R Statistical Software - An Introduction

UCLA - Econ 102 - Fall 2018

François Geerolf

Disclaimer: this is optional

This note is a very quick introduction to the R statistical software. You do not need to do any of this, or even download R Statistical Software, to follow 102. This is only extra for those who want to play with the data, do their own research, or even fact-check me, etc.

What is R?

R is a free software environment for statistical computing and graphics. It is released under the GNU General Public License, and an alternative to other popular commercial softwares such as Stata. Many statisticians and data scientists use R (and python) for exploring data.

Again, you do not have to, but you might want use Econ 102 as an excuse to teach yourself some elements of R. Macroeconomic data is constantly evolving in real-time, and R allows you to retrieve real-time data from up-to-date sources, as well as work with these datasets, quite easily. Learning a statistical software would also help you think critically about macroeconomic analysis that is typically found in the *Wall Street Journal*, the *New York Times*, and other reliable source of information, which often make heavy use of data. R make computations easy (such as transforming values to growth rate), detrending, as well as presenting data in the best way possible (such as plotting the time series of economic quantities, etc.).

Moreover, California is currently experiencing a shortage for data science skills. Even the consulting and the finance industry increasingly require knowing how to manipulate datasets. I view macroeconomics as an occasion to teach you these general purpose skills. Finally, these skills might also prove useful when you take Economics 103, in which you will learn more thoroughly the tools of regression analysis (if you have not already). Again, you do not at all need to learn R in order to succeed in this class, I am only providing you some material for those who are interested and want to do more.

Getting started with R Statistical Software

Downloading. You need to install `R` and `Rstudio`:

1. First you must get the **R statistical software**, which you may download on the UCLA website here. The latest release (2018-07-02, Feather Spray) is version 3.5.1. For Mac OSX: download here. For Windows: download here.
2. Second, I recommend you use a Graphical User Interface (GNU) for R such as **R Studio**. R Studio's latest release is 1.1.456: download here.

Introduction to R. Cheatsheets are a great way to get started on R. Many are available here, but the 2 main cheatsheets are:

- Base R Cheatsheet.
- Advanced R Cheatsheet

Packages

I use `tidyverse`, from Hadley Wickham, for data manipulation as well as plotting data. This cheatsheet has a beginner's introduction to `tidyverse`. `tidyverse` is a powerful collection of R packages that are data tools for transforming and visualizing data. Datacamp has a free tutorial for `tidyverse`, which can get you started. The following packages are particularly useful:

- `dplyr` for data manipulation. Cheatsheet. Note, in particular, the use of pipes %>%:
  - x %>% f(y) is the same as f(x, y).
  - y %>% f(x, ., z) is the same as f(x, y, z).
  - "Piping" with %>% makes code more readable.

```
iris %>%
  group_by(Species) %>%
  summarise(avg = mean(Sepal.Width)) %>%
  arrange(avg)
```

```
## # A tibble: 3 x 2
##   Species      avg
##   <fct>      <dbl>
## 1 versicolor  2.77
## 2 virginica   2.97
## 3 setosa      3.43
```

- `ggplot2` for data visualization. Cheatsheet.
- `stringr` for string manipulation. Cheatsheet. Cheatsheet on Regular Expressions.

In addition to the `tidyverse` collection of R packages, I also use the following packages:

- `lubridate` for working with dates (very useful in macroeconomics !). Cheatsheet.

`tidyverse` also contains `readr` which allows to read in data. Cheatsheet

R-markdown

My lecture notes are created using `R-markdown`, which you can learn using this cheatsheet, as well as this reference guide.

An example from Lecture 1

The power of R is now illustrated using the first Figure in Lecture 1.

Data for GDP is taken from the Bureau of Economic Analysis's National Income and Product Accounts (NIPA) here and data for Personal Consumption Expenditures is taken from there. Note that the BEA's website is here, and that you get access to this data clicking on **Section 1 - Domestic Product and Income** and then on Table 1.1.5. which has **Gross Domestic Product (A) (Q)** (**A**nnual and **Q**uarterly). However, instead of retrieving this data from the BEA, I use the `rdbnomics` R-package which retrieves data from this website. You must first install and load this package through the following code:

```
devtools::install_github("dbnomics/rdbnomics")
pklist <- c("tidyverse", "lubridate", "rdbnomics")
source("https://fgeerolf.github.io/datasets/load-packages.R")
```

Note that I also loaded `tidyverse` containing in particular `ggplot2`, `dplyr`and `stringr` mentioned above, as well as `lubridate` for working easily with dates. The code that you can see in lecture 1 showing the R-code looks like this:

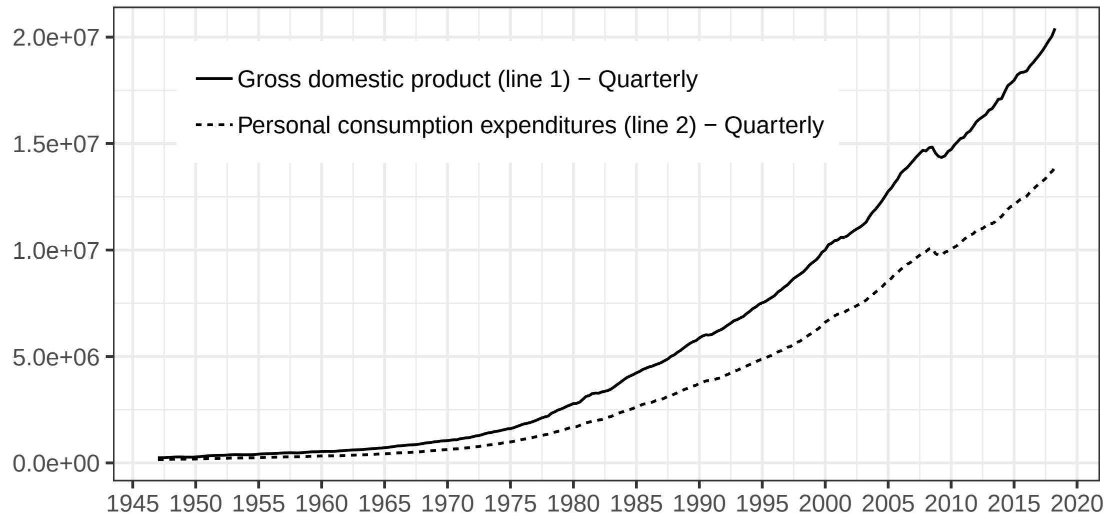

Figure 1: US GDP from NIPA (BEA)

```
rdb(ids = c('BEA/NIPA-T10105/A191RC-Q', "BEA/NIPA-T10105/DPCERC-Q")) %>%
  mutate(year = year(period),
         month = month(period),
         yearmonth = year + (month - 1)/12) %>%
  select(yearmonth, value, series_name) %>%
  ggplot(aes(x = yearmonth, y = value, linetype = series_name)) + geom_line() + theme_bw() +
  theme(legend.title = element_blank(),
        legend.position = c(0.4, 0.8)) +
  scale_x_continuous(breaks = seq(1920, 2025, 5)) + xlab("") + ylab("")
```

The first command loads in the data:

```
rdb(ids = c('BEA/NIPA-T10105/A191RC-Q', "BEA/NIPA-T10105/DPCERC-Q")) %>%
  as.tibble %>%
  head
```

```
## # A tibble: 6 x 12
##   series_name Frequency dataset_name `@frequency` Concept concept
##   <chr>       <chr>     <chr>        <chr>        <chr>   <chr>
## 1 Gross dome~ Quarterly Table 1.1.5~ quarterly    Gross ~ gross-~
## 2 Gross dome~ Quarterly Table 1.1.5~ quarterly    Gross ~ gross-~
## 3 Gross dome~ Quarterly Table 1.1.5~ quarterly    Gross ~ gross-~
## 4 Gross dome~ Quarterly Table 1.1.5~ quarterly    Gross ~ gross-~
## 5 Gross dome~ Quarterly Table 1.1.5~ quarterly    Gross ~ gross-~
## 6 Gross dome~ Quarterly Table 1.1.5~ quarterly    Gross ~ gross-~
## # ... with 6 more variables: provider_code <chr>, series_code <chr>,
## #   FREQ <chr>, dataset_code <chr>, period <date>, value <dbl>
```

A series of manipulations then transform the dates into numeric form, and keeps only the useful information for plotting: the date, the value, and the series name:

```
rdb(ids = c('BEA/NIPA-T10105/A191RC-Q', "BEA/NIPA-T10105/DPCERC-Q")) %>%
  mutate(year = year(period),
         month = month(period),
         yearmonth = year + (month - 1)/12) %>%
  select(series_name, yearmonth, value) %>%
```

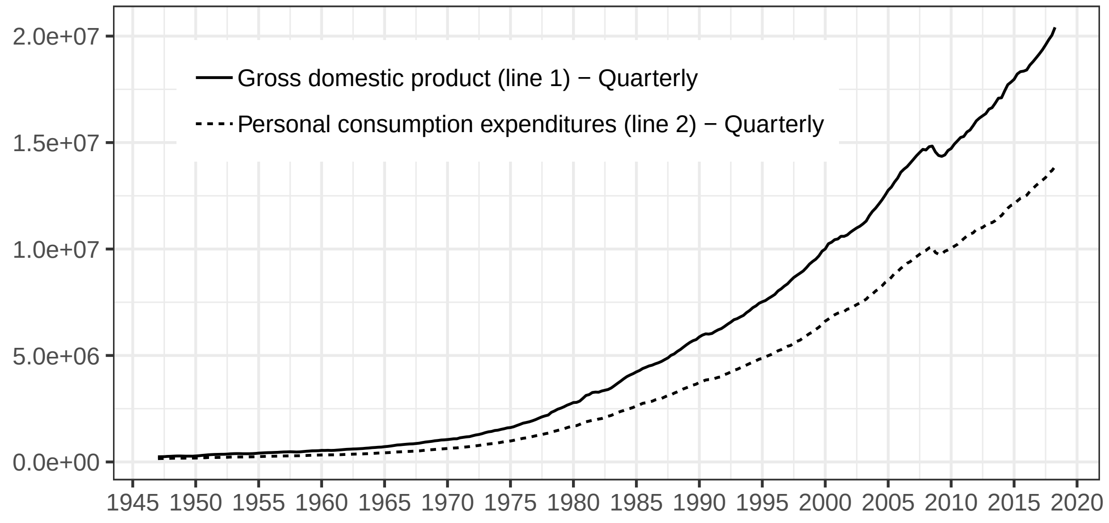

Figure 2: US GDP from NIPA (BEA)

```
  as.tibble %>%
  head
```

```
## # A tibble: 6 x 3
##   series_name                                 yearmonth  value
##   <chr>                                           <dbl>  <dbl>
## 1 Gross domestic product (line 1) - Quarterly     1947  243164
## 2 Gross domestic product (line 1) - Quarterly     1947. 245968
## 3 Gross domestic product (line 1) - Quarterly     1948. 249585
## 4 Gross domestic product (line 1) - Quarterly     1948. 259745
## 5 Gross domestic product (line 1) - Quarterly     1948  265742
## 6 Gross domestic product (line 1) - Quarterly     1948. 272567
```

Finally, ggplot2 is used to plot this data:

```
rdb(ids = c('BEA/NIPA-T10105/A191RC-Q', "BEA/NIPA-T10105/DPCERC-Q")) %>%
  mutate(year = year(period),
         month = month(period),
         yearmonth = year + (month - 1)/12) %>%
  select(yearmonth, value, series_name) %>%
  ggplot(aes(x = yearmonth, y = value, linetype = series_name)) + geom_line() + theme_bw() +
  theme(legend.title = element_blank(),
        legend.position = c(0.4, 0.8)) +
  scale_x_continuous(breaks = seq(1920, 2025, 5)) + xlab("") + ylab("")
```

legend.title = element_blank() removes the legend title, legend.position puts the legend into the graph. You can figure out using the ggplot2 Cheatsheet, or just googling them, what the other options are doing.