Lesson 1 - Cordero - The 6th of February, 2023

Luca Visentin

2023-02-06

Course Outline

- We will start with some R basics, some bioinformatics basics, and then some modelling.
- People familiar with R can just solve the exercises, without having to follow along.
- The exam will be a small report of all these lessons, with perhaps a small input from each of us.

There's a Moodle website for this course.

We don't need to install R studio or R as we can use the hosted versions provided by the department. People who have the R/Rstudio combo already in their laptops can just use their local version. I'm User13. I've saved the login info on a `secrets.txt` file that is not committed.

The machines will remain online for the duration of the course.

Covid19

We will analyze Covid-19 data. There is a `COVID19` package to install, which contains a large data frame with many variables regarding Covid-19 infections.

The granularity of the dataset is country-level plus the Grand Princess cruise ship (and other large ships), that was studied in particular. For each country, there is one entry per day.

There are videos on Moodle for people who do not know R, recorded by the professors to speed things along.

Follow the `.pdf` for today's exercises on this data frame.

Introduction

We will start by analyzing some Covid-19 data.

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0      v purrr   1.0.0
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.5.0
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
covid <- COVID19::covid19(start="2019-01-01", verbose = FALSE)

pprint <- function(...) {
  str <- paste0(...)
```

```
  cat(str)
}
```

Let's start with some exploratory data description:

There are many variables in this dataset. For instance: - `date`: When the measure was taken; - `confirmed`: The number of confirmed cases of covid19, cumulative with previous days; - `deaths`: The number of confirmed deaths due to covid19, cumulative with previous days; - `administrative_area_level`: These variables refer to the country or region that they refer to.

We will select just a subset of the variables of interest to work with:

```
covid |>
  select(
    c(
      "date", # The day the measure was taken
      "confirmed", # N. of confirmed cases, cumulative
      "deaths", # N. of confirmed deaths due to covid19, cumulative
      "recovered", # N. of recovered people from covid19, cumulative
      "tests", # N. of tests performed, cumulative
      "people_fully_vaccinated", # N. of fully vaccinated people against covid19, cumulative
      "administrative_area_level_1", # Country of referral
      "population" # Population of the country of interest (at some point? Latest census?)
    )
  ) %>%
  rename(country = administrative_area_level_1) -> covid
```

We can see the normal summary:

```
summary(covid)
```

```
##       date              confirmed             deaths          recovered
##  Min.   :2020-01-01   Min.   :        0   Min.   :      0   Min.   :       0
##  1st Qu.:2020-10-28   1st Qu.:     5125   1st Qu.:    101   1st Qu.:    3242
##  Median :2021-07-30   Median :    43280   Median :    833   Median :   30809
##  Mean   :2021-07-29   Mean   :  1254797   Mean   :  18740   Mean   :  938102
##  3rd Qu.:2022-04-28   3rd Qu.:   389758   3rd Qu.:   6953   3rd Qu.:  263710
##  Max.   :2023-02-14   Max.   :102598932   Max.   :1120904   Max.   :37545675
##                       NA's   :20110       NA's   :37191     NA's   :178702
##      tests            people_fully_vaccinated   country
##  Min.   :         0   Min.   :0.000e+00       Length:251573
##  1st Qu.:    342331   1st Qu.:2.059e+05       Class :character
##  Median :   1990500   Median :2.251e+06       Mode  :character
##  Mean   :  22654351   Mean   :2.202e+07
##  3rd Qu.:  10965456   3rd Qu.:9.973e+06
##  Max.   :9214000000   Max.   :1.277e+09
##  NA's   :    165227   NA's   :193281
##    population
##  Min.   :5.000e+01
##  1st Qu.:4.846e+05
##  Median :5.516e+06
##  Mean   :3.340e+07
##  3rd Qu.:1.975e+07
##  Max.   :1.393e+09
##  NA's   :1118
```

```r
# Number of days that are in the dataset
pprint("There are ", nrow(covid), " days in the dataset.\n\n")
```

```
## There are 251573 days in the dataset.
```

```r
# Number of countries (with the Grand Princess as an extra)
pprint("There are ", length(unique(covid$country)), " unique countries: ", paste0(sort(unique(covid$cou
```

```
## There are 236 unique countries: Afghanistan, Albania, Algeria, American Samoa, Andorra, Angola, Angu
```

```r
# There's the Holy See too!
```

Through sorting, we can see the countries with the most number of cases, overall. This would be nice to be normalized by the number of people per country, as larger countries may tend to have more infected people.

Reporting practices also need to be taken into account. Places like North Korea probably reported a lot less cases than what actually happened.

Additionally, the normalization assumes that people may only get infected once. So it makes not too much sense when you take into account re-infections. In any case, it's a rough way to get a sense of how well a country has responded to the pandemic.

For vaccination data, the rate of "full" vaccination changes based on what the definition of "full" vaccination is.

```r
# Add the normalized values for confirmed cases and vaccinated people
covid$norm_confirmed <- covid$confirmed / covid$population
covid$norm_vaccinated <- covid$people_fully_vaccinated / covid$population

# The number of cases can only grow, so we can get just one (recent) in the
# dataset, and use that as the max value for the country
today <- covid[covid$date == "2023-02-01",] # @ the first of february

extract_stat <- function(data, title, interest_col, n = 10, round = TRUE) {
  data[order(data[[interest_col]], decreasing = TRUE), c("country", interest_col)] %>%
    head(n) -> data

  if (round) {
    data[[interest_col]] <- round(data[[interest_col]], 3)
  }

  str <- paste(data$country, data[[interest_col]], sep = ": ") %>% paste0(collapse = ", ")

  pprint(title, str, "\n\n")
}

extract_stat(today, "The top 10 countries with most cases overall: ", "confirmed")
```

```
## The top 10 countries with most cases overall: United States: 102179838, China: 98527660, India: 4468
```

```r
extract_stat(today, "The top 10 countries with most percentage of cases: ", "norm_confirmed")
```

```
## The top 10 countries with most percentage of cases: Cook Islands: 0.816, Faroe Islands: 0.715, San Ma
```

```r
# Same as above, but with countries of at least 1 million in population
big_today <- filter(today, population > 1e6)

extract_stat(big_today, "The top 10 big countries with most cases overall: ", "confirmed")
```

```
## The top 10 big countries with most cases overall: United States: 102179838, China: 98527660, India: 4
```

```
extract_stat(big_today, "The top 10 big countries with most percentage of cases: ", "norm_confirmed")
```

```
## The top 10 big countries with most percentage of cases: Austria: 0.655, Denmark: 0.587, Korea, South
```

```
# Similarly, vaccinations can only grow over time
extract_stat(today, "The top 10 countries with most overall vaccinations: ", "people_fully_vaccinated")
```

```
## The top 10 countries with most overall vaccinations: India: 951722099, United States: 229785560, Braz
```

```
extract_stat(today, "The top 10 countries with most vaccination rate: ", "norm_vaccinated")
```

```
## The top 10 countries with most vaccination rate: Hong Kong: 0.912, Cuba: 0.883, Taiwan: 0.877, Malays
```

```
extract_stat(big_today, "The top 10 big countries with most overall vaccinations: ", "people_fully_vacci
```

```
## The top 10 big countries with most overall vaccinations: India: 951722099, United States: 229785560,
```

```
extract_stat(big_today, "The top 10 big countries with most vaccination rate: ", "norm_vaccinated")
```

```
## The top 10 big countries with most vaccination rate: Hong Kong: 0.912, Cuba: 0.883, Taiwan: 0.877, Ma
```

The first country to start the vaccination policy was...

```
# It's not super easy to find. The record may not start at 1 for each country, and some countries
# may have started vaccinating on the same day.

# Let's transform the data: for each country, we will save the first day that the
# vaccination vector is not zero.

get_first_vaccination_day <- function(data) {
  countries <- unique(data$country)

  res <- list()
  for (country in countries) {
    cdata <- data[data$country == country,]
    cdata <- cdata[! is.na(cdata$people_fully_vaccinated), ]
    # When the data should be 0, it is signed as NA
    res[[country]] <- cdata$date[order(cdata$date)][1]
  }

  res
}

get_first_vaccination_day(covid) %>% unlist() %>% as.Date(lubridate::origin) %>% sort() %>% head(10)
```

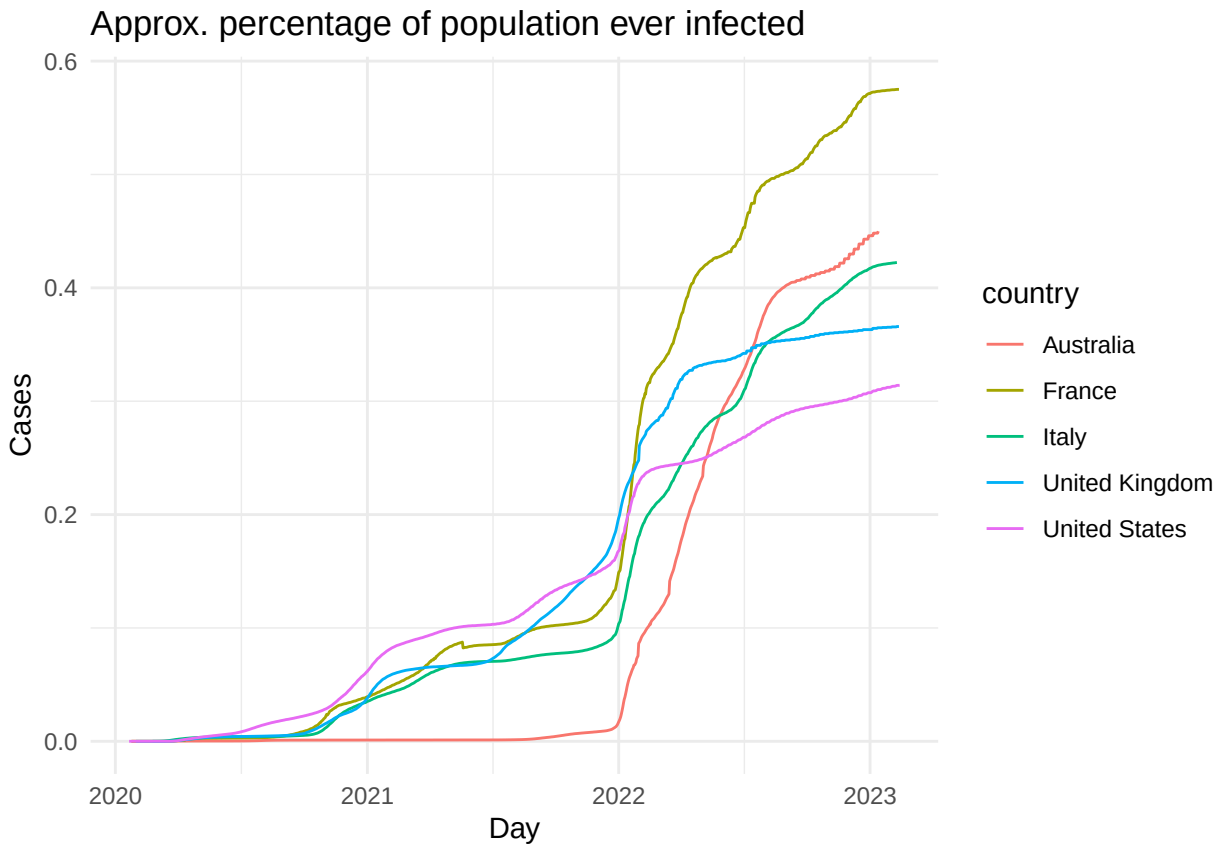
```
##     Lithuania          Peru United States        Israel   Switzerland
##  "2020-03-20"  "2020-04-27"  "2020-12-13"  "2020-12-19"  "2020-12-21"
## Liechtenstein        Canada         Chile       Austria    Martinique
##  "2020-12-21"  "2020-12-22"  "2020-12-24"  "2020-12-27"  "2020-12-27"
```

```
# Lithuania and Peru? How is that possible?
```

Select Italy and 5 countries of your interest. Then, plot with `ggplot2` the ratio of cases to the population over time.

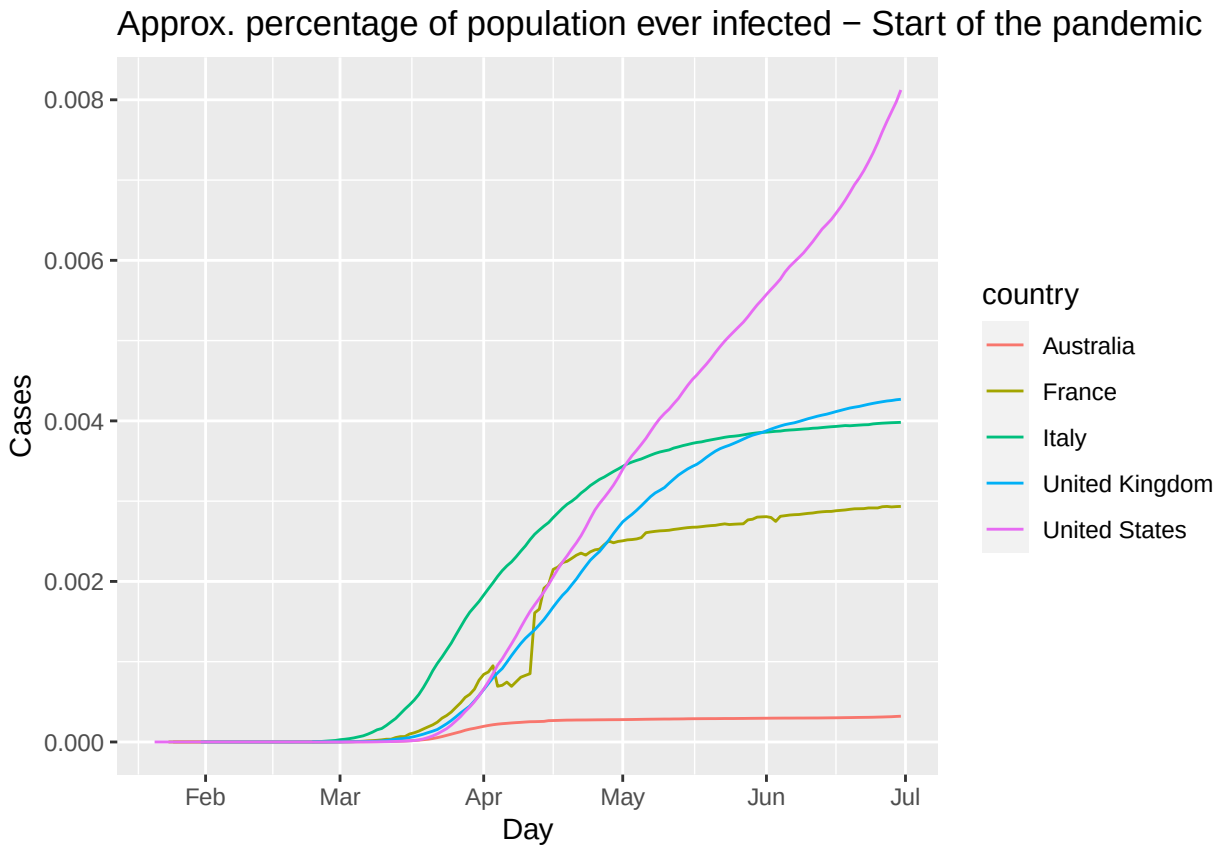
```
countries_of_interest = c("Italy", "United States", "Australia", "United Kingdom", "France")

covid %>% filter(country %in% countries_of_interest) -> plot_covid
```

```
plot_covid %>% drop_na("date", "norm_confirmed") %>%
  ggplot(aes(x = date, y = norm_confirmed, color = country)) +
  ggtitle("Approx. percentage of population ever infected") +
  xlab("Day") + ylab("Cases") +
  theme_minimal() +
  geom_line()
```



Let's see just the start of the pandemic, from January 2020 to June 2020:

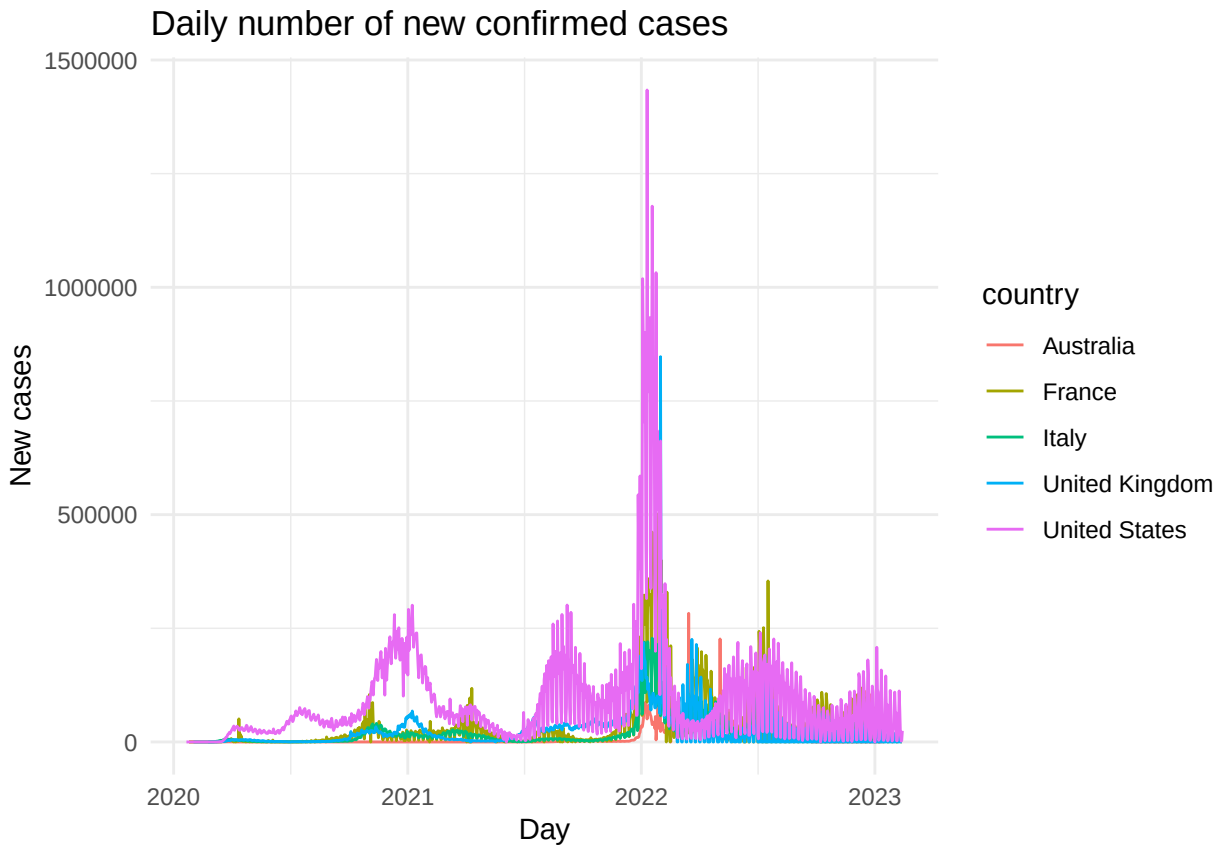
```
plot_covid %>% drop_na("date", "norm_confirmed") %>%
  filter(date >= "2020-01-01" & date < "2020-07-01") %>%
  ggplot(aes(x = date, y = norm_confirmed, color = country)) +
  ggtitle("Approx. percentage of population ever infected - Start of the pandemic") +
  xlab("Day") + ylab("Cases") +
  geom_line()
```



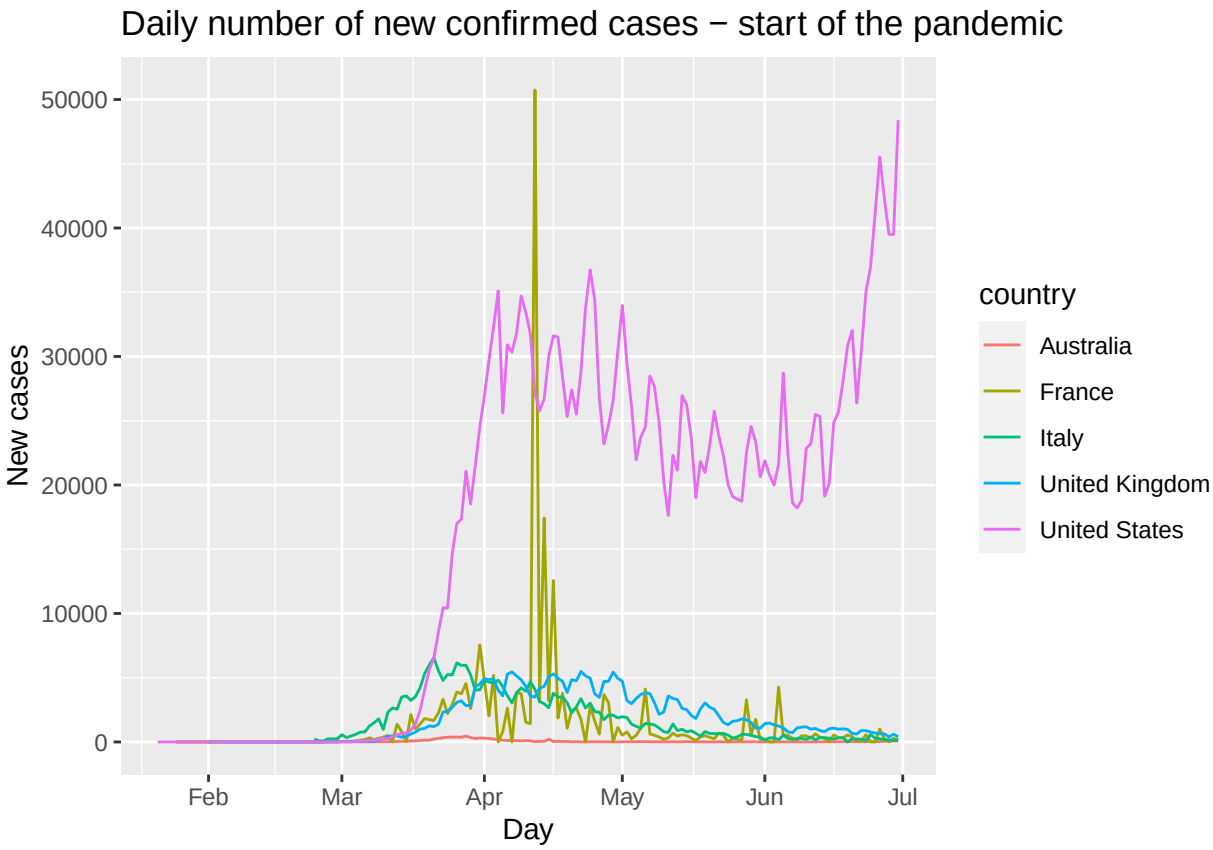
We can see the very early bump that happened in Italy at the start of the pandemic.

We can also plot the number of new cases, but we first need to compute them:

```
covid %>%
  group_by(country) %>%
  mutate(new_cases = c(confirmed[1], diff(confirmed))) -> covid

# There might be less than zero due to recounts. We just set the recounts
# to zero new cases
covid$new_cases[covid$new_cases < 0] <- 0

# Need to regen the plot_covid dataset
covid %>% filter(country %in% countries_of_interest) -> plot_covid

plot_covid %>% drop_na("date", "new_cases") %>%
  ggplot(aes(x = date, y = new_cases, color = country)) +
  ggtitle("Daily number of new confirmed cases") +
  xlab("Day") + ylab("New cases") +
  theme_minimal() +
  geom_line()
```




```
plot_covid %>% drop_na("date", "new_cases") %>%
  filter(date >= "2020-01-01" & date < "2020-07-01") %>%
  ggplot(aes(x = date, y = new_cases, color = country)) +
  ggtitle("Daily number of new confirmed cases - start of the pandemic") +
  xlab("Day") + ylab("New cases") +
  geom_line()
```



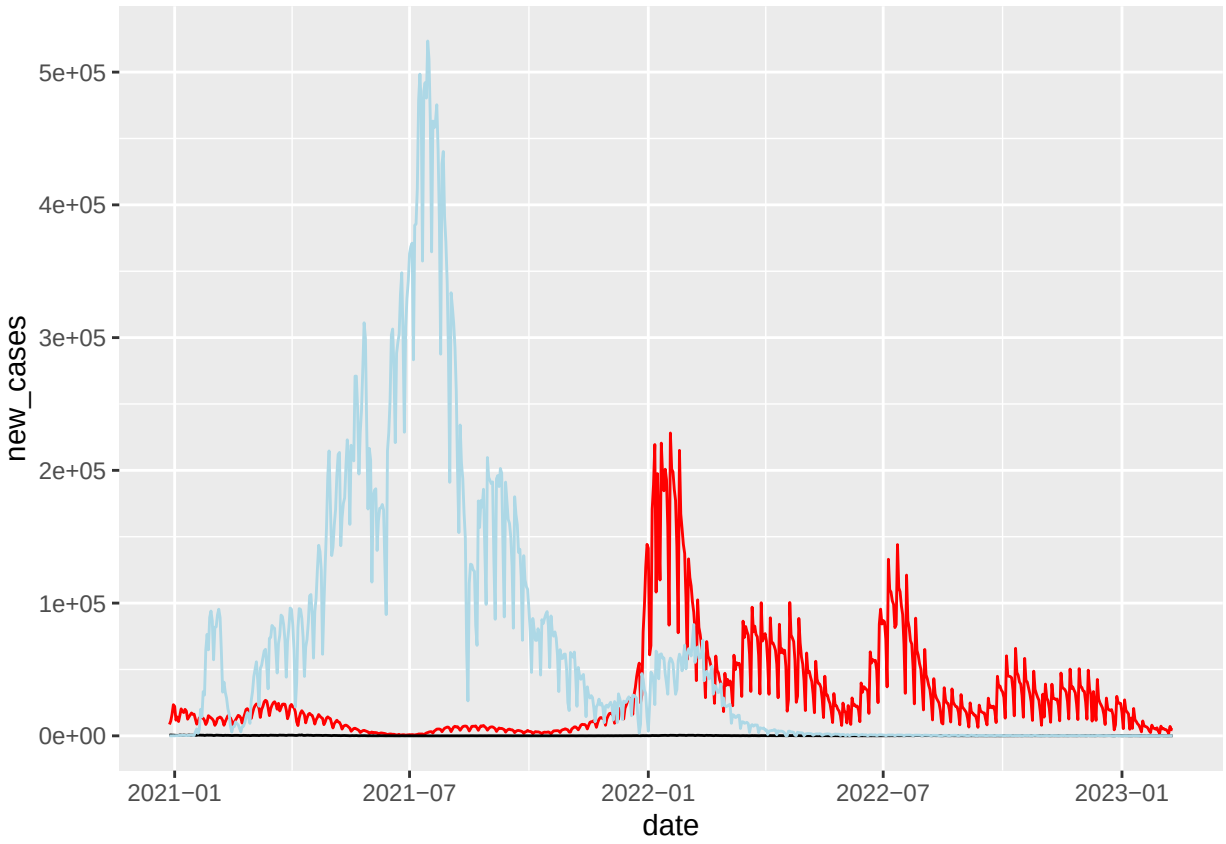
See how there are spikes in the reporting? We might be able to smooth them out by taking a window-average of the data, but it's a bit outside of the scope.

I have accidentally done the next step, of calculating the daily cases. I'll expand it by also calculating the number of daily deaths, and the daily number of new vaccinations:

```
covid %>%
  group_by(country) %>%
  arrange(date, .by_group = TRUE) %>%
  mutate(
    new_vaccinations = c(people_fully_vaccinated[1], diff(people_fully_vaccinated)),
    new_deaths = c(deaths[1], diff(deaths))
  ) -> covid
```

To show the data in the clearest manner we can try to see a plot:

```
covid %>%
  filter(country == "Italy") %>%
  drop_na("new_cases", "new_deaths", "new_vaccinations") %>%
  ggplot(aes(x = date)) +
  geom_line(aes(y = new_cases), color = "red") +
  geom_line(aes(y = new_deaths), color = "black") +
  geom_line(aes(y = new_vaccinations), color = "lightblue")
```



There are a few issues with the above plot: - The data is very spiky; - The number of deaths (thank god) are on a very different scale than the number of new cases and vaccinations.

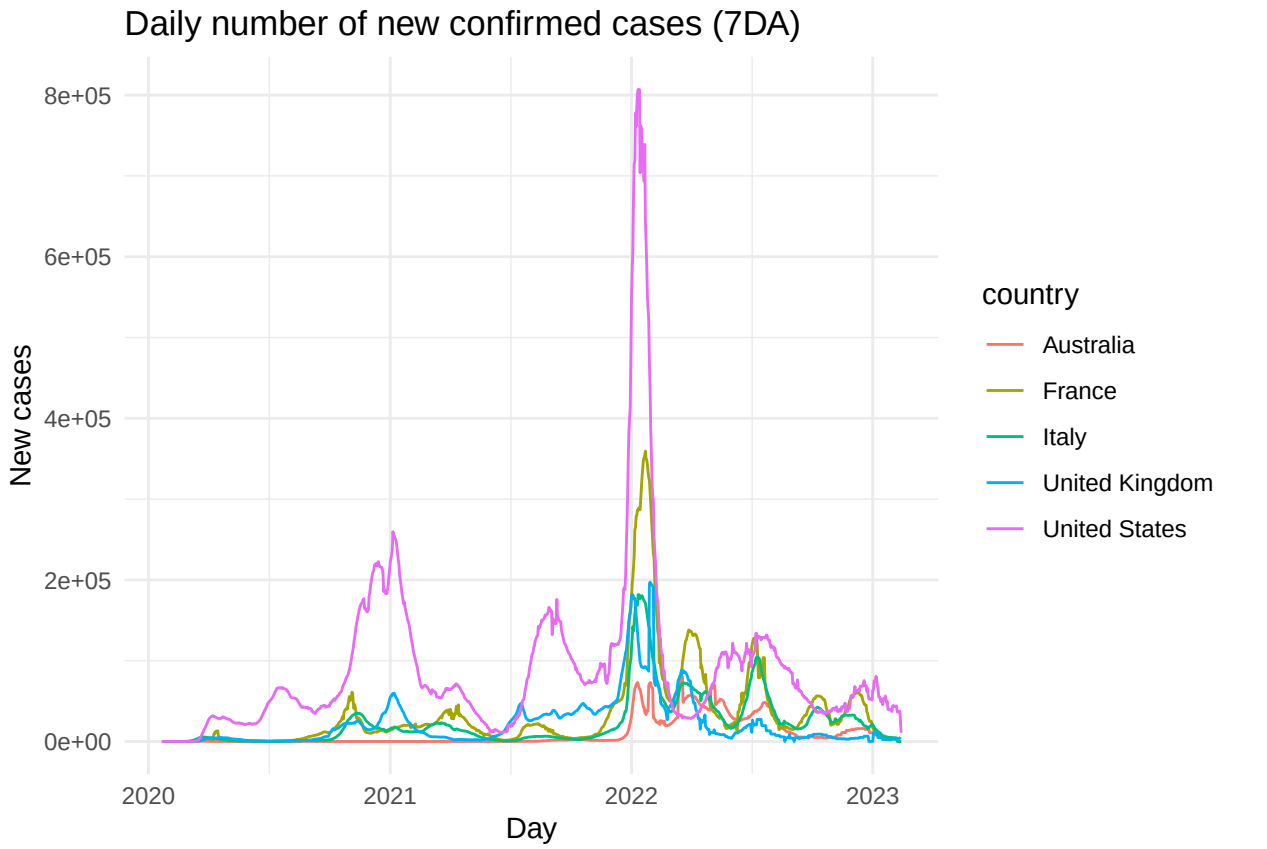
Let's address the above considerations. I really need to run a rolling average. I guess a weekly average is ok:

```
# For this you need to have the library "zoo" installed
rolling_average <- function(x, window = 7) {
  zoo::rollapply(x, width = window, FUN = mean, partial = TRUE, align = "center")
}

covid %>%
  group_by(country) %>%
  arrange(date, .by_group = TRUE) %>%
  mutate(
    smooth_new_cases = rolling_average(new_cases),
    smooth_new_deaths = rolling_average(new_deaths),
    smooth_new_vaccinations = rolling_average(new_vaccinations)
  ) -> covid
```

We can now see if it worked with some plotting:

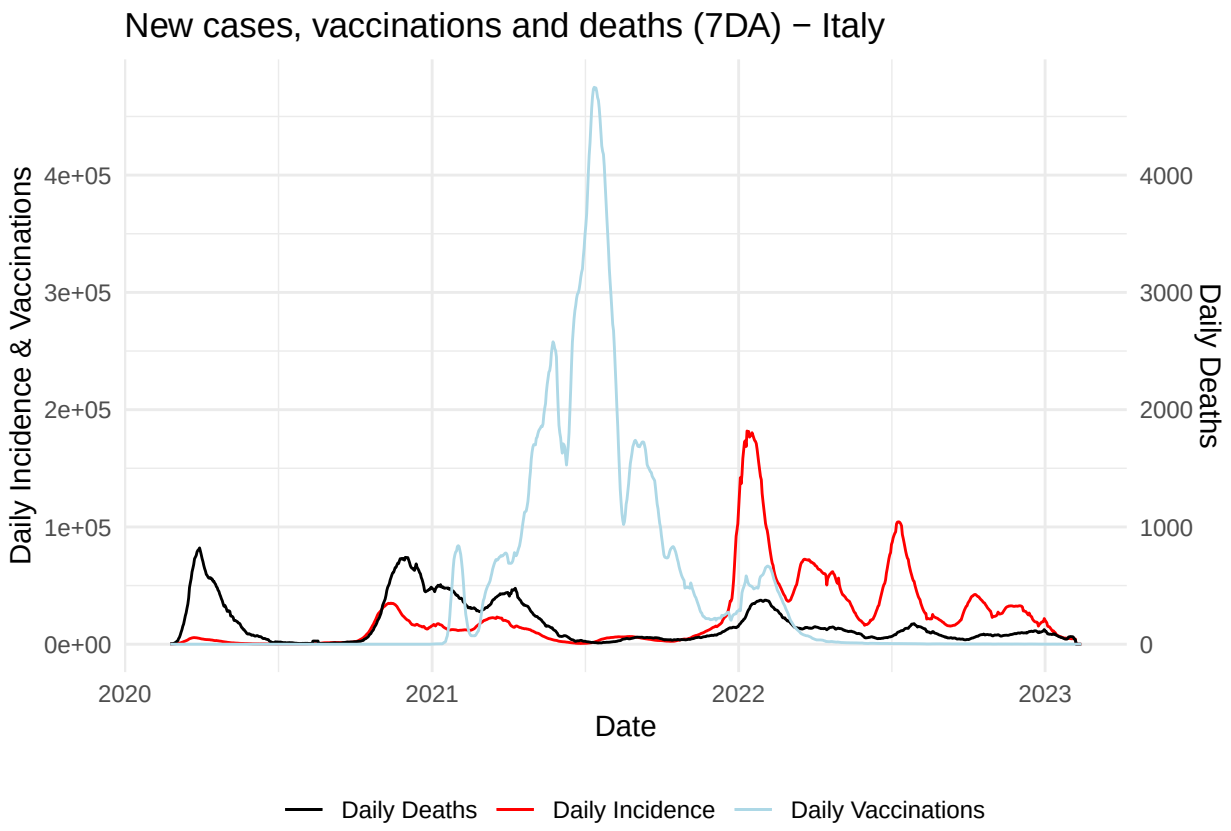
```
# Need to regen the plot_covid dataset
covid %>% filter(country %in% countries_of_interest) -> plot_covid

plot_covid %>% replace_na(list("smooth_new_cases" = 0)) %>%
  ggplot(aes(x = date, y = smooth_new_cases, color = country)) +
  ggtitle("Daily number of new confirmed cases (7DA)") +
  xlab("Day") + ylab("New cases") +
  theme_minimal() +
```

```
  geom_line()
```



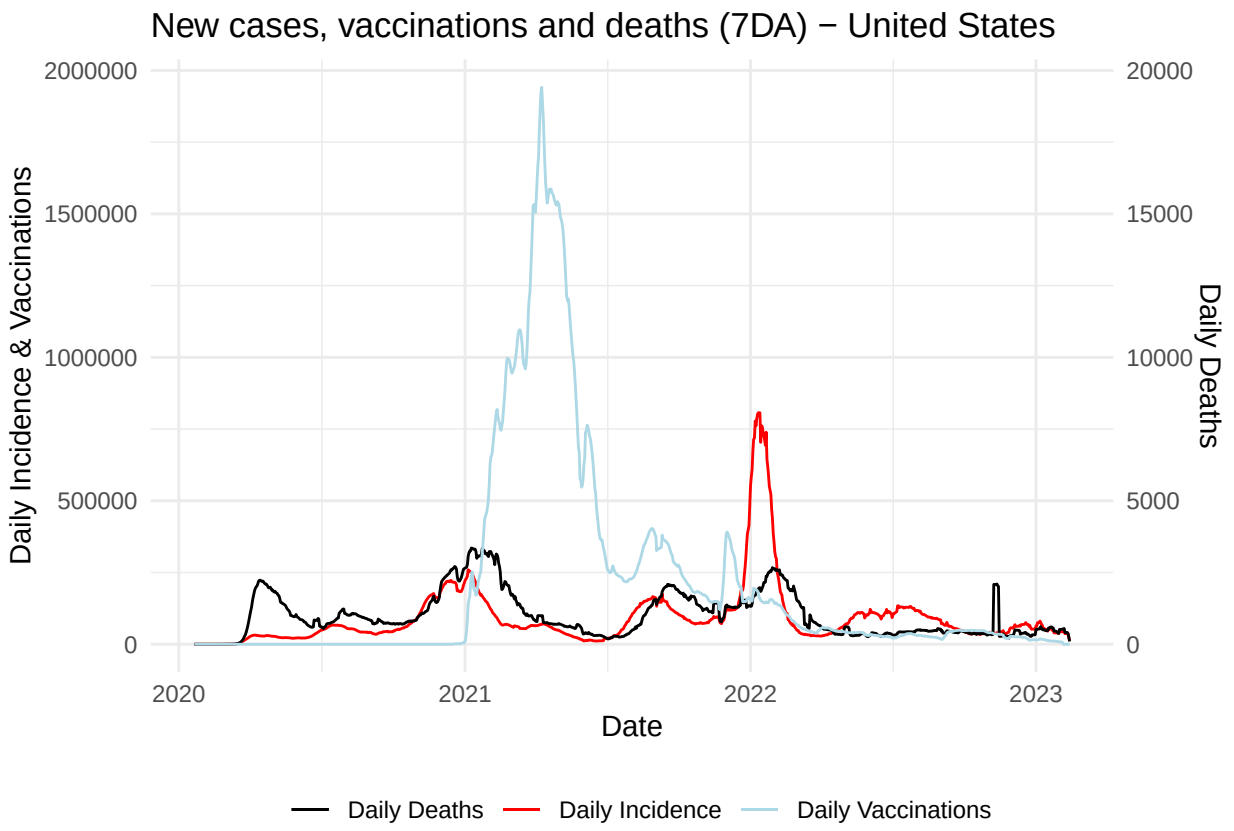
Smooth like butter! Going back to the actual exercise:

```
plot_covid_trend <- function(covid, country_name) {
  covid %>%
    filter(country == country_name) %>%
    replace_na(list(smooth_new_cases = 0, smooth_new_deaths = 0, smooth_new_vaccinations = 0)) %>%
    ggplot(aes(x = date)) +
    geom_line(aes(y = smooth_new_cases, color = "Daily Incidence")) +
    geom_line(aes(y = smooth_new_deaths * 100, color = "Daily Deaths")) +
    geom_line(aes(y = smooth_new_vaccinations, color = "Daily Vaccinations")) +
    scale_y_continuous(
      name = "Daily Incidence & Vaccinations", sec.axis = sec_axis(trans = ~./ 100, name = "Daily Deaths
    ) +
    scale_color_manual(values=c("black", "red", "lightblue")) +
    xlab("Date") +
    ggtitle(paste0("New cases, vaccinations and deaths (7DA) - ", country_name)) +
    theme_minimal() +
    theme(
      legend.position = "bottom", legend.title = element_blank()
    ) -> p

  p
}

plot_covid_trend(covid, "Italy")
```

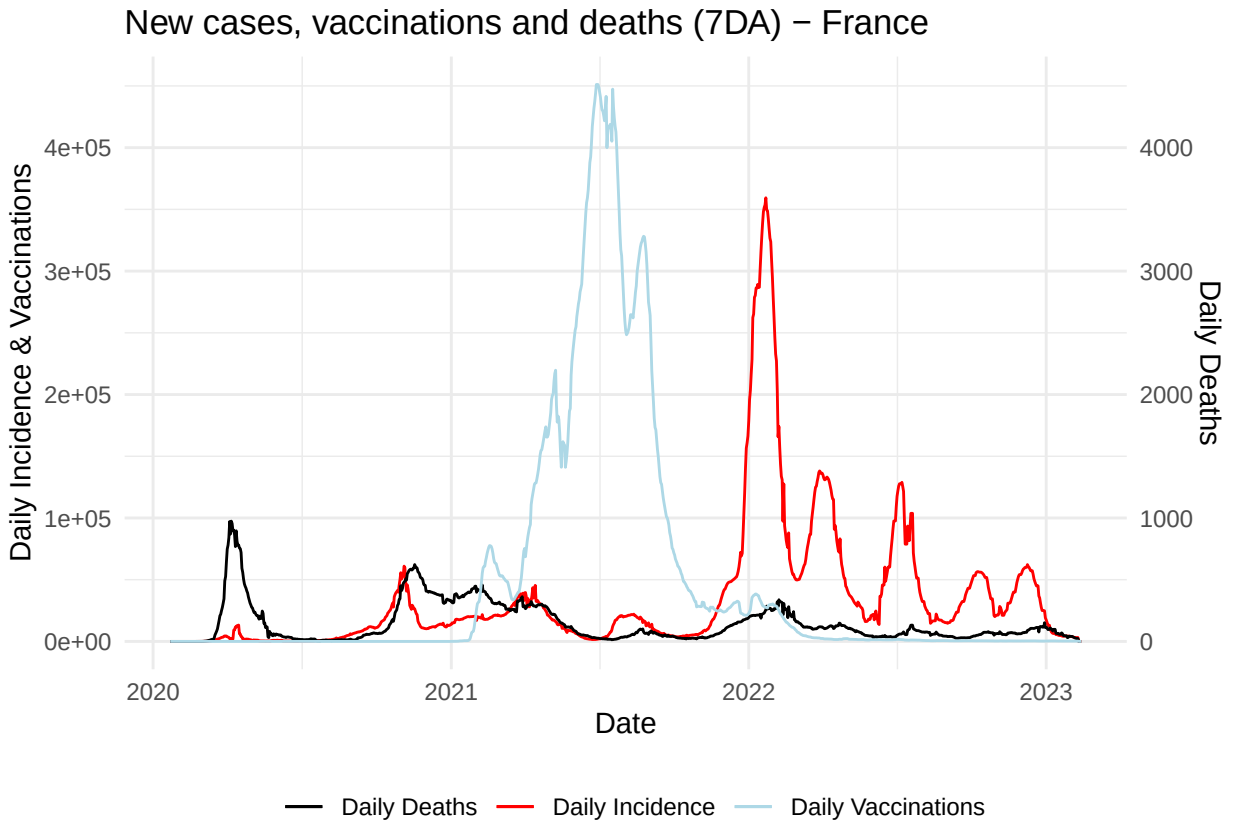


The effect of the vaccination is dramatic!

```
plot_covid_trend(covid, "United States")
```

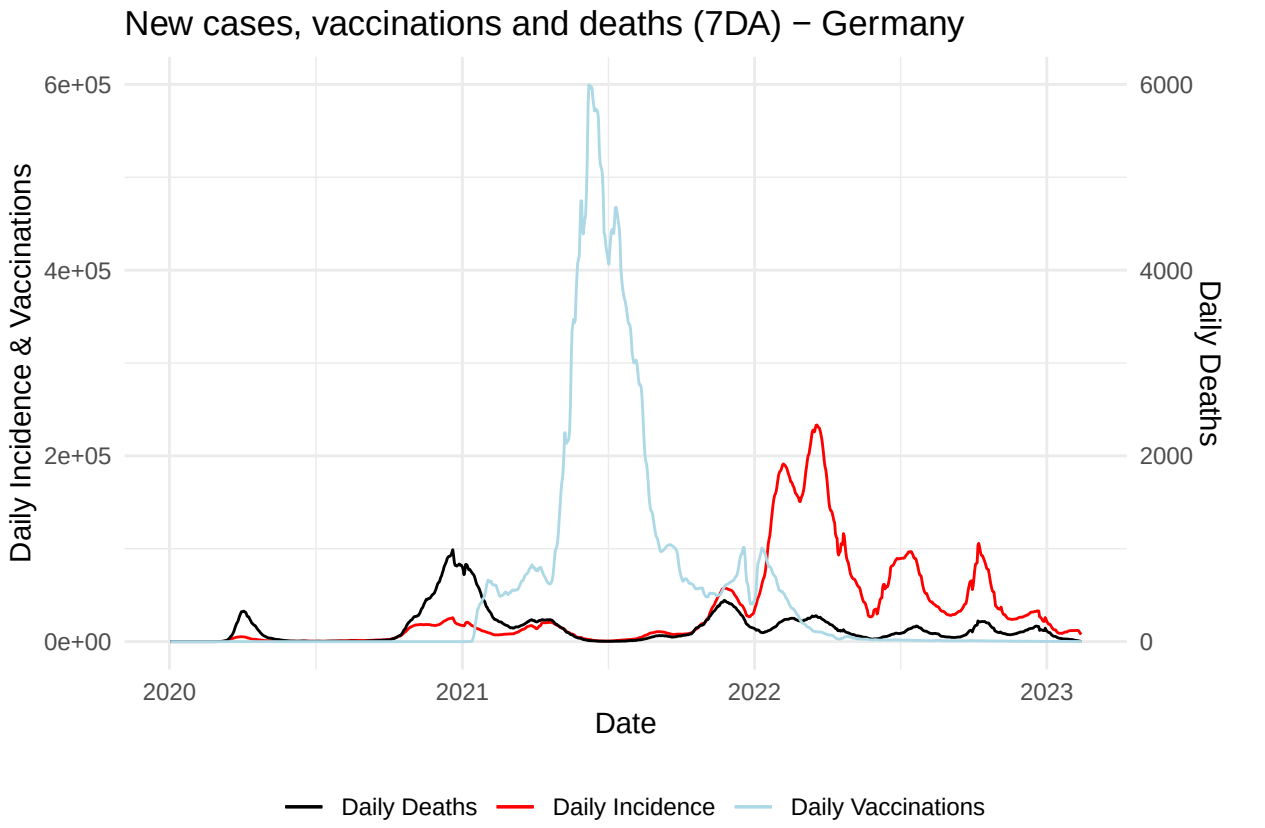



```
plot_covid_trend(covid, "France")
```

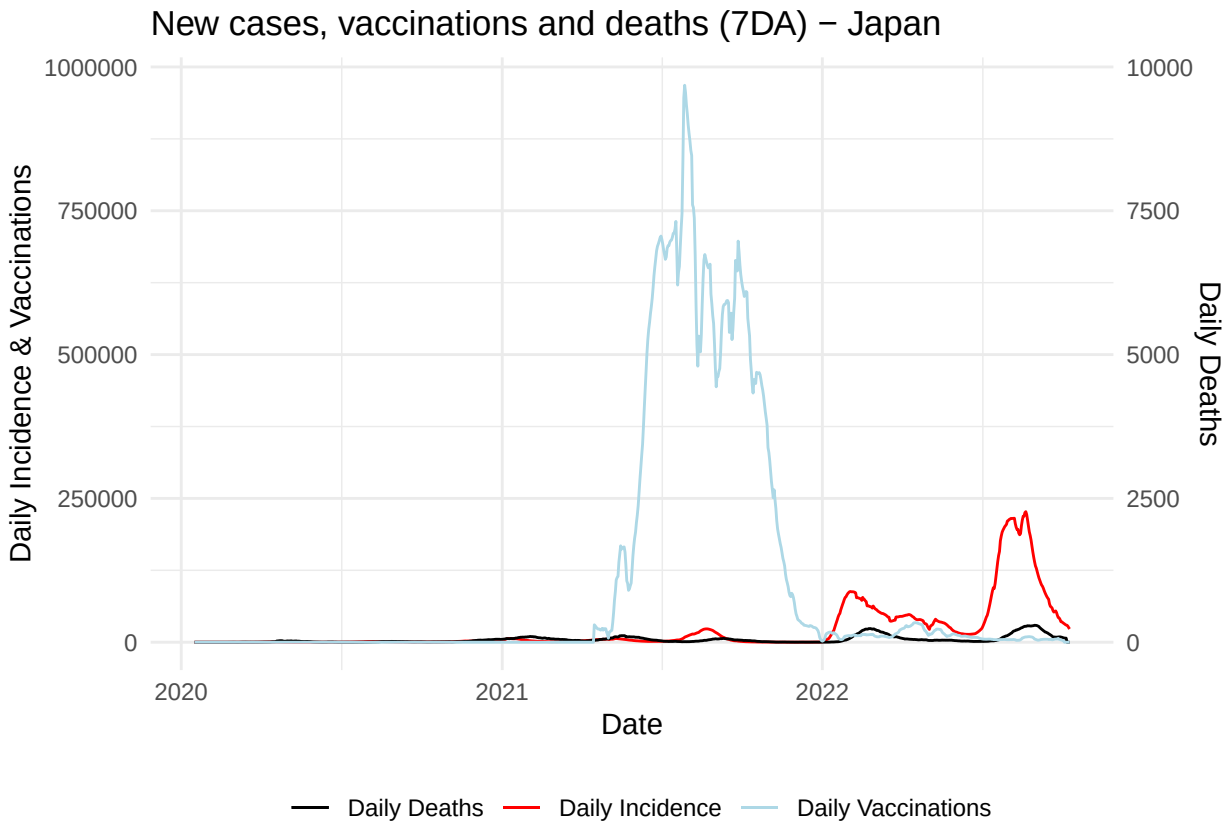


France is very similar to Italy in terms of these trends.

```
plot_covid_trend(covid, "Germany")
```

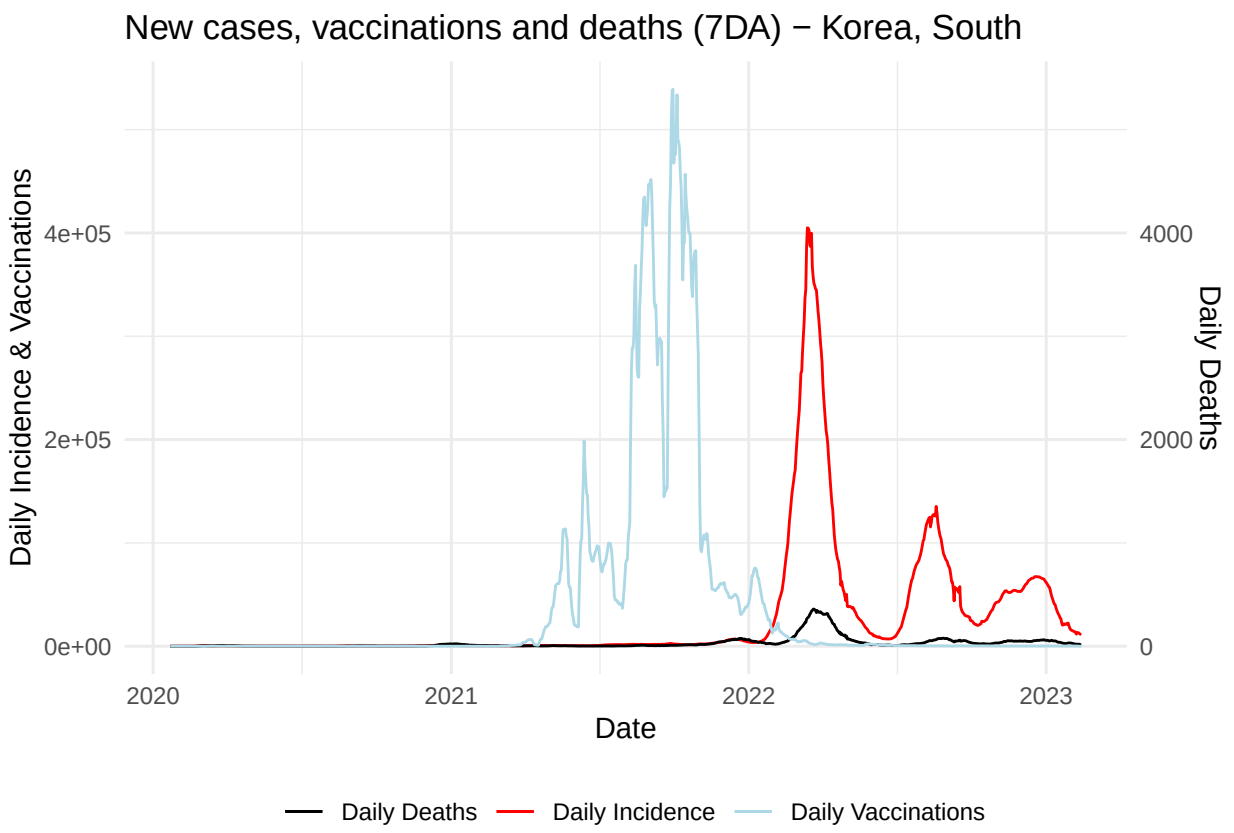



```
plot_covid_trend(covid, "Japan")
```

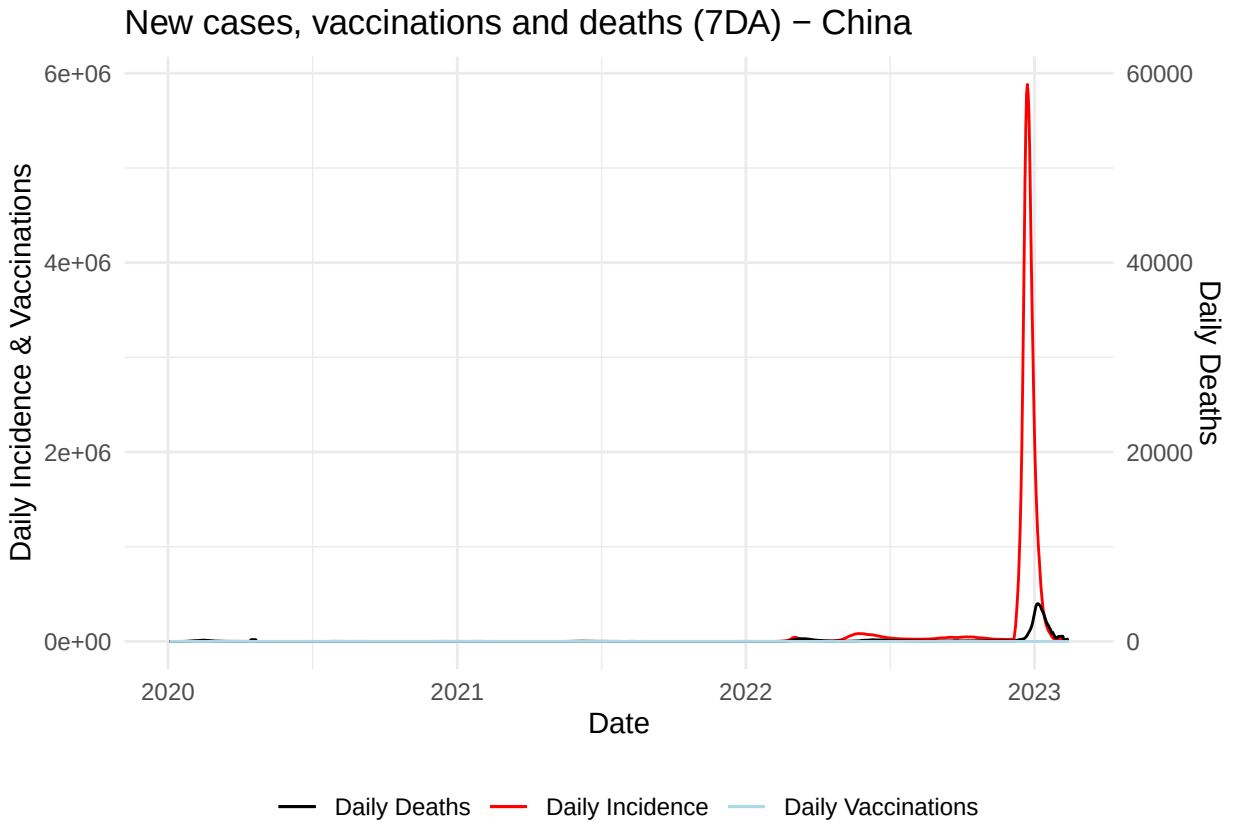


Japan was very successful in preventing the worse of the pandemic.

```
plot_covid_trend(covid, "Korea, South")
```

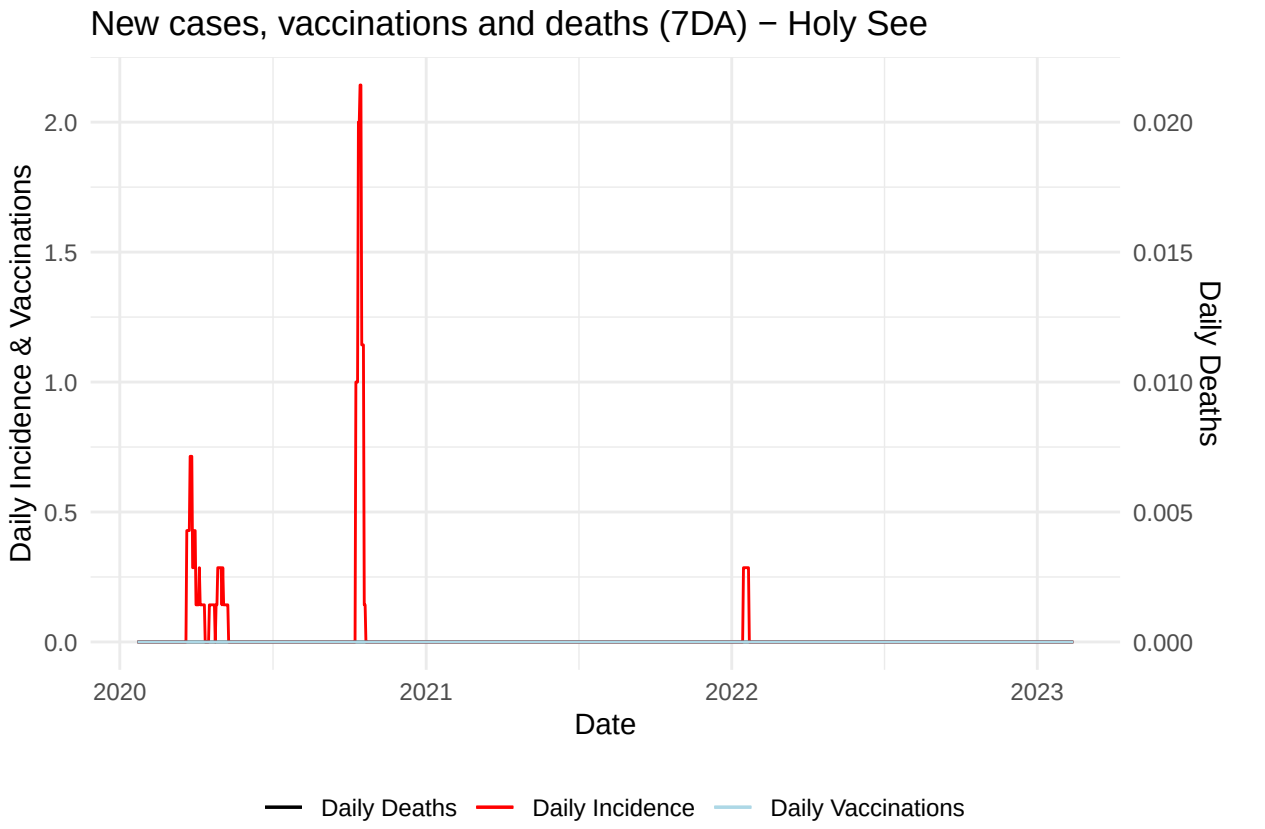



```
plot_covid_trend(covid, "China")
```



There was literally no reporting of what happened in China.

```
# For laughs
plot_covid_trend(covid, "Holy See")
```



TODO

> Considering the previous dataframe composed of number of daily cases daily deaths divide the data BEFORE and AFTER the start day of the vaccination campaign. Visualize the evolution of the pandemic. Compute the appropriate statistical test to compare BEFORE versus AFTER data.

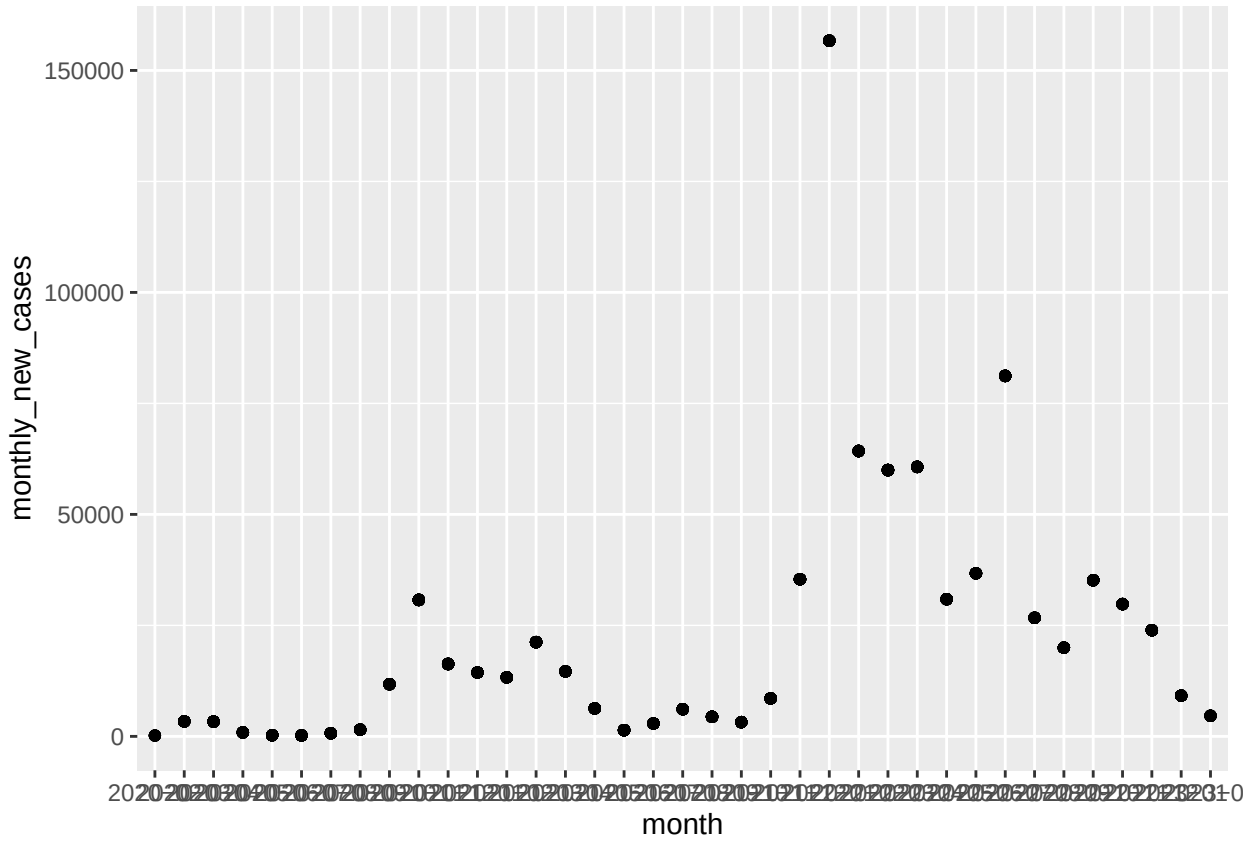
The visualization of the pandemic trend is the same as above, where we see the peak of vaccination rates and then a general drop of the mortality of the disease.

Finding a time-series statistical test is hard. The points are autocorrelated with their predecessors, more with the day directly before, slightly less with the day before, and so on. There are special tests that one can run to handle time series but I guess they are out of scope. We will simply see if there is a difference in the average mortality rate (e.g. number of deaths / number of cases) in various bins of data (e.g. every month), considering as if the bins are not correlated with one another.

```
# First, we need to create the binned data for every time-period that we choose
covid %>%
  group_by(country) %>%
  mutate(month = format(date, "%Y-%m")) %>%
  group_by(month, .add = TRUE) %>%
  mutate(
    monthly_new_cases = na_if(mean(new_cases, na.rm = TRUE), NaN),
    monthly_new_deaths = na_if(mean(new_deaths, na.rm = TRUE), NaN),
    monthly_new_vaccinations = na_if(mean(new_vaccinations, na.rm = TRUE), NaN)
```

```
  ) -> covid
```

```
covid %>%
  filter(country == "Italy") %>%
  ggplot(aes(x = month, y = monthly_new_cases)) +
  geom_point()
```



Starting for the original dataset: (i) select the country in which the vaccination policy started earlier (ii) the country in which the vaccination policy started the latest. Compare the daily trend of the cases with a suitable plot and comment on the results achieved.