Data Analysis in R: An Introduction

Anatoly Arlashin

Boston College

February 10, 2017

The latest version of this tutorial can be found here.
Outline

1 R's basics

2 Importing data
The two essential things are R itself and the coding environment for it called RStudio.

- www.r-project.org
- www.rstudio.com

Installation of both packages is pretty straightforward, the default options will be sufficient for all our needs.
Online resources

A small selection of tutorials:

- **An Introduction to R** — a manual from R Project, the creators of R
- **A (very) short introduction to R** — a 12 page summary of the most basic features of R and RStudio
- **Cookbook for R** — a collection of detailed and colorful tutorials
- **R bloggers** — a blog platform with lots of small tutorials and Q&As
- **Quick-R** — yet another collection

Online courses:

- **Rstudio Webinars** — free video tutorials on R and RStudio
- **Online learning** — a collection of links from RStudio’s authors
UI and coding basics

Tutorials on user interface of Rstudio, and some basic stuff about coding:

- Introduction to R and RStudio — #1
- Introduction to R and RStudio — #2
- Introduction to R and RStudio — #3
- Coding Style guide — useful advice on how to write clean and readable code
Importing data

The way you import data into R depends on two things:

- the existing format and structure of the data
- the desired format and structure of the data

As with almost any other aspect of R, there exist a variety of packages designed to import specific data types. Some of those packages are minimalistic, while others offer a wider selection of options.

The differences in various packages are mostly about

- data formats that are supported (*.xls, *.dta, *.csv, etc.)
- type of R object that will contain the imported data (matrix, list, dataframe, etc.)
- speed of import (only matters for big data)
Working directory

It is useful to tell R explicitly what folder/directory on your computer you would like to use as the current location, so that you will know where to put imported data files or where the output will be saved to by default.

```r
# displays the current working directory
getwd()

# sets the working directory to "C:/MSAE/"
setwd("C:/MSAE/")
```

Note that the path you put in `setwd("")` command must have forward slashes.
Excel data (*.xls, *.xlsx)

Many ways to import Excel data, we’ll look at the `gdata` package and the `read.xls()` and `read.xlsx()` functions it provides:

```r
# load package
library(gdata)
# load help documentation
help(read.xls)
# read the contents of the file "mydata.xls"
# into a new R object called "mydata.xls"
mydata.xls <- read.xls("mydata.xls")
```

Both functions return a data frame. The default option is to read the contents of the first sheet in Excel workbook.
Stata data (*.dat)

We'll use `readstata13` package, it supports importing data from Stata versions 14 and older.

```r
# load package
library(readstata13)

help(read.dta13)
# read the contents of the file "mydata.dta"
# into a new R object called "mydata.dta"
mydata.dta <- read.dta13("mydata.dta") # read file
```

The function returns a data frame.
Tabular separated data (*.csv, *.txt)

This is the type of data stored in a raw text format with variables separated by either symbols (comma, semicolon, etc) or spaces.

We'll use the build-in functions `read.csv()` and `read.table()`:

```r
# load help documentation
help(read.table)
# read the contents of the file "mydata.csv"
# into a new R object called "mydata.csv"
mydata.csv <- read.csv("mydata.csv")
```

The function returns a data frame. While technically `read.csv()` is a special case of `read.table()`, the difference between them is that `read.csv()` is able to correctly read various csv files without specifying extra options, while `read.table()` may fail to do so.