Statistical Computing with R

Lecture 1: statistical computing; getting started with R; basic operations; types of objects in R; vectors

Mirko Signorelli
🏠: mirkosignorelli.github.io
✉: statcompr [at] gmail.com

Mathematical Institute
Leiden University

Master in Statistics and Data Science (2023-2024)

Universiteit Leiden

Statistical computing

More about R

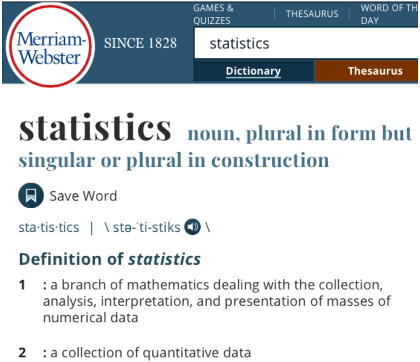
Installing R and RStudio

Using R as a calculator

Types of objects

Vectors

Statistics: science, or art?



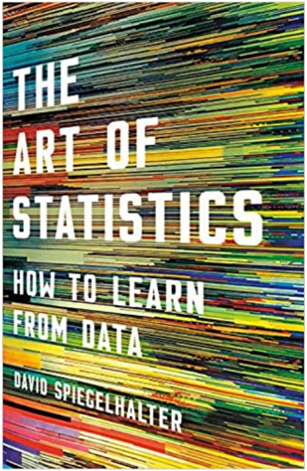
statistics noun, plural in form but singular or plural in construction

sta·tis·tics | \ stə-ˈti-stiks \

Definition of *statistics*

1 : a branch of mathematics dealing with the collection, analysis, interpretation, and presentation of masses of numerical data

2 : a collection of quantitative data



Statistical computing

- Statisticians / data scientists generate information from data
- To do this, they need to perform computations
- Nowadays, most computations are too difficult (or tedious) to be performed by a human
- Luckily, we can let computers do the job for us!

Computing power

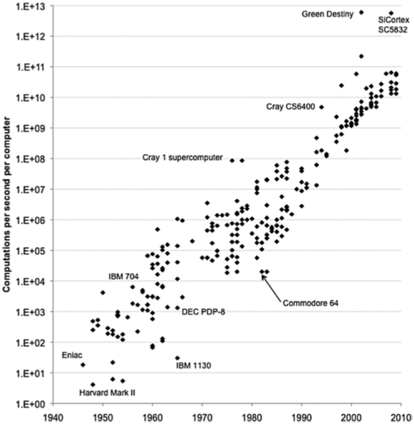


Figure 1: Exponentially increasing computational capacity over time (computations per second) – Koomey, Berard, Sanchez, and Wong (2011). Source: https://ourworldindata.org/technological-progress

In a nutshell. . .

Statistical computing = set of computational, graphical and numerical approaches that can be used to perform statistical analyses

What this course is about:

- learning how to “instruct” computers to perform statistical analyses (real & simulated data)
- we will do this using `R` (and `RStudio`)

Statistical computing

More about R

Installing R and RStudio

Using R as a calculator

Types of objects

Vectors

Statistical computing before R

For decades, statisticians mostly relied on proprietary software for their computations and data analysis. Some examples:

- SPSS (SPSS Inc., 1968. Now: IBM)
- S (Bell Labs, 1976)
- SAS (SAS Institute, 1976)
- Stata (StataCorp, 1985)

Major downsides:

- license costs (often high)
- slow implementation of new methodology

The R revolution

- R: programming language for statistics and data science
- Officially launched in 1995 (v 1.0.0 released in 2000)
- Open-source version of S

Major breakthroughs for statisticians:

- free software (no licence costs!)
- collaborative project - anyone can contribute new packages

R nowadays

- R started as a language developed and used by academics
- Over the years, it increasingly made its way in industry
- Nowadays: R & Python are the two programming languages more widely used by statisticians and data scientists
- Big, world-wide community of R users:
  1. extensive material (courses, tutorials, blogs, . . .) about R available online, usually for free
  2. many in-person and virtual events (UseR! and eRum conferences, local R users meetings, . . .)

Statistical computing

More about R

Installing R and RStudio

Using R as a calculator

Types of objects

Vectors

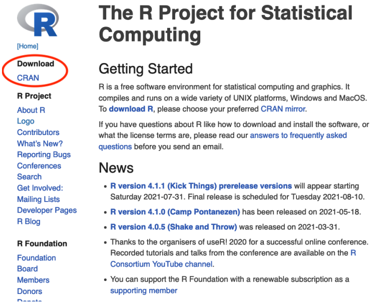
Foreword

- The slides in this section show you the main steps involved in the installation of `R` and RStudio
- During the lecture, I will browse quickly through them
- During the coding session, you can try to install `R` and RStudio on your laptops and get help from the TAs if needed

Installing R

To install R:

1. Go to the R Project website: https://www.r-project.org/
2. Select *CRAN* from the left menu
3. Choose your favourite "CRAN mirror" to download from
4. Select your operating system (OS)



[Home]

Download

CRAN

R Project

About R
Logo
Contributors
What's New?
Reporting Bugs
Conferences
Search
Get Involved: Mailing Lists
Developer Pages
R Blog

R Foundation

Foundation
Board
Members
Donors

The R Project for Statistical Computing

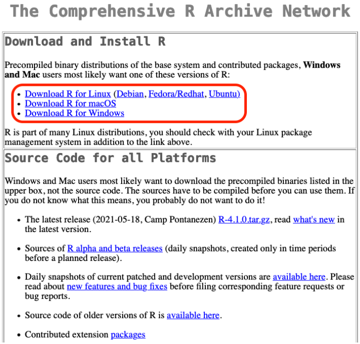
Getting Started

R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS. To **download R**, please choose your preferred CRAN mirror.

If you have questions about R like how to download and install the software, or what the license terms are, please read our answers to frequently asked questions before you send an email.

News

- **R version 4.1.1 (Kick Things) prerelease versions** will appear starting Saturday 2021-07-31. Final release is scheduled for Tuesday 2021-08-10.
- **R version 4.1.0 (Camp Pontanezen)** has been released on 2021-05-18.
- **R version 4.0.5 (Shake and Throw)** was released on 2021-03-31.
- Thanks to the organisers of useR! 2020 for a successful online conference. Recorded tutorials and talks from the conference are available on the R Consortium YouTube channel.
- You can support the R Foundation with a renewable subscription as a supporting member



The Comprehensive R Archive Network

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- Download R for Linux (Debian, Fedora/Redhat, Ubuntu)
- Download R for macOS
- Download R for Windows

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (2021-05-18, Camp Pontanezen) R-4.1.0.tar.gz, read what's new in the latest version.
- Sources of R alpha and beta releases (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are available here. Please read about new features and bug fixes before filing corresponding feature requests or bug reports.
- Source code of older versions of R is available here.
- Contributed extension packages

Installing R (cont'd)

What to do next depends on your OS:

- Windows user: select the base package
- MacOS user:
  1. two versions depending on processor (arm64 or Intel)
  2. if you bought your laptop in the last 2 years, most likely it has an M1 / M2 chip $\Rightarrow$ install the arm64 package (but: the Intel version should work as well)
- Linux: different instructions depending on your Linux distribution

Irrespective of your OS: choose the latest release of R (4.3.1, or higher)

RStudio

- RStudio is an Integrated Development Environment (IDE) for R
- It makes it easier to interface yourself with R, and comes with extra functionalities

⇒ we are going to code in R through RStudio
rather than directly in R

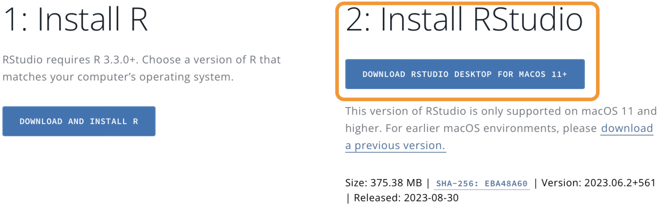
To keep in mind:

- R is a community project, mostly run by volunteers
- RStudio is a product developed by Posit, a business that offers some of its products for free

Installing RStudio

To install RStudio:

1. Go to https://posit.co
2. Click on DOWNLOAD RSTUDIO (top menu)
3. In the next webpage, scroll to the RStudio Desktop box and click on DOWNLOAD
4. The next page should suggest you the right installer for your OS



1: Install R

RStudio requires R 3.3.0+. Choose a version of R that matches your computer's operating system.

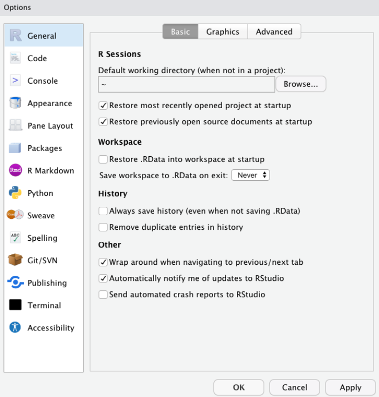
2: Install RStudio

This version of RStudio is only supported on macOS 11 and higher. For earlier macOS environments, please download a previous version.

Size: 375.38 MB | SHA-256: EBA48A60 | Version: 2023.06.2+561 | Released: 2023-08-30

Changing default settings

- To edit RStudio's default settings, select "Preferences" in RStudio's menu:



R Sessions

Default working directory (when not in a project):

~

☑ Restore most recently opened project at startup

☑ Restore previously open source documents at startup

Workspace

□ Restore .RData into workspace at startup

Save workspace to .RData on exit: Never

History

□ Always save history (even when not saving .RData)

□ Remove duplicate entries in history

Other

☑ Wrap around when navigating to previous/next tab

☑ Automatically notify me of updates to RStudio

□ Send automated crash reports to RStudio

Installation problems

If you encounter installation problems: ask the TAs to help you fixing them during the Coding Session!

Statistical computing

More about R

Installing R and RStudio

Using R as a calculator

Types of objects

Vectors

Finding your way in RStudio

After having installed both `R` and `RStudio`, open `RStudio`

1. Type `5 + 2` in the Console
2. Type `?mean` in the Console
3. In the menu, click on: File > New File > Rscript
4. Type `5 + 2` in the newly created script, then click Ctrl+Enter[1]
5. Alternatively: select `5 + 2` and click on ‘Run’
6. To save the R script: File > Save / Save as…

[1] MacOS users: both control+Enter and command+Enter work!

Basic operations

- Sum: +, difference: −, product: ∗, division: /
- Numeric constants: $\overset{\pi}{\texttt{pi}}$, $\overset{e}{\texttt{exp}}$`(power_value)`

```
25 + 32
```

```
## [1] 57
```

```
12*4 - 50/10
```

```
## [1] 43
```

```
pi
```

```
## [1] 3.141593
```

```
exp(1); exp(2)
```

```
## [1] 2.718282
## [1] 7.389056
```

Basic operations (cont'd)

- Powers: ^ or **
- Integer part of a division: %/% 整数
- Remainder of a division: %% 余数

```
5^2; 5**3
```

```
## [1] 25
```

```
## [1] 125
```

```
5^(1/2)
```

```
## [1] 2.236068
```

```
10 %/% 3
```

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

```
10 %% 3
```

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

Parentheses

- Parentheses / brackets in expressions: use ( ) and/or { }
- ⚠ Don't use [ ]! (subsetting operator) ⚠

```
(24/3 - 7)*3
```

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

```
((3+1)*2 - 1)/2
```

```
## [1] 3.5
```

```
{(3+1)*2 - 1}/2
```

```
## [1] 3.5
```

- Try using [(3+1)*2−1]/2: does it work?

Assignment operators

- Assignment operators are used to create (or assign a value to) an R object, or to define a function
- Assignment operators in R: = or <- or ->

```
# create a scalar
x1 = 5
x2 <- -3
7 + 5 -> x3
x1; x2; x3
```

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

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

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

# and ;

- In the previous slide(s) I used # and ;:

  1. # is used to add comments to your scripts
  2. comments are not executed by R. You can use them to "comment" your code
  3. in R 1 line = 1 command, but: ; can be used to put multiple commands on a single line
  4. you won't usually need to use ;, but in my slides I will often use it to save vertical space ☺

# and ; (cont'd)

```
# this is a first comment
5 + 3 # this is a second one
```

```
## [1] 8
```

```
### this is a comment too! :)
# notice that the following are the same:
# 1) code on 3 lines
x = 3
x = x - 5
x
```

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

```
# 2) all code on a single line
x = 3; x = x - 5; x
```

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

Assignment operators (cont'd)

- Further examples:

```
# create a vector with elements 4 5 6 7 8 9 10
y = 4:10
y
```

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

```
# define a function
f = function(x) sum(x) / length(x)
f(y)
```

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

Now try it yourself!

Exercises

1. Compute the value of the expression $\frac{3(x-2)}{4} + 1$ when $x \in \{1, 2, 3\}$
2. Compute the circumference and area of a circle of radius 5
3. Maria deposits 4000 € into her savings account, which offers a 1.5% interest rate. How much interest will she have accumulated after 3 years?

⚠ Solutions in the next slide. Try to solve the problems by yourself before checking the answers! ⚠

Solutions

```
# Ex 1
x = 1:3
3*(x-2)/4 + 1
```

```
## [1] 0.25 1.00 1.75
```

```
# Ex 2
r = 5
2*pi*r
```

```
## [1] 31.41593
```

```
pi*(r^2)
```

```
## [1] 78.53982
```

```
# Ex 3
4000*(1.015)^3 - 4000
```

```
## [1] 182.7135
```

Statistical computing

More about R

Installing R and RStudio

Using R as a calculator

Types of objects

Vectors

Object types in R

Most common types of objects in `R`:

- vectors
- matrices (and arrays)
- data frames (and tibbles)
- lists

Today: vectors
Next weeks: matrices, data frames and lists

Statistical computing

More about R

Installing R and RStudio

Using R as a calculator

Types of objects

Vectors

Vectors

A vector is a unidimensional list of elements, sequentially ordered.

$$v = (4 \ \text{apple} \ 5\pi)$$

- 4 is the first element of v
- apple is the second
- 5π is the third

Creating and subsetting a vector

- c( ) creates a vector, commas separate its elements:

```
v = c(4, 'apple', 5*pi)
v
```

```
## [1] "4"                "apple"
## [3] "15.707963267949"
```

- [ ] allows to subset elements from the vector

```
v[3]
```

```
## [1] "15.707963267949"
```

position从1开始

```
v[c(1, 3)]
```

```
## [1] "4"                "15.707963267949"
```

```
v[-3]
```

去掉第3个

```
## [1] "4"      "apple"
```

Concatenating vectors

- `c( )` can also be used to concatenate multiple vectors

```r
v1 = c(3, 7, 14)
v2 = 1:4
c(v1, v2)
```

```
## [1]  3  7 14  1  2  3  4
```

Vector types

Most common types of vectors:

- logical: a vector whose only elements are `TRUE` / `FALSE`
- numeric: a vector that contains **numbers**
- character: a vector that contains **strings** of text
- factor: a “formatted” character with restrictions on the possible values of its elements

Logical vectors

- A logical vector is a vector whose values can only be TRUE / FALSE
- You may find it useless now, but TRUE / FALSE will become quite useful when programming!

```
v1 = c(T, F, F, T, T)
v1[4]
```

```
## [1] TRUE
```

```
v2 = c(18, 23, 14, 42)
# which elements are > 25?
v2 > 25
```

```
## [1] FALSE FALSE FALSE  TRUE
```

```
# which elements are even?
v2 %% 2 == 0
```

```
## [1]  TRUE FALSE  TRUE  TRUE
```

Numeric vectors

- A numeric vector contains numbers

```
v1 = c(3, 7, 14)
v1
```

```
## [1]  3  7 14
```

```
v2 = 10:15
v2
```

```
## [1] 10 11 12 13 14 15
```

```
v3 = seq(20, 30, by = 2)
# type ?seq in the console to find out how seq( ) works!
v3
```

```
## [1] 20 22 24 26 28 30
```

```
c(v1, v3)
```

```
## [1]  3  7 14 20 22 24 26 28 30
```

Numeric vectors (cont'd)

```
v1 = c(3, 7, 14)
3*v1
```

```
## [1]  9 21 42
```

```
3*v1 - 15
```

```
## [1] -6  6 27
```

```
v1^2
```

```
## [1]   9  49 196
```

Characters

- A character vector contains strings of text
- ⚠ If you put together numbers and strings, the resulting vector will be of type character ⚠

```
v1 = c('Maria', 'Joost', 'Pedro')
v1
```

```
## [1] "Maria" "Joost" "Pedro"
```

```
v2 = 1:3
v3 = c(v1, v2)
is.numeric(v3)
```

```
## [1] FALSE
```

```
is.character(v3)
```

```
## [1] TRUE
```

Factors

- A factor is a particular type of character vector used to store categorical data
- Idea: if a variable X can take only a restricted number of entries (categories), it might be more efficient to store a character vector as a numeric vector
- In other words: a factor is a character vector that is stored as a numeric vector

```
v = c('Male', 'Female', 'Female', 'Male', 'Female')
v = factor(v)
v
```

```
## [1] Male   Female Female Male   Female
## Levels: Female Male
```

```
# factors can be easily converted into a numeric vector:
as.numeric(v)
```

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

Your turn

Exercises

1. Compute $\sum_{x=5}^{20} \frac{1}{x}$ (tip: check `?sum` out!)
2. Create a vector that contains all **odd** numbers between 0 and 26 using the ‘seq‘ function (type ‘?seq‘ in the console to find out more about ‘seq‘!)
3. Let $x = (3\ 4\ 5)$ and $y = (7\ 5\ 3)$. Compute $x - y$

Solutions

```r
# Ex 1
v1 = 5:20
sum(1/v1)
```

```
## [1] 1.514406
```

```r
# Ex 2
seq(1, 26, by = 2)
```

```
##  [1]  1  3  5  7  9 11 13 15 17 19 21 23 25
```

```r
# Ex 3
x = c(3, 4, 5)
y  =c(7, 5, 3)
x-y
```

```
## [1] -4 -1  2
```

Reminders

- Next week:
    1. no SCwR class
    2. free LaTeXworkshop, Wednesday 13/9, 10.00-12.00, Snellius 1.74
- Lecture 2: Wed. 20/9 (11-15), lecture 3: Fri 22/9 (9-13)