R Laboratory
Introduction to R

Francesco Schirripa

francesco.schirripa@ec.unipi.it

October 5, 2018

What is R?

- "*R is a free software environment for statistical computing and graphics*"

- It is open source and available at `https://www.r-project.org/`.
- Open source means that you do not have to pay for it...but it is much more: it provides full access to algorithms and their implementation; it gives you the ability to fix bugs and extend software; it promotes reproducible research...
- It is *multiplatform* (Windows, Linux, MacOsX)

How does R work?

- Once R is started, a console is displayed where commands can be written (at the prompt >).
  However, it is a good practice to store all comments in a script file with extension .R.

  Two windows:
  - Script (new or existing)
  - Terminal – output and temporary input (unsaved)
- RStudio (`http://www.rstudio.com`): integrated development environment for R including:
  1. Source code (script file);
  2. Console (output);
  3. Workspace;
  4. Windows for graphics/packages/online help

Online help

The online help is a very useful tool to familiarise with R and its commands:

1. `help.start()` opens the html main page of R online help;
2. `help(cmd1)` (or `?cmd1`) provides details about how command cmd1 works;
3. `help.search("keyword")` (or `??keyword`) performs an online search based on keyword;
4. `apropos("keyword")` returns all the commands containing keyword in its name.

R objects

R is an **object-oriented program**: every operation is made on and produces objects. All objects in R have a class, reported by the function `class()`.
Possible class are: `numeric`, `logical`, `character`, `list`, `matrix`, `array`, `factor` and `data.frame`.

Assignment is performed by the <- or the = operator.
Example:

```
> x=5
> x
[1] 5
```

Rules for assignment

Names defining objects cannot contain spaces or mathematical operators/special characters (except for the dot .), nor can they begin with a number; Some peculiar values:

- `NA` (Not Available) is the code denoting a missing numerical or character element (warning: "NA" is a valid character string);
- `NaN` (Not a Number) is the result of impossible or undefined expressions like a division by zero;
- `Inf` and `-Inf` denote $\pm\infty$
- These and other R keywords (`for`,`while`,`if`,`TRUE`,`FALSE` etc.) are not available for assignment.

Some basic functions

- Once you open R you are inside the memory of the computer. The part of the memory in which you are currently working is called **working directory**.
  You can find out which is the current working directory by running the `getwd()` - get working directory - function.
- `dir()` displays all the files in the directory;
- `setwd("new path")`: change the current working directory. In RStudio you can set the working directory using the menu.
- `ls()`: shows the list of all the objects stored in the workspace;
- `rm(list=ls())`: cleans the whole workspace;
- `load("mydata.Rdata")`: imports objects stored in mydata.Rdata;
- `source("mycode.R")`: runs all the commands saved in mycode.R;
- `save.image("myfile.Rdata")`: saves all the objects in a workspace;
- `save(x,file="x.Rdata")`: saves the object x in a dedicated workspace.

Vectors (1)

- A vector is a sequence of data elements of the same basic type. To define a vector we use the function `c()`;
- `length()`: returns the length of a vector;
- `x=seq(from=a,to=b,by=s)`: returns a vector with elements from `a` to `b` with step `s`;
- `x=rep(x,times=a)`: repeats `x` a times (`x` can be a vector);
- Specific elements can be selected using square brackets: `x[a]`;
- Basic mathematical functions can also be applied to vectors. Such functions are performed element-by-element, i.e. *elementwise*; for example, suppose we have two vectors a and b:

```
> a <- c(1, 3, 5, 7)
> b <- c(1, 2, 4, 8)
```

Then, if we multiply a by 5, we would get a vector with each of its members multiplied by 5.

```
> 5 * a
[1]  5 15 25 35
```

Vectors (2)

- In other cases - like for basic statistical summary measures - the whole vector is the input of the function
  Example:

```
> s1 <- c (6, 1, 5, 9, 4, 7, 8, 2, 5, 8)
>median(s1)
[1] 5.5
>range(s1)
[1] 1 9
```

Logical values & operators

- Some operators return the logical values TRUE and FALSE:
  1. <: less than; <=: less or equal than;
  2. >: greater than; >=: greater or equal than;
  3. ==: equal to; !=: different from;
  4. `is.element()`: set membership indicator;
  5. `is.na()` indicates the elements of the vectors that represent missing data
- Logical operators:
  1. &: logical intersection (AND);
  2. |: logical union (OR);
  3. !: logical negation
- `which()` returns the indices of elements satisfying a logical condition.

Factors & lists

- **Factors**:
  1. are objects encoding categorical variables;
  2. are often used to group other (usually quantitative) variables;
  3. are defined by the command `factor()`;
  4. have a reference level which can be changed;
- **Lists**:
  1. are set of objects with different nature/dimension;
  2. are defined by the command `list()`;
  3. are useful to summarise an analysis or as output of complex functions.

Matrices & Arrays

- A matrix is a collection of data elements arranged in a two-dimensional rectangular layout:
  `matrix(data,nrow,ncol)` defines a `nrow`×`ncol` matrix;
- Arrays are the R data objects which can store data in more than two dimensions:
  `array(data,dim=c(n1,...,np))` defines a p-dimensional array;
- *elementwise* operations like for vectors can be performed;
- Most common **linear algebra operators** readily available:
  1. %*%: matrix product (conformable matrix dimensions needed);
  2. `det()`: matrix determinant;
  2. `t()`: matrix/vector transposition;
  2. `solve()`: solution of a linear system (can be used to invert matrices);
  2. `diag()`: extracts the diagonal of a matrix or builds a diagonal matrix.

Dataframes

- Data frame is a two dimensional data structure in R. It is a special case of a list which has each component of equal length.
- Data frame is of particular importance in data analysis. It represents the matrix of data where each row is an observation and each column is a variable (the variables may be of different type)
  Dataframes are defined by command:
  `data.frame(data,nrow,ncol)`;
- Variable `X` of dataframe `df` is obtained by `df$X`;
- `data()`: returns the list of all R datasets;
- `names(df)` or `colnames(df)`: returns the names of the columns of `df`;
- `attach(df)`: allows to refer to variables in `df` without using `df$X`;
- `detach(df)`: stops the effect of `attach(df)`.

Importing and exporting data

- The main commands to read external data are:
  1. `read.table()` for .txt files;
  2. `read.csv()` and `read.csv2()` for .csv files;
- Most common argument of these functions are (be careful because the default values of some of these argument in the two functions are different!)
  1. `file`: name with extension (and eventually pattern) of the data file;
  2. `header`: whether the first row contains the column names;
  3. `sep`: the column separator (space, comma, tabulation, semi-colon);
  4. `dec`: the decimal separator (dot or comma).
- Data can be exported with the commands: `write.table()`, `write.csv()`, `write.csv2()`