NBS8186: Computer Lab 1

Edu Gonzalo Almorox

14/11/2016

Introduction

Goals for session 1.

- Load data in R
- Manipulate data
- Fit and interpret econometric models

R in a nutshell

What is R? *programming language*, *environment*, *software*. . .

Pros:

- Object programming
- Open source and free
- Compatibility with other languages i.e., Phyton, Javascript

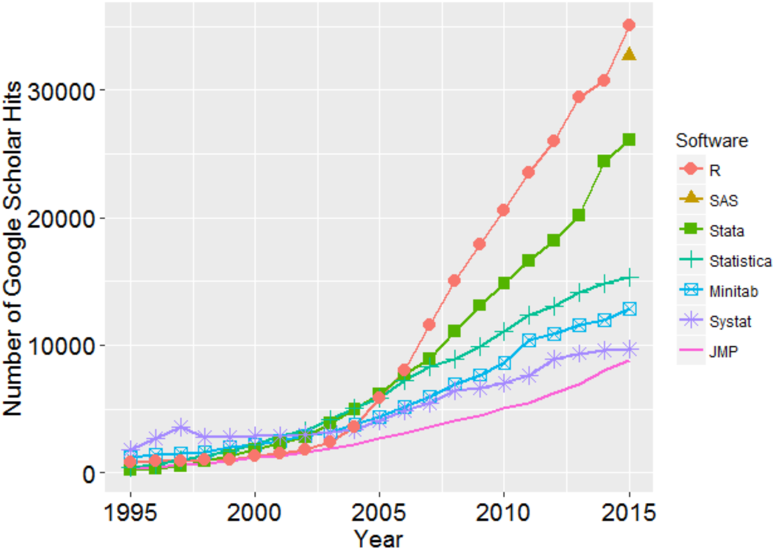
Cons:

- Important learning curve
- Documentation sometimes far from perfect.

What can you do with R?

- Data analysis
- Data visualisation
- Dynamic documents
- . . .

Is R a good investment?



Source

www.r4stats.com

Data structures

In R every element is regarded as an object. Objects are data structures that group data according to specific attributes. Most general data structures are organised by two elements

- Dimensionality
- Type of the contents (homogeneous, heterogeneous)
    1. **`numeric or character`**: single number of letter
    2. **`Vector`**: 1 dimension, homogeneous objects.
    3. **`List`**: 1 dimension, heterogenous objects - (different objects grouped together)
    4. **`Matrix`**: more than 1 dimension, homogenous objects
    5. **`Data frame`**: more than 1 dimension, heterogenous objects.

Data structures: examples

This is a `vector`

```
## [1] 1 2 3 4
```

This is a `list`

```
## [[1]]
## [1] 1 2 3 4 5
##
## [[2]]
## [1] "a" "b" "c" "d" "e"
```

This is a `data.frame`

```
##   numbers letters
## 1       1       a
## 2       2       b
```

Data frames

`data.frames` are the most common data structure for gathering information.

- **Variables**: Collect different arguments associated with the information to be analysed - diffrent formats (numbers, strings, factors, dates, . . .)
- **Observations**: Units of analysis (individuals, firms, etc. . .) - e.g. the rows of your dataset.

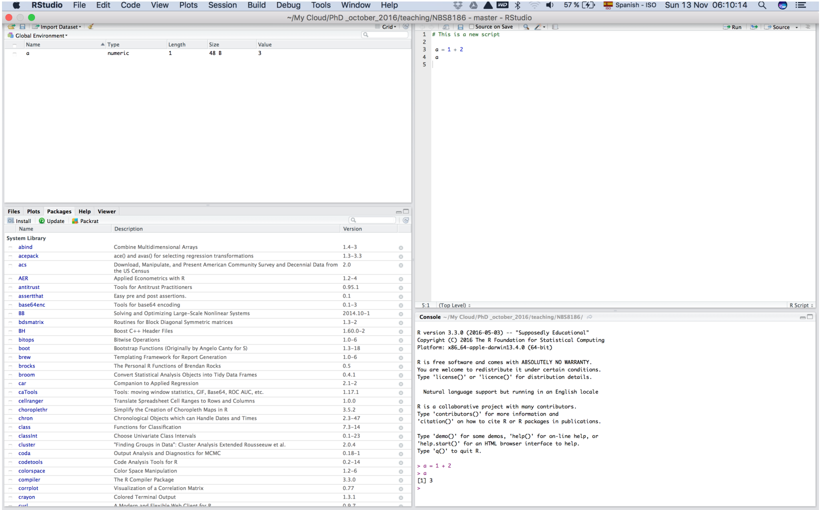
```
##   marr  wage exper age coll games minutes
## 1    1 1.002     4  27    4    77    2867
## 2    1 2.030     5  28    4    78    2789
## 3    0 0.650     1  25    4    74    1149
## 4    0 2.030     5  28    4    47    1178
## 5    0 0.755     3  24    4    82    2096
```

Before you start

In the (likely) case of crisis

- Specialised websites - e.g. stackoverflow.com
- R Mailing lists
- `help`, `help.search()`, `??(name package/name function)`

R Studio



```
# This is a new script

a = 1 + 2
a
```

Name	Type	Length	Size	Value
a	numeric	1	48 B	3

Name	Description	Version
System Library		
abind	Combine Multidimensional Arrays	1.4-3
acepack	ace() and avas() for selecting regression transformations	1.3-3.3
acs	Download, Manipulate, and Present American Community Survey and Decennial Data from the US Census	2.0
AER	Applied Econometrics with R	1.2-4
antitrust	Tools for Antitrust Practitioners	0.95.1
assertthat	Easy pre and post assertions.	0.1
base64enc	Tools for base64 encoding	0.1-3
BB	Solving and Optimizing Large-Scale Nonlinear Systems	2014.10-1
bdsmatrix	Routines for Block Diagonal Symmetric matrices	1.3-2
BH	Boost C++ Header Files	1.60.0-2
bitops	Bitwise Operations	1.0-6
boot	Bootstrap Functions (Originally by Angelo Canty for S)	1.3-18
brew	Templating Framework for Report Generation	1.0-6
brocks	The Personal R Functions of Brendan Rocks	0.5
broom	Convert Statistical Analysis Objects into Tidy Data Frames	0.4.1
car	Companion to Applied Regression	2.1-2
caTools	Tools: moving window statistics, GIF, Base64, ROC AUC, etc.	1.17.1
cellranger	Translate Spreadsheet Cell Ranges to Rows and Columns	1.0.0
choroplethr	Simplify the Creation of Choropleth Maps in R	3.5.2
chron	Chronological Objects which can Handle Dates and Times	2.3-47
class	Functions for Classification	7.3-14
classInt	Choose Univariate Class Intervals	0.1-23
cluster	"Finding Groups in Data": Cluster Analysis Extended Rousseeuw et al.	2.0.4
coda	Output Analysis and Diagnostics for MCMC	0.18-1
codetools	Code Analysis Tools for R	0.2-14
colorspace	Color Space Manipulation	1.2-6
compiler	The R Compiler Package	3.3.0
corrplot	Visualization of a Correlation Matrix	0.77
crayon	Colored Terminal Output	1.3.1
curl	A Modern and Flexible Web Client for R	0.9.7

```
R version 3.3.0 (2016-05-03) -- "Supposedly Educational"
Copyright (C) 2016 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin13.4.0 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> a = 1 + 2
> a
[1] 3
>
```

Figure 1: RStudio screen

Programming your analysis

Why writing code?

- Helps to keep track of what you are doing.
- Reduces the sources of error.
- Increases your productivity and efficiency - similar code for differnet analyses.
- Enhances collaborations.

R language

- Packages contain libraries that perform functions.
- Functions are composed by arguments.

```r
df = data.frame(numbers = c(1,2), letters = c("a", "b") )
```

Task 1: Load the data in R

There are two possible ways to input information:

- *Manually*
- *Import* from somewhere

The majority of the analyses import data:

- Data are delivered in different formats.
- Important to understand how the information is structured.

```r
# working directory
setwd("your_PC/comp_lab1")

install.packages("") # for installing packages
library("") # for loading libraries
```

Task 1: Load the data in R'cont

QA: *Download the data set from Blackboard and save it on your h: drive. Then open the data set in R and make it the default data set.*

```
# working directory
setwd("")

install.packages("") # for installing packages
library("") # for loading libraries

# loading data

read.csv()
import() #'rio() package'
```

Task 2: Preliminar exploratory analyses

```r
nba = read.csv("nba.csv", sep = ",", header = TRUE)
```

- How is the structure of your data?

```r
head(df) # gives the first lines

tail(df) # gives the last lines

str(df) # types of variables
```

Task 2: Summary

QB: *Have a look at the summary statistics of the data set. What is the average age of the players?*

- `summary()` is used to get a summary statistics of the variables in your data frame.

```r
summary(nba) # also referred to variables
```

- An alternative way to obtain the average age would be by calling directly the variable age using the operator $.

```r
mean(nba$age)
```

```
## [1] 27.38951
```

Task 2'cont: Counts of categories

QB cont': *How many play forwards?*

- What class of data is `forward`?
- `table()` summarises the number of categories in a factor.[1].

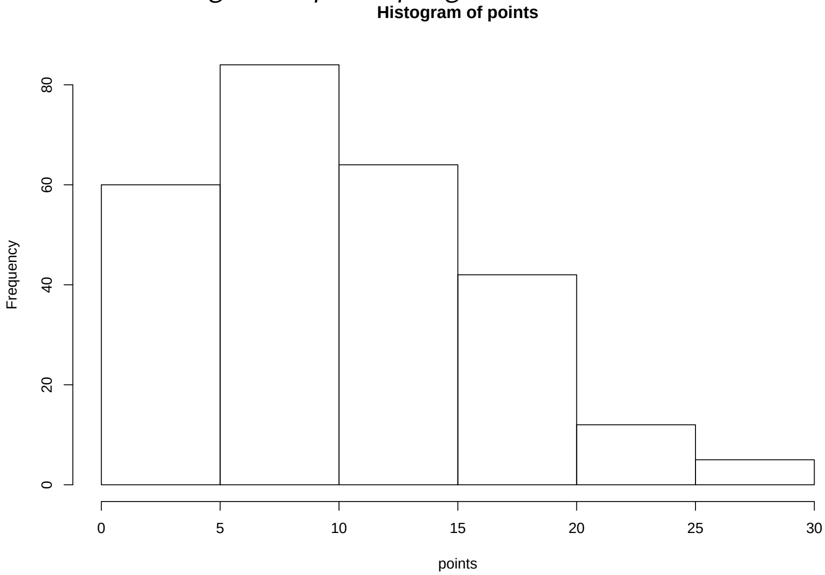
```r
table(nba$forward)
```

```
##
##   0   1
## 158 109
```

[1] There are alternative and more efficient ways to carry out this task. `data.table` and `dplyr` are the most suited packages when there are bigger samples.

Task 2'cont: Histograms

QC: *Plot a histogram of points-per-game.*



Task 2'cont: Histograms

- hist() is the simplest way for plotting a histogram.[2]

```
# Histogram

    # xlab = rename the axis X
    # main = title of the plot

hist(nba$points,
     xlab= "points",
     main = "Histogram of points")
```

[2] Package ggplot2 offers a wide range of histograms and other plotting alternatives.

Task 2'cont: Scatterplots

QD:*Produce a scatterplot of points-per-game versus years in league.*

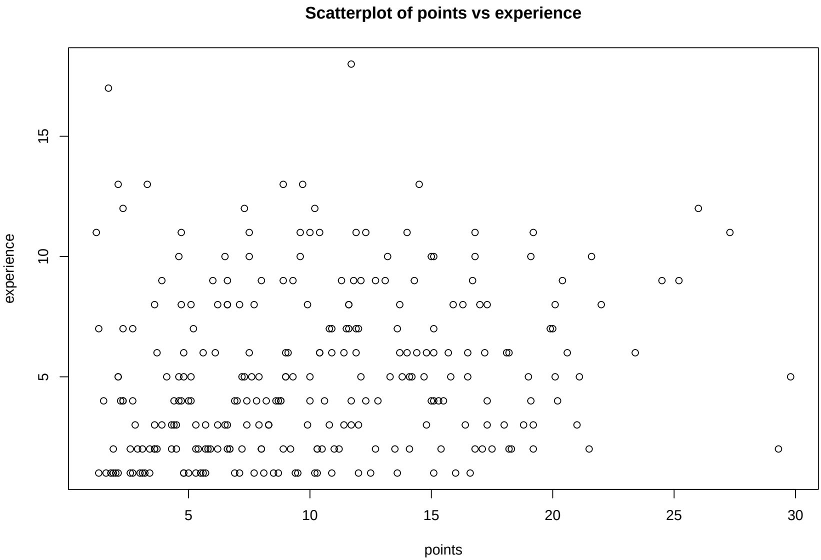
- Scatterplots represent the association between two variables.
- A way of doing it is by using function `plot()` and `with()` to *attach* the data frame and use the variables independently

```r
# Scatterplot

    # xlab = rename the axis X
    # ylab = rename the axis Y
    # main = title of the plot

with(nba, plot(points, exper,
               xlab= "points",
               ylab = "experience",
               main = "Scatterplot of points vs
               experience"))
```

Task 2'cont: Scatterplots



Task 3: Regression models

QE: *Run a regression of points-per-game on years in league, age, years played in college and position dummies.*

- We need libraries `stats` and `AER`
- `lm` estimates a linear model using ordinary least squares (OLS).
- The variable before "~" indicates the dependent variable whereas the variables in the right side are considered the set of explanatory regressors.
- `model1` is a fitted-model object.

```
library(stats)
library(AER)

 model1 = lm(points ~ exper + age + coll +
               forward + center,
               data = nba)

 summary(model1)
```

Task 3: Regression models

What can we say of our fitted model?

- Experience has a statistically significant influence in the perfomance - an additional year of experience implies 1.4 additional points per game.
- Age and years playing at college (`coll`) play a negative role (Question F)
- All the coefficients are jointly signifcant.

Task 3: Correlation matrix

QG: *Look at the correlation matrix*

```r
library(Hmisc)
library(dplyr)

# select variables from the model
vars_mod = nba %>% select(exper, age, coll,
                          forward, center)
    # note: subsetting using pipes

# correlation matrix
cor_mat <- rcorr(as.matrix(vars_mod), type = "pearson")
emphasize.strong.cells(which(cor_mat[[3]] < 0.001,
                             arr.ind = TRUE))
```

Task 3: Correlation matrix

QG'cont: *Do you need to worry about multicollinearity?*

- How is the Pearson correlation coefficient?
- Is this correlation significant?

Task 3: Generate new variables

QH:*Generate a new variable which is experience squared and include it in the regression.*

- Simplest solution[3] - e.g. indexing

```
nba$expersq =nba$exper^2
```

QH'cont: *Holding age, coll, center and forward fixed, at what value of experience does the next year of experience reduce points-per-game?*

```
# include 'expersq'
model2 = lm(points ~ exper+expersq+age+coll+
                   center+forward,
                  data = nba)
```

[3] This solution includes a base package. Yet, `dplyr` presents more flexible options for creating various variables under a number of conditions

Task 3: Transform variables

- Sometimes we need to transform variables.
- Log transformation is normally used.
- Interpretation of coeficients may change.

QI:*Now you want to explain the log(wage)*

```r
nba$logwage = with(nba, log(wage))
```

- `model3` is expressed as follows

```r
# include 'logwage'
model3 <- lm(logwage~points+exper+expersq+age+coll,
             data = nba)
```

Task 3: Transform variables cont'

How do you interpret the results?
A log transformation in the depedent variable in this case will interpreted as a percent change.

- Points obtained would suppose an increase of 7% in the wage.
- An additional year of experience would suppose an increase of the 22.3%.

Task 3: Model comparison - ANOVA test

QJ: *Test whether age and coll are jointly significant in the regression from (i). What does this imply about whether age and education have a separate effect on wage, once productivity and senority are controlled for?*

```r
mod_unrest <- lm(logwage~points+exper+expersq+
                   age+coll, data = nba)
mod_rest <- lm(logwage~points+exper+expersq, data = nba)

anova(mod_rest, mod_unrest)
```

Recap

- Writing code helps to control the workflow.
- Loading data depends notably on how what type of format you have - normally is `.csv`.
- There are different ways to access data in R. Common ways are through $and functions such as `which`.
- It is important to understand how to define the relationship between the dependent and independent variables. Also, variables may have transformations and it can have implications in terms of interpretation.