Intro R read and write on disk

Roberto Molowny-Horas

April 24-25, 2023

Reading and writing ASCII .csv files

One of the most common ways of reading data from files is the use of the function **read.table** and its derivatives **read.csv**, **read.csv2** (see also **read.delim** and **read.delim2**). We'll see how they work.

```
out <- read.table(file="Example1.csv",header=T)
```

As we can see, it is important that we specify all characteristics of the .csv file that may affect the way R reads it. In this case, the default separation is a semicolon and we have to ask **read.table** to take that into account. Finally, keep in mind that files containing real numbers may use a decimal separator (i.e. comma or point) depending on the local settings of your computer. If that is not specified, numbers may be completely wrong.

```
out <- read.table(file="Example1.csv",header=T,sep=";",dec=".")
```

To write data to disk we can use **write.table**, **write.csv** or **write.csv2**.

```
x <- sample(10:20,100,replace=T)
l <- data.frame(Age=x,Height=135+(x-10)*3.5)
l[["Country"]] <- sample(c("Spain","Norway","Australia"),100,replace=T)
l[["Sex"]] <- sample(c("Male","Female","Unknown"),100,replace=T)
write.table(l,file="Example course R.csv",dec=".",sep=";",row.names=F)
```

Reading and writing .xls or .xlsx files

If, instead of an ASCII file, we have been given an Excel .xls or .xlsx file, we can read it by using the **Import Dataset** functionality imbedded in RStudio, which utilizes the **readxl** R package. For writing, however, **readxl** does not include any functionality, so we must turn to other packages like e.g. **openxlsx**.

```
# install.packages("openxlsx")
# library(openxlsx)
a <- openxlsx::read.xlsx("Tree-plot data.xlsx")
```

Writing is, in this case, far more complicated. We must create a new Excel workbook, add a new worksheet, write the data and save it.

```
wb <- openxlsx::createWorkbook("Myself")  # Use your own name or any other.
openxlsx::addWorksheet(wb, "Data selection")
b <- a[a$Plot != "A",]          # Only plots of type B and C.
openxlsx::writeData(wb, "Data selection", b, colNames = T, rowNames = F)
openxlsx::saveWorkbook(wb, file="My example openxlsx.xlsx", overwrite = T)
```

Functions **save** and **load**

An interesting way of saving whatever objects and stuff we have generated is **save**. With **save** you can choose what you want to save for latter, unlike **save.image()** (see below), and the result can be read back with **load**.

```
save(a, b, file = "An example of the save and load functions.Rdata", compress = T)
load(file = "An example of the save and load functions.Rdata")
```

Function **save.image()**

R keeps a log of all commands you have entered during your session and also has all variables available (unless you have erased them) in your working environment. It may be very useful to save all the work you have done so far, variables included, in a file for later, so that you can come back to the very same working environment. This can be achieved with **save.image()**, which saves everything onto disk exactly as it is. Notice that such file can use up significant disk space and take long time to write. For the former you may use **compress** and **compression_level** input parameters, whereas for the latter there is no solution but to wait until it finishes.

*Exercise. Create a 500x500 matrix and save it twice with **save** and with two different compression levels (see ?save for help). Compare their size on disk.