

# Autour du SARS-CoV-2 (Covid-19)

Franck Bonardi

## Contents

Subject . . . . .	1
Data preprocessing . . . . .	1
Exploring data . . . . .	2
Formatting data . . . . .	3

## Subject

The goal here is to reproduce graphs similar to those of the South China Morning Post (SCMP), on the The Coronavirus Pandemic page and which show for different countries the cumulative number (i.e. the total number of cases since beginning of the epidemic) of people with coronavirus disease 2019.

## Data preprocessing

The data that we will use initially are compiled by the Johns Hopkins University Center for Systems Science and Engineering (JHU CSSE) and are made available on GitHub. It is more particularly on the data `time_series_covid19_confirmed_global.csv` (chronological suites in csv format) available at the address: [https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse\\_covid\\_19\\_data/csse\\_covid\\_19\\_time\\_series\\_virtual\\_series\\_virtual\\_series\\_video\\_series\\_video\\_series\\_video\\_series\\_video\\_series\\_video\\_series\\_video\\_social](https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_time_series/csse_covid_19_time_series_virtual_series_virtual_series_video_series_video_series_video_series_video_series_video_series_video_social), which we will focus on.

```
#Load librairies
library(dplyr)
library(tidyr)
library(ggplot2)
library(scales)
library(directlabels)
library(magrittr)
```

```
data_url = "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_time_series/csse_covid_19_time_series_virtual_series_virtual_series_video_series_video_series_video_series_video_series_video_series_video_social"
```

The time series table that we use is for the global confirmed cases. Australia, Canada and China are reported at the province/state level. Dependencies of the Netherlands, the UK, France and Denmark are listed under the province/state level. The US and other countries are at the country level.

This is the quick documentation of the data :

Column name	Description
Province/State	If completed, state or province name.
Country/Region	Name of the country/Region

Column name	Description
Lat	Latitude of country, state or province
Long	Longitude of country, state or province

Then, each column represents a day from January 25th 2020 until now.

## Download

Reading the CSV file and convert all dates in each column to **mm/dd/yy** format

```
full.data = read.csv(data_url, stringsAsFactors = F)
names(full.data)[-1:-4] <- format(as.Date(names(full.data)[-1:-4]),
                                format = "%m/%d/%y"), "%m/%d/%y")
```

## Exploring data

Let's have a look at what we got:

```
head(full.data[,1:10])
```

```
## Province.State Country.Region Lat Long 01/22/20 01/23/20
## 1 Afghanistan 33.93911 67.70995 0 0
## 2 Albania 41.15330 20.16830 0 0
## 3 Algeria 28.03390 1.65960 0 0
## 4 Andorra 42.50630 1.52180 0 0
## 5 Angola -11.20270 17.87390 0 0
## 6 Antigua and Barbuda 17.06080 -61.79640 0 0
## 01/24/20 01/25/20 01/26/20 01/27/20
## 1 0 0 0 0
## 2 0 0 0 0
## 3 0 0 0 0
## 4 0 0 0 0
## 5 0 0 0 0
## 6 0 0 0 0
```

```
tail(full.data[,1:10])
```

```
## Province.State Country.Region Lat Long 01/22/20
## 261 Western Sahara 24.21550 -12.88580 0
## 262 Sao Tome and Principe 0.18640 6.61310 0
## 263 Yemen 15.55273 48.51639 0
## 264 Comoros -11.64550 43.33330 0
## 265 Tajikistan 38.86100 71.27610 0
## 266 Lesotho -29.61000 28.23360 0
## 01/23/20 01/24/20 01/25/20 01/26/20 01/27/20
## 261 0 0 0 0 0
## 262 0 0 0 0 0
## 263 0 0 0 0 0
## 264 0 0 0 0 0
## 265 0 0 0 0 0
## 266 0 0 0 0 0
```

Are there missing data points?

```
na_records = apply(full.data, 1, function (x) any(is.na(x)))
sum(na_records)
```

```
## [1] 0
```

```
# full.data[na_records,]
```

There are 0 missing values in this dataset.

## Formatting data

For this analysis, we need to perform several transformations. First of all, we keep the raw data in the variable **full.data** then by using the verbs of dplyr we remove the columns on latitude and longitude because they are not useful to us. Then, we keep only the countries that we want to analyze in this project (see list of countries in the variable **countries**, knowing that for France, the United Kingdom and Netherlands, we are not interested in the colonies, so we filter to keep only the metropolitan territories.

```
# Define the countries we want to subset for the analysis
countries <- c("Belgium", "China", "Hong Kong", "France", "Germany",
              "Iran", "Italy", "Japan", "Korea, South", "Netherlands",
              "Portugal", "Spain", "United Kingdom", "US")

# Select desired countries for the analysis and remove latitude and longitude informations
# and apply a custom filter for the countries
data <- full.data %>%
  select(-starts_with("Lat"), -starts_with("Long")) %>%
  filter(Country.Region %in% countries) %>%
  filter(!(Country.Region %in% c("France", "United Kingdom", "Netherlands")
    & Province.State != ""))
```

In this study, China is a special case. We need to isolate Hong Kong from its country. This is why we voluntarily modify the variable **Country.Region** for the region of Hong Kong and we replace “China” by “China, Hong Kong” Then we need to bring all the provinces of China together in one piece of information. We have chosen to sum up all of these provinces and bring them together in a single row.

```
# Rename country for the specific case of Hong Kong
data$Country.Region[which(data$Province.State == "Hong Kong"
  & data$Country.Region == "China")] <- "China, Hong-Kong"

data$Country.Region <- as.factor(data$Country.Region)
data$Province.State <- as.factor(data$Province.State)

# Summarize the information for all the provinces of China, except for Hong Kong
data <- data[,2:ncol(data)] %>%
  group_by(Country.Region) %>%
  summarise_all(funs(sum))
```

## Inspection

Finally, we can look at our data by plotting.

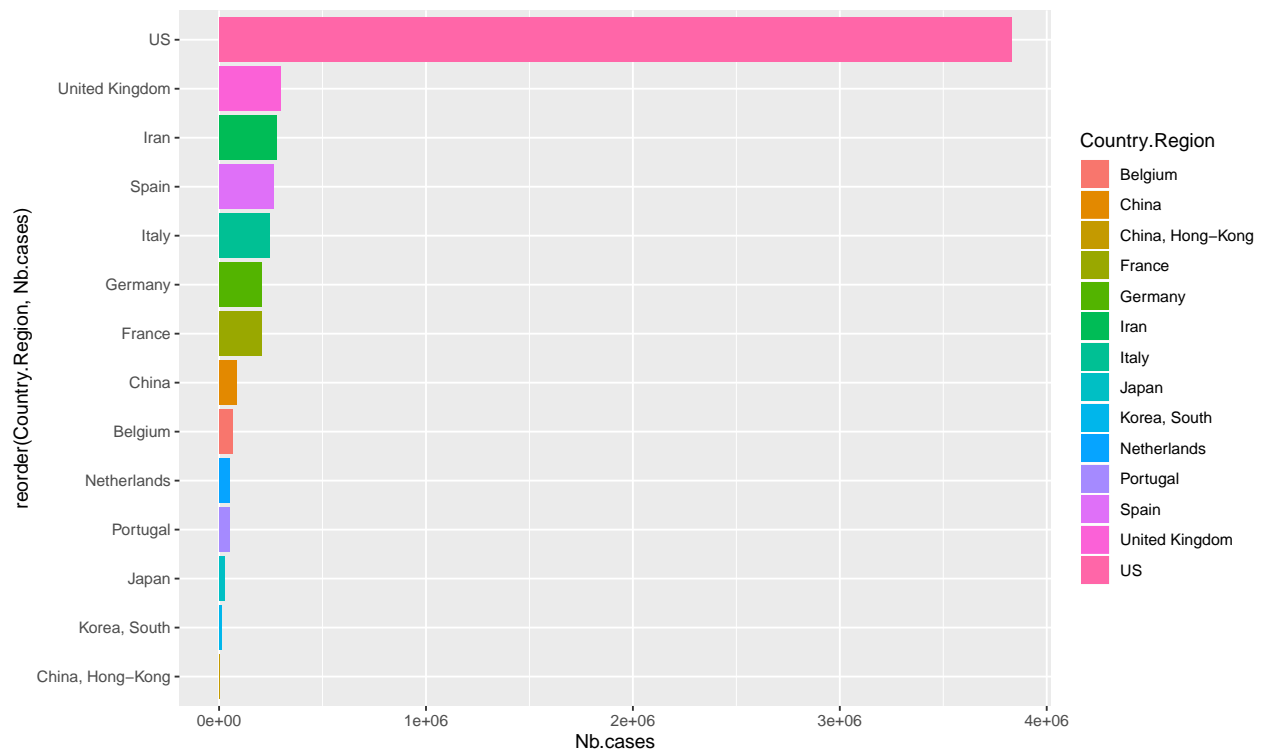
The first plot that we propose is to look at the number of cumulative cases for all the countries that we have decided to observe. Here we can clearly see that the US has a very high number of cases compared to other countries. It will however be necessary to wait for the end of the epidemic to make a plot which will take into account the population of each country, by normalizing for example per 1000 inhabitants.

```
mini.data <- gather(data, "Date", "Nb.cases", 2:ncol(data))
mini.data$Date <- as.Date(mini.data$Date, format="%m/%d/%Y")
class(mini.data$Date)
```

```
## [1] "Date"
```

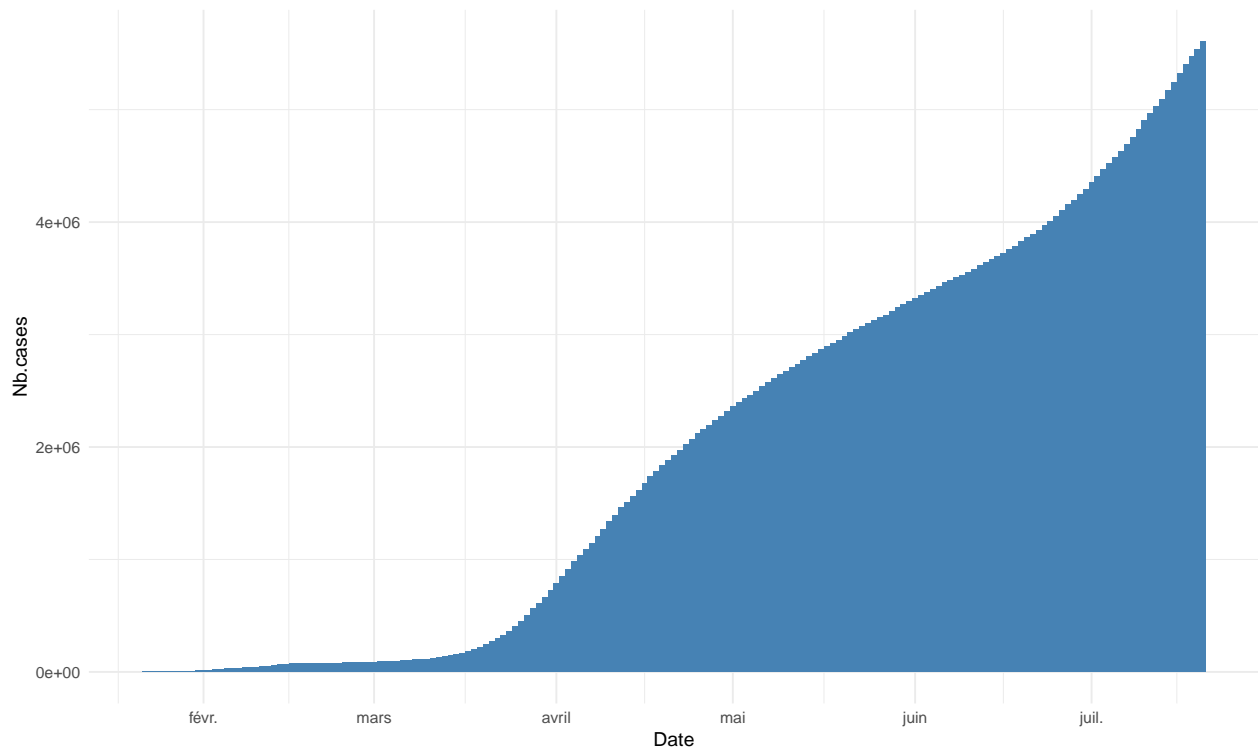
```
last.data <- mini.data %>%
  group_by(Country.Region) %>%
  summarise_each(funs(max))
```

```
ggplot(last.data, aes(x = reorder(Country.Region, Nb.cases), y = Nb.cases)) +
  geom_col(aes(fill = Country.Region)) +
  coord_flip()
```



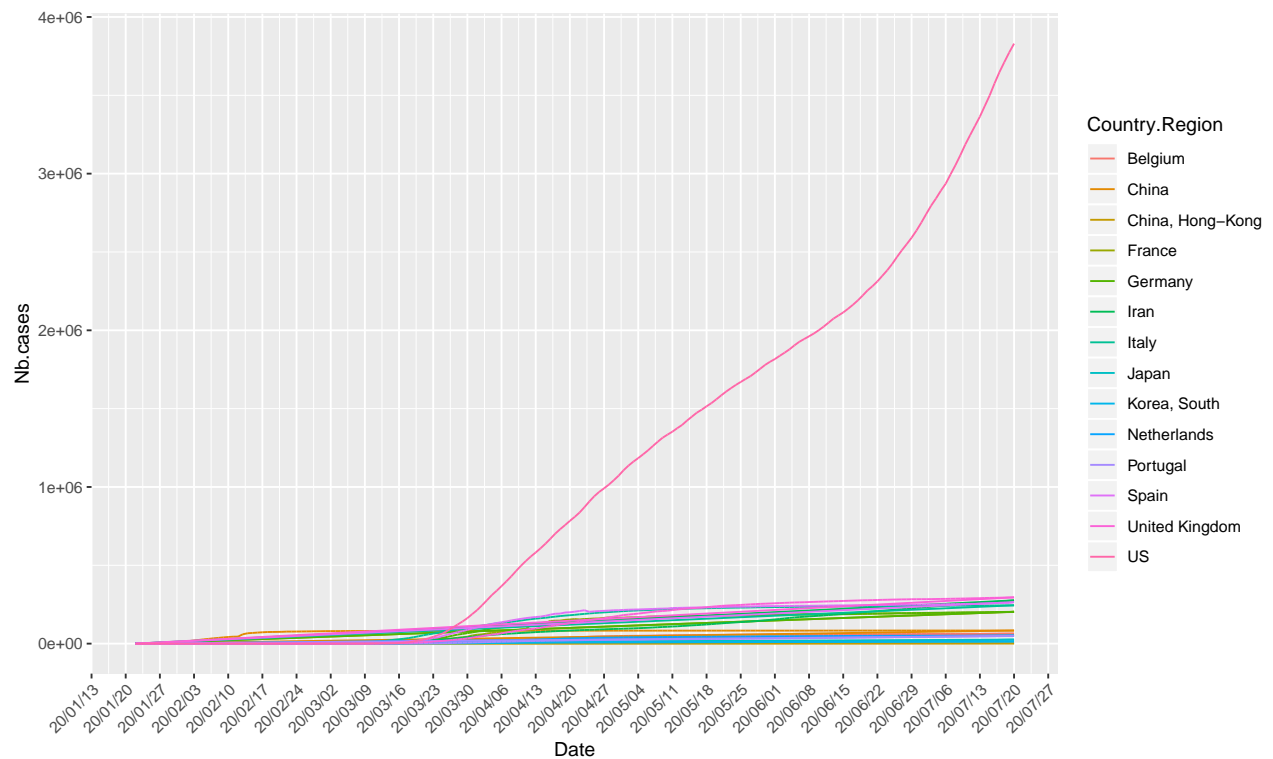
Another plot shows this time the number of cases accumulated over time in all countries (among those chosen at the start of this study)

```
# Cumulative histogram for the number of cases accross selected countries
ggplot(mini.data, aes(x=Date, y=Nb.cases, by= Country.Region)) +
  geom_bar(stat="identity", fill="steelblue") +
  theme_minimal()
```

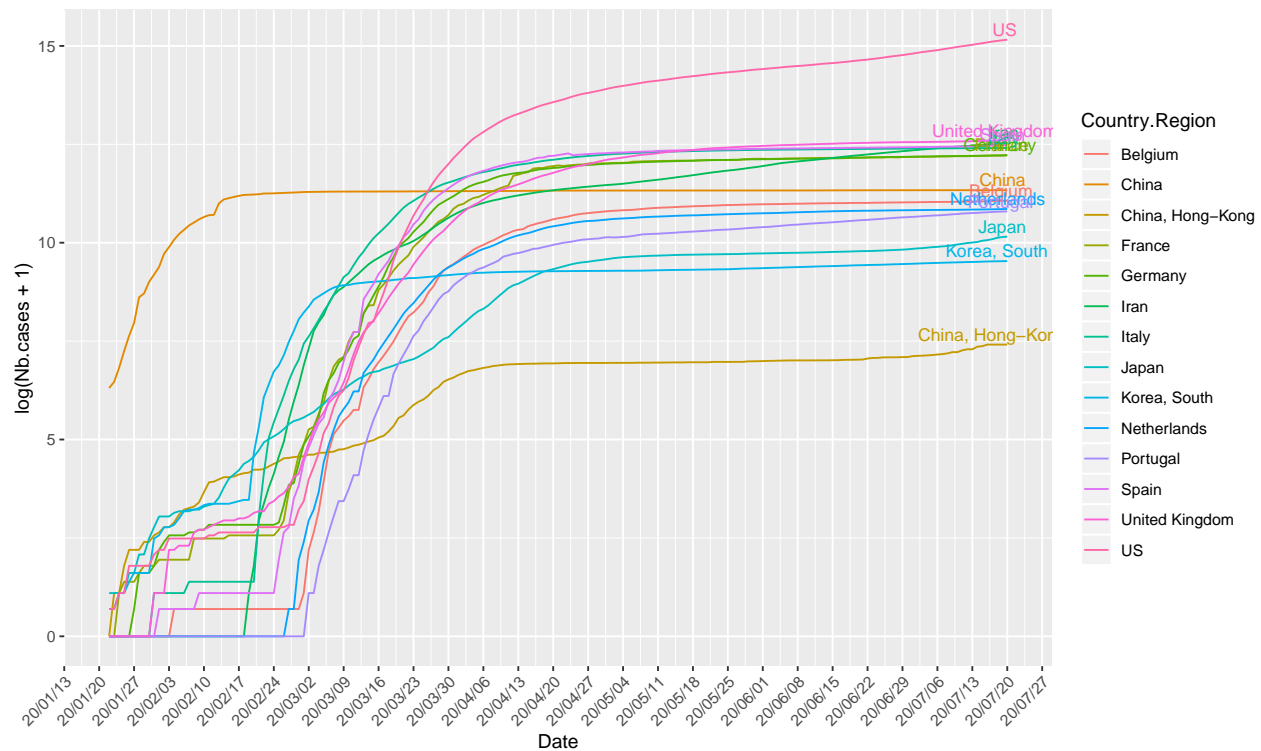


Finally, a graph with the date on the abscissa and the cumulative number of cases on this date on the ordinate. The first graph is made with a linear scale and the second with a logarithmic scale.

```
# Multiple line plot
ggplot(mini.data, aes(x = Date, y = Nb.cases)) +
  geom_line(aes(color = Country.Region), group = 1) +
  scale_x_date(breaks = "week", labels=date_format("%y/%m/%d"))+
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



```
# Multiple line plot with log-scale
ggplot(mini.data, aes(x = Date, y = log(Nb.cases+1))) +
  geom_line(aes(color = Country.Region)) +
  scale_x_date(breaks = "week", labels=date_format("%y/%m/%d"))+
  geom_dl(aes(label = Country.Region, colour=Country.Region),
    method = list(dl.combine("last.points"),
      cex=0.8, rot = 0, vjust=-0.3, hjust=0.6)) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



The logarithmic scale is more suitable for comparing the progression of the disease between countries. We are more able to see the rapid progression or not of the disease.

`sessionInfo()`

```
## R version 3.4.4 (2018-03-15)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 16.04.6 LTS
##
## Matrix products: default
## BLAS: /usr/lib/libblas/libblas.so.3.6.0
## LAPACK: /usr/lib/lapack/liblapack.so.3.6.0
##
## locale:
##  [1] LC_CTYPE=fr_FR.UTF-8      LC_NUMERIC=C
##  [3] LC_TIME=fr_FR.UTF-8      LC_COLLATE=fr_FR.UTF-8
##  [5] LC_MONETARY=fr_FR.UTF-8  LC_MESSAGES=fr_FR.UTF-8
##  [7] LC_PAPER=fr_FR.UTF-8     LC_NAME=C
##  [9] LC_ADDRESS=C             LC_TELEPHONE=C
## [11] LC_MEASUREMENT=fr_FR.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats      graphics  grDevices  utils      datasets  methods   base
##
## other attached packages:
## [1] magrittr_1.5      directlabels_2020.6.17 scales_1.0.0
## [4] ggplot2_3.2.1     tidyr_1.0.0       dplyr_0.8.3
##
## loaded via a namespace (and not attached):
## [1] Rcpp_1.0.1      knitr_1.25      munsell_0.5.0   tidyselect_0.2.5
## [5] colorspace_1.4-1 R6_2.4.0        quadprog_1.5-5  rlang_0.4.1
```

## [9]	stringr_1.4.0	tools_3.4.4	grid_3.4.4	gtable_0.3.0
## [13]	xfun_0.10	withr_2.1.2	ellipsis_0.3.0	htmltools_0.3.6
## [17]	lazyeval_0.2.2	yaml_2.2.0	assertthat_0.2.1	digest_0.6.18
## [21]	tibble_2.1.3	lifecycle_0.1.0	crayon_1.3.4	purrr_0.3.3
## [25]	vctrs_0.2.1	zeallot_0.1.0	glue_1.3.1	evaluate_0.14
## [29]	rmarkdown_2.1	labeling_0.3	stringi_1.4.3	compiler_3.4.4
## [33]	pillar_1.4.3	backports_1.1.3	pkgconfig_2.0.2	