1. Introduction to programming in R

Martina Danielova Zaharieva

CUNEF Universidad

Academic year 2022/23

Overview

General information

- Objective: Learn how to use R for business and data science
- Prerequisites:
  1. Linear algebra (school level)
  2. Statistics (school level)
- Laptop (any standard operating system), with Wifi internet access

Literature

Essential: W. John Braun and Duncan J.Murdoch
A First Course in Statistical Programming with R
Third Edition. Cambridge University Press.

Additional: Wickham, H. and Grolemund G.
R for Data Science, O'Reilly, 2016.

→ see the syllabus for more

Schedule

1. Introduction to the programming environment in R: help, libraries and packages, R-objects
2. Basic concepts of the R Language: operators and functions; data import and export; data manipulation and transformation
3. Programming in R: control structures (*if-else, for, while, repeat*), loops and *apply* family functions, custom functions
4. Advanced programming in `tidyverse`
5. Advanced visualization techniques: the `ggplot2` package

Evaluation

- First midterm exam: 20 % of final grade
- Second midterm exam: 20% of final grade
- Final exam: 60% of final grade with a minimum grade 5.0
- Preliminary dates will be announced soon
- A continuous class participation is expected!

About R

- S is an object-oriented statistical computing language
- The language S is implemented as S-Plus (commercial) and R (OpenSource)
- The differences between S-Plus and R are minimal
- Similar programming languages: Matlab, GAUSS, Julia
- R is available for Windows, Linux and MacOS
- Internet site: `www.r-project.org`
- Comprehensive R Archive Network : `cran.at.r-project.org`

Installation (for Windows)

- Open `www.r-project.org` in your browser
- In the left menu, choose CRAN (or click "download R")
- Choose a mirror (e.g. Spain)
- Choose you operating system (e.g. Windows)
- Choose `base`
- Download the newest version of R
- Execute `R-3.4.x-win.exe` and follow the instructions
- Start R

Command window

Command window (R Console) Prompt: `>` You can input commands and execute them
(by pressing the RETURN key)

Examples

```
> 1+1
> 1+1 # This is a comment:  1+1
> (1+2)*3
> (5/3)^4.5
> 5+2; 7+3; 2*5
```

Command window

Concatenation: `c()` Assignment operator: `<-` or `=`

Examples

```
> c(1,4,7)
> a <- c(1,4,7)
> print(a)
> a
> A
> b <- c(1,a,3)
> b
> mean(b)
```

Command window

Quitting

- Quit R by the command `q()`
- In general, do *not* save your workspace

Editors

- Long computations should not be done interactively in the command window
- Use an editor to write a program and then execute it in R
- There is a built-in editor in R: File – New script
- External editors:
  - R-Studio, `http://www.rstudio.com/ide/download/`
  - Tinn-R, `http://sciviews.org/Tinn-R/`
  - Notepad++: `http://www.notepad-plus-plus.org/`
- We will focus on **R-Studio**

RStudio Cloud

- Open `https://www.rstudio.com/products/cloud/`
- Go to GET STARTED FOR FREE and register (sign up)
- Start a new project

Script in R-Studio

- Instead of typing in commands in the console, create scripts and save your work
- You will be able to comment, edit and recompile your code
- In R-Studio: top left corner R script
- Type a few lines of commands, e.g.

```
a <- c(1,4,7)
mean(a)
mean(a)^2
```

- Execute a single line by pressing Ctrl-Enter
- Execute multiple lines by marking them and then pressing Ctrl-Enter
- Save the script, quit R, restart R, open the script and execute it

Style Guide (1)

- To create an R script follow the Style Guide
- Variable names and commands are case-sensitive
- Names may include letters, numbers, dots
- Use a consistent style (e.g Google's R style guide or at least the recommendations of the next slides, based on "Advanced R" by Hadley Wickham)

Style Guide (2)

- File names should be meaningful and end in .R
- Variable and function names should be lowercase. Use an underscore to separate words within a name. Generally, variable names should be nouns and function names should be verbs. Strive for names that are concise and meaningful (this is not easy!).
- Avoid using names of existing functions and variables.
- Place spaces around all infix operators (=, +, -, <-, etc.).
- Always put a space after a comma, and never before (just like in regular English).

Style Guide (3)

- Do not place spaces around code in parentheses or square brackets (unless there is a comma).
- An opening curly brace should never go on its own line and should always be followed by a new line. A closing curly brace should always go on its own line, unless it is followed by else.
- Strive to limit your code to 80 characters per line.
- Use <-, not =, for assignment.
- Comment your code. Each line of a comment should begin with the comment symbol and a single space: #.
  Comments should explain the why, not the what.

Help

- To obtain details about a command, type
  `?command`
  or
  `help(command)`
- Example: `?mean` or `help(mean)`
- Start the "help center": `help.start()`
- In R-Studio: bottom right corner
- Task Views on CRAN
- R Journal on CRAN

Packages

- One of the strengths of R is the large and growing collection of packages that can be downloaded from CRAN (or installed off-line)
- Offline installation: `install.packages("packagename")`
- Installed packages are activated by `library(packagename)`
- Help about packages: `library(help=packagename)`
- In R-Studio: bottom right corner
- Install, activate and look into the help of the package MASS

R objects

- R is object oriented
- An object can be anything: scalar, vector, matrix, string, table, factor, list, data frame, regression results, ...
- The object type determines how some commands work (e.g. `plot`, `summary`)
- Every object has a unique name
- List of all objects: `ls()`
- Delete (**rm**ove) objects: `rm()`

R objects

Examples

```
> x <- c("A","B","C")
> class(x)
> y <- c(1,2,5)
> class(y)
> ls()
> rm(x)
> ls()
```

R objects

Examples

```
> x <- c("A","B","C")
> class(x)
> y <- c(1,2,5)
> class(y)
> ls()
> rm(x)
> ls()
```

$\longrightarrow$ Let's do some exercises

Exercise 1

Install the packages `MASS`, `foreign`, `xlsx`, `rgl`.

Exercise 2

The current working directory (where R reads and writes files) can be found by the command `getwd()`. Find your current working directory.

Exercise 3

Use the command `setwd("c:/path")` to change the working directory to drive `c:` and path `/path`. Note that the path name is structured by slashes (/), *not* backslashes (\). Change the working directory to `c:/temp` and check if the change has been successful.[1]

[1] The working directory can also be changed via the menu: Tools – Options ...

Exercise 4

Open a new script file. Type the commands to perform the following assignments:

$$
\begin{aligned}
a &= \frac{3 \cdot (4+9)}{8 - 12.5} \\
b &= (1, 4, 1999, 2011) \\
d &= 2\pi \\
e &= a + d
\end{aligned}
$$

Save the script and quit R.

Exercise 5

Start R and re-open the script. Mark all lines (Ctrl-A) and execute them (Ctrl-R). Print a, b, d, e. Why is the variable name c not used?

R Markdown

- Formatting tool for creating of scientific reports
- Includes embedded R code chunks
- Extension . Rmd
- Internet site: `https://rmarkdown.rstudio.com/`
- Cheat Sheet: `https://www.rstudio.com/wp-content/uploads/2015/02/rmarkdown-cheatsheet.pdf`

Exercise 6

- Create an HTML format of Markdown file (see cheat sheet) called “My first Markdown”
- Use the “Knit” button
- Save your Markdown file
- Create your own Markdown report for the solution of Exercises 4 – 5
- Try the option `{r echo=FALSE}`