

# 1- Introduction to the R language

Alex Sanchez, Miriam Mota and Santiago Perez-Hoyos

Statistics and Bioinformatics Unit. Vall d'Hebron Institut de Recerca

## Section 1

# Introduction to R

# Outline

- A first contact with R & Rstudio.
  - How does one work with R
- A primer of data import
  - Reading data into R
- A primer of communication report
  - R Notebooks and RMarkdown

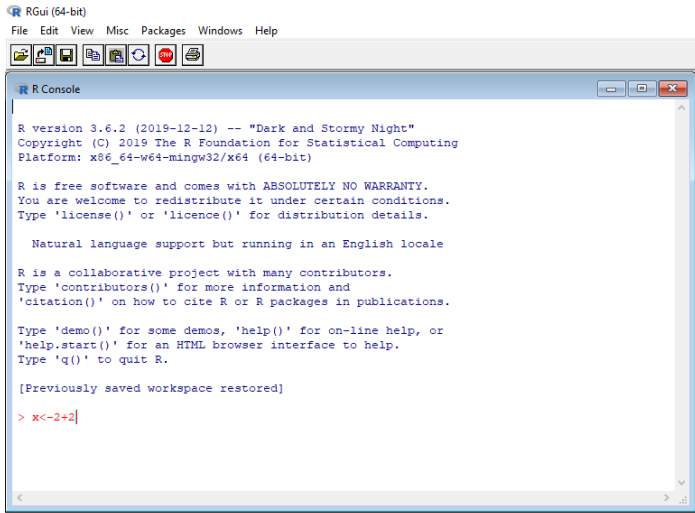
## Key Terms

- **R** is a *language and environment* for statistical computing and graphics.
- **R Studio** Graphical User Interface for easier use of R
- **Objects** Everything you store in R (datasets, variables, lists, results and graphs outputs) that can be referenced and reused
- **Functions** Pre-built lines of code that execute actions after inputting some parameters.
- **Packages or library** Shareable bundle of code and documentation that contains pre-defined functions. R contains base packages and for some analysis you must install and call specific ones.
- **Scripts** Document file that hold your commands that can be run later.
- **Rmarkdown** Special type of Script that can mixed text and comments with R commands that can be compiled in a final pdf

## How is R used

- Traditionally R was used from an Operating System console (“Terminal”)
- This is an intimidating approach for many users
- A variety of options exist to decrease the learning curve.
  - Use a supportive development environment such as **Rstudio**
  - Use an interface to Statistical tools with menus, such as **Rcommander** or **Jamovi** allowing to concentrate on Statistics, not in commands.

# A raw R console



```
RGui (64-bit)
File Edit View Misc Packages Windows Help

R Console

R version 3.6.2 (2019-12-12) -- "Dark and Stormy Night"
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

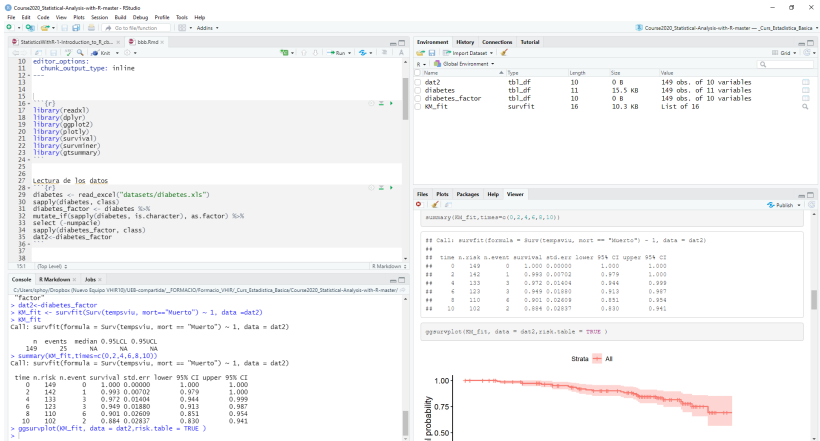
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

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.

[Previously saved workspace restored]

> x<-2+2|
```

# An “enhanced” console: Rstudio



# An “enhanced” console: Rstudio

The image shows the RStudio interface with several annotations:

- Source: Script Edition and Run:** Points to the top toolbar icons for running and saving the script.
- Environment objects:** Points to the Environment pane on the right, which lists objects like `tbl_df`, `diabetes`, `diabetes_factor`, and `KM_fit`.
- R Console:** Points to the bottom-left pane showing R code and its output.
- Plots, Packages and Help:** Points to the bottom-right pane displaying a Kaplan-Meier survival plot.

**R Console Output:**

```
library(readr)
library(dplyr)
library(ggplot2)
library(plotly)
library(survival)
library(survminer)
library(ggsurvfit)

# ... (code for loading data and fitting models) ...

summary(KM_fit$time&lt;=(0,2,4,6,8,10))

## Call: survfit(formula = Surv(timeperiod, most == "Muerto") ~ 1, data = dat2)
##
## time n.risk n.event survival std.err lower 95% CI upper 95% CI
## 0 149 0 1.000 0.00000 1.000 1.000
## 2 142 1 0.993 0.00702 0.979 1.000
## 4 133 3 0.972 0.01404 0.944 0.999
## 6 123 3 0.949 0.02100 0.913 0.987
## 8 110 6 0.902 0.02404 0.851 0.954
## 10 102 2 0.884 0.02837 0.830 0.941

ggsurvplot(KM_fit, data = dat2, risk.table = TRUE)
```

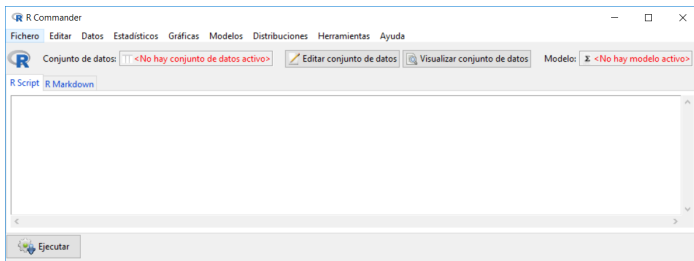
**Environment Objects Table:**

Name	Type	Size	Value
tbl_df	tbl_df	10	0 0
diabetes	tbl_df	11	149 obs. of 10 variables
diabetes_factor	tbl_df	10	149 obs. of 10 variables
KM_fit	survfit	16	List of 16

**Survival Plot:** A Kaplan-Meier survival plot showing the probability of survival over time. The y-axis is labeled "Surv" and ranges from 0.50 to 1.00. The x-axis is labeled "time" and ranges from 0 to 10. The plot shows a single survival curve with a shaded confidence interval.



## Something that is not a console: Rcommander



## Exercise

- Open R-Studio
- Identify Panes in R
- Calculate  $2+2$  in the console

Open new Script

- Run  $2+2$  Command line in Script

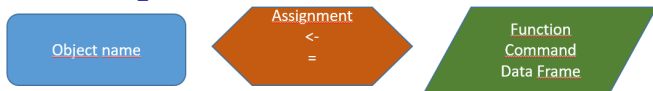
## Section 2

# Using R

# Setting

- One recommendation is to set a working directory
  - Session --> Set Working directory --> To source file location...
  - `getwd()` returns the path of the working directory
  - All files by default will be saved at the working directory
  - Suggestion: create a datasets folder in the course working directory and download data from webpage

# Commands, Objects and Functions



## Examples

- Vector

```
x1<-c(3,4,6,9,12)
```

```
x2<-c(3,4,6,9,20)
```

- Data Frames

```
dades<-data.frame(x1,x2)
```

- Results of execution of Functions

```
summary(x1,dat=dades)
```

## Exercise

- Create a Script with the commands of the previous slide and see the results

```
x1<-c(3,4,6,9,12)  # Create vector
x1  # Show vector
x2<-c(3,4,6,9,20)  # Create vector
x2  # Show vector
dades<-data.frame(x1,x2)  # Create database
dades  # Show database
summary(x1,dat=dades)  # Summary measures from database
```

## Exercise

```
x1<-c(3,4,6,9,12) # Create vector  
x1 # Show vector
```

```
## [1] 3 4 6 9 12
```

```
x2<-c(3,4,6,9,20) # Create vector  
x2 # Show vector
```

```
## [1] 3 4 6 9 20
```

```
dades<-data.frame(x1,x2) # Create database  
dades # Show database
```

```
##   x1 x2  
## 1  3  3  
## 2  4  4  
## 3  6  6  
## 4  9  9  
## 5 12 20
```

```
summary(x1,dats=dades) # Summary measures from database
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.  
##       3.0      4.0       6.0      6.8      9.0     12.0
```

## Section 3

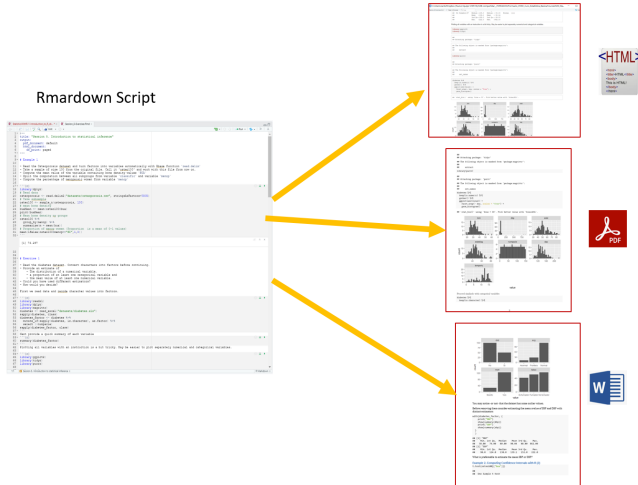
# Dynamic output with Rmarkdown



# Reproducible research with Rmarkdown

- R and Rstudio are strongly involved in promoting reproducibility and reproducible research.
- This is implemented in **Rmarkdown**
- A Rmarkdown combines
  - Natural language text, e.g. describing what we are doing in our own words.
  - R code with the instructions needed to do the data management or the analysis.
  - The output of the analysis

# Reports in Rmarkdown



# Creating Rmarkdown

- A Rmarkdown can be created in Rstudio with
  - File --> New File --> Rmarkdown
- The Rmarkdown contains example text and code so it is straightforward to adapt it to your analysis.
- To produce an html file with text, code and output:
  - Press the button “Knitr to Html”

# Rmarkdown Script

```
1 ---  
2 title: "Demo Rmarkdown"  
3 author: "UEB"  
4 date: "27/5/2021"  
5 output: html_document  
6 ---  
7  
8 ```{r setup, include=FALSE}  
9 knitr::opts_chunk$set(echo = TRUE)  
10 ```  
11  
12 ## R Markdown  
13  
14 This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF,  
15 and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.  
16  
17 When you click the Knit button a document will be generated that includes both content as  
18 well as the output of any embedded R code chunks within the document. You can embed an R code  
19 chunk like this:  
20  
21 ```{r cars}  
22 summary(cars)  
23 ```  
24  
25 ## Including Plots  
26  
27 You can also embed plots, for example:  
28  
29 ```{r pressure, echo=FALSE}  
30 plot(pressure)  
31 ```  
32  
33 Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R  
34 code that generated the plot.
```

YAML

R Chunk

Markdown

R Chunk

Markdown

R Chunk

Markdown

## Exercise

- Create a new Rmarkdown document
  - Include a title and your name
  - Compile document with 'knitr to html'

## Section 4

# R packages

## R packages

- R can be used for many different types of data processing and analysis from distinct fields, besides statistics such as Ecology, Omics Sciences, Psychology etc.
- All these capabilities are not present from the beginning because most of them will never be used by most users.
- Instead, they can be added when needed by
  - ❶ installing and
  - ❷ loading the appropriate packages.

## Installing and loading packages

We want to analyze some data using cox proportional hazards model.

```
res.cox <- coxph(Surv(time, status) ~ sex, data = lung)
```

```
Error in coxph(Surv(time, status) ~ sex, data = lung)  
: could not find function "coxph"
```

We need to install and load the package before we can use it.

```
install.packages("survival")  
library(survival)  
res.cox <- coxph(Surv(time, status) ~ sex, data = lung)
```



## Pacman Suggestion

- We suggest to use pacman package( short of package manager) to manage package and library
- pacman p\_load function install a package if it is not installed and activate it

*# Install ( if necessary) and load packages for use*

```
pacman::p_load(rio, tidyverse, here)
```

- WE can use package commands using two two colons like in previous chunk

packagename:: command(parameters)

# The tidyverse

- The tidyverse is an opinionated collection of R packages designed for data science.
- All packages share an underlying design philosophy, grammar, and data structures.
- The complete tidyverse collection can be installed with:

```
install.packages("tidyverse")
```

- <https://www.tidyverse.org/>

## Exercise

- Install the rio package from menu.
- Load the rio package.

## Section 5

# Getting data into R

## Importing data with Rstudio

- The easiest way to get data into R is to click on the Import Datasets button.
- Alternatively R code can be written using functions from Base R, the tidyverse or rio package
  - Base R functions start with `read.:` `read.table`, `read.csv`
  - tidyverse functions start with `read_:` `read_delim`, `read_csv` or `read_excel`
  - rio function is `import`

## Reading Excel or csv files

- Files can be read from any location, let it be a physical support or a web site.
- To read files from disk be sure to indicate their location.
- Alternatively the default working directory can be set to the folder where the file is located.
- Assume files `Diabetes.xls` and `Osteoporosis.csv` have been downloaded from url to a sub-folder named `datasets`
- Start setting the default directory to the folder where you have saved the `datasets` folder.
  - Session --> Set Working directory --> To source file location...

## Reading Excel or csv files (continued)

The code generated for reading the files can be reused any time changing the file name if needed.

```
# Read Excel file  
library(readxl)  
diabetes <- read_excel("../datasets/diabetes.xls")
```

## Reading text files

- Text files may require that more information is provided about delimiters, decimal sign, locale (language) or page encoding (UTFB for Mac or Linux vs ISO-8859-1 for Windows).
- All options can be selected from the rstudio importer

```
library(readr)
osteoporosis <- read_delim("../datasets/osteoporosis.csv",
  "\t", escape_double = FALSE, locale = locale(date_name = "full",
  decimal_mark = ",", encoding = "ISO-8859-1"))
```



## Reading Excel or csv files with rio

```
require(rio)
import("../datasets/diabetes.xls")
import("../datasets/osteoporosis.csv", dec = ",")
```

## Interlude: Summarizing data

- Once a dataset is available it is easy to “have a look at it”

```
head(diabetes)
str(diabetes)
dim(diabetes)
summary (diabetes)
```

## Section 6

# Resources and exercises

# Introductory materials

The web is full of all types of materials about R

Below there are a couple of brief introductions:

- A short introduction to R
- Getting started with R

## Exercise

- Select a dataset with which you wish to work along the course.
- Read it into R
  - How many variables are there in it
  - What are their types
- Try to summarize it briefly
- Create an Rmarkdown to encapsulate all your steps and share it with somebody.