ETC3250 2017 - Lab 1

Souhaib Ben Taieb

26 July 2017

The aim of this first lab session is to make sure everyone is familiar with:

- R and RStudio
- RStudio Projects
- RMarkdown
- R syntax and Basic functions

What is R?

From Wikipedia: "R is a programming language and software environment for statistical computing and graphics supported by the R Foundation for Statistical Computing. The R language is widely used among statisticians and data miners for developing statistical software and data analysis."

R is free to use and has more than 8500 user contributed add-on packages on the Comprehensive R Archive Network (CRAN).

What is RStudio?

From Julie Lowndes:

If R were an airplane, RStudio would be the airport, providing many, many supporting services that make it easier for you, the pilot, to take off and go to awesome places. Sure, you can fly an airplane without an airport, but having those runways and supporting infrastructure is a game-changer.

The RStudio integrated development environment (IDE) has multiple components including:

1. Source editor:

- Docking station for multiple files,
- Useful shortcuts ("Knit"),
- Highlighting/Tab-completion,
- Code-checking (R, HTML, JS),
- Debugging features

2. Console window:

- Highlighting/Tab-completion,
- Search recent commands

3. Other tabs/panes:

- Graphics,
- R documentation,
- Environment pane,
- File system navigation/access,
- Tools for package development, git, etc

What is RMarkdown?

- R Markdown is an authoring format that enables easy creation of dynamic documents, presentations, and reports from R.
- It combines the core syntax of **markdown** (an easy-to-write plain text format) **with embedded R code chunks** that are run so their output can be included in the final document.
- R Markdown documents are fully reproducible (they can be automatically regenerated whenever underlying R code or data changes).

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document.

Equations can be included using LaTeX (https://latex-project.org/) commands like this:

```
$$s^2 = \frac{1}{n-1}\sum_{i=1}^n (x_i-\bar{x})^2.$$
```

produce

$$s^2 = \frac{1}{n-1}\sum_{i=1}^n (x_i-\bar{x})^2.$$

We can also use inline mathematical symbols such as `$\alpha$` and `$\infty$`, which produce α and ∞, respectively.

For more details on using R Markdown see http://rmarkdown.rstudio.com. Spend a few minutes looking over that website before continuing with this document.

RStudio Projects

- Project directories keep your work organized since you will keep your data, your code, your results all located in one place.
- For the unit ETC2420, I have created a project on my laptop called `ETC2420`. Note that the name of the current project can be seen at the top right of the RStudio window.

- Each time you start RStudio for this class, be sure to open the right project.

Exercise 1

Create a project for this unit, in the directory.

- File -> New Project -> Existing Directory -> Empty Project

Exercise 2

Open a new Rmarkdown document. You are going to want to call it `Lab1` (it will automatically get the file extension `.Rmd` when you save it).

- File -> New File -> R Markdown -> OK -> Knit HTML

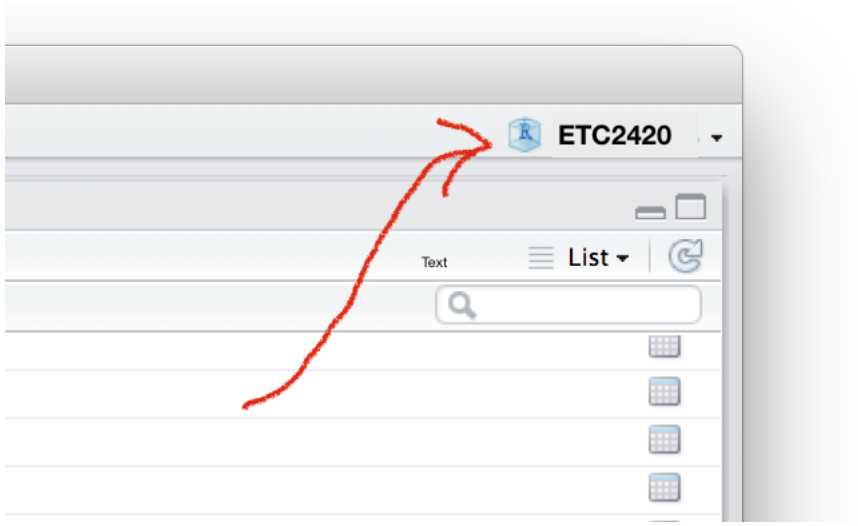

Figure 1: Using projects to organise your work

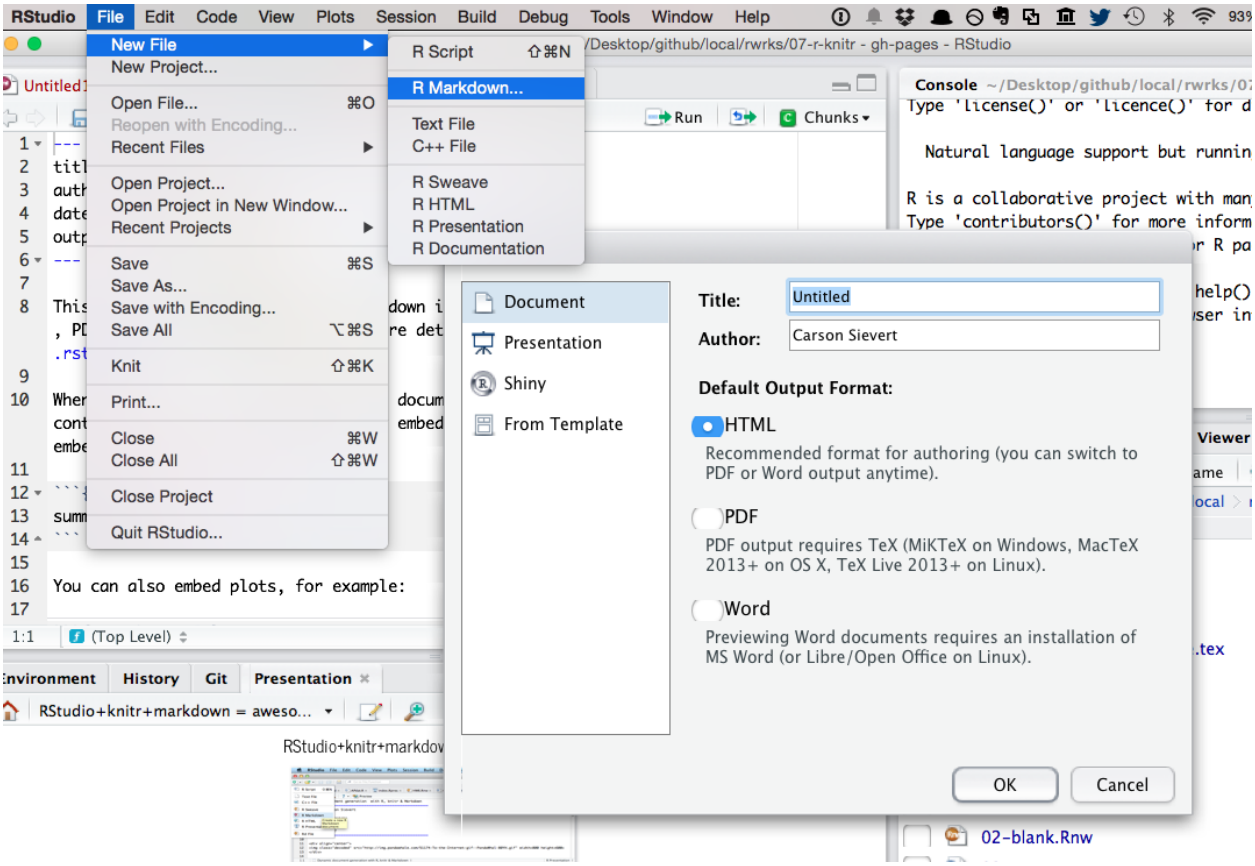

Figure 2: Writing and computing with the one document

Exercise 3

Look at the text in the `lab1.Rmd` document.

- What is R code?
- How does `knitr` know that this is code to be run?
- Using the RStudio IDE, work out how to run a chunk of code. Run this chunk, and then run the next chunk.
- Using the RStudio IDE, how do you run just one line of R code?
- Using the RStudio IDE, how do you highlight and run multiple lines of code?
- What happens if you try to run a line that starts with ““‘{r}”? Or try to run a line of regular text from the document?
- Using the RStudio IDE, `knit` the document into a Word document.

Some R Basics

- Type and Figure out what each of the following command is doing:

```
(100+2)/3
5*10^2
1/0
0/0
(0i-9)^(1/2)
sqrt(2*max(-10,0.2,4.5))+100
x <- sqrt(2*max(-10,0.2,4.5))+100
x
log(100)
log(100,base=10)
```

- Check that these are equivalent: `y <- 100`, `y = 100` and `100 -> y`
- Find the help page for the `mean` command, either from the help menu, or by typing one of these: `help(mean)` and `?mean`. Most help pages have examples at the bottom.
- The `summary` command can be applied to almost anything to get a summary of the object. Try `summary(c(1, 3, 3, 4, 8, 8, 6, 7))`

Data Types

- `list`’s are heterogeneous (elements can have different types)
- `data.frame`’s are heterogeneous but elements have same length (`dim` reports the dimensions and `colnames` shows the column names)
- `vector`’s and `matrix`’s are homogeneous (elements have the same type), which would be why `c(1, "2")` ends up being a character string.
- `function`’s can be written to save repeating code again and again
- Try to understand these commands: `class`, `typeof`, `is.numeric`, `is.vector` and `length`
- See Hadley Wickham’s online chapters on data structures for more

Operations

- Use built-in *vectorized* functions to avoid loops

```
set.seed(1000)
x <- rnorm(6)
x
# [1] -0.44577826 -1.20585657  0.04112631  0.63938841 -0.78655436 -0.38548930
sum(x + 10)
# [1] 57.85684
```

- `R` has rich support for documentation, see `?sum`

- Use `[` to extract elements of a vector.

```
x[1]
# [1] -0.4457783
x[c(T, F, T, T, F, F)]
# [1] -0.44577826  0.04112631  0.63938841
```

- Extract *named* elements with `$`, `[[`, and/or `[`

```
x <- list(
  a = 10,
  b = c(1, "2")
)
x$a
# [1] 10
x[["a"]]
# [1] 10
x["a"]
# $a
# [1] 10
```

Examining 'structure'

- `str()` is a very useful `R` function. It shows you the "structure" of (almost) *any* R object (and *everything* in R is an object!!!)

```
str(x)
# List of 2
#  $ a: num 10
#  $ b: chr [1:2] "1" "2"
```

Missing Values

- `NA` is the indicator of a missing value in R
- Most functions have options for handling missings

```
x <- c(50, 12, NA, 20)
mean(x)
# [1] NA
mean(x, na.rm=TRUE)
# [1] 27.33333
```

Counting Categories

- the `table` function can be used to tabulate numbers

```
table(c(1, 2, 3, 1, 2, 8, 1, 4, 2))
#
# 1 2 3 4 8
# 3 3 1 1 1
```

Functions

One of the powerful aspects of R is to build on the reproducibility. If you are going to do the same analysis over and over again, compile these operations into a function that you can then apply to different data sets.

```
average <- function(x)
{
  return(sum(x)/length(x))
}

y1 <- c(1,2,3,4,5,6)
average(y1)
# [1] 3.5

y2 <- c(1, 9, 4, 4, 0, 1, 15)
average(y2)
# [1] 4.857143
```

Now write a function to compute the mode of some vector, and confirm that it returns `4` when applied on `y <- c(1, 1, 2, 4, 4, 4, 9, 4, 4, 8)`

Exercise 4

- What's an R `package`?
- How do you install a package?
- How does the `library()` function relates to a `package`?
- How often do you load a `package`?
- Install and load the package `ISLR`

Getting data

Data can be found in R packages

```
library(dplyr)
data(economics, package = "ggplot2")
# data frames are essentially a list of vectors
glimpse(economics)
# Observations: 574
# Variables: 6
# $ date     <date> 1967-07-01, 1967-08-01, 1967-09-01, 1967-10-01, 1967...
# $ pce      <dbl> 507.4, 510.5, 516.3, 512.9, 518.1, 525.8, 531.5, 534....
# $ pop      <int> 198712, 198911, 199113, 199311, 199498, 199657, 19980...
# $ psavert  <dbl> 12.5, 12.5, 11.7, 12.5, 12.5, 12.1, 11.7, 12.2, 11.6,...
# $ uempmed  <dbl> 4.5, 4.7, 4.6, 4.9, 4.7, 4.8, 5.1, 4.5, 4.1, 4.6, 4.4...
# $ unemploy <int> 2944, 2945, 2958, 3143, 3066, 3018, 2878, 3001, 2877,...
```

These are not usually kept up to date but are good for practicing your analysis skills on.

Or in their own packages

```
library(gapminder)
glimpse(gapminder)
# Observations: 1,704
# Variables: 6
# $ country   <fctr> Afghanistan, Afghanistan, Afghanistan, Afghanistan,...
# $ continent <fctr> Asia, Asia, Asia, Asia, Asia, Asia, Asia, Asia, Asi...
# $ year      <int> 1952, 1957, 1962, 1967, 1972, 1977, 1982, 1987, 1992...
# $ lifeExp   <dbl> 28.801, 30.332, 31.997, 34.020, 36.088, 38.438, 39.8...
# $ pop       <int> 8425333, 9240934, 10267083, 11537966, 13079460, 1488...
# $ gdpPercap <dbl> 779.4453, 820.8530, 853.1007, 836.1971, 739.9811, 78...
```

I primarily use the `readr` package for reading data now. It mimics the base R reading functions but is implemented in `C` so reads large files quickly, and it also attempts to identify the types of variables.

```
library(readr)
ped <- read_csv("https://raw.githubusercontent.com/bsouhaib/BA2017/master/data/Pedestrian_Counts.csv")
glimpse(ped)
# Observations: 1,392,618
# Variables: 4
# $ Date_Time     <chr> "01-MAY-2009 00:00", "01-MAY-2009 00:00", "01-MA...
# $ Sensor_ID     <int> 4, 17, 18, 16, 2, 1, 13, 15, 9, 10, 12, 11, 5, 6...
# $ Sensor_Name   <chr> "Town Hall (West)", "Collins Place (South)", "Co...
# $ Hourly_Counts <int> 209, 28, 36, 22, 52, 53, 17, 124, 5, 8, 2, 5, 15...
```

You can pull data together yourself, or look at data compiled by someone else.

Question 1

- Look at the `economics` data in the `ggplot2` package. Can you think of two questions you could answer using these variables?
- Write these into your `.Rmd` file.

Question 2

- Read the documentation for `gapminder` data. Can you think of two questions you could answer using these variables?
- Write these into your `.Rmd` file.

Question 3

- Read the documentation for `pedestrian sensor` data. Can you think of two questions you could answer using these variables?
- Write these into your `.Rmd` file.

Question 4

1. Read in the OECD PISA data (available at https://raw.githubusercontent.com/bsouhaib/BA2017/master/data/student_sub.rds)
2. Tabulate the countries (CNT)
3. Extract the values for Australia (AUS) and Shanghai (QCN)
4. Compute the average and standard deviation of the reading scores (PV1READ), for each country
5. Write a few sentences explaining what you learn about reading in these two countries.

TURN IN

- Your `.Rmd` file
- Your Word (or pdf) file that results from knitting the Rmd.
- DUE: Sunday 11:55pm, loaded into Moodle

Resources

- RStudio IDE cheat sheet
- rmarkdown cheat sheet
- Q/A site: http://stackoverflow.com
- Dynamic Documents with R and knitr, Yihui Xie,