TDS
Facility

Introduction to R for flow cytometry data analysis Day 1

João Lourenço, Tania Wyss & Nadine Fournier

Translational Data Science – Facility

SIB Swiss Institute of Bioinformatics

With slides from Diana Marek, Thomas Junier, Wandrille Duchemin, Leonore Wigger

From: First steps with R in Life Sciences

The Translational Data Science Facility

- Part of the **SIB Swiss Institute of Bioinformatics**
- Located at the AGORA Cancer Research Center in **Lausanne**
- Provides **statistics, bioinformatics and computational expertise** to molecular biology and applied research labs.
- Participates in fundamental and translational research by providing expertise in **data analysis** of single-cell and bulk multi-omics, spatial transcriptomics, flow cytometry, etc

For core facility service inquiry: nadine.fournier@sib.swiss
https://agora-cancer.ch/scientific-platforms/translational-data-science-facility/
https://www.sib.swiss/raphael-gottardo-group

Tell us about yourself !

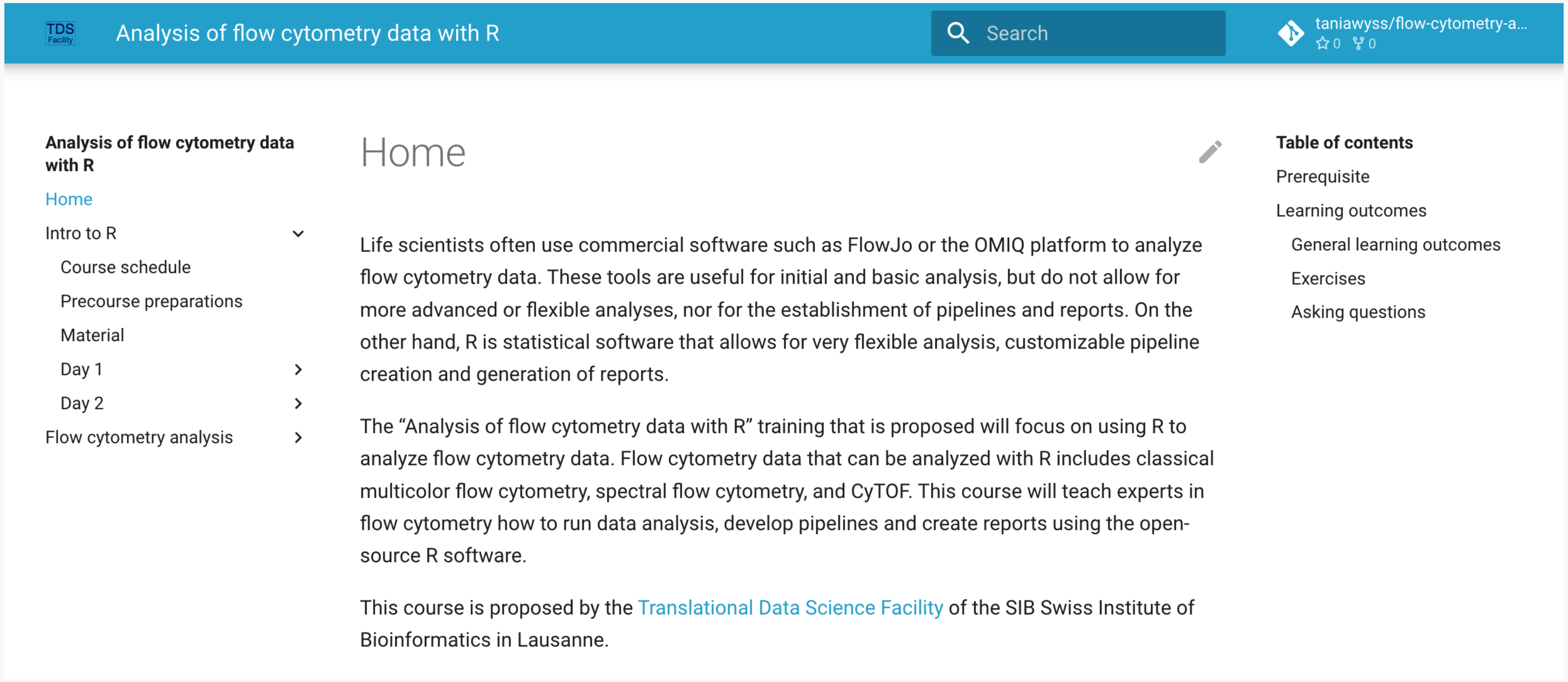
Share about yourself and your research, experience with programming, etc

Photo by National Cancer Institute, Unsplash

Photo by Scott Graham, Unsplash

Course material

1. Website
https://taniawyss.github.io/flow-cytometry-analysis-with-R


TDS Facility
Analysis of flow cytometry data with R
Search
taniawyss/flow-cytometry-a...
☆0 ⑂0

Home

Life scientists often use commercial software such as FlowJo or the OMIQ platform to analyze flow cytometry data. These tools are useful for initial and basic analysis, but do not allow for more advanced or flexible analyses, nor for the establishment of pipelines and reports. On the other hand, R is statistical software that allows for very flexible analysis, customizable pipeline creation and generation of reports.

The "Analysis of flow cytometry data with R" training that is proposed will focus on using R to analyze flow cytometry data. Flow cytometry data that can be analyzed with R includes classical multicolor flow cytometry, spectral flow cytometry, and CyTOF. This course will teach experts in flow cytometry how to run data analysis, develop pipelines and create reports using the open-source R software.

This course is proposed by the Translational Data Science Facility of the SIB Swiss Institute of Bioinformatics in Lausanne.

Table of contents

2. Google doc for exchange of additional information and questions

Outline & Schedule

Day 1 (morning)

01

Introduction to R and the RStudio environment
Exercises
(9:00 – 10:30)
10:30 – 10:50 Coffee break

02

Working with scripts files
Exercises
(10:50 -12:00)
12:00 – 13:00 Lunch break

Outline & Schedule

Day 1 (afternoon)

03 **Data types and data structures**
Exercises
(13:00 – 15:30)
15:30 -15:50 Coffee break

04 **Importing, formating and exporting data with R**
Exercises
(15:50 – 16:50)

16:50 - 17:00 Feedback and end of day

Outline & Schedule

Day 2

05 Building graphics in R (basic plotting)

06 Rmarkdown and report generation

07 Starting to work with flow cytometry data

Examples and exercises are integrated in the chapters

Questions and Exercises

Feel free to interrupt with questions by asking them directly or raising your (virtual) hand.

Use the Q&A in Google Doc (or Zoom chat), we will provide answers.

Add a ✓ when you are done with the current exercise.

Exercises in R: 👍

We will try to debug as much as possible

We are happy if you share your results or alternative code!

Course Content

R is vast and can't be learned overnight. The scope of this course:

- basic understanding and concepts behind R
- implement and interpret a data analysis workflow
- the example data was specifically chosen to reflect the type of data handled for flow cytometry analysis

This course is only the first step in your R journey!

01 Introduction to R and the RStudio environment

What is R ?

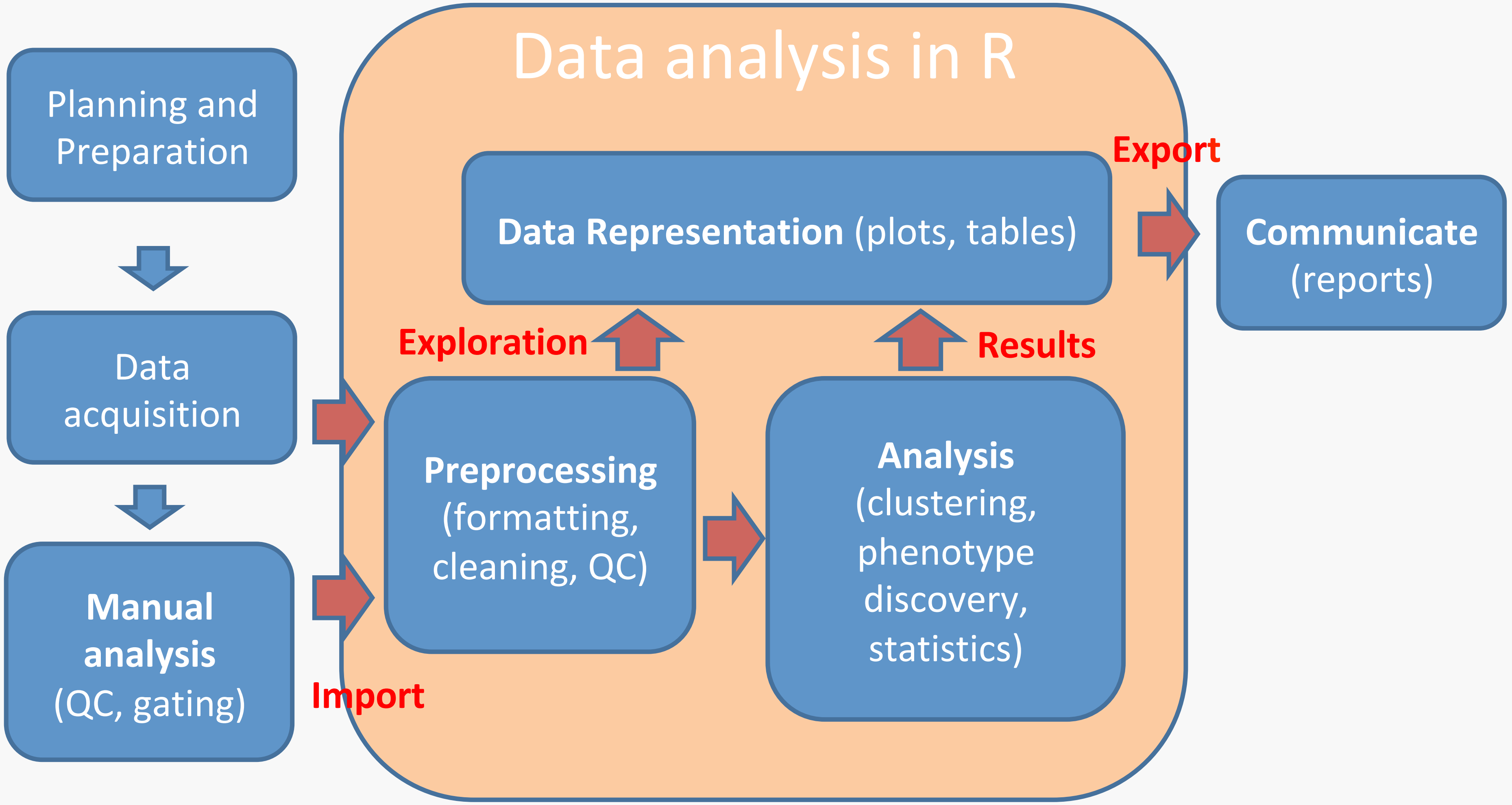
- R is a programming language and an environment for statistical computation and graphics.

  - A simple development environment with a console and a text editor

  - Facilities for data import, manipulation and storage
  - Functions for calculations on vectors and matrices
  - Large collections of data analysis tools
  - Graphical tools

https://www.r-project.org/

Taking Advantage of R For Your Work


Planning and Preparation
Data acquisition
Manual analysis (QC, gating)
Import
Data analysis in R
Preprocessing (formatting, cleaning, QC)
Analysis (clustering, phenotype discovery, statistics)
Exploration
Results
Data Representation (plots, tables)
Export
Communicate (reports)

R's user community

- Group of core developers who maintain and upgrade the basic R installation. New version every 6 months.

- Anyone can contribute with add-on packages which provide additional functionality (thousands of such packages available) and help for each function.

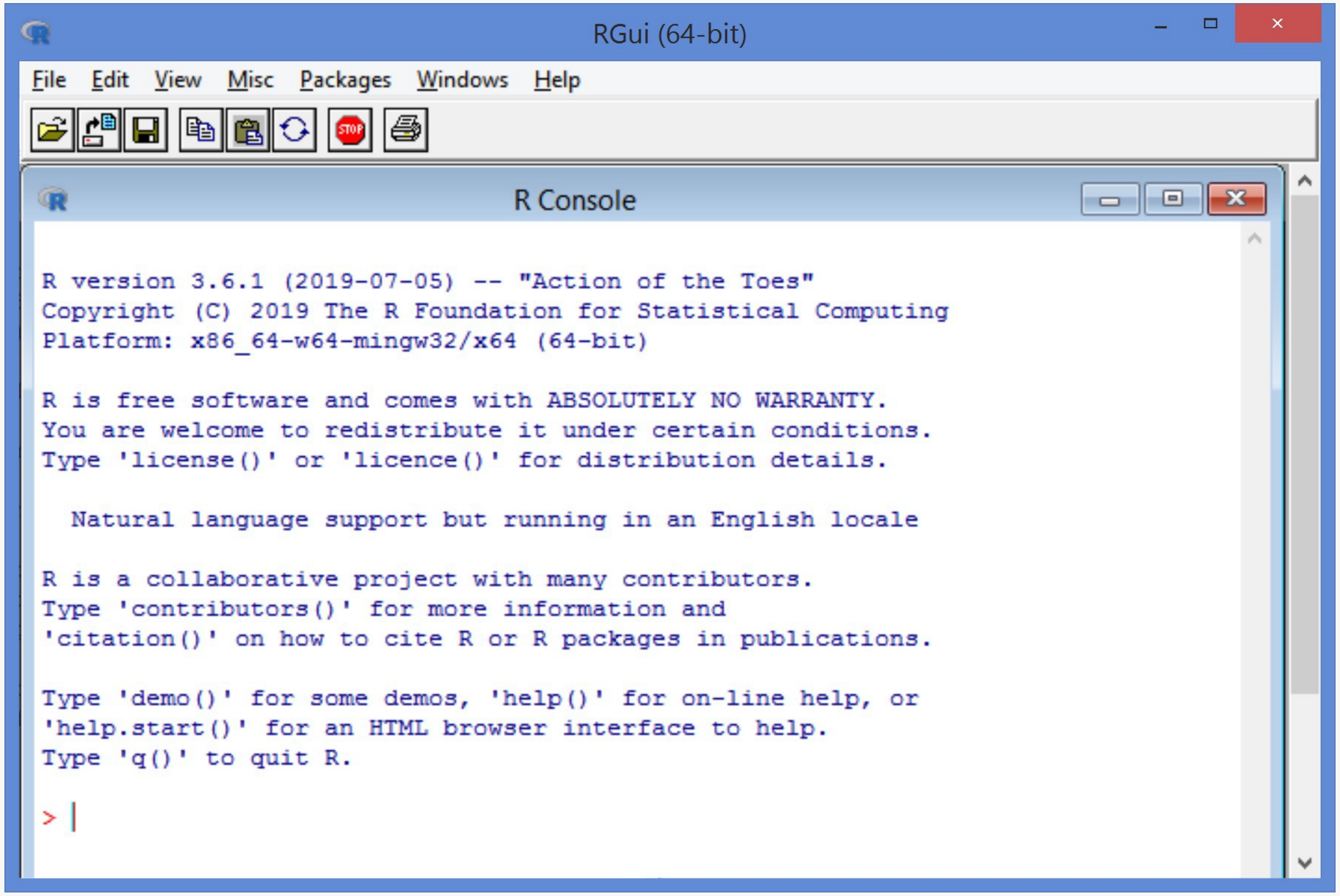
- Online help

  - in user group forums, *eg*:
  https://stat.ethz.ch/mailman/listinfo/r-help
  http://stackoverflow.com/questions/tagged/r

  - in countless online tutorials, books, blogs

RGui (R Graphical user interface)

- Together with the programming language, a (minimal) graphical user interface is installed.



```
R version 3.6.1 (2019-07-05) -- "Action of the Toes"
Copyright (C) 2019 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (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 Combined with RStudio

https://posit.co/products/open-source/rstudio/

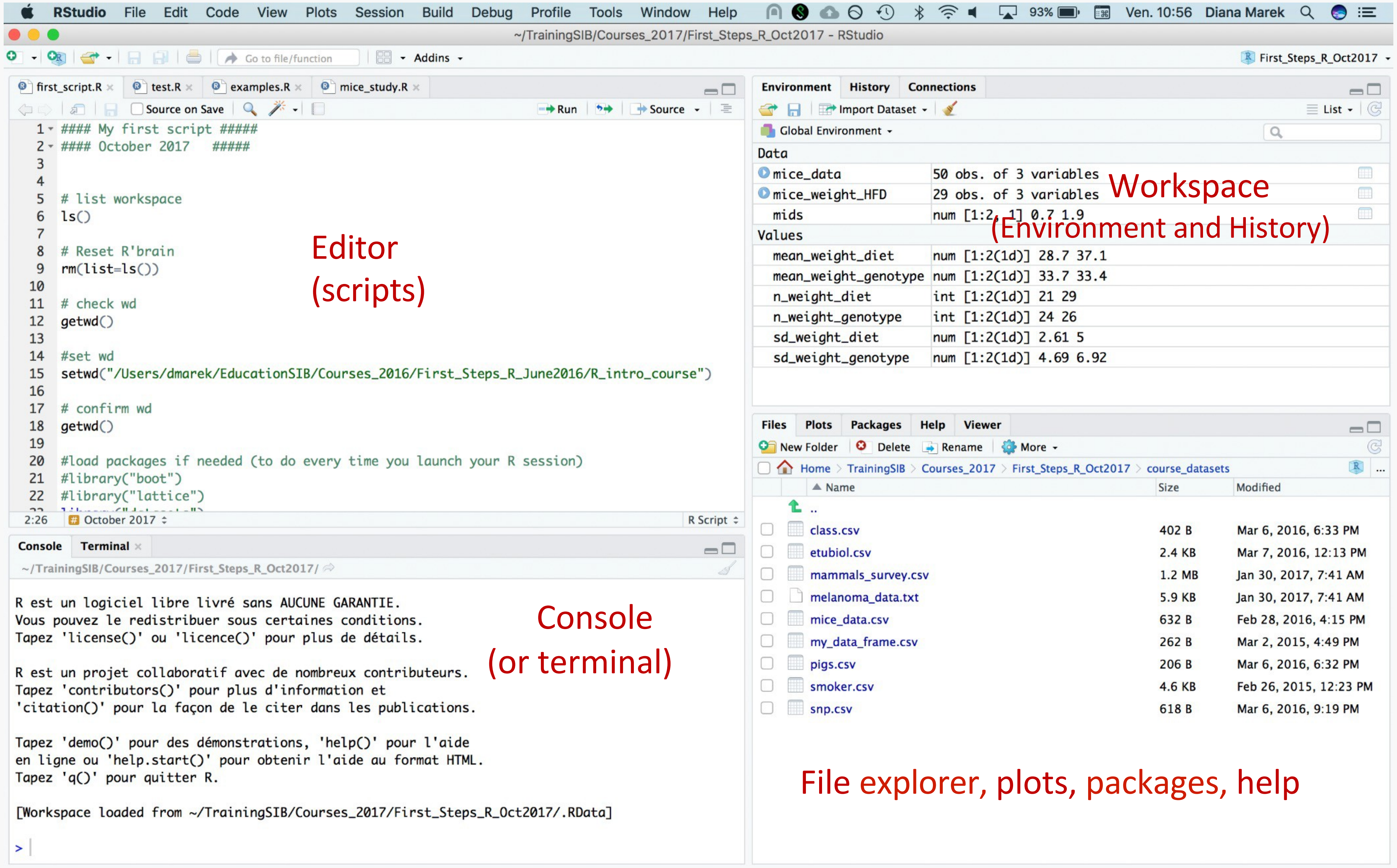
RStudio is an integrated development environment (IDE), designed to help you be more productive with R

It includes:

- A console
- A syntax-highlighting editor that supports direct code execution
- Tools for viewing the workspace and the history
- A file explorer, a package explorer, plot and help display areas

We suggest RStudio as a more powerful, more comfortable alternative to the RGUI

RStudio interface



RStudio File Edit Code View Plots Session Build Debug Profile Tools Window Help
93% Ven. 10:56 Diana Marek
~/TrainingSIB/Courses_2017/First_Steps_R_Oct2017 - RStudio
Go to file/function
Addins
First_Steps_R_Oct2017
first_script.R test.R examples.R mice_study.R
Source on Save
Run
Source
2:26
October 2017
R Script
Console Terminal
~/TrainingSIB/Courses_2017/First_Steps_R_Oct2017/
Environment History Connections
Import Dataset
List
Global Environment
Files Plots Packages Help Viewer
New Folder Delete Rename More
Home > TrainingSIB > Courses_2017 > First_Steps_R_Oct2017 > course_datasets
Editor
(scripts)
Console
(or terminal)
Workspace
(Environment and History)
File explorer, plots, packages, help

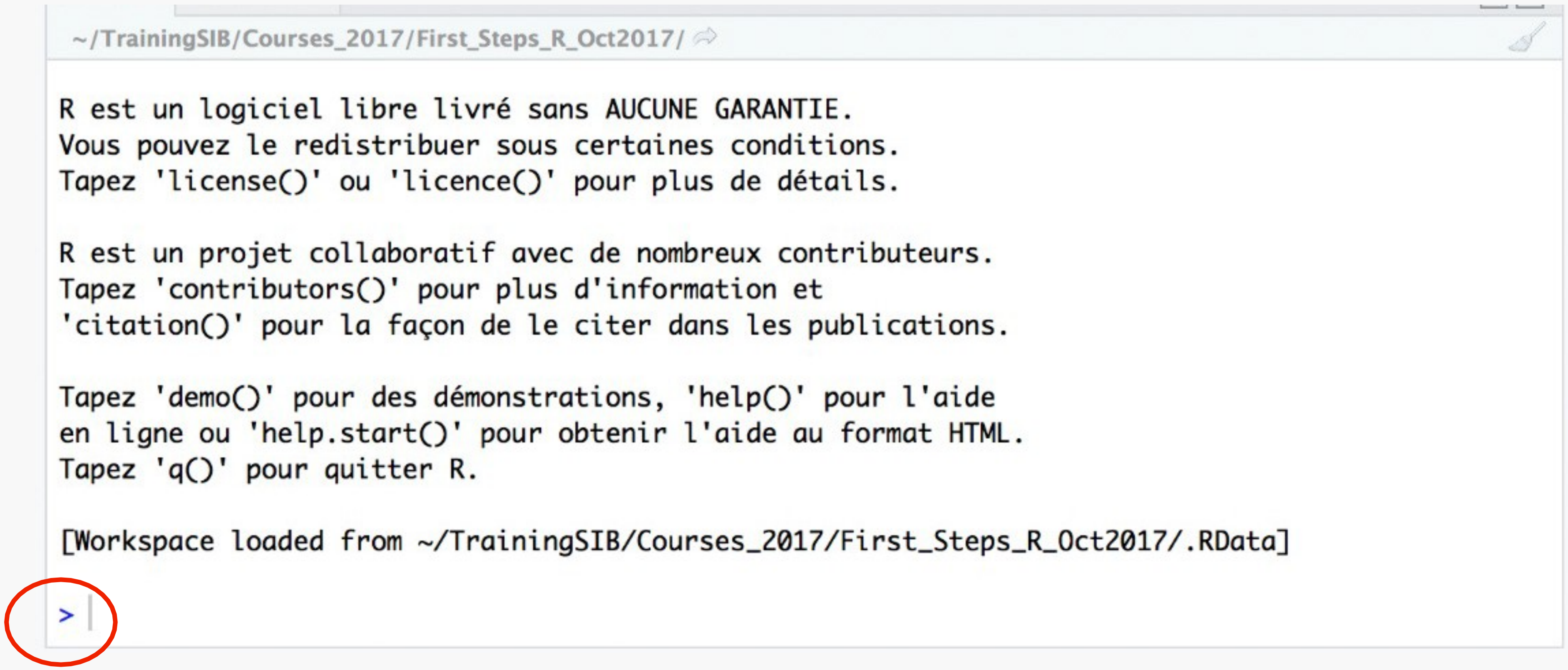
```
#### My first script #####
#### October 2017    #####


# list workspace
ls()

# Reset R'brain
rm(list=ls())

# check wd
getwd()

#set wd
setwd("/Users/dmarek/EducationSIB/Courses_2016/First_Steps_R_June2016/R_intro_course")

# confirm wd
getwd()

#load packages if needed (to do every time you launch your R session)
#library("boot")
#library("lattice")
```

Data	
mice_data	50 obs. of 3 variables
mice_weight_HFD	29 obs. of 3 variables
mids	num [1:2, 1] 0.7 1.9
Values	
mean_weight_diet	num [1:2(1d)] 28.7 37.1
mean_weight_genotype	num [1:2(1d)] 33.7 33.4
n_weight_diet	int [1:2(1d)] 21 29
n_weight_genotype	int [1:2(1d)] 24 26
sd_weight_diet	num [1:2(1d)] 2.61 5
sd_weight_genotype	num [1:2(1d)] 4.69 6.92

Name	Size	Modified
..		
class.csv	402 B	Mar 6, 2016, 6:33 PM
etubiol.csv	2.4 KB	Mar 7, 2016, 12:13 PM
mammals_survey.csv	1.2 MB	Jan 30, 2017, 7:41 AM
melanoma_data.txt	5.9 KB	Jan 30, 2017, 7:41 AM
mice_data.csv	632 B	Feb 28, 2016, 4:15 PM
my_data_frame.csv	262 B	Mar 2, 2015, 4:49 PM
pigs.csv	206 B	Mar 6, 2016, 6:32 PM
smoker.csv	4.6 KB	Feb 26, 2015, 12:23 PM
snp.csv	618 B	Mar 6, 2016, 9:19 PM

```
R est un logiciel libre livré sans AUCUNE GARANTIE.
Vous pouvez le redistribuer sous certaines conditions.
Tapez 'license()' ou 'licence()' pour plus de détails.

R est un projet collaboratif avec de nombreux contributeurs.
Tapez 'contributors()' pour plus d'information et
'citation()' pour la façon de le citer dans les publications.

Tapez 'demo()' pour des démonstrations, 'help()' pour l'aide
en ligne ou 'help.start()' pour obtenir l'aide au format HTML.
Tapez 'q()' pour quitter R.

[Workspace loaded from ~/TrainingSIB/Courses_2017/First_Steps_R_Oct2017/.RData]

>
```

Console: The Command Line



```
~/TrainingSIB/Courses_2017/First_Steps_R_Oct2017/

R est un logiciel libre livré sans AUCUNE GARANTIE.
Vous pouvez le redistribuer sous certaines conditions.
Tapez 'license()' ou 'licence()' pour plus de détails.

R est un projet collaboratif avec de nombreux contributeurs.
Tapez 'contributors()' pour plus d'information et
'citation()' pour la façon de le citer dans les publications.

Tapez 'demo()' pour des démonstrations, 'help()' pour l'aide
en ligne ou 'help.start()' pour obtenir l'aide au format HTML.
Tapez 'q()' pour quitter R.

[Workspace loaded from ~/TrainingSIB/Courses_2017/First_Steps_R_Oct2017/.RData]

> 
```

The prompt ">" indicates that R is waiting for you to type a command

Try it out...

Type the following at the command prompt:

Simple calculations

```
> 1 + 1
```

Assign values to a variable names

```
> x  <- 128.5
```

Display content of variables

```
> x
```

Pre-defined functions

```
> abs(-11)
```

After each command, hit the return key.

This causes R to execute it.

R Key Concepts

- **Variable: A storage space in memory that has a name and can hold data.**

  ```
  temp <- -5.5    # Create a variable named temp, holding value -5.5
  ```

- **Numeric constant:** a number, such as 128.5
- **Character constant:** a text sequence, such a "Hello" (enclosed in quotes)

- **Function: pre-written code that performs a specific task and can be executed by "calling" the function.**

  Write the function's name, followed by parentheses. Inside the parenthesis, pass variables or values to the function code (function arguments).

  ```
  abs(temp)        # the absolute value of temp
  log2(16)         # the base 2-logarithm of 16
  q()              # quit R  (no function arguments necessary)
  ```

- **Operator:** a special function for arithmetic, logical or other operations.

  Examples of arithmetic operators: +, -, *, /, ^, …

Creating an R Project

- RStudio projects make it straightforward to divide your work into multiple contexts, each with their own working directory, workspace, history, and source documents.
- You can create an RStudio project in a brand new directory or an existing directory where you already have your data

  - Go to File > New project or click on “Project” in the upper right corner of RStudio.
  - Choose New Directory, then New Project, give a name to the directory and set its location.
  - This creates a new directory which contains a .Rproj file (same name as the directory).

  - OR choose Existing Directory, click Browse, navigate to a folder, then click Create Project
  - This creates an .Rproj file inside the directory (same name as the directory).

The project directory becomes automatically the working directory.

This is one of the ways RStudio adds convenience

Let’s practice – 1

1) Outside Rstudio: Prepare course data for exercises

Download the course material for day 1 from :

https://taniawyss.github.io/flow-cytometry-analysis-with-R/introR/material/

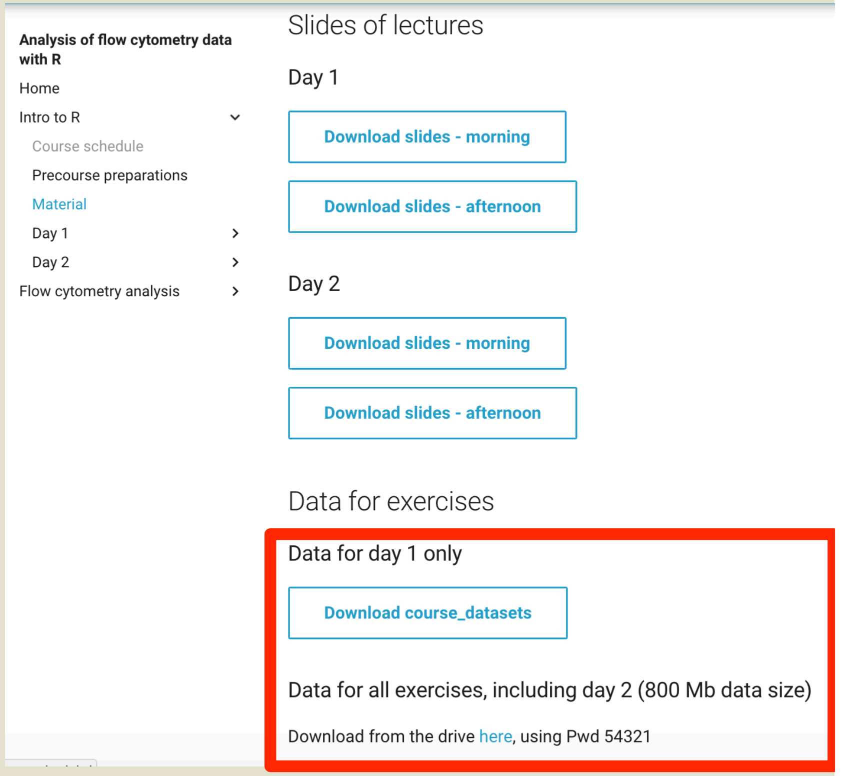
For day 1 + day 2 data (slower, launch during break) :

from : https://drive.switch.ch/index.php/s/Nb91u9CTiOghq6w

and **unzip** then **move** folder where you want it on your computer and where the Rproj will be created.

2) Inside Rstudio: Project set-up

Within RStudio, create a new project in an existing directory, and save it within the folder where the course material was downloaded.



Data for all exercises, including day 2 (800 Mb data size)

Download from the drive here, using Pwd 54321

Working Directory

R can read and write files. The working directory is the folder on your computer where it will look for files if you don’t specify the path.

See the current working directory:

```
> getwd()
[1] "C:/Users/lwigger/Documents/Rcourse2022"
```

Change the working directory to any existing folder on your hard drive or system using setwd() and the file path, e.g.

```
> setwd("D:/R_exercises/")
```

In an RStudio project, we usually do not need to change the working directory

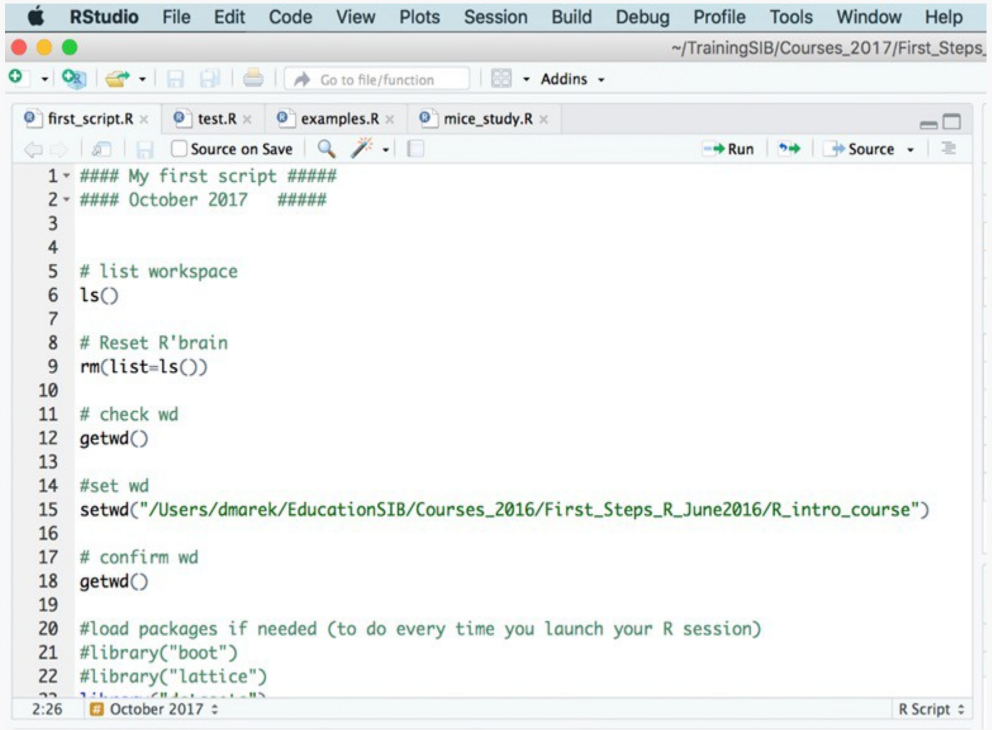
02 Working with script files

Editor: Write code to a script file

A script is a file that contains commands to be executed in succession.

Write your code into a script and save it

- to have documentation later of what you did
- to be able to re-use the code and create variations
- for easy execution


RStudio File Edit Code View Plots Session Build Debug Profile Tools Window Help
~/TrainingSIB/Courses_2017/First_Steps_
Go to file/function
Addins
first_script.R
test.R
examples.R
mice_study.R
Source on Save
Run
Source
2:26
October 2017
R Script

```
#### My first script #####
#### October 2017    #####


# list workspace
ls()

# Reset R'brain
rm(list=ls())

# check wd
getwd()

#set wd
setwd("/Users/dmarek/EducationSIB/Courses_2016/First_Steps_R_June2016/R_intro_course")

# confirm wd
getwd()

#load packages if needed (to do every time you launch your R session)
#library("boot")
#library("lattice")
```

Notice the syntax highlighting

Create a new script and type code

- Create a new script using File > New File > R script. **Don't forget to save your script often.**
- By default, scripts are saved to the working directory.
- Files can be saved to other locations (File -> Save As…)
- Start **Typing code** at the top of the script

```
# My first program and command:
2 + 3
```

- **Notice the syntax highlighting**
- **Comments** : “#” at the beginning of a line or before a command: helping text ; everything that follows is ignored by the during executing ; R does not support multi-line comments

Send Code From a Script to the Console

Run individual lines, one by one:

- In RStudio: put the cursor anywhere in a line, hit

  Ctrl + enter (Windows)

  Cmd + return (Mac)

  or click the "Run" button

Tip: Run part of a line or multiple lines: Highlight the code, then proceed as above

Save, close and open scripts

- **Save a script:** File > Save or 💾
- **Close and open a script:** File > Close and File > Open File

Tips:

- Most of your code should be developed and saved in scripts.
  - You can execute individual lines of code interactively while you are writing it.
  - You can run the entire script once it is ready and debugged.

Workspace

The workspace is the internal memory where R stores the objects you created during the session.

Explore your workspace using the command line:

- To list what is in your workspace, type

```
> ls()
```

- To remove (delete) an object from the workspace, use function rm:

```
> rm(x)
```

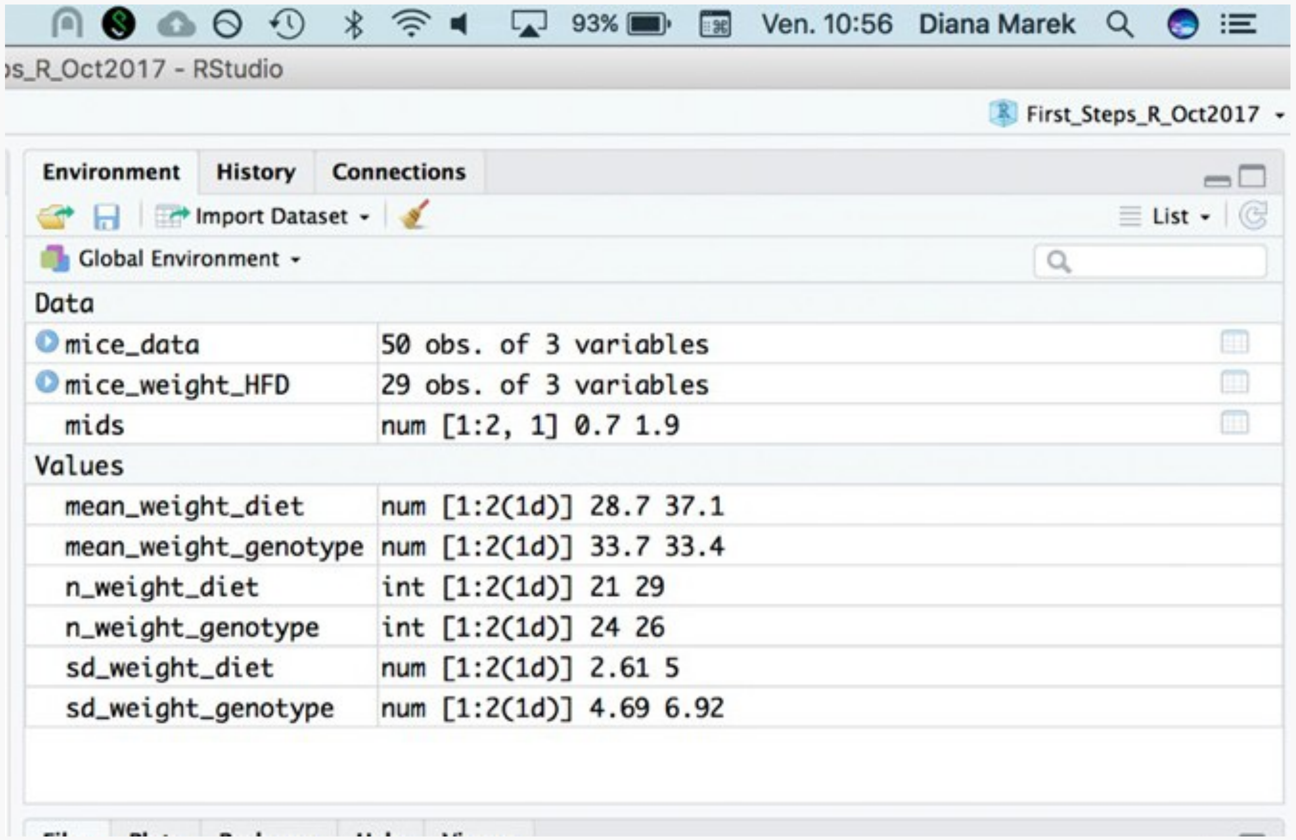
- To remove (delete) all objects from the workspace, type

```
> rm(list=ls())
```

Workspace in RStudio

Explore your workspace using Rstudio's GUI:

- See the upper right quadrant, tab "Environment": all objects are listed
- To remove all objects form the workspace, click the broom icon.



```
Data
mice_data             50 obs. of 3 variables
mice_weight_HFD       29 obs. of 3 variables
mids                  num [1:2, 1] 0.7 1.9
Values
mean_weight_diet      num [1:2(1d)] 28.7 37.1
mean_weight_genotype  num [1:2(1d)] 33.7 33.4
n_weight_diet         int [1:2(1d)] 21 29
n_weight_genotype     int [1:2(1d)] 24 26
sd_weight_diet        num [1:2(1d)] 2.61 5
sd_weight_genotype    num [1:2(1d)] 4.69 6.92
```

Let’s practice – 2

1) Prepare your first script

- Open a script file and save it with file name "`ex1.R`"
- Comment it (#, pound sign at the beginning of the line).
- Type or paste the following code:

```
# First Steps, ex 1
w <- 3
h <- 0.5
area <- w * h
area
```

2) Look at the script (before trying to run it)

- Can you understand each line? What do you expect it to print to the console?

3) Run the script and explore RStudio features

- Run the script line by line. Try both the "Run" button and the keyboard shortcut. Watch variables appear in the Environment panel (top right).
- Watch what is printed to the console (bottom left). Does it match your expectation?

Closing or Switching Projects

- Close a project:
  File -> Close

- Switch to another project, which will close the current one:
  File -> Open Project...

- Open another project and keep the current one open as well:
  File -> Open Project in New Session...

Reopening an R Project from a File

You can access your R project directly from your hard drive

- Find the .Rproj file and open it (double click on many systems).
- RStudio will automatically start if it is not already running.

Workspace (.Rdata) and History (.Rhistory) Options

- When closing or switching projects, the workspace and the history are automatically cleared.

Your RStudio project can be configured to save your workspace and history to a file upon closing a project - or not.

Menu:

Tools -> Global Options -> General (set default for all projects)

Tools -> Project Options -> General (settings for one project)

CAUTION: .Rdata files can be very large.
Save only when you have space on your hard drive!

Let’s practice – 2bis

4) Look at the project options (RStudio’s *Tools* menu). If needed, modify them to save your workspace and history and to restore them at startup.

Check if this works:

- Close RStudio.
- In your course folder, (double-)click the .rproj file.
- Does your project open? Are your variables still in the Environment?

Check if this works, too:

- Close RStudio.
- Open RStudio again.
- Verify that your project is currently closed. How do you see this?
- From inside RStudio, open your project. Are your variables now in the Environment?

Packages

- Sets of related functions (and sometimes data sets)
- A small number of packages are part of the basic R installation.
- Many, many packages are developed by the user community and can be installed later as needed.

- There are two (three) main repositories that provide R packages of interest in the life sciences. Their content can be browsed on the web :
  - CRAN (Comprehensive R Archive Network, http://cran.r-project.org/) : the main R repository, with over 18600 packages (June 2022).
  - Bioconductor (http://www.bioconductor.org) : a separate project specialized in the analysis of high-throughput genomic and transcriptomic data, with 2040 packages (June 2022).
  - (Github (https://github.com/): not restricted to hosting R packages).

Installing packages from CRAN

Install packages from CRAN with the install.packages() function

```
>install.packages("stringi") # character string
manipulations
```

Installation necessary packages only once until you upgrade R to a new version.

Loading functions from packages

Once a package is installed, its content needs to be made accessible to R.

`library()` loads the package for the current session.

It is good practice to load all needed packages at the top of a script.

```
# My Script

library(limma)
library(DESeq2)
library(MASS)
library(ggplot2)

# Here my data analysis begins
...
```

Session information

- **R.version.string** prints the currently used R version.
- **sessionInfo()** prints version information about R and attached or loaded packages.

Tip: Run sessionInfo() at the end of your data analysis session and save the output.

This information is useful when you want to redo an analysis later, generate a report, or post a question on an online forum, …

Working at the command prompt in RStudio

Shortcuts for both R console and RStudio:

- TAB key for command auto completion.
- Up and down arrows to scroll through the command history.
- Ctrl-l to clear console window.

Shortcuts specific to RStudio:

- Ctrl-1 and Ctrl-2 to jump between the script and the console windows (shift focus).

Support for incomplete statements (R console and Rstudio):

- If you hit return while your statement is incomplete, the command prompt (>) will change to +. R is waiting for you to finish writing it.
  - Keep typing and hit return again when done
  - OR hit “Esc” to abandon the unfinished command

In a Nutshell

- R and RStudio environments
  - Use of R project to gather all documents related to a project together.
  - The working directory becomes the directory of your project.
  - Possibility to save/load workspace (.Rdata file) containing your objects.
  - Possibility to save/load history of commands (.Rhistory file).
  - Help and packages available.

- Now let's get familiar with R syntax and objects.