R Project Covid19

Aurelien Sofia Julien

2023-12-19

Introduction

This project aims to leverage the covid19 package in R, for a comprehensive analysis of COVID-19 data. The package provides a convenient interface for accessing up-to-date information on cases, deaths, and recoveries. The analysis will focus on understanding the temporal trends, geographical distribution, and demographic impact of the pandemic.

Analysis Steps:

Install and Load the covid19 Package:

- Install the covid19 package.
- Load the package into the R environment

```
#install.packages("COVID19")
library("COVID19")
library("tinytex")
library("knitr")
library("rmarkdown")
library("tidyverse")
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse
2.0.0 ──
## ✓ dplyr     1.1.3     ✓ readr     2.1.4
## ✓ forcats   1.0.0     ✓ stringr   1.5.0
## ✓ ggplot2   3.4.3     ✓ tibble    3.2.1
## ✓ lubridate 1.9.3     ✓ tidyr     1.3.0
## ✓ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────
tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all
conflicts to become errors
```

Data Retrieval and Loading:

- Utilize the covid19 package functions to retrieve Italian and region-specific COVID19 data.

- Load the data into R for analysis.

We can find the features of covid19 package and especially how are stored the data here Package 'COVID19'. Besides, for more information on the way variables are encode we can find information in the Documentation.

On the columns with have access information like:

- confirmed
- death
- recovered
- tests
- vaccines
- hosp
- population
- much more features for each country or region in a country.

We stored the data from Italy.

```
covid19_ITA = covid19(country='ITA',verbose = FALSE) # verbose = False to
avoid the printing of a message from the authors.
covid19_ITA
```

We select the variables that we want. Here we want "confirmed", "deaths" and "recovered".

```
sample_covid19_ITA =
select(covid19_ITA,date,confirmed,deaths,recovered,population)
sample_covid19_ITA
```

We also stored the data from all region in Italy.

```
covid19_ITA_region = covid19(country = 'ITA',level=2 , verbose = FALSE)
```

Same as above we select only the variable that we want.

```
sample_covid19_ITA_region =
select(covid19_ITA_region,date,confirmed,deaths,recovered,administrative_area
_level_2,population)
sample_covid19_ITA_region
```

Data Cleaning and Pre-processing:

- Address missing values and ensure data consistency
- Format dates and variables for analysis.

Before analysis we have to check if there are missing values (NA) in our data frame.

```
sum(is.na(sample_covid19_ITA)) # we sum the number of NA on the whole data
frame

## [1] 0
```

We can do the same thing on the region data frame.

```
sum(is.na(sample_covid19_ITA_region))
```

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

Great the same results as above.

It's great, we can also verify if the column "date" is in the good type.

```
class(sample_covid19_ITA$date)
```

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

Up to now our idea is to transform the cumulative variables to variables with number on date because it's clearly more useful for analysis especially if we want some descriptive statistics.

We start by concatenate the first value for each column("confirmed","deaths","recovered") and the value index (n+1) subtract by the value index (n) to pass from cumulative number to number per day.

We can store the dimension :

```
dim_covid19_ITA = dim(sample_covid19_ITA) # to store the dimension of the
data frame
dim_covid19_ITA
```

```
## [1] 1417    5
```

```
confirmed_on_date =
c(sample_covid19_ITA$confirmed[1],sample_covid19_ITA$confirmed[2:dim_covid19_
ITA[1]]-sample_covid19_ITA$confirmed[1:dim_covid19_ITA[1]-1])
####
deaths_on_date =
c(sample_covid19_ITA$deaths[1],sample_covid19_ITA$deaths[2:dim_covid19_ITA[1]
]-sample_covid19_ITA$deaths[1:dim_covid19_ITA[1]-1])
####
recovered_on_date =
c(sample_covid19_ITA$recovered[1],sample_covid19_ITA$recovered[2:dim_covid19_
ITA[1]]-sample_covid19_ITA$recovered[1:dim_covid19_ITA[1]-1])
```

We create a new data frame for each columns.

```
confirmed_on_date = data.frame(confirmed_on_date)
deaths_on_date = data.frame(deaths_on_date)
recovered_on_date = data.frame(recovered_on_date)
```

We use cbind() to transform our data frame and add columns that we want.

```
sample_covid19_ITA =
cbind(sample_covid19_ITA,c(confirmed_on_date,deaths_on_date,recovered_on_date
))
```

We make the choice to conserve only relevant variables which means the variables that we have just created.

```
sample_covid19_ITA = select(sample_covid19_ITA,-(confirmed:recovered))
```

Perfect !! Now just check if there are errors in our variables, we should have only positive value or equal to 0 for “confirmed_per_day”,“deaths_per_day” and “recovered_per_day”.

```
filter(sample_covid19_ITA,(confirmed_on_date<0 | deaths_on_date<0
|recovered_on_date<0))
```

```
##         date population confirmed_on_date deaths_on_date recovered_on_date
## 1 2020-06-19   60421760              -148             47              1363
## 2 2020-06-24   60421760               577            -31              1526
## 3 2023-07-01   60421760               562             10              -302
## 4 2024-01-05   60421760              4054            -40              7454
```

We can see that we have three negative values which doesn’t make sense so we can remove these rows for the analysis.

```
sample_covid19_ITA = filter(sample_covid19_ITA,(confirmed_on_date>=0 &
deaths_on_date>=0 & recovered_on_date>=0))
```

Now we can apply the same logic to the data frame with regions :

We can store the number of regions and also see the name of the different regions :

```
numbers_regions =
n_distinct(sample_covid19_ITA_region$administrative_area_level_2)
unique(sample_covid19_ITA_region$administrative_area_level_2)
```

```
##  [1] "Friuli Venezia Giulia" "P.A. Bolzano"          "Molise"
##  [4] "Campania"              "Veneto"                "Basilicata"
##  [7] "Lazio"                 "Lombardia"             "P.A. Trento"
## [10] "Piemonte"              "Valle d'Aosta"         "Sicilia"
## [13] "Marche"                "Calabria"              "Liguria"
## [16] "Umbria"                "Emilia-Romagna"        "Abruzzo"
## [19] "Puglia"                "Toscana"               "Sardegna"
```

We apply a for loop to create iterively a new data frame which contains relevant variables (the same processing than above for each region) .

```
sample_covid19_ITA_region_final = data.frame()

for (region in
unique(sample_covid19_ITA_region$administrative_area_level_2)){

  df = filter(sample_covid19_ITA_region,administrative_area_level_2==region)

  confirmed_on_date_reg =
c(df$confirmed[1],df$confirmed[2:dim_covid19_ITA[1]]-
df$confirmed[1:dim_covid19_ITA[1]-1])
```

```
  deaths_on_date_reg = c(df$deaths[1],df$deaths[2:dim_covid19_ITA[1]]-
df$deaths[1:dim_covid19_ITA[1]-1])

  recovered_on_date_reg=
c(df$recovered[1],df$recovered[2:dim_covid19_ITA[1]]-
df$recovered[1:dim_covid19_ITA[1]-1])

  confirmed_on_date_reg = data.frame(confirmed_on_date_reg)
  deaths_on_date_reg = data.frame(deaths_on_date_reg)
  recovered_on_date_reg = data.frame(recovered_on_date_reg)


df=cbind(df,c(confirmed_on_date_reg,deaths_on_date_reg,recovered_on_date_reg)
)

  sample_covid19_ITA_region_final=rbind(sample_covid19_ITA_region_final,df)

}
sample_covid19_ITA_region_final = select(sample_covid19_ITA_region_final,-
(confirmed:recovered))
```

Like above we should have only positive value or equal to 0 for "confirmed_per_day_reg","deaths_per_day_reg" and "recovered_per_day_reg".

```
filter(sample_covid19_ITA_region_final,(confirmed_on_date_reg<0 |
deaths_on_date_reg<0 |recovered_on_date_reg<0))
```

We can see that we have many negative values as above we can remove these rows for the analysis.

```
sample_covid19_ITA_region_final =
filter(sample_covid19_ITA_region_final,(confirmed_on_date_reg>=0 &
deaths_on_date_reg>=0 & recovered_on_date_reg>=0))
```

Descriptive Statistics:

- Calculate and visualize key summary statistics for global and regional COVID-19 data.
- Explore the distribution of cases, deaths, and recoveries.

We can import "ggplot2" library to visualize the covid19 data.

```
library("ggplot2")
library("gridExtra")

##
## Attachement du package : 'gridExtra'
```

```
## L'objet suivant est masqué depuis 'package:dplyr':
##
##     combine
```

```
"We use summarize() method to compute different statistics for each columns"
```

```
## [1] "We use summarize() method to compute different statistics for each
columns"
```

```
confirmed_stats=summarise(sample_covid19_ITA,min=min(confirmed_on_date),q1=qu
antile(confirmed_on_date,c(0.25)),mean=mean(confirmed_on_date),median=median(
confirmed_on_date),sd=sd(confirmed_on_date),mad=mad(confirmed_on_date),q3=qua
ntile(confirmed_on_date,c(0.75)),IQR=IQR(confirmed_on_date),max=max(confirmed
_on_date))
######
deaths_stats=summarise(sample_covid19_ITA,min=min(deaths_on_date),q1=quantile
(deaths_on_date,c(0.25)),mean=mean(deaths_on_date),median=median(deaths_on_da
te),sd=sd(deaths_on_date),mad=mad(deaths_on_date),q3=quantile(deaths_on_date,
c(0.75)),IQR=IQR(deaths_on_date),max=max(deaths_on_date))
######
recovered_stats=summarise(sample_covid19_ITA,min=min(recovered_on_date),q1=qu
antile(recovered_on_date,c(0.25)),mean=mean(recovered_on_date),median=median(
recovered_on_date),sd=sd(recovered_on_date),mad=mad(recovered_on_date),q3=qua
ntile(recovered_on_date,c(0.75)),IQR=IQR(recovered_on_date),max=max(recovered
_on_date))
```

We combine this three data frame into one using rbind()

```
descriptive_stats_ITA=rbind(confirmed_stats,deaths_stats,recovered_stats)
```

We change the rows names

```
rownames(descriptive_stats_ITA) =
c("confirmed_ITA","deaths_ITA","recovered_ITA")
```

We can display global statistics for covid19 in Italy.

```
descriptive_stats_ITA
```

```
##                min   q1       mean median         sd       mad    q3   IQR
## confirmed_ITA  78 2194 18886.9115   6652 30877.9141 8692.4838 22360 20166
## deaths_ITA      1   24   138.3638     63   177.8264   72.6474   164   140
## recovered_ITA   0 2155 18598.9186   6402 29125.3429 8582.7714 22075 19920
##                  max
## confirmed_ITA 228123
## deaths_ITA       993
## recovered_ITA 248971
```

Using the previous data frame we can have some information about the impact of covid19 in Italy. Especially we have a mean of more 19000 confirmed cases per day and near 140 deaths in mean per day.

We can compute the lethality it means the total number of deaths over the total number of cases.

```
lethality_ITA=(sum(sample_covid19_ITA$deaths_on_date)/sum(sample_covid19_ITA$
confirmed_on_date))*100
lethality_ITA
```

```
## [1] 0.7325907
```

This means that a person in Italy who contract covid19 has approximately 0.75% to die.

We can also compute the same statistics for each regions in Italy using group_by() method. We create a data frame which contains the statistics values for each regions.

```
by=group_by(sample_covid19_ITA_region_final,administrative_area_level_2,popul
ation)

"We use summarize() method to compute different statistics for each columns"
```

```
## [1] "We use summarize() method to compute different statistics for each
columns"
```

```
confirmed_stats_reg=summarise(by,min=min(confirmed_on_date_reg),q1=quantile(c
onfirmed_on_date_reg,c(0.25)),mean=mean(confirmed_on_date_reg),median=median(
confirmed_on_date_reg),sd=sd(confirmed_on_date_reg),mad=mad(confirmed_on_date
_reg),q3=quantile(confirmed_on_date_reg,c(0.75)),IQR=IQR(confirmed_on_date_re
g),max=max(confirmed_on_date_reg),.groups = )
```

```
## `summarise()` has grouped output by 'administrative_area_level_2'. You can
## override using the `.groups` argument.
```

```
######
deaths_stats_reg=summarise(by,min=min(deaths_on_date_reg),q1=quantile(deaths_
on_date_reg,c(0.25)),mean=mean(deaths_on_date_reg),median=median(deaths_on_da
te_reg),sd=sd(deaths_on_date_reg),mad=mad(deaths_on_date_reg),q3=quantile(dea
ths_on_date_reg,c(0.75)),IQR=IQR(deaths_on_date_reg),max=max(deaths_on_date_r
eg))
```

```
## `summarise()` has grouped output by 'administrative_area_level_2'. You can
## override using the `.groups` argument.
```

```
######
recovered_stats_reg=summarise(by,min=min(recovered_on_date_reg),q1=quantile(r
ecovered_on_date_reg,c(0.25)),mean=mean(recovered_on_date_reg),median=median(
recovered_on_date_reg),sd=sd(recovered_on_date_reg),mad=mad(recovered_on_date
_reg),q3=quantile(recovered_on_date_reg,c(0.75)),IQR=IQR(recovered_on_date_re
g),max=max(recovered_on_date_reg))
```

```
## `summarise()` has grouped output by 'administrative_area_level_2'. You can
## override using the `.groups` argument.
```

```
descriptive_stats_ITA_reg=rbind(confirmed_stats_reg,deaths_stats_reg,recovere
d_stats_reg)
```

We create a new column containing the names of the various variables involved.

```
type =
data.frame(type=c(rep("confirmed",numbers_regions),rep("deaths",numbers_regio
ns),rep("recovered",numbers_regions)))

descriptive_stats_ITA_reg = cbind(descriptive_stats_ITA_reg,type)
```

As above we have some information about the impact of covid19 for each region

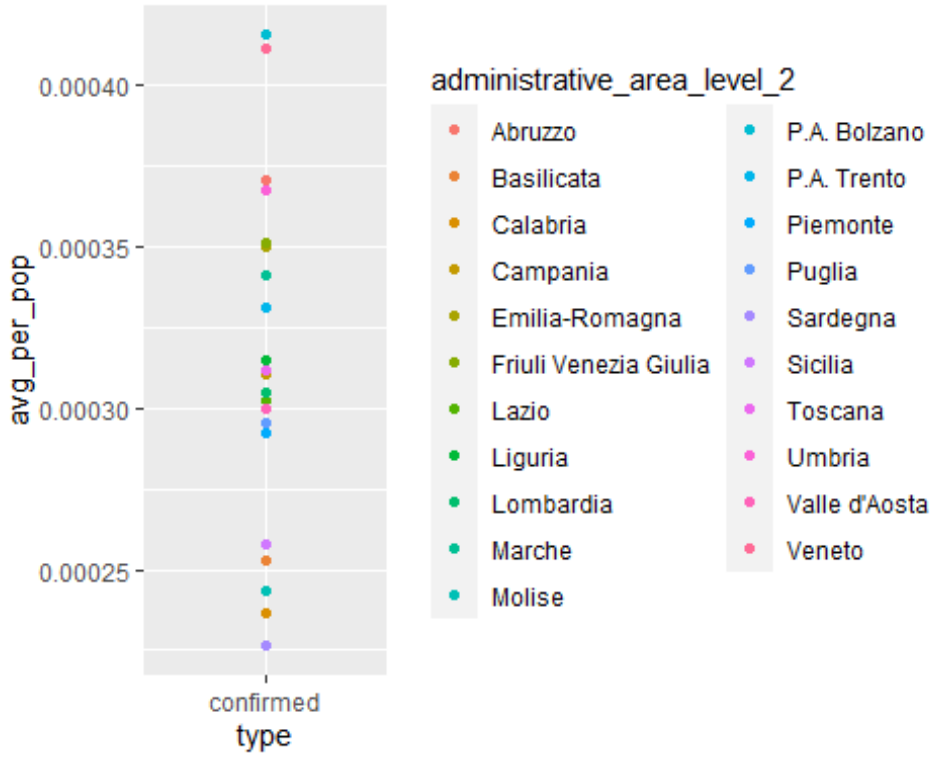
```
descriptive_stats_ITA_reg
```

We can add the mean by population to see the demographic impact of the pandemic and have a more precise idea because there are a wide disparity among regions (In Lombardi we have more than 10 billions people while in Valle d'Aosta we have just 100 K).

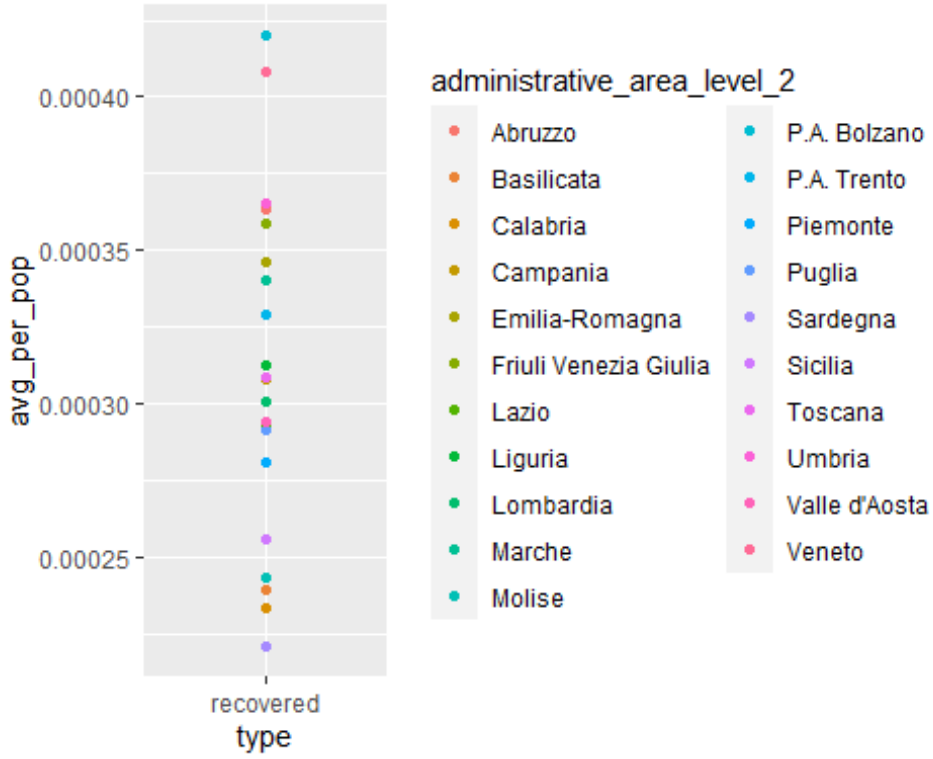
```
descriptive_stats_ITA_reg=mutate(descriptive_stats_ITA_reg,avg_per_pop=mean/p
opulation)
```

We can display the average by population for each regions/characteristics.

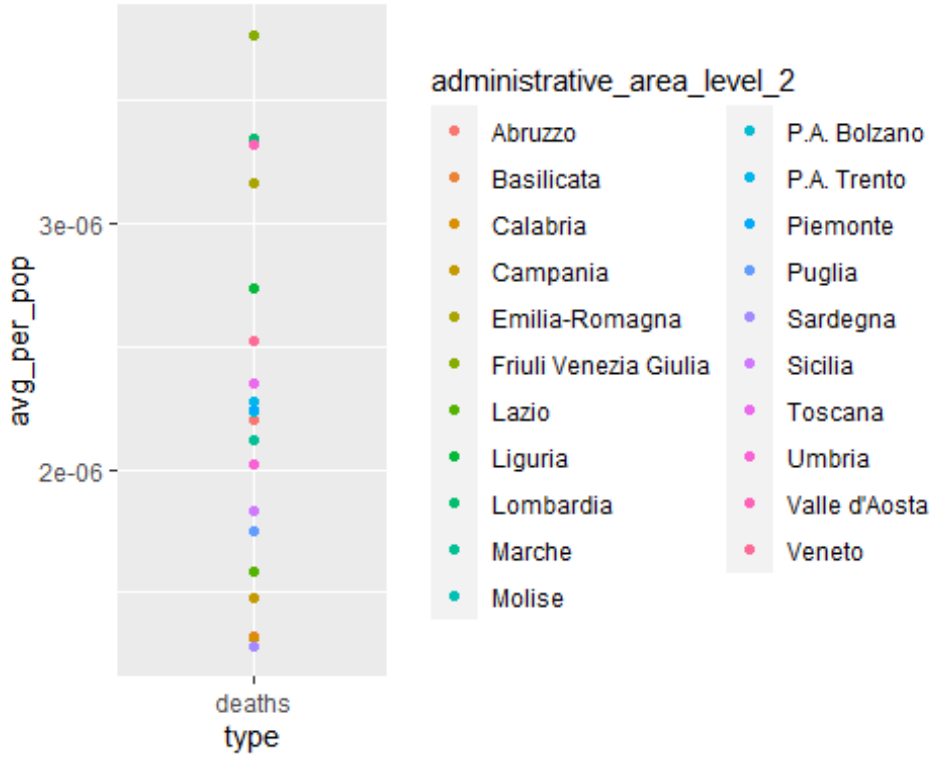
```
ggplot(filter(descriptive_stats_ITA_reg,type=="confirmed"))+geom_point(mappin
g = aes(x=type,y=avg_per_pop,color=administrative_area_level_2))
```




```
ggplot(filter(descriptive_stats_ITA_reg,type=="recovered"))+geom_point(mappin
g = aes(x=type,y=avg_per_pop,color=administrative_area_level_2))
```




```
ggplot(filter(descriptive_stats_ITA_reg,type=="deaths"))+geom_point(mapping =
aes(x=type,y=avg_per_pop,color=administrative_area_level_2))
```



We can extract the best and the worst regions with respect to confirmed cases, deaths and recovered :

```
dy = group_by(descriptive_stats_ITA_reg,type)

filter(dy,avg_per_pop==max(avg_per_pop)) # we extract the max avg_per_pop
```

```
## # A tibble: 3 × 13
## # Groups:   type [3]
##   administrative_area_…¹ population   min    q1   mean median     sd   mad
q3
##   <chr>                       <int> <int> <dbl>  <dbl>  <dbl>  <dbl> <dbl>
<dbl>
## 1 P.A. Bolzano               527750     0    13 220.       63 483.   87.5
260.
## 2 Friuli Venezia Giulia     1215538     0     0   4.58      2   6.93  2.97
5
## 3 P.A. Bolzano               527750     0    12 222.       53 600.   74.1
252.
## # ℹ abbreviated name: ¹administrative_area_level_2
## # ℹ 4 more variables: IQR <dbl>, max <int>, type <chr>, avg_per_pop <dbl>
```

The region with the most confirmed cases and recovered is P.A Bolzano and the region with the most number of deaths is Friuli Venezia Giulia.

```
filter(dy,avg_per_pop==min(avg_per_pop)) # we extract the min avg_per_pop
```

```
## # A tibble: 3 × 13
## # Groups:   type [3]
##   administrative_area_level_2 population   min    q1   mean median     sd
mad
##   <chr>                            <int> <int> <dbl>  <dbl>  <dbl>  <dbl>
<dbl>
## 1 Sardegna                       1648176     0    34 373.      110 612.
148.
## 2 Sardegna                       1648176     0     0   2.11      1   3.21
1.48
## 3 Sardegna                       1648176     0    16 365.       85 675.
123.
## # ℹ 5 more variables: q3 <dbl>, IQR <dbl>, max <int>, type <chr>,
## #   avg_per_pop <dbl>
```

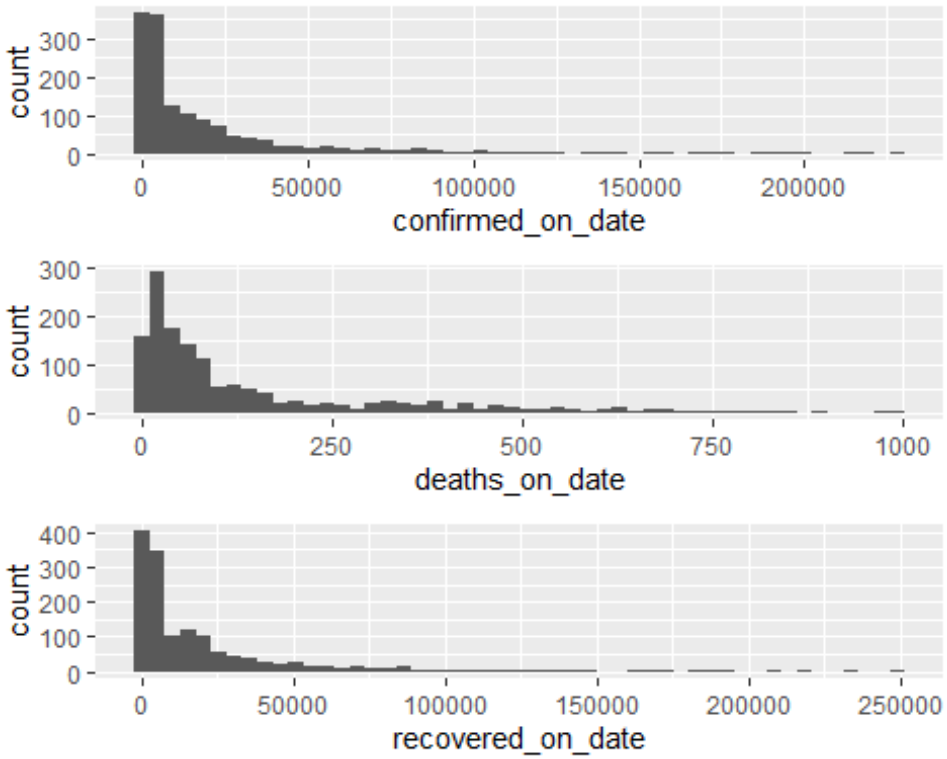
The region with the less confirmed cases, deaths and recovered is Sardegna. Sardegna seems to have been spared

We can visualize the distribution of the confirmed cases, deaths and recovered using ggplot2.

We can display first the histogram for each variables :

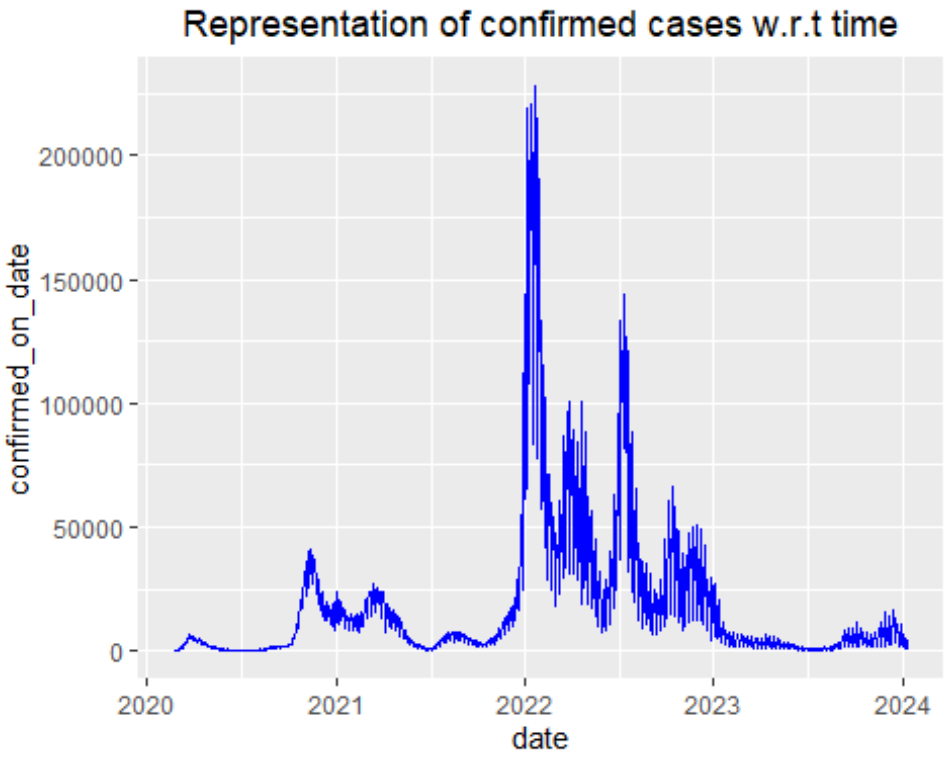
```
# here we use grid.arrange from grid.extra package to make a subplot
h1=ggplot(sample_covid19_ITA)+geom_histogram(mapping =
```

```
aes(x=confirmed_on_date),bins=50)

h2=ggplot(sample_covid19_ITA)+geom_histogram(mapping =
aes(x=deaths_on_date),bins=50)

h3=ggplot(sample_covid19_ITA)+geom_histogram(mapping =
aes(x=recovered_on_date),bins=50)

grid.arrange(h1,h2,h3,nrow=3)
```

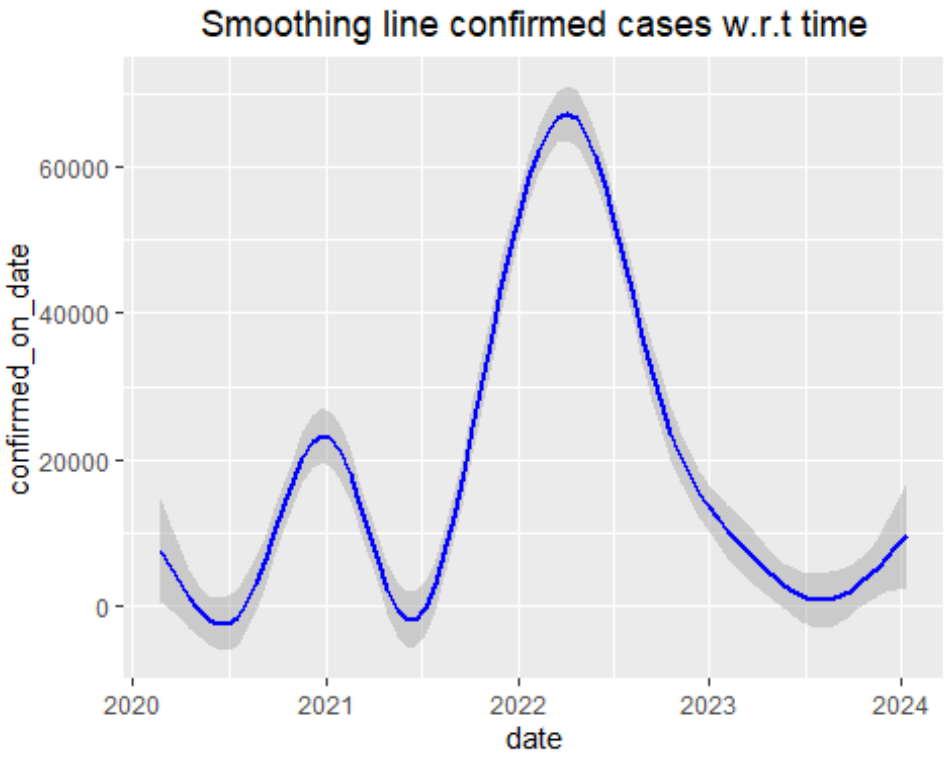


To have a more precise idea we can display each variables with respect to time and also we can display a smoothing line of the representation :

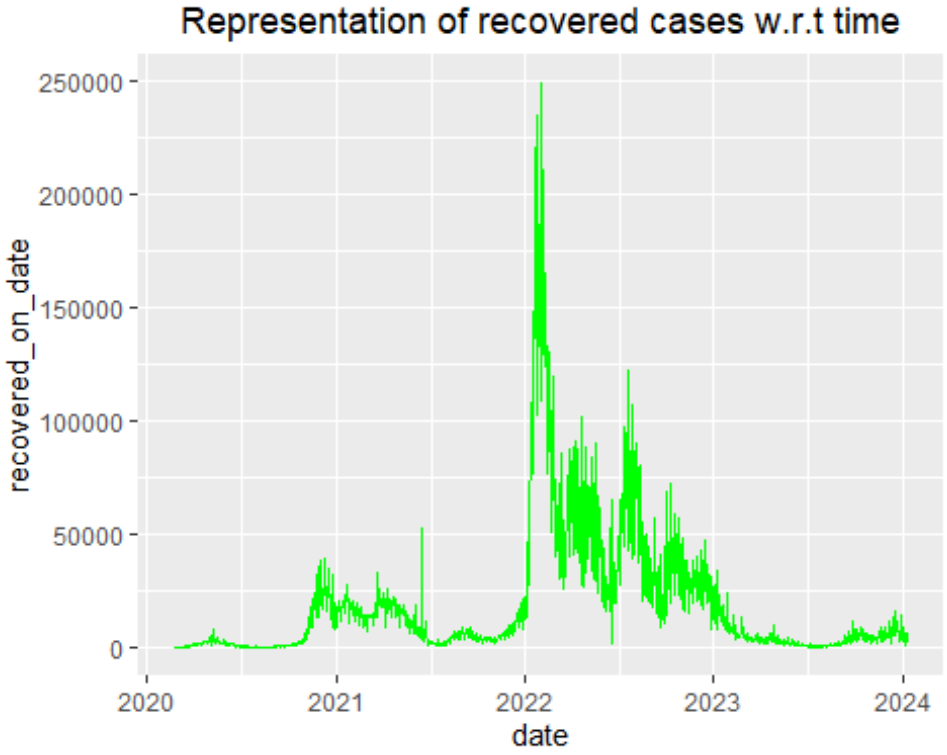
```
ggplot(sample_covid19_ITA)+geom_line(mapping =
aes(x=date,y=confirmed_on_date), color = "blue")+ggtitle("Representation of
confirmed cases w.r.t time")+theme(plot.title = element_text(hjust = 0.5))
```



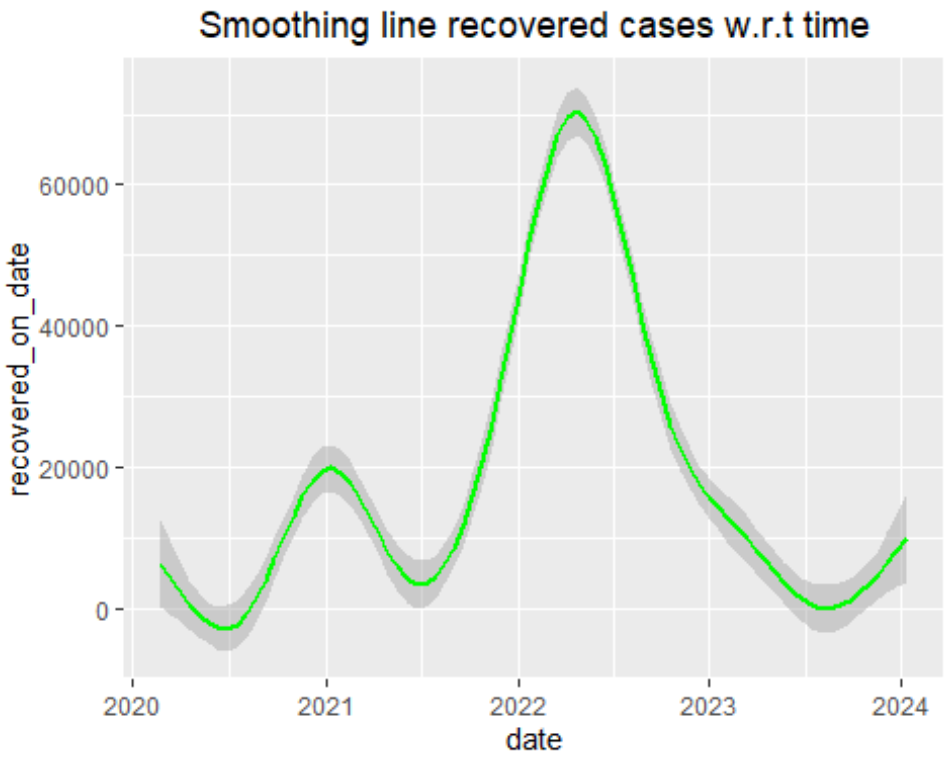

```
ggplot(sample_covid19_ITA)+geom_smooth(mapping =
aes(x=date,y=confirmed_on_date), color = "blue")+ggtitle("Smoothing line
confirmed cases w.r.t time")+theme(plot.title = element_text(hjust = 0.5))

## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```



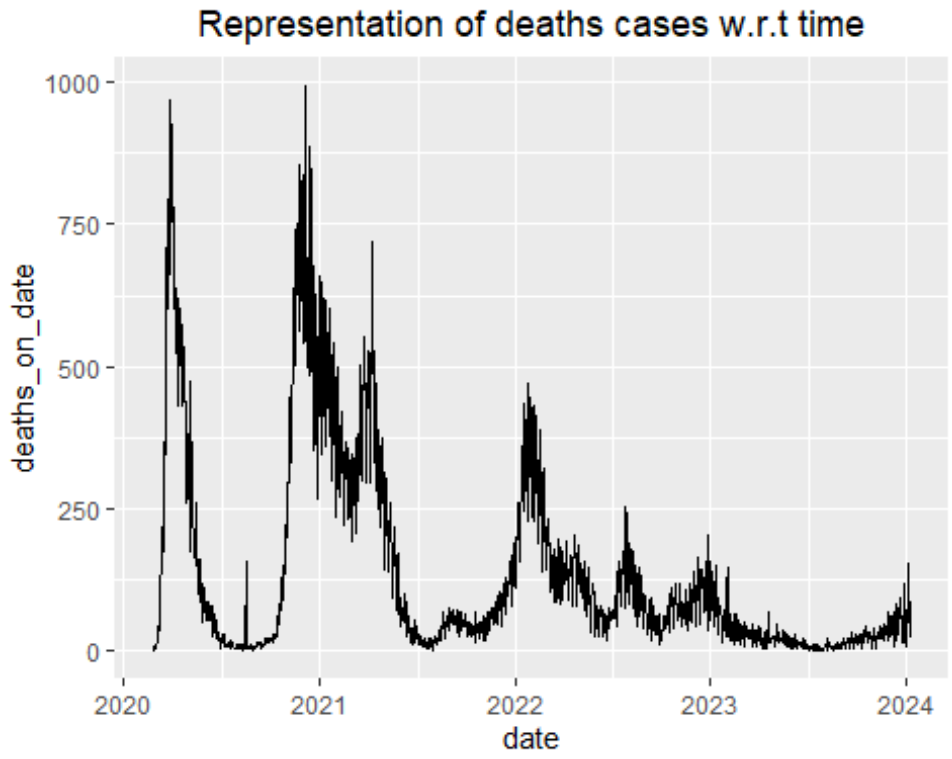

```
ggplot(sample_covid19_ITA)+geom_line(mapping =
aes(x=date,y=recovered_on_date), color = "green")+ggtitle("Representation of
recovered cases w.r.t time")+theme(plot.title = element_text(hjust = 0.5))
```



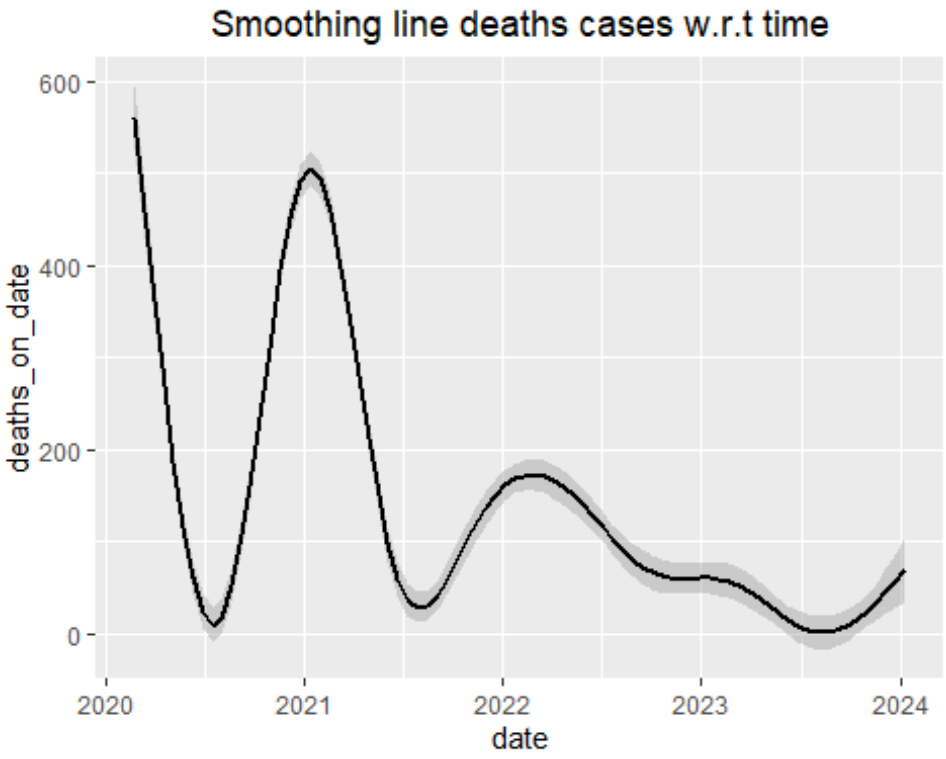
```
ggplot(sample_covid19_ITA)+geom_smooth(mapping =
aes(x=date,y=recovered_on_date), color = "green")+ggtitle("Smoothing line
recovered cases w.r.t time")+theme(plot.title = element_text(hjust = 0.5))

## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```




```
ggplot(sample_covid19_ITA)+geom_line(mapping = aes(x=date,y=deaths_on_date),
color = "black")+ggtitle("Representation of deaths cases w.r.t
time")+theme(plot.title = element_text(hjust = 0.5))
```

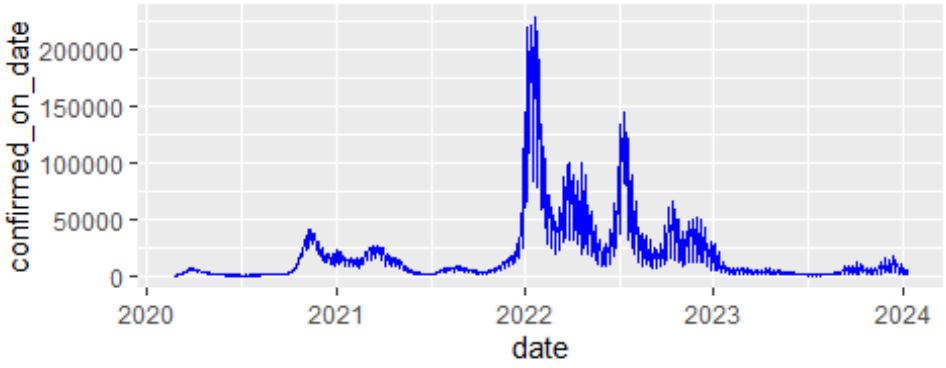


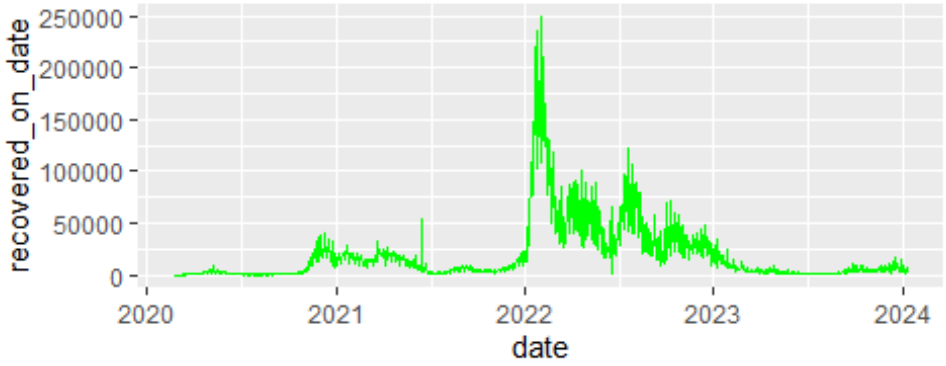

```
ggplot(sample_covid19_ITA)+geom_smooth(mapping =
aes(x=date,y=deaths_on_date), color = "black")+ggtitle("Smoothing line deaths
cases w.r.t time")+theme(plot.title = element_text(hjust = 0.5))

## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```



In particular we can see that the distribution of the confirmed cases seems very similar of the distribution of recovered.

```
s1=ggplot(sample_covid19_ITA)+geom_line(mapping =
aes(x=date,y=confirmed_on_date),color="blue")

s2=ggplot(sample_covid19_ITA)+geom_line(mapping =
aes(x=date,y=recovered_on_date),color="green")

grid.arrange(s1,s2)
```





The two graphs have the same trend, we can check whether there is a positive correlation between them.

```
cor(select(sample_covid19_ITA,confirmed_on_date,deaths_on_date,recovered_on_d
ate))
```

```
##                    confirmed_on_date deaths_on_date recovered_on_date
## confirmed_on_date          1.0000000      0.2351355         0.8127269
## deaths_on_date             0.2351355      1.0000000         0.2577448
## recovered_on_date          0.8127269      0.2577448         1.0000000
```

We can see that their correlation is close to 1 (>0.8) so it's clear that there is a positive relationship between both. Moreover it seems logical because the majority of people survive of covid19.

We can also see the numerous waves and we may ask ourselves if we can find a pattern which is repeated over time (see later with time-series).

Temporal Trends:

- Analyze temporal patterns in COVID-19 cases, deaths, and recoveries over time using time-series plots.
- Identify peak periods and variations.

We can use time series to apply some useful functions to have a idea concerning the trend and the seasonality of our variables and also we can make predictions.

We can record the start day of the time series. Here it's the 54 days of 2020 which corresponds to February 24.

```
start_day=as.Date(head(sample_covid19_ITA$date,n = 1))-as.Date("2020-01-01")
# we use head to have directly access of the first value of a column
start_day=as.numeric(start_day)
start_day
```

```
## [1] 54
```

We use decompose method to display the components of our time series.

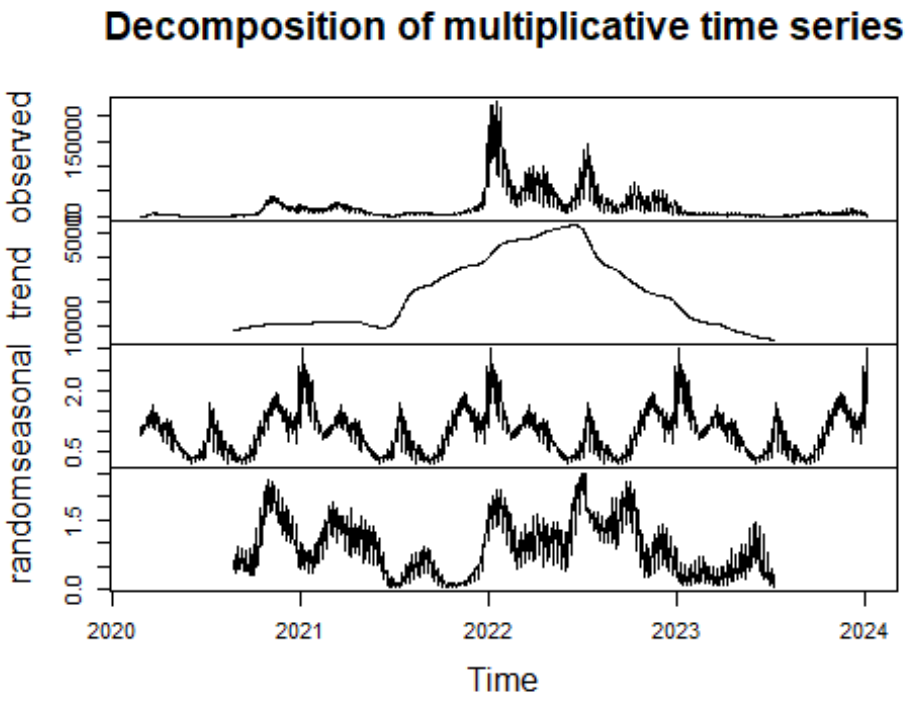
For the following we use decomposition of type "multiplicative" as below :

$$Y_t = \hat{S}_t * \hat{T}_t * \hat{R}_t$$

Where :

- $\hat{S}$ is the seasonality term
- $\hat{T}$ is the trend
- $\hat{R}$ is the random term

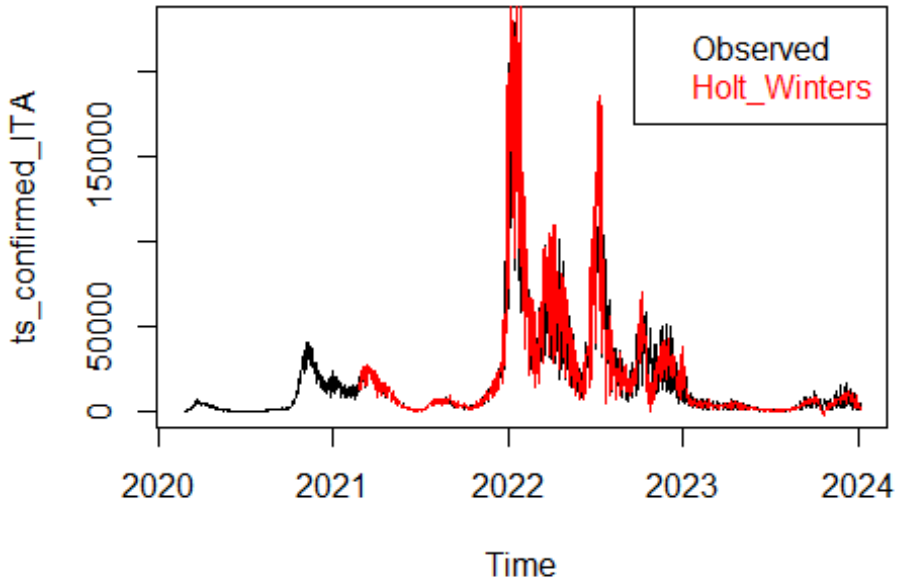
```
ts_confirmed_ITA= ts(data =
sample_covid19_ITA$confirmed_on_date,frequency=365,start = c(2020,start_day))
components_ts_confirmed_ITA = decompose(ts_confirmed_ITA,type
="multiplicative") # we use decompose to store in particular the values of
the trend and the seasonality coefficients
plot(components_ts_confirmed_ITA)
```



The seasonality component show the waves of covid19 cases.

We can use a exponential smoothing as Holt-Winters to approximate our time series and make predictions.

```
HW_ts_confirmed_ITA = HoltWinters(ts_confirmed_ITA,seasonal =
"multiplicative")

plot(x=ts_confirmed_ITA,col="black")
lines(HW_ts_confirmed_ITA$fitted[,1],col="red")
legend("topright",legend=c("Observed","Holt_Winters"),text.col =
c("black","red"))
```



We can make prediction on the 10 next days

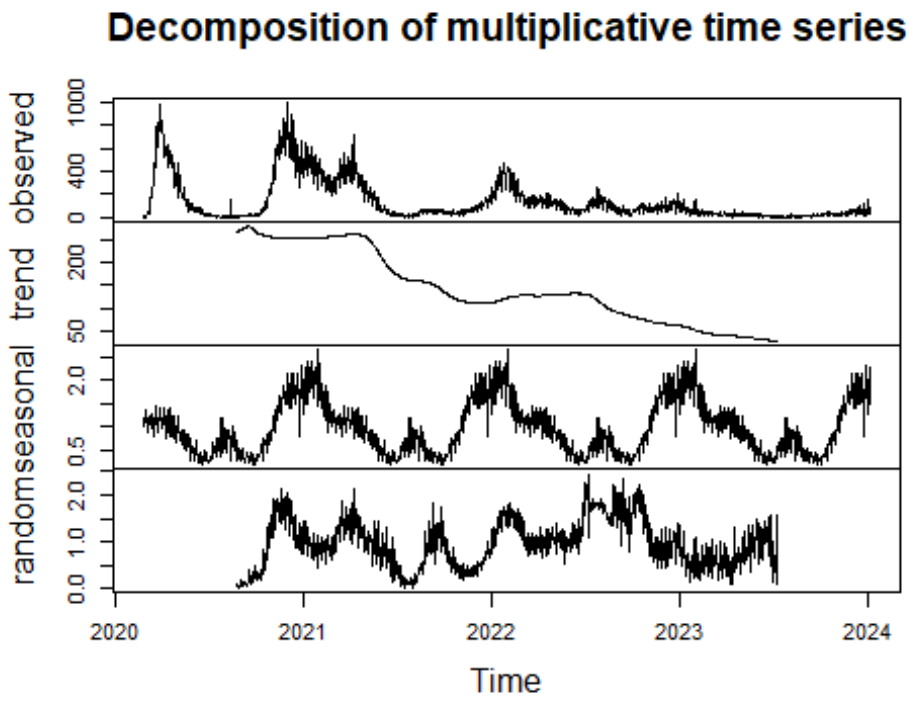
```
HW_ts_confirmed_ITA_pred =
predict(HW_ts_confirmed_ITA,10,prediction.interval=FALSE,level=0.95)
HW_ts_confirmed_ITA_pred
```

```
## Time Series:
## Start = c(2024, 7)
## End = c(2024, 16)
## Frequency = 365
##               fit
##  [1,] 2204.15646
##  [2,] 2246.94337
##  [3,] 1304.75848
##  [4,]  960.61216
##  [5,]  634.27922
##  [6,]  307.35059
##  [7,]  -61.21872
##  [8,] -373.33263
##  [9,] -713.31179
## [10,] -822.50136
```

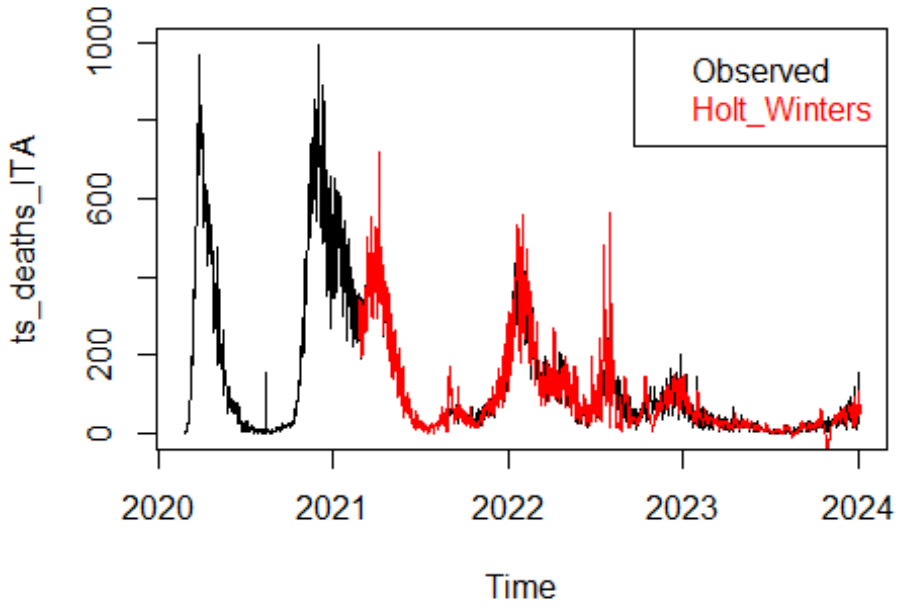
We can see that the numbers of confirmed cases might decrease in the next day (don't regard the possible negatives values it's just a prediction trend we can't have negative value).

We can do the same for deaths.

```r
ts_deaths_ITA= ts(data =
sample_covid19_ITA$deaths_on_date,frequency=365,start = c(2020,start_day))
components_ts_deaths_ITA = decompose(ts_deaths_ITA,type ="multiplicative") #
we use decompose to store in particular the values of the trend and the
seasonality coefficients
plot(components_ts_deaths_ITA)
```




```r
HW_ts_deaths_ITA = HoltWinters(ts_deaths_ITA,seasonal = "multiplicative")

plot(x=ts_deaths_ITA,col="black")
lines(HW_ts_deaths_ITA$fitted[,1],col="red")
legend("topright",legend=c("Observed","Holt_Winters"),text.col =
c("black","red"))
```




```
HW_ts_deaths_ITA_pred =
predict(HW_ts_deaths_ITA,10,prediction.interval=FALSE,level=0.95)
HW_ts_deaths_ITA_pred
```

```
## Time Series:
## Start = c(2024, 7)
## End = c(2024, 16)
## Frequency = 365
##             fit
##  [1,]  72.62134
##  [2,] 105.93433
##  [3,]  83.94918
##  [4,] 101.54547
##  [5,]  90.38365
##  [6,]  92.89077
##  [7,]  60.88168
##  [8,]  92.74316
##  [9,]  90.68098
## [10,]  93.06017
```

We can see that in the next day the number of deaths might increase

Identify peak period and variations :

We use “quantmod” package Quantitative Financial Modelling Framework which offers many options to deals with peaks/valleys. The idea is to visualize the peaks and the valleys on each variables.

```
library("quantmod")

## Le chargement a nécessité le package : xts

## Le chargement a nécessité le package : zoo

##
## Attachement du package : 'zoo'

## Les objets suivants sont masqués depuis 'package:base':
##
##     as.Date, as.Date.numeric

##
## ############################ Warning from 'xts' package
############################
## #
#
## # The dplyr lag() function breaks how base R's lag() function is supposed
to  #
## # work, which breaks lag(my_xts). Calls to lag(my_xts) that you type or
#
## # source() into this session won't work correctly.
#
## #
#
## # Use stats::lag() to make sure you're not using dplyr::lag(), or you can
add #
## # conflictRules('dplyr', exclude = 'lag') to your .Rprofile to stop
#
## # dplyr from breaking base R's lag() function.
#
## #
#
## # Code in packages is not affected. It's protected by R's namespace
mechanism #
## # Set `options(xts.warn_dplyr_breaks_lag = FALSE)` to suppress this
warning.  #
## #
#
##
################################################################################
##

##
## Attachement du package : 'xts'

## Les objets suivants sont masqués depuis 'package:dplyr':
##
##     first, last

## Le chargement a nécessité le package : TTR
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

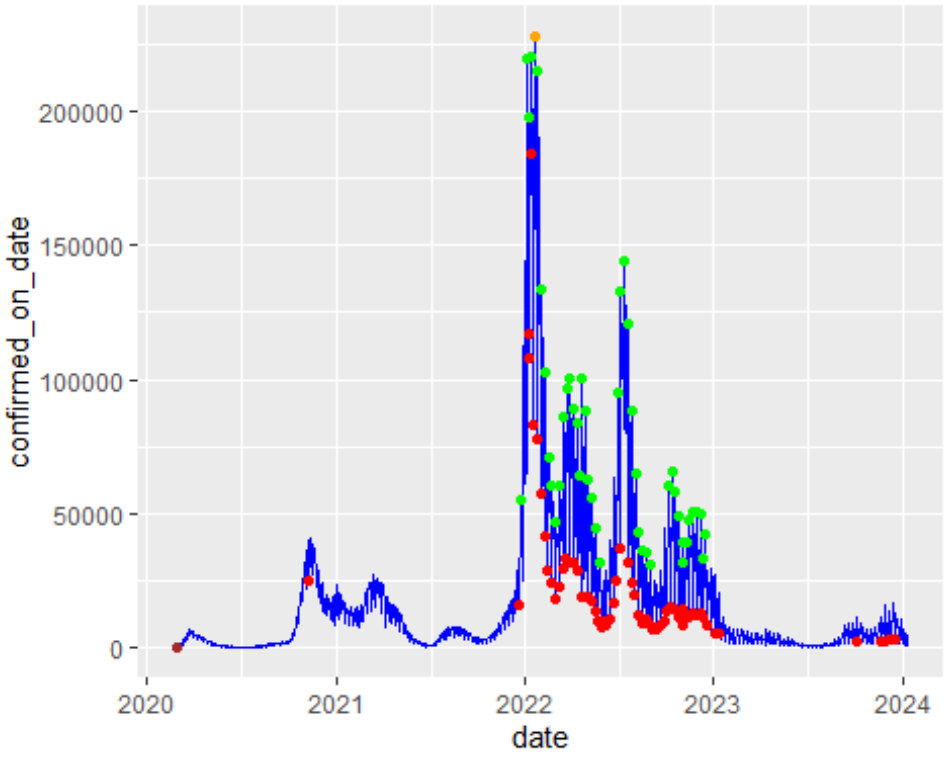
We use the mean absolute deviation as threshold because it's a robust indicator of the spread in data less sensitive to extreme value than standard deviation.

We also apply findPeaks and findValleys methods from the package which allows to collect the index of peaks and valleys.

```
threshold=descriptive_stats_ITA["confirmed","mad"] # a threshold is use here
to materialize the difference in terms of values between the peaks/valleys
peaks= findPeaks(sample_covid19_ITA$confirmed_on_date,thresh =threshold)-1
valleys= findValleys(sample_covid19_ITA$confirmed_on_date,thresh=threshold)-1

i_max=which.max(sample_covid19_ITA$confirmed_on_date)
i_min=which.min(sample_covid19_ITA$confirmed_on_date)
```

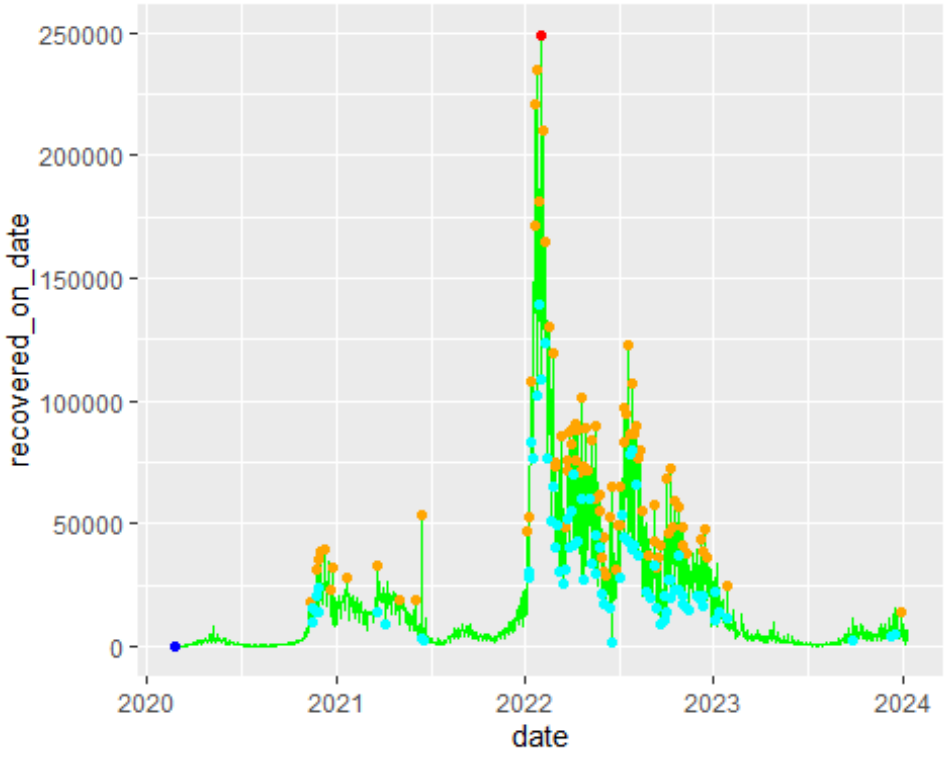
Using ggplot we put points for peaks and valleys with different colors.

```
ggplot(sample_covid19_ITA,aes(date,confirmed_on_date))+geom_line(color="blue"
)+geom_point(data = ~ .[peaks,], color = "green")+geom_point(data = ~
.[valleys,], color =
"red")+geom_point(aes(x=date[i_max],y=max(confirmed_on_date)),color="orange")
+geom_point(aes(x=date[i_min],y=min(confirmed_on_date)),color="brown")
```

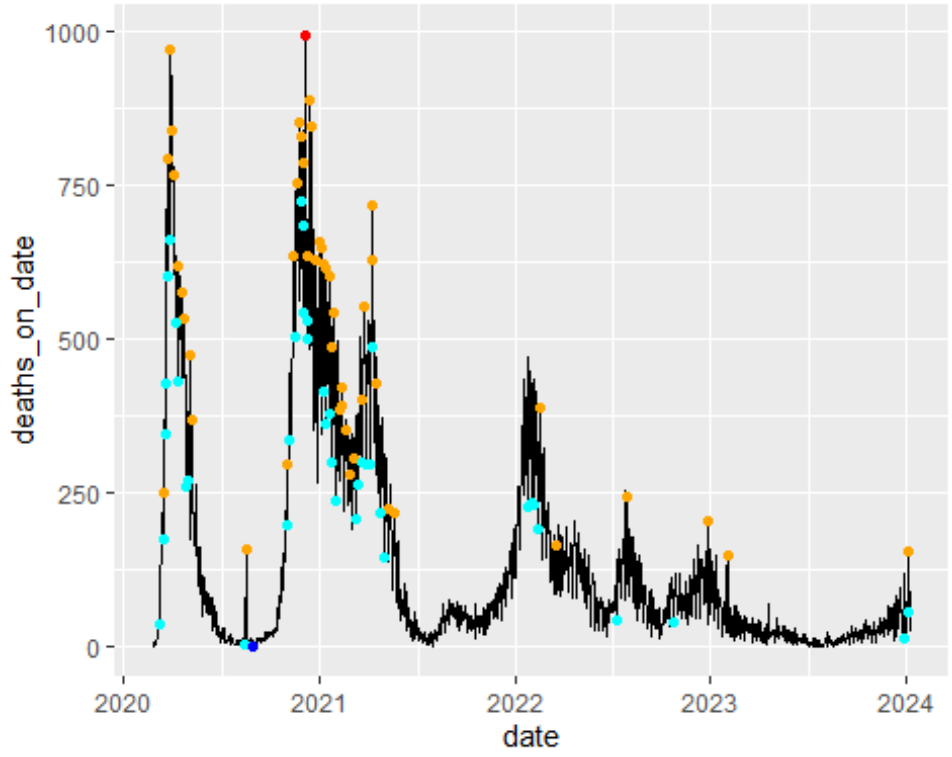


We apply the same logic for others variables.

```r
threshold=descriptive_stats_ITA["recovered","mad"] # a threshold is use here to materialize the difference in terms of values between the peaks/valleys
peaks= findPeaks(sample_covid19_ITA$recovered_on_date,thresh =threshold)-1
valleys= findValleys(sample_covid19_ITA$recovered_on_date,thresh=threshold)-1

i_max=which.max(sample_covid19_ITA$recovered_on_date)
i_min=which.min(sample_covid19_ITA$recovered_on_date)

ggplot(sample_covid19_ITA,aes(date,recovered_on_date))+geom_line(color="green")+geom_point(data = ~ .[peaks,], color = "orange")+geom_point(data = ~ .[valleys,], color = "cyan")+geom_point(aes(x=date[i_max],y=max(recovered_on_date)),color="red")+geom_point(aes(x=date[i_min],y=min(recovered_on_date)),color="blue")
```




```r
threshold=descriptive_stats_ITA["deaths","mad"] # a threshold is use here to materialize the difference in terms of values between the data
peaks= findPeaks(sample_covid19_ITA$deaths_on_date,thresh =threshold)-1
valleys= findValleys(sample_covid19_ITA$deaths_on_date,thresh=threshold)-1

i_max=which.max(sample_covid19_ITA$deaths_on_date)
i_min=which.min(sample_covid19_ITA$deaths_on_date)

ggplot(sample_covid19_ITA,aes(date,deaths_on_date))+geom_line(color="black")+geom_point(data = ~ .[peaks,], color = "orange")+geom_point(data = ~ .[valleys,], color = "cyan")+geom_point(aes(x=date[i_max],y=max(deaths_on_date)),color="red")+geom_point(aes(x=date[i_min],y=min(deaths_on_date)),color="blue")
```



What we can observe in general way is that for confirmed cases and recovered we have high variations in the data from 2022 to 2023. Since this date the situation is now less volatile. For "deaths" we have lot of variability in the first and second wave and since 2022 the situation is in improvement probably link to government measures and vaccination.

Comparative Analysis:

- Compare the progression of COVID-19 in different countries or regions

France:

We reuse the same principles of data cleaning and pre-processing as before

```
covid19_FR = covid19(country='FR',verbose = FALSE) # verbose = False to avoid
the printing of a message from the authors.
covid19_FR

sample_covid19_FR =
select(covid19_FR,date,confirmed,deaths,recovered,population)
sample_covid19_FR
```

Let's check the Nan values.

```
sum(is.na(sample_covid19_FR))

## [1] 1089
```

We have lot of Nan values. To have more precise idea, we compute the numbers of Nan for each columns.

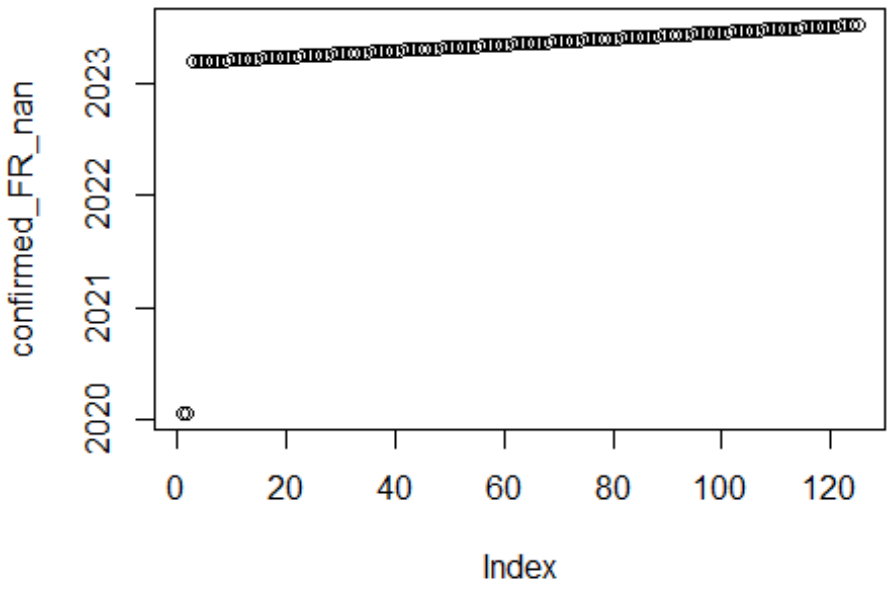
```r
colSums(is.na(sample_covid19_FR))
```

```
##       date  confirmed     deaths  recovered population
##          0        125        147        817          0
```

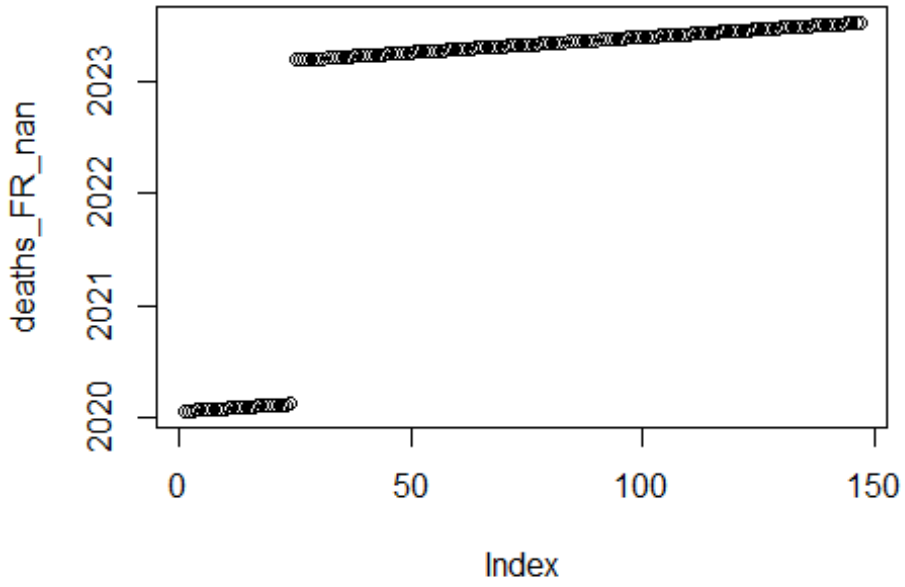
We directly see that recovered variable is useless because there are nearly 66% of Nan values.

For confirmed cases and deaths we have lots of Nan values and we have to check if we can deal with before make analysis.

```r
confirmed_FR_nan=filter(sample_covid19_FR,is.na(confirmed))$date # we use filter and $date to conserve only the date of Nan values
deaths_FR_nan=filter(sample_covid19_FR,is.na(deaths))$date
plot(confirmed_FR_nan)
```




```r
plot(deaths_FR_nan)
```



From the 2 figure above we can derive that the Nan values corresponds of the beginnings days of the pandemic where we don't have information and the days since October 2023 where it seems that we stop to count the cases. So the idea is simple, remove the rows where there are Nan values. If we choose to replace by 0 the Nan values of "confirmed" and "deaths" we can negatively affected the trend and the statistics that we want have.

Now we can remove "recovered" variable and the Nan values in "confirmed" and "deaths"

```
sample_covid19_FR = filter(subset(sample_covid19_FR,select = -
recovered),(!is.na(confirmed)& !is.na(deaths)))
sample_covid19_FR
```

We can check that "date" variable is effectively in the Date type

```
class(sample_covid19_FR$date)

## [1] "Date"
```

We can store the dimension as for Italia data.

```
dim_covid19_FR = dim(sample_covid19_FR)
dim_covid19_FR

## [1] 1119    4
```

As before the idea is to transform cumulative variable into variable with value for each day

```
confirmed_on_date_FR =
c(sample_covid19_FR$confirmed[1],sample_covid19_FR$confirmed[2:dim_covid19_FR
[1]]-sample_covid19_FR$confirmed[1:dim_covid19_FR[1]-1])
####
deaths_on_date_FR =
c(sample_covid19_FR$deaths[1],sample_covid19_FR$deaths[2:dim_covid19_FR[1]]-
sample_covid19_FR$deaths[1:dim_covid19_FR[1]-1])

confirmed_on_date_FR = data.frame(confirmed_on_date_FR)
deaths_on_date_FR = data.frame(deaths_on_date_FR)

sample_covid19_FR =
cbind(sample_covid19_FR,c(confirmed_on_date_FR,deaths_on_date_FR))
```

We can remove the old variable

```
sample_covid19_FR = subset(sample_covid19_FR,select= -(confirmed:deaths))
```

We apply a filter to remove negatives values. When studying the data it seems that in France there are lots of weeks-end where we don't count the cases (cases Saturday/Sunday equal cases Friday) so we transform the cumulative variable we obtain lots of 0. It seems logical to remove also these points otherwise it can biased analysis.

```
sample_covid19_FR=filter(sample_covid19_FR,(confirmed_on_date_FR>0 &
deaths_on_date_FR>0))
```

We obtain our desired data frame of covid19 for France.

```
sample_covid19_FR
```

As before for Italy we compute descriptive statistics for France and we display the trend for each variables.

```
confirmed_stats=summarise(sample_covid19_FR,min=min(confirmed_on_date_FR),q1=
quantile(confirmed_on_date_FR,c(0.25)),mean=mean(confirmed_on_date_FR),median
=median(confirmed_on_date_FR),sd=sd(confirmed_on_date_FR),mad=mad(confirmed_o
n_date_FR),q3=quantile(confirmed_on_date_FR,c(0.75)),IQR=IQR(confirmed_on_dat
e_FR),max=max(confirmed_on_date_FR))
######
deaths_stats=summarise(sample_covid19_FR,min=min(deaths_on_date_FR),q1=quanti
le(deaths_on_date_FR,c(0.25)),mean=mean(deaths_on_date_FR),median=median(deat
hs_on_date_FR),sd=sd(deaths_on_date_FR),mad=mad(deaths_on_date_FR),q3=quantil
e(deaths_on_date_FR,c(0.75)),IQR=IQR(deaths_on_date_FR),max=max(deaths_on_dat
e_FR))

descriptive_stats_FR=rbind(confirmed_stats,deaths_stats)

rownames(descriptive_stats_FR)=c("confirmed_FR","deaths_FR")

descriptive_stats_FR
```

```
##                min      q1       mean  median         sd        mad       q3
## confirmed_FR     4 4589.00 40302.9686 19214.0 66443.1217 23153.7642 44964.00
```

```
## deaths_FR      1   39.75   163.7918    94.5   198.3059    98.5929   207.75
##                IQR    max
## confirmed_FR 40375 501635
## deaths_FR      168   1436
```

Using the previous data frame we can have some information about the impact of covid19 in France. Especially we have a mean of more 40000 confirmed cases per day and near 160 deaths in mean per day.

We can compute the letality it means the total number of deaths over the total number of cases.

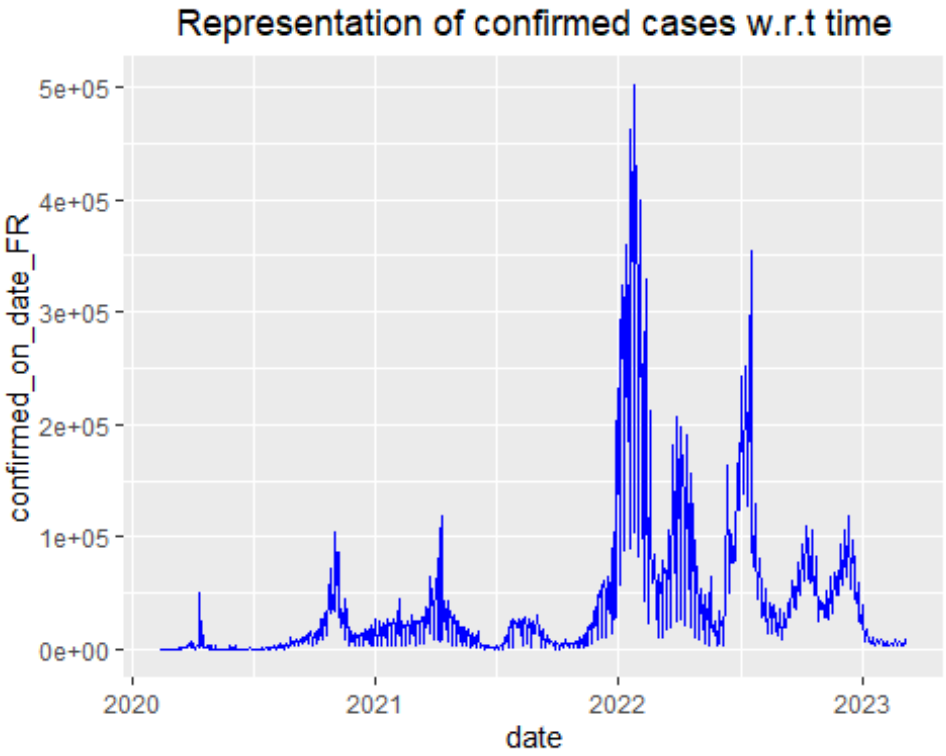
```
lethality_FR=(sum(sample_covid19_FR$deaths_on_date_FR)/sum(sample_covid19_FR$
confirmed_on_date_FR))*100
lethality_FR
```

```
## [1] 0.4064014
```

This results means that if we contract covid19 we have approximetaly 0.4% to die.

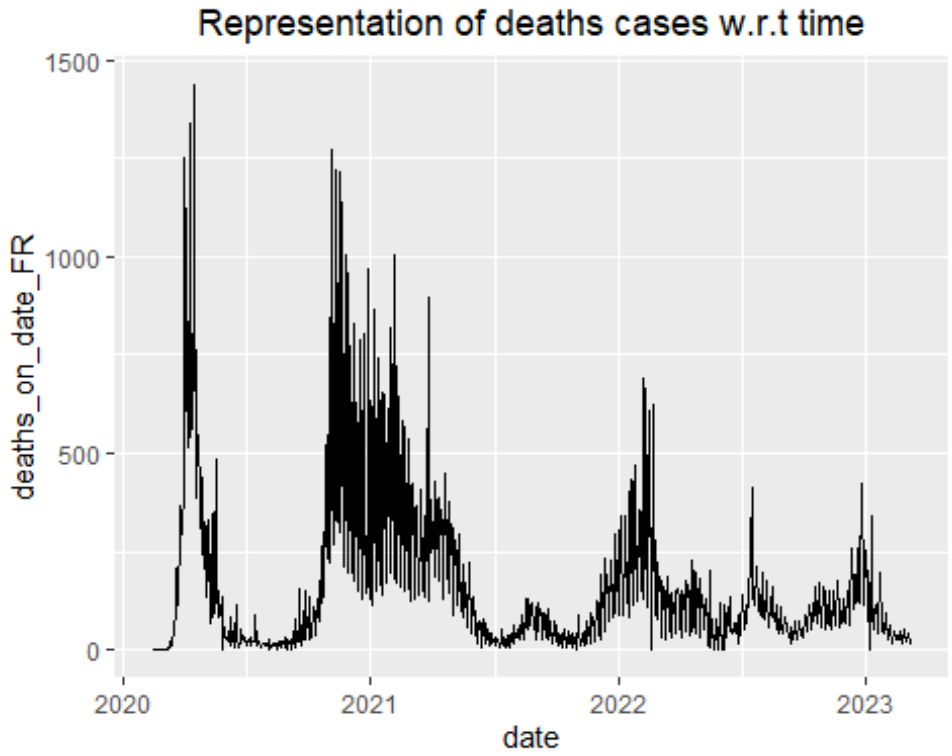
We can display the graph of the data :

```
ggplot(sample_covid19_FR)+geom_line(mapping =
aes(x=date,y=confirmed_on_date_FR), color = "blue")+ggtitle("Representation
of confirmed cases w.r.t time")+theme(plot.title = element_text(hjust = 0.5))
```



The trend seems very similar to that in Italy. For the “deaths” we have :

```
ggplot(sample_covid19_FR)+geom_line(mapping =
aes(x=date,y=deaths_on_date_FR), color = "black")+ggtitle("Representation of
deaths cases w.r.t time")+theme(plot.title = element_text(hjust = 0.5))
```



Idem the distribution seems to be very close to Italy. We keep these information for France and we apply the same principle for Germany.

Germany

```
covid19_GER = covid19(country='DEU',verbose = FALSE) # verbose = False to
avoid the printing of a message from the authors.
covid19_GER
```

As before we extract only our interested columns

```
sample_covid19_GER =
select(covid19_GER,date,confirmed,deaths,recovered,population)
sample_covid19_GER
```

As before we have to check Nan values/Missing Values:

```
sum(is.na(sample_covid19_GER))

## [1] 36

colSums(is.na(sample_covid19_GER))

##       date  confirmed     deaths  recovered population
##          0         12         12         12          0
```

Studying the data for Germany, we can see that the Nan values are situated on the same rows and all except one are on the beginnings day. The other row with Nan values is the last data point, where they stop to count. So it's seems logical to purely remove these rows.

```
sample_covid19_GER=filter(sample_covid19_GER,(!is.na(confirmed)&!is.na(deaths
)&!is.na(recovered)))

sample_covid19_GER
```

We store the dimension as for the others countries.

```
dim_covid19_GER = dim(sample_covid19_GER)
```

We transform cumulative variables into variables with values per days.

```
confirmed_on_date_GER =
c(sample_covid19_GER$confirmed[1],sample_covid19_GER$confirmed[2:dim_covid19_
GER[1]]-sample_covid19_GER$confirmed[1:dim_covid19_GER[1]-1])
####
deaths_on_date_GER =
c(sample_covid19_GER$deaths[1],sample_covid19_GER$deaths[2:dim_covid19_GER[1]
]-sample_covid19_GER$deaths[1:dim_covid19_GER[1]-1])
####
recovered_on_date_GER =
c(sample_covid19_GER$recovered[1],sample_covid19_GER$recovered[2:dim_covid19_
GER[1]]-sample_covid19_GER$recovered[1:dim_covid19_GER[1]-1])

confirmed_on_date_GER=data.frame(confirmed_on_date_GER)
deaths_on_date_GER=data.frame(deaths_on_date_GER)
recovered_on_date_GER=data.frame(recovered_on_date_GER)

sample_covid19_GER=cbind(sample_covid19_GER,c(confirmed_on_date_GER,deaths_on
_date_GER,recovered_on_date_GER))
```

We remove the old variables:

```
sample_covid19_GER = subset(sample_covid19_GER,select= -
(confirmed:recovered))
```

We remove negative values:

```
sample_covid19_GER=filter(sample_covid19_GER,(confirmed_on_date_GER>=0&deaths
_on_date_GER>=0&recovered_on_date_GER>=0))
```

As before for France and Italy, we compute the same statistics and the trend of the data.

```
confirmed_stats=summarise(sample_covid19_GER,min=min(confirmed_on_date_GER),q
1=quantile(confirmed_on_date_GER,c(0.25)),mean=mean(confirmed_on_date_GER),me
dian=median(confirmed_on_date_GER),sd=sd(confirmed_on_date_GER),mad=mad(confi
rmed_on_date_GER),q3=quantile(confirmed_on_date_GER,c(0.75)),IQR=IQR(confirme
d_on_date_GER),max=max(confirmed_on_date_GER))
######
deaths_stats=summarise(sample_covid19_GER,min=min(deaths_on_date_GER),q1=quan
```

```
tile(deaths_on_date_GER,c(0.25)),mean=mean(deaths_on_date_GER),median=median(
deaths_on_date_GER),sd=sd(deaths_on_date_GER),mad=mad(deaths_on_date_GER),q3=
quantile(deaths_on_date_GER,c(0.75)),IQR=IQR(deaths_on_date_GER),max=max(deat
hs_on_date_GER))
######
recovered_stats=summarise(sample_covid19_GER,min=min(recovered_on_date_GER),q
1=quantile(recovered_on_date_GER,c(0.25)),mean=mean(recovered_on_date_GER),me
dian=median(recovered_on_date_GER),sd=sd(recovered_on_date_GER),mad=mad(recov
ered_on_date_GER),q3=quantile(recovered_on_date_GER,c(0.75)),IQR=IQR(recovere
d_on_date_GER),max=max(recovered_on_date_GER))

descriptive_stats_GER = rbind(confirmed_stats,deaths_stats,recovered_stats)

rownames(descriptive_stats_GER)=c("confirmed_GER","deaths_GER","recovered_GER
")

descriptive_stats_GER
```

```
##                min      q1       mean  median         sd        mad      q3
## confirmed_GER   1 2359.25 33944.6385 12223.0 53756.2832 16519.1292 37940.5
## deaths_GER      0   18.00   149.7185    81.5   194.8442   110.4537   208.0
## recovered_GER   0 2189.75 33559.6430 11272.5 53809.6045 15485.0157 37813.0
##                    IQR    max
## confirmed_GER 35581.25 307816
## deaths_GER      190.00   1298
## recovered_GER 35623.25 307502
```

Using the previous data frame we can have some information about the impact of covid19 in France and Italy. Especially we have a mean of nearly 34000 confirmed cases and recovered per day and near 150 deaths in mean per day.

We can compute the letality it means the total number of deaths over the total number of cases.

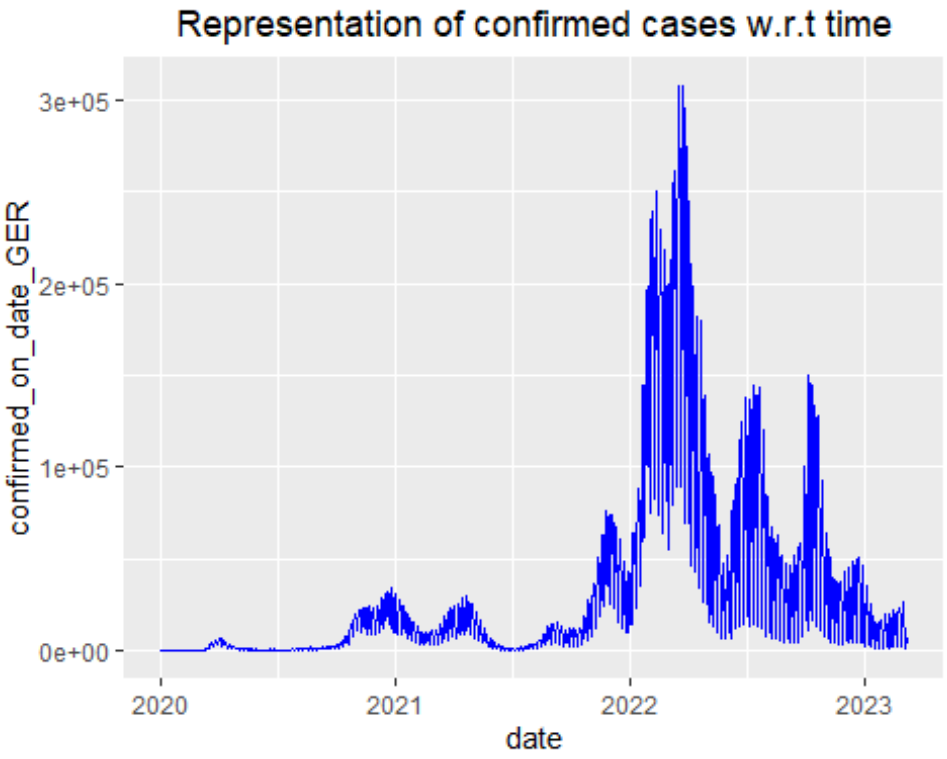
```
lethality_GER=(sum(sample_covid19_GER$deaths_on_date_GER)/sum(sample_covid19_
GER$confirmed_on_date_GER))*100
lethality_GER
```

```
## [1] 0.4410666
```

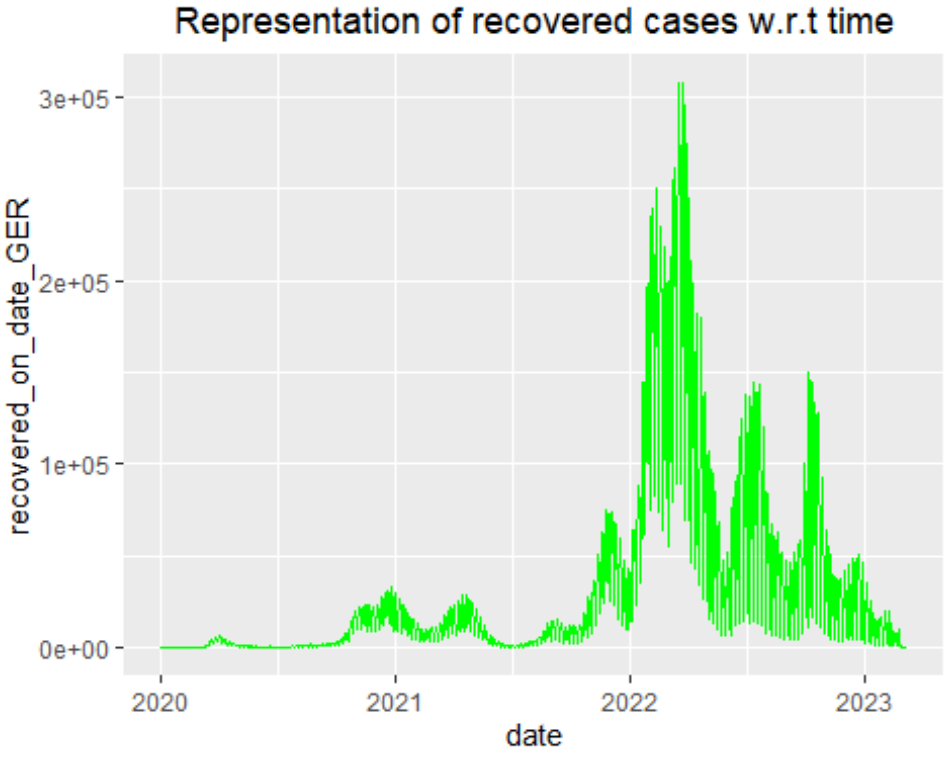
We obtain letality of approximately 0.45% it's close to value for France.

We can display the representation of the data in Germany for each variables:

```
ggplot(sample_covid19_GER)+geom_line(mapping =
aes(x=date,y=confirmed_on_date_GER), color = "blue")+ggtitle("Representation
of confirmed cases w.r.t time")+theme(plot.title = element_text(hjust = 0.5))
```

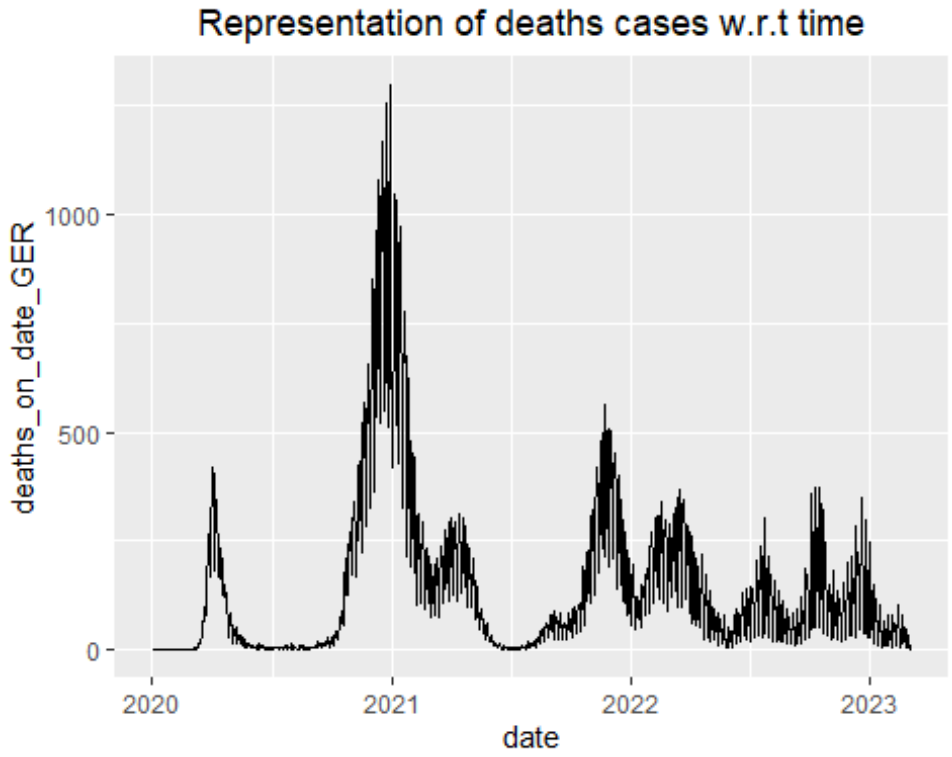



```
ggplot(sample_covid19_GER)+geom_line(mapping =
aes(x=date,y=recovered_on_date_GER), color = "green")+ggtitle("Representation
of recovered cases w.r.t time")+theme(plot.title = element_text(hjust = 0.5))
```



As before in Italy, distribution of confirmed cases and recovered are so close. Let's see the distribution for "deaths".

```
ggplot(sample_covid19_GER)+geom_line(mapping =
aes(x=date,y=deaths_on_date_GER), color = "black")+ggtitle("Representation of
deaths cases w.r.t time")+theme(plot.title = element_text(hjust = 0.5))
```



The distribution of "deaths" seems to be very similar to Italy and France.

Comparison between countries:

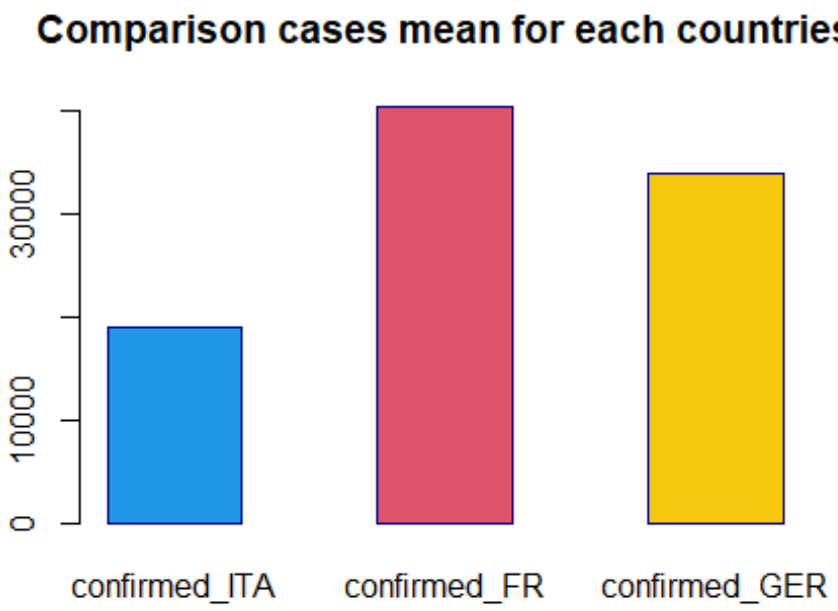
We can combine the three descriptive into one.

```
descriptive_stats_Global=rbind(descriptive_stats_ITA,descriptive_stats_FR,des
criptive_stats_GER)

descriptive_stats_Global

##                min      q1       mean  median         sd        mad
q3
## confirmed_ITA  78 2194.00 18886.9115  6652.0 30877.9141  8692.4838
22360.00
## deaths_ITA      1   24.00   138.3638    63.0   177.8264    72.6474
164.00
## recovered_ITA   0 2155.00 18598.9186  6402.0 29125.3429  8582.7714
22075.00
## confirmed_FR    4 4589.00 40302.9686 19214.0 66443.1217 23153.7642
44964.00
```

```
## deaths_FR       1   39.75   163.7918    94.5   198.3059    98.5929
207.75
## confirmed_GER   1 2359.25 33944.6385 12223.0 53756.2832 16519.1292
37940.50
## deaths_GER      0   18.00   149.7185    81.5   194.8442   110.4537
208.00
## recovered_GER   0 2189.75 33559.6430 11272.5 53809.6045 15485.0157
37813.00
##                     IQR    max
## confirmed_ITA 20166.00 228123
## deaths_ITA      140.00    993
## recovered_ITA 19920.00 248971
## confirmed_FR  40375.00 501635
## deaths_FR       168.00   1436
## confirmed_GER 35581.25 307816
## deaths_GER      190.00   1298
## recovered_GER 35623.25 307502
```

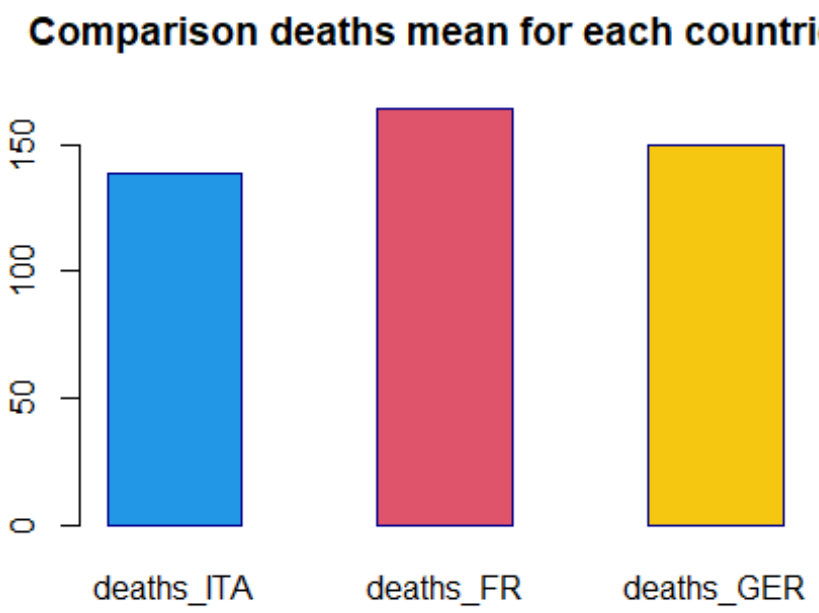
We can display several statistics to see difference between countries

```
barplot(height = cbind(confirmed_ITA =
descriptive_stats_Global["confirmed_ITA","mean"] ,confirmed_FR =
descriptive_stats_Global["confirmed_FR","mean"],confirmed_GER =
descriptive_stats_Global["confirmed_GER","mean"]),
        beside = TRUE,
        col = c(4,2,7),
        legend.text = TRUE,
        border = "dark blue",
        main = "Comparison cases mean for each countries",
        args.legend = list("topleft"))
```



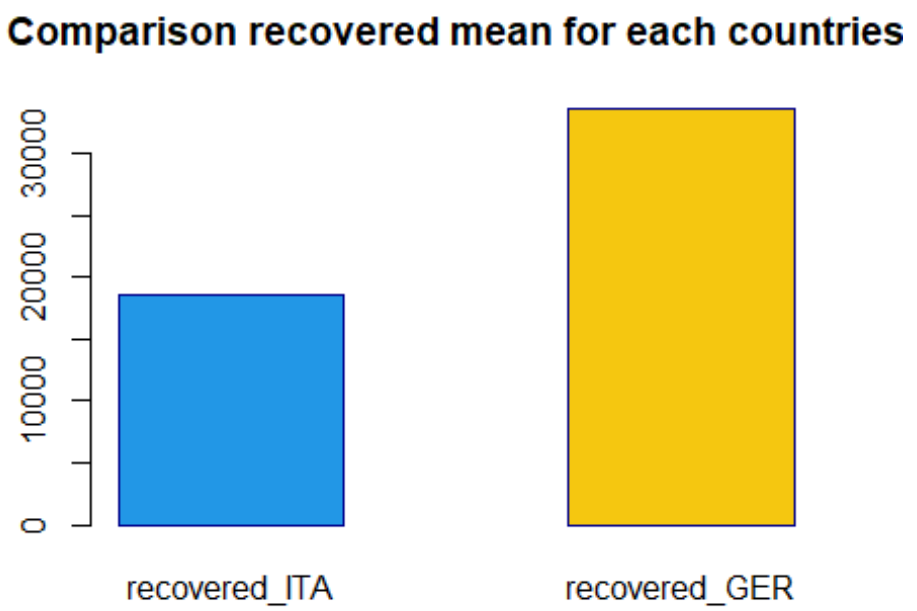
We can see that France is the country with highest cases mean with nearly 40000 cases per day, it is ahead of Germany and Italy.

```
barplot(height = cbind(deaths_ITA =
descriptive_stats_Global["deaths_ITA","mean"] ,deaths_FR =
descriptive_stats_Global["deaths_FR","mean"],deaths_GER =
descriptive_stats_Global["deaths_GER","mean"]),
        beside = TRUE,
        col = c(4,2,7),
        legend.text = TRUE,
        border = "dark blue",
        main = "Comparison deaths mean for each countries",
        args.legend = list(x = "topleft"))
```



We can see that the number of deaths in mean is nearly the same for each country. Let's study "recovered" :

```
barplot(height = cbind(recovered_ITA =
descriptive_stats_Global["recovered_ITA","mean"] ,recovered_GER =
descriptive_stats_Global["recovered_GER","mean"]),
        beside = TRUE,
        col = c(4,7),
        legend.text = TRUE,
        border = "dark blue",
        main = "Comparison recovered mean for each countries",
        args.legend = list(x = "topleft"))
```



We only have Italy and Germany data but we can clearly see that the number of recovered is higher on Germany. For both it seems logical because it follows the value of confirmed cases in mean.

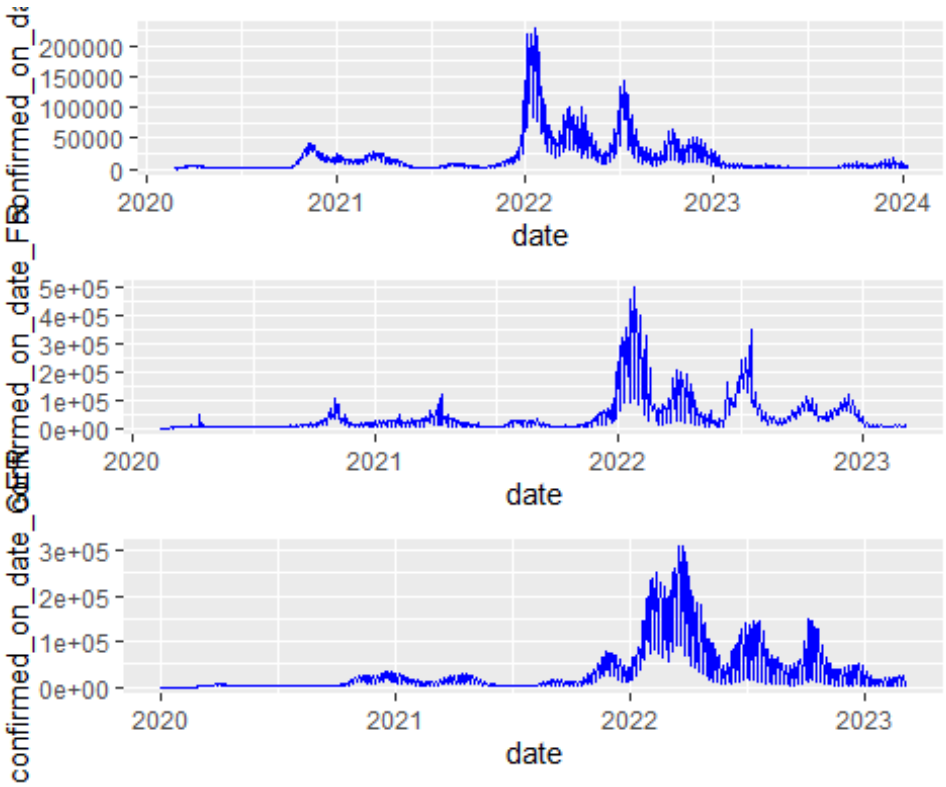
We can show the letality of covid19 :

```
c(lethality_ITA,lethality_FR,lethality_GER)
```

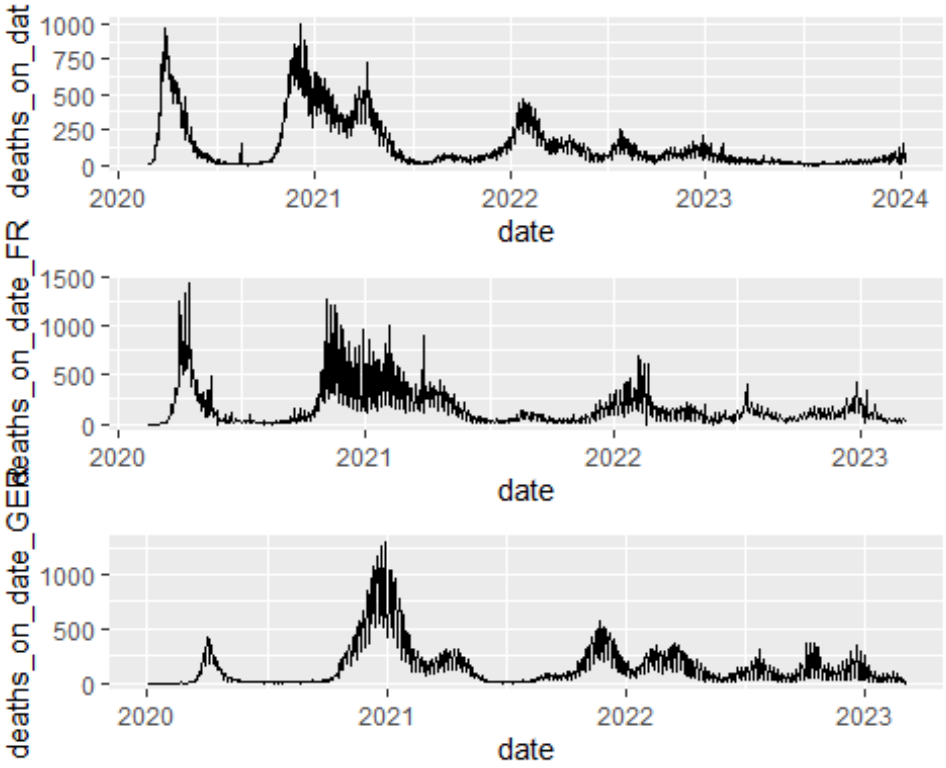
```
## [1] 0.7325907 0.4064014 0.4410666
```

In Italy we have the highest probability to die if we contract covid19. We compare the representation for each countries :

```
IT=ggplot(sample_covid19_ITA)+geom_line(mapping =
aes(x=date,y=confirmed_on_date), color = "blue")
FR=ggplot(sample_covid19_FR)+geom_line(mapping =
aes(x=date,y=confirmed_on_date_FR), color = "blue")
GE=ggplot(sample_covid19_GER)+geom_line(mapping =
aes(x=date,y=confirmed_on_date_GER), color = "blue")
grid.arrange(IT,FR,GE)
```




```
IT=ggplot(sample_covid19_ITA)+geom_line(mapping =
aes(x=date,y=deaths_on_date), color = "black")
FR=ggplot(sample_covid19_FR)+geom_line(mapping =
aes(x=date,y=deaths_on_date_FR), color = "black")
GE=ggplot(sample_covid19_GER)+geom_line(mapping =
aes(x=date,y=deaths_on_date_GER), color = "black")
grid.arrange(IT,FR,GE)
```



We can see that the distribution for each variable is fairly close, even though we don't have the same number of data points and so there is a small lag, so we can easily recognize the same pattern with the different waves. Focusing on "deaths", we can see for all of them the first wave at the beginning of 2020, then a large valley and the second wave at the end of 2020, which is the most deadly wave. For all, we see a gradual decrease and a new series of small waves.

Communication of Results:

- Create visualizations and summary reports to effectively communicate key findings.

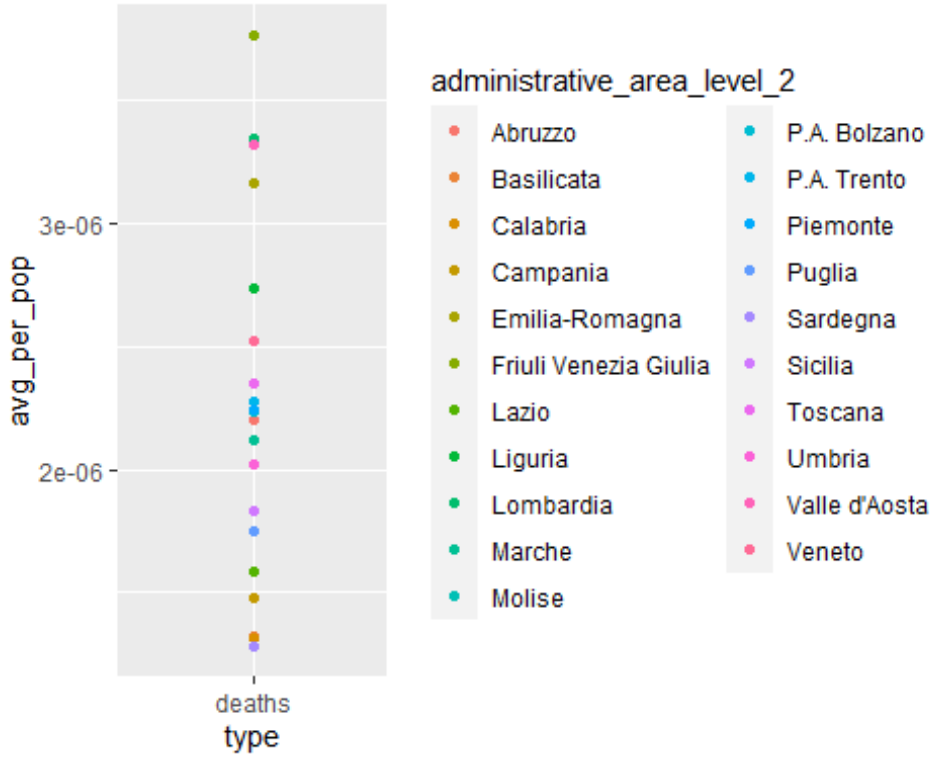
Key findings :

1) First we focus our analysis on Italy covid19 data frame :

We studied the data of Italy, we compute several statistics as min, max, mean, standard deviation etc that we have stored in a new data frame. We apply the same logical for the regions in Italy and again we have stored the statistics in a data frame. These data allows us to give for example the good student and the bad student.

- Sardegna is the region with lowest number of deaths in mean by population.
- Friuli Venezia is the reverse.

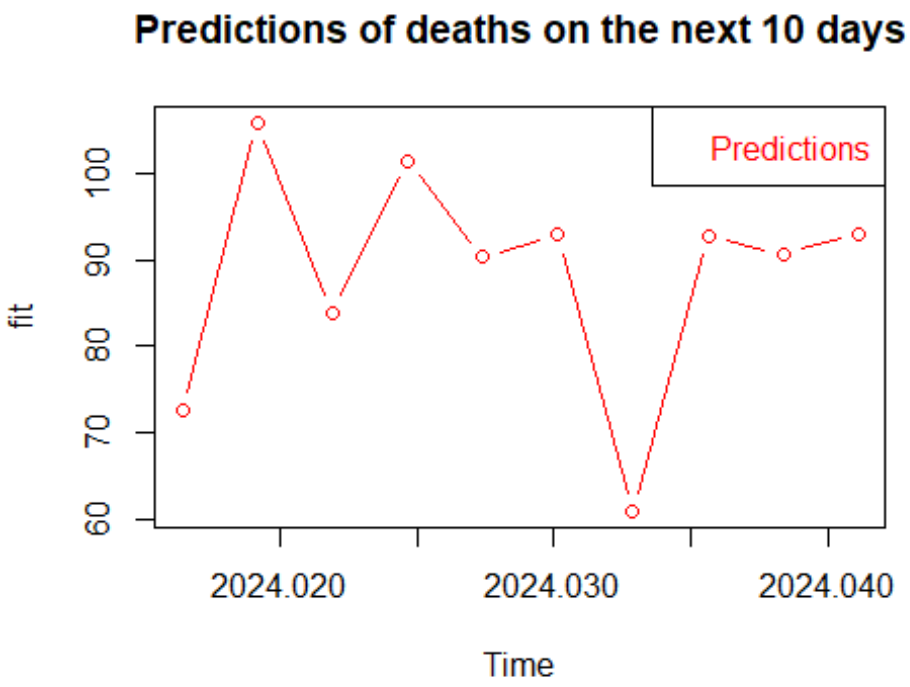
```
ggplot(filter(descriptive_stats_ITA_reg,type=="deaths"))+geom_point(mapping =
aes(x=type,y=avg_per_pop,color=administrative_area_level_2))
```



2) Temporal trends on Italy:

Then we apply time series to modeling our data. Especially we apply a exponential smoothing with Holt-Winters and we make predictions on the next 10 days :

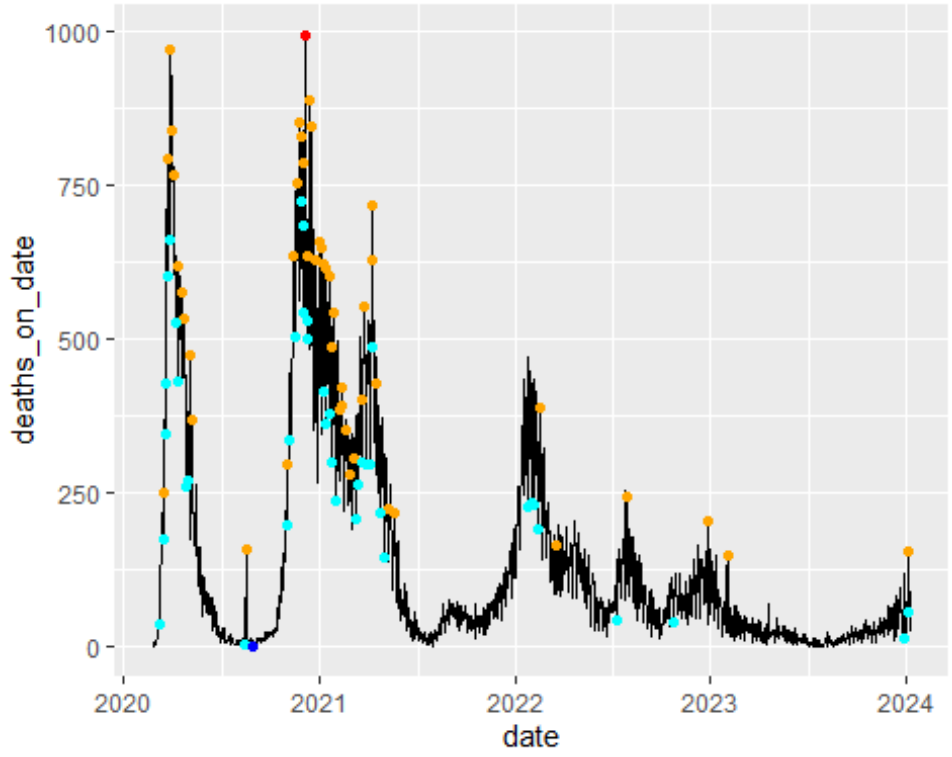
```
plot(HW_ts_deaths_ITA_pred,type="b",main="Predictions of deaths on the next
10 days",col="red")
legend("topright",legend = "Predictions",text.col = "red")
```



In this case we have shown that for 10 next days the number of deaths will oscillate in range [60,110] in Italy.

We were also interested of the variations in the data. For example, for the "deaths" we have plot the peaks and the valleys (min/max local) using a dispersion measure (standard deviation, mediane absolute deviation) to show where are the period with a lot of variability in the data.

```
ggplot(sample_covid19_ITA,aes(date,deaths_on_date))+geom_line(color="black")+
geom_point(data = ~ .[peaks,], color = "orange")+geom_point(data = ~
.[valleys,], color =
"cyan")+geom_point(aes(x=date[i_max],y=max(deaths_on_date)),color="red")+geom
_point(aes(x=date[i_min],y=min(deaths_on_date)),color="blue")
```



3) Comparison with others countries :

We realized the same principle in terms preprocessing, data analysis, computation of statistics, visualization for France and Germany. We have build a data frame which conbine all the statistics per country. Using this we have seen :

- France has the highest number of cases in mean.
- Italy had a less number of cases but his number of deaths in mean is nearly the same than others.
- The lethality of covid19 is higher in Italy (0.73) .(0.4-0.45) in France and Germany.

All the statistics we have calculated, especially the descriptive ones, can be considered as an indication, a trend, but not as an exact value that clearly represents the situation, because there is a different number of samples for each country, we don't know exactly how the data is recorded, and there is also a huge uncertainty linked to the data processing we had to do.