Data Import

Irfan

October 12, 2015

Data Import

In this document I will try to walk you through the most common data import functions. Once you learn the basics, you can use the same principles to any type of analysis task at hand. *Make sure you are using the same project as we did in the introduction.*

Through out this document you will see two types of code blocks. First group (exact representation may vary depending on media type) is blue text in light gray boxes. Those are R commands you can type in to the R console. Second group is the black text on white back ground, preceded by ##. Which is the R output of said command. Below is an example.

```
print("Hello MSBA Students")
```

```
## [1] "Hello MSBA Students"
```

Before Beginning

Make sure you are using the same project as we did in the introduction. Copy the files in the zip file provided to your project directory.

Comma Separated Value Files

This is the most common file format there is to transfer data, with good reason I might add. It is human readable (not binary), and is compatible with almost any system out there. The basic idea is, the columns are separated by commas and observations are separated by new lines. I would recommend you use this format unless you have compelling reason to do otherwise.

R uses read.csv function to import this file type. Let us learn more about this function.

```
help(read.csv)
```

As you can see, the function takes quite a few parameters but most are optional with reasonable defaults (meaning you can safely leave them blank).

You are provided with a file named “country.csv” (in the git repository)[https://github.com/iekanat/rworkshop]. If you placed this file in your project directory you can import its contents as follows.

```
# Save file contents into a variable called csv
csv <- read.csv("country.csv")
# view the contents (Output omited here to save space)
csv
```

To view the contents of this new variable in a spreadsheet format, double click the csv in Environment pane of R studio (Area B), or write the command below.

```r
View(csv) # we utilize View function to display results in R Studio
```

You can also explore the file contents via R functions.

```r
str(csv) # Structure
```

```
## 'data.frame':    29 obs. of  3 variables:
##  $ ip_iso2: Factor w/ 29 levels "AR","AU","AZ",..: 2 27 28 14 10 11 3 4 23 8 ...
##  $ Supply : int  20 80 100 50 70 40 20 10 50 30 ...
##  $ Those  : int  0 1 0 0 0 0 1 1 1 1 ...
```

```r
summary(csv) # Descriptive Statistics
```

```
##     ip_iso2       Supply           Those       
##  AR     : 1   Min.   :  0.00   Min.   :0.0000  
##  AU     : 1   1st Qu.: 25.00   1st Qu.:0.0000  
##  AZ     : 1   Median : 50.00   Median :0.0000  
##  BD     : 1   Mean   : 47.52   Mean   :0.4828  
##  BO     : 1   3rd Qu.: 64.00   3rd Qu.:1.0000  
##  BR     : 1   Max.   :100.00   Max.   :1.0000  
##  (Other):23
```

Excel Files

This file format is very common due to the popularity of Microsoft's office suite.

To import this file type we need two additional steps, installing and loading packages of which we will learn quite a bit more later.

```r
# Below is commented out, since I do not want R to install package every time you run this.
# Uncomment the line the first time and run.
#install.packages("xlsx") # Install the xlsx package that enables XLSX import function.
library(xlsx) # Enable the package
```

```
## Loading required package: rJava
## Loading required package: xlsxjars
```

Now let us read the xlsx file into R. Note that I had to specify which worksheet to import.

```r
xlsx <- read.xlsx("country.xlsx", sheetIndex = 1) # Import the file
xlsx # view contents
```

```
##    ip_iso2 Supply Those
## 1       AU     20     0
## 2       TR     80     1
## 3       US    100     0
## 4       GB     50     0
## 5       DE     70     0
## 6       DK     40     0
## 7       AZ     20     1
## 8        BD      10      1
## 9        MY      50      1
## 10       CN      30      1
## 11       MX      90      0
## 12       CA      60      0
## 13       IN      60      1
## 14       GT       0      0
## 15       IS      60      0
## 16       JM      25      1
## 17       JP      77      0
## 18       GA      32      1
## 19       GH       5      1
## 20       BO      64      1
## 21       BR      41      0
## 22       EG      72      1
## 23       SA      28      0
## 24       RU      53      1
## 25       LY      95      0
## 26       AR      21      1
## 27       SO      52      0
## 28       ZA      10      1
## 29       CZ      63      0
```

The package has other functionality that you can discover on your own.

Databases

R can connect to a variety of databases. For this workshop we currently do not have access to a database to demonstrate this functionality. Below is an example for MySQL.

```
library(RMySQL) # Load the necessary package
channel <- dbConnect(MySQL(), user = 'uname', password = 'pwd',
                     host = '127.0.01', dbname='exampledata') # Establish database connection
sql <- dbGetQuery(channel, "SELECT * FROM table;") # Execute query over connection, save results
```

Other Statistical Packages

R enables importing data from other statistical packages as well. I would recommend exporting the data from other packages in a csv and then importing it that way to minimize incompatibilities. If you have no access to the offending statistical package, then you can easily use one of the packages listed below.

I won't go into details as this may well turn out to be a fringe case not useful to all. For the purposes of this workshop, I will only discuss the packages and functions used and not actually import any data. You can use the principles used in earlier steps to import data on your own.

SAS

Package used for sas data format after version 7 is valled 'sas7bdat'. The function used is called read.sas7bdat().

If you have exported the data from SAS, you can use 'foreign' package with read.xport() function.

STATA

Stata data format has changed after version 13. If you wish to import a file of this sort use ‘readstata13’ package with read.dta13() function.

Earlier versions can be imported with ‘foreign’ package’s read.dta() function.

SPSS

SPSS sav files can be imported with ‘foreign’ package’s read.spss() function.

Exported por files can be imported with ‘Hmisc’ package’s spss.get() function.