An Introduction to Programming in R (Part 1)

Philipp Buschmann

Hochschule Fresenius - Market Research and Empirical Research Methods

Winter term 2024

Contents

1. Organization
2. Why to learn R [cf. H22 chap. 2]
3. Installing R and R-Studio [cf. H22 chap. 3]
4. Our first R-codes [cf. HB22 chap. 5]

Schedule

- Sessions: Tuesdays 7:30 - 9:00 a.m.
- Session dates: March 18, ~~April 1~~, April 15, May 6, May 13, May 20, June 3, June 17.
- Exam? (20 points of a 90 points marketing research exam)

References

- The slides are based on the lecture notes *Analyzing Data with R – An Introduction* from Prof. Dr. Stephan Huber.
- I refer to the lecture notes using the shortcut [H22].
- You can find the slides and the lecture notes in *ILIAS*.
- [H22] goes beyond the scope of this course but it is very good for additional reading.
- Literature references help you to find the relevant chapters of [H22].
- The contents discussed in the weekly sessions are sufficient for passing the exam successfully!

Aims

- **Practical** introduction to programming in R.
- Writing **simple** scripts and functions in R.

→ We focus on applications - not on theoretical computer science or statistics!

Contents

1. Organization
2. Why to learn R [cf. H22 chap. 2]
3. Installing R and R-Studio [cf. H22 chap. 3]
4. Our first R-codes [cf. HB22 chap. 5]

Contents

1. Organization
2. Why to learn R [cf. H22 chap. 2]
3. Installing R and R-Studio [cf. H22 chap. 3]
4. Our first R-codes [cf. HB22 chap. 5]

Motivation

- R is an artist! (`https://r-graph-gallery.com/`)
- R is an employment insurance.
- Excel is bad!
- R is good, proprietary software is bad!
- R is big!
- R is the future!

Contents

1. Organization
2. Why to learn R [cf. H22 chap. 2]
3. Installing R and R-Studio [cf. H22 chap. 3]
4. Our first R-codes [cf. HB22 chap. 5]

What is R-Studio?

- The term R does not really refer to a specific application on your computer.
- R rather refers to the underlying programming language. You can use this language through lots of different user interfaces.
- The download of R includes a rather rudimentary graphical user interface (GUI). It is limited to what is necessary!
- There are many text editors or IDEs (integrated development environments) which simplify programming with R.

→ We use the IDE R-Studio!

Installation

- You should install R before installing R-Studio!
- You find a detailed guide in [H22] chap. 3 for downloading and installing R and R-Studio on Windows, Linux, and Mac
  → Homework!

Contents

1. Organization
2. Why to learn R [cf. H22 chap. 2]
3. Installing R and R-Studio [cf. H22 chap. 3]
4. Our first R-codes [cf. HB22 chap. 5]

Using R as a pocket calculator

```
15 + 27
12.5 - 3.7
333.3 / 3
12 * 7
7 * (2 + 4)
7 * 2 + 4
6 / 2 * 3
6 / (2 * 3)
3^2
16^0.5
```

Task 1 - Using R as a pocket calculator

Write the code to compute the following mathematical statements in R:

a) $\sqrt{1+\left(\frac{5}{2}\right)^3}$

b) $\frac{9}{3\cdot3}$

c) $\frac{9}{3}\cdot3$

d) $6\cdot\frac{3}{2}$

e) $6\cdot3\cdot\frac{1}{2}$

f) $\frac{\sqrt{36}}{\sqrt{3+1}}$

g) $\frac{\sqrt{36}}{\sqrt{3}+1}$

h) $2^5\cdot3^5$

Storing a number as a variable

- One of the most important things to be able to do in R is to store information in variables or so-called objects.
- In a statistical analysis in R all of your data will be stored in variables.
- To create a variable and assign a value to this variable we use the assignment operator:

```
<-
```

- We create a variable with the name `sales` and a assign value of 350 to `sales`:

```
sales <- 350
```

→ This code does not print anything to the console. To print the value of sales one simply executes the code:

```
sales
```

The name of a variable

A variable name may consist of letters, numbers, the dot and the underline sign. The name is not allowed to begin with a number. If it begins with the underline sign or a dot, it is impossible to take a number for the second sign.

→ For a detailed description of conventions for naming variables in R see [H22] chap. 5.3.3.

Task 2 - Calculating with variables

Write an R-code that creates the variables $y = 10$ and $z = -10$. Then, the code should perform the following calculations:

a) $y + z$

b) $y \cdot z$

c) y^z

d) $\sqrt{z^2}$

e) $2 \cdot z$

f) $3 \cdot \frac{z}{y}$

g) $y^3 + y^2 + y$

Task 3 - Calculating with variables

Assume that the following code is run in R. What is printed in line 3, 4, 7, and 8 to the console?

```
x <- 5
y <- 6
x + 1          # line 3
y              # line 4
y <- x + y
z <- y
2*y            # line 7
z              # line 8
```

Task 4 - Calculating with variables

Assume that the following code is run in R. What is printed in line 3, 6, 7, 8 and 10 to the console?

```
x <- 2
y <- 1
z <- 3        # line 3
z <- 4
y <- z * x
y             # line 6
z / (2 * 2)   # line 7
y <- 2 * y    # line 8
z <- z + 2
z / 3 * y     # line 10
```

Using functions to do calculations

- So far our calculations based on standard arithmetic operators such as +, -, * etc.
- To do more advanced calculations, you’re going to need to start using functions.
- When we use a function to do something, we generally refer to this as calling the function.
- The values that we type into the function (there can be more than one) are referred to as the arguments of that function.

Some functional expressions

```
sqrt(16)
exp(1)
sin(0.5 * pi)
log(exp(5))
log2(2^5)
abs(-13)
round(3.1415)
round(3.1415, 2)
round(2.967, 2)
floor(2.99)
ceiling(2.1)
```

Functions with multiple arguments (1/2)

Example:

```
round(3.1415, 2)
```

- The first argument is the number that needs to be rounded.
- The second argument is the number of decimal places that the first number should be rounded to.
- Here: Easy to remember which argument comes first and which one comes second → For more complicated functions this is not easy!
- Therefore, most functions make use of argument names.
- Here: `x` is the variable name of the number to be rounded; `digits` the name of number of decimal places.

We can specify the arguments to the function by name:

```
round(x = 3.1415, digits = 2)
```

Functions with multiple arguments (2/2)

- Specifying the arguments by name involves more typing:

→ However, it is easier to read!

→ When specifying the arguments using their names, it does not matter in which order you type them.

```
round(x = 3.1415, digits = 2)
round(digits = 2, x = 3.1415)
round(3.1415, 2)
```

```
  round(2, 3.1415)  # returns 2
```

Task 5 - Calculating with functions

Write the code to compute the following mathematical statements using R-functions:

a) $\log_2(4)$

b) $\cos(\frac{1}{2}\pi)$

c) $\frac{5 \cdot 4}{\ln(10)}$

d) e^5

e) $\sqrt{1 + \left(\frac{5}{2}\right)^3}$

f) $5 + |-5|$

g) $|-3 \cdot \frac{1}{2}|$

Task 6 - Calculating with functions

Write an R-Code to...

a) **calculate and store** (in a variable `x`) the result of the expression:

$$\sqrt{8} \cdot \left(\frac{3}{4} + 7^{-2} \right) \cdot 9.$$

b) **round** the result (**three digits**) and store the result in an object entitled `result_rounded`.

Functions - Help!

- There are a lot of R functions, all of which have their own arguments.

  → Do we have to remember all arguments? No!

- The R help documentation helps to call functions correctly.
- Among others, the help documentation specifies the arguments and the return value of a function.
- You call the help documentation by

```
    help(name_of_function)
```

- . . . or alternatively by

```
    ?name_of_function
```

- Example:

```
    help(round)
    ?round
```