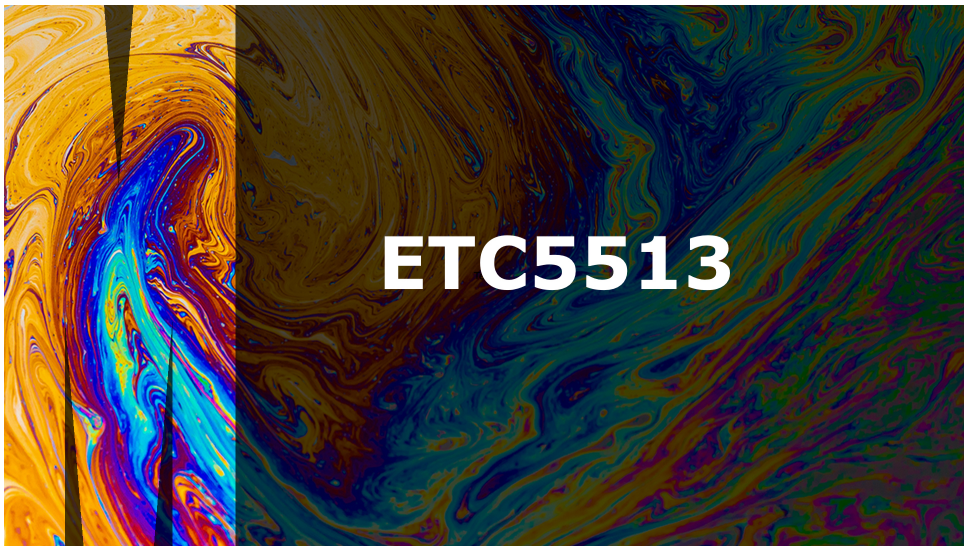

Reproducible and Collaborative Practices

Tutorial 2

Patricia Menendez

(updated: 2020-03-13)

Contents

Tutorial objectives:

- Working on a reproducible Rstudio project
- Produce an html report and examine different YAML themes
- Practice markdown syntax
- Practise R
- Explore R chunk options
- Gain experience on data wrangling using the *tidyverse* suit of packages.
- Producing exploratory data analysis figures using the package *ggplot2*
- Learn how to add figure captions
- Create html tables and learn how to add captions

1 Exercise 1: Rstudio project using Rstudio Cloud

I have created a project for you on Rstudio Cloud to get you started in the usage of R, Rstudio and markdown. Please follow the instructions below:

1. Click on this link and login into your Rstudio Cloud.
2. Save the project locally in your Rstudio Cloud space.
3. Open the file called Tutorial2.Rmd
4. Knit the file
5. Write your name as an author in the YAML
6. Change the html theme to *cerulean*

During the tutorial you will be adding text and R chunks into this file to create your own report.

2 Exercise 2: YAML and R chunk options

1. Carefully inspect the file YAML as well as the first Rcode chunk.
2. What is the first R chunk of the code doing? (**Hint:** All the libraries used in any analysis should be listed together at the top of the file.)
3. Change the R chunck option from $message = FALSE$ to $message = TRUE$ and see what happends when you knit the file.
4. Create a new section (**Hint:** Use #) called Introduction and type using markdown the following: "In this tutorial we are looking at the **Corona virus** cases detected within the Hubei area as reported in the Lancet journal website as of March 12, 2019."
5. Remove all the R chunk messages from the Chunk called (Chunk 1) and write the following under that section using markdown: "In this section we are loading all the required libraries for the tutorial."
6. For the same R chunk (Chunk 1) add the R chunk option $echo = FALSE$ and see what is the effect of including that command.
7. Using markdown, link the word "Lancet" to the website https://www.thelancet.com/coronavirus

3 Exercise 3: Hands on practice with COVID19 data

1. The data for the tutorial is inside the folder called Data inside the Rstudio Cloud project. Have a look at it on the lower right pane.
2. Create a new section heading in your Rmd document to read the data with the following title "Reading corona virus data". (**Hint:** Use #)
3. Inside your new section, create a new R chunk (with options $echo = TRUE$, $warning = FALSE$, $message = FALSE$) called "Reading data" and insert the following code:

```
dat <- read_csv("Data/COVID19_March12_Hubei.csv")
```

4. Insert a new R chuck as follows and find out which information about the data can you get from the command

```
head(dat)
```

5. Modify the command *head* to display 10 rows
6. Create another two R chunks and use in each of them the R functions glimpse() and str()

- What information can you get from those commands? (**Hint:** For more info on R functions type in the R console ?glimpse())
- Using **R inline commands** write the the dimension of the data set in a sentence. (**Hint:** Have a look at *ncol* and *nrow*)

7. Add a new subsection headding (###) with "Why is it important to know the dimension of your data set?" and write a brief sentence with the explanation.
8. Add a new subsection headding (###) with "What are the variable names in the data set?" and display the names of the data set variables using R (**Hint:** Querry → ?names() in the R console)

- Select two variables and use a markdown list to briefly explain what each of the variables are measuring.

4 Exercise 4: COVID19 Data wrangling

1. Using the R package *dplyr* loaded within the *tidyverse* library and using pipes %>%, create a new data set called *dat2* that only contains the following variables: **country, age, sex, city, province, latitude, logitude** (**Hint: see code below**)

```
dat2 <- dat %>% dplyr::select(country,
                              age,
                              sex,
                              city,
                              province,
                              latitude,
                              longitude)
```

2. Inspect *dat2* and describe using a markdown a list the type of variables in this new data set. Write the name of the variables in bold font. Do you think the variable attributes are correct?
3. Convert the variable age into a numeric vector
4. Inspect the first 20 values of *age*. What do you observe? What is the proportion of missing values in the variable *age*? Make sure you round the results to two decimal numbers.
5. Remove cases for which we don't have information on the person age and keep cases for which the gender of the patience is known. Give this new data set the name dat3.
6. What is the dimension of this new data set? Compare it with the dimension of dat2? How many cases have we lost?
7. Examine the variable age using the function *summary()*. Do you see any problems in the data?
8. Remove patients entries with age below 1 and name the new data set dat4
9. Provide a table summary of the variable age using the *kable()* function from the #kableExtra* package.

10. Visualize the age distribution using a histogram and give an explanation about the information that a histogram convey. In addition change the x label in the plot to *Age* and remove the y axis label. (**Hint**: Use ggplot2 package within the tidyverse library)
11. Change the color of the histogram using *geom_histogram(color = "blue", fill = "white")*
12. Visualize the age distribution for females and males and add the following caption to the figure *"Age frequencies of COVID19 patients in China per gender"* (**Hint**: facet_wrap())
13. Count the number of cases per province and display a table of the top 25 countries. Store results into an object called dat5. **Hint**: Replace XXX by the adequate variable names in the code below.

```
dat5 <- dat4 %>%

  dplyr::select(XXX) %>%

  dplyr::filter(!is.na(XXX)) %>%

  group_by(province) %>%

  mutate(Cases = n()) %>%

  unique() %>%

  arrange(-XXX)
```