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.

> Install and load the R COVID19 package. Provide a description of the data. In particular: 1. Number of countries; 2. Number of observations per country, 3. the countries with the higher number of confirmed cases and people fully vaccinated and 4. the first country starting the vaccination policy.

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0     v purrr   1.0.1
## v tibble  3.1.8     v dplyr   1.1.0
## v tidyr   1.3.0     v stringr 1.5.0
## v readr   2.1.3     v forcats 1.0.0
## -- 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(
       # The day the measure was taken
      "date",
       # N. of confirmed cases, cumulative
      "confirmed",
      # N. of confirmed deaths due to covid19, cumulative
      "deaths",
      # N. of recovered people from covid19, cumulative
      "recovered",
      # N. of tests performed, cumulative
      "tests",
      # N. of fully vaccinated people against covid19, cumulative
      "people_fully_vaccinated",
      # Country of referral
      "administrative_area_level_1",
      # Population of the country of interest (at some point? Latest census?)
      "population"
    )
  ) %>%
  rename(country = administrative_area_level_1) -> covid
```

We can see the standard summary of the data (we are considering cruise ships as countries for all intents and purposes...):

```
summary(covid)
```

```
##       date              confirmed             deaths          recovered
##  Min.   :2020-01-01   Min.   :        0   Min.   :      0   Min.   :       0
##  1st Qu.:2020-10-28   1st Qu.:     5133   1st Qu.:    101   1st Qu.:    3244
##  Median :2021-07-31   Median :    43364   Median :    833   Median :   30849
##  Mean   :2021-07-30   Mean   :  1256029   Mean   :  18746   Mean   :  938585
##  3rd Qu.:2022-04-29   3rd Qu.:   389935   3rd Qu.:   6957   3rd Qu.:  263734
##  Max.   :2023-02-15   Max.   :102626386   Max.   :1121294   Max.   :37545675
##                       NA's   :20125       NA's   :37210     NA's   :178835
##      tests             people_fully_vaccinated   country
##  Min.   :         0   Min.   :0.000e+00       Length:251710
##  1st Qu.:    342427   1st Qu.:2.057e+05       Class :character
```

```
##  Median :   1991072   Median :2.251e+06       Mode  :character
##  Mean   :  22659170   Mean   :2.201e+07
##  3rd Qu.:  10966972   3rd Qu.:9.973e+06
##  Max.   :9214000000   Max.   :1.277e+09
##  NA's   :    165351   NA's   :193373
##    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   :1120
```

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

```
## There are 251710 days in the dataset.
```

```
# Per-country number of observations
# (not really, the number of dates where we have new cases)
covid %>%
  group_by(country) %>%
  summarize(length(na.omit(confirmed))) %>%
  print(n = 10)
```

```
## # A tibble: 236 x 2
##    country             `length(na.omit(confirmed))`
##    <chr>                                      <int>
##  1 Afghanistan                                 1086
##  2 Albania                                     1073
##  3 Algeria                                      955
##  4 American Samoa                               511
##  5 Andorra                                     1079
##  6 Angola                                      1062
##  7 Anguilla                                    1053
##  8 Antigua and Barbuda                         1069
##  9 Argentina                                   1050
## 10 Armenia                                     1081
## # ... with 226 more rows
```

```
# Number of countries (with the Grand Princess & Co. as an extra)
pprint(
  "There are ", length(unique(covid$country)), " unique countries: ",
  paste0(sort(unique(covid$country)), collapse=", ")
)
```

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

```
# You can't see it in the PDF correctly, but all the countries are listed here.
# This was to choose the countries of interest later (and copy-paste them).
# 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.

```
# 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)
  }

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

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

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

```
## The top 10 countries with most cases overall:
## United States: 102179838,
## China: 98527660,
## India: 44684248,
## France: 38487384,
## Germany: 37801461,
## Brazil: 36837943,
## Korea, South: 30213928,
## Italy: 25482610,
## United Kingdom: 24274357,
## Russia: 21649414
```

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

```
## The top 10 countries with most percentage of cases:
## Cook Islands: 0.816,
## Faroe Islands: 0.715,
## San Marino: 0.693,
## Austria: 0.655,
```

```
## Brunei: 0.644,
## Andorra: 0.621,
## Martinique: 0.61,
## Gibraltar: 0.605,
## Falkland Islands (Malvinas): 0.604,
## Réunion: 0.603
```

```
# 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:\n",
  "confirmed"
)
```

```
## The top 10 big countries with most cases overall:
## United States: 102179838,
## China: 98527660,
## India: 44684248,
## France: 38487384,
## Germany: 37801461,
## Brazil: 36837943,
## Korea, South: 30213928,
## Italy: 25482610,
## United Kingdom: 24274357,
## Russia: 21649414
```

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

```
## The top 10 big countries with most percentage of cases:
## Austria: 0.655,
## Denmark: 0.587,
## Korea, South: 0.585,
## France: 0.575,
## Cyprus: 0.54,
## Israel: 0.539,
## Greece: 0.517,
## Latvia: 0.506,
## Slovakia: 0.489,
## Georgia: 0.487
```

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

```
## The top 10 countries with most overall vaccinations:
## India: 951722099,
## United States: 229785560,
## Brazil: 175525337,
## Germany: 63555902,
## France: 53746289,
## Italy: 50008330,
```

```
## Korea, South: 44428570,
## Spain: 40727618,
## Canada: 31762055,
## Malaysia: 27538649
```

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

```
## The top 10 countries with most vaccination rate:
## Hong Kong: 0.912,
## Cuba: 0.883,
## Taiwan: 0.877,
## Malaysia: 0.873,
## Spain: 0.87,
## Korea, South: 0.861,
## Canada: 0.857,
## New Zealand: 0.855,
## Uruguay: 0.84,
## Brazil: 0.838
```

```
extract_stat(
  big_today, "The top 10 big countries with most overall vaccinations:\n",
  "people_fully_vaccinated"
  )
```

```
## The top 10 big countries with most overall vaccinations:
## India: 951722099,
## United States: 229785560,
## Brazil: 175525337,
## Germany: 63555902,
## France: 53746289,
## Italy: 50008330,
## Korea, South: 44428570,
## Spain: 40727618,
## Canada: 31762055,
## Malaysia: 27538649
```

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

```
## The top 10 big countries with most vaccination rate:
## Hong Kong: 0.912,
## Cuba: 0.883,
## Taiwan: 0.877,
## Malaysia: 0.873,
## Spain: 0.87,
## Korea, South: 0.861,
## Canada: 0.857,
## New Zealand: 0.855,
## Uruguay: 0.84,
## Brazil: 0.838
```

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

```r
# 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$country == country,]
    indexes <- which(cdata$people_fully_vaccinated > 0) # Get the index of the True-s

    # This if is just to avoid running `min` on an empty vector, which
    # makes noisy warnings
    if (length(indexes) == 0) {
      res[[country]] <- NA
    } else {
      res[[country]] <- cdata$date[min(indexes)]
    }
  }

  res
}

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

```
##          Peru United States        Israel   Switzerland        Canada
##  "2020-05-08"  "2020-12-13"  "2020-12-19"  "2020-12-21"  "2020-12-22"
##        France         Italy       Estonia        Latvia       Ireland
##  "2020-12-27"  "2020-12-27"  "2020-12-28"  "2020-12-28"  "2020-12-28"
```

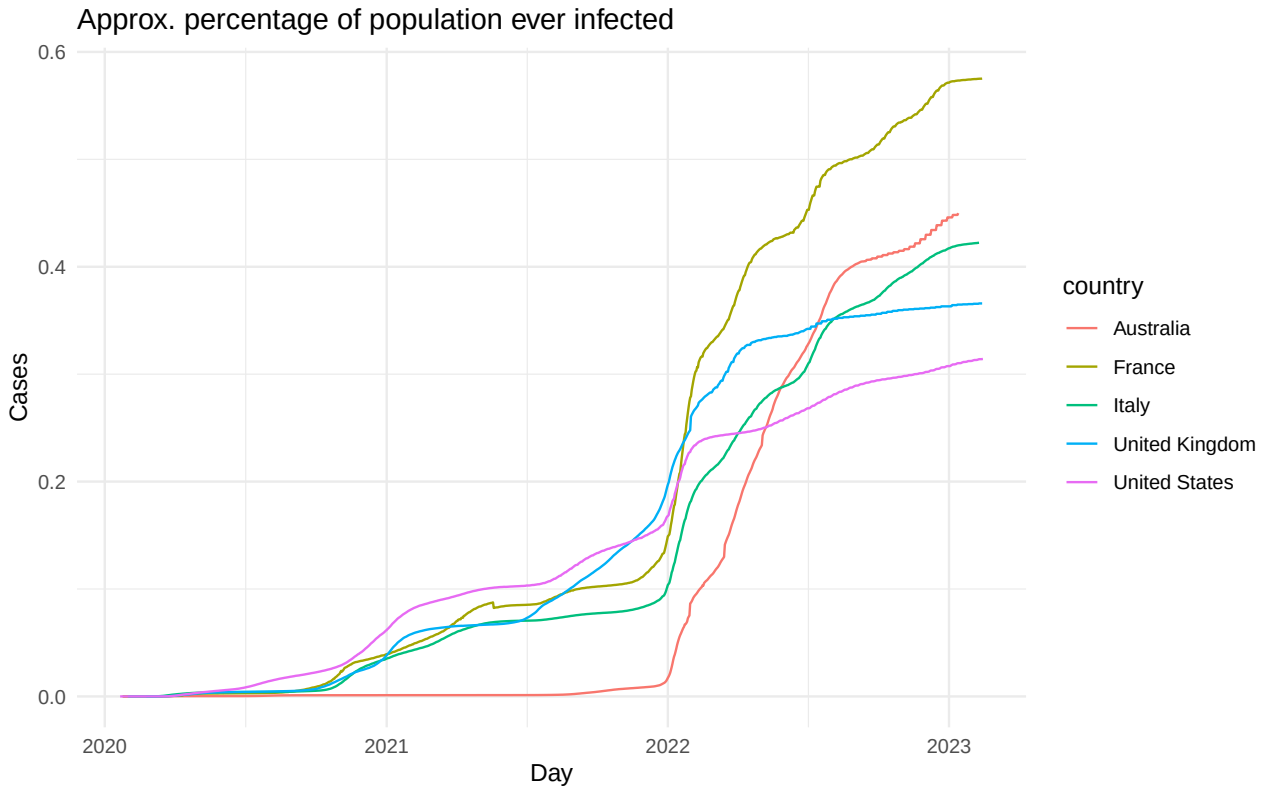
```r
# Peru? In May of 2020? There's got to be something wrong...

# Further examination shows a rogue "1" in a sea of NAs. It must be an error.
```

We can say the the first country to vaccinate was the United States, closely followed by Israel, Switzerland, Canada, France and Italy.

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

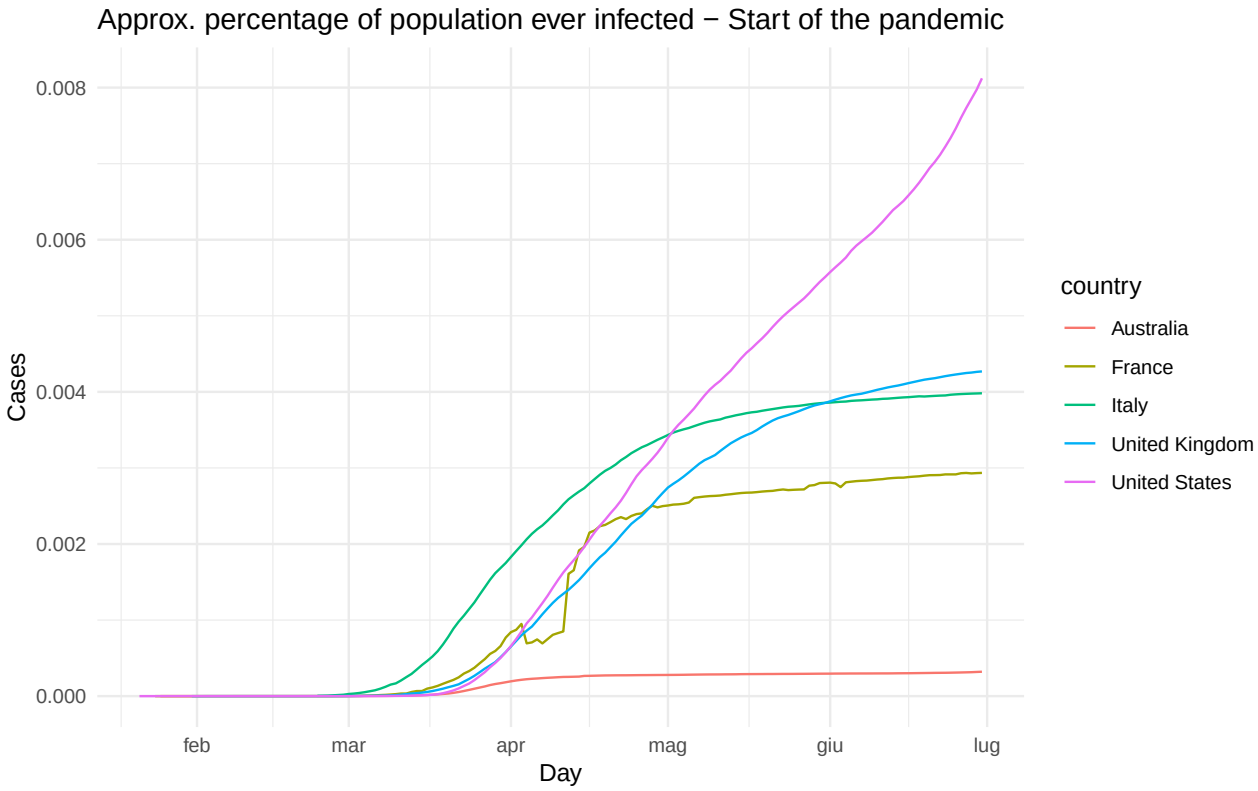
```r
countries_of_interest <- c(
  "Italy", "United States", "Australia", "United Kingdom", "France"
)

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()
```



The later number of cases are so large that the start of the pandemic is a bit hard to see. Let's see just this period, from January 2020 to June 2020:

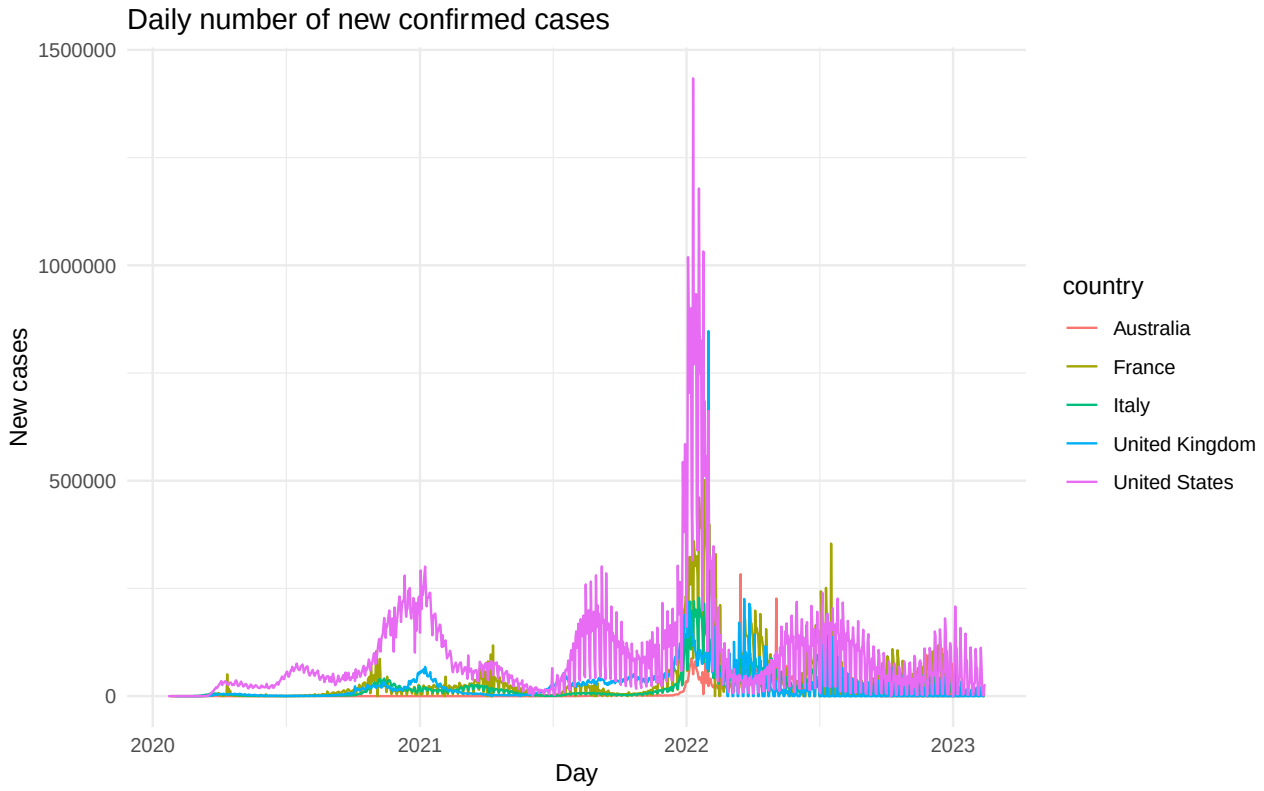
```
plot_covid %>% drop_na("date", "norm_confirmed") %>%
  filter(date >= "2020-01-01" & date < "2020-07-01") %>%
  ggplot(aes(x = date, y = norm_confirmed, color = country)) +
  ggtitle("Approx. percentage of population ever infected - Start of the pandemic") +
  xlab("Day") + ylab("Cases") +
  theme_minimal()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



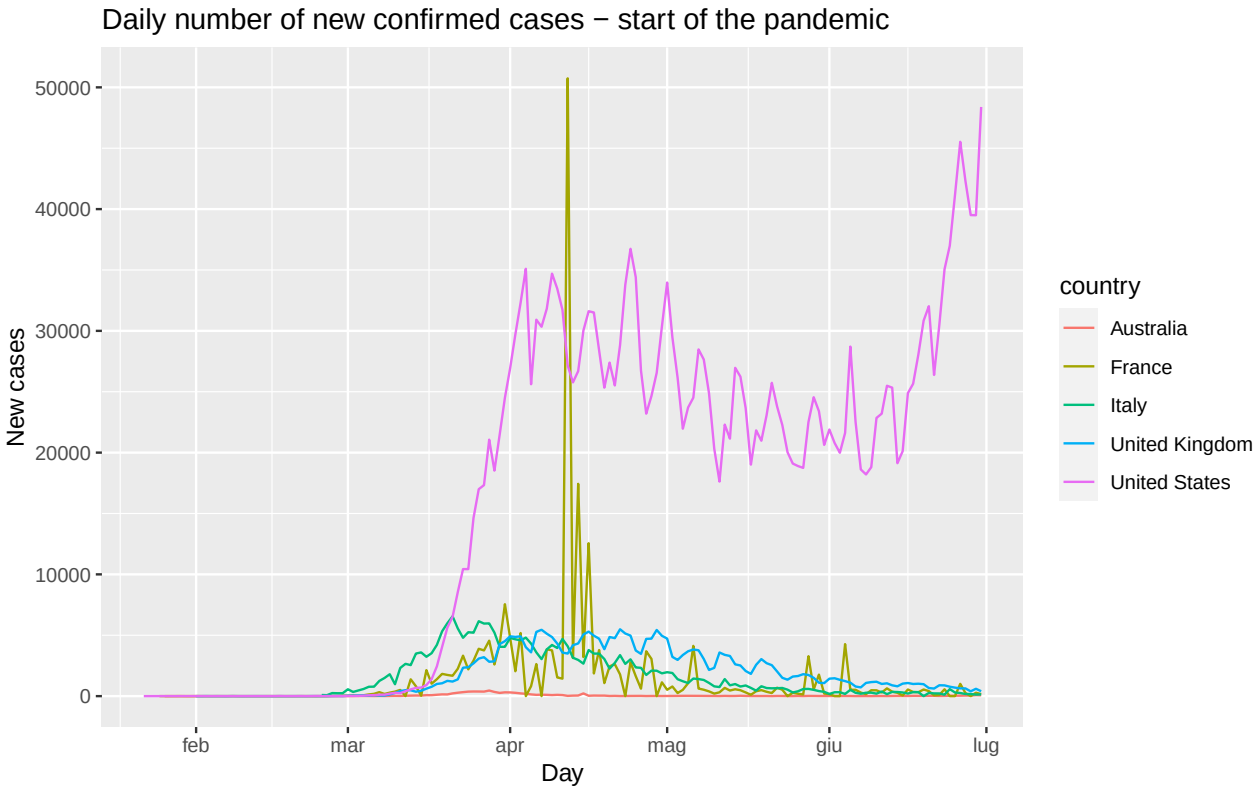
We can see the very early bump that happened in Italy at the start of the pandemic.

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

```
covid %>%
  group_by(country) %>%
  arrange(date, .by_group = TRUE) %>%
  mutate(new_cases = confirmed - lag(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? This could be due to the nature of the reporting (there were lags in the reporting on the weekends), various delays, and the number of tests per day (again, lower during the weekend). 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 at this point.

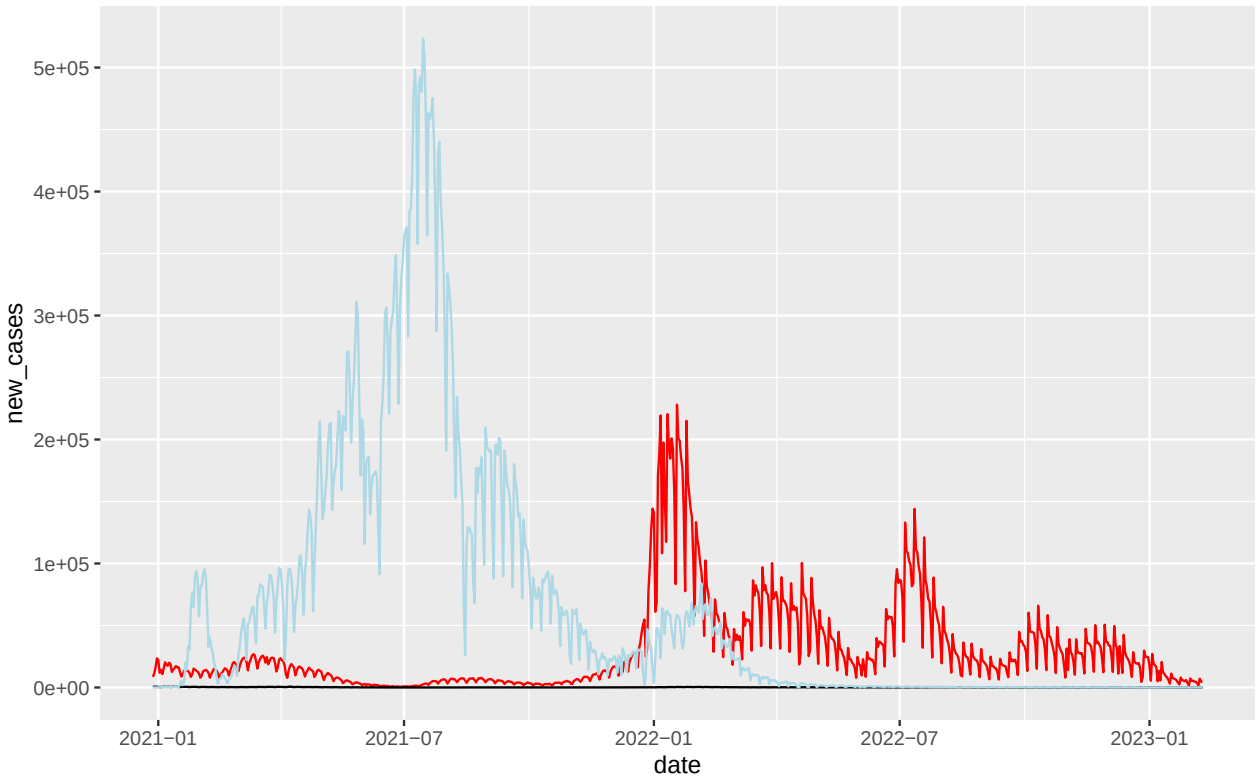
> Calculate the number of daily cases, the daily fully vaccinated subjects and the daily deaths. Visualize the data in the most clear manner.

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")
```



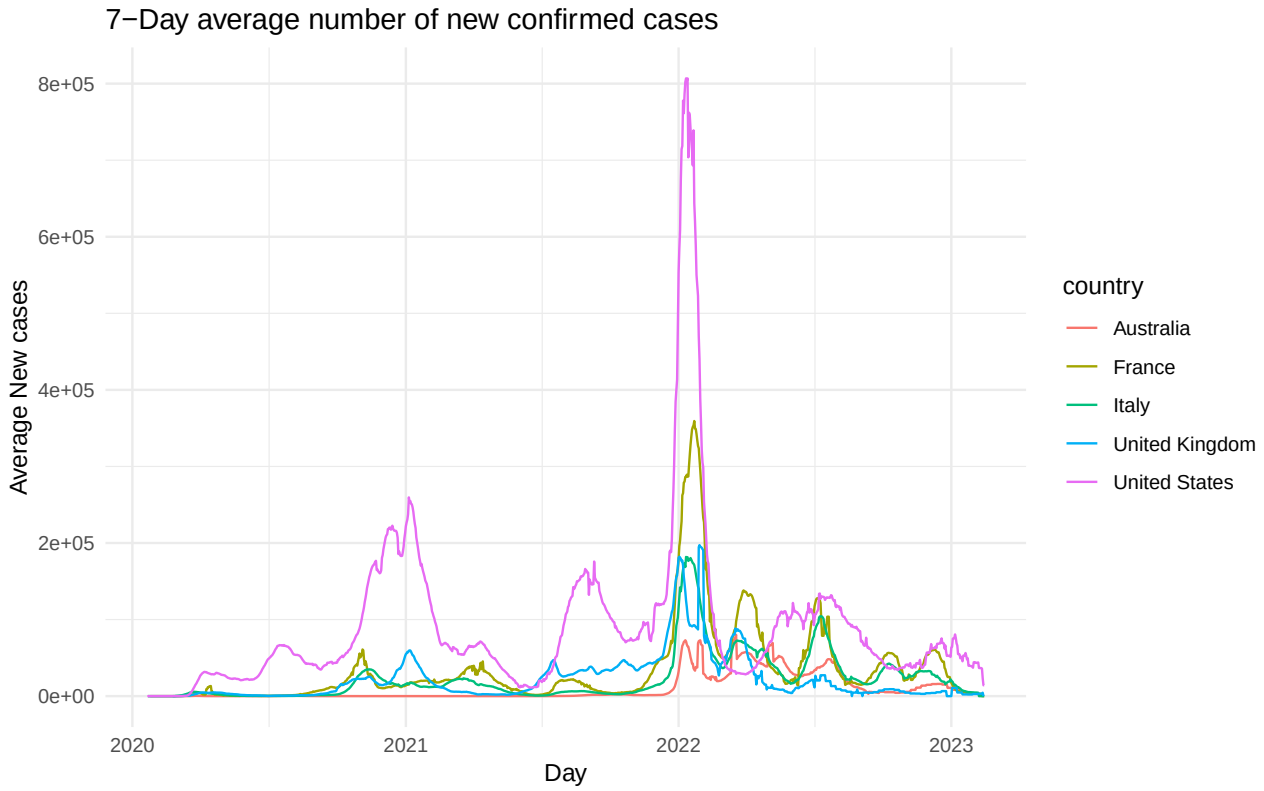
There are a few issues with the above plot: - The data is very spiky; - The number of deaths (fortunately) are on a very different scale than the number of new cases and vaccinations. - There is no legend, or title, or good labels.

Let's address the above considerations. I guess a weekly average will smooth out all the "spikyness":

```
# 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. I use the country plot above:

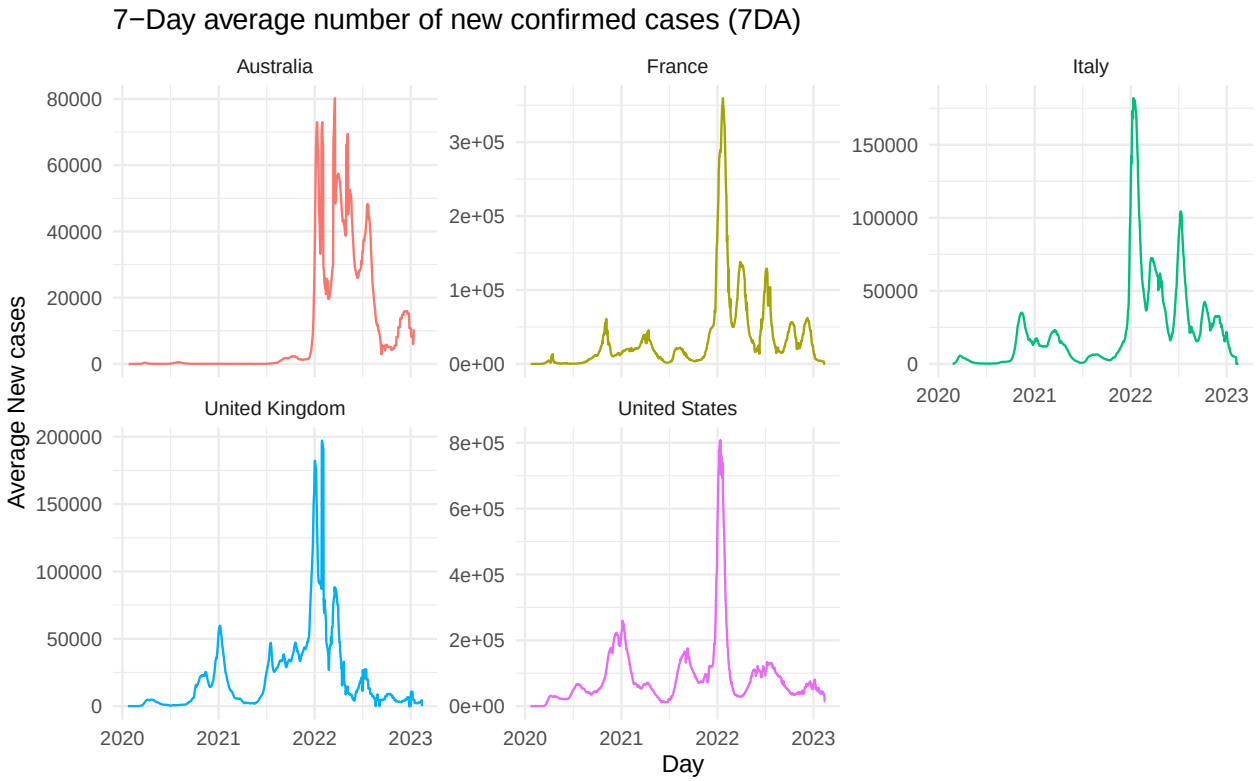
```
# Need to regen the plot_covid dataset
covid %>% filter(country %in% countries_of_interest) -> plot_covid

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



Smooth like butter! The plot is a bit crowded, however.

We can plot the various lines in different plots with `facet_wrap` or `grid.arrange` from the `grid` package:

```
plot_covid %>% replace_na(list("date" = 0, "smooth_new_cases" = 0)) %>%
  ggplot(aes(x = date, y = smooth_new_cases, color = country)) +
  ggtitle("7-Day average number of new confirmed cases (7DA)") +
  xlab("Day") + ylab("Average New cases") +
  theme_minimal() +
  geom_line() +
  theme(legend.position = "none") + # Suppress the legend, as it is superfluous
  facet_wrap(~country, scale = "free_y")
```

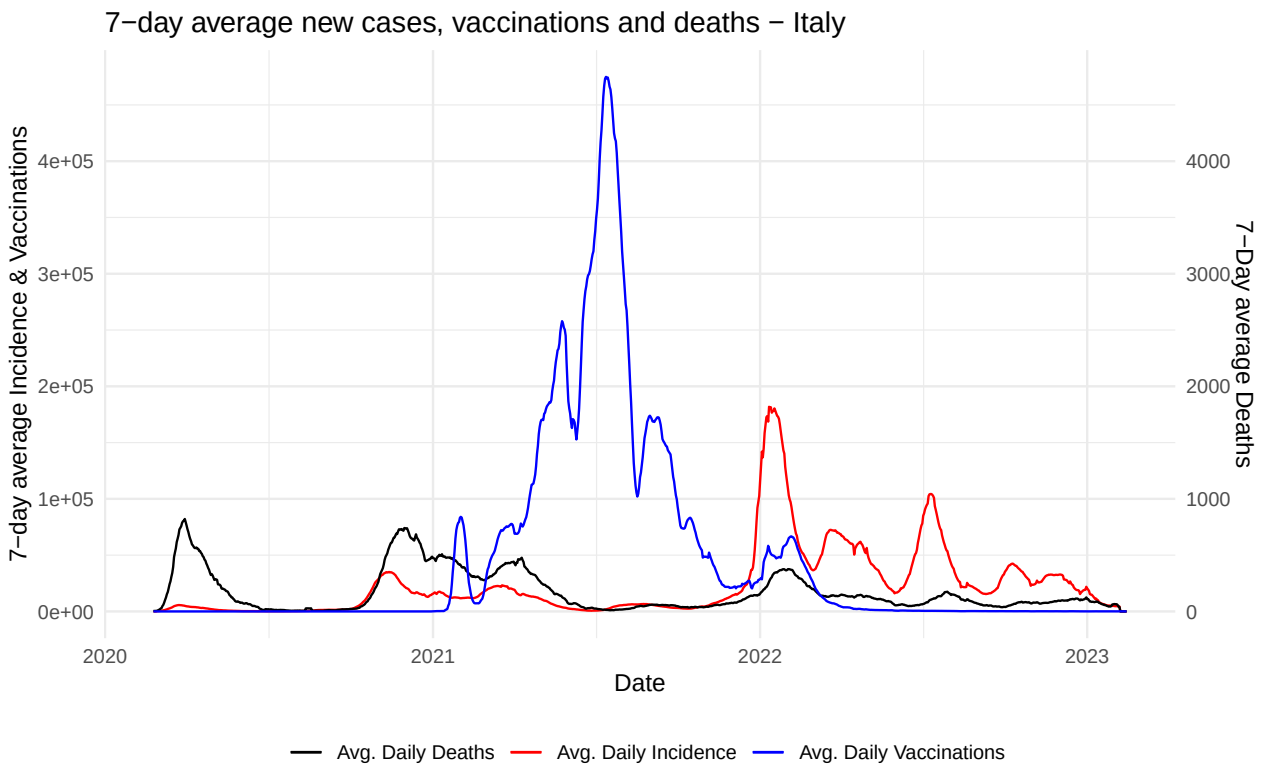



```
  # Wrap based on country into different plots, allowing the y axis to fluctuate.
```

Going back to the actual exercise:

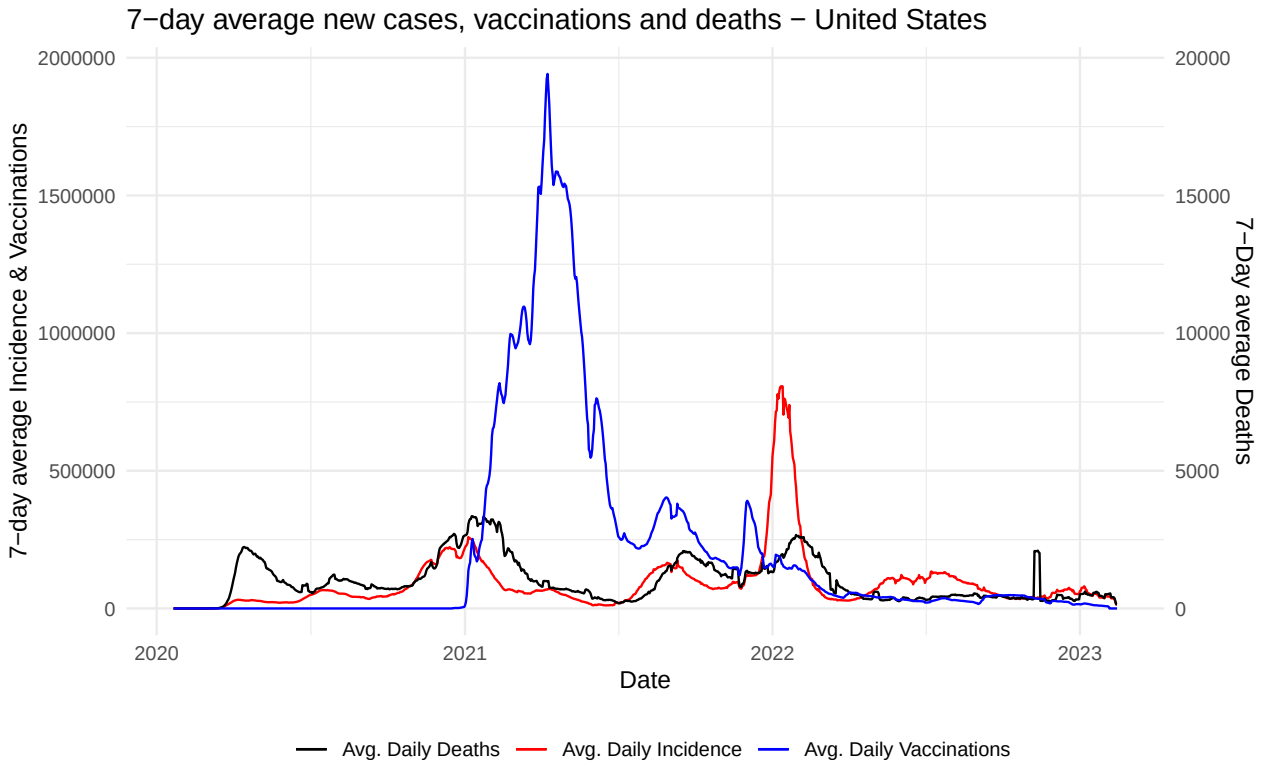
```
# Let's make a function to generate any trend plot that we might want
plot_covid_trend <- function(covid, country_name) {
  covid %>%
    filter(country == country_name) %>%
    # Get rid of NAs, to avoid warnings later. They can also be assimilated to
    # 0 quite decently (most missing values are in actuality 0).
    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 = "Avg. Daily Incidence")) +
    geom_line(aes(y = smooth_new_deaths * 100, color = "Avg. Daily Deaths")) +
    geom_line(aes(y = smooth_new_vaccinations, color = "Avg. Daily Vaccinations")) +
    # Add a second axis for the deaths, as they are on a whole different scale
    scale_y_continuous(
      name = "7-day average Incidence & Vaccinations",
      sec.axis = sec_axis(trans = ~./ 100, name = "7-Day average Deaths")
    ) +
    scale_color_manual(values=c("black", "red", "blue")) +
    xlab("Date") +
    ggtitle(paste0("7-day average new cases, vaccinations and deaths - ", country_name)) +
    theme_minimal() +
    theme(
      legend.position = "bottom", legend.title = element_blank()
    ) -> p
```

```
  p
}

plot_covid_trend(covid, "Italy")
```

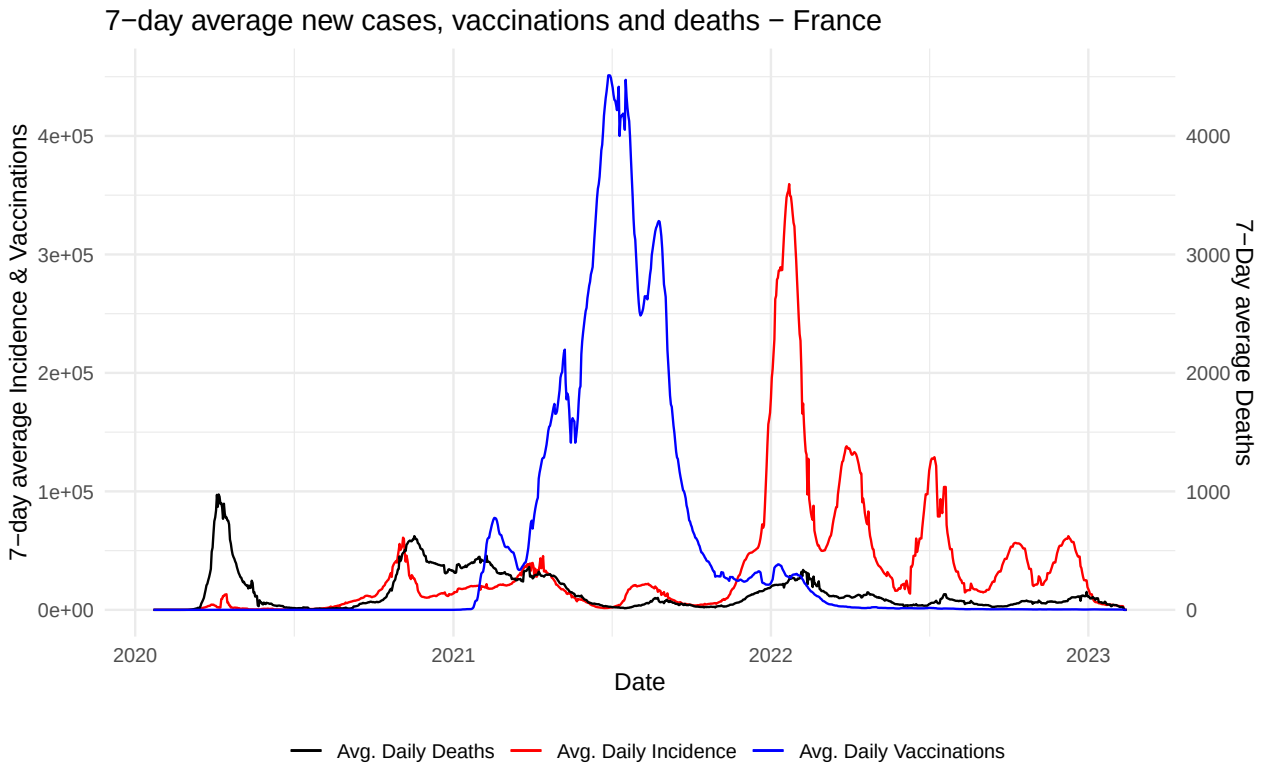


The effect of the vaccination is dramatic! Let's take a look at other countries:

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

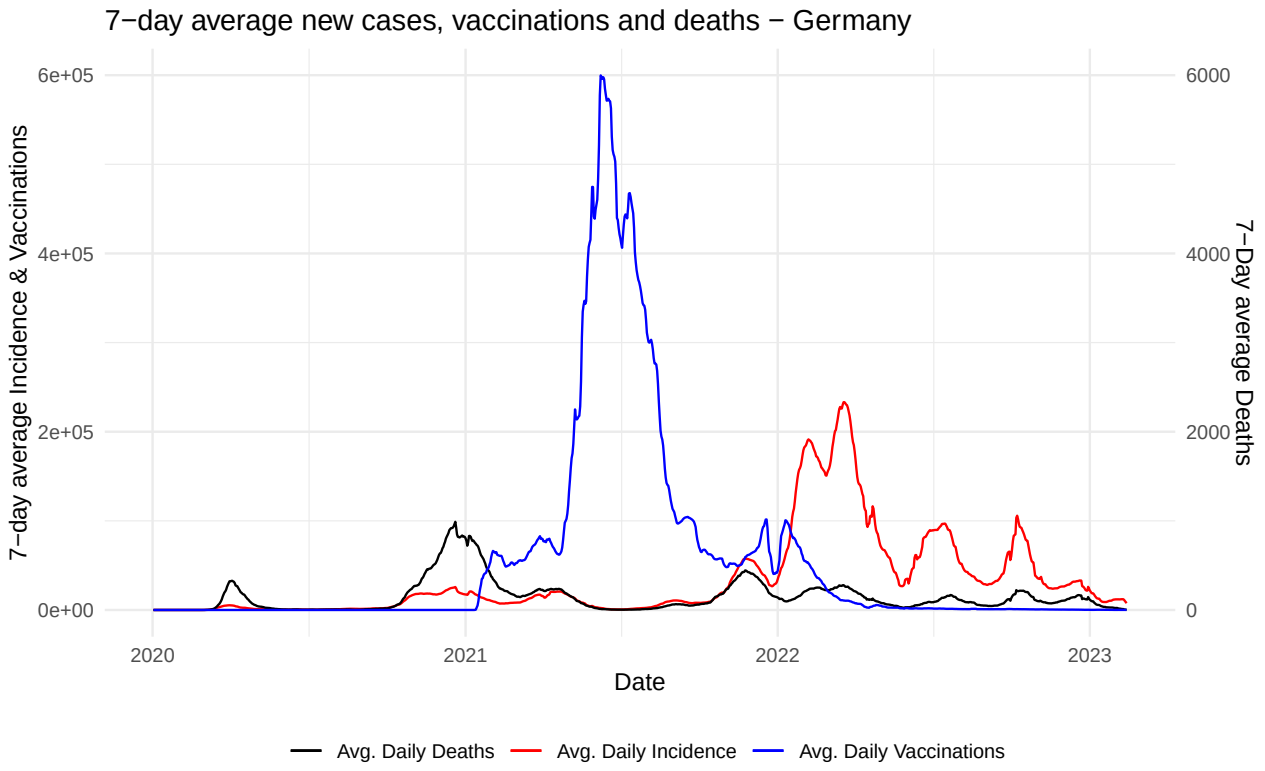



```
plot_covid_trend(covid, "France")
```

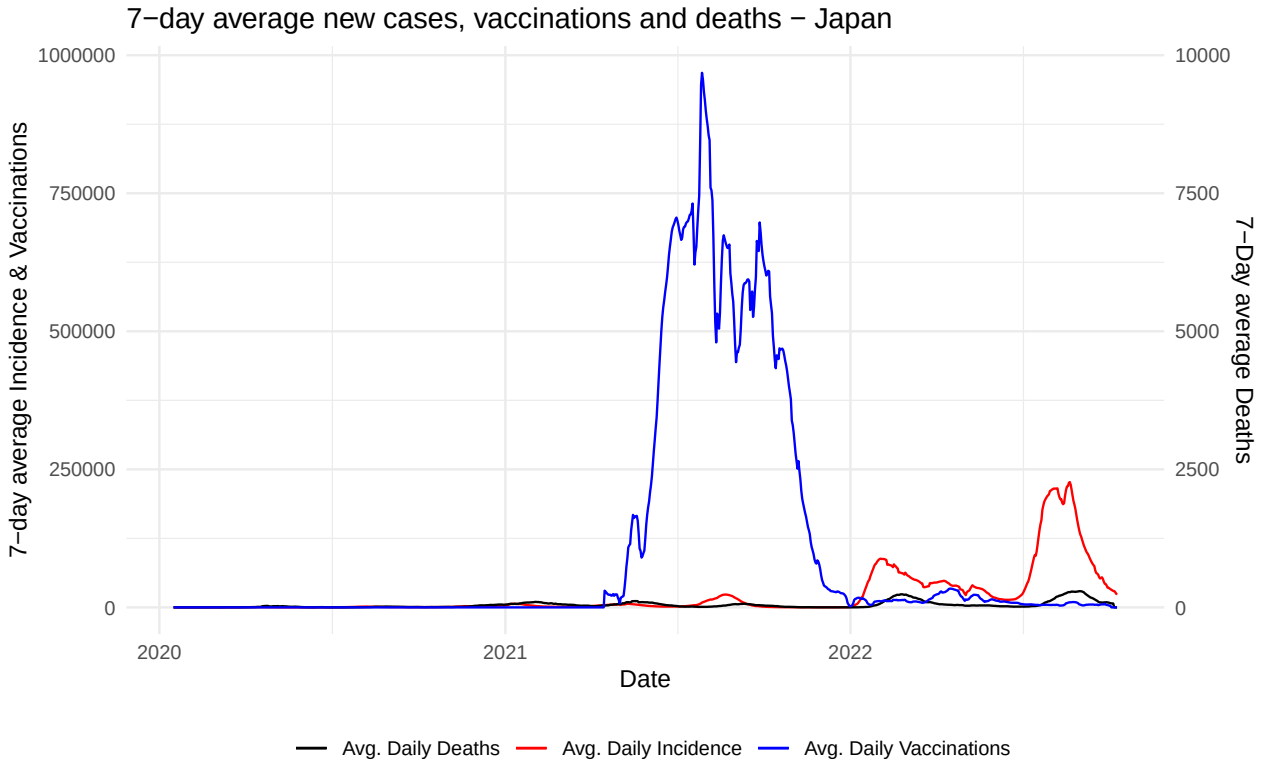


France is very similar to Italy in terms of these trends.

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

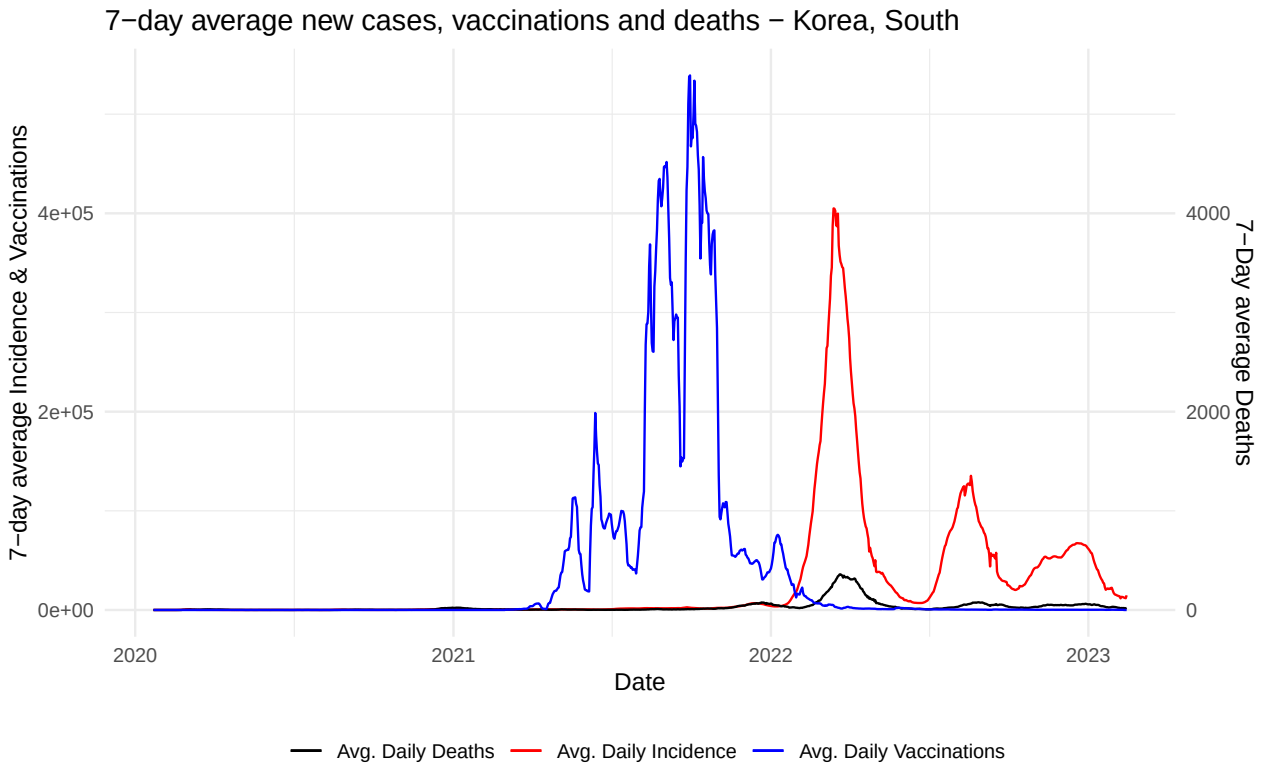



```
plot_covid_trend(covid, "Japan")
```

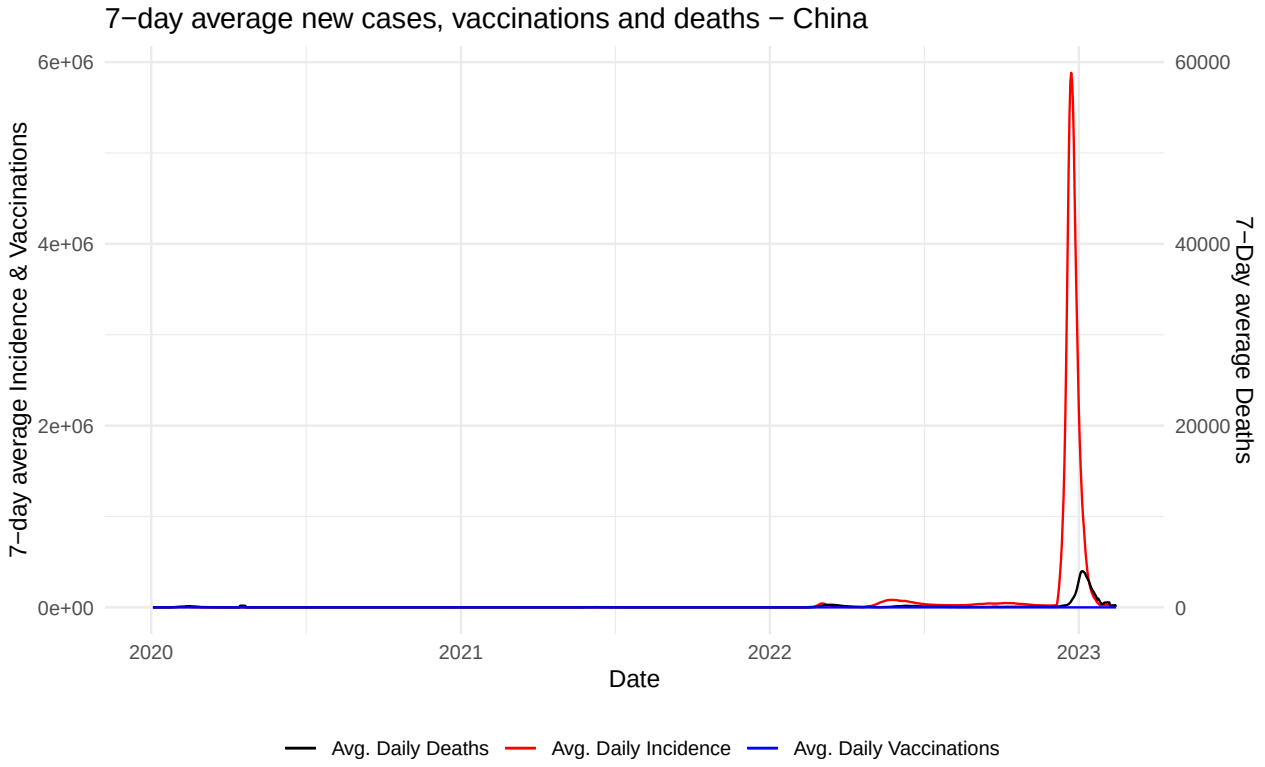


Japan was very successful in preventing the worse of the pandemic.

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

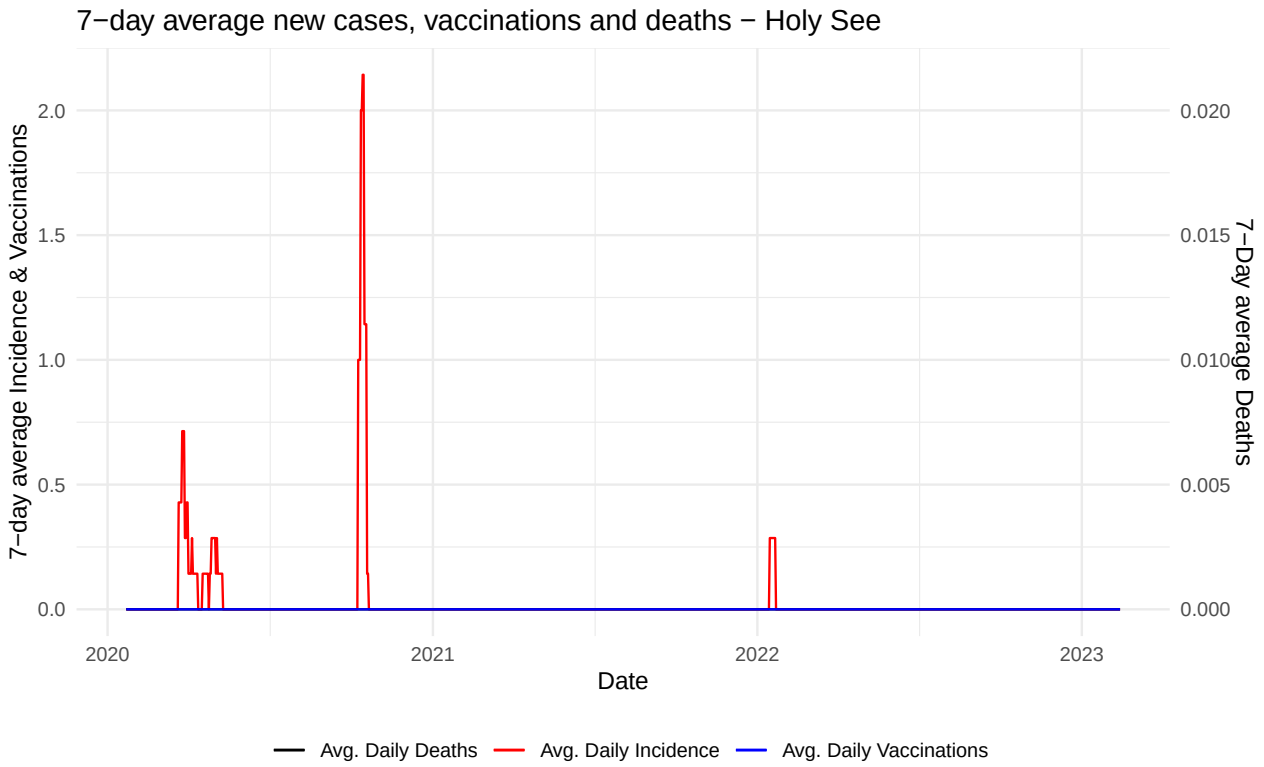



```
plot_covid_trend(covid, "China")
```



There was literally no reporting of what happened in China.

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



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 auto-correlated 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 (we can assume that the larger the bins, the less they are correlated).

Note that this is not a correct calculation for "mortality". I assume here that all the people that get infected in January (or any other month) will either die or recover in January. But this is obviously not true, so there is a bit of a "bleed-through" effect between months that is not considered here. But I think it's good enough for now.

```
# 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) %>%
  summarize(
    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),
    .groups = "keep"
  ) -> monthly_covid

monthly_covid %>%
  replace_na(list(monthly_new_cases = 0, monthly_new_deaths = 0, monthly_new_vaccinations = 0)) %>%
  mutate(monthly_mortality = monthly_new_deaths / monthly_new_cases) -> monthly_covid
```

We can now get the first month were the vaccinations started. In my opinion, it would be better to put the line at where the population is somewhat vaccinated, like at 50%. But for simplicity's sake, let's not.

```
get_first_vaccination_month <- function(daily_data, country) {
  data <- get_first_vaccination_day(daily_data)

  interest <- data[[country]]

  format(interest, "%Y-%m")
}

get_first_vaccination_month(covid, "Italy")
```

```
## [1] "2020-12"
```

We can now run a simple t-test (or a non-parametric Mann-Whiteney test, but I don't check if the requirements of the t-test are respected, so I just do a t-test) to see if there are differences between the mortality rates before and after the start of the vaccination say, in Italy:

```
run_mortality_t_test <- function(data, monthly_data, target_country) {
  first_month <- get_first_vaccination_month(data, target_country)

  data_subs <- monthly_data[monthly_data$country == target_country, ]

  t.test(
    data_subs$monthly_mortality[data_subs$month <= first_month],
    data_subs$monthly_mortality[data_subs$month > first_month]
  )
}

run_mortality_t_test(covid, monthly_covid, "Italy")
```

```
##
##  Welch Two Sample t-test
##
## data:  data_subs$monthly_mortality[data_subs$month <= first_month] and data_subs$monthly_mortality[da
## t = 2.8969, df = 10.169, p-value = 0.01566
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.01491195 0.11331223
## sample estimates:
##  mean of x  mean of y
## 0.07413824 0.01002615
```

There seems to be a significant difference! The average mortality over the months dropped from 0.07 (7%) to 0.01 (1%). It seems that the vaccine really does lower the mortality rate. Let's see the same for France:

```
run_mortality_t_test(covid, monthly_covid, "France")
```

```
##
##  Welch Two Sample t-test
##
## data:  data_subs$monthly_mortality[data_subs$month <= first_month] and data_subs$monthly_mortality[da
## t = 2.356, df = 11.076, p-value = 0.03793
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.003009388 0.087382061
## sample estimates:
##   mean of x   mean of y
## 0.050851730 0.005656006
```

This is also significant, from 0.05 to 0.005, 10x less! Finally, let's see the United States:

```
run_mortality_t_test(covid, monthly_covid, "United States")
```

```
##
##  Welch Two Sample t-test
##
## data:  data_subs$monthly_mortality[data_subs$month <= first_month] and data_subs$monthly_mortality[da
## t = 1.7775, df = 12.698, p-value = 0.09943
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.002265204  0.023014959
## sample estimates:
##  mean of x  mean of y
## 0.02291259 0.01253772
```

Slightly non-significant, if we consider the canonical alpha. But the mortality still drops, from around 0.02 to 0.01.

Reporting practices aside, consider that this information makes no assumption on the actual effectiveness of the vaccination policy (e.g. how much of the population is vaccinated) nor to the state-specific definition of "fully vaccinated". It also does not take into account other policies that might reduce the mortality rate, such as better preparedness of the public healthcare system or better treatments.

> 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.

Let's see the two counties in question:

```
first_days <- get_first_vaccination_day(covid)
# Let's get rid of Peru, which has something wrong in it...
first_days$Peru <- NA

# The first country
first_days[which(unlist(first_days) == min(unlist(first_days), na.rm = TRUE))]
```

```
## $`United States`
## [1] "2020-12-13"
```

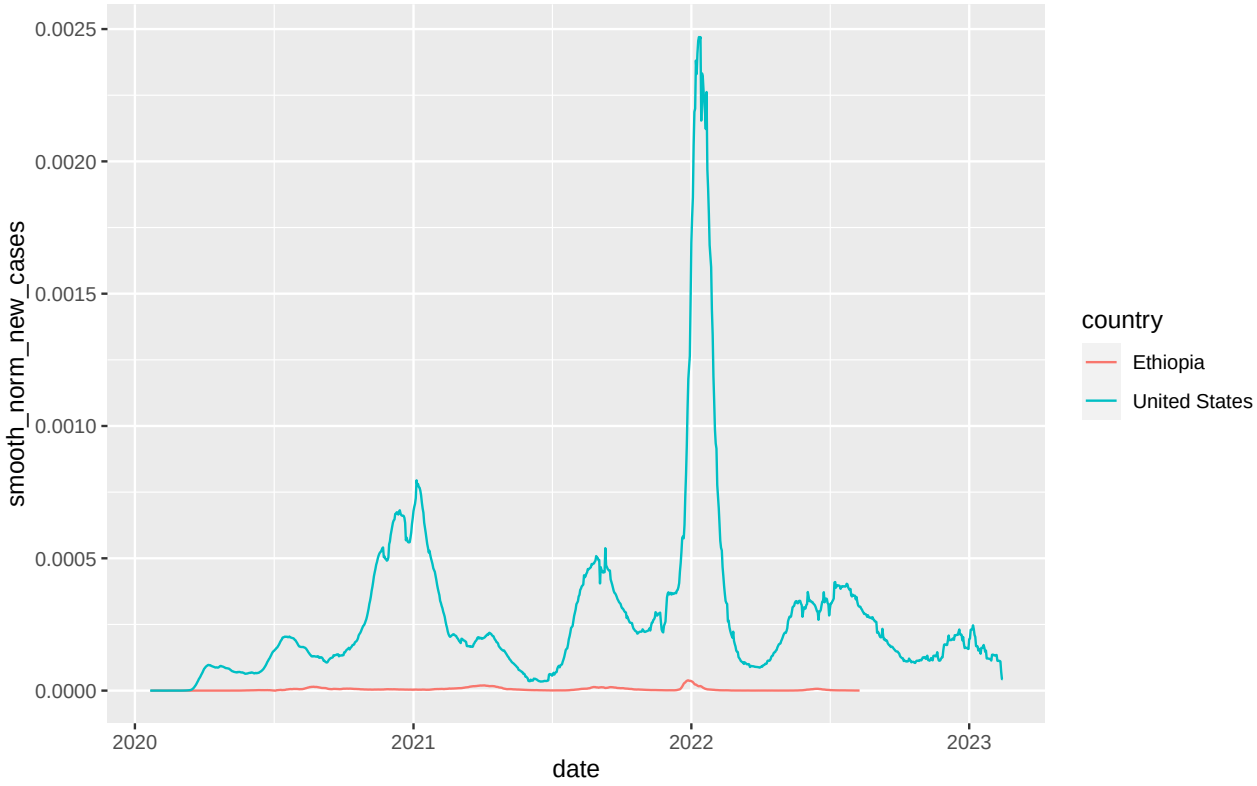
```
# The last country
first_days[which(unlist(first_days) == max(unlist(first_days), na.rm = TRUE))]
```

```
## $Ethiopia
```

```
## [1] "2022-03-06"
```

Let's plot the United States vs Ethiopia:

```
covid %>%
  filter(country %in% c("United States", "Ethiopia")) %>%
  replace_na(
      list(smooth_new_cases = 0, smooth_new_deaths = 0, smooth_new_vaccinations = 0)
    ) %>%
  group_by(country) %>%
  mutate(smooth_norm_new_cases = smooth_new_cases / population) %>%
  ggplot(aes(x = date, color = country)) +
  geom_line(aes(y = smooth_norm_new_cases))
```



This is probably due to the poor data collection in the different countries. Let's answer the question but considering only a selection of western countries:

```
considered_countries <- c(
  "Italy", "United States", "Germany", "United Kingdom", "France", "Spain",
  "Romania", "Norway", "Finland", "Canada", "Mexico", "Portugal", "Greece",
  "Denmark", "Austria", "Belgium", "Czech Republic", "Netherlands"
)

covid %>% filter(country %in% considered_countries) -> covid_subset

first_days <- get_first_vaccination_day(covid_subset)

# The first country
first_country <- first_days[which(unlist(first_days) == min(unlist(first_days), na.rm = TRUE))]
print(first_country)
```

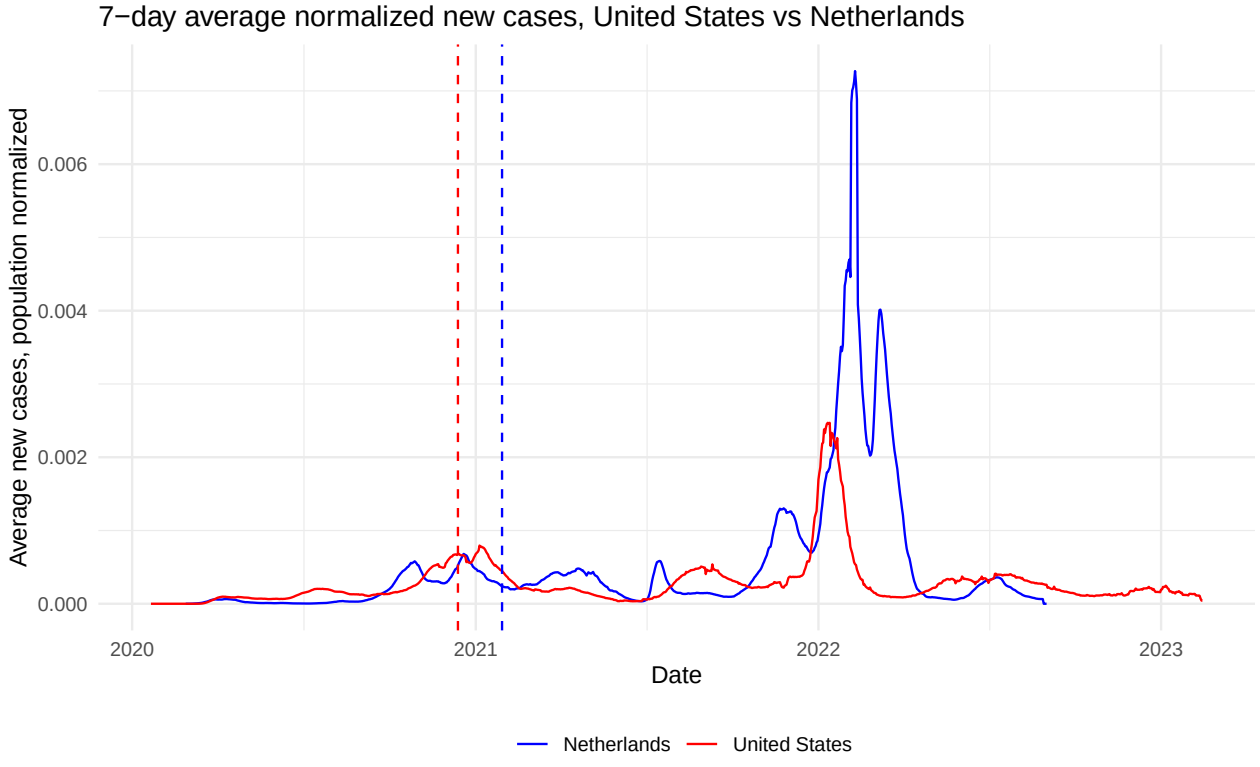
```
## $`United States`
## [1] "2020-12-13"
```

```
# The last country
last_country <- first_days[which(unlist(first_days) == max(unlist(first_days), na.rm = TRUE))]
print(last_country)
```

```
## $Netherlands
## [1] "2021-01-29"
```

Let's see the United States vs the Netherlands:

```
covid %>%
  filter(country %in% c("United States", "Netherlands")) %>%
  replace_na(
      list(smooth_new_cases = 0, smooth_new_deaths = 0, smooth_new_vaccinations = 0)
    ) %>%
  group_by(country) %>%
  mutate(smooth_norm_new_cases = smooth_new_cases / population) %>%
  ggplot(aes(x = date, color = country)) +
  geom_line(aes(y = smooth_norm_new_cases)) +
  geom_vline(xintercept = unlist(first_country), linetype = "dashed", colour = "red") +
  geom_vline(xintercept = unlist(last_country), linetype = "dashed", colour = "blue") +
  scale_color_manual(values=c("blue", "red")) +
  xlab("Date") +
  ylab("Average new cases, population normalized") +
  ggtitle(paste0("7-day average normalized new cases, United States vs Netherlands")) +
  theme_minimal() +
  theme(
    legend.position = "bottom", legend.title = element_blank()
  )
```



The vertical lines represent the first vaccination day of the two countries. As the starting dates were similar, it is a bit unclear to me what knowledge we can derive from this plot, however. Perhaps having more countries of interest, it would be a better visualization.