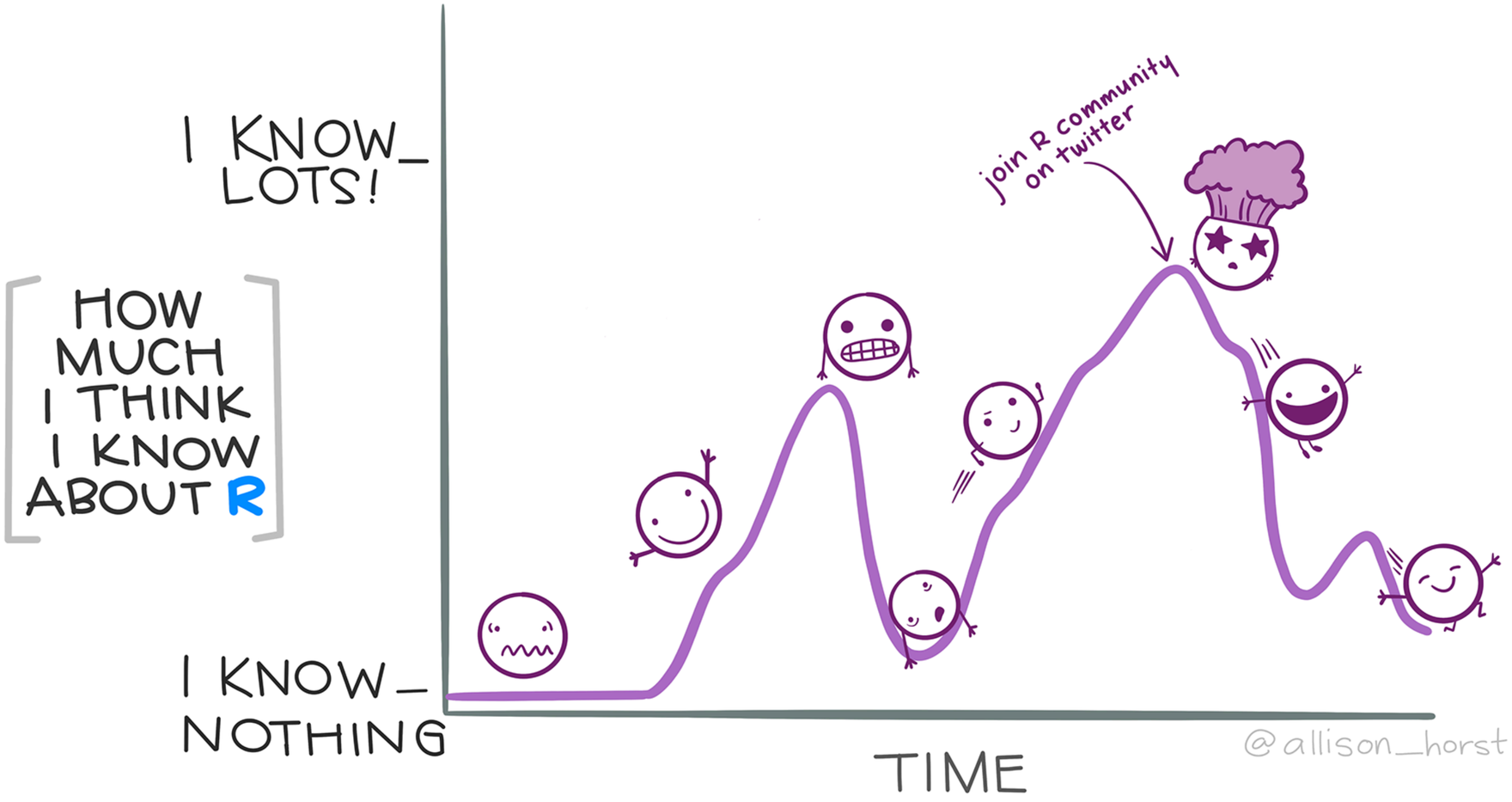
Help

Access to manual pages for all installed packages

Plots

Figure output appears here

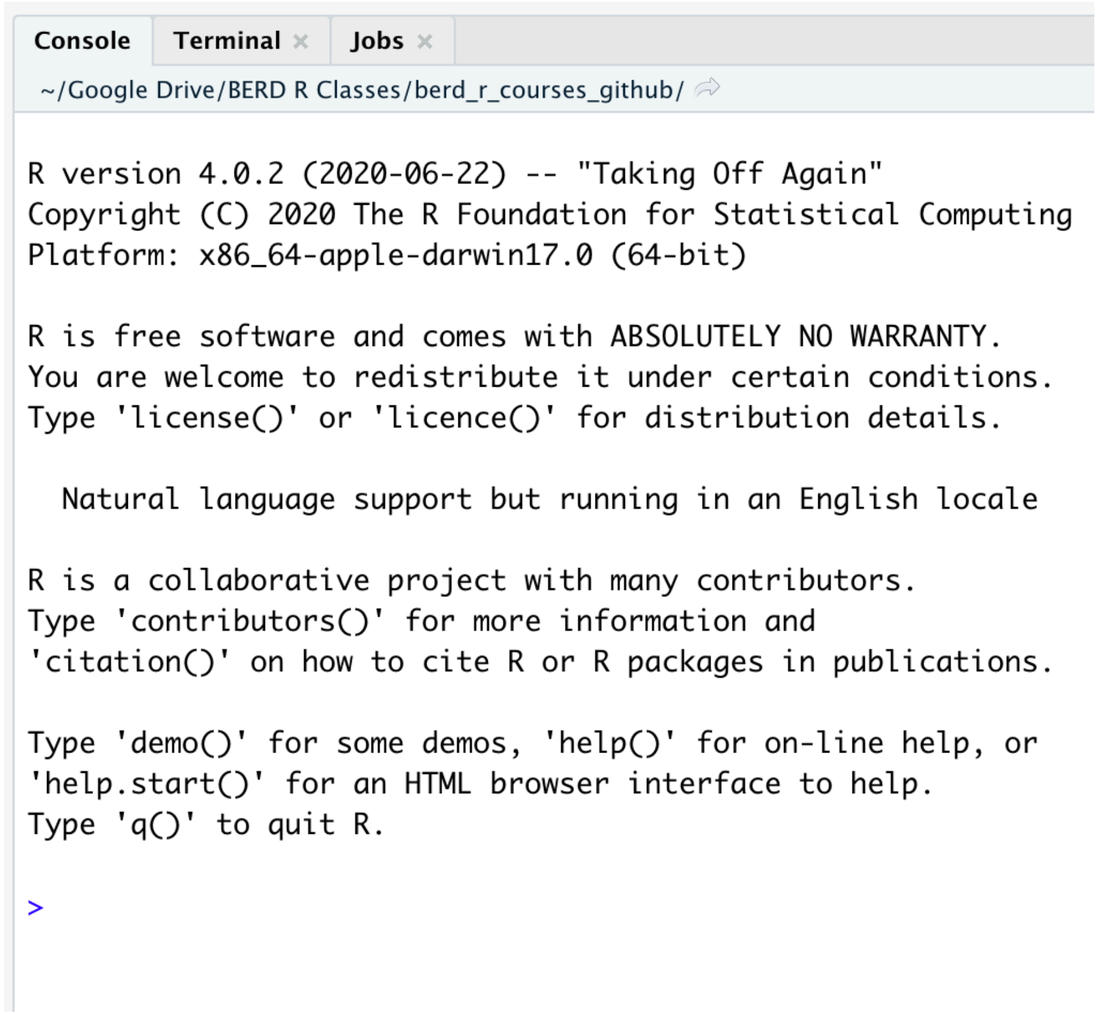
Let's code! R Basics



Artwork by @allison_horst

Coding in the console

When you first open R, the console should be empty.



Typing and executing code in the console

- Type code in the console (blue text)
- Press **return** to execute the code
- Output shown below in black

```
> 7
[1] 7
> 3 + 5
[1] 8
> "hello"
[1] "hello"
> # this is a comment, nothing happens
> # 5 - 8
> # separate multiple commands with ;
> 3 + 5; 4 + 8
[1] 8
[1] 12
>
```

We have an option of where to write our code

- We can use the console directly
  - BUT our work won't be saved

- We can also open up a file called a `.R` file
  - Hit the new document looking button, then click `R script`
  - Now we can type in the R script
  - In the R script, we need to press **cmd + return** or **ctrl + return** to execute the code
  - Output will show up in the Console!

- Example by Nicky here!

Math calculations using R

- We can use R as a calculator!!
- Rules for order of operations are followed
- Spaces between numbers and characters are ignored

```
1 10^2
```

```
[1] 100
```

```
1 3 ^ 7
```

```
[1] 2187
```

```
1 6/9
```

```
[1] 0.6666667
```

```
1 9-43
```

```
[1] -34
```

```
1 4^3-2* 7+9 /2
```

```
[1] 54.5
```

The equation above is computed as

$$4^3 - (2 \cdot 7) + \frac{9}{2}$$

Variables (saved R objects)

Variables are used to store data, figures, model output, etc.

- Can assign a variable using either = or <–
  - **Using <– is preferable** for certain occasions
  - I usually just use = because less typing hehe

Assign just one value:

```
1 x = 5
2 x
```

```
[1] 5
```

```
1 x <- 5
2 x
```

```
[1] 5
```

Assign a **vector** of values

- Consecutive integers using :

```
1 a <- 3:10
2 a
```

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

- **Concatenate** a string of numbers

```
1 b <- c(5, 12, 2, 100, 8)
2 b
```

```
[1]   5  12   2 100   8
```

Let’s try it out!

- Create a new variable `y` that is assigned the value of 8
- Create a new variable `c` that is assigned the vector of values 15 through 20
- Create a new variable `d` that is assigned the vector of values 16 through 19 and 22.

- Did you notice anything in the `Environment` section of Rstudio?

Doing math with variables

Math using variables with just one value

```
1 x <- 5
2 x
```

```
[1] 5
```

```
1 x + 3
```

```
[1] 8
```

```
1 y <- x^2
2 y
```

```
[1] 25
```

Math on vectors of values: **element-wise** computation

```
1 a <- 3:6
2 a
```

```
[1] 3 4 5 6
```

```
1 a+2; a*3
```

```
[1] 5 6 7 8
[1]  9 12 15 18
```

```
1 a*a
```

```
[1]  9 16 25 36
```

Let’s try it out!

- Use the variable name y to find the addition of y and 5
- Add 5 to the vector c

Variables can include text (characters)

```
1 hi <- "hello"
2 hi
```

```
[1] "hello"
```

```
1 greetings <- c("Guten Tag", "Hola", hi)
2 greetings
```

```
[1] "Guten Tag" "Hola"      "hello"
```

Using functions

- `mean()` is an example of a function
- functions have "arguments" that can be specified within the `()`
- `?mean` in console will show help file for `mean()`

Function arguments specified by name:

```
1 mean(x = 1:4)
```

```
[1] 2.5
```

```
1 seq(from = 1, to = 12, by = 3)
```

```
[1]  1  4  7 10
```

```
1 seq(by = 3, to = 12, from = 1)
```

```
[1]  1  4  7 10
```

Function arguments not specified, but listed in order:

```
1 mean(1:4)
```

```
[1] 2.5
```

```
1 seq(1, 12, 3)
```

```
[1]  1  4  7 10
```

Now let's use some functions for summary statistics

- We will calculate the mean for `c`
- Let's also calculate the standard deviation for `c`
  - Recall, our function is `sd()`
  - Use `?sd` in the console to identify the arguments for `c`

- If you have more time, you can try to calculate the median and IQR for `c`

Getting help with R

There are many ways to get help when you are stuck

- Use the `?` in front of the function name to get more information!
  - Usually if I need help with the arguments for a function

- Google or go to stackoverflow.com
  - Often when I Google, I get redirected to something like stackoverflow
  - For example, let's say my `mean` function was outputting `NA`. I would Google something like "keep getting NA for mean in R" Then end up here

- I can also go to my favoriate AI tool to get help
  - This is most useful for getting code started if it's complicated (we're not really at that level yet)
  - I asked ChatGPT "can you give me the code for calculating the mean in R"
    - This is what I got
  - For code generation, it gives you WAY too much
  - I also asked ChatGPT "Why is the mean function in R giving me an NA?" (in above link)

More on AI usage

- In the syllabus
- If you cannot trace code back to the class notes, then do NOT use it!
  - There's different coding practices and functions out there
  - I'm giving you a specific set of tools that will serve as a good introduction
  - You should be able to explain all your code and work