R Capstone Project: COVID-19 Data Part 1

Carl Klein

2023-08-11

Required Packages

Part 1 - Basic Exploration of US Data The New York Times (the Times) has aggregated reported COVID-19 data from state and local governments and health departments since 2020 and provides public access through a repository on GitHub. One of the data sets provided by the Times is county-level data for cumulative cases and deaths each day. This will be your primary data set for the first two parts of your analysis.

County-level COVID data from 2020, 2021, and 2022 has been imported below. Each row of data reports the cumulative number of cases and deaths for a specific county each day. A FIPS code, a standard geographic identifier, is also provided which you will use in Part 2 to construct a map visualization at the county level for a state.

Additionally, county-level population estimates reported by the US Census Bureau has been imported as well. You will use these estimates to calculate statistics per 100,000 people.

```
# Import New York Times COVID-19 data
# Import Population Estimates from US Census Bureau

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

us_population_estimates <- read_csv("data/fips_population_estimates.csv")
```

Question 1 Your first task is to combine and tidy the 2020, 2021, and 2022 COVID data sets and find the total deaths and cases for each day since March 15, 2020 (2020-03-15). The data sets provided from the NY Times also includes statistics from Puerto Rico, a US territory. You may remove these observations from the data as they will not be needed for your analysis. Once you have tidied the data, find the total COVID-19 cases and deaths since March 15, 2020. Write a sentence or two after the code block communicating your results. Use inline code to include the `max_date`, `us_total_cases`, and `us_total_deaths` variables. To write inline code use `r`.

```
# 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')

# starting with 3/15/2020, find the total cases and deaths by date
df_sums <- df %>%
  group_by(date) %>%
  summarise(total_cases = sum(cases), total_deaths = sum(deaths)) %>%
  filter(date >= '2020-03-15')

# max_date: the last date entry of the dataset
```

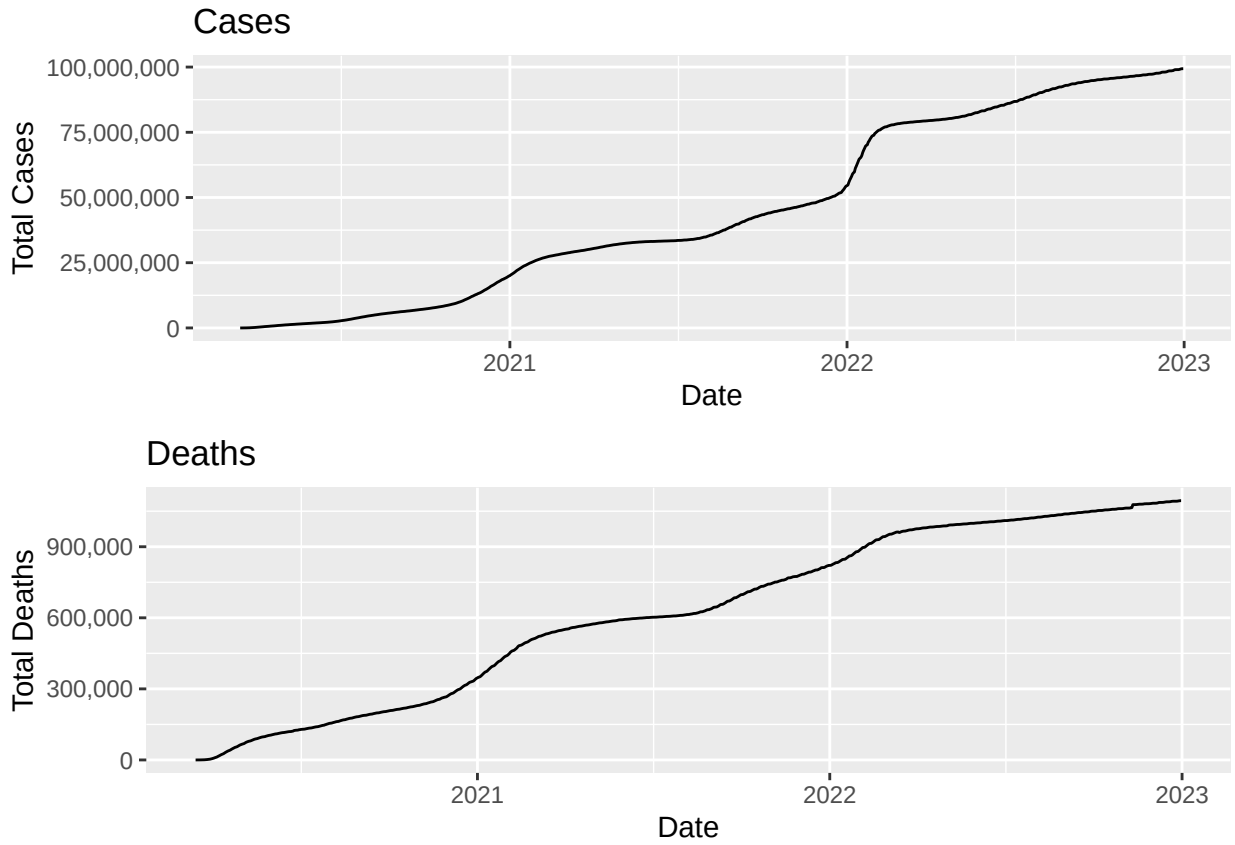
```r
max_date <- max(df_sums$date)

# since the cases are cumulative, by using the maximum dates for a combination
# of state and fips code, we can sum the cases and deaths through the dataset.
us_total_cases <- max(df_sums$total_cases)
us_total_deaths <- max(df_sums$total_deaths)

df_sums
```

```
## # A tibble: 1,022 x 3
##    date       total_cases total_deaths
##    <date>           <dbl>        <dbl>
##  1 2020-03-15        3595           68
##  2 2020-03-16        4502           91
##  3 2020-03-17        5901          117
##  4 2020-03-18        8345          162
##  5 2020-03-19       12387          212
##  6 2020-03-20       17998          277
##  7 2020-03-21       24507          359
##  8 2020-03-22       33050          457
##  9 2020-03-23       43474          577
## 10 2020-03-24       53899          783
## # i 1,012 more rows
```

As of December 31, 2022, out of 99,374,764 total case in the US, there have been 1,094,296 deaths.

Question 2 Create a visualization for the total number of deaths and cases in the US since March 15, 2020. Before you create your visualization, review the types of plots you can create using the ggplot2 library and think about which plots would be effective in communicating your results. After you have created your visualization, write a few sentences describing your visualization. How could the plot be interpreted? Could it be misleading?

```r
# Create a visualization for the total number of US cases and deaths
# since March 15, 2020.

# plot for the total cases
plot1 <- ggplot(data = df_sums) +
  geom_line(aes(x = date, y = total_cases)) +
  labs(title = 'Cases') +
  xlab('Date') +
  ylab('Total Cases') +
  scale_y_continuous(labels = scales::comma)

# plot for the total deaths
plot2 <- ggplot(data = df_sums) +
  geom_line(aes(x = date, y = total_deaths)) +
  labs(title = 'Deaths') +
  xlab('Date') +
  ylab('Total Deaths') +
  scale_y_continuous(labels = scales::comma)

# put the plots in a grid together (uses gridExtra)
grid.arrange(plot1, plot2)
```



Initially, I wanted to display both lines on the same plot, however the ratio of cases to deaths was of such a high magnitude that the death's curve looked flat. I could've scaled the death's data with something like a logarithmic method, but I think that would've been misleading as it would've looked like the number of deaths were close to the number of cases. One thing that is apparent in these graphs is that the peaks seem to increase almost together.

Still, these aren't perfect. One potential outcome is if the reader glances over the differences of the scaling on the y-axis. The Cases y-axis reaches almost 100 million while the Death y-axis barely breaks 1 million.

Question 3 While it is important to know the total deaths and cases throughout the COVID-19 pandemic, it is also important for local and state health officials to know the the number of new cases and deaths each day to understand how rapidly the virus is spreading. Using the table you created in Question 1, calculate the number of new deaths and cases each day and a seven-day average of new deaths and cases. Once you have organized your data, find the days that saw the largest number of new cases and deaths. Write a sentence or two after the code block communicating your results.

```
# Create a new table, based on the table from Question 1, and calculate the
# number of new deaths and cases each day and a seven day average of new deaths
# and cases.

# calculate the difference in cases
new_cases = c()
for (i in 1:length(df_sums$total_cases) - 1) {
  new_cases[i] = df_sums$total_cases[i + 1] - df_sums$total_cases[i]
}
new_cases = c(0, new_cases)
```

```r
# calculate the difference in deaths
new_deaths = c()
for (i in 1:length(df_sums$total_deaths) - 1) {
  new_deaths[i] = df_sums$total_deaths[i + 1] - df_sums$total_deaths[i]
}
new_deaths = c(0, new_deaths)

# calculate the 7 day average for new cases
cases_roll_7 <- df_sums %>%
  rollmean(x = new_cases, k = 7, align = 'right', fill = NA) %>%
  lag(n = 1)

# calculate the 7 day average for new deaths
death_roll_7 <- df_sums %>%
  rollmean(x = new_deaths, k = 7, align = 'right', fill = NA) %>%
  lag(n = 1)

# add the new and rolling data
df_sums <- df_sums %>%
  mutate(new_cases, new_deaths, cases_roll_7, death_roll_7)

max_new_cases_date <- df_sums %>%
  filter(new_cases == max(new_cases)) %>%
  select(date)
max_new_cases_date = max(max_new_cases_date$date)
max_new_deaths_date <- df_sums %>%
  filter(new_deaths == max(new_deaths)) %>%
  select(date)
max_new_deaths_date = max(max_new_deaths_date$date)

df_sums
```

```
## # A tibble: 1,022 x 7
##    date       total_cases total_deaths new_cases new_deaths cases_roll_7
##    <date>           <dbl>        <dbl>     <dbl>      <dbl>        <dbl>
##  1 2020-03-15        3595           68         0          0           NA
##  2 2020-03-16        4502           91       907         23           NA
##  3 2020-03-17        5901          117      1399         26           NA
##  4 2020-03-18        8345          162      2444         45           NA
##  5 2020-03-19       12387          212      4042         50           NA
##  6 2020-03-20       17998          277      5611         65           NA
##  7 2020-03-21       24507          359      6509         82           NA
##  8 2020-03-22       33050          457      8543         98        2987.
##  9 2020-03-23       43474          577     10424        120        4208.
## 10 2020-03-24       53899          783     10425        206        5567.
## # i 1,012 more rows
## # i 1 more variable: death_roll_7 <dbl>
```

After using some indexing methods, I calculated the new cases and deaths, and then used that data to find the 7 day rolling average for both.

The highest number of new cases occured on January 10, 2022, while the day with the highest number of deaths occured on November 11, 2022.

```
# Create a new table, based on the table from Question 3, and calculate the
# number of new deaths and cases per 100,000 people each day and a seven day
# average of new deaths and cases per 100,000 people.

# Hint: To calculate per 100,000 people, first tidy the population estimates
# data and calculate the US population in 2020 and 2021. Then, you will need to
# divide  each statistic by the estimated population and then multiply by 100,000.

# us population estimates
us_pops <- us_population_estimates %>%
  group_by(Year) %>%
  summarize(annual_est = sum(Estimate))

# per 100,000 estimates
df_per_100 <- df_sums %>%
  mutate(us_annual_pop = case_when(
    grepl('2020', date) ~ as.integer(us_pops[1,2]),
    grepl('2021', date) ~ as.integer(us_pops[2,2]))) %>%
  mutate(cases_100 = (total_cases / us_annual_pop) * 100000) %>%
  mutate(deaths_100 = (total_deaths / us_annual_pop) * 100000) %>%
  mutate(new_cases_100 = (new_cases / us_annual_pop) * 100000) %>%
  mutate(new_deaths_100 = (new_deaths / us_annual_pop) * 100000) %>%
  mutate(roll_cases_100 = (cases_roll_7 / us_annual_pop) * 100000) %>%
  mutate(roll_deaths_100 = (death_roll_7 / us_annual_pop) * 100000) %>%
  select(date, cases_100, deaths_100,
         new_cases_100, new_deaths_100,
         roll_cases_100, roll_deaths_100,
         us_annual_pop) %>%
  filter(!is.na(us_annual_pop)) %>%
  select(-one_of('us_annual_pop'))

df_per_100
```

Question 4

```
## # A tibble: 657 x 7
##    date       cases_100 deaths_100 new_cases_100 new_deaths_100 roll_cases_100
##    <date>         <dbl>      <dbl>         <dbl>          <dbl>          <dbl>
##  1 2020-03-15      1.08     0.0205         0             0               NA
##  2 2020-03-16      1.36     0.0275         0.274         0.00694         NA
##  3 2020-03-17      1.78     0.0353         0.422         0.00784         NA
##  4 2020-03-18      2.52     0.0489         0.737         0.0136          NA
##  5 2020-03-19      3.74     0.0640         1.22          0.0151          NA
##  6 2020-03-20      5.43     0.0836         1.69          0.0196          NA
##  7 2020-03-21      7.39     0.108          1.96          0.0247          NA
##  8 2020-03-22      9.97     0.138          2.58          0.0296           0.901
##  9 2020-03-23     13.1      0.174          3.14          0.0362           1.27
## 10 2020-03-24     16.3      0.236          3.14          0.0621           1.68
## # i 647 more rows
## # i 1 more variable: roll_deaths_100 <dbl>
```

There were a couple of steps involved in this. First, we needed to sum the us population estimate data between 2020 and 2021. Using this data, we could separate the data we've worked on in previous steps between 2020 and 2021 years, assigning the 2020 and 2021 us annual estimates to those rows. From here, a

quick mutate was needed to turn the variables into per 100,000 variables.

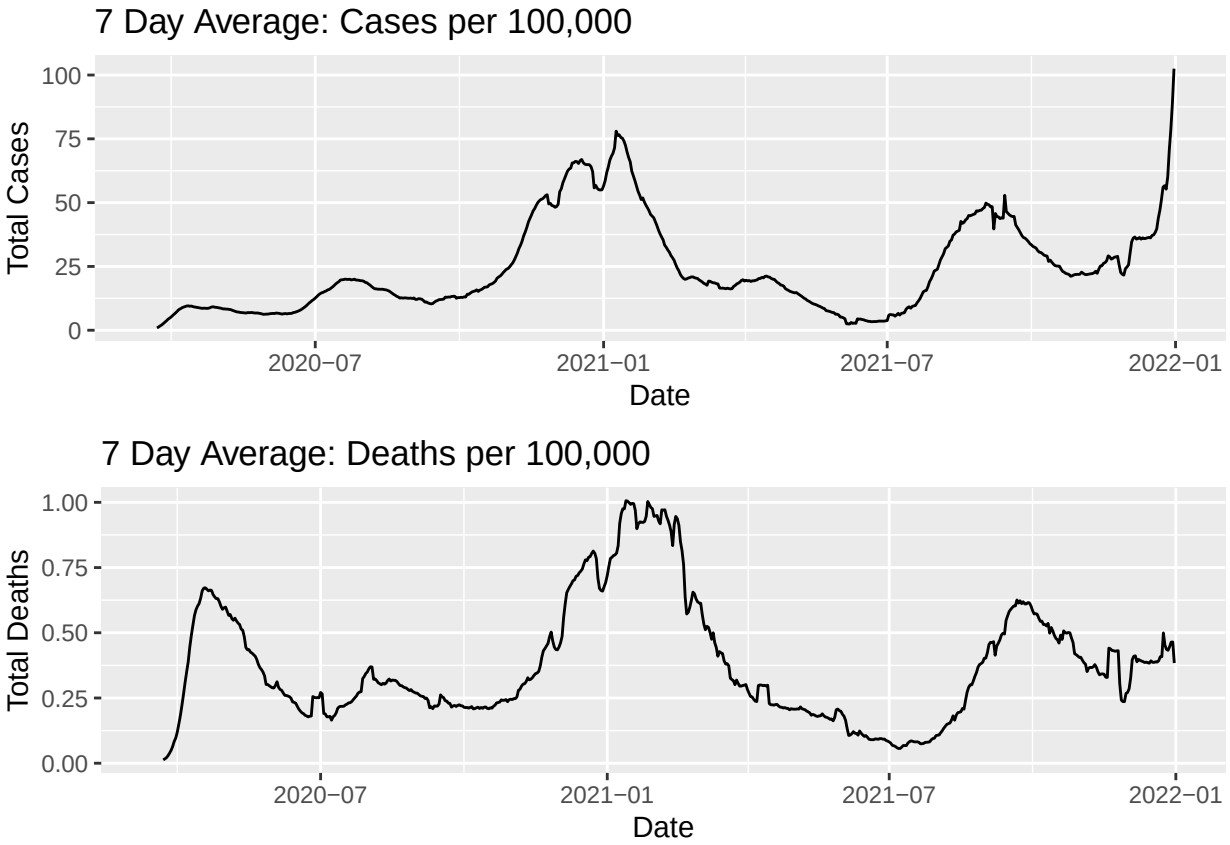
Since the us population estimates were not applicable to the 2022 year, we filtered the 2022 rows out.

Straying from the assignment instructions, it would be interesting to use the population estimates per county to get more accurate figures.

```
# Create a visualization to compare the seven-day average cases and deaths per
# $100,000 people.

# plot for the total cases
plot1 <- ggplot(data = df_per_100) +
  geom_line(aes(x = date, y = roll_cases_100)) +
  labs(title = '7 Day Average: Cases per 100,000') +
  xlab('Date') +
  ylab('Total Cases') +
  scale_y_continuous(labels = scales::comma)

# plot for the total deaths
plot2 <- ggplot(data = df_per_100) +
  geom_line(aes(x = date, y = roll_deaths_100)) +
  labs(title = '7 Day Average: Deaths per 100,000') +
  xlab('Date') +
  ylab('Total Deaths') +
  scale_y_continuous(labels = scales::comma)

# put the plots in a grid together (uses gridExtra)
grid.arrange(plot1, plot2)
```

Question 5

```
## Warning: Removed 7 rows containing missing values (`geom_line()`).
## Removed 7 rows containing missing values (`geom_line()`).
```



Using a similar format to the total cases and deaths, this maps the 7 day average per 100,000 people.