

Introductory course on the R software

Benoit Liquet¹

¹University of Pau & Pays de L'Adour
Laboratory of Mathematics and Its Application
France

University Sebalas Maret
Department of Mathematics
Faculty of Mathematics and Natural Science
February 28, 2017

<http://benoit-liquet.github.io>

Goals of today lecture

Describing the instructions

- to enter data directly in R ;
- to import or export data, to and from other software :
 - Excel ;
 - SPSS ;
 - Minitab ;
 - SAS ;
 - Matlab.

Importing data from a text file

The three main R functions to import data from a text file.

Function name	Description
<code>read.table()</code>	Best suited for data sets presented as tables, as it is often the case in Statistics.
<code>read.ftable()</code>	Reads contingency tables.
<code>scan()</code>	Much more flexible and powerful. Use this in all other cases.

Reading data with `read.table()`

To read data present in an ASCII file :

```
my.data <- read.table(file=file.choose(), header=TRUE,  
  sep="\t", dec=".", row.names=1)
```

Argument name	Description
<code>file=path/to/file</code>	Location and name of the file to be read.
<code>header=TRUE</code>	Indicates whether the variable names are given on the first line of the file.
<code>sep="\t"</code>	The values on each line are separated by this character (" <code>\t</code> "=TABULATION; " <code>"</code> "=whitespace; " <code>,</code> ", " <code>=</code> ", " <code>;</code> " etc.).
<code>dec="."</code>	Decimal mark for numbers (" <code>."</code> " or " <code>,</code> ").
<code>row.names=1</code>	1st column of the file gives the individuals' names.

Reading data with `read.table()`

The path can be specified explicitly :

```
my.data <- read.table(file="C:/MyFolder/somedata.txt")
```

or (doubling the backslashes) :

```
my.data <- read.table(file="C:\\MyFolder\\somedata.txt")
```

We can also use the function `setwd()` to change the work directory (equivalent to using the menu “File/Change current directory”) :

```
setwd("C:/MyFolder")  
my.file <- "mydata.txt"  
data <- read.table(file=my.file)  
head(data)
```

Do it yourself

You can perform the “**Do it yourself**” on page 66.

Reading data with read.ftable()

To read contingency tables like this one :

	"alcohol"	"nondrinker"	"occasional drinker"	"regular drinker"
"GENDER"	"tobacco"			
"M"	"non-smoker"	6	19	7
	"former smoker"	0	9	0
	"smoker"	1	6	5
"F"	"non-smoker"	12	26	2
	"former smoker"	3	5	1
	"smoker"	1	6	1

use the instructions :

```
Intima.table <- read.ftable("Intima_ftable.txt",
  row.var.names=c("GENDER", "tobacco"),
  col.vars=list("alcohol"=c("nondrinker",
    "occasional drinker", "regular drinker")))
ftable(Intima.table)
```

Reading data with the function scan()

Function `scan()` should be used when the data are not organized as a rectangular table.

For example, suppose your data file contains the following lines :

File description:

The individual data are registered for nine variables
in the following order:

GENDER AGE height weight tobacco packyear SPORT measure alcohol

Data:

```
1 33 170 70 1 1 0 0,52 1 2 33 177 67 2 20 0 0,42 1
2 53 164 63 1 30 0 0,65 0 2 42 169
76 1 26 1 0,48 1
```


Reading data with the function scan()

Here are the commands we suggest you use to read this file. The argument `skip=n` is used to omit reading the first n lines of the file.

```
# Reading variable names:
```

```
variable.name <- scan("Intima_Media2.txt", skip=4,  
                      nlines=1, what="")
```

```
# Reading data:
```

```
data <- scan("Intima_Media2.txt", skip=7, dec=",")  
mytable <- as.data.frame(matrix(data, ncol=9, byrow=TRUE))  
colnames(mytable) <- variable.name
```

Note : you can read data files directly from Internet with `read.table()` and `scan()`.

```
read.table("http://www.biostatisticien.eu/springer/  
           temperature.dat")
```

Importing data from Excel or the Open Office spreadsheet

First approach : using copy-paste

Using the mouse, select the range of the data (in the spreadsheet) which you wish to incorporate into R. Once the data are selected, copy them to the clipboard (from the Edit menu, or with the keyboard shortcuts CTRL+C on Windows or COMMAND+C on a Mac).

All you need to do now is type the following instructions in the R console to transfer the data from the clipboard.

```
x <- read.table(file("clipboard"), sep="\t",  
                header=TRUE, dec=", ")
```

Importing data from Excel or the Open Office spreadsheet

Second approach : Using an intermediary ASCII file

Save your file in an ASCII format, then refer to the previous section.

Third approach : Using specialized packages

- Use function `read.xls()` from package `gdata` (needs PERL, can be installed on Windows via the file `Rtools29.exe`) ;
- Use function `read.xlsx()` from packages `xlsx` or `openxlsx`.

Importing data from SPSS, Minitab, SAS or Matlab

TABLE: Packages and R importation functions from common software.

Software	Package	R function	File extension	Output format
SPSS	foreign	read.spss()	*.sav	list
Minitab	foreign	read.mtp()	*.mtp	list
SAS	foreign	read.xport()	*.xpt	data.frame
Matlab	R.matlab	readMat()	*.mat	list

Large data files

R can handle large data sets quickly and efficiently. For this, you need to specify explicitly the type of each column using the argument `colClasses` (e.g., `=rep("character",3)`) of the function `read.table()`.

```
tm <- Sys.time() # Gets the current time.  
dbsnp <- read.table("dbsnp123.dat")  
Sys.time() - tm  
Time difference of 5.063645 mins
```

```
tm <- Sys.time()  
dbsnp <- read.table("dbsnp123.dat",  
                    colClasses=rep("character",3))  
Sys.time() - tm  
Time difference of 13.75810 secs
```

Exporting data to an ASCII text file :

```
write.table(mydata, file = "myfile.txt", sep = "\t")
```

Exporting data to Excel or OpenOffice Calc :

```
X<-data.frame(Weight=c(80,90,75),Height=c(182,190,160))  
write.table(X,file("clipboard"),sep="\t",dec=".",  
            row.names=FALSE)
```

The data have now been copied to the clipboard. You can now paste them into your spreadsheet, for example by typing Ctrl+V.

See also the function `write.xlsx()` from packages `xlsx` or `openxlsx`.

Entering toy data : the `c()`, `seq()` and `:` functions

```
> c(1,5,8,2.3)
[1] 1.0 5.0 8.0 2.3
> seq(from=4,to=5)
[1] 4 5
> seq(from=4,to=5,by=0.1)
[1] 4.0 4.1 4.2 4.3 4.4 4.5 4.6 4.7 4.8 4.9 5.0
> seq(from=4,to=5,length=8)
[1] 4.000000 4.142857 4.285714 4.428571 4.571429 4.714286
[7] 4.857143 5.000000
> 1:12
[1] 1 2 3 4 5 6 7 8 9 10 11 12
```

Entering toy data : the rep() function

```
> rep(1,4)
[1] 1 1 1 1
> rep(1:4, 2)
[1] 1 2 3 4 1 2 3 4
> rep(1:4, each = 2)
[1] 1 1 2 2 3 3 4 4
> rep(1:4, c(2,1,2,3))
[1] 1 1 2 3 3 4 4 4
> rep(1:4, each = 2, len = 4)
[1] 1 1 2 2
> rep(1:4, each = 2, len = 10)
[1] 1 1 2 2 3 3 4 4 1 1
> rep(1:4, each = 2, times = 3)
[1] 1 1 2 2 3 3 4 4 1 1 2 2 3 3 4 4 1 1 2 2 3 3 4 4
```


Generating pseudo-random numbers

The function `runif()` generates a sequence of randomly generated numbers (at uniform).

```
> runif(5)
[1] 0.4344968 0.7153407 0.4561363 0.9580362 0.7260245
> runif(5,min=2,max=7)
[1] 5.634204 4.046403 5.415685 5.251441 2.209174
```

The function `rnorm()` generates a sequence of random numbers from a normal distribution.

```
> rnorm(4)
[1] 0.13585341 -0.09483162 -2.12326103 0.45974393
> rnorm(4,mean=2,sd=3)
[1] -0.8673785 3.5660222 0.9401026 3.4794672
```

Entering data from a hard copy

- **Creating a vector with the function `scan()`**

In this context, `scan()` is more user-friendly than `c()`. It can be used to easily enter data as you go.

```
> z <- scan() # R is waiting for you to enter data.
```

```
1: 4.2
```

```
2: 5.6
```

```
3: 8.9
```

```
4: 1
```

```
5: 2.3
```

```
6:          # Press ENTER after an empty line  
          # to halt the procedure.
```

```
Read 5 items
```

```
> z
```

```
[1] 4.2 5.6 8.9 1.0 2.3
```

Entering data from a hard copy

- **Creating several vectors of different lengths**

```
data.entry("")
```

You can change the names of the variables (columns) and enter data. Columns can contain different numbers of observations. If you leave the mini spreadsheet and type in the instruction `ls()`, you will see the variables you have created.

- **Creating and individual \times variables table**

To enter data directly into R's mini spreadsheet (as if using Excel), simply use the function `de()` (for *data entry*), as shown in the following instruction.

```
X <- as.data.frame(de(""))
```

Change the names of the variables and the types of the columns by clicking on the cells on the first row.

Your turn to work !

You can now try to do the **Worksheet** of Chapter 4 (A and C) on pages 81–83.

<http://biostatisticien.eu/springeR/Rbook-chap4.pdf>