Data Analytics in R Session 2

Maria Kunevich

Course: materials and communication

Course webpage:

https://github.com/Maria-13/DataAnalytics_R

Communication:

https://qithub.com/Maria-13/DataAnalytics_R/discussions

Setting up the course environment

- 1. Ensure you have R and RStudio running on your laptop
- 2. Get familiar with RStudio environment
- 3. Create a GitHub account
- 4. Find the repository:

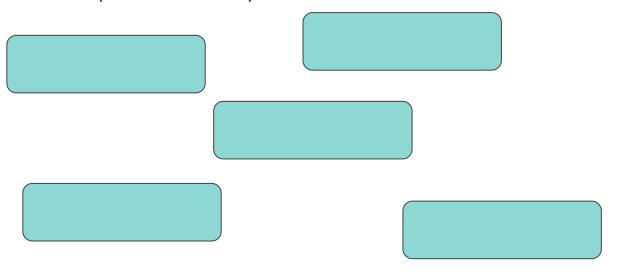
https://github.com/Maria-13/DataAnalytics_R

- 5. Start "watching" the repository: if you "watch" a project, you're notified whenever there are any updates
- 6. Introduce yourself on our Miro board:

https://miro.com/welcomeonboard/SUw3RHpXWjBpUDF2V0dwTWhkOFVVUnIUTnZ4Qm1oV GtSTVN0SmNCUmJyODhydU9hUzA3VUpNZVZHRnNBenhqVHwzMDc0NDU3MzY2NDE00 TE3Njc4?share_link_id=308470562965

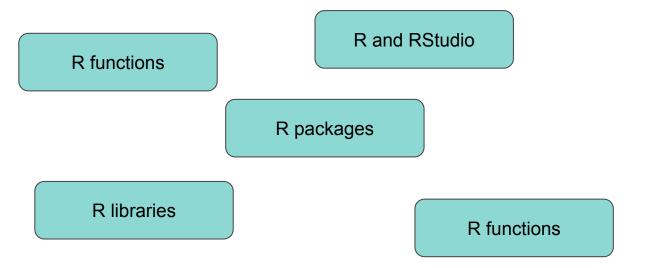
Recap

What concepts/words do you remember from our first session?



Recap

What concepts/words do you remember from our first session?



Plan for today

- 1. My first R script writing readable code
- 2. Basic R calculations in R
- 3. Functions and their arguments
- 4. R objects: pre-loaded datasets
- 5. R objects: vectors
- 6. Basic data types
- 7. Setting up a working directory

Write readable code

Write your code in a clear, understandable way

- 1) for "future you", i.e keep your R code as a script for future reuse
- 2) for other people who might want to understand or use your code

There are several simple rules that can significantly improve the readability of your code:

Write comments!

R ignores lines that start with #, so you can write anything you want and it won't affect your code, aim at explaining why something was done

- Put spaces between and around operators (= + * /) and object names
- Break up long lines of code
- Use meaningful object names made of two or three words (object names are CASE sensitive)
- Follow a consistent style for naming things

Let's create our first R script!

Functions and their arguments

- Functions in R allow to do some sophisticated operations and tasks. To use a function, just write the name of the function and in parentheses the data you want to operate the function on:
 - -> seq(56)
- Every function has a Help page with the description and usage examples
 - -> Help pages -> ?rm()
- The arguments of a function are the set of inputs it accepts. Some of the inputs will be used to calculate the output, while some might be different options that affect how the calculation happens
 - -> mean(), mean(x)

R objects

- Storing results (<-) assign operator
- You can save data by storing it inside an R object
- An object is a name that you can use to store and later retrieve stored data
- R comes with many toy data sets that are pre-loaded

Objects: naming conventions

You can use anything you want to name an object, but there are a few rules:

- a name cannot start with a number
- a name cannot use **some special symbols**, like ^, !, \$, @, +, -, /, or *
- if you use the same name, **R will overwrite any previous information** stored in an object without asking you for permission -> do not use names that are already taken

Good names	Names that cause errors
a	1trial
b	\$
FOO	^mean
my_var	2nd
.day	!bad

R objects: vectors and lists

• The most basic type of R object is a vector

For example, with the : operator you can create a vector, a one-dimensional set of numbers ->

1:56

An atomic vector is just a simple vector of data.

R recognizes six basic types of atomic vectors: **doubles, integers, characters, logicals, complex, and raw**

• There is really only one rule about vectors in R: A vector can only contain objects of the same class.

But! A list is represented as a vector but can contain objects of different classes.

- Each type of atomic vector has its own convention. R will recognize the convention and use it to create an atomic vector of the appropriate type
- If you have more than more than one element in your vector, you can combine an element with the c function

R data types

R has five basic or "atomic" classes of objects:

- character: "a", "name"
- numeric: 2, 15.5
- **integer:** i_{2L} (the L tells R to store this as an integer)
- logical: TRUE, FALSE
- **complex:** 1+41 (complex numbers with real and imaginary parts)

R provides many functions to examine features of vectors and other objects, for example

- class() what kind of object is it (high-level)?
- typeof() what is the object's data type (low-level)?
- length() how long is it? What about two dimensional objects?
- attributes() does it have any metadata?

Workspace and working directory

You can explicitly check your working directory with: getwd()

Setting the directory:

- 1. Command line: setwd ("~/myRprojects")
- 2. RStudio Files pane
- 3. The best: R projects

Keeping all the files associated with a project organized together – input data, R scripts, analytical results, figures – is such a wise and common practice that RStudio has built-in support for this via its projects.

To create a project do this: *File > New Project....* The directory name you choose here will be the project name.

Create an RStudio project for an analytical project

- Keep inputs there (we'll soon talk about importing)
- Keep scripts there; edit them, run them in bits or as a whole from there
- Keep outputs there (like the PDF written above)

Things to do before the next session

- Set up the working environment and the course environment
- A nice tutorial about programming basics in R to revise everything we've covered today:

https://rstudio.cloud/learn/primers/1.2

Thank you!