Biostatistics I: Introduction to R

Basics

Eleni-Rosalina Andrinopoulou

Department of Biostatistics, Erasmus Medical Center

e.andrinopoulou@erasmusmc.nl

@erandrinopoulou

In this Section

- Using R
- Examples with Data
- Getting Familiar with R
- Importing data and saving your work
- A lot of practice

Using R

- R is a command-based procedural language
  - write and execute commands
  - use and define functions
- You may write the commands in the R console (Windows) or in a shell (Linux)

You will become more familiar with the syntax as you use it

Using R

- Strongly advisable to use a suitable text editor - Some available options:
  - RWinEdt (for Windows; you also need WinEdt installed)
  - Tinn-R (for Windows; http://sciviews.org/Tinn-R/)
  - Rkward (for Linux)
  - Emacs (w. ESS, all platforms)
  - Visual Studio (for Windows)
  - Rstudio (all major platforms; http://www.rstudio.org/)
  - for more check https://r-dir.com/blog/2013/01/list-of-r-editors.html

Using R

- For this course: Rstudio (http://www.rstudio.org/)
  - free
  - works fine in Windows, MacOS and Linux
  - helpful with errors
  - alternative output options

Using R

- Can I use R without Rstudio?
- Can I use Rstudio without R?

Practical Examples

- **Package `survival` - pbc data set**

id	time	status	trt	age	sex	bili	chol
1	400	2	1	58.76523	f	14.5	261
2	4500	0	1	56.44627	f	1.1	302
3	1012	2	1	70.07255	m	1.4	176
4	1925	2	1	54.74059	f	1.8	244
5	1504	1	2	38.10541	f	3.4	279

Practical Examples

- **id**: case number
- **time**: number of days between registration and the earlier of death, transplantation, or study analysis in July, 1986
- **status**: status at endpoint, 0/1/2 for censored, transplant, dead
- **trt**: 1/2/NA for D-penicillamine, placebo, not randomised
- **age**: in years
- **sex**: m/f
- **bili**: serum bilirunbin (mg/dl)
- **chol**: serum cholesterol (mg/dl)

More details:
https://stat.ethz.ch/R-manual/R-devel/library/survival/html/pbc.html

Practical Examples

- What is a **scalar**/vector/matrix

id	time	status	trt	age	sex	bili	chol	dt
1	400	2	1	58.76523	f	14.5	261	1
2	4500	0	1	56.44627	f	1.1	302	2
3	1012	2	1	**70.07255**	m	1.4	176	3
4	1925	2	1	54.74059	f	1.8	244	4
5	1504	1	2	38.10541	f	3.4	279	5

Practical Examples

- What is a scalar/**vector**/matrix

id	time	status	trt	age	sex	bili	chol
1	400	**2**	1	58.76523	f	14.5	261
2	4500	**0**	1	56.44627	f	1.1	302
3	1012	**2**	1	70.07255	m	1.4	176
4	1925	**2**	1	54.74059	f	1.8	244
5	1504	**1**	2	38.10541	f	3.4	279

Practical Examples

- What is a scalar/**vector**/matrix

id	time	status	trt	age	sex	bili	chol
1	**400**	2	1	58.76523	f	14.5	261
2	**4500**	0	1	56.44627	f	1.1	302
3	**1012**	2	1	70.07255	m	1.4	176
4	**1925**	2	1	54.74059	f	1.8	244
5	**1504**	1	2	38.10541	f	3.4	279

Practical Examples

- What is a scalar/vector/**matrix**

id	time	status	trt	age	sex	bili	chol
1	**400**	**2**	1	58.76523	f	14.5	261
2	**4500**	**0**	1	56.44627	f	1.1	302
3	**1012**	**2**	1	70.07255	m	1.4	176
4	**1925**	**2**	1	54.74059	f	1.8	244
5	**1504**	**1**	2	38.10541	f	3.4	279

Practical Examples

- What is a scalar/vector/**matrix**

id	time	status	trt	age	sex	bili	chol
1	400	2	1	58.76523	f	**14.5**	**261**
2	4500	0	1	56.44627	f	**1.1**	**302**
3	1012	2	1	70.07255	m	**1.4**	**176**
4	1925	2	1	54.74059	f	**1.8**	**244**
5	1504	1	2	38.10541	f	**3.4**	**279**

Practical Examples

- Common questions
  - What is the average age?
  - What is the average serum bilirubin?
  - What is the average serum cholesterol?
  - What is the percentage of females?
  - How many missing values do we have for serum cholesterol?

All these questions can be answered using R!

Getting Familiar with R

- Elementary commands: **expressions** and **assignments**
- An **expression** given as command is evaluated printed and discarded
- An **assignment** evaluates an expression and passes the value to a variable - the result is not automatically printed

Getting Familiar with R

Expression is given as a command,

```
103473
```

```
[1] 103473
```

- However, it cannot be viewed again unless the command is rerun.

Getting Familiar with R

Expression is given as a command,

```
103473
```

```
[1] 103473
```

- However, it cannot be viewed again unless the command is rerun.

In order to store information, the expression should assign the command

```
x <- 103473
x
```

```
[1] 103473
```

Getting Familiar with R

You can use R as a calculator!

- Basic arithmetics

```
+, -, *, /, ^
```

```
y <- 103473 + 100000
y
```

```
[1] 203473
```

- Complicated arithmetics

Getting Familiar with R

Tips:

- R is case sensitive, e.g.,
  - **"sex"** is different than **"Sex"**
- Commands are separated by a semi-colon or by a newline
- Comments can be put anywhere, starting with a hashmark **#**: everything to the end of the line is a comment
- Assign a value to an object by **<-** or **=**
- Working directory: get with **getwd()** and set with **setwd()**

Getting Familiar with R

- Missing values
  - are coded as **NA** (i.e., not available) **is.na()**
- Infinity
  - is coded as **Inf** (plus infinity) or **-Inf** (minus infinity) **is.finite()**
- The Null objects
  - are coded as **NULL** (undefined) **is.null()**
- Not a number
  - is coded as **NaN** (Not a Number). Example:

```
0/0
```

```
[1] NaN
```

Importing Data

- function: **read.table()**, **read.csv()** and its variants
  - note: use forward slashes or double backward slashes in the file names, e.g.,
    "`C:/Documents and Settings/User/Data/file.txt`" or
    "`C:\\Documents and Settings\\User\\Data\\file.txt`"

- Specialized functions for importing data from other programs
  - package: **foreign**, function: **read.spss()**, **read.dta()**
  - package: **Hmisc**, function: **sas.get()**
  - package: **openxlsx**, function: **read.xlsx()**
  - package: **readxl**, function: **read_excel()**
  - package: **haven**, function: **read_spss()**
  - *etc*

Exporting Data

- Specialized functions for exporting data to other programs
  - function: **write.table()**, **write.csv()**
  - package: **foreign**, function: **write.spss()**, **write.dta()**
  - package: **openxlsx**, function: **write.xlsx()**
  - *etc*

Saving and Loading your Work

Multiple objects:

- You can save your R objects using **save()**
  - be careful about overwriting
- You can load your saved R objects using **load()**

Single object:

- Using **saveRDS()** you can save a single R object
- Using **readRDS()** you can load a single R object
  - we will need an assignment statement to store the results

Save your code by using the tab File in Rstudio!

Saving and Loading your Work

Tips:

- Short names are preferred over longer names
- Try to avoid using names that contain symbols
- Avoid spaces in names
- Remove any comments in your data set
- Make sure that any missing values in your data set are indicated with the same value (or no value)

Summary

Basic functions

- `getwd()`, `setwd()`,
- `is.na()`, `is.finite()`, `is.null()`

Import/Export

- `read.csv()`, `write.csv()`
- `read.xlsx()`, `write.xlsx()`
- `read.table()`, `write.table()`

Save/Load

- `save()`, `saveRDS()`
- `load()`, `readRDS()`