# R Capstone Project: COVID-19 Data Part 3

Carl Klein

2023-08-11

```
library(tidyverse)
library(lubridate)
library(usmap)
library(gridExtra)
library(zoo)
```

```
# df: the combined US county data from 20, 21, 22
us_counties_2020 <- read_csv("https://raw.githubusercontent.com/nytimes/covid-19-data/master/us-counties
us_counties_2021 <- read_csv("https://raw.githubusercontent.com/nytimes/covid-19-data/master/us-counties
us_counties_2022 <- read_csv("https://raw.githubusercontent.com/nytimes/covid-19-data/master/us-counties

# Combine and tidy the 2020, 2021, and 2022 COVID data sets.
df <- rbind(us_counties_2020, us_counties_2021, us_counties_2022) %>%
  filter(state != 'Puerto Rico')
```

```
# Import global COVID-19 statistics aggregated by the Center for Systems Science
# and Engineering (CSSE) at Johns Hopkins University.
# Import global population estimates from the World Bank.

csse_global_deaths <- read_csv("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_c
csse_global_cases <- read_csv("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_co
csse_us_deaths <- read_csv("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_
csse_us_cases <- read_csv("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_

global_pop <- read_csv("Data/global_population_estimates.csv")
```

**Part 3 - Global Comparison**

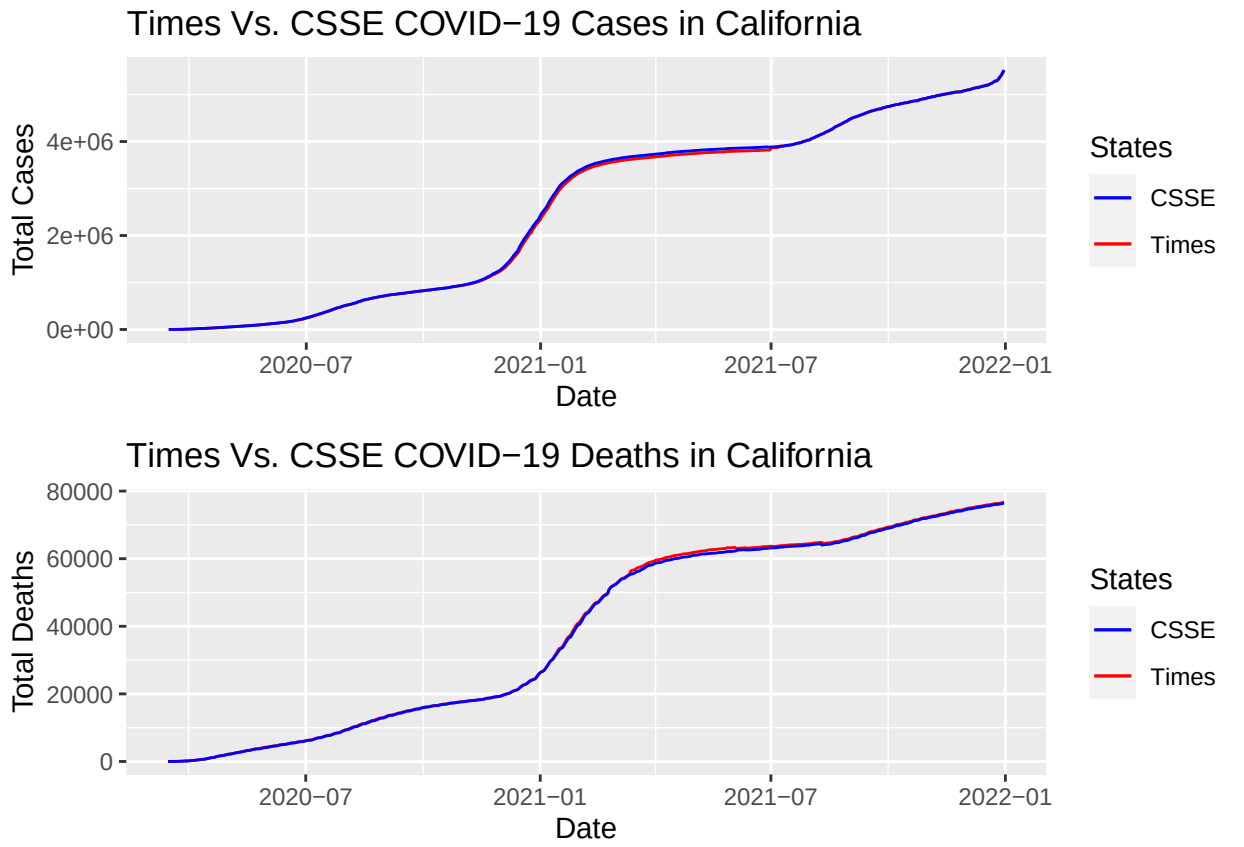
**Question 1** Using the state you selected in Part 2 Question 2 compare the daily number of cases and deaths reported from the CSSE and NY Times.

```
# Tidy the US CSSE death and cases data
# cases tidied
csse_us_cases <- csse_us_cases %>%
  pivot_longer(cols = matches('.*/.*/.*'),
               names_to = 'date',
               values_to = 'cases') %>%
  relocate(date)

# deaths tidied
csse_us_deaths <- csse_us_deaths %>%
  pivot_longer(cols = matches('.*/.*/.*'),
```

```
                names_to = 'date',
                values_to = 'deaths') %>%
  relocate(date)

# Join the CSSE datasets, will need to reformat the date data
csse_data <- csse_us_deaths %>%
  full_join(csse_us_cases) %>%
  relocate(date, Province_State, Admin2, deaths, cases, Population) %>%
  mutate(date = as.Date(parse_date_time(date, 'mdy')))

# Plot the CSSE and NY Times cases and deaths
# we'll use our analysis standard plot dates of 2020-03-15 to 2021-12-31
compare_data <- csse_data %>%
  select(date, Province_State, Admin2, deaths, cases, Population) %>%
  rename(c('state' = 'Province_State',
           'county' = 'Admin2',
           'population' = 'Population')) %>%
  full_join(df, by = join_by(date, state, county)) %>%
  rename(c('csse_deaths' = 'deaths.x',
           'csse_cases' = 'cases.x',
           'times_cases' = 'cases.y',
           'times_deaths' = 'deaths.y')) %>%
  filter(state == 'California') %>%
  filter(date >= '2020-03-15') %>%
  filter(date <= '2021-12-31') %>%
  group_by(date, state) %>%
  summarise(csse_deaths_ca = sum(csse_deaths, na.rm = TRUE),
            csse_cases_ca = sum(csse_cases, na.rm = TRUE),
            times_deaths_ca = sum(times_deaths, na.rm = TRUE),
            times_cases_ca = sum(times_cases, na.rm = TRUE))
```

```
# plot for the cases
plot1 <- ggplot(data = compare_data) +
  geom_line(aes(x = date, y = times_cases_ca, color = 'Times')) +
  geom_line(aes(x = date, y = csse_cases_ca, color = 'CSSE')) +
  labs(title = 'Times Vs. CSSE COVID-19 Cases in California') +
  xlab('Date') +
  ylab('Total Cases') +
  scale_color_manual(name = 'States',
                     values = c('Times' = 'red',
                                'CSSE' = 'blue'))

# plot for the total deaths
plot2 <- ggplot(data = compare_data) +
  geom_line(aes(x = date, y = times_deaths_ca, color = 'Times')) +
  geom_line(aes(x = date, y = csse_deaths_ca, color = 'CSSE')) +
  labs(title = 'Times Vs. CSSE COVID-19 Deaths in California') +
  xlab('Date') +
  ylab('Total Deaths') +
  scale_color_manual(name = 'States',
                     values = c('Times' = 'red',
                                'CSSE' = 'blue'))

# put the plots in a grid together (uses gridExtra)
```

```
grid.arrange(plot1, plot2)
```




– Methodology, results, and Interpretation here – First, the CSSE US data sets needed tidying by performing a pivot_longer to turn the dates into a single variable. I chose to pair match with regex to find all of the columns that were dates, rather than trying to match column numbers. After tidying the CSSE cases and deaths datasets, I joined them together.

Next, we wanted to compare the CSSE data with the Times data that has been used in the previous sections. After filtering and merging the data, plots for cases and deaths were created to compare CSSE and Times. Visually, the data was almost identical. I chose to use the same time period we've focused on in the previous sections, which was March 15, 2020 to December 31, 2021.

---

**Question 2** Now that you have verified the data reported from the CSSE and NY Times are similar, combine the global and US CSSE data sets and identify the top 10 countries in terms of deaths and cases per 100,000 people between March 15, 2020, and December 31, 2021.

```
# First, combine and tidy the CSSE death and cases data sets. You may wish to keep the two sets separat

# We can reuse the tidied and combined CSSE US data from Q1, named case data.

# Looked at the CSSE Global data, we can use a similar pivot_longer method to
# tidy that data, and then combine the sets in a similar manner.
csse_global_cases <- csse_global_cases %>%
  pivot_longer(cols = matches('.*/.*/.*'),
```

```
               names_to = 'date',
               values_to = 'cases') %>%
  relocate(date)

# deaths tidied
csse_global_deaths <- csse_global_deaths %>%
  pivot_longer(cols = matches('.*/.*/.*'),
               names_to = 'date',
               values_to = 'deaths') %>%
  relocate(date)

# Join the CSSE datasets, will need to reformat the date data
csse_global_data <- csse_global_deaths %>%
  full_join(csse_global_cases) %>%
  mutate(date = as.Date(parse_date_time(date, 'mdy')))

# Then, tidy the global population estimates. While tidying your data, remember to include columns that
# You will notice that the population estimates data does not include every country reported in the CSS

# tidy the global population data
global_pop_tidy <- global_pop %>%
  rename(c('country' = 'Country Name',
           'pop_20' = '2020 [YR2020]',
           'pop_21' = '2021 [YR2021]')) %>%
  select(country, pop_20, pop_21) %>%
  mutate(pop_20 = as.numeric(pop_20)) %>%
  mutate(pop_21 = as.numeric(pop_21)) %>%
  filter(!is.na(pop_20)) %>%
  filter(!is.na(pop_21))

# merge on global_pop, change population to correspond to correct year
csse_global_data_with_pop <- csse_global_data %>%
  full_join(global_pop_tidy, join_by(`Country/Region` == country)) %>%
  mutate(population = ifelse(date < '2021-01-01', pop_20, pop_21)) %>%
  relocate(population, .before = pop_20) %>%
  select(-c(pop_20, pop_21)) %>%
  rename(c('prov_state' = 'Province/State',
           'country' = 'Country/Region'))

# find top 10 countries by deaths and cases per 100k
top_countries <- csse_global_data_with_pop %>%
  group_by(date, country, population) %>%
  summarise(country_deaths = sum(deaths), country_cases = sum(cases)) %>%
  filter(!is.na(population)) %>%
  filter(date == '2021-12-31') %>%
  mutate(deaths_100k = (country_deaths / population) * 100000) %>%
  mutate(cases_100k = (country_cases / population) * 100000) %>%
  arrange(-deaths_100k, -cases_100k) %>%
  select(country, deaths_100k, cases_100k)

head(top_countries, n = 10)
```

```
## # A tibble: 10 x 4
## # Groups:   date, country [10]
```

```
##    date       country                deaths_100k cases_100k
##    <date>     <chr>                        <dbl>      <dbl>
##  1 2021-12-31 Peru                          608.      6885.
##  2 2021-12-31 Bulgaria                      450.     10856.
##  3 2021-12-31 Bosnia and Herzegovina        412.      8928.
##  4 2021-12-31 Hungary                       403.     12925.
##  5 2021-12-31 Moldova                       393.     14390.
##  6 2021-12-31 Montenegro                    388.     27381.
##  7 2021-12-31 North Macedonia               384.     10861.
##  8 2021-12-31 Georgia                       372.     25182.
##  9 2021-12-31 Croatia                       312.     17770.
## 10 2021-12-31 Romania                       307.      9443.
```

– Methodology, results, and Interpretation here – After tidying and joining the CSSE global data using a similar method to how we tidied the CSSE US data, we started massaging the global population data. This consisted of mostly renaming and filtering out countries with missing data. Then, we combined the tidied sets and performed renaming to make the variables easier to call.

Finally, we grouped and summed a smaller subset of variables. This was mostly to aggregate the state/province variable into the country variable. Since the death and cases data is cumulative, we used our common end date of December 31, 2021, mutated the deaths and cases into per 100,000 people (country population specific), arrnaged the data in descending order by deaths and cases then returned the top 10!

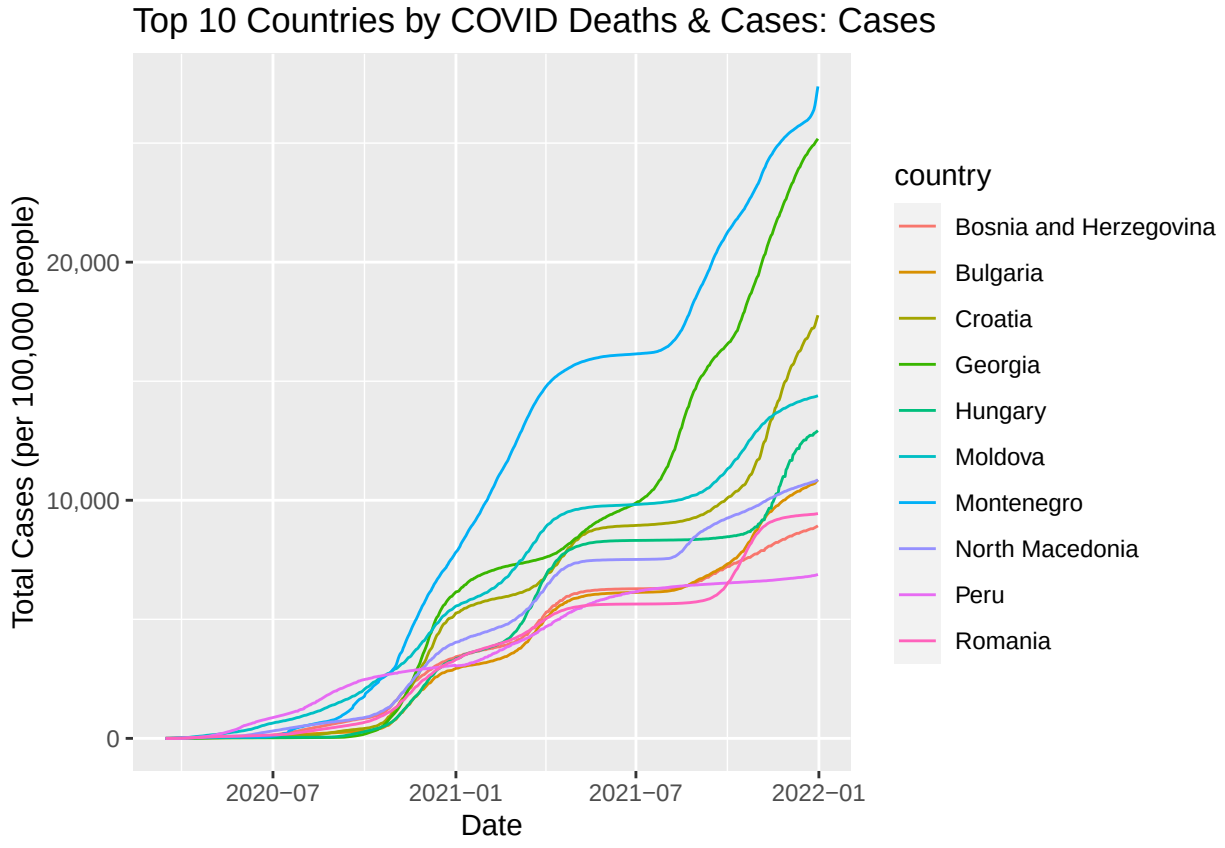
---

**Question 3** Construct a visualization plotting the 10 countries in terms of deaths and cases per 100,000 people between March 15, 2020, and December 31, 2021. In designing your visualization keep the number of data you will be plotting in mind. You may wish to create two separate visualizations, one for deaths and another for cases.

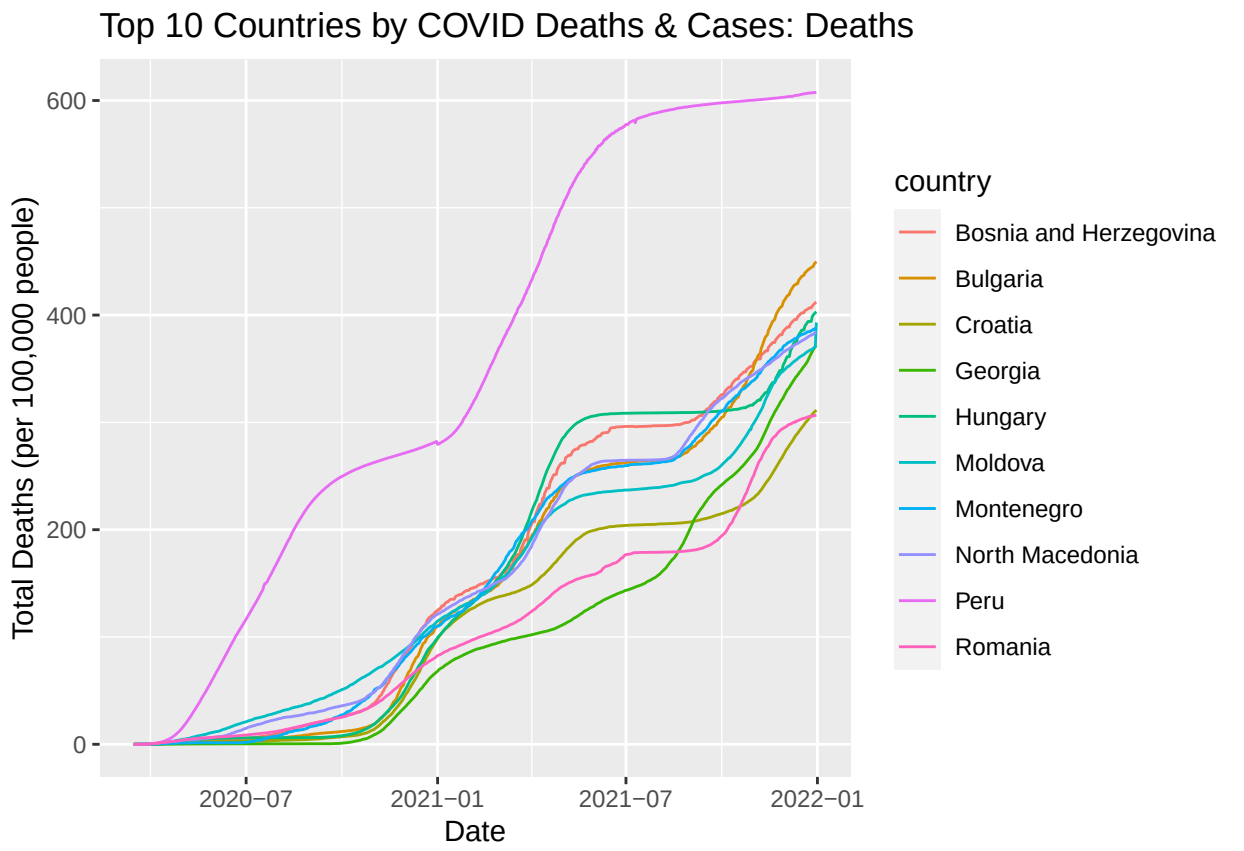
```r
# we can reuse a lot of the code we've written with just some minor tweaks
# take the top 10 coutries from Q2
top_10 <- top_countries %>%
  head(n = 10) %>%
  select(country)

# instead of filtering by our common data enddate, we'll first filter by country
top_10_plot_data <- csse_global_data_with_pop %>%
  group_by(date, country, population) %>%
  summarise(country_deaths = sum(deaths), country_cases = sum(cases)) %>%
  filter(!is.na(population)) %>%
  filter(country %in% top_10$country) %>%
  mutate(deaths_100k = (country_deaths / population) * 100000) %>%
  mutate(cases_100k = (country_cases / population) * 100000) %>%
  filter(date >= '2020-03-15') %>%
  filter(date <= '2021-12-31')
```

```r
# plot the data
# plot for the cases
plot1 <- ggplot(data = top_10_plot_data,
                aes(group = country, color = country)) +
  geom_line(aes(x = date, y = cases_100k))+
  labs(title = 'Top 10 Countries by COVID Deaths & Cases: Cases') +
  xlab('Date') +
  ylab('Total Cases (per 100,000 people)') +
```

```
  scale_y_continuous(labels = scales::comma)

# plot for the total deaths
plot2 <- ggplot(data = top_10_plot_data,
                aes(group = country, color = country)) +
  geom_line(aes(x = date, y = deaths_100k)) +
  labs(title = 'Top 10 Countries by COVID Deaths & Cases: Deaths') +
  xlab('Date') +
  ylab('Total Deaths (per 100,000 people)') +
  scale_y_continuous(labels = scales::comma)

# print the plots
plot1
```




```
plot2
```




– Methodology, results, and Interpretation here – We slightly altered the methods from Q2, this time filtering by the top 10 countries and including our analysis standard range of March 15, 2020 to December 31, 2021.

Unlike the previous sections, we used two separate plots for cases and deaths, as ten categories, including their legend, can be space consuming. In my opinion, ten categories on a plot makes it hard to understand, but it could be just the choice of plot type.

---

**Question 4** Finally, select four countries from one continent and create visualizations for the daily number of confirmed cases per 100,000 and the daily number of deaths per 100,000 people between March 15, 2020, and December 31, 2021.

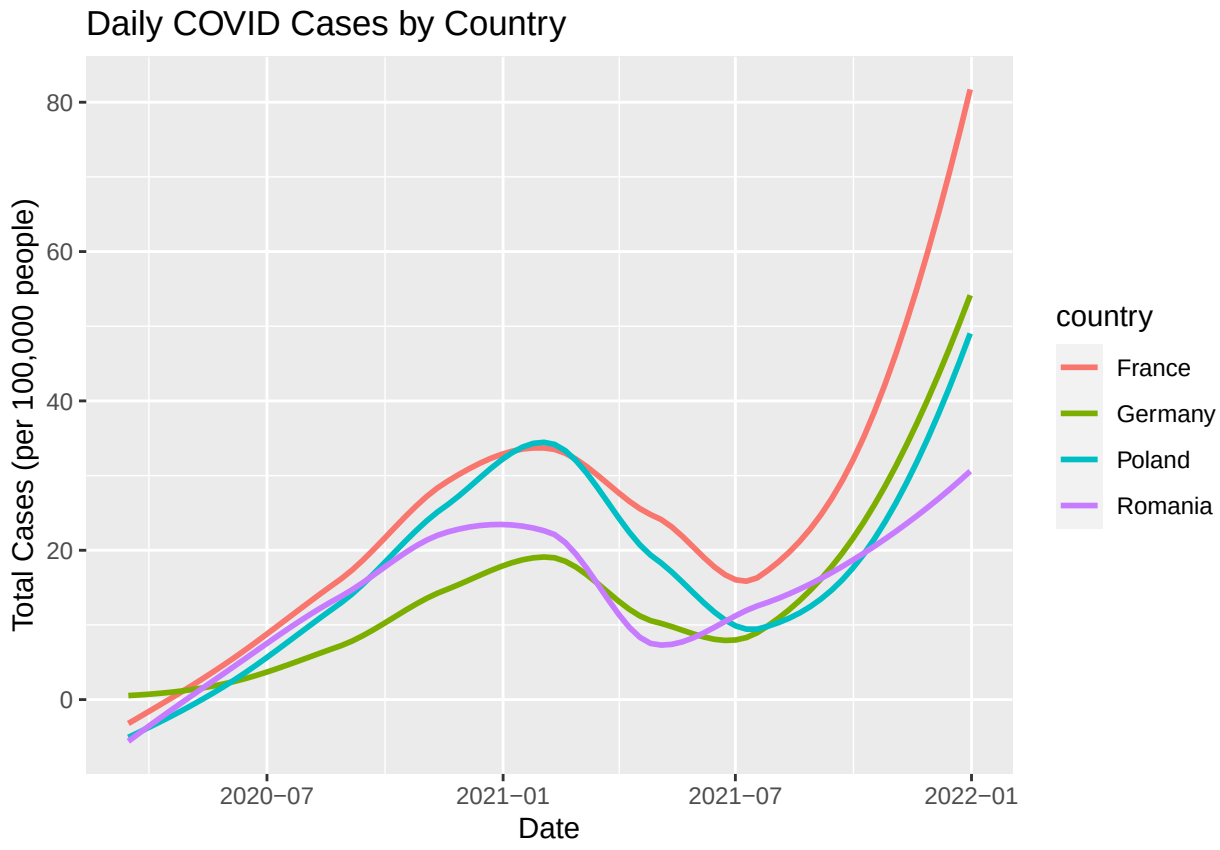
```
# for this we can continue to build from previous code, this time altering the
# data to create daily or "new" cases and deaths columns, and also should
# separate out by country prior to creating the daily new cases

# since we'll repeat the code for 4 countries, let's create a function
# we can start with some slightly refined CSSE  w/ world population data
df_global <- csse_global_data_with_pop %>%
  group_by(date, country, population) %>%
  summarise(country_deaths = sum(deaths), country_cases = sum(cases)) %>%
  filter(!is.na(population))
```

```
country_daily <- function(df = df_global, country_name) {
  # filter by country and date
  df_country <- df %>%
```

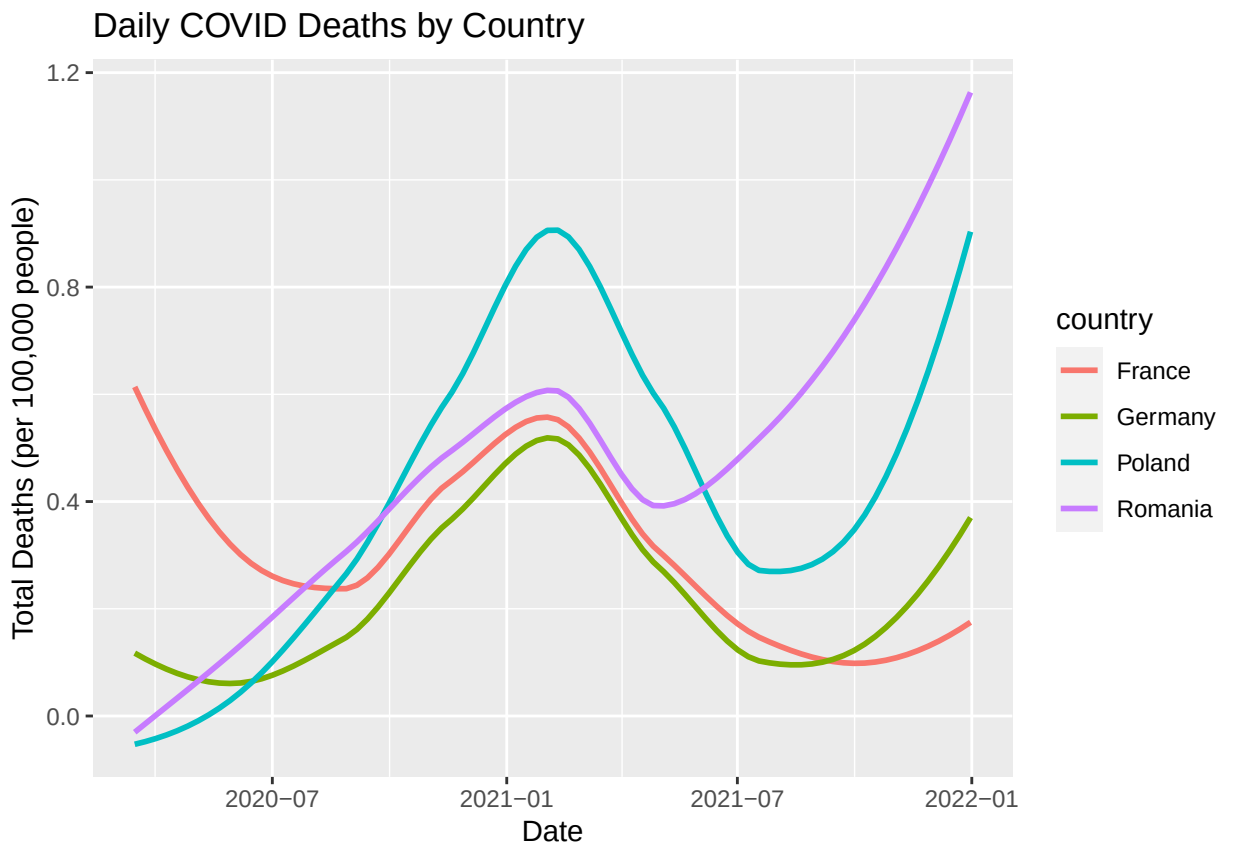
```
    filter(country == country_name) %>%
    filter(date >= '2020-03-15') %>%
    filter(date <= '2021-12-31')

  # calculate new cases
  new_cases = c()
  for (i in 1:length(df_country$country_cases) - 1) {
    new_cases[i] = df_country$country_cases[i + 1] - df_country$country_cases[i]
  }
  new_cases = c(0, new_cases)

  # calculate new deaths
  new_deaths = c()
  for (i in 1:length(df_country$country_deaths) - 1) {
    new_deaths[i] = df_country$country_deaths[i + 1] - df_country$country_deaths[i]
  }
  new_deaths = c(0, new_deaths)

  # bind the new cases and deaths to the dataframe and choose required variables
  df_country <- df_country %>%
    cbind('new_cases' = new_cases, 'new_deaths' = new_deaths) %>%
    mutate(cases_100k = (new_cases / population) * 100000) %>%
    mutate(deaths_100k = (new_deaths / population) * 100000) %>%
    select(date, country, cases_100k, deaths_100k)

  return(df_country)
}
```

```
# pick 4 countries from the same continent
# germany, france, poland, romania
df_germany <- country_daily(df = df_global, country_name = 'Germany')
df_france <- country_daily(df = df_global, country_name = 'France')
df_poland <- country_daily(df = df_global, country_name = 'Poland')
df_romania <- country_daily(df = df_global, country_name = 'Romania')

# combine them back together
df_combined <- rbind(df_germany, df_france, df_poland, df_romania)

# create plots
# cases
plot1 <- ggplot(data = df_combined,
                aes(group = country, color = country)) +
  geom_smooth(aes(x = date, y = cases_100k), se = FALSE) +
  labs(title = 'Daily COVID Cases by Country') +
  xlab('Date') +
  ylab('Total Cases (per 100,000 people)') +
  scale_y_continuous(labels = scales::comma)

# deaths
plot2 <- ggplot(data = df_combined,
                aes(group = country, color = country)) +
  geom_smooth(aes(x = date, y = deaths_100k), se = FALSE) +
  labs(title = 'Daily COVID Deaths by Country') +
  xlab('Date') +
```

```
  ylab('Total Deaths (per 100,000 people)') +
  scale_y_continuous(labels = scales::comma)

# plot
plot1
```




```
plot2
```




– Methodology, results, and Interpretation here – We broke apart the data and put it back together a few times to create the desired visual. This time, we were tasked with plotting daily cases and deaths per 100,000 people for 4 countries on the same continent.

We started by creating a function which would take our dataset containing dates, countries, total cases, and deaths by date, as well as a string input for which country we wanted to analyze. The function would return the new cases and deaths per 100,000 people (population being country specific), by date.

We ran this for four arbitrary countries from Europe, those being Germany, France, Poland, and Romania. Instead of merging, we just combined the data together with a row bind function. This essentially stacked one dataset on top of the other. Like with the plots before this (top 10 plots), it was aesthetically efficient to create two separate plots. Thankfully, we didn't need to rearrange the rows by date and country, as the plotting software automatically accounts for date.

Another difference from the plots in most sections was the use of geom_smooth() to plot the lines. The other plots mostly use geom_line(), but the daily increases and decreases of the cases and deaths created too much static and made for visual interpretation difficult.