Silesian
University
of Technology

RESEARCH
UNIVERSITY
EXCELLENCE INITIATIVE
Ministry of Science
and Higher Education

Programming for Cognitive Science

Lecture 1 – Introduction. R basics.

Joanna Tobiasz, PhD
Anna Papiez, PhD

Department of Data Science and Engineering

What to expect?

Joanna Tobiasz, PhD

joanna.tobiasz@polsl.pl

Gliwice, Faculty of Automatic Control, Electronics and Computer Science; Department of Data Science and Engineering, room 534

Office hours:

1) Thursdays: 2:00-3:00 pm
2) Fridays: 1:45-2:45 pm

01 **Course organization**

02 **Introduction to programming**

03 **R basics**

Part 1

Course organization

Organization of the course

6 weeks of classes:

- 15h of lectures – presence recommended
- 15h of laboratories (5x3h every week) – presence obligatory

Please check carefully the locations and hours every week.

Platform course: https://platforma.polsl.pl/roz/course/view.php?id=1406
Password: PfCS1_ct_2024

Rules are at the Platform.

Laboratory classes

- Attendance at the laboratory is required. Students have to attend all laboratory classes.
- Students are obligated to come on time to the laboratory classes and be prepared within the scope of the lecture.
- During the laboratory classes, students work individually.
- At the beginning of each laboratory classes, there will be the initial test to check the student's preparation for the classes. Each test will be graded from 0% to 100%. The test will be based on the material covered during previous lectures and laboratories.
- The final grade from the laboratory will be established as an average from all laboratory test grades.
- To pass the laboratory part, a student needs to achieve the final laboratory grade of 40% or higher.

Laboratory classes

- Excuses for absences are made only on the basis of a sick leave document provided by the medical doctor. In other justified cases, excuses are considered on an individual basis.
- After a sick leave, the student must make up the initial test for which he/she was absent within one week of the date of the missed laboratories or of his/her recovery.
- Any changes to the course schedule proposed by students will be discussed solely with the group leader.
- Students are required to enroll to “Programming for Cognitive Science” course at Remote Education Platform (PZE).
- Students are required to regularly check their student mail and information provided through the Remote Education Platform (PZE). Any information and announcements posted in this PZE course will be treated as having reached the students.

Test and final grade

- The test will be conducted during the last lecture.
- The test will be graded from 0% – 100%. To pass the test student needs to obtain a grade of 40% or higher.
- At first, Fg will be calculated using the following formula:

$$\boldsymbol{F_g = 0.5\, Test_{grade} + 0.5\, Lab_{grade}}$$

where:

$Test_{grade}$ – the grade from the final test.

Lab_{grade} – laboratory grade calculated as an average grade.

Course plan

1) Basics of R.
2) Graphics and visualization.
3) R for data analysis.
4) Text-mining.
5) Documentation of reproducibility.
6) R reports and web applications.
7) Helpful programming tools.

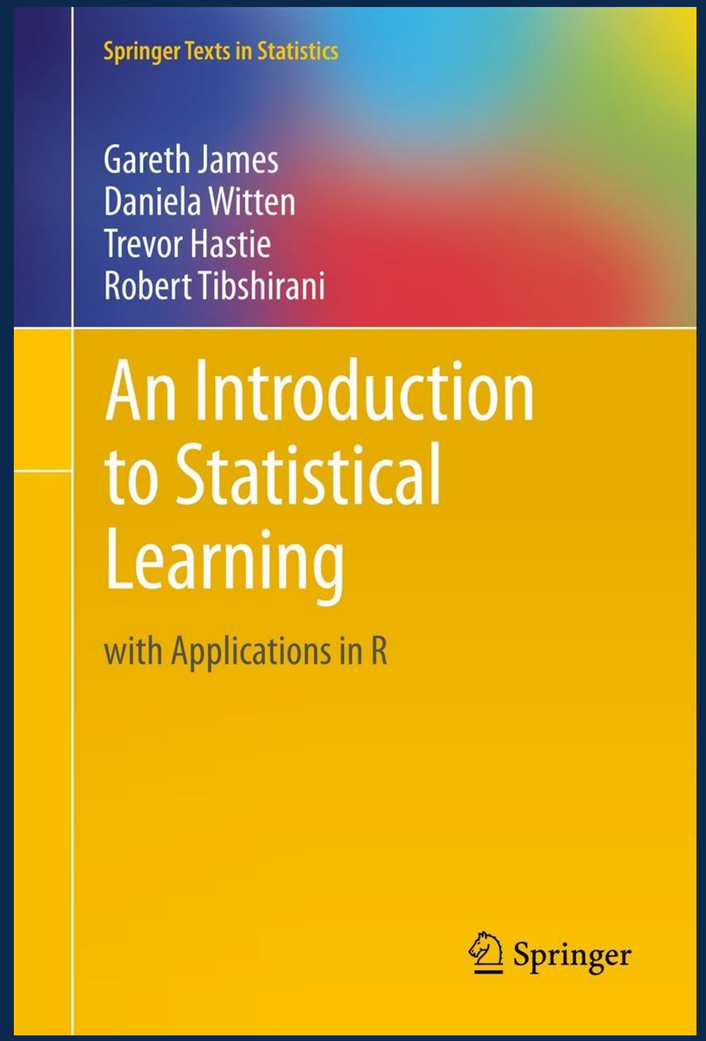
Schedule

Date	Lecture	Laboratory
12.11.2024	Lecture 1	
19.11.2024	Lecture 2	Laboratory 1: R basics
26.11.2024	Lecture 3	Laboratory 2: R basics, data visualization
03.12.2024	Lecture 4	Laboratory 3: Data analysis in R
10.12.2024	Lecture 5	Laboratory 4: Text-mining
17.12.2024	TEST + Lecture 6	Laboratory 5: Rmarkdown and Shiny

Literature/sources

James Gareth, Daniela Witten, Trevor Hastie, and Robert Tibshirani.
An introduction to statistical learning. New York: Springer, 2013.

http://www-bcf.usc.edu/~gareth/ISL/getbook.html

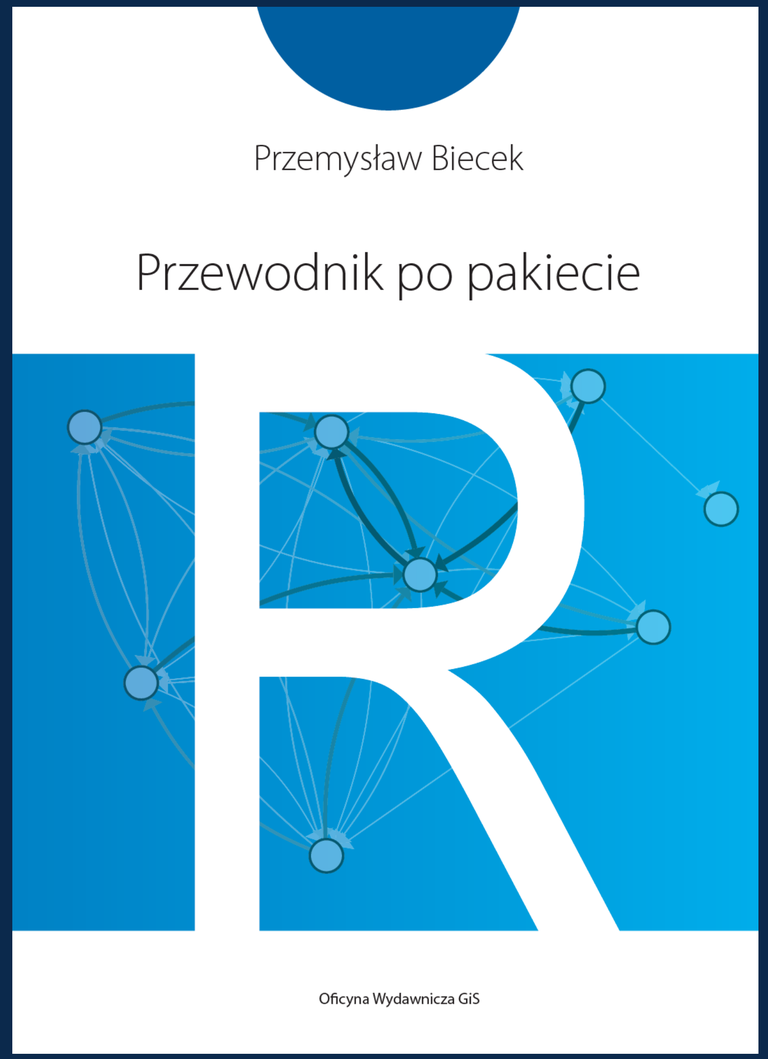


Literature/sources

Unfortunately, only in Polish:

Przemysław Biecek. Przewodnik po pakiecie R. Oficyna Wydawnicza "GIS", 2017.

http://www.biecek.pl/R/

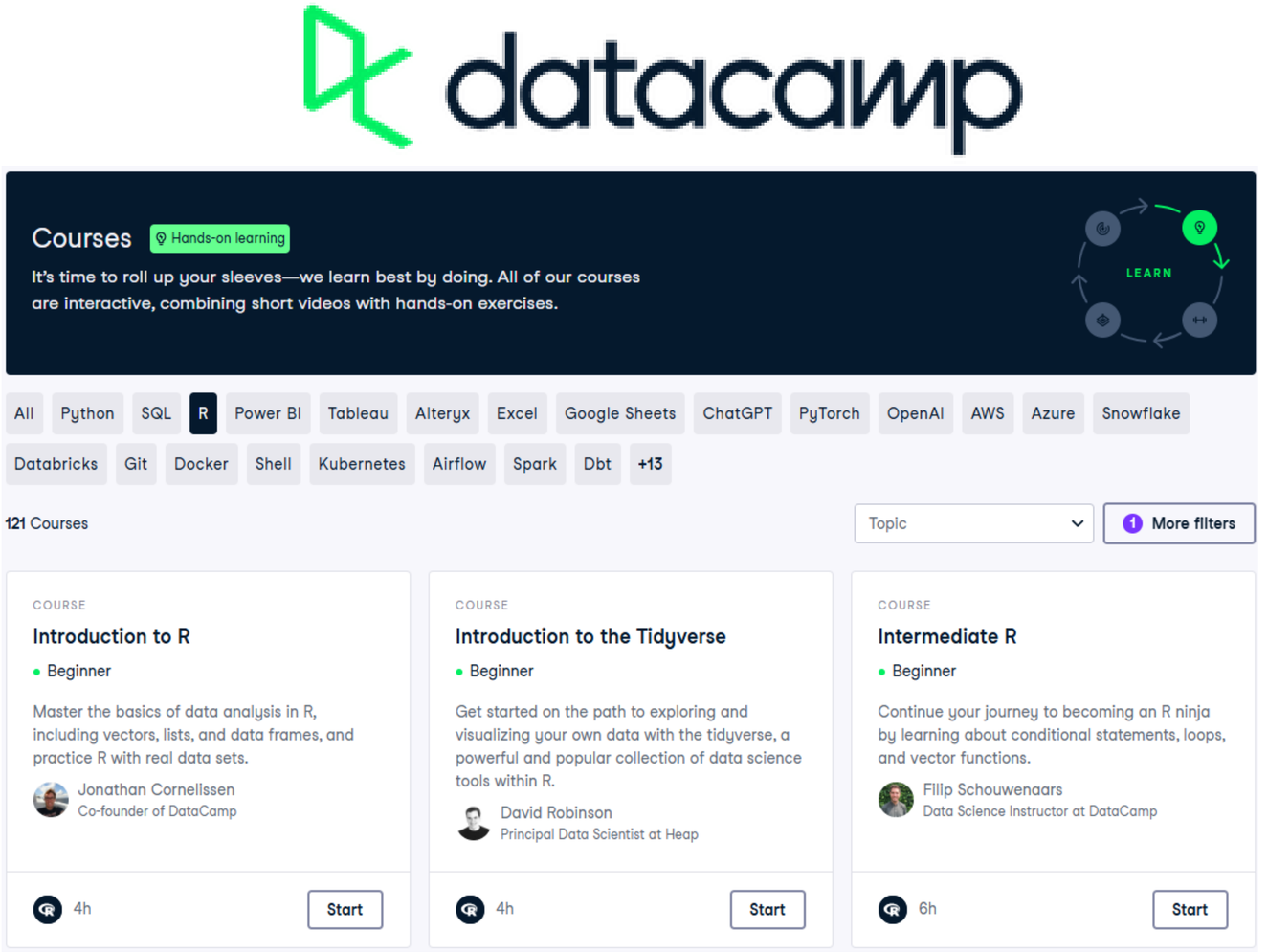


Literature/sources

DataCamp – website with online courses on coding, data analysis, and AI.

datacamp.com

Free access for our students.
Contact to Joanna Żyła, PhD (joanna.zyla@polsl.pl).



Part 2

Introduction to programming

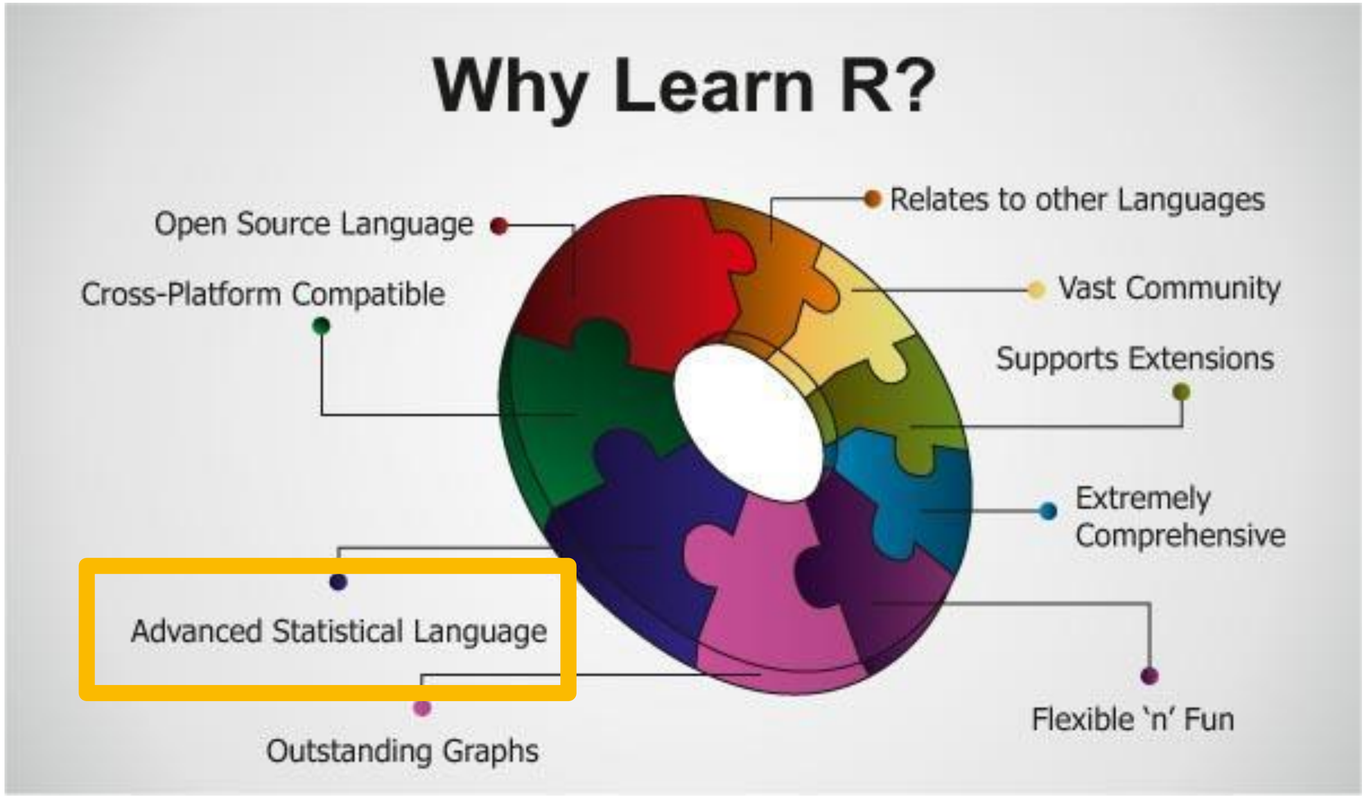
Why are we here?

- Computer programs are ubiquitous.
- The ability to implement your own algorithms is useful in any job.
- Programming makes it easier to find a job (provided you are programming at the right level).
- Fast, easy, and attractive data analysis and visualization.
- Fun! ☺

How to learn programming?

- Learning programming is not learning by heart!
- You need to understand the principles in order to apply them later to write your own programs.
- To learn programming, you must start programming.
- Reading 100 books on programming will not give practical skills.

Why are we here?



Source: https://blogs.umass.edu/gwis/2015/05/21/crash-course-in-r-programming/

Why R?

One of the greatest advantages of R: getting your work done better and in less time.

Frank Harrell
Member of R foundation, Professor of Biostatistics, Vanderbilt University

Why R?

- Intuitive and compact syntax.
- Automatic memory management.
- Support of many platforms.
- Wide range of libraries and packages.
- Open-source licence.
- Other reasons you'll have to discover yourself ☺

A little bit of history

- Language for statistical computing.
- Particularly suited for math and data analysis.
- Successor of S language – language created in 1976 for an interactive data analysis.
- Named after creators (Ross Ihaka and Robert Gentleman – R&R), from the Faculty of Statistics of Auckland University.
- They initially created its first version in 1991 for students as a statistics teaching aid.

Interpreted/script language

- Interpreter translates and executes code line by line → we can run each part of the code separately and view the results.
- Different interpreters for various hardware and OS.
- Same program for various hardware and OS.
- Other interpreted/script languages: Python, Perl, Ruby, MATLAB.

How to start?

- ✓ Interpreter

- ✓ Development environment (e.g., RStudio)

- ❑ Third-party packages

And over 10,000 more...

R

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

[Home]

Download

CRAN

R Project

The R Project for Statistical Computing

Getting Started

R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS. To **download R**, please choose your preferred CRAN mirror.

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.

https://cran.rstudio.com/

The Comprehensive R Archive Network

Download and Install R

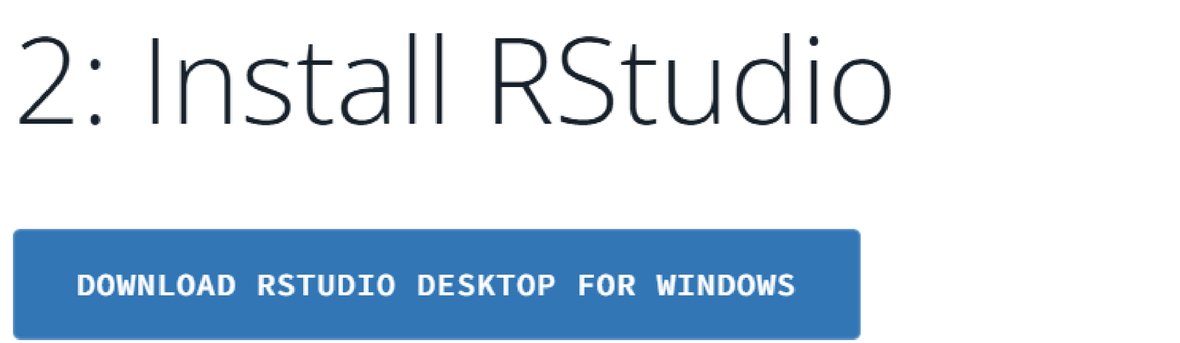
Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- Download R for Linux (Debian, Fedora/Redhat, Ubuntu)
- Download R for macOS
- Download R for Windows

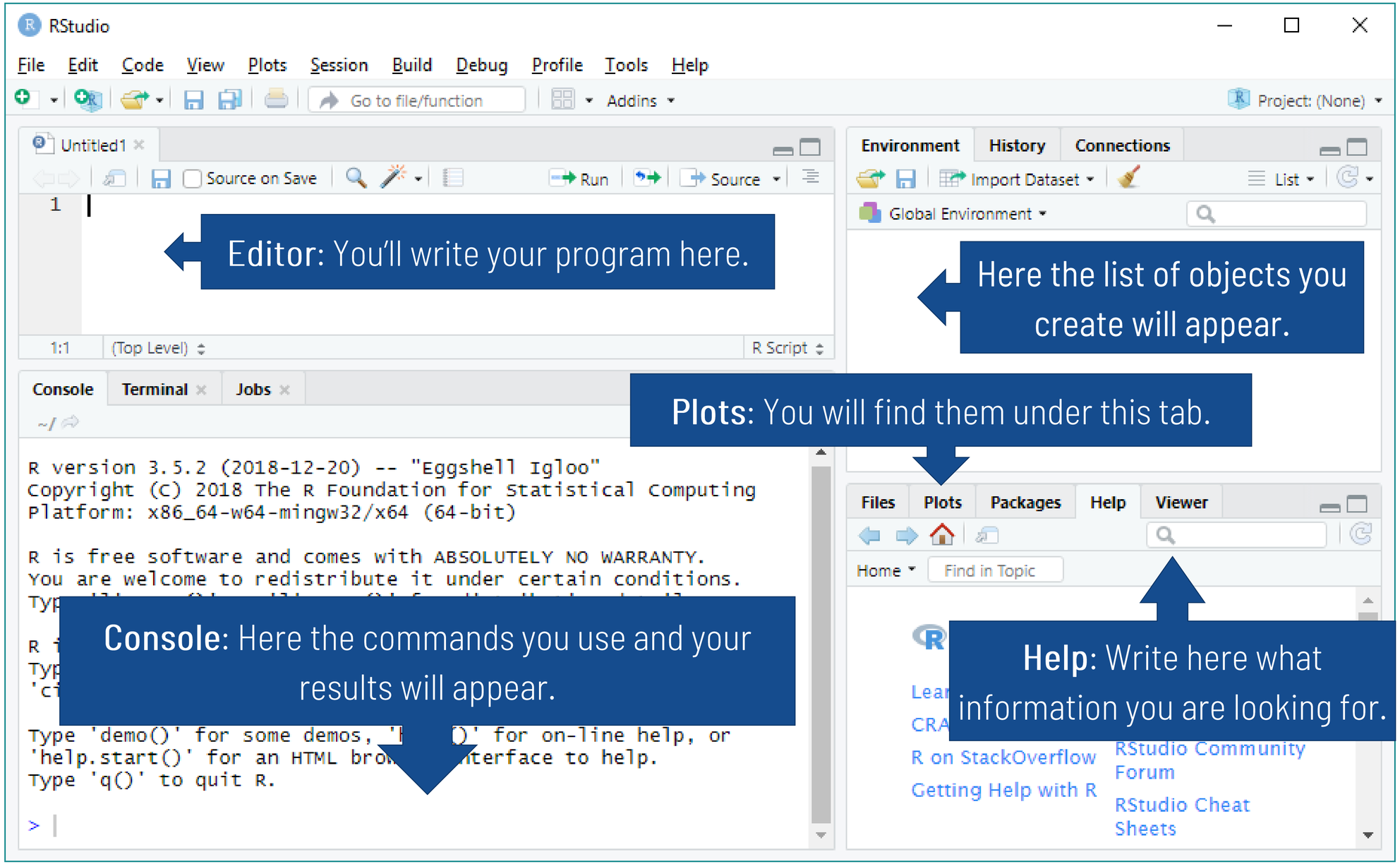
R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

RStudio

https://posit.co/download/rstudio-desktop/

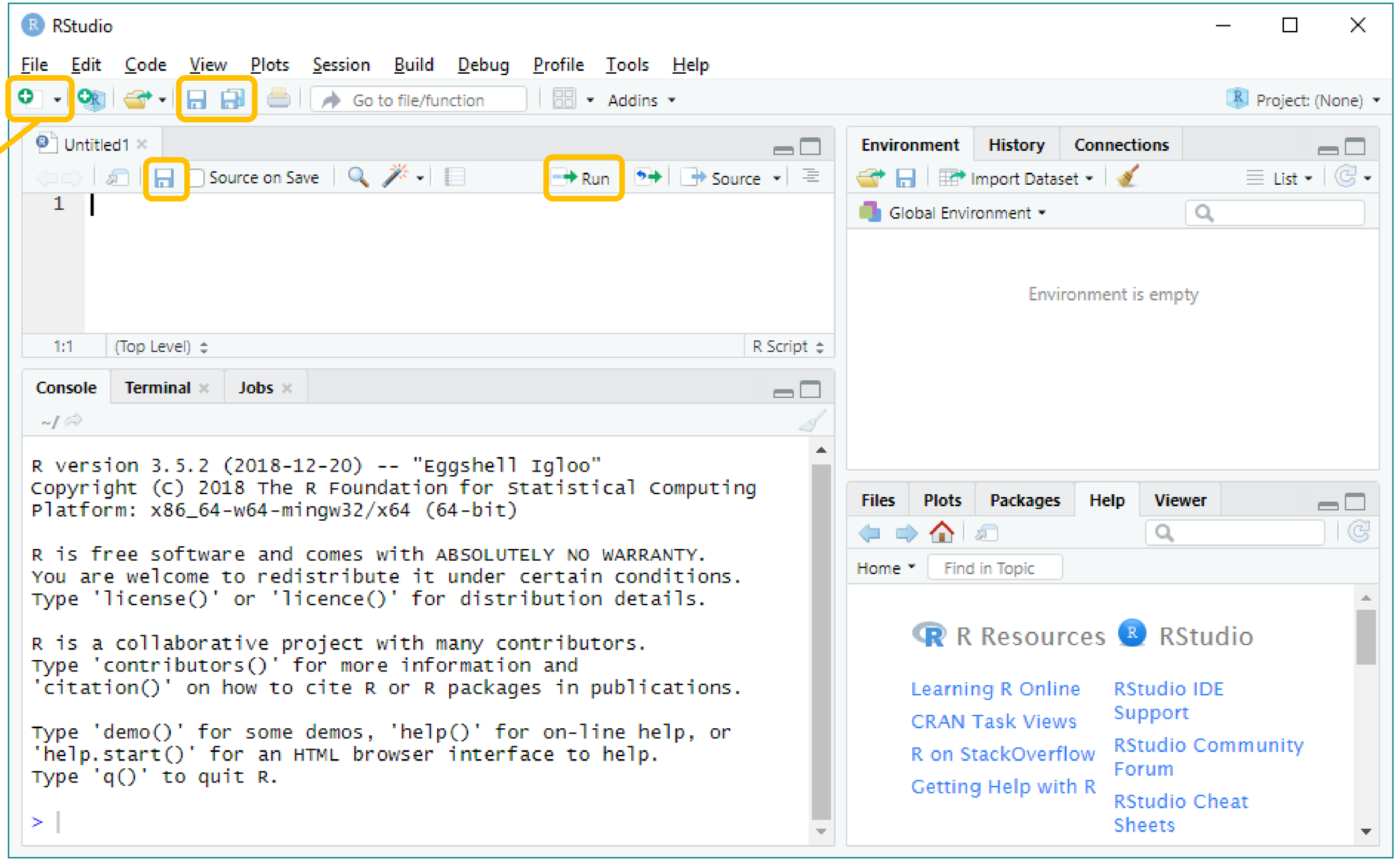


RStudio


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Untitled1
Source on Save
Run
Source
1
Editor: You'll write your program here.
1:1
(Top Level)
R Script
Environment History Connections
Import Dataset
List
Global Environment
Here the list of objects you create will appear.
Console Terminal Jobs
~/
R version 3.5.2 (2018-12-20) -- "Eggshell Igloo"
Copyright (C) 2018 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 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
>
Plots: You will find them under this tab.
Files Plots Packages Help Viewer
Home
Find in Topic
R on StackOverflow
Getting Help with R
RStudio Community Forum
RStudio Cheat Sheets
Console: Here the commands you use and your results will appear.
Help: Write here what information you are looking for.

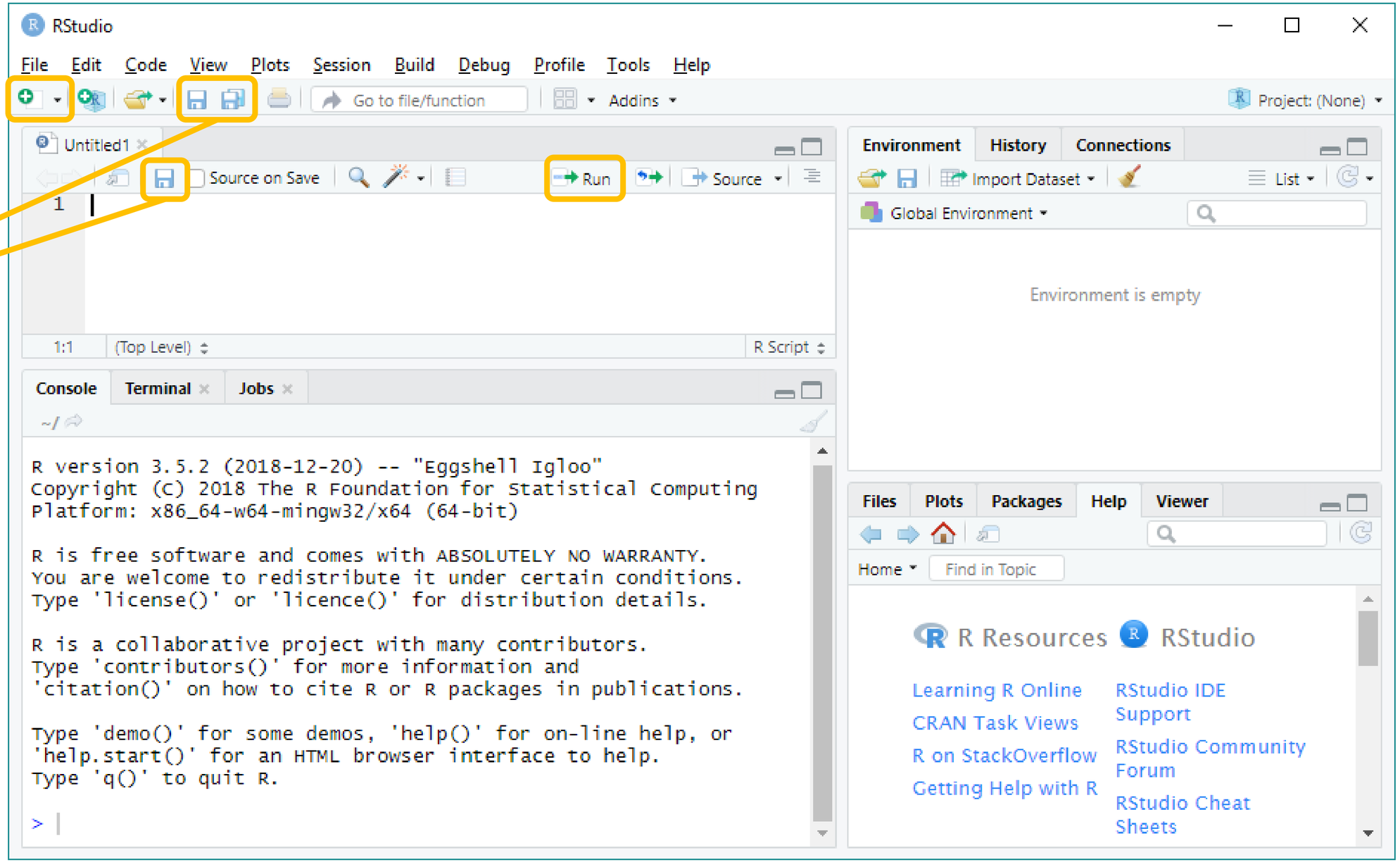
RStudio

This button is used to create a new script, i.e. a text file in which the programme is saved.



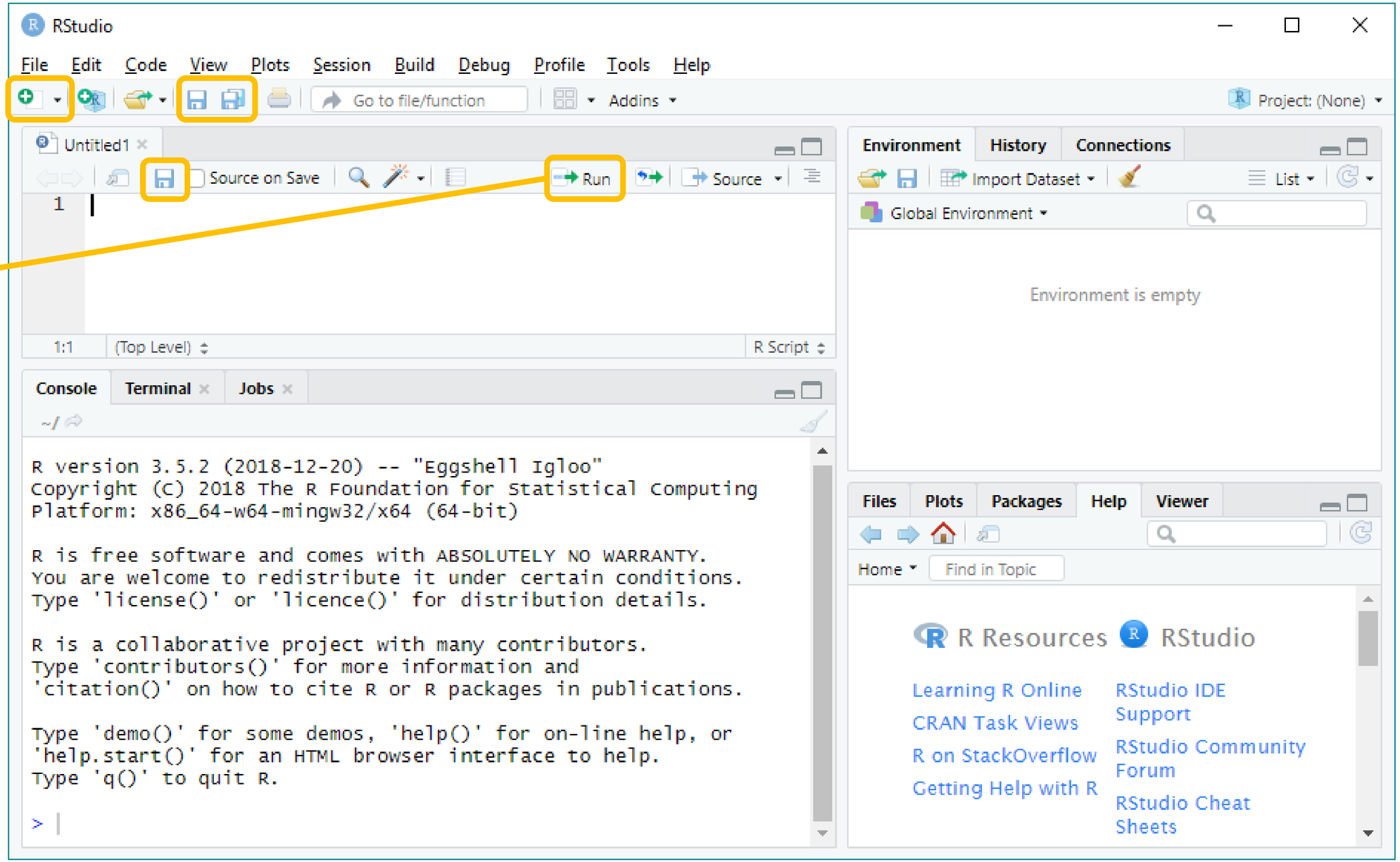
RStudio

These buttons are used to save the results of your work.



RStudio

It is used to run the selected part of the script or line where the cursor is located. The shortcut Ctrl+Enter does the same. The run section will be displayed in the console along with the result.



Silesian
University
of Technology

RESEARCH
UNIVERSITY
EXCELLENCE INITIATIVE
Ministry of Science
and Higher Education

Part 3

Basics of R

Scalar variables

```
A <- 5          numeric
B <- "gene"     character
C <- TRUE       logical
```

Arithmetic operators

Operator	Description
+	addition
-	subtraction
*	multiplication
/	division
^ or **	exponentiation
x %% y	modulus (x mod y) (5%%2 is 1)
x %/% y	integer division (5%/%2 is 2)

Logical operators

Operator	Description
<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to
==	exactly equal to
!=	not equal to
!x	not x
x & y	x AND y
x | y	x OR y

RESULT

Logical values:
TRUE
FALSE

Vectors

```
a <- c(1, 2, 3, -4, 4.5)
b <- c("gene", "protein", "transcript")
c <- c(TRUE, FALSE, TRUE, TRUE)
```

Let’s move on to coding...

Factors

```
gender <- c(rep("male",20), rep("female", 30))
gender <- factor(gender)

gender
[1] male   male   male   male   male   male   male   male   male   male   male
male   male   male   male   male   male   male
male   male   female female female female ...
Levels: female male

summary(gender)

female   male
     30     20

levels(gender) <- c("F", "M")

gender
[1] M M M M M M M M M M M M M M M M M M M M F F F F F F F F F F F F F F F F F F
F F F F F F F F F F F F
```

Matrices

```
v <- c(1, 2, 1, 2, 1, 2)
m <- matrix(v, nrow=2, ncol=3, byrow=FALSE)
     [,1] [,2] [,3]
[1,]1     1     1
[2,]2     2     2
m <- matrix(v, nrow=2, ncol=3, byrow=TRUE)
     [,1] [,2] [,3]
[1,]1     2     1
[2,]2     1     2
```

Lists

```
a <- 1:10
y <- matrix(1:6, nrow=2, ncol=3, byrow=FALSE)
l <- list(name="Anne", mynumbers=a, mymatrix=y, age=15.3)

$name
[1] "Anne"
$mynumbers
 [1]  1  2  3  4  5  6  7  8  9 10
$mymatrix
     [,1] [,2] [,3]
[1,] 1    3    5
[2,] 2    4    6
$age
[1] 15.3
```

Data frames

```
id <- c(1, 2, 3, 4)
name <- c("John", "Anne", "Tom", "Kate")
status <- c(TRUE, TRUE, FALSE, FALSE)
d <- data.frame(id, name, status)
names(d) <- c("ID", "Name","Status")
```

```
  ID  Name Status
1  1  John   TRUE
2  2  Anne   TRUE
3  3   Tom  FALSE
4  4  Kate  FALSE
```

Data structures

Data type	create from scratch	Function to: create from other data types	check the data type
Scalar	-	-	-

Vector	`c()`	`c()`
		`as.character()`
		`as.numeric()`
			`is.character()`
			`is.numeric()`

Factor	`factor()`	`factor()` `as.factor()`	`is.factor()`
Matrix	`matrix()`	`as.matrix()`	`is.matrix()`
List	`list()`	`as.list()`	`is.list()`
Data frame	`data.frame()`	`as.data.frame()`	`is.data.frame()`

Let’s move on to coding...

Useful functions

- `class()` – checks the class of an object (e.g. data frame, factor)
- `str()` – shows structure of an object
- `summary()` – prints summary of the object
- `length()` – gives number of elements in a vector/factor/list
- `dim(), nrow(), ncol()` – give dimensions of data frame/matrix
- `table()` - gives the counts at each combination in matrix/data frame/vector/factor
- `print()` – prints the object in the console
- `head()/tail()` – print first/last six elements or rows

Useful math functions

- `sum()` – returns the sum of values in a vector/matrix
- `rowSums()/colSums()` – gives the sum of values in each row/column
- `mean()` – returns the mean of values in a vector/matrix
- `median()` – returns the median of values in a vector/matrix
- `sd()` – returns the standard deviation of values in a vector/matrix
- `sqrt()` – gives a square root of the number

Let’s move on to coding...

Functions

```
myfunction <- function(arg1, arg2=NULL, ... ){

what_my_function_does

return(object) #kinda optional

}

myfunction(x,5)
```

Packages

Installing packages:

1) `install.packages("packageName")`
2) `install_github("author/package")` `#devtools package`
3) Install from source (.zip/tar.gz file)

To use an installed package:

```
library(package_name)
```

Packages

If you get an error looking like:

```
Error in arrange() : could not find function “function_name“,
```

the package is not loaded yet.

Try: `library(package_name)`

If the package fails to load, it means it is not installed yet.

Packages

To use a function from an installed and loaded package:

```
function_name(arg1, arg2, …)
```

To use a dataset from an installed and loaded package:

```
data("dataset_name")
attach(dataset_name)
```

Comments

- Everything after the sign # is ignored by the interpreter – it is not executed.
- It can be a description written by the author or a piece of code that should be ignored for now.
- You should write comments for chunks of source code (for another programmer and yourself in some time).


?functionName

Let’s move on to coding...

Control structures

```
if (i == 1){
}

for (i in 1:N) {
}

while (i<N){
}
```

Control structures

```
if (<something we want to test>){
    <some instructions>
} else {
    <some instructions>
}
```

Logical expression that gives TRUE or FALSE

Piece of code executed only when test result gives TRUE

Piece of code executed only when test result gives FALSE

Control structures

```
if (i > 0){
}
```

Can be applied to an object in a single line of code with the function **`ifelse()`**:

```
x <- c(-5, 8, -3)
ifelse(x > 0, TRUE, FALSE)
```

Result:

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

Let’s move on to coding...

Silesian
University
of Technology

RESEARCH
UNIVERSITY
EXCELLENCE INITIATIVE
Ministry of Science
and Higher Education

I APPRECIATE YOUR ATTENTION