R markdown guide

Jongoh Kim

21 December, 2022

Contents

1 Introduction

This short markdown file is written to provide a simple example of an R markdown to LISER researchers. The aim of this document to cover all the essential cases that could be used in R markdown. Again, the gapminder data set from the *gapminder* package will be used. You can check the gapminder website by clicking this link.

2 Code chunk in R markdown

2.1 Creating a code chunk

To create an R code chunk, type 3 backticks(‘), curley brackets({}), and ‘r’ inside those brackets. Backtick is a symbol located next to 1 for a *QWERTY* keyboard. Then close the code chunk by typing 3 backticks again. For instance,

```
#declaring libraries
packages <- c("dplyr", "gapminder", "ggplot2")
lapply(packages, require, character.only = T)
```

```
## Loading required package: dplyr

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

## Loading required package: gapminder

## Loading required package: ggplot2

## [[1]]
## [1] TRUE
##
## [[2]]
## [1] TRUE
##
## [[3]]
## [1] TRUE
```

```
print("Packages are loaded!")
```

```
## [1] "Packages are loaded!"
```

I have set the name of the code chunk as “reading-data”. If an error occurs in this code chunk, the console will produce an error message saying that an error occurred in the “reading-data” code chunk.

2.2 Not including the code lines of a code chunk

Actually, there is no reason to show the code lines calling the packages. It is more aesthetically appealing to write which packages were used and not to show the code lines. Let's try not to include those code lines in our document.

```
## [[1]]
## [1] TRUE
##
## [[2]]
## [1] TRUE
##
## [[3]]
## [1] TRUE

## [1] "Packages are loaded!"
```

Now the document does not show any code line.

2.3 Not including the output of the codes

For this case, the output of the code lines, printing packages are loaded or TRUE values from lapply function, does not have to be shown in the document. On the contrary, it is aesthetically better to remove them.

As you can see, the code chunk & output have disappeared magically.

2.4 Not running the code chunk

On the other hand, sometimes, you simply want to show the code lines but not want them to run. Let's see how this could be done.

```
# I simply want to show this code chunk
"for no reason I just feel like it"
print(data == bad)
```

Voilà!

3 Text format

3.1 Bold text

To make a text bold simply type asterisk(*) two times when you would like to start and another two asterisks at the end. For instance, if you want to make Luxembourg Institute of Socio-Economic Research bold, **Luxembourg Institute of Socio-Economic Research**.

3.2 Italic text

Italic is almost same as bold text but typing asterisk only one time. Let's add LISER right next to our bold text in parentheses.

Luxembourg Institute of Socio-Economic Research(*LISER*)

3.3 Bullet points

If you want to make a list of somethings and put a bullet point in front of them, simply use a hyphon(-) in front of them. Don't forget to give a space between the hyphon and the character for the markdown to recognize that you want to create an un-ordered list with bullet points! Also an empty line should be placed between the last sentence and the first line of the bullet point.

To list three major research departments of LISER:

Luxembourg Institute of Socio-Economic Research(*LISER*)

- Labor Market(LM)
- Urban Development and Mobility(UDM)
- Living Conditions(LC)

3.4 Ordered list

If you want to put numbers instead of bullet points, simply use the number and a full stop(.). Of course you can combine both types. For instance,

Luxembourg Institute of Socio-Economic Research(*LISER*)

1. Labor Market(LM)
2. Urban Development and Mobility(UDM)
    - ACROSS
3. Living Conditions(LC)

3.5 Mathematical expressions

The way to write mathematical expressions in R Markdown is identical to Latex.

$Y = C + I + G + NX$

To add an aligning option,

$$Y = C + I + G + NX$$

3.6 adding a hyperlink

Sometimes you want to add a hyperlink to certain words. Let's add a hyperlink to LISER in the next line.

LISER

3.7 adding a footnote

Adding a footnote is much similar to adding a hyperlink. I will try to add a footnote in the next line.

Luxembourg has three official languages[1].

[1]The three languages are Luxembourgish, French, and German.

4 Real Example

4.1 Setting the working directory

In case your primary data file is not stored in the same path as your project, it easier to set the working directory to where the data is located. For instance,

```
knitr::opts_knit$set(root.dir = "C:/Users/jongoh/Dropbox/LISER RA/LISER/training/R A-Z")
```

After this, you can read data files that are located in "C:\Users\Jongoh\Dropbox\training\R A-Z\data" by simply typing *fread("data\data_file.csv")*.

4.2 Calling packages

```
#declaring libraries
packages <- c("knitr", "dplyr", "gapminder", "ggplot2")
lapply(packages, require, character.only = T)
```

```
## Loading required package: knitr
```

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

4.3 Reading in data

```
#reading the gapminder data
data <- gapminder
```

4.4 Simple summary statistics

```
print(str(data))
```

```
## tibble [1,704 x 6] (S3: tbl_df/tbl/data.frame)
##  $ country  : Factor w/ 142 levels "Afghanistan",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ continent: Factor w/ 5 levels "Africa","Americas",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ year     : int [1:1704] 1952 1957 1962 1967 1972 1977 1982 1987 1992 1997 ...
##  $ lifeExp  : num [1:1704] 28.8 30.3 32 34 36.1 ...
```

```
##  $ pop      : int [1:1704] 8425333 9240934 10267083 11537966 13079460 14880372 12881816 13867957 163
##  $ gdpPercap: num [1:1704] 779 821 853 836 740 ...
## NULL
```

```
print(summary(data$continent))
```

```
##   Africa Americas     Asia   Europe  Oceania
##      624      300      396      360       24
```

```
print(summary(data$lifeExp))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   23.60   48.20   60.71   59.47   70.85   82.60
```

4.5 Simple summary table

Creating a simple summary table for life expectancy by continent!

```
#creating a simple summary table for life expectancy by continent!
sum.df <- data %>%
  group_by(continent) %>%
  summarise( Life_Min = min(lifeExp),
             Life_Average = mean(lifeExp),
             Life_Median = median(lifeExp),
             Life_Max = max(lifeExp),
             Count =n())
sum.df
```

```
## # A tibble: 5 x 6
##   continent Life_Min Life_Average Life_Median Life_Max Count
##   <fct>        <dbl>        <dbl>       <dbl>    <dbl> <int>
## 1 Africa        23.6         48.9        47.8     76.4   624
## 2 Americas      37.6         64.7        67.0     80.7   300
## 3 Asia          28.8         60.1        61.8     82.6   396
## 4 Europe        43.6         71.9        72.2     81.8   360
## 5 Oceania       69.1         74.3        73.7     81.2    24
```

Let's put in the document neatly!

```
#showing the table neatly
kable(sum.df, caption = "Simple Summary Table of Life Expectancy by Continent")
```

Table 1: Simple Summary Table of Life Expectancy by Continent

continent	Life_Min	Life_Average	Life_Median	Life_Max	Count
Africa	23.599	48.86533	47.7920	76.442	624
Americas	37.579	64.65874	67.0480	80.653	300
Asia	28.801	60.06490	61.7915	82.603	396
Europe	43.585	71.90369	72.2410	81.757	360

continent	Life_Min	Life_Average	Life_Median	Life_Max	Count
Oceania	69.120	74.32621	73.6650	81.235	24

4.6 Declare variables and use them in a document

One of the most annoying thing to do when one is writing a paper or a report is when a number has changed and you have to go through the whole document to change the numbers. This is avoidable in R Markdown and I will show you how.

Let's calculate the number of countries by continent.

```
count_country.df <- data %>%
  group_by(continent, year) %>%
  summarize(Count = n(),
            .groups = "keep") %>% #keeping the grouped variables
  ungroup() %>%
  select(-year) %>%
  unique()
count_country.df
```

```
## # A tibble: 5 x 2
##   continent Count
##   <fct>     <int>
## 1 Africa       52
## 2 Americas     25
## 3 Asia         33
## 4 Europe       30
## 5 Oceania       2
```

Now let's assign the values to each variable.

```
africa_num <- count_country.df %>% filter(continent=="Africa") %>% select(Count) %>% unlist()
america_num <- count_country.df %>% filter(continent=="Americas") %>% select(Count) %>% unlist()
asia_num <- count_country.df %>% filter(continent=="Asia") %>% select(Count) %>% unlist()
europe_num <- count_country.df %>% filter(continent=="Europe") %>% select(Count) %>% unlist()
oceania_num <- count_country.df %>% filter(continent=="Oceania") %>% select(Count) %>% unlist()
```

Then you can easily use them in the text.

For instance,

After counting the number of countries in each continent, Africa has 52 countries, Americas 25, Asia 33, Europe 30, and Oceania 2.

Amazing right? Let's assume that we obtained additional observation from a beautiful island Fiji. If this was a word document, I should go through the whole document to find where I mentioned the number of Oceaninan countries.

In R Markdown, if the dataset was updated, you don't have to change anything. I will show you.

I will manually update the *data* variable as a showcase.

```r
#first changing the factor variable to character
data$country <- as.character(data$country)
#adding a row
data <- rbind(data, list(country = rep("Fiji", length(unique(data$year))),
                  continent = rep("Oceania", length(unique(data$year))),
                  year = unique(data$year),
                  lifeExp = rep(999, length(unique(data$year))),
                  pop=rep(999, length(unique(data$year))),
                  gdpPercap = rep(999, length(unique(data$year)))
                  )
              )
#changing to factor again
data$country <- as.factor(data$country)
print(data %>% filter(continent == "Oceania") %>% select(country) %>% unique())
```

```
## # A tibble: 3 x 1
##   country
##   <fct>
## 1 Australia
## 2 New Zealand
## 3 Fiji
```

Now, let's write the same line after declaring the numbers again.

```r
#the data frame again
count_country.df <- data %>%
  group_by(continent, year) %>%
  summarize(Count = n(),
            .groups = "keep") %>%
  ungroup() %>%
  select(-year) %>%
  unique()

#numbers
africa_num <- count_country.df %>% filter(continent=="Africa") %>% select(Count) %>% unlist()
america_num <- count_country.df %>% filter(continent=="Americas") %>% select(Count) %>% unlist()
asia_num <- count_country.df %>% filter(continent=="Asia") %>% select(Count) %>% unlist()
europe_num <- count_country.df %>% filter(continent=="Europe") %>% select(Count) %>% unlist()
oceania_num <- count_country.df %>%
  filter(continent=="Oceania") %>%
  select(Count) %>%
  unlist()
```

After counting the number of countries in each continent, Africa has 52 countries, Americas 25, Asia 33, Europe 30, and Oceania 3.

4.7 putting nice graphs

Let's use the graph we produced in the ggplot section.

```
include_graphics("C:/Users/jongoh/Dropbox/LISER RA/LISER/training/R A-Z/img/output.jpg")
```

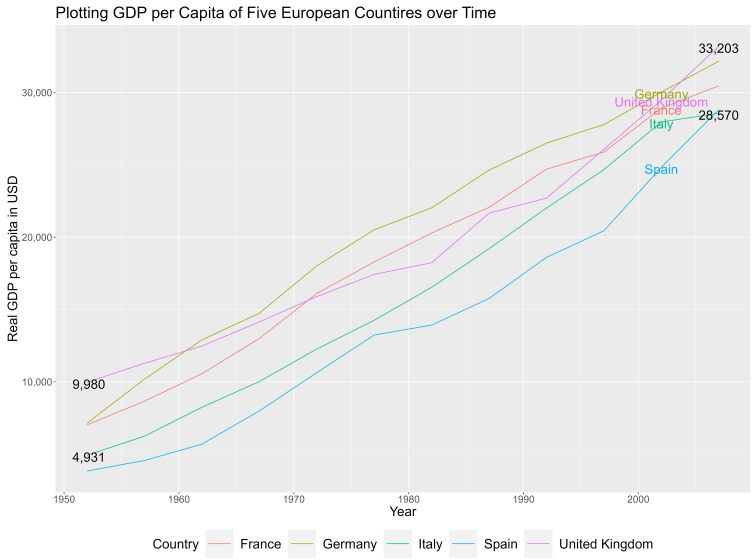

Figure 1: Pretty Graph

5 Concluding

That's all for this R Markdown example file. I hope this file was clear for you and if you have any questions, don't hesitate to contact me. jongoh.kim@liser.lu