Fundamentals of R

Objects, Class, and Data Structures

Henrique Sposito and Livio Silva-Muller

INSTITUT DE HAUTES
ÉTUDES INTERNATIONALES
ET DU DÉVELOPPEMENT

GRADUATE INSTITUTE
OF INTERNATIONAL AND
DEVELOPMENT STUDIES

What did we learn online?

Conceptually:

- What is R?
- Why R?
- The R panes
- How to get help online
- *Clear conceptual thinking with data is important in R!*

Practically:

```
setwd()
install.packages()
library()
read_csv()
View()
# +, -, *, / arithmetic operations.
# ? asking for help.
# ← the assign operator.
```

- R *messages, warnings, and error are informative, read them!*

Failure is another stepping stone to greatness, in R at least…

- Questions about the online exercises or the videos?

Lecture:

- R basics (objects and classes)
- Data (from vector to data frames)
- First wrangling steps (with base R)

Practical:

- Download, load, and explore data
- Grouping and subsetting data
- Mind the gap with **GAPMINDER**

R Basics: Objects

"All I know is what I have words for" - Ludwig Wittgenstein

- What are objects in R?

Can be anything, as long as it is assigned...

```
"Am I an object?"
```

```
#> [1] "Am I an object?"
```

```
object_1 ← "Or I am an object?"
object_1
```

```
#> [1] "Or I am an object?"
```

The rules of the game (class 1/6)

Object class is the grouping of objects that can be described in terms of the attributes its members have in common.

Can anything be a class?

Yes, as long as the rules of the game are set as methods and are coherent!

But we are only talking about a few basic ones today...

Character (class 2/6)

Just text...

```
object_character ← "Or I am an object?"
class(object_character)
```

```
#> [1] "character"
```

```
# What can we do with character objects?
paste(object_character, object_character, sep = " - ")
```

```
#> [1] "Or I am an object? - Or I am an object?"
```

`

Numeric (class 3/6)

Some classes allow for arithmetic operations

```
object_numeric ← 3
class(object_numeric)
```

```
#> [1] "numeric"
```

```
# Can we operate on numeric objects?
(object_numeric*object_numeric)/(object_numeric+object_numeric)
```

```
#> [1] 1.5
```

```
# Remember, parenthesis are important for the order of operations!
```

`

Logical (class 4/6)

Either TRUE or FALSE

```
object_logical ← TRUE
class(object_logical)
```

```
#> [1] "logical"
```

```
isTRUE(object_logical) # Logical check
```

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

`

Dates (class 5/6)

Dates in R can be annoying, specially if you are using base R...

```
Sys.Date() # Get the current date
```

```
#> [1] "2023-08-31"
```

```
object_date ← Sys.Date()
class(object_date) # What can we do with data?
```

```
#> [1] "Date"
```

```
Sys.Date() - 1
```

```
#> [1] "2023-08-30"
```

Some R packages are excellent for this though ´{lubridate}´ and ´{messydates}´

`

Beware: object ≠ type

Type refers to how the data is stored; the 5 basic types in R are double, integer, complex, logical, character

```
class(1)
```

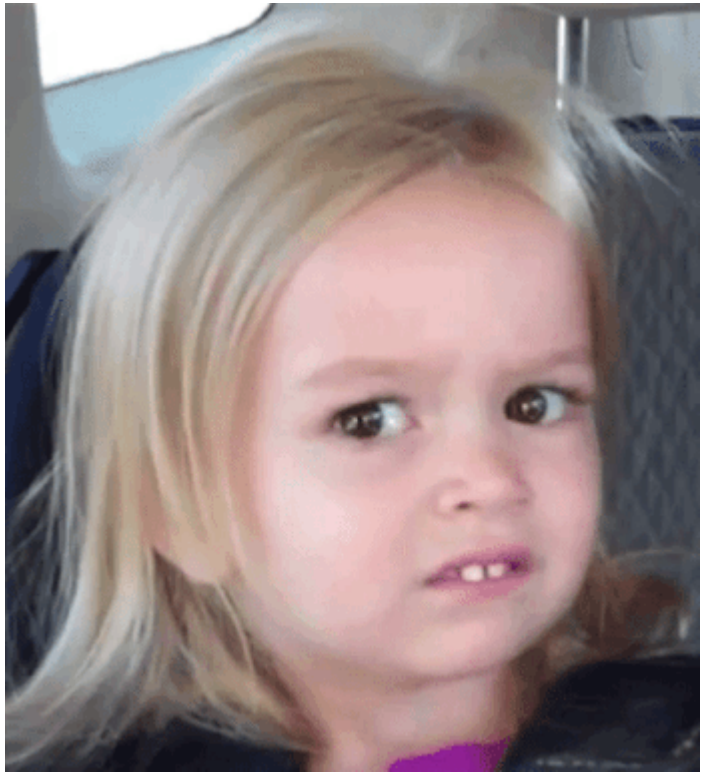
```
#> [1] "numeric"
```

```
typeof(1)
```

```
#> [1] "double"
```

But if this is confusing to you, do not worry about it ...

Paying attention to class right now?



So far...

We create objects by assigning things.

Objects have a class, which defines what sort of operations I can do with objects.

Class is different than type.

Now let's learn about data structures:

- vectors,
- matrices,
- dataframes,
- and lists.

Vectors (data structures 1/4)

Vector: one-dimensional with one class of data

```
lecturers ← c("Henrique", "Livio") #
character
age ← c(81, 18) # numeric
speaks_portuguese ← c(TRUE, TRUE) # logical
```

Matrixes (data structures 2/4)

Matrix: two dimensional objects with one class and a fixed number of rows/columns

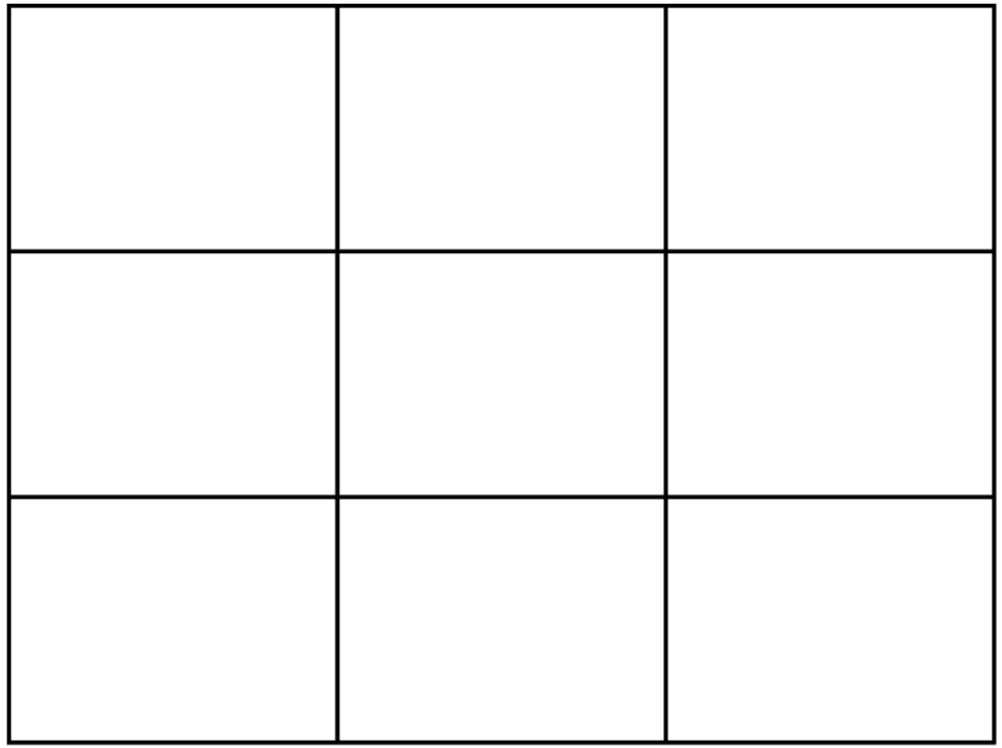
```
matrix_1 ← matrix(0, ncol = 2, nrow = 2)
matrix_1
```

```
#>      [,1] [,2]
#> [1,]    0    0
#> [2,]    0    0
```

```
dim(matrix_1)
```

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

There are also *arrays*, three or more dimensional objects of one class, but do not worry about it now.



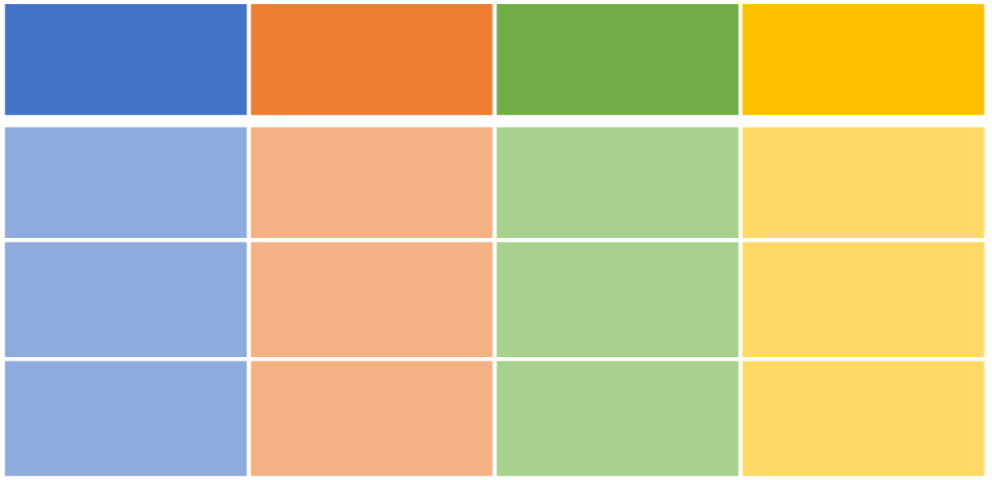
Data Frames (data structures 3/4)

Data frames objects can have multiple possible classes but a fixed number of rows and columns

```
course ← data.frame(cbind(lecturers, age,
speaks_portuguese))
course
```

```
#>   lecturers age speaks_portuguese
#> 1  Henrique  81              TRUE
#> 2     Livio  18              TRUE
```

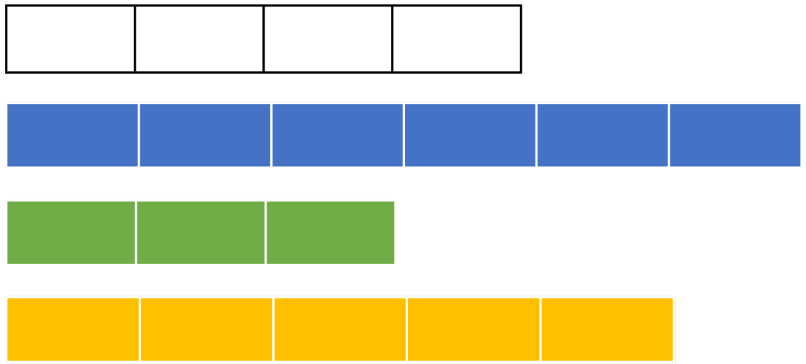
```
# cbind() binds columns, rbind() binds rows
```



Lists (data structures 4/4)

Lists: group together R objects of diverse classes or dimensions.

```
list ← list(lecturers, age,
speaks_portuguese, 3,course)
```



```
list
```

```
#> [[1]]
#> [1] "Henrique" "Livio"
#>
#> [[2]]
#> [1] 81 18
#>
#> [[3]]
#> [1] TRUE TRUE
#>
#> [[4]]
#> [1] 3
#>
#> [[5]]
#>   lecturers age speaks_portuguese
#> 1  Henrique  81               TRUE
#> 2     Livio  18               TRUE
```

You can navigate (through data) if you have $ (functions 1/5)

Is the "$" operator a function in R? (Hint: see ´?"$"´)

```
# The $ operator can be used to extract part from an object:
course$lecturers
```

```
#> [1] "Henrique" "Livio"
```

```
# Or replace parts of an object:
course$lecturers ← c("Sposito", "Silva-Muller")
```

An operator in R is a function that can be written without parentheses...

`

If you do not have $, friends can help :) (functions 2/5)

Brackets make some great friends (run ´?"["´ for help)

```
course$lecturers[1] # first element in vector
```

```
#> [1] "Sposito"
```

```
course[1,] # first row in all columns
```

```
#>   lecturers age speaks_portuguese
#> 1   Sposito  81              TRUE
```

```
course[,1] # all rows in first column
```

```
#> [1] "Sposito"      "Silva-Muller"
```

```
#course[1, 1] # first value in first column
```

Logical operators (functions 3/5)

- "=="/"!=" are values equal/different?
- is bigger/bigger or equal ">"/">=" than?
- is smaller/smaller or equal "<"/"<=" than?

```
course$lecturers[1] == course$lecturers[2]
```

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

```
course$age[2] < course$age[1]
```

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

For multiple conditions, use "&"/"|" (and/or)

```
course$lecturers[1] != course$lecturers[2] & course$age[2] < course$age[1]
```

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

`

If it is true, then yes! (functions 4/5)

We function ´ifelse()´ works with conditional element selection... (wait what?)

```
ifelse(test, yes, no)
```

```
course$old ← ifelse(course$age > 30, "old", "young")
course
```

```
#>      lecturers age speaks_portuguese   old
#> 1       Sposito  81              TRUE   old
#> 2 Silva-Muller  18              TRUE young
```

`

GREP (Grammar of Regular Expressions) (functions 5/5)

Functions to find, match and replace strings!

```
# Can we find if there is an "silva" in a variable?
grepl("silva", course$lecturers, ignore.case = TRUE)
```

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

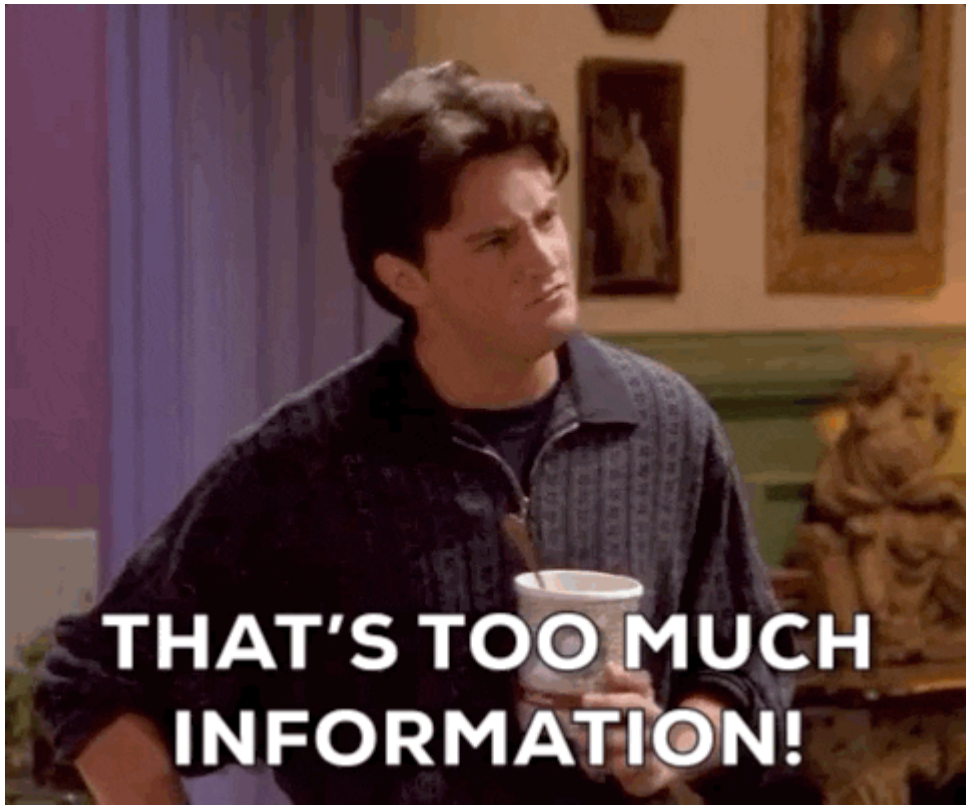
```
# What if we want to replace it?
course$lecturers ← gsub("Silva", "Awesome", course$lecturers)
course
```

```
#>        lecturers age speaks_portuguese   old
#> 1        Sposito  81              TRUE   old
#> 2 Awesome-Muller  18              TRUE young
```

`

Fun break...

```
install.packages("beepr")
library(beepr)
beep(8)
```



Gotcha you!

One last thing: factors!

Factors (class 6/6)

Factor in R refers to a variables that have a fixed and known set of possible values(days of the week, months, possible survey responses,...). *Anything can be a treated as a factor, if you want*:

```
course$age ← factor(course$age, levels = c(0:122))
course$age
```

```
#> [1] 81 18
#> 123 Levels: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 ... 122
```

```
summary(course$lecturers)
```

```
#>    Length     Class      Mode
#>         2 character character
```

```
course$lecturers ← factor(course$lecturers, levels = c("Sposito","Awesome-Muller","Yildiz"))
summary(course$lecturers)
```

```
#>        Sposito Awesome-Muller         Yildiz
#>              1              1              0
```

`

Other useful functions we forgot?

Henrique

```
summary(course)
```

Livio

```
unique(c(2, 2, 2, "three", "three",TRUE))
```

```
#> [1] "2"     "three" "TRUE"
```

```
mean(c(18, 81, 28))
```

```
#> [1] 42.33333
```

```
median(c(18, 81, 28, NA))
```

```
#> [1] NA
```

```
median(c(18, 81, 28, NA), na.rm=TRUE)
```

```
#> [1] 28
```

`

Base R

- All operators we cover today are base R
- Above all, teaching base R facilitates comprehension of how R, as a software and programming language, works for beginners

Summing Up!

- You can create objects as long as you assign it.
- Objects have a class (character, numeric, logical, factor, etc...) and also a type (but do not worry about this now).
- Depending on dimensions and class, combined objects become datastructures: vectors, matrices, dataframes, lists, etc...
- The datastructure we will use the most are dataframes, which stores vectors of different classes but same lenght.
- You can navigate through data structures and do operations in it.

Housekeeping

```
1  # Title: Homework 1
2  # Author: Henrique Sposito & Livio Silva-Muller
3  # Date: 08/03/2023
4
5  # Question 1
6
7  1 + 1
8
9  # The answer is 2!
10
11 # Question 2
12
13 1 * 1
14
15 # The answer in 1!
16
17 # BLAblablablablablabalabalabalabalab - This line is too long, so this is why
18 # I splited into 2 lines./
19
```


QUESTIONS?

References

For more cool and informative slides about R stuff, see favstats

Douglas, A., Ross, D., Mancini, F., Couto, A. & Lusseau, D. (2022). An Introduction to R

Mesquita, E. B., & Fowler, A. (2021). Thinking clearly with data: A guide to quantitative reasoning and analysis. Princeton University Press.