Introduction

Javier García-Bernardo (based on material by Gerko Vink)

Introduction to R and RStudio

Goal of this course

- Gentle introduction to:
  - Programming languages
  - R for data science
- Provide a foundation for the courses Gerko Vink's course and Erik-Jan van Kesteren

Why do we care about R?

Why programming languages?

- Faster
- Easier
- Communication: Beautiful plots/interactive
- Flexibly (new methods)
- Reproducibility! (open an old Excel file and try to understand what you did)

Why R:

- Designed for statistics/data science
- Huge community of users
- From a personal perspective: You know R = you are employed

R for data science

What is R / RStudio?

Communicating with the comptuer

R ~ English

- Programming language, allows you to communicate with the computer

RStudio: Integrated Development Environment ~ Word

- Makes using R as effective and efficient as possible:
  - Code editor
  - Nice extras: Syntax highlighting; Code completion; File explorer; Help

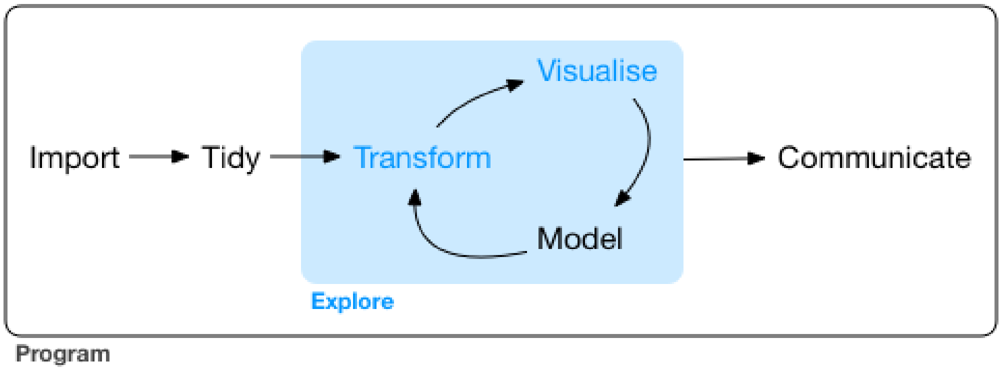


Figure 1: Source: R for Data Science

R packages ~ Tabs in Word

- Extend the functionalities of base R
- You can install and use new packages:

```
install.packages("ggplot2") #Install new package (you only need to do it once)
library(ggplot2) #Load the package
```

RStudio

Source editor Write your R code (load data, clean it, model it, etc)

Environment/Workplace All the variables that you have defined

Files File explorer, find your files.

Help Get information about code (super useful!)

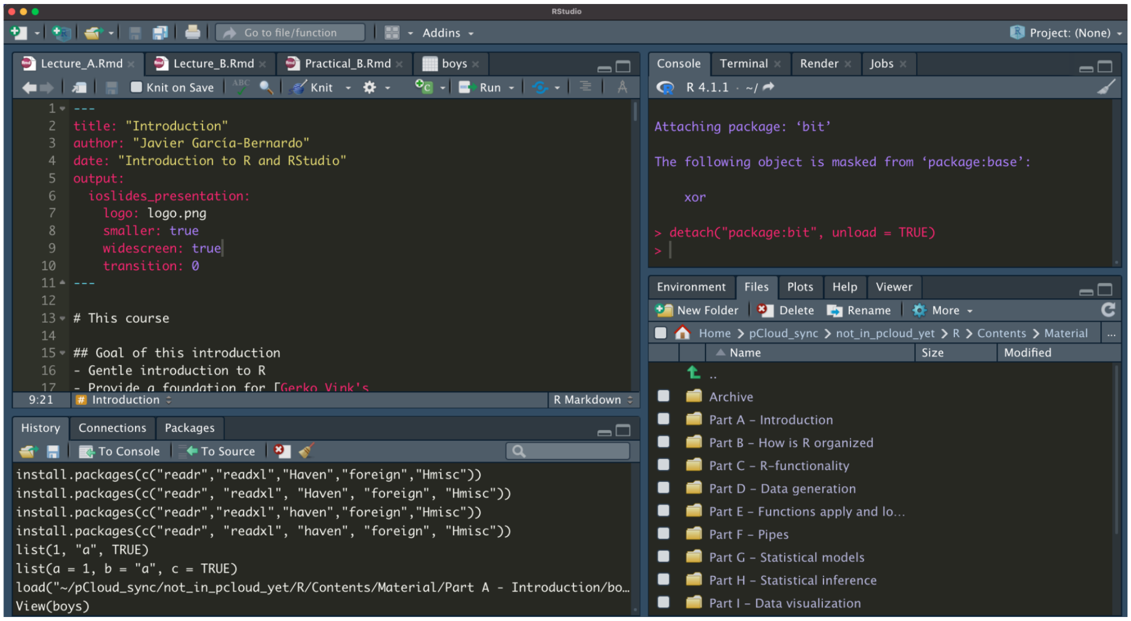
Console Write R code (not recommended at this point) and see the output of your R scripts

Plots See the plots, and export it

History History of all the code you have run.

Packages All packages that you have loaded (I don't recommend loading/unloading packages this way)

Terminal Run commands on your terminal (this is not R, you won't need to use this)



```
---
title: "Introduction"
author: "Javier Garcia-Bernardo"
date: "Introduction to R and RStudio"
output:
  ioslides_presentation:
    logo: logo.png
    smaller: true
    widescreen: true
    transition: 0
---

# This course

## Goal of this introduction
- Gentle introduction to R
- Provide a foundation for [Gerko Vink's
```

```
Attaching package: 'bit'

The following object is masked from 'package:base':

    xor

> detach("package:bit", unload = TRUE)
>
```

..
Archive
Part A – Introduction
Part B – How is R organized
Part C – R-functionality
Part D – Data generation
Part E – Functions apply and lo...
Part F – Pipes
Part G – Statistical models
Part H – Statistical inference
Part I – Data visualization

```
install.packages(c("readr","readxl","Haven","foreign","Hmisc"))
install.packages(c("readr", "readxl", "Haven", "foreign", "Hmisc"))
install.packages(c("readr","readxl","haven","foreign","Hmisc"))
install.packages(c("readr", "readxl", "haven", "foreign", "Hmisc"))
list(1, "a", TRUE)
list(a = 1, b = "a", c = TRUE)
load("~/pCloud_sync/not_in_pcloud_yet/R/Contents/Material/Part A - Introduction/bo…
View(boys)
```

Figure 2: RStudio

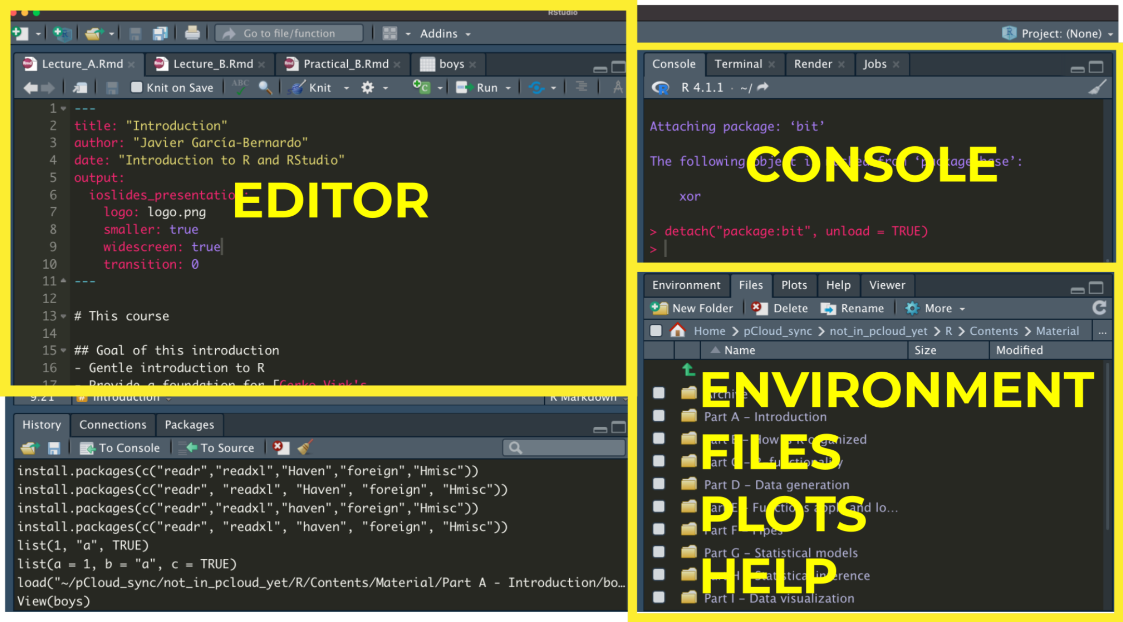
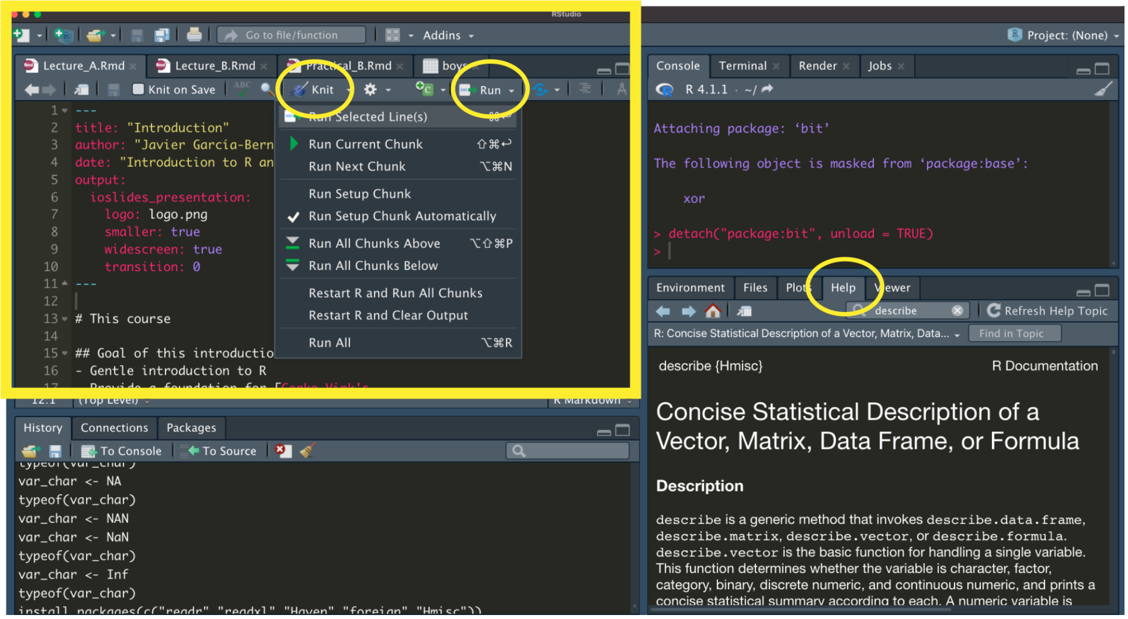


Figure 3: Basic units

Basic units of RStudio



```
---
title: "Introduction"
author: "Javier García-Bern
date: "Introduction to R an
output:
  ioslides_presentation:
    logo: logo.png
    smaller: true
    widescreen: true
    transition: 0
---

# This course

## Goal of this introductio
- Gentle introduction to R
  Provide a foundation for [Corko Vink's
```

```
Attaching package: ‘bit’

The following object is masked from ‘package:base’:

    xor

> detach("package:bit", unload = TRUE)
> 
```

describe {Hmisc} R Documentation

Concise Statistical Description of a Vector, Matrix, Data Frame, or Formula

Description

describe is a generic method that invokes describe.data.frame, describe.matrix, describe.vector, or describe.formula. describe.vector is the basic function for handling a single variable. This function determines whether the variable is character, factor, category, binary, discrete numeric, and continuous numeric, and prints a concise statistical summary according to each. A numeric variable is

```
typeof(var_char)
var_char <- NA
typeof(var_char)
var_char <- NAN
var_char <- NaN
typeof(var_char)
var_char <- Inf
typeof(var_char)
install.packages(c("readr", "readxl", "Haven", "foreign", "Hmisc"))
```

Figure 4: Run code

Run pieces of code

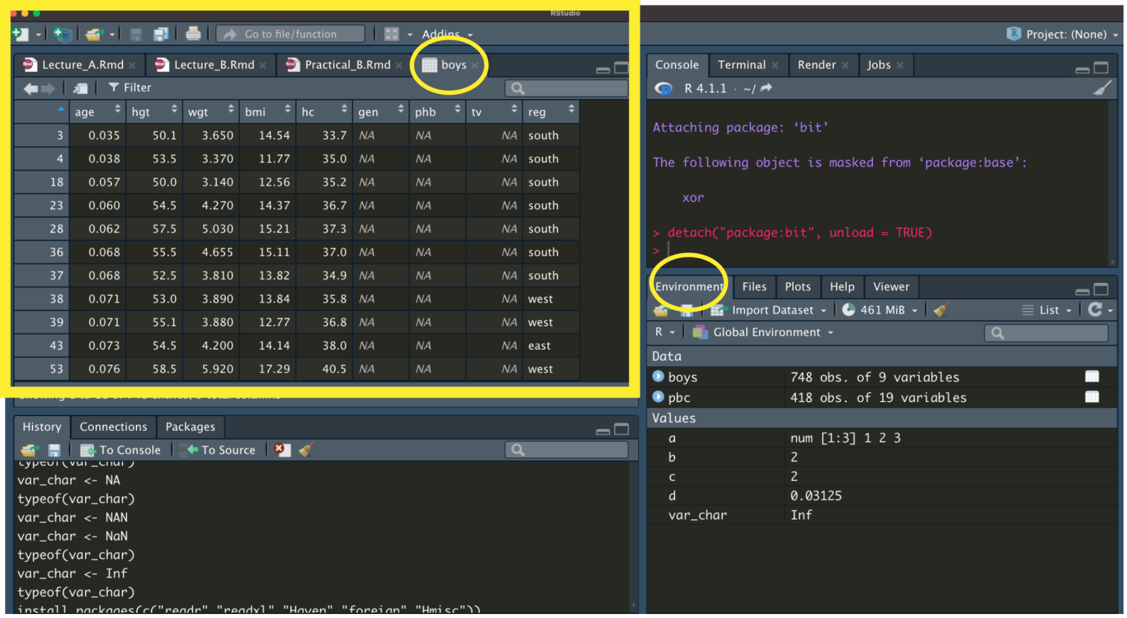
View your data/objects

Create a new file: RScript and RNotebook

Paths & working directory

Paths and working directory (folder)

- Your computer finds files by their path:
  - "/Users/javiergb/Desktop/somefile.csv" (Mac/Linux)
  - "C:\Users\javiergb\Desktop\somefile.csv" (Windows)
- In Windows:
  - Always use "/" (R will convert it)
  - "\" is a "escape character", with a special meaning
- Paths can be:



	age	hgt	wgt	bmi	hc	gen	phb	tv	reg
3	0.035	50.1	3.650	14.54	33.7	NA	NA	NA	south
4	0.038	53.5	3.370	11.77	35.0	NA	NA	NA	south
18	0.057	50.0	3.140	12.56	35.2	NA	NA	NA	south
23	0.060	54.5	4.270	14.37	36.7	NA	NA	NA	south
28	0.062	57.5	5.030	15.21	37.3	NA	NA	NA	south
36	0.068	55.5	4.655	15.11	37.0	NA	NA	NA	south
37	0.068	52.5	3.810	13.82	34.9	NA	NA	NA	south
38	0.071	53.0	3.890	13.84	35.8	NA	NA	NA	west
39	0.071	55.1	3.880	12.77	36.8	NA	NA	NA	west
43	0.073	54.5	4.200	14.14	38.0	NA	NA	NA	east
53	0.076	58.5	5.920	17.29	40.5	NA	NA	NA	west

```
Attaching package: ‘bit’

The following object is masked from ‘package:base’:

    xor

> detach("package:bit", unload = TRUE)
> 
```

Data	
boys	748 obs. of 9 variables
pbc	418 obs. of 19 variables
Values	
a	num [1:3] 1 2 3
b	2
c	2
d	0.03125
var_char	Inf

```
typeof(var_char)
var_char <- NA
typeof(var_char)
var_char <- NAN
var_char <- NaN
typeof(var_char)
var_char <- Inf
typeof(var_char)
install.packages(c("readr", "readxl", "Haven", "foreign", "Hmisc"))
```

Figure 5: View data

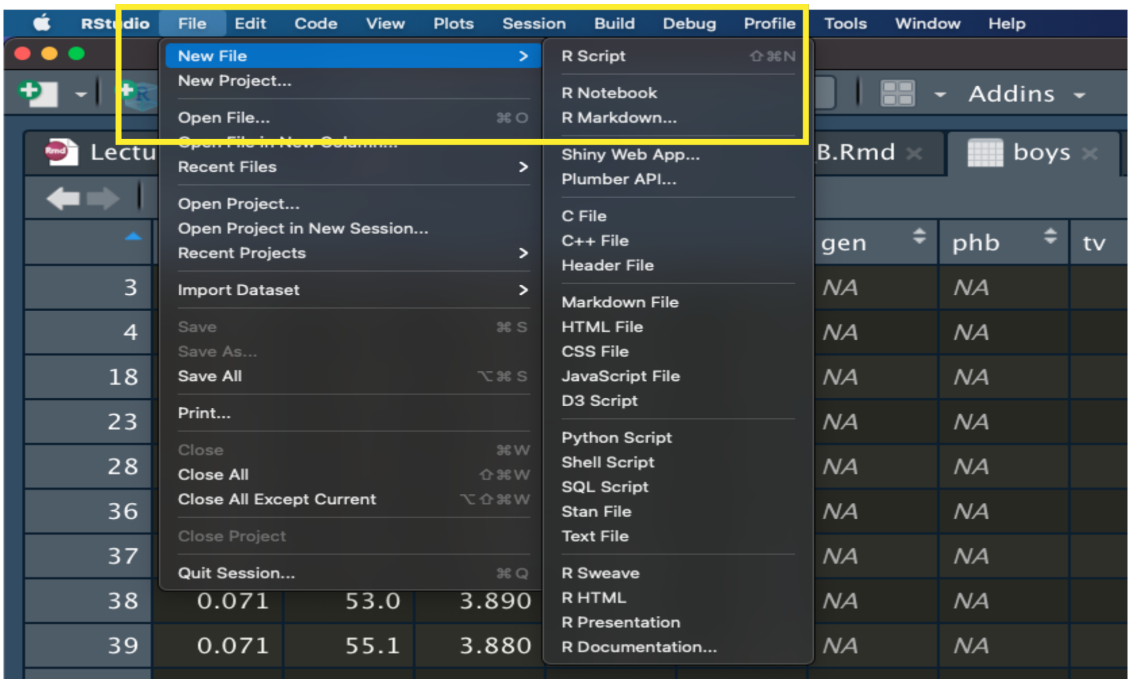


Figure 6: Create new files

  - Absolute (as the ones above). They are defined from the root directory, the top-most directory.
  - Relative. They are defined from the working directory
    - e.g. “data/somefile.csv” –> in our working directory, find the folder “data”, and inside, the file “somefile.csv”
    - It makes possible to use the code in different computers.

Paths and working directory (cont)

Important shortcuts for relative paths:

- “`../`somefile.csv”: find “somefile.csv” one level down
- “`../../`somefile.csv”: find “somefile.csv” two levels down
- “`./`somefile.csv”: find “somefile.csv” in the current level (not so useful, it is identical to “somefile.csv”)
- “`~/`somefile.csv”: find “somefile.csv” in your home directory

Changing the working directory with RStudio:

- “Session (top menu) -> Set Working directories”
- “Files (RStudio unit) -> More –> Set as working directory”
- On a notebook –> Working directory = directory of the notebook

Type of R documents

R-scripts (.R)

- Text file with R code (you can open it in any text editor)
- Working directory:
  - If associated to a project: your home directory (e.g. “/Users/javiergb”)
  - If not associated to a project: the project directory
- Output in the console/plots menus

Rstudio-projects (.Rproj)

- Contains a .Rproj file within the directory, with project options.
- Working directory: the project directory where the .Rproj file resides.
- More info on setting up projects

R-notebooks (.Rmd)

- R-markdown files (like this one!) – combines chunks of texts and code
- Working directory: the directory where the notebook is located
- Outputs directly in the editor. Can be knitted to HTML/PDF/Word
- Most useful for data science

How does R (and Python) work

Variables and code

Variables:

- Tell the computer to save an `object` (a number, a string, a spreadsheet) with a name.
- Creating variables in R is very straightforward:
  - you just use `<-` (assignment operator)
- For example, if you assign the value `100` (an element) to variable `a`, you would type

```
a <- 100
print(a)
```

```
## [1] 100
```

Code:

- Instructions to modify variables
- Can be organized in functions: blocks of code that take some input and return some output

Objects in R:

- Everything is an object in R, and can be assigned to a variable name

Basic data types (elements)

- `character`: "some text"
- `numeric`: e.g., 2.1
- `integer`: e.g., 2L
- `logical`: TRUE/FALSE
- `factor`: e.g., factor("amsterdam")

Basic data structures

- Consist of data types and functions to transform them
  - `vector`: c(2, 4, 2)
  - `list`: list(first_col = 1, second = "a", third = TRUE)
  - `matrix`: matrix(c(4, 4, 4, 4), nrow = 2, ncol = 2)
  - `data.frame`: The most important ~ spreadsheet

The help

- Everything that is published on the Comprehensive R Archive Network (CRAN) and is aimed at R users, must be accompanied by a help file.
- If you know the name of the function that performs an operation, e.g. `anova()`, then you just type `?anova` or `help(anova)` in the console, or use the "Help" menu.
- If you do not know the name of the function: type `??` followed by your search criterion. For example `??anova` returns a list of all help pages that contain the word 'anova'
- Alternatively, the internet will tell you almost everything you'd like to know and sites such as http://www.stackoverflow.com and http://www.stackexchange.com, as well as `Google` can be of tremendous help.
    - If you google R related issues; use 'R:' as a prefix in your search term

Calling objects

- You just use type the name you have given to the object
- For example, we assigned the value `100` to object `a`.

```
a <- 100
```

To call object `a`, we would type

```
a
```

```
## [1] 100
```

Writing code

1. Using functions

```
# This is a comment, it won't be read by R
student_number <- 4
paste("The number of students is: ", student_number, sep = " ")
```

```
## [1] "The number of students is:  4"
```

```
#sep can be any character, or "\n" (newline), "\t" (tab),
```

---

2. Using packages

```
# install.packages("tidyverse") #installing packages
library(readr) #loading the library to read csv, usually on top of the file

# Using the readr library (the readr:: is optional, but useful when the function)
data <- readr::read_csv("../common_datasets/dataset_boys.csv", col_select = c("age","hgt"))
## Rows: 748 Columns: 2
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (2): age, hgt
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
# Summary statistics
summary(data)
##       age              hgt
##  Min.   : 0.035   Min.   : 50.00
##  1st Qu.: 1.581   1st Qu.: 84.88
##  Median :10.505   Median :147.30
##  Mean   : 9.159   Mean   :132.15
##  3rd Qu.:15.267   3rd Qu.:175.22
##  Max.   :21.177   Max.   :198.00
##                   NA's   :20
```

---

3. Some other important considerations

The computer cannot read your mind

```
a <- 100
print()
```

```
## Error in print.default(): argument "x" is missing, with no default
```

The computer reads from the top to the bottom

```
a <- 100
print(b)
b <- 10
```

```
## Error in print(b): object 'b' not found
```

Practical A

Goal: Get used to RStudio using R as a calculator, and install one library

1. Create an R script
2. Create an R project
3. Create an R notebook