Read Excel files from R

*July 21, 2013*

By [Nicola Sturaro Sommacal](http://www.r-bloggers.com/author/nicola-sturaro-sommacal/)

(This article was first published on [**Milano R net**](http://www.milanor.net/blog/?p=779), and kindly contributed to [R-bloggers)](http://www.r-bloggers.com/)

Many people still save their data into Microsoft Excel files. This is an unhappy choice for many reasons but many was already written about this topic. Furthermore, unfortunately Excel become a *de facto* standard in many business environment and this routine seems to be difficult to strike out.

Many solutions have been implemented to read Excel files from R: each one has advantages and disadvantages, so an universal solution is not available. Get an overview of all the solutions, allows the choice of the best solution case-by-case.

Save Excel files into text

Saving Excel files into CSV can be done directly from Excel or through some external tools that allows batch operations. Native R functions for text data import can so be used.

**[?](http://www.ericbess.com/ericblog/2008/03/03/wp-codebox/" \l "examples" \o "WP-CodeBox HowTo?" \t "_blank)[View Code](javascript:;" \t "_blank) RSPLUS**

|  |
| --- |
| **df** = **read.table**("myfile.csv", header = TRUE) |

Copy and paste from Excel to R

This is a fast solutions, but it has one main drawbacks: it requires to open Excel file, select data and copy. By the way, this is the best compromise when you're in a hurry.

**[?](http://www.ericbess.com/ericblog/2008/03/03/wp-codebox/" \l "examples" \o "WP-CodeBox HowTo?" \t "_blank)[View Code](javascript:;" \t "_blank) RSPLUS**

|  |
| --- |
| **df** = **read.table**("clipboard") |

ODBC connection

For many years this was the easiest solutions based on R code for Windows users. Nowadays it still support only 32 bit versions of R and this limit discourage the use of this package. Besides Microsoft Windows and 32-bit R, it requires the Excel ODBC driver installed.

**[?](http://www.ericbess.com/ericblog/2008/03/03/wp-codebox/" \l "examples" \o "WP-CodeBox HowTo?" \t "_blank)[View Code](javascript:;" \t "_blank) RSPLUS**

|  |
| --- |
| **[require](http://astrostatistics.psu.edu/su07/R/html/graphics/html/require.html" \t "_blank)**(RODBC) conn = odbcConnectExcel("myfile.xlsx") # open a connection to the Excel file sqlTables(conn)$TABLE\_NAME # show all sheets **df** = sqlFetch(conn, "Sheet1") # read a sheet **df** = sqlQuery(conn, "select \* from [Sheet1 $]") # read a sheet (alternative SQL sintax) **[close](http://astrostatistics.psu.edu/su07/R/html/graphics/html/close.html" \t "_blank)**(conn) # close the connection to the file |

gdata package

In my experience the function provided by gdata package provides a good cross platform solutions. It is available for Windows, Mac or Linux. gdata requires you to install additional Perl libraries. Perl is usually already installed in Linux and Mac, but sometimes require more effort in Windows platforms.

**[?](http://www.ericbess.com/ericblog/2008/03/03/wp-codebox/" \l "examples" \o "WP-CodeBox HowTo?" \t "_blank)[View Code](javascript:;" \t "_blank) RSPLUS**

|  |
| --- |
| **[require](http://astrostatistics.psu.edu/su07/R/html/graphics/html/require.html" \t "_blank)**(gdata) **df** = read.xls ("myfile.xlsx"), sheet = 1, header = TRUE) |

xlsReadWrite package

xlsReadWrite is reported here for didactically purposes only although it is very fast: it doesn't support .xlsx files and this is not acceptable nowadays. Furthermore, it uses proprietary third party code and it should be downloaded from GitHub, CRAN cannot host it. It is available for Windows only.

**[?](http://www.ericbess.com/ericblog/2008/03/03/wp-codebox/" \l "examples" \o "WP-CodeBox HowTo?" \t "_blank)[View Code](javascript:;" \t "_blank) RSPLUS**

|  |
| --- |
| **[require](http://astrostatistics.psu.edu/su07/R/html/graphics/html/require.html" \t "_blank)**(xlsReadWrite) xls.getshlib() **df** = read.xls("myfile.xls", sheet = 1) |

XLConnect package

XLConnect is a Java-based solution, so it is cross platform and returns satisfactory results. For large data sets it may be very slow.

**[?](http://www.ericbess.com/ericblog/2008/03/03/wp-codebox/" \l "examples" \o "WP-CodeBox HowTo?" \t "_blank)[View Code](javascript:;" \t "_blank) RSPLUS**

|  |
| --- |
| **[require](http://astrostatistics.psu.edu/su07/R/html/graphics/html/require.html" \t "_blank)**(XLConnect) wb = loadWorkbook("myfile.xlsx") **df** = readWorksheet(wb, sheet = "Sheet1", header = TRUE) |

xlsx package

xlsx package read (and write) .xlsx and .xls files using Java. It is cross platform and uses rJava to deal with Java. Comments and examples below are taken from[a stackoverflow answer](http://stackoverflow.com/questions/6099243/read-an-excel-file-directly-from-a-r-script" \t "_blank). It probably returns the best results but requires some more options.

However, read.xlsx() function may be slow, when opening large Excel files. read.xlsx2() function is considerably faster, but does not quess the vector class of data.frame columns. You have to use colClasses()command to specify desired column classes, if you use read.xlsx2()function:

* read.xlsx("filename.xlsx", 1) reads your file and makes the data.frame column classes nearly useful, but is very slow for large data sets.
* read.xlsx2("filename.xlsx", 1) is faster, but you will have to define column classes manually. A shortcut is to run the command twice.character specification converts your columns to factors. Use Dateand POSIXct options for time.

**[?](http://www.ericbess.com/ericblog/2008/03/03/wp-codebox/" \l "examples" \o "WP-CodeBox HowTo?" \t "_blank)[View Code](javascript:;" \t "_blank) RSPLUS**

|  |
| --- |
| **[require](http://astrostatistics.psu.edu/su07/R/html/graphics/html/require.html" \t "_blank)**(xlsx) read.xlsx("myfile.xlsx", sheetName = "Sheet1") read.xlsx2("myfile.xlsx", sheetName = "Sheet1") |

**[?](http://www.ericbess.com/ericblog/2008/03/03/wp-codebox/" \l "examples" \o "WP-CodeBox HowTo?" \t "_blank)[View Code](javascript:;" \t "_blank) RSPLUS**

|  |
| --- |
| **[require](http://astrostatistics.psu.edu/su07/R/html/graphics/html/require.html" \t "_blank)**(xlsx) coln = **[function](http://astrostatistics.psu.edu/su07/R/html/graphics/html/function.html" \t "_blank)**(x) { # A function to see column numbers y = **[rbind](http://astrostatistics.psu.edu/su07/R/html/graphics/html/rbind.html" \t "_blank)**(**[seq](http://astrostatistics.psu.edu/su07/R/html/graphics/html/seq.html" \t "_blank)**(1, **[ncol](http://astrostatistics.psu.edu/su07/R/html/graphics/html/ncol.html" \t "_blank)**(x))) **[colnames](http://astrostatistics.psu.edu/su07/R/html/graphics/html/colnames.html" \t "_blank)**(y) = **[colnames](http://astrostatistics.psu.edu/su07/R/html/graphics/html/colnames.html" \t "_blank)**(x) **[rownames](http://astrostatistics.psu.edu/su07/R/html/graphics/html/rownames.html" \t "_blank)**(y) = "col.number" **[return](http://astrostatistics.psu.edu/su07/R/html/graphics/html/return.html" \t "_blank)**(y) } **data** = read.xlsx2("myfile.xlsx", 1) # open the file coln(**data**) # check the column numbers you want to have as factors x = 3 # Say you want columns 1-3 as factors, the rest numeric **data** = read.xlsx2("myfile.xlsx", 1, colClasses = **[c](http://astrostatistics.psu.edu/su07/R/html/graphics/html/c.html" \t "_blank)**(**[rep](http://astrostatistics.psu.edu/su07/R/html/graphics/html/rep.html" \t "_blank)**("character", x), **[rep](http://astrostatistics.psu.edu/su07/R/html/graphics/html/rep.html" \t "_blank)**("numeric", **[ncol](http://astrostatistics.psu.edu/su07/R/html/graphics/html/ncol.html" \t "_blank)**(**data**)-x+1)) ) |

A self made function

Finally, I found [on the web](http://housesofstones.com/blog/2013/06/20/quickly-read-excel-xlsx-worksheets-into-r-on-any-platform/" \t "_blank) a self made function to easily import xlsx files. It should work in all platforms and use XML. It doesn't work with old .xls files. It allows to read more sheets of a file in the same time, using the keep\_sheetsargument.

**[?](http://www.ericbess.com/ericblog/2008/03/03/wp-codebox/" \l "examples" \o "WP-CodeBox HowTo?" \t "_blank)[View Code](javascript:;" \t "_blank) RSPLUS**

|  |
| --- |
| **[source](http://astrostatistics.psu.edu/su07/R/html/graphics/html/source.html" \t "_blank)**("https://gist.github.com/schaunwheeler/5825002/raw/3526a15b032c06392740e20b6c9a179add2cee49/xlsxToR.r") xlsxToR = **[function](http://astrostatistics.psu.edu/su07/R/html/graphics/html/function.html" \t "_blank)**("myfile.xlsx", header = TRUE) |

I always suggest to read .csv files into R. By the way, for small dataset and testing use, my favourite solution is given by XLConnect.

And you, what do you use to import your own Excel files into R? Please leave a comment!