

# ICO Workshop R & RStudio

## Part 1

*Introduction to R and RStudio*

Sven De Maeyer & Tine van Daal

2nd - 4th July, 2024

1 / 43

## Overzicht

1. Some practical stuff --- ([click here](#))
2. What is R and why should you use R? --- ([click here](#))
3. Install R & RStudio --- ([click here](#))
4. RStudio interface --- ([click here](#))
5. The basics in R ([click here](#))
6. Installing and using packages --- ([click here](#))
7. Importing data --- ([click here](#))
8. Working with R-projects --- ([click here](#))

2 / 43

# About us

- University of Antwerp; Faculty of Social Sciences; dept. Training- en Education Sciences
- diehard users of R

## About Sven?

- *Research interests?* Psychometry, writing research, learning strategies, education for sustainable development, linguistics, ...
- *Main focus?* Comparative Judgement + Learning from comparisons

Blog: <https://svendemaeyer.netlify.app/>  
Twitter: @svawa  
github: <https://github.com/Sdemaeyer2>

## About Tine?

- *Research interests?* Comparative judgement, learning from complementary texts, assessment with technology, replication studies, data visualisation, ...
- *Main focus?* (Peer) assessment, learning (processes), educational technology, learning analytics

Twitter: @TinevanDaal  
github: <https://github.com/tvdaal>



3 / 43



4 / 43

# 1. Some practical stuff

5 / 43



## Materials

Various types of files:

- Rmd-files (for slides) (*.Rmd*)
- Qmd-files (for exercises) (*.Qmd*)
- R-scripts (as example) (*.R*)
- data sets (*.sav*, *.csv*, ...)
- html-version of slides and exercises (*.html*)

All these files can be found at the dedicated web-page: <https://icoworkshop-rstudio-2024.netlify.app/#quick-overview>

6 / 43

# At work

We will provide you with the necessary information to start coding yourself.

⚠️ *Make it interactive! Feel free to ask questions!*

Of course, we will also put you to work! Do you get stuck? Is R not working along?

⚠️ *Do not hesitate to ask our help!*

## 2. What is R and why should you use R?

# What is R?

Powerful tool for:

- Data Science
- Statistical analysis
- Statistical programming
- Data visualisation ...

A 'language'

→ We will have to learn some **vocabulary** and **grammar** ...



# Why R?

- Free AND open source!
- In constant development
- Thousands of "add-ins" (packages)
- Almost every algorithm and technique is implemented in R
- R is ahead of commercial statistical software
- Can facilitate reproducibility and open science practices
- Large community (Help will always be given ...)

# 3. Install R and RStudio

## Install R



<https://cran.r-project.org/>

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 (2021-08-10, Kick Things) [R-4.1.1.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](#)

**Questions About R**

- 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.

**What are R and CRAN?**

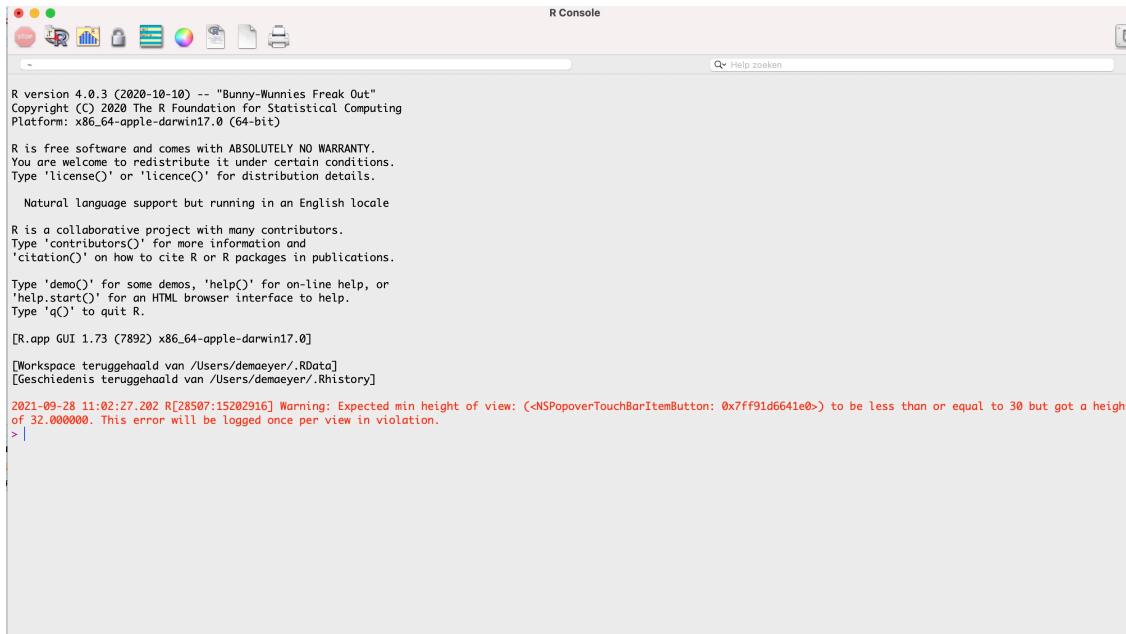
R is 'GNU S', a freely available language and environment for statistical computing and graphics which provides a wide variety of statistical and graphical techniques: linear and nonlinear modelling, statistical tests, time series analysis, classification, clustering, etc. Please consult the [R project homepage](#) for further information.

CRAN is a network of ftp and web servers around the world that store identical, up-to-date, versions of code and documentation for R. Please use the CRAN [mirror](#) nearest to you to minimize network load.

**Submitting to CRAN**

» *In case you haven't already installed R. Do it now! Are you stuck? Let us know.*

# A quick tour in R ...



```
R version 4.0.3 (2020-10-10) -- "Bunny-Munnies Freak Out"
Copyright (C) 2020 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin17.0 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[R.app GUI 1.73 (7892) x86_64-apple-darwin17.0]

[Workspace teruggehaald van /Users/demaeyer/.RData]
[Geschiedenis teruggehaald van /Users/demaeyer/.Rhistory]

2021-09-28 11:02:27.202 R[28507:15202916] Warning: Expected min height of view: (<NSPopoverTouchBarItemButton: 0x7ff91d6641e0>) to be less than or equal to 30 but got a height
of 32.000000. This error will be logged once per view in violation.
> |
```

13 / 43

## Install RStudio

Go to <https://posit.co/>

Click on *Download RStudio*

RStudio is now Posit,  
our mission continues

At Posit, our goal is to make data science more open, intuitive, accessible, and collaborative. We provide tools that make it easy for individuals, teams, and enterprises to leverage powerful analytics and gain insights they need to make a lasting impact.

14 / 43

# Install RStudio

Click on *Download RStudio*

RStudio Desktop      RStudio Server

## RStudio Desktop

Find out more about RStudio Desktop and RStudio Desktop Pro below.

[DOWNLOAD RSTUDIO](#)



15 / 43

# Install RStudio

Scroll down

Click on *Download RStudio*

Choose the right operating and download RStudio

RStudio Desktop      RStudio Server

## RStudio Desktop

Find out more about RStudio Desktop and RStudio Desktop Pro below.

[DOWNLOAD RSTUDIO](#)



» *In case you haven't already installed RStudio. Do it now! Are you stuck? Let us know.*

16 / 43

## 4. RStudio interface

17 / 43

### RStudio environment

 Let us discover *RStudio* together!

18 / 43

# 5. The basics in R

19 / 43

## Type of objects

- *vectors*
  - *matrices*
  - *arrays*
  - *data frames*
  - *lists*
  - *functions*
- ...

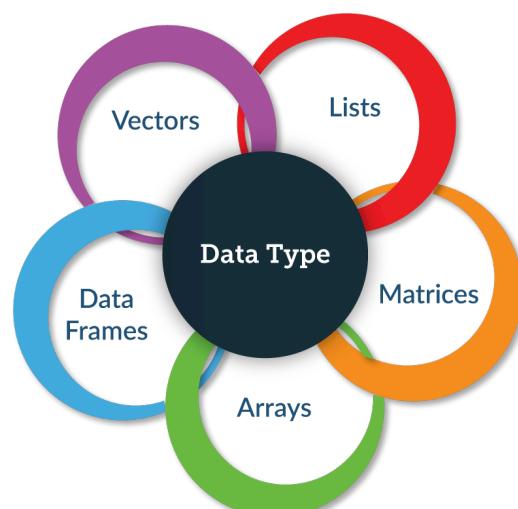


Figure from <https://www.edureka.co/blog/r-programming-language>

20 / 43

# Vectors

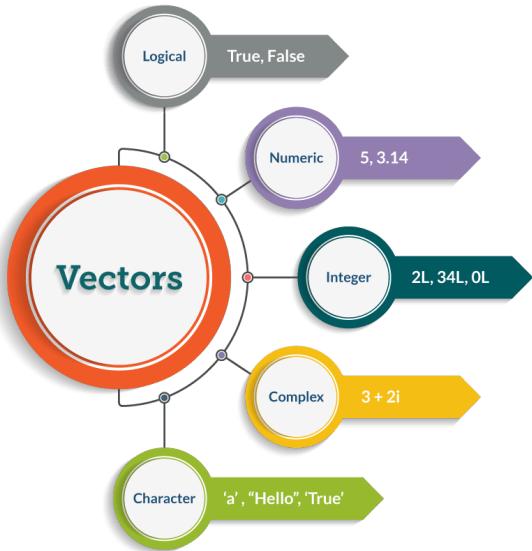


Figure from <https://www.edureka.co/blog/r-programming-language>

21 / 43

## First steps in R ; let's make a script ...

**</> Time to get started with the real work and code together**

You can find the code on the next slides.

There is also a script `Fruit.R` that contains the same code (see course material for Part 1 online)

22 / 43

## </> Code-Blocks: character vector

We create a vector and call it *Fruit*

```
Fruit <- c("Apples", "Bananas", "Lemons", "Berries", "Peaches", NA)
```

- `c()` represents *concatenate*, all elements between the brackets ( separated by ,) are 'merged' into one element
- `<-` means that we want to store the result in an object (a vector) that we call `Fruit`
- the " " that surround the elements indicate that these are of the type *character*

Let's look at the object `Fruit` by printing it to the console

```
Fruit
```

```
## [1] "Apples" "Bananas" "Lemons" "Berries" "Peaches" NA
```

We can also check the structure of the object `Fruit` by using the `str()` function

```
str(Fruit)  
## chr [1:6] "Apples" "Bananas" "Lemons" "Berries" "Peaches" NA
```

23 / 43

## </> Code-Blocks: numeric vector

We create a vector called `Weight`

```
Weight <- c(230, 191, 93, 100, 48, 244)
```

In this case, the elements are 'recorded' as numeric elements

```
str(Weight)  
## num [1:6] 230 191 93 100 48 244
```

Now, we can start calculating... For example, we can apply the function `mean( )`

```
mean(Weight)  
## [1] 151
```

24 / 43

## </> Code-Blocks: logical vector

We create a vector called *Yellow*

```
Yellow <- c(F, T, T, F, F, F)
```

The element of the vector *Yellow* are 'logical operators' (TRUE or FALSE)

```
str(Yellow)  
## logi [1:6] FALSE TRUE TRUE FALSE FALSE FALSE
```

Let us have a closer look at the vector *Yellow*

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

25 / 43

## </> Code-Blocks: a data frame

We create a data.frame called *Fruit\_data* using the function **data.frame( )**

```
Fruit_data <- data.frame(Fruit, Weight, Yellow)
```

Let's have a look at \**Fruit\_data*\*

```
Fruit_data  
## #> #> Fruit Weight Yellow  
## #> 1 Apples 230 FALSE  
## #> 2 Bananas 191 TRUE  
## #> 3 Lemons 93 TRUE  
## #> 4 Berries 100 FALSE  
## #> 5 Peaches 48 FALSE  
## #> 6 <NA> 244 FALSE
```

We can check the structure of the object *Fruit\_data*

```
str(Fruit_data)  
## 'data.frame': 6 obs. of 3 variables:  
## $ Fruit : chr "Apples" "Bananas" "Lemons" "Berries" ...  
## $ Weight: num 230 191 93 100 48 244  
## $ Yellow: logi FALSE TRUE TRUE FALSE FALSE FALSE
```

26 / 43

## </> Code-Blocks: subsetting a data frame

The **\$ operator** is used to refer to a vector

```
Fruit_data$Weight  
## [1] 230 191 93 100 48 244
```

By **indexing** specific columns- (and/or) row-numbers can be selected [row, column]

### Examples of indexing:

- Retrieve element that is located at row 1 and column 1

```
Fruit_data[1,1]
```

```
## [1] "Apples"
```

- Retrieve all elements in column 3

```
Fruit_data[,3]
```

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

- Retrieve all elements in row 3

```
Fruit_data[3,]
```

```
##      Fruit Weight Yellow  
## 3 Lemons     93    TRUE
```

27 / 43

## 6. Install and use packages

28 / 43

# Packages?

- The universum is in constant development
  - Package = extensions of the Base-functions
  - Range from specialised to generic/universal packages
  - Overview on [https://cran.r-project.org/web/packages/available\\_packages\\_by\\_name.html](https://cran.r-project.org/web/packages/available_packages_by_name.html)
  - Which package(s) to use?



29 / 43

# Welcome to the tidyverse

A **MUST-HAVE** for everyone!

# Tidyverse

Packages   Blog   Learn   Help   Contribute

R packages for data science

The tidyverse is an opinionated collection of R packages designed for data science. All packages share an underlying design philosophy, grammar, and data structures.

Install the complete tidyverse with:

```
install.packages("tidyverse")
```

🔗 More information at <https://tidyverse.org>

# install.packages( ) and library( )

Packages can be **downloaded and installed**

- by 'clicking' in RStudio
- **by using a function in the console or script**

*What Sven usually does: install.packages()*

*What Tine usually does: click and go!*

For example,

```
install.packages("tidyverse",  
                 dependencies = T)
```

Packages should be **activated** at the start of the session

- by 'clicking' in RStudio
- **by using a function in de console or script**

*What we both usually do: library()*

For example,

```
library("tidyverse")
```

## 7. Working with R-projects

# Handling all these files

During analyses, we will regularly handle different files:

- data sets
- scripts
- output (html-files; figures; ...)

Consequently, we should refer to these files in our code.

33 / 43

## Taking the right path ...

There are two ways to refer to these files.

### USING ABSOLUTE PATHS

We can do this using an *ABSOLUTE path*. Below is an example of an absolute path:

```
'c:/Users/Sven/Mijn Documenten/UAntwerpen/Ana
```

Absolute paths make it difficult to share your work; you get into trouble when you change laptops; ...

### USING RELATIVE PATHS

To avoid these problems, RStudio introduced the concept of an *R-Project*.

Within a project you can use *RELATIVE paths* (that starts from the folder in which you save the project).

Below is an example of a relative path:

```
'~/R_Script/Analysescript1.R'
```

34 / 43

# Sharing projects?

If I put a project of Sven on my laptop, the **RELATIVE paths** might get me into trouble.  
For example, the Rmd-file 'Slides\_part1.Rmd' is located at

```
'c:/Users/Sven/Dropbox/ICO_R_2022/Presentations/Part 1/Slides_part1.Rmd'
```

Of course, I do not have the same structure on my laptop...

To facilitate sharing of projects, the package `here` has been created.

35 / 43

## Creating relative paths with function `here()`

The function `here()` creates a paths relative to the top-level directory of my laptop.

```
library(here)  
here()
```

```
## [1] "/Users/demaeyer/Documents/Praatjes/Workshops/R Course ICO 2024"
```

```
here("Presentations", "Part 1", "Slides_part1_Rmd")
```

```
## [1] "/Users/demaeyer/Documents/Praatjes/Workshops/R Course ICO 2024/Presentations/Part 1/Slides_part1_
```

36 / 43

# Exercises

## Creating a project in RStudio

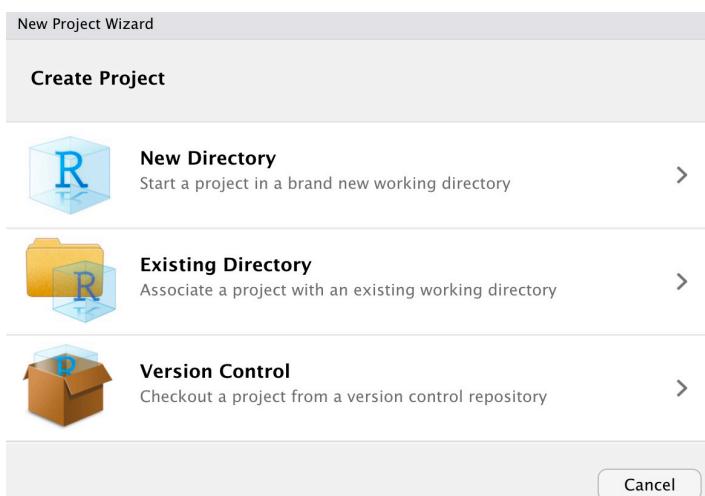
You can find the folder **Exercises** at the Github:  
[https://github.com/Sdemaeyer2/R\\_workshop\\_ICO2024/](https://github.com/Sdemaeyer2/R_workshop_ICO2024/). Download this folder and put it somewhere on your laptop.

37 / 43

## Creating a project in RStudio

Click on **File/New Project...**

Next, you can choose:



38 / 43

# 8. Importing data

39 / 43

## Data comes in different formats

Data exist in various formats, but the most common ones are:

- MS Excel
- SPSS / Stata / SAS
- text (csv, tab-delimited)

Various methods and packages have been developed to import these types of data

We will show some possibilities

40 / 43

# Importing MS Excel data with the `readxl` pakket

Make sure that the package is installed on your laptop Then, you can use the code below to import the data

```
library(readxl)
Data <- read_excel(
  path = "<path to map and file>"
)
```

*Imagine:* you have an excel-file **Datacollection1** with multiple sheets

*Goal:* you only want to import the sheet with name **Group2** and save it as a data frame **Data\_Group2**

```
Data_Group2 <- read_excel(
  path = "<path to map and file>/Datacollection1.xlsx",
  sheet = "Group1"
)
```

You shouldn't use the argument `sheet` if:

- the excel-file only includes 1 sheet
- there are multiple sheets but you only want to import the first one

41 / 43

# Importing SPSS data with package `foreign`

Make sure that the package is installed on your laptop Then, you can use the code below to import the data

```
library(foreign)
Data <- read.spss(
  file = "<path to map and file>",
  use.value.labels = FALSE,
  to.data.frame = TRUE
)
```

`use.value.labels` argument:

In SPSS, data is labelled. Do you want these labels to appear in the R data-frame?

*For example. Variable with 3 categories: 1 = "Low", 2 = "Average", 3 = "High". These labels are included in the SPSS-file. You can choose to import these labels in the R data frame (`use.value.labels = TRUE`) or only the numbers 1, 2 and 3 that represent these categories (`use.value.labels = FALSE`)*

42 / 43

# Importing csv-files with function `read.table( )`

csv-files usually look like this (separated by `,` or `;`)

```
Column1, Column2, Column3  
1, 3, 5  
2, 4, 6  
8, 10, 99
```

To import these data, the function `read.table( )` is most convenient...

```
Data <- read.table(  
  file = "<path to map and file>",  
  header = TRUE,  
  sep = ",",  
  dec = ".")
```

- **header** argument:

The first row includes column (variable) names or not?

- **sep** argument:

Which character is used to separate the values in the different columns?

- **dec** argument:

Which character is used to indicate a decimal point? (Can be a point or a comma)