MOOC_COVID_Analysis

VB (feb2301522924f68234e7a552680f397)

2023-05-24

Analyse de l'incidence du syndrôme grippal avec une copie locale des données

Le but est ici de reproduire des graphes semblables à ceux du [South China Morning Post]https://www.scmp.com/ (https://www.scmp.com/) (SCMP), sur la page The Coronavirus Pandemic (https://www.scmp.com/coronavirus?src=homepage_covid_widget) et qui montrent pour différents pays le nombre cumulé (c'est-à-dire le nombre total de cas depuis le début de l'épidémie) de personnes atteintes de la maladie à coronavirus 2019.

Les données que nous utiliserons dans un premier temps sont compilées par le Johns Hopkins University Center for Systems Science and Engineering (JHU CSSE) (https://systems.jhu.edu/) et sont mises à disposition sur GitHub (https://github.com/CSSEGISandData/COVID-19). C'est plus particulièrement sur les données `time_series_covid19_confirmed_global.csv` (des suites chronologiques au format csv) disponibles à l'adresse : https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_time_series/time_series_covid19_confirmed_global.csv (https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_time_series/time_series_covid19_confirmed_global.csv), que nous allons nous concentrer.

Vous commencerez par télécharger les données pour créer un graphe montrant l'évolution du nombre de cas cumulé au cours du temps pour les pays suivants : la Belgique (*Belgium*), la Chine - toutes les provinces sauf Hong-Kong (*China*), Hong Kong (*China, Hong-Kong*), la France métropolitaine (*France*), l'Allemagne (*Germany*), l'Iran (*Iran*), l'Italie (*Italy*), le Japon (*Japan*), la Corée du Sud (*Korea, South*), la Hollande sans les colonies (*Netherlands*), le Portugal (*Portugal*), l'Espagne (*Spain*), le Royaume-Unis sans les colonies (*United Kingdom*), les États-Unis (*US*).

Le nom entre parenthèses est le nom du « pays » tel qu'il apparaît dans le fichier `time_series_covid19_confirmed_global.csv` . Les données de la Chine apparaissent par province et nous avons séparé Hong-Kong, non pour prendre parti dans les différences entre cette province et l'état chinois, mais parce que c'est ainsi qu'apparaissent les données sur le site du SCMP. Les données pour la France, la Hollande et le Royaume-Uni excluent les territoires d'outre-mer et autres « résidus coloniaux ».

Ensuite vous ferez un graphe avec la date en abscisse et le nombre cumulé de cas à cette date en ordonnée. Nous vous proposons de faire deux versions de ce graphe, une avec une échelle linéaire et une avec une échelle logarithmique.

The rest of this RMarkdown file will be in english.

Installing and loading required packages

In this analysis, I will use 3 packages : **tidyverse/stringr** to format the data and **ggplot2/ggrepel** for the graphical representation. The next R lines will detect if the packages are installed, and if not, it should install them.

```
list.of.packages <- c("ggplot2", "tidyverse", "ggrepel","stringr")
new.packages <- list.of.packages[!(list.of.packages %in% installed.packages()[,"Package"])]
if(length(new.packages)) install.packages(new.packages)
```

Downloading and loading the data in R

The next chunk of code will load the data in R. If no local copy of the csv file is present, it will be downloaded.

```
data_url= "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_time_series/time_series_covid19_confirmed_global.csv"

data_file = "time_series_covid19_confirmed_global.csv"
if (!file.exists(data_file)) {
    download.file(data_url, data_file, method="auto")}

data=read.csv("time_series_covid19_confirmed_global.csv",sep=",") #sep= allows the loading of "," separated csv
```

I then check that the loading is correct, since the csv is “,” separated.

```
head(colnames(data)) # Allows us to see that the columns were well separated
```

```
## [1] "Province.State" "Country.Region" "Lat"           "Long"
## [5] "X1.22.20"       "X1.23.20"
```

```
head(data$Country.Region)
```

```
## [1] "Afghanistan" "Albania"     "Algeria"     "Andorra"     "Angola"
## [6] "Antarctica"
```

It is correctly loaded, but the data frame will need some manipulations before I can use it with ggplot, the package I will use for the graphical representation.

Building the data frame to show the number of cases over time

Since I want to exclude all “colonial territories” except China’s regions, I will exclude all of the non-metropolitan territories by **creating a data frame from only the rows containing an empty “Province.State”**. The only purpose of this data frame is to simplify the selection of the countries that I want for the analysis.

```
data_noProvince<-data[(data$Province.State==""),]
```

I will **check that the selection is correct** by comparing the number of rows in the two data frame and verifying that France is only shown once.

```
nrow(data_noProvince)
```

```
## [1] 198
```

```
nrow(data)
```

```
## [1] 289
```

```
head(data_noProvince[data_noProvince$Country.Region=="France",c(1,2)])
```

```
##     Province.State Country.Region
## 132                        France
```

Now that the separation is complete, **I will select the countries I want**. China and Hong Kong are exceptions. I will need to add up the numbers of all the regions of China, except Hong-Kong since it will have it's own category.

```
list_of_countries <- c("France","Belgium","Germany","Iran","Italy","Japan","Netherlands","Por
tugal","Spain","United Kingdom","US","Korea, South")

dt_countries<-data_noProvince[data_noProvince$Country.Region%in%list_of_countries,] # Here, I
use the prepared data frame without the non-metropolitan territories

china_data<-data[data$Country.Region=="China"&data$Province.State!="Hong Kong",] #I get all t
he numbers associated with China, except Hong-Kong

hong_kong<-data[data$Country.Region=="China"&data$Province.State=="Hong Kong",]
hong_kong$Country.Region<-"Hong Kong" #I isolated the numbers associated with Hong-Kong
```

For China, I will add up the infected-per-day numbers, which will give me a single row for the final data frame.

```
china<-cbind(china_data[1,c(1,2)],t(data.frame(list(colSums(china_data[,-c(1,2)])))))

#cbind() allows the fusion of the columns of multiple data frame with a same number of rows.

#t() is used to transpose the axes of the data frame, for formatting purpose for tidyverse. c
olSums provides the sums of all the columns
```

Finally, I will **assemble the 3 data frames** (China, Hong-Kong, and the other countries). I will put the country names as row_names, for formatting purpose.

```
dt_countries<-rbind(china, hong_kong,dt_countries) #rbind is similar to cbind, but to fuse ro
ws
colnames(dt_countries)<-colnames(data) # to conserve original column names
rownames(dt_countries)<-dt_countries$Country.Region
```

To allow the graphical representation, **I will remove all the unnecessary variables** (State, Latitude and Longitude). **I transpose the data frame so that countries are now the variables and the date are discriminating values**. Again, it is mostly for formatting.

```
dt_countries_onlydata<-as.data.frame(t(dt_countries[,c(5:ncol(dt_countries))]))# remove unnec
essary variables

dt_countries_onlydata$date<-rownames(dt_countries_onlydata) # adding dates as variable
```

Using tidyverse, I will **prepare the data for processing by ggplot**.

To be more precise, it will associate a key (here, countries) to each value (here, cases). Values and keys are ordered with the “date” variable. When ggplot will plot the data, it will plot the values to lines specified by their keys i.e. plot the number of cases from France to a line “France”. If it is not done this way, ggplot will not understand that I want to plot multiple lines and the output is unreadable.

```
library(tidyverse,quietly=TRUE)
```

```
## Warning: le package 'tidyverse' a été compilé avec la version R 4.2.3
```

```
## Warning: le package 'ggplot2' a été compilé avec la version R 4.2.3
```

```
## Warning: le package 'tibble' a été compilé avec la version R 4.2.3
```

```
## Warning: le package 'tidyr' a été compilé avec la version R 4.2.3
```

```
## Warning: le package 'readr' a été compilé avec la version R 4.2.3
```

```
## Warning: le package 'purrr' a été compilé avec la version R 4.2.3
```

```
## Warning: le package 'dplyr' a été compilé avec la version R 4.2.3
```

```
## Warning: le package 'forcats' a été compilé avec la version R 4.2.3
```

```
## Warning: le package 'lubridate' a été compilé avec la version R 4.2.3
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.2     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to be
come errors
```

```
df<-dt_countries_onlydata %>%
  select(colnames(dt_countries_onlydata)) %>%
  gather(key="Country",value="Cases",-date)
```

We can compare the two data frames to see the different layouts.

```
head(dt_countries_onlydata)
```

```
##          China Hong Kong Belgium France Germany Iran Italy Japan Korea, South
## X1.22.20   548         0       0      0       0    0     0     2            1
## X1.23.20   641         2       0      0       0    0     0     2            1
## X1.24.20   918         2       0      2       0    0     0     2            2
## X1.25.20  1401         5       0      3       0    0     0     2            2
## X1.26.20  2067         8       0      3       0    0     0     4            3
## X1.27.20  2869         8       0      3       1    0     0     4            4
##          Netherlands Portugal Spain US United Kingdom     date
## X1.22.20           0        0     0  1              0 X1.22.20
## X1.23.20           0        0     0  1              0 X1.23.20
## X1.24.20           0        0     0  2              0 X1.24.20
## X1.25.20           0        0     0  2              0 X1.25.20
## X1.26.20           0        0     0  5              0 X1.26.20
## X1.27.20           0        0     0  5              0 X1.27.20
```

```
head(df)
```

```
##       date Country Cases
## 1 X1.22.20   China   548
## 2 X1.23.20   China   641
## 3 X1.24.20   China   918
## 4 X1.25.20   China  1401
## 5 X1.26.20   China  2067
## 6 X1.27.20   China  2869
```

I will finally convert the obscure date format to one that R can recognize

```
library(stringr,quietly=TRUE)
df$date<-as.Date(str_sub(df$date,2,-1),format="%m.%d.%y")# str_sub is used to remove the "X" before the date
```

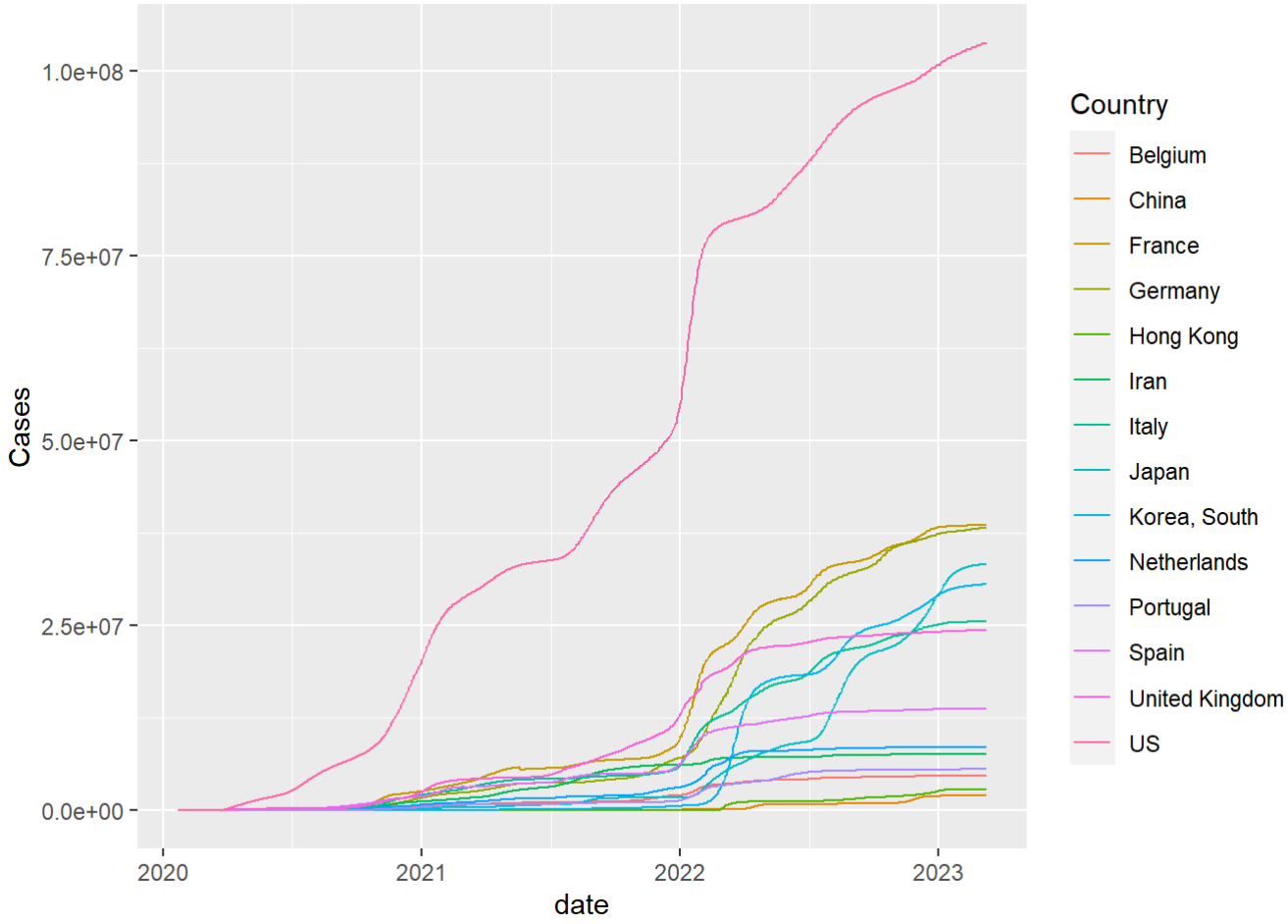
Now, the data are ready to be drawn by ggplot

Graphical representation

```
library(ggplot2,quietly=TRUE)
library(ggrepel,quietly=TRUE)#package used to ad the label shown below
```

```
## Warning: le package 'ggrepel' a été compilé avec la version R 4.2.3
```

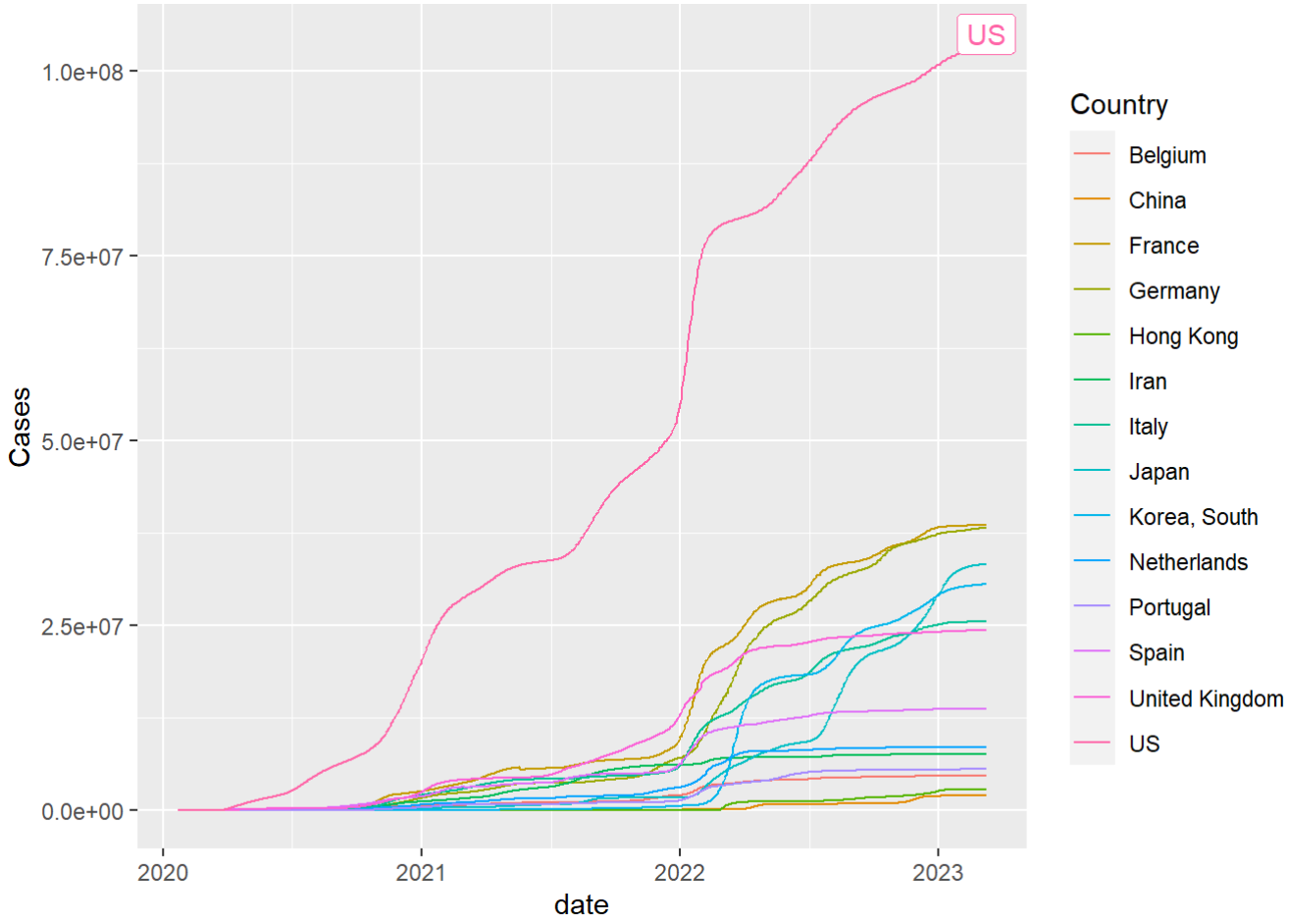
```
ggplot(df, aes(x = date, y = Cases)) +
  geom_line(aes(color = Country, group = Country))
```



I will add a label to identify the most infected country

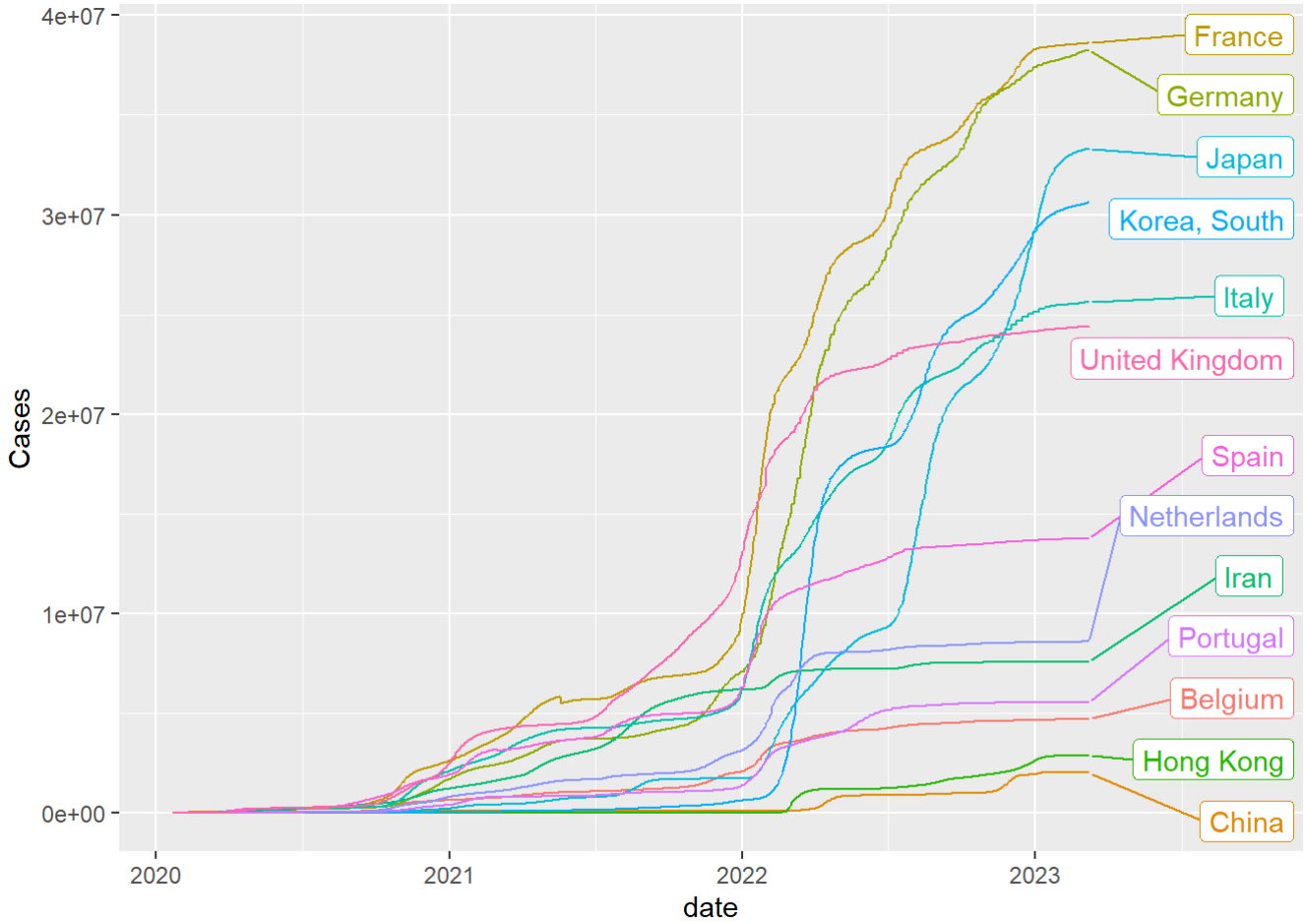
```
label = if_else(df$date == max(df$date), as.character(df$Country), NA_character_)
  ggplot(df, aes(x = date, y = Cases)) +
  geom_line(aes(color = Country, group = Country))+
  geom_label_repel(aes(label = label,color=Country),
                   nudge_x = 0, nudge_y=0, max.overlaps=1, direction = "y",
                   na.rm = TRUE,show.legend=F)
```

```
## Warning: ggrepel: 13 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



It is not easy to see the other countries. ggrepel indicates that there is no room to put the labels of the other countries. Let's mask the US and see how it looks like now

```
zoomed_df<-df[df$Country!="US",]

label = if_else(zoomed_df$date == max(zoomed_df$date), as.character(zoomed_df$Country), NA_ch
aracter_)
  ggplot(zoomed_df, aes(x = date, y = Cases)) +
  geom_line(aes(color = Country, group = Country))+
  geom_label_repel(aes(label = label,color=Country),
                   nudge_x = 200, nudge_y=50, max.overlaps=20, direction = "y", force=10,
                   na.rm = TRUE,show.legend=F)+ theme(legend.position = "none")
```



Something is off about the number of cases from China. WHO organization indicates a cumulative cases number of nearly 100 millions (source (https://covid19.who.int/region/wpro/country/cn)). Let's check the total number of cases in the world and then I'll come back to it.

Total number of case over time

I will use the original data frame imported from the website. First, let's change its name.

```
dtpart2<-data
```

I need to observe the total number of cases over time, so I will **add up all the columns** to give me the sum of all cases at a given day.

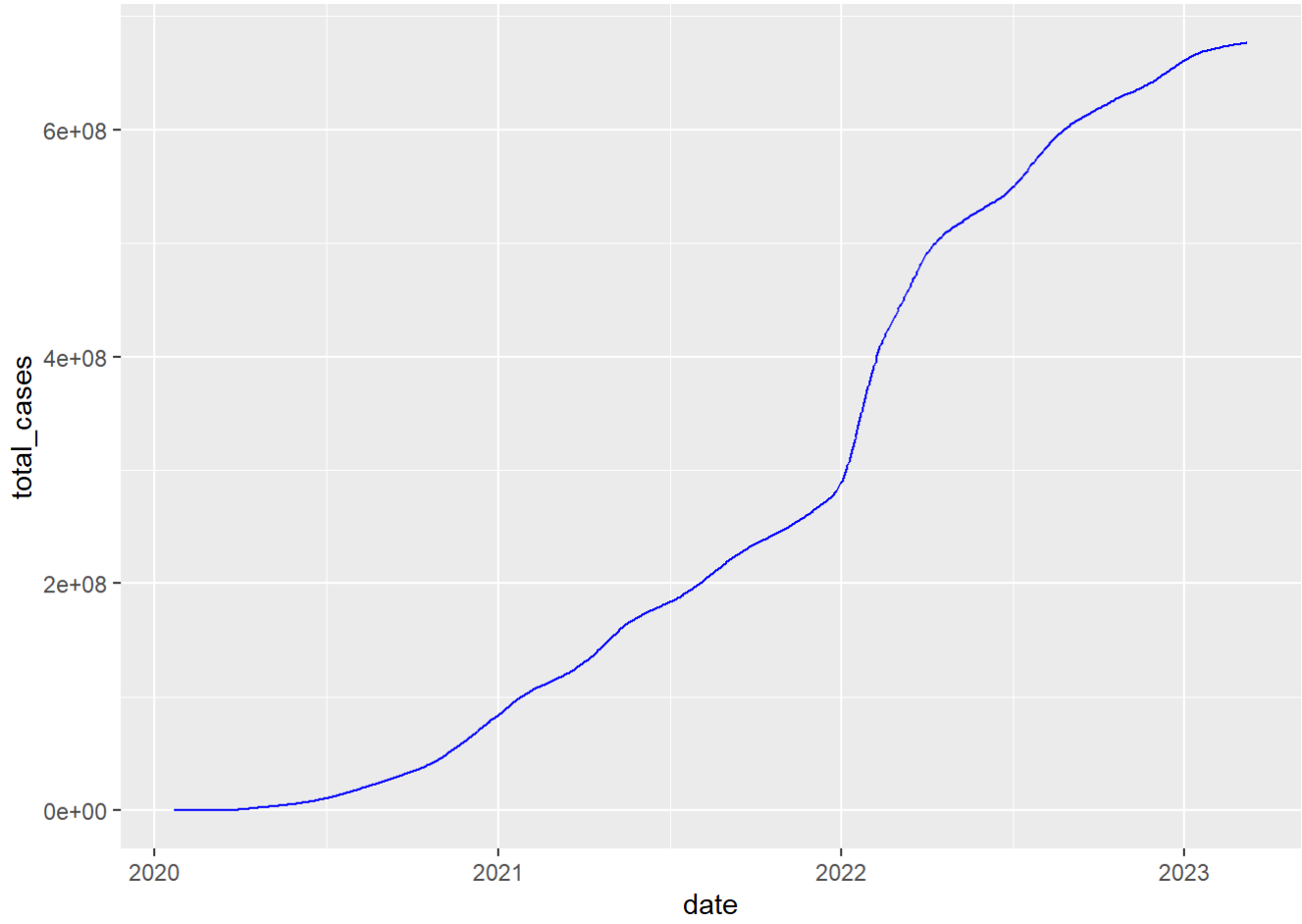
```
dt_total_cases<-colSums(dtpart2[,-c(1:4)])
```

Now, I will produce the data frame to give to ggplot I also convert the date to a better format, using the same code than above

```
df2<-data.frame(date=names(dt_total_cases),total_cases=dt_total_cases)
df2$date<-as.Date(str_sub(df2$date,2,-1),format="%m.%d.%y")
```

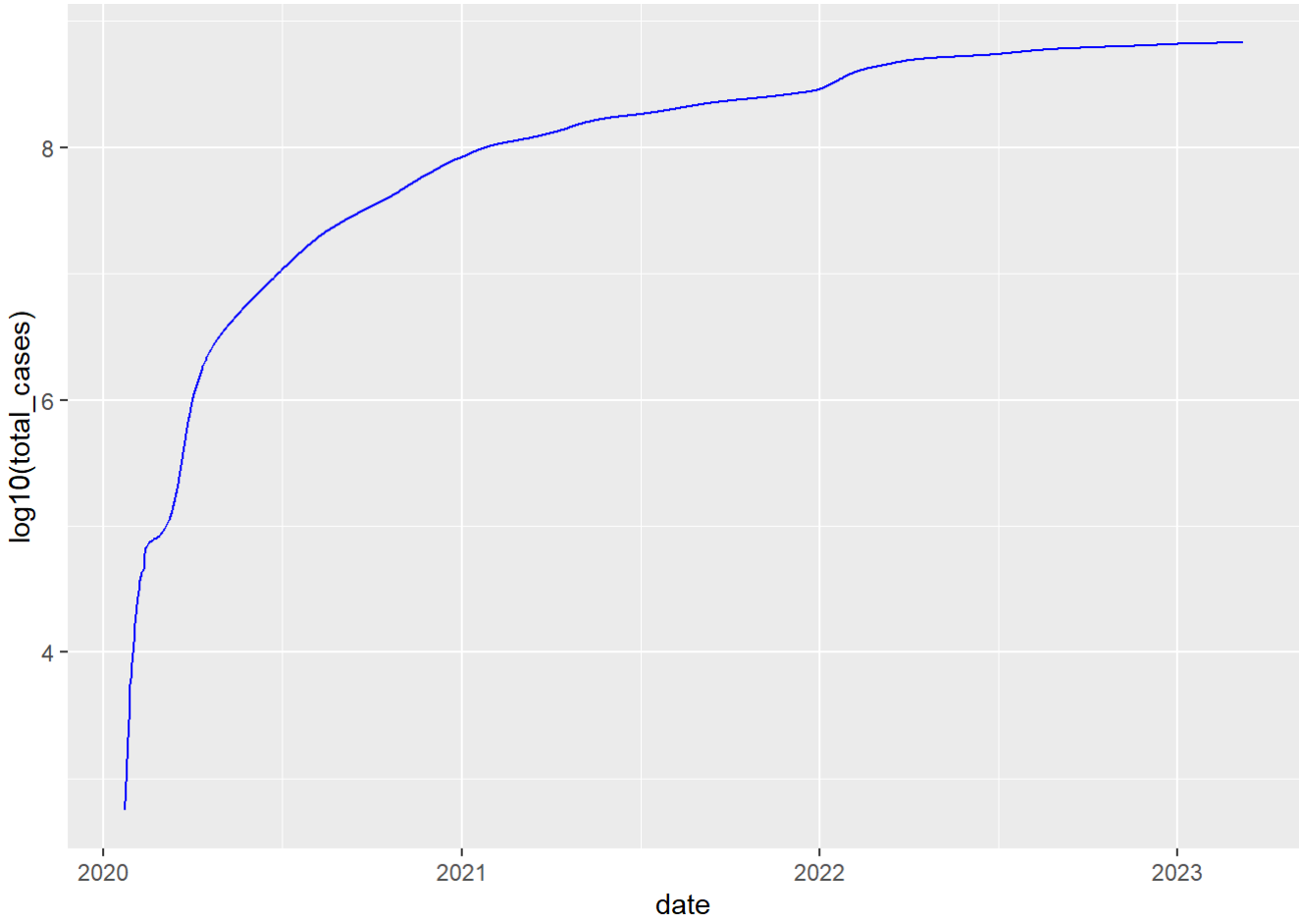
Let's do the graphical representation

```
ggplot(df2, aes(x = date, y = total_cases)) +
  geom_line(color="blue")
```



Let’s see how it looks after a logarithmic transformation

```
ggplot(df2, aes(x = date, y = log10(total_cases))) +
  geom_line(color="blue")
```



I previously suspected there was a problem with China's data. I first checked my analysis by comparing what I obtained with the map drawn by Hohns Hopkins University (https://coronavirus.jhu.edu/map.html).

They obtained the same total cases number than me, but it seems that they did not use China's data.

I compared their results with WHO data set visible here (https://covid19.who.int/?mapFilter=cases) and they are indeed quite different.

I will perform the analysis again but using WHO data set obtainable here (https://covid19.who.int/WHO-COVID-19-global-table-data.csv)

Same analysis using WHO data

I will **load the data** using the same code than above.

```
data_url= "https://covid19.who.int/WHO-COVID-19-global-data.csv"

data_file = "WHO-COVID-19-global-data.csv"
if (!file.exists(data_file)) {
    download.file(data_url, data_file, method="auto")}

data=read.csv("WHO-COVID-19-global-data.csv",sep=",") #sep= allows the loading of "," separat
ed csv
```

I will **select the same countries as before**. Sadly, Hong-Kong is not available in the list from WHO

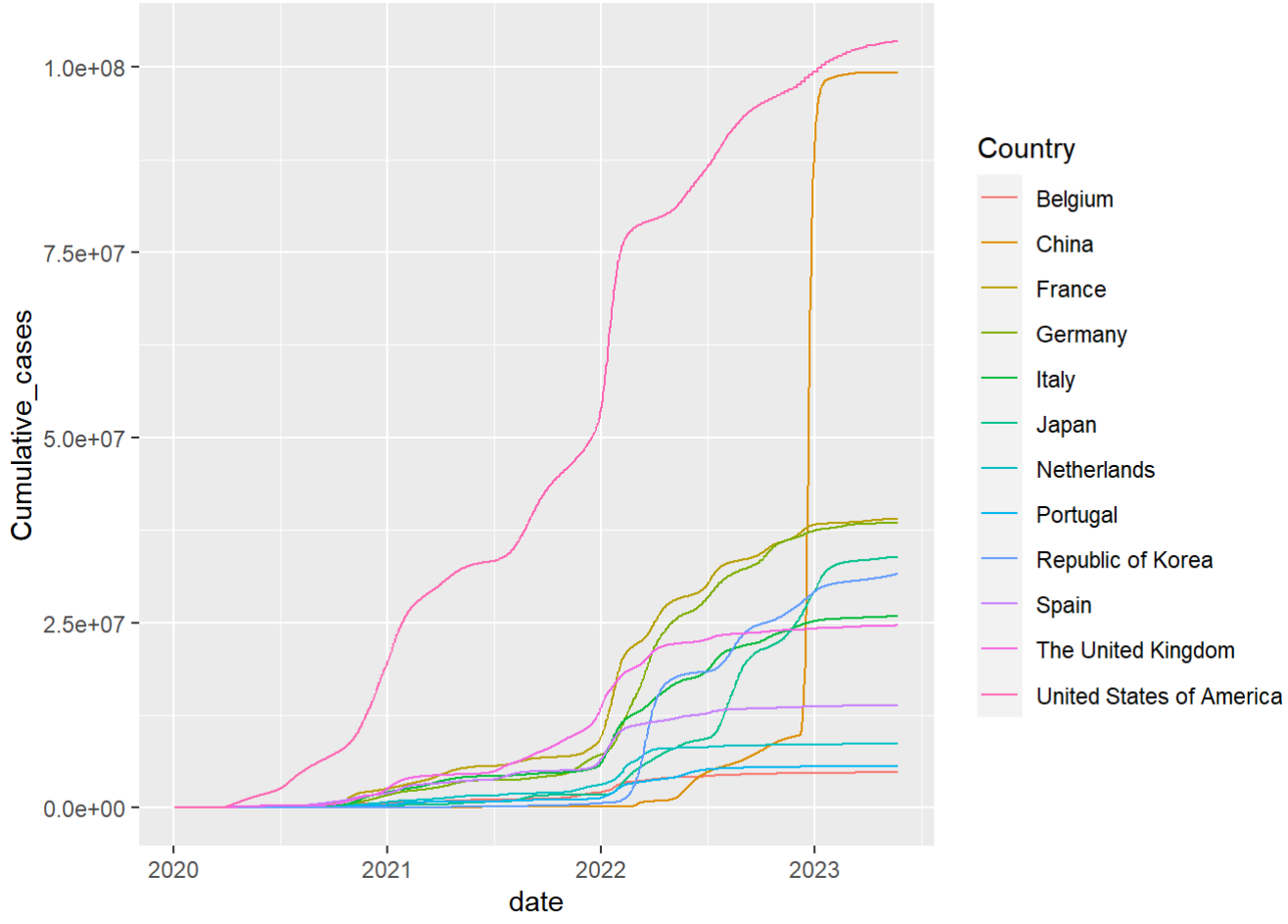
```
list_of_countries <- c("France","Belgium","Germany","Iran","Italy","Japan","Netherlands","Portugal","Spain","The United Kingdom","United States of America","Republic of Korea","China")

dt_countries<-data[data$Country%in%list_of_countries,]
```

I will **format the date** for R to understand

```
dt_countries$date<-as.Date(dt_countries[,1])#the format of the date in the file is already well formatted but was not recognized as such by R
```

This time, the data frame imported from the source is well formatted and is already usable with ggplot2. **Let's plot the data**

```
  ggplot(dt_countries, aes(x = date, y = Cumulative_cases )) +
  geom_line(aes(color = Country, group = Country))
```



It is better. Now let's **calculate the total number of cases**. I will replace all the Country name by "worldwide" so ggplot will group everything for plotting

```
dt_worldwide<-data
dt_worldwide$date<-as.Date(dt_worldwide[,1])
dt_worldwide$Country<-"worldwide"
```

The formatting is a little more complex here. To generate the worldwide cumulative deaths numbers, I will first **generate a data frame with the data ordered by date**.

```
df<-dt_worldwide %>%
  select(date,Cumulative_cases) %>%
  gather(key="date",value="Cumulative_cases")

df<-df[order(df$date),]
```

I will create a function which will return the **worldwide cumulative deaths number of each date**

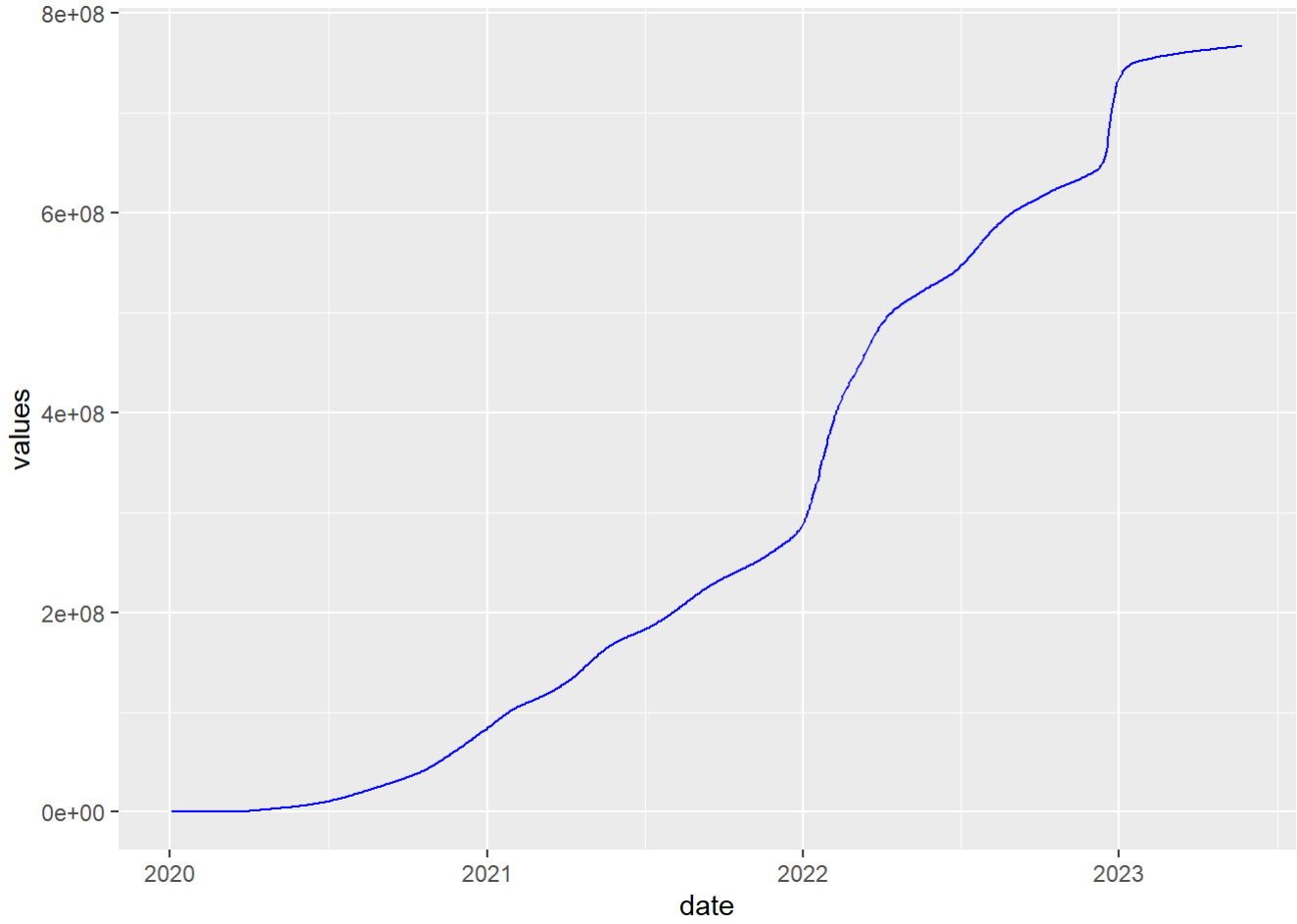
```
collapse_column<-function(x){
    cdf<-df[df$date==x,]
    output<-sum(cdf[,2])
    return(output)}
```

Then, I will **apply this function to each dates** and return the result in a new data frame.

```
df_total_cases<-data.frame(date=unique(df$date),values=unlist(lapply(unique(df$date),collapse
_column)))
```

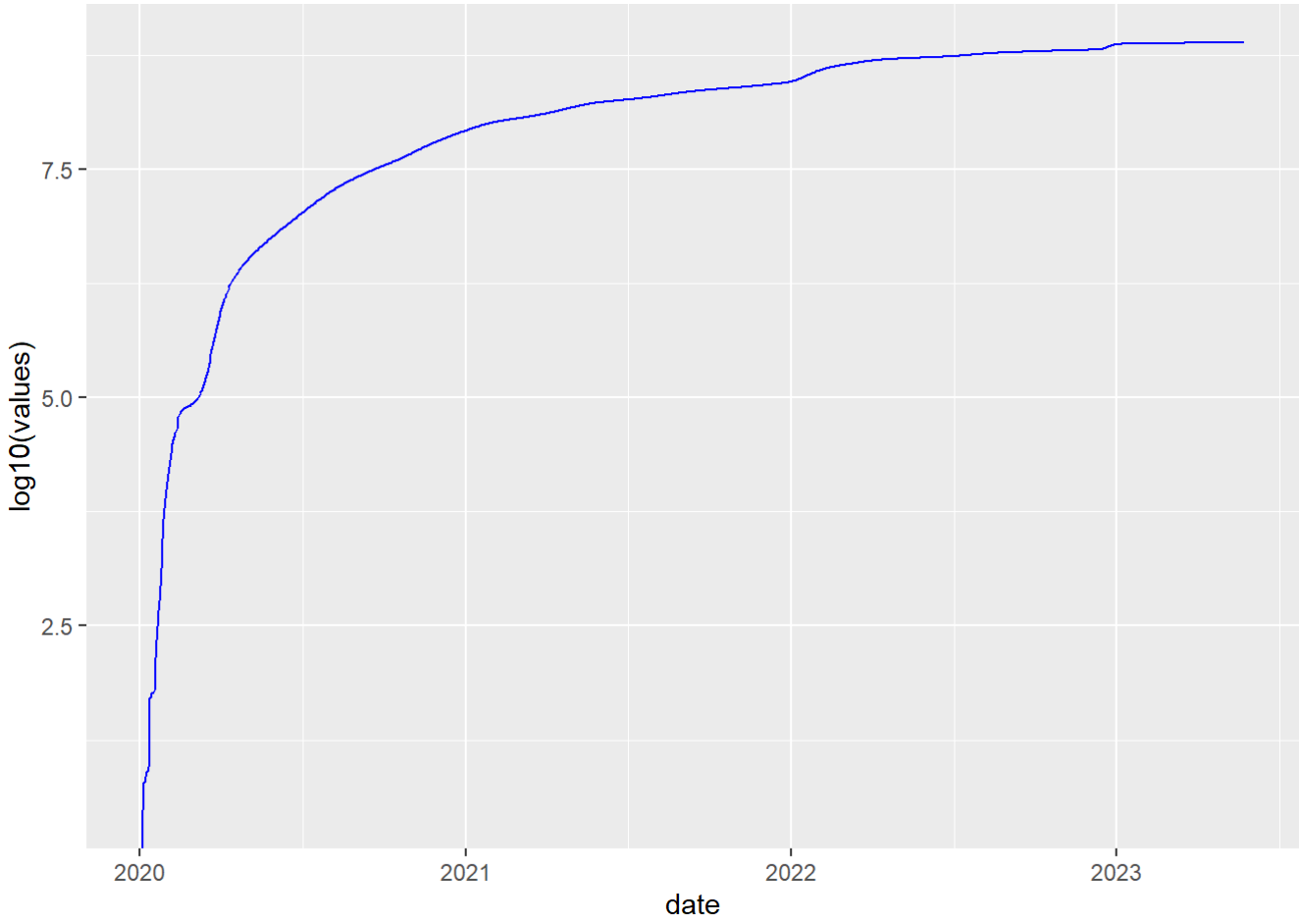
Now let's see the graph

```
  ggplot(df_total_cases, aes(x = date, y = values )) +
  geom_line(color = "blue")
```



Let's see how it looks after a logarithmic transformation

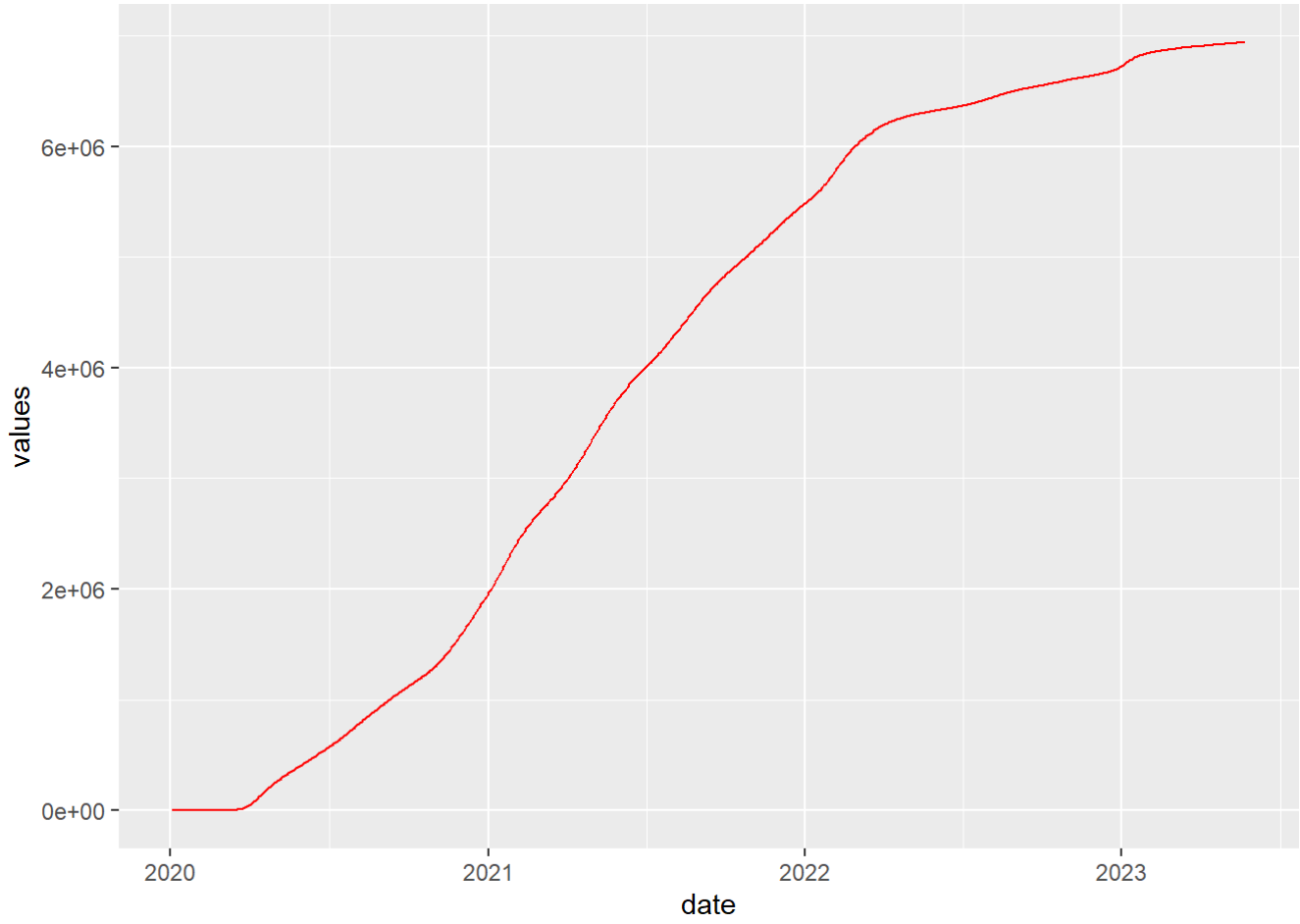
```
ggplot(df_total_cases, aes(x = date, y = log10(values))) +
  geom_line(color="blue")
```



What about the deaths ?

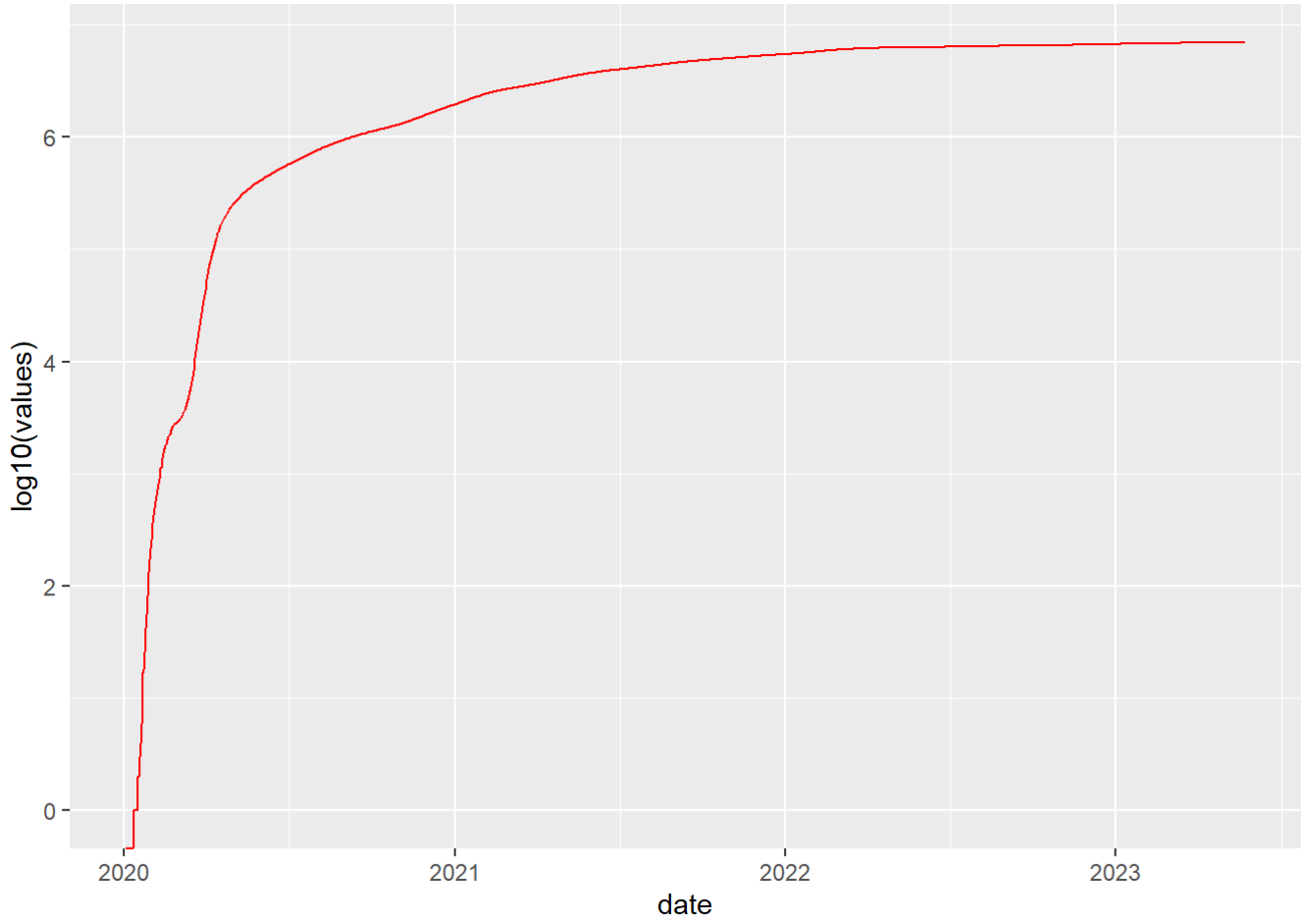
Let's have a look to the deaths with a similar approach.

```
df<-dt_worldwide %>%
  select(date,Cumulative_deaths) %>%
  gather(key="date",value="Cumulative_deaths")

df<-df[order(df$date),]

collapse_column<-function(x){
    cdf<-df[df$date==x,]
    output<-sum(cdf[,2])
    return(output)}

df_total_death<-data.frame(date=unique(df$date),values=unlist(lapply(unique(df$date),collapse
_column)))

  ggplot(df_total_death, aes(x = date, y = values )) +
  geom_line(color = "red")
```



Same graph with logarithmic scale

```
ggplot(df_total_death, aes(x = date, y = log10(values ))) +
geom_line(color = "red")
```



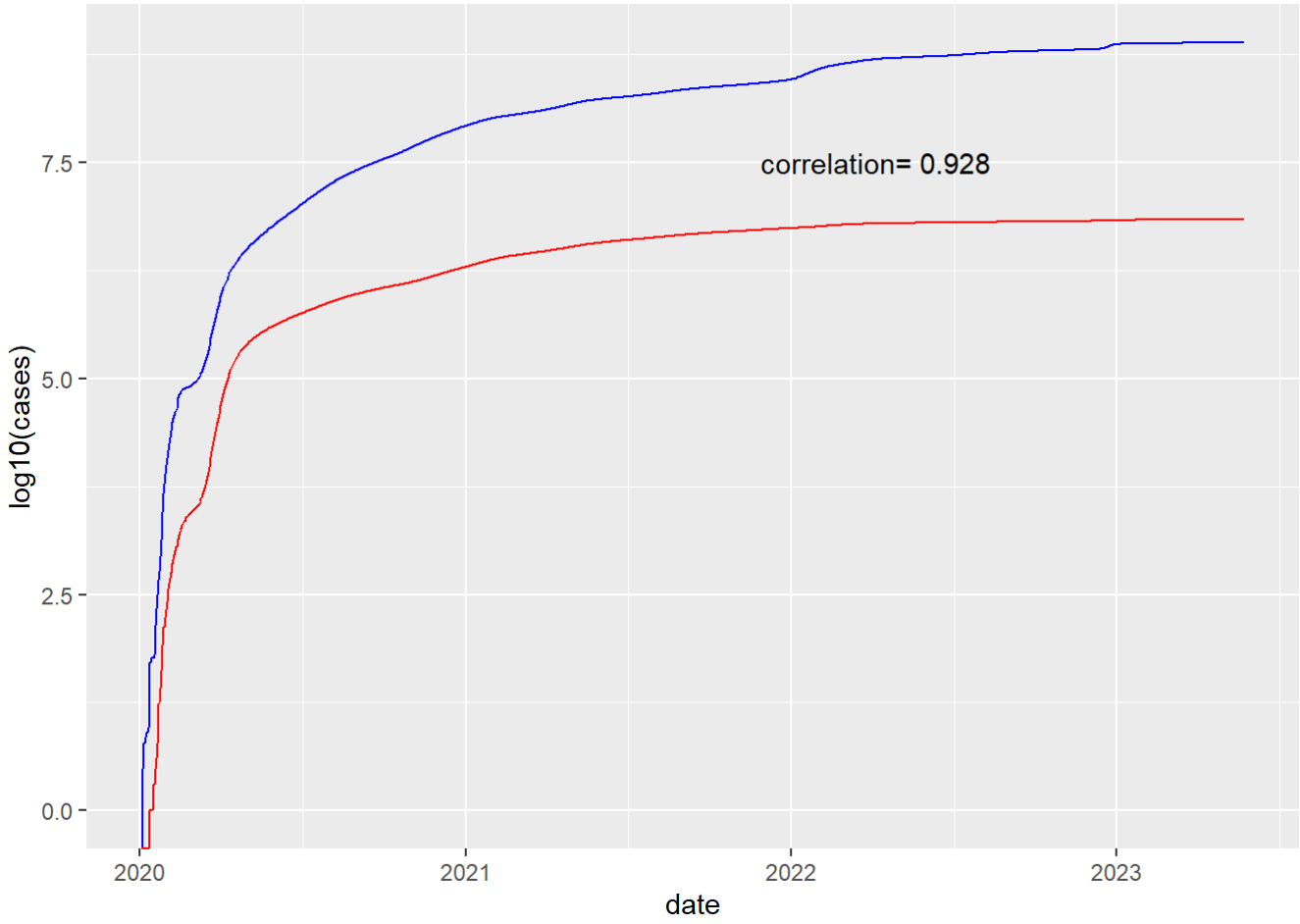
Cases and deaths correlation

Just for the exercise, I would like to see if cases and death numbers are related. First, I will **build the data frame**

```
dt_cases_and_death<-cbind(df_total_cases,df_total_death[,2])
colnames(dt_cases_and_death)<-c("date","cases","deaths")
```

And now, the graph and the correlation. Due to the high number of samples, I **will apply a Pearson correlation** (default with R) to see if deaths and cases numbers are linked.

```
ggplot(dt_cases_and_death, aes(x = date, y = log10(cases) )) +
  geom_line(color = "blue")+
  geom_line(aes(x = date, y = log10(deaths)),color="red")+
  annotate("text", x=dt_cases_and_death$date[nrow(dt_cases_and_death)/1.5], y = 7.5,label=pas
te0("correlation= ",round(cor(dt_cases_and_death$cases,dt_cases_and_death$deaths),3)))
```



Without surprise, **cases and deaths are linked**. It is not very scientifically useful, but since I deviated a little from the original exercise, I will stop here