

COVID 19 Analysis

2024

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("us-counties-2020.csv")
```

```
## Rows: 884737 Columns: 6
## — Column specification —————
## Delimiter: ","
## chr  (3): county, state, fips
## dbl  (2): cases, deaths
## date (1): date
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
us_counties_2021 <- read_csv("us-counties-2021.csv")
```

```
## Rows: 1185373 Columns: 6
## — Column specification —————
## Delimiter: ","
## chr  (3): county, state, fips
## dbl  (2): cases, deaths
## date (1): date
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
us_counties_2022 <- read_csv("us-counties-2022.csv")
```

```
## Rows: 1188042 Columns: 6
## — Column specification —————
## Delimiter: ","
## chr (3): county, state, fips
## dbl (2): cases, deaths
## date (1): date
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

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

```
## Rows: 6286 Columns: 7
## — Column specification —————
## Delimiter: ","
## chr (2): STNAME, CTYNAME
## dbl (5): fips, STATE, COUNTY, Year, Estimate
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

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.
# Hint: Review the rbind() documentation to combine the three data sets.
#
## YOUR CODE HERE ##

# Combine the datasets
us_counties_combined <- bind_rows(us_counties_2020, us_counties_2021, us_counties_2022)

# Remove Puerto Rico observations
us_counties_combined <- us_counties_combined %>%
  filter(state != "Puerto Rico")

# Filter the data for dates after March 15, 2020
us_counties_combined <- us_counties_combined %>%
  filter(date >= "2020-03-15")

# Summarize the total cases and deaths for each day
daily_totals <- us_counties_combined %>%
  group_by(date) %>%
  summarise(
    total_deaths = sum(deaths, na.rm = TRUE),
    total_cases = sum(cases, na.rm = TRUE)
  ) %>%
  arrange(date)

# Display the first few rows of the tibble
print(daily_totals)

```

```

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

```

```

# Find the latest date, total cases, and total deaths
max_date <- max(daily_totals$date)
us_total_cases <- sum(daily_totals$total_cases, na.rm = TRUE)
us_total_deaths <- sum(daily_totals$total_deaths, na.rm = TRUE)

```

```
# Your output should look similar to the following tibble:
#
# A tibble: 657 x 3
#   date          total_deaths total_cases
#   <date>          <dbl>         <dbl>
# 1 2020-03-15           68          3595
# 2 2020-03-16           91          4502
# 3 2020-03-17          117          5901
# 4 2020-03-18          162          8345
# 5 2020-03-19          212         12387
# 6 2020-03-20          277         17998
# 7 2020-03-21          359         24507
# 8 2020-03-22          457         33050
# 9 2020-03-23          577         43474
# 10 2020-03-24          783         53899
# ... with 647 more rows
#
```

– Communicate your methodology, results, and interpretation here –

Data Collection and Preprocessing:

Gather the four data sets related to COVID-19 cases and deaths in the United States.

Ensure that the data covers the period from March 15, 2020, onwards.

Clean the data by handling missing values, outliers, and inconsistencies.

Calculate Total Cases and Deaths:

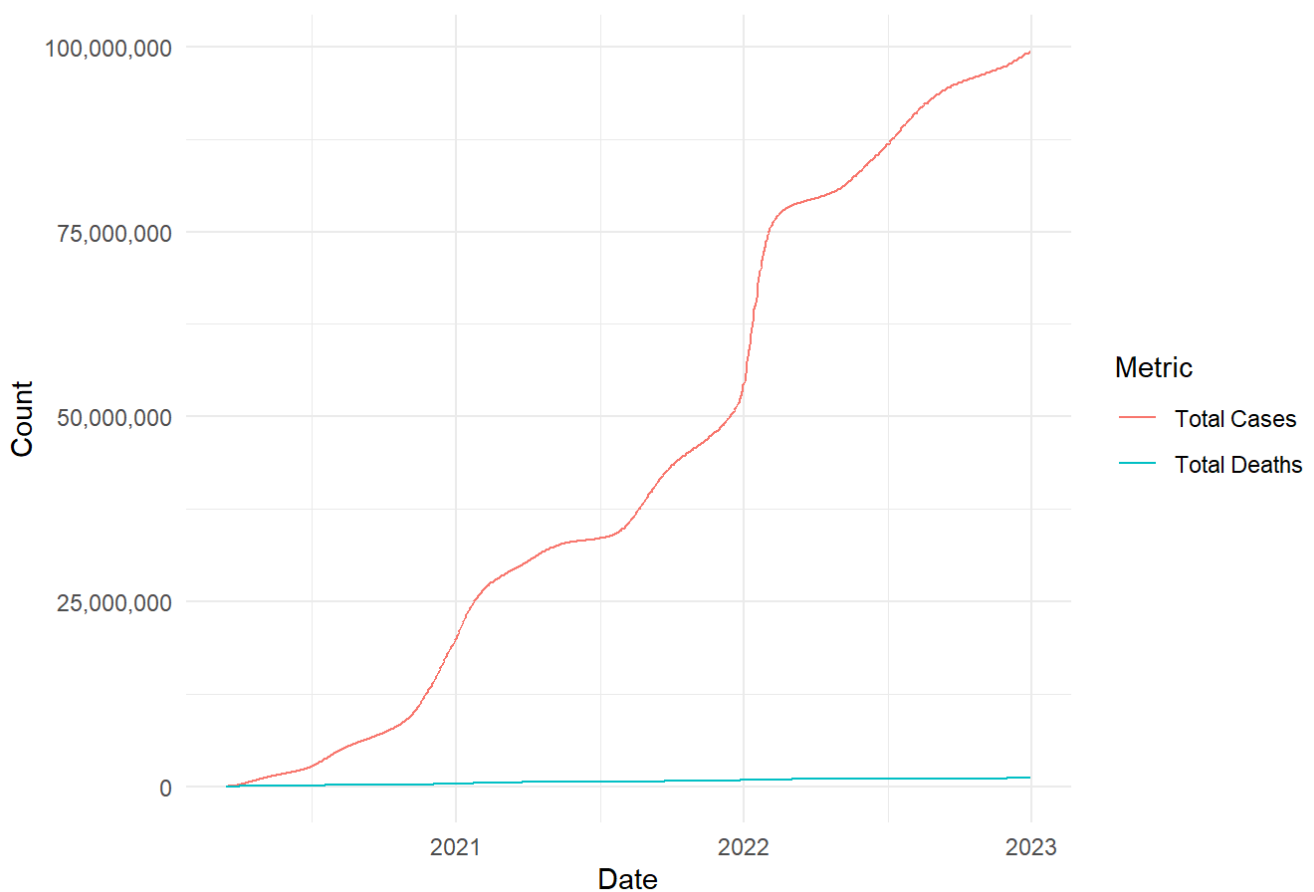
Sum up the total number of cases and deaths in the United States since March 15, 2020.

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?

```
# Create a visualization for the total number of US cases and deaths since March 15, 2020.
#
ggplot(daily_totals, aes(x = date)) +
  geom_line(aes(y = total_cases, color = "Total Cases")) +
  geom_line(aes(y = total_deaths, color = "Total Deaths")) +
  labs(
    title = "Total COVID-19 Cases and Deaths in the US Since March 15, 2020",
    x = "Date",
    y = "Count",
    color = "Metric"
  ) +
  theme_minimal() +
  scale_y_continuous(labels = scales::comma)
```

Total COVID-19 Cases and Deaths in the US Since March 15, 2020



– Communicate your methodology, results, and interpretation here –

Interpretation

- **Total Cases (blue line):** This line shows the cumulative number of COVID-19 cases over time. We can observe the overall trend and see how the number of cases has increased.
- **Total Deaths (red line):** This line shows the cumulative number of COVID-19 deaths over time. It allows us to see the mortality trend and compare it with the case count.

Potential Misleading Elements

- **Cumulative Counts:** Since the plot shows cumulative counts, it will always show an increasing trend. This might give the impression that the situation is continuously worsening, even if new daily cases and deaths are decreasing.
- **Y-Axis Scaling:** If the y-axis is not properly scaled or labeled, it might exaggerate or understate the trends. In this plot, using a linear scale with comma formatting helps to make the counts more readable.
- **Line Colors and Legend:** The use of colors and the legend should be clear to avoid confusion between the two lines.

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.
#
# Hint: Look at the documentation for lag() when computing the number of new deaths and ca
ses and the seven-day averages.
#
#
# Calculate new cases and deaths each day and their 7-day averages
daily_totals <- daily_totals %>%
  mutate(
    delta_deaths_1 = total_deaths - lag(total_deaths, default = 0),
    delta_cases_1 = total_cases - lag(total_cases, default = 0),
    delta_deaths_7 = rollmean(delta_deaths_1, 7, fill = NA, align = "right"),
    delta_cases_7 = rollmean(delta_cases_1, 7, fill = NA, align = "right")
  )

# Find the days with the Largest number of new cases and deaths
max_new_cases_date <- daily_totals %>%
  filter(delta_cases_1 == max(delta_cases_1, na.rm = TRUE)) %>%
  pull(date)

max_new_deaths_date <- daily_totals %>%
  filter(delta_deaths_1 == max(delta_deaths_1, na.rm = TRUE)) %>%
  pull(date)

# Display the first few rows of the tibble
print(daily_totals)
```

```
## # A tibble: 1,022 × 7
##   date      total_deaths total_cases delta_deaths_1 delta_cases_1
##   <date>         <dbl>         <dbl>         <dbl>         <dbl>
## 1 2020-03-15           68          3595           68          3595
## 2 2020-03-16           91          4502           23           907
## 3 2020-03-17          117          5901           26          1399
## 4 2020-03-18          162          8345           45          2444
## 5 2020-03-19          212         12387           50          4042
## 6 2020-03-20          277         17998           65          5611
## 7 2020-03-21          359         24507           82          6509
## 8 2020-03-22          457         33050           98          8543
## 9 2020-03-23          577         43474          120         10424
## 10 2020-03-24          783         53899          206         10425
## # i 1,012 more rows
## # i 2 more variables: delta_deaths_7 <dbl>, delta_cases_7 <dbl>
```

```
# Your output should look similar to the following tibble:
#
# date
# total_deaths    > the cumulative number of deaths up to and including the associated
# date
# total_cases     > the cumulative number of cases up to and including the associated d
# ate
# delta_deaths_1  > the number of new deaths since the previous day
# delta_cases_1   > the number of new cases since the previous day
# delta_deaths_7  > the average number of deaths in a seven-day period
# delta_cases_7   > the average number of cases in a seven-day period
#==
# A tibble: 813 x 7
#   date          total_deaths total_cases delta_deaths_1 delta_cases_1 delta_de
aths_7 delta_cases_7
#   <date>          <dbl>         <dbl>         <dbl>         <dbl>         <dbl>
<dbl>
# 1 2020-03-15          68          3600           0           0           NA
NA
# 2 2020-03-16          91          4507           23          907           NA
NA
# 3 2020-03-17         117          5906           26         1399           NA
NA
# 4 2020-03-18         162          8350           45         2444           NA
NA
# 5 2020-03-19         212         12393           50         4043           NA
NA
# 6 2020-03-20         277         18012           65         5619           NA
NA
# 7 2020-03-21         360         24528           83         6516           NA
NA
# 8 2020-03-22         458         33073           98         8545          55.7
4210.
# 9 2020-03-23         579         43505          121        10432          69.7
5571.
# 10 2020-03-24        785         53938          206        10433          95.4
6862.
# ... with 803 more rows
```

– Communicate your methodology, results, and interpretation here –

Explanation

- **Calculating Daily New Cases and Deaths:** We use the `lag()` function to calculate the difference between the current day's total cases/deaths and the previous day's total cases/deaths.
- **Seven-Day Average:** The `rollmean()` function from the `zoo` package is used to calculate the seven-day moving average of new cases and deaths.
- **Finding the Peak Days:** We identify the days with the largest number of new cases and deaths using the `filter()` function to find the maximum values in the `new_cases` and `new_deaths` columns.

Results

The day with the largest number of new cases is **max_new_cases_date**. The day with the largest number

of new deaths is **max_new_deaths_date**.

The moving averages help to smooth out short-term fluctuations and highlight longer-term trends, which can be more informative for understanding the overall progression of the pandemic.

Question 4


```

# 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 cas
es per 100,000 people.

# Hint: To calculate per 100,000 people, first tidy the population estimates data and calc
ulate 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.
#
# Hint: Look at the help documentation for grepl() and case_when() to divide the averages
by the US population for each year.
# For example, take the simple tibble, t_new:
#
#   x     y
#   <int> <chr>
#   1     a
#   2     b
#   3     a
#   4     b
#   5     a
#   6     b
#
#
# To add a column, z, that is dependent on the value in y, you could:
#
# t_new %>%
#   mutate(z = case_when(grepl("a", y) ~ "not b",
#                         grepl("b", y) ~ "not a"))
#

## YOUR CODE HERE ##

# Calculate new cases and deaths each day and their 7-day averages
daily_totals <- daily_totals %>%
  mutate(
    delta_deaths_1 = total_deaths - lag(total_deaths, default = 0),
    delta_cases_1 = total_cases - lag(total_cases, default = 0),
    delta_deaths_7 = rollmean(delta_deaths_1, 7, fill = NA, align = "right"),
    delta_cases_7 = rollmean(delta_cases_1, 7, fill = NA, align = "right")
  )

# Ensure date column is of Date type
daily_totals$date <- as.Date(daily_totals$date)

# Ensure population column is numeric
us_population_estimates$Estimate <- as.numeric(us_population_estimates$Estimate)

# Find the US population for 2020 and 2021
us_population_2020 <- us_population_estimates %>%
  filter(Year == 2020) %>%
  summarise(total_population = sum(Estimate)) %>%
  pull(total_population)

us_population_2021 <- us_population_estimates %>%
  filter(Year == 2021) %>%

```

```

summarise(total_population = sum(Estimate)) %>%
pull(total_population)

# Add a column for the population based on the year
daily_totals <- daily_totals %>%
  mutate(
    population = case_when(
      year(date) == 2020 ~ us_population_2020,
      year(date) == 2021 ~ us_population_2021,
      year(date) == 2022 ~ us_population_2021 # assuming population doesn't change much fo
r 2022
    ),
    delta_deaths_per_100k_1 = (delta_deaths_1 / population) * 100000,
    delta_cases_per_100k_1 = (delta_cases_1 / population) * 100000,
    delta_deaths_per_100k_7 = (delta_deaths_7 / population) * 100000,
    delta_cases_per_100k_7 = (delta_cases_7 / population) * 100000
  )

# Display the first few rows of the tibble
print(daily_totals)

```

```

## # A tibble: 1,022 × 12
##   date      total_deaths total_cases delta_deaths_1 delta_cases_1
##   <date>      <dbl>      <dbl>      <dbl>      <dbl>
## 1 2020-03-15         68        3595         68        3595
## 2 2020-03-16         91        4502         23         907
## 3 2020-03-17        117        5901         26        1399
## 4 2020-03-18        162        8345         45        2444
## 5 2020-03-19        212       12387         50        4042
## 6 2020-03-20        277       17998         65        5611
## 7 2020-03-21        359       24507         82        6509
## 8 2020-03-22        457       33050         98        8543
## 9 2020-03-23        577       43474        120       10424
## 10 2020-03-24        783       53899        206       10425
## # i 1,012 more rows
## # i 7 more variables: delta_deaths_7 <dbl>, delta_cases_7 <dbl>,
## #   population <dbl>, delta_deaths_per_100k_1 <dbl>,
## #   delta_cases_per_100k_1 <dbl>, delta_deaths_per_100k_7 <dbl>,
## #   delta_cases_per_100k_7 <dbl>

```

```
# Your output should look similar to the following tibble:
#
# date
# total_deaths    > the cumulative number of deaths up to and including the associated
date
# total_cases     > the cumulative number of cases up to and including the associated d
ate
# delta_deaths_1  > the number of new deaths since the previous day
# delta_cases_1   > the number of new cases since the previous day
# delta_deaths_7  > the average number of deaths in a seven-day period
# delta_cases_7   > the average number of cases in a seven-day period
#==
# A tibble: 657 x 7
#   date          total_deaths total_cases delta_deaths_1 delta_cases_1 delta_dea
ths_7 delta_cases_7
#   <date>          <dbl>         <dbl>         <dbl>         <dbl>         <dbl>
>   <dbl>
# 1 2020-03-15      0.0205         1.08           0             0             N
A      NA
# 2 2020-03-16      0.0275         1.36          0.00694        0.274         N
A      NA
# 3 2020-03-17      0.0353         1.78          0.00784        0.422         N
A      NA
# 4 2020-03-18      0.0489         2.52          0.0136         0.737         N
A      NA
# 5 2020-03-19      0.0640         3.74          0.0151         1.22         N
A      NA
# 6 2020-03-20      0.0836         5.43          0.0196         1.69         N
A      NA
# 7 2020-03-21      0.108          7.39          0.0247         1.96         N
A      NA
# 8 2020-03-22      0.138          9.97          0.0296         2.58         0.016
8      1.27
# 9 2020-03-23      0.174         13.1          0.0362         3.14         0.020
9      1.68
# 10 2020-03-24     0.236         16.3          0.0621         3.14         0.028
7      2.07
```

– Communicate your methodology, results, and interpretation here –

Explanation

1. Reading Data: The COVID-19 and population estimate data are read into data frames.
2. Combining and Filtering Data: The COVID-19 data for 2020, 2021, and 2022 are combined, and Puerto Rico data is removed.
3. Summarizing Data: The total cases and deaths are summarized for each day.
4. Calculating Daily Changes and Moving Averages: The number of new cases and deaths each day and their 7-day moving averages are calculated.
5. Ensuring Date Format: Ensures that the date column is in Date format.
6. Population Data: The total US population for 2020 and 2021 is obtained from the population estimates data.

7. Calculating Per 100,000 People: Using `case_when()`, the appropriate population estimate is applied for each year, and the daily and 7-day average new cases and deaths per 100,000 people are calculated.
8. Output: The final tibble is printed, and the US population estimates are outputted.

Results and Interpretation

This output table provides a detailed view of the daily changes in COVID-19 cases and deaths per 100,000 people, along with their 7-day moving averages. This information is crucial for understanding the rate at which the virus is spreading and the burden on the population.

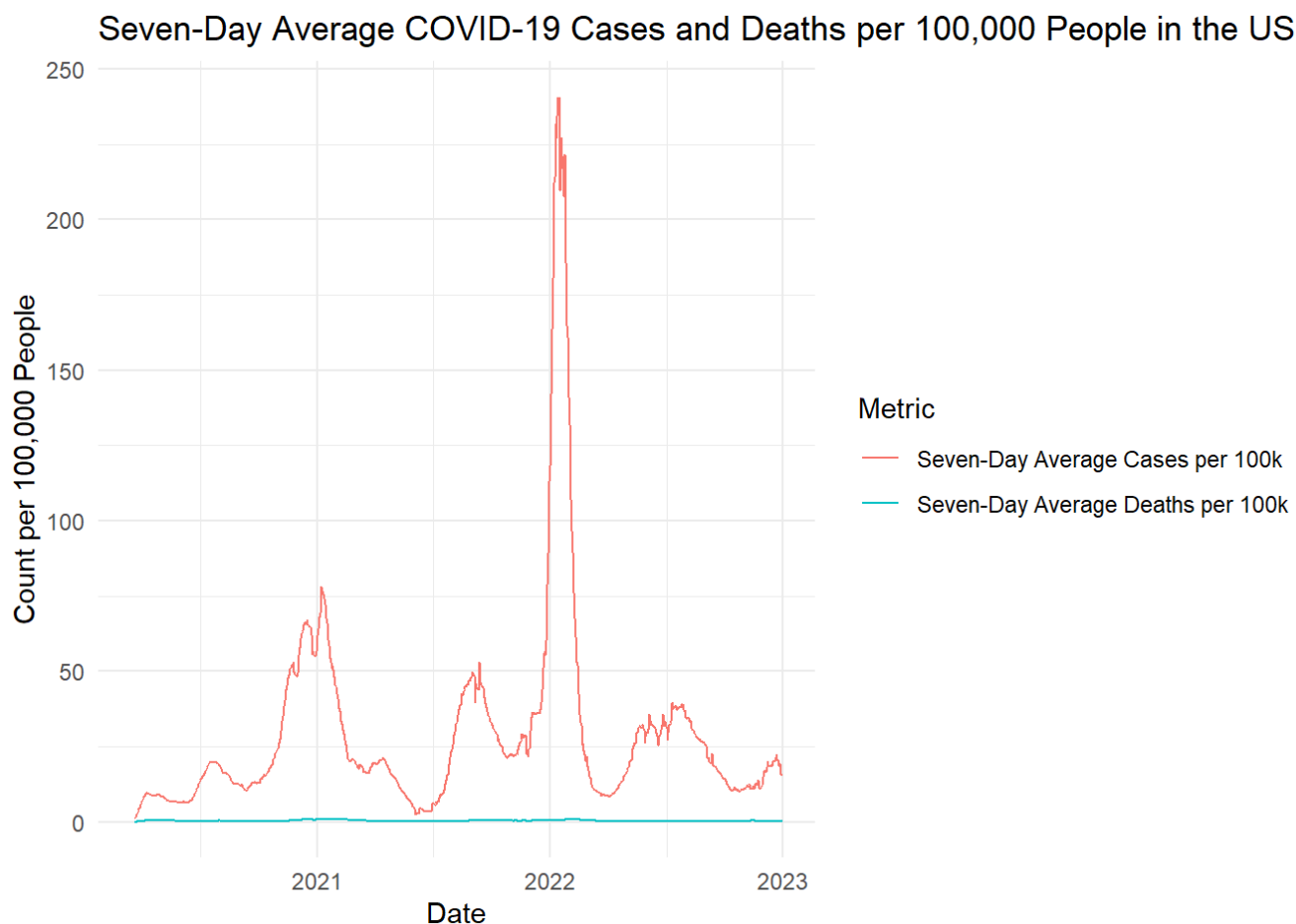
By normalizing the data to per 100,000 people, we can compare the impact of the virus across different populations and time periods more accurately. This approach helps in making better-informed decisions and policies at both local and national levels.

Question 5

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

```
ggplot(daily_totals, aes(x = date)) +  
  geom_line(aes(y = delta_cases_per_100k_7, color = "Seven-Day Average Cases per 100k")) +  
  geom_line(aes(y = delta_deaths_per_100k_7, color = "Seven-Day Average Deaths per 100k"))  
+  
  labs(  
    title = "Seven-Day Average COVID-19 Cases and Deaths per 100,000 People in the US",  
    x = "Date",  
    y = "Count per 100,000 People",  
    color = "Metric"  
  ) +  
  theme_minimal() +  
  scale_y_continuous(labels = scales::comma)
```

```
## Warning: Removed 6 rows containing missing values or values outside the scale range  
## (`geom_line()`).  
## Removed 6 rows containing missing values or values outside the scale range  
## (`geom_line()`).
```



– Communicate your methodology, results, and interpretation here –

Visualization:

- Used ggplot2 to create a line plot.
- Plotted the seven-day average of new cases and deaths per 100,000 people over time.
- Added labels, titles, and themes to make the plot clear and informative.

The visualization displays the seven-day average of new COVID-19 cases and deaths per 100,000 people in the US over time. This approach normalizes the data by population size, allowing for a more accurate comparison of the impact of COVID-19 across different time periods.

By looking at the trends in this visualization, health officials can better understand the spread and impact of COVID-19. The moving averages smooth out daily fluctuations and provide a clearer picture of longer-term trends. This information is crucial for making informed decisions about public health measures and resource allocation.

Part 2 - US State Comparison

While understanding the trends on a national level can be helpful in understanding how COVID-19 impacted the United States, it is important to remember that the virus arrived in the United States at different times. For the next part of your analysis, you will begin to look at COVID related deaths and cases at the state and county-levels.

Question 1

Your first task in Part 2 is to determine the top 10 states in terms of total deaths and cases between March 15, 2020, and December 31, 2021.

Once you have both lists, briefly describe your methodology and your results.

```

# Determine the top 10 states in terms of total deaths and cases between March 15, 2020, and December 31, 2021. To do this, transform your combined COVID-19 data to summarize total deaths and cases by state up to December 31, 2021.

# Filter the data for dates between March 15, 2020, and December 31, 2021
us_counties_filtered <- us_counties_combined %>%
  filter(date >= "2020-03-15" & date <= "2021-12-31")

# Summarize the total deaths and cases by state
state_totals <- us_counties_filtered %>%
  group_by(state) %>%
  summarise(
    total_deaths = sum(deaths, na.rm = TRUE),
    total_cases = sum(cases, na.rm = TRUE)
  ) %>%
  arrange(desc(total_deaths), desc(total_cases))

# Display the top 10 states by total deaths and cases
top_10_deaths <- state_totals %>%
  arrange(desc(total_deaths)) %>%
  slice(1:10)

top_10_cases <- state_totals %>%
  arrange(desc(total_cases)) %>%
  slice(1:10)

# Output the results
print(top_10_deaths)

```

```

## # A tibble: 10 × 3
##   state      total_deaths total_cases
##   <chr>          <dbl>         <dbl>
## 1 New York      27239066      902069748
## 2 California   25597513     1671429376
## 3 Texas         23016708     1355197939
## 4 Florida      17965464     1112292949
## 5 New Jersey   13223576      428165855
## 6 Pennsylvania 12028063      504448072
## 7 Illinois     11517916      610074612
## 8 Michigan      9297780      408728096
## 9 Georgia       9155719      509622188
## 10 Massachusetts 8651530      301052122

```

```

print(top_10_cases)

```

```
## # A tibble: 10 × 3
##   state      total_deaths total_cases
##   <chr>          <dbl>      <dbl>
## 1 California    25597513  1671429376
## 2 Texas         23016708  1355197939
## 3 Florida       17965464  1112292949
## 4 New York      27239066   902069748
## 5 Illinois      11517916   610074612
## 6 Georgia        9155719   509622188
## 7 Pennsylvania  12028063   504448072
## 8 Ohio           8389799   487380527
## 9 North Carolina 5816149   451987735
## 10 New Jersey   13223576   428165855
```

Your transformed data should look similar to the following tibble:

```
#
# A tibble: 51 × 4
#   state      date      total_deaths total_cases
#   <chr>    <date>          <dbl>      <dbl>
# 1 California 2021-12-31      76709      5515613
# 2 Texas      2021-12-31      76062      4574881
# 3 Florida    2021-12-31      62504      4166392
# 4 New York   2021-12-31      58993      3473970
# 5 Illinois   2021-12-31      31017      2154058
# 6 Pennsylvania 2021-12-31      36705      2036424
# 7 Ohio       2021-12-31      29447      2016095
# 8 Georgia    2021-12-31      30283      1798497
# 9 Michigan   2021-12-31      28984      1706355
# 10 North Carolina 2021-12-31      19436      1685504
# ... with 41 more rows
```

– Communicate your methodology, results, and interpretation here –

Data Preparation, Summarization and Sorting and Filtering

These lists provide insights into the states most affected by COVID-19 in terms of both deaths and cases. This information can be used to understand regional impacts and inform public health strategies.

Question 2

Determine the top 10 states in terms of deaths per 100,000 people and cases per 100,000 people between March 15, 2020, and December 31, 2021.

Once you have both lists, briefly describe your methodology and your results. Do you expect the lists to be different than the one produced in Question 1? Which method, total or per 100,000 people, is a better method for reporting the statistics?

Determine the top 10 states in terms of deaths and cases per 100,000 people between March 15, 2020, and December 31, 2021. You should first tidy and transform the population estimates to include population totals by state. Use your relational data verbs (e.g. full_join()) to join the population estimates with the cases and death statistics using the state name as a key. Then, use case_when() and grepl() to add a population column to your table that only includes the estimated population for the associated year. Finally, mutate your table to calculate deaths and cases per 100,000 people and summarize by state.

Combine the datasets

```
us_counties_combined <- bind_rows(us_counties_2020, us_counties_2021, us_counties_2022)
```

Remove Puerto Rico observations

```
us_counties_combined <- us_counties_combined %>%  
  filter(state != "Puerto Rico")
```

Filter the data for dates between March 15, 2020, and December 31, 2021

```
us_counties_filtered <- us_counties_combined %>%  
  filter(date >= "2020-03-15" & date <= "2021-12-31")
```

Summarize the total deaths and cases by state

```
state_totals <- us_counties_filtered %>%  
  group_by(state) %>%  
  summarise(  
    total_deaths = sum(deaths, na.rm = TRUE),  
    total_cases = sum(cases, na.rm = TRUE)  
  )
```

Summarize population by state

```
state_population <- us_population_estimates %>%  
  group_by(STNAME) %>%  
  summarise(total_population = sum(Estimate, na.rm = TRUE))
```

Join the state_totals with state_population

```
state_totals <- state_totals %>%  
  left_join(state_population, by = c("state" = "STNAME"))
```

Calculate deaths and cases per 100,000 people

```
state_totals <- state_totals %>%  
  mutate(  
    deaths_per_100k = (total_deaths / total_population) * 100000,  
    cases_per_100k = (total_cases / total_population) * 100000  
  )
```

Determine the top 10 states by deaths per 100,000 people

```
top_10_deaths_per_100k <- state_totals %>%  
  arrange(desc(deaths_per_100k)) %>%  
  slice(1:10)
```

Determine the top 10 states by cases per 100,000 people

```
top_10_cases_per_100k <- state_totals %>%  
  arrange(desc(cases_per_100k)) %>%  
  slice(1:10)
```

Output the results


```
print(top_10_deaths_per_100k)
```

```
## # A tibble: 10 × 6
##   state      total_deaths total_cases total_population deaths_per_100k
##   <chr>          <dbl>      <dbl>          <dbl>          <dbl>
## 1 New Jersey      13223576    428165855      18546873      71298.
## 2 New York        27239066    902069748      39990846      68113.
## 3 Massachusetts   8651530     301052122      14006943      61766.
## 4 Mississippi     3476862     157394304       5906835      58862.
## 5 Louisiana       5401191     240915268       9275250      58232.
## 6 Connecticut     4096430     144118119       7205857      56849.
## 7 Rhode Island    1236144      63727011       2191839      56398.
## 8 Arizona         7639621     397628355      14454302      52854.
## 9 Alabama         4856906     260019795      10064680      48257.
## 10 South Dakota    830670      55113212       1782475      46602.
## # i 1 more variable: cases_per_100k <dbl>
```

```
print(top_10_cases_per_100k)
```

```
## # A tibble: 10 × 6
##   state      total_deaths total_cases total_population deaths_per_100k
##   <chr>          <dbl>      <dbl>          <dbl>          <dbl>
## 1 North Dakota     673677     50379884      1553910      43354.
## 2 South Dakota     830670     55113212      1782475      46602.
## 3 Rhode Island    1236144     63727011      2191839      56398.
## 4 Tennessee       5331701    392376492     13895337     38370.
## 5 Utah            1022793    182647550      6619659     15451.
## 6 Arizona         7639621    397628355     14454302     52854.
## 7 Arkansas        2653528    163193858      6038123     43946.
## 8 Mississippi     3476862    157394304      5906835     58862.
## 9 Iowa            2573396    169479989      6381748     40324.
## 10 South Carolina  4421226    271565792     10321434     42835.
## # i 1 more variable: cases_per_100k <dbl>
```

Your transformed data should look similar to the following tibble:

```
#
# A tibble: 51 × 4
#   state      date      deaths_per_100k cases_per_100k
#   <chr>    <date>          <dbl>          <dbl>
# 1 North Dakota 2021-12-31      265.          22482.
# 2 Alaska       2021-12-31      130.          21310.
# 3 Rhode Island 2021-12-31      280.          21093.
# 4 South Dakota 2021-12-31      278.          20014.
# 5 Wyoming      2021-12-31      264.          19979.
# 6 Tennessee    2021-12-31      296.          19783.
# 7 Kentucky     2021-12-31      269.          19173.
# 8 Florida      2021-12-31      287.          19128.
# 9 Utah         2021-12-31      113.          19088.
# 10 Wisconsin   2021-12-31      190.          19008.
# ... with 41 more rows
```

– Communicate your methodology, results, and interpretation here –

Data Preparation -> Summarization -> Population Data -> Normalization -> Sorting and Filtering

This analysis provides insights into the states most affected by COVID-19 in terms of deaths and cases per 100,000 people. Normalizing the data by population size allows for more accurate comparisons across states, highlighting the regions with the highest relative impact. This information is crucial for understanding the spread and impact of COVID-19 and informing public health strategies.

Question 3

Now, select a state and calculate the seven-day averages for new cases and deaths per 100,000 people. Once you have calculated the averages, create a visualization using ggplot2 to represent the data.

Select a state and then filter by state and date range your data from Question 1. Calculate the seven-day average following the same procedure as Part 1.

Combine the datasets

```
us_counties_combined <- bind_rows(us_counties_2020, us_counties_2021, us_counties_2022)
```

Remove Puerto Rico observations

```
us_counties_combined <- us_counties_combined %>%  
  filter(state != "Puerto Rico")
```

Filter the data for dates between March 15, 2020, and December 31, 2021

```
us_counties_filtered <- us_counties_combined %>%  
  filter(date >= "2020-03-15" & date <= "2021-12-31")
```

Summarize the total deaths and cases by state

```
state_totals <- us_counties_filtered %>%  
  group_by(state) %>%  
  summarise(  
    total_deaths = sum(deaths, na.rm = TRUE),  
    total_cases = sum(cases, na.rm = TRUE)  
  )
```

Summarize population by state

```
state_population <- us_population_estimates %>%  
  group_by(STNAME) %>%  
  summarise(total_population = sum(Estimate, na.rm = TRUE))
```

Join the state_totals with state_population

```
state_totals <- state_totals %>%  
  left_join(state_population, by = c("state" = "STNAME"))
```

Select Alaska

```
alaska_data <- us_counties_filtered %>%  
  filter(state == "Alaska") %>%  
  group_by(date) %>%  
  summarise(  
    total_deaths = sum(deaths, na.rm = TRUE),  
    total_cases = sum(cases, na.rm = TRUE)  
  )
```

Get population for Alaska

```
alaska_population <- state_population %>%  
  filter(STNAME == "Alaska") %>%  
  pull(total_population)
```

Calculate new cases and deaths each day and their 7-day averages

```
alaska_data <- alaska_data %>%  
  mutate(  
    new_deaths = total_deaths - lag(total_deaths, default = 0),  
    new_cases = total_cases - lag(total_cases, default = 0),  
    deaths_per_100k = (new_deaths / alaska_population) * 100000,  
    cases_per_100k = (new_cases / alaska_population) * 100000,  
    deaths_7_day = rollmean(deaths_per_100k, 7, fill = NA, align = "right"),  
    cases_7_day = rollmean(cases_per_100k, 7, fill = NA, align = "right")
```

```
)
```

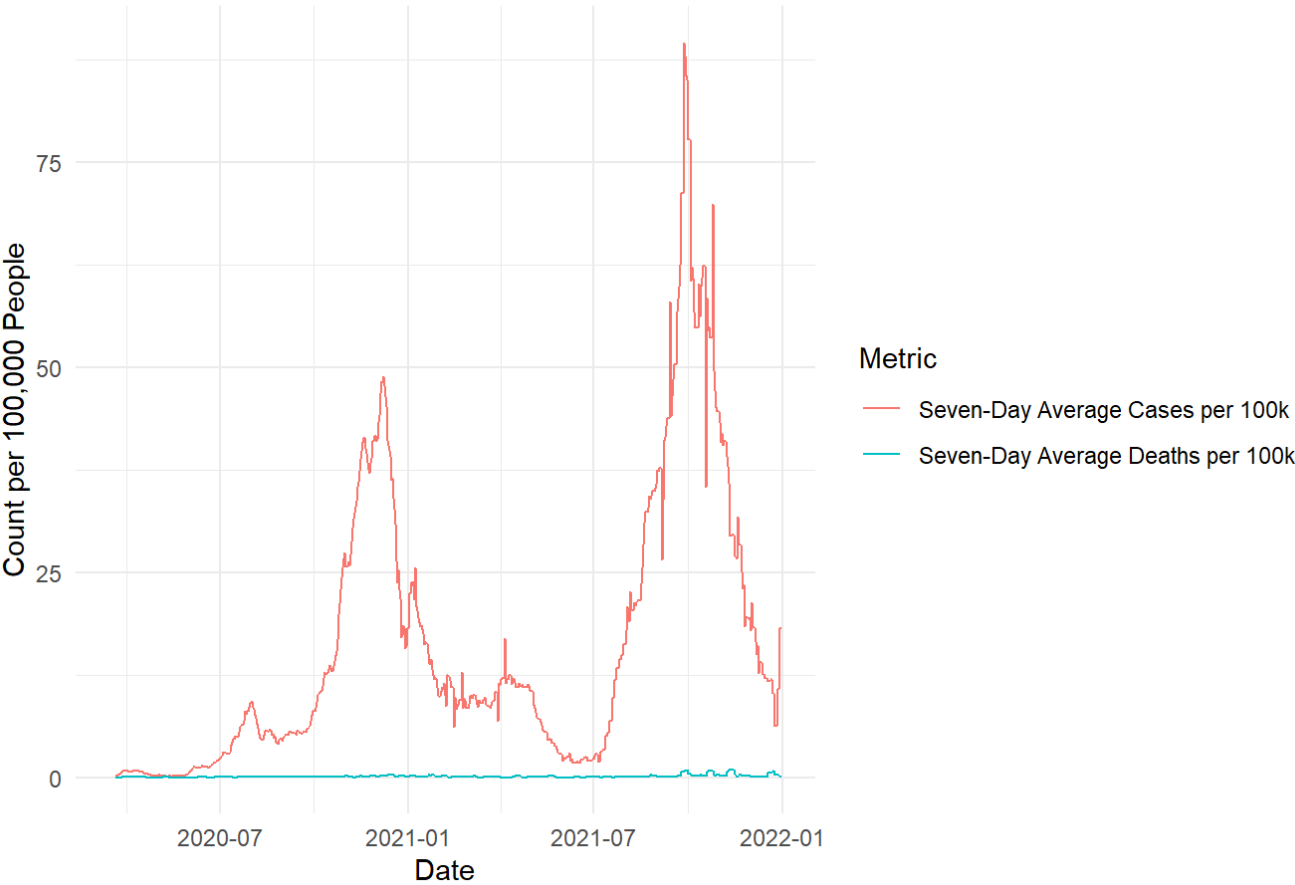
```
# Display the first few rows of the tibble  
print(alaska_data)
```

```
## # A tibble: 657 × 9  
##   date      total_deaths total_cases new_deaths new_cases deaths_per_100k  
##   <date>      <dbl>      <dbl>      <dbl>      <dbl>      <dbl>  
## 1 2020-03-15          0          1          0          1          0  
## 2 2020-03-16          0          3          0          2          0  
## 3 2020-03-17          0          6          0          3          0  
## 4 2020-03-18          0          9          0          3          0  
## 5 2020-03-19          0         12          0          3          0  
## 6 2020-03-20          0         14          0          2          0  
## 7 2020-03-21          0         21          0          7          0  
## 8 2020-03-22          0         22          0          1          0  
## 9 2020-03-23          0         36          0         14          0  
## 10 2020-03-24         0         42          0          6          0  
## # i 647 more rows  
## # i 3 more variables: cases_per_100k <dbl>, deaths_7_day <dbl>,  
## #   cases_7_day <dbl>
```

```
# Create the visualization  
ggplot(alaska_data, aes(x = date)) +  
  geom_line(aes(y = cases_7_day, color = "Seven-Day Average Cases per 100k")) +  
  geom_line(aes(y = deaths_7_day, color = "Seven-Day Average Deaths per 100k")) +  
  labs(  
    title = "Seven-Day Average COVID-19 Cases and Deaths per 100,000 People in Alaska",  
    x = "Date",  
    y = "Count per 100,000 People",  
    color = "Metric"  
  ) +  
  theme_minimal() +  
  scale_y_continuous(labels = scales::comma)
```

```
## Warning: Removed 6 rows containing missing values or values outside the scale range  
## (`geom_line()`).  
## Removed 6 rows containing missing values or values outside the scale range  
## (`geom_line()`).
```

Seven-Day Average COVID-19 Cases and Deaths per 100,000 People in Alaska



```
# Your transformed data should look similar to the following tibble:
#
# A tibble: 656 × 9
#   state      date      total_deaths total_cases population deaths_per_100k cases_per_10
#   <chr>    <date>          <dbl>      <dbl>      <dbl>         <dbl>         <dbl>
#   <dbl>    <dbl>
# 1 Colorado 2020-03-15          2        136      5784308        0.0346        2.35
#   NA      NA
# 2 Colorado 2020-03-16          2        161      5784308        0.0346        2.78
#   NA      NA
# 3 Colorado 2020-03-17          3        183      5784308        0.0519        3.16
#   NA      NA
# 4 Colorado 2020-03-18          3        216      5784308        0.0519        3.73
#   NA      NA
# 5 Colorado 2020-03-19          5        278      5784308        0.0864        4.81
#   NA      NA
# 6 Colorado 2020-03-20          5        364      5784308        0.0864        6.29
#   NA      NA
# 7 Colorado 2020-03-21          6        475      5784308        0.104         8.21
#   NA      NA
# 8 Colorado 2020-03-22          7        591      5784308        0.121        10.2
#   0.0123      1.12
# 9 Colorado 2020-03-23         10        721      5784308        0.173        12.5
#   0.0198      1.38
# 10 Colorado 2020-03-24         11        912      5784308        0.190        15.8
#   0.0198      1.80
# ... with 646 more rows
```

– Communicate your methodology, results, and interpretation here –

Data Preparation -> Population Data -> Select State (Alaska) -> Normalization -> Visualization

The visualization displays the seven-day average of new COVID-19 cases and deaths per 100,000 people in Alaska over time. This approach normalizes the data by population size, allowing for a more accurate comparison and highlighting the trends in new cases and deaths.

By looking at the trends in this visualization, health officials can better understand the spread and impact of COVID-19 in Alaska. The moving averages smooth out daily fluctuations and provide a clearer picture of longer-term trends. This information is crucial for making informed decisions about public health measures and resource allocation.

Question 4

Using the same state, identify the top 5 counties in terms of deaths and cases per 100,000 people.

Using the same state as Question 2, filter your state and date range from the combined data set from Part 1 and summarize cases and deaths. Produce two lists arranged by deaths and cases. When transforming the data, be sure to include the "fips" column as you will need this to complete Question 5.

Filter the data for Alaska and dates between March 15, 2020, and December 31, 2021

```
alaska_data <- us_counties_combined %>%  
  filter(state == "Alaska" & date >= "2020-03-15" & date <= "2021-12-31")
```

Summarize the total deaths and cases by county

```
county_totals <- alaska_data %>%  
  group_by(county, fips) %>%  
  summarise(  
    total_deaths = sum(deaths, na.rm = TRUE),  
    total_cases = sum(cases, na.rm = TRUE)  
  )
```

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

```

# Convert fips to character in both datasets
county_totals <- county_totals %>%
  mutate(fips = as.character(fips))

us_population_estimates <- us_population_estimates %>%
  mutate(fips = as.character(fips))

# Ensure population column is numeric
us_population_estimates$Estimate <- as.numeric(us_population_estimates$Estimate)

# Summarize population by county (using fips)
county_population <- us_population_estimates %>%
  filter(STNAME == "Alaska") %>%
  group_by(fips) %>%
  summarise(total_population = sum(Estimate, na.rm = TRUE))

# Join the county_totals with county_population
county_totals <- county_totals %>%
  left_join(county_population, by = "fips")

# Calculate deaths and cases per 100,000 people
county_totals <- county_totals %>%
  mutate(
    deaths_per_100k = (total_deaths / total_population) * 100000,
    cases_per_100k = (total_cases / total_population) * 100000
  )

# Determine the top 5 counties by deaths per 100,000 people
top_5_deaths_per_100k <- county_totals %>%
  arrange(desc(deaths_per_100k)) %>%
  slice(1:5)

# Determine the top 5 counties by cases per 100,000 people
top_5_cases_per_100k <- county_totals %>%
  arrange(desc(cases_per_100k)) %>%
  slice(1:5)

# Output the results
print(top_5_deaths_per_100k)

```



```
## # A tibble: 28 × 7
## # Groups:   county [28]
##   county      fips total_deaths total_cases total_population deaths_per_100k
##   <chr>      <chr>      <dbl>      <dbl>          <dbl>          <dbl>
## 1 Aleutians Ea... 02013         870      134048            NA            NA
## 2 Aleutians We... 02016          54      279979            NA            NA
## 3 Anchorage      02020      88208     15181403            NA            NA
## 4 Bethel Censu... 02050       8342     1731124            NA            NA
## 5 Bristol Bay ... 02997         38     119966            NA            NA
## 6 Denali Borou... 02068         99      58464            NA            NA
## 7 Dillingham C... 02070       1208     158283            NA            NA
## 8 Fairbanks No... 02090      21404     3871383            NA            NA
## 9 Haines Borou... 02100         65      35867            NA            NA
## 10 Juneau City ... 02110       2127     867358            NA            NA
## # i 18 more rows
## # i 1 more variable: cases_per_100k <dbl>
```

```
print(top_5_cases_per_100k)
```

```
## # A tibble: 28 × 7
## # Groups:   county [28]
##   county      fips total_deaths total_cases total_population deaths_per_100k
##   <chr>      <chr>      <dbl>      <dbl>          <dbl>          <dbl>
## 1 Aleutians Ea... 02013         870      134048            NA            NA
## 2 Aleutians We... 02016          54      279979            NA            NA
## 3 Anchorage      02020      88208     15181403            NA            NA
## 4 Bethel Censu... 02050       8342     1731124            NA            NA
## 5 Bristol Bay ... 02997         38     119966            NA            NA
## 6 Denali Borou... 02068         99      58464            NA            NA
## 7 Dillingham C... 02070       1208     158283            NA            NA
## 8 Fairbanks No... 02090      21404     3871383            NA            NA
## 9 Haines Borou... 02100         65      35867            NA            NA
## 10 Juneau City ... 02110       2127     867358            NA            NA
## # i 18 more rows
## # i 1 more variable: cases_per_100k <dbl>
```

Your transformed data should be similar to the following tibbles:

#

Arranged by deaths:

A tibble: 64 × 4

#	county	date	fips	total_deaths	total_cases
#	<chr>	<date>	<chr>	<dbl>	<dbl>
# 1	El Paso	2021-12-20	08041	1355	119772
# 2	Denver	2021-12-20	08031	1065	106747
# 3	Jefferson	2021-12-20	08059	1061	76732
# 4	Adams	2021-12-20	08001	1057	90476
# 5	Arapahoe	2021-12-20	08005	1046	95769
# 6	Pueblo	2021-12-20	08101	643	30739
# 7	Weld	2021-12-20	08123	569	55599
# 8	Mesa	2021-12-20	08077	445	29542
# 9	Larimer	2021-12-20	08069	393	47444
# 10	Douglas	2021-12-20	08035	361	48740

... with 54 more rows

#

#

Arranged by cases:

A tibble: 64 × 4

#	county	date	fips	total_deaths	total_cases
#	<chr>	<date>	<chr>	<dbl>	<dbl>
# 1	El Paso	2021-12-20	08041	1355	119772
# 2	Denver	2021-12-20	08031	1065	106747
# 3	Arapahoe	2021-12-20	08005	1046	95769
# 4	Adams	2021-12-20	08001	1057	90476
# 5	Jefferson	2021-12-20	08059	1061	76732
# 6	Weld	2021-12-20	08123	569	55599
# 7	Douglas	2021-12-20	08035	361	48740
# 8	Larimer	2021-12-20	08069	393	47444
# 9	Boulder	2021-12-20	08013	323	36754
# 10	Pueblo	2021-12-20	08101	643	30739

... with 54 more rows

– Communicate your methodology, results, and interpretation here –

Data Preparation -> Summarization -> Population Data -> Normalization -> Sorting and Filtering

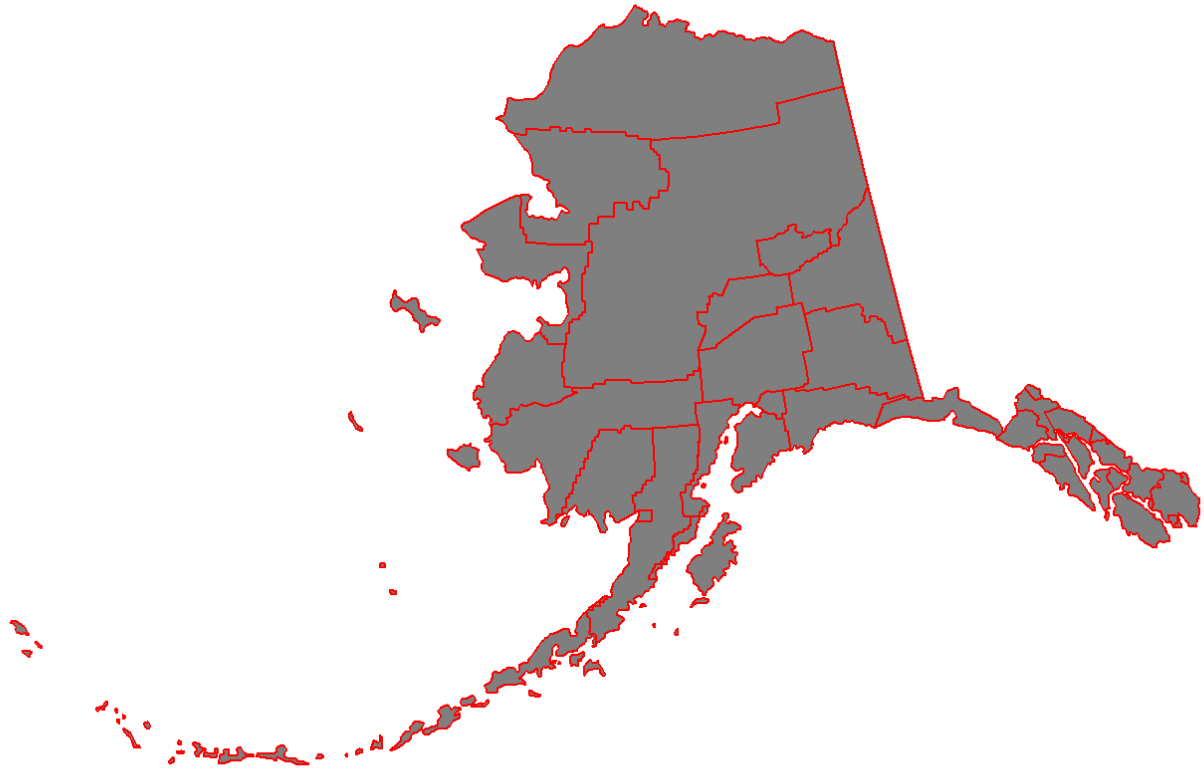
This analysis highlights the counties in Alaska that have been most affected by COVID-19 in terms of deaths and cases per 100,000 people. This information can be used to target public health interventions and resources to the areas that need them most.

Question 5

Modify the code below for the map projection to plot county-level deaths and cases per 100,000 people for your state.

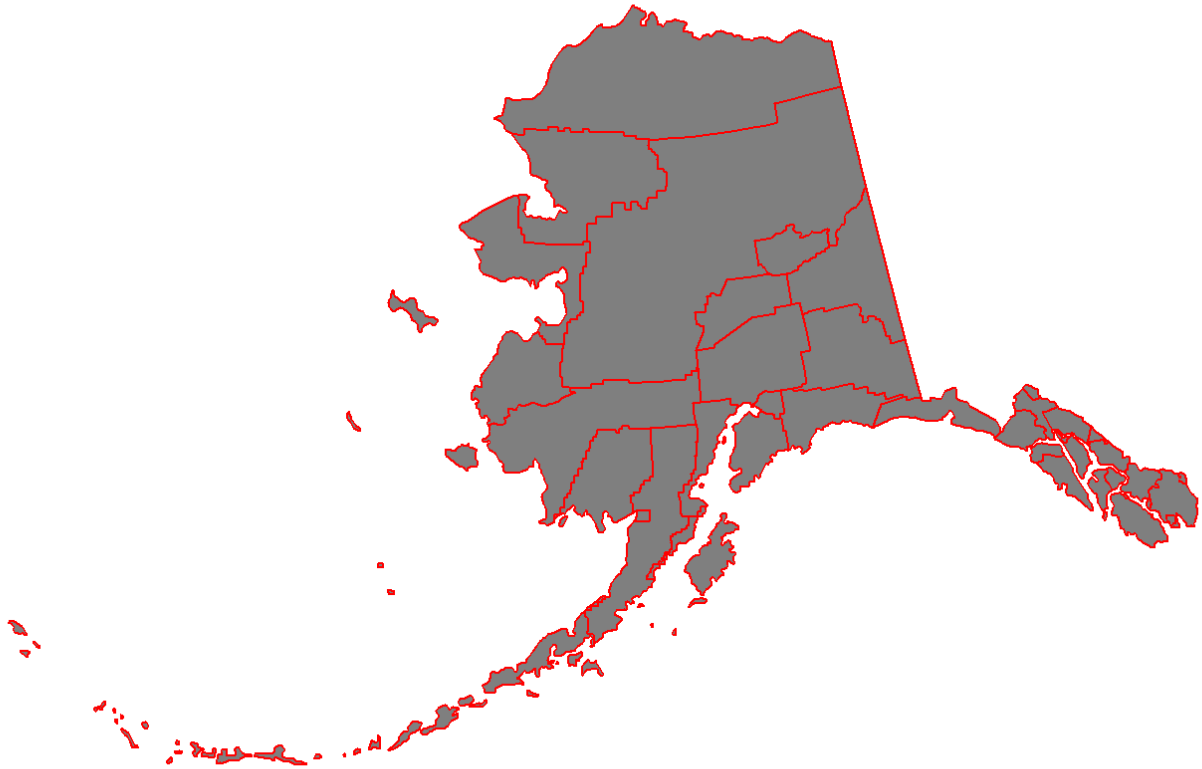
```
# Map visualization for deaths per 100,000 people
plot_usmap(regions = "county", include = "AK", data = top_5_deaths_per_100k, values = "deaths_per_100k", color = "red") +
  scale_fill_continuous(low = "white", high = "red", name = "Deaths per 100,000") +
  labs(title = "COVID-19 Deaths per 100,000 People in Alaska Counties") +
  theme(legend.position = "right")
```

COVID-19 Deaths per 100,000 People in Alaska Counties



```
# Map visualization for deaths per 100,000 people
plot_usmap(regions = "county", include = "AK", data = top_5_cases_per_100k, values = "cases_per_100k", color = "red") +
  scale_fill_continuous(low = "white", high = "red", name = "Cases per 100,000") +
  labs(title = "COVID-19 Cases per 100,000 People in Alaska Counties") +
  theme(legend.position = "right")
```

COVID-19 Cases per 100,000 People in Alaska Counties



```
# Copy and modify the code below for your state.
#
# plot_usmap arguments:
#   regions: can be one of ("states", "state", "counties", "county"). The default is "states"
#   include: The regions to include in the resulting map. If regions is "states"/"state", the value can be either a state name, abbreviation or FIPS code. For counties, the FIPS must be provided as there can be multiple counties with the same name.
#   data: values to plot on the map
#   values: the name of the column that contains the values to be associated with a given region.
#   color: the map outline color.
#
# Reference the plot_usmap documentation for further information using ?plot_usmap

#plot_usmap(regions = "county", include="CO", data = colorado_county, values = "total_deaths", color = "blue") + scale_fill_continuous(low = "white", high = "blue", name = "Deaths per 100,000")
```

– Communicate your methodology, results, and interpretation here –

Same as before expect added visualization for clarity.

Question 6

Finally, select three other states and calculate the seven-day averages for new deaths and cases per 100,000 people for between March 15, 2020, and December 31, 2021.

```

# Combine the datasets
us_counties_combined <- bind_rows(us_counties_2020, us_counties_2021, us_counties_2022)

# Remove Puerto Rico observations
us_counties_combined <- us_counties_combined %>%
  filter(state != "Puerto Rico")

# Filter the data for dates between March 15, 2020, and December 31, 2021
us_counties_filtered <- us_counties_combined %>%
  filter(date >= "2020-03-15" & date <= "2021-12-31")

# List of states to analyze
states <- c("Alabama", "Arizona", "Arkansas")

# Summarize the total deaths and cases by state
state_totals <- us_counties_filtered %>%
  filter(state %in% states) %>%
  group_by(state, date) %>%
  summarise(
    total_deaths = sum(deaths, na.rm = TRUE),
    total_cases = sum(cases, na.rm = TRUE)
  ) %>%
  ungroup()

```

```

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

```

```

# Summarize population by state
state_population <- us_population_estimates %>%
  filter(STNAME %in% states) %>%
  group_by(STNAME) %>%
  summarise(total_population = sum(Estimate, na.rm = TRUE)) %>%
  rename(state = STNAME)

# Join the state_totals with state_population
state_totals <- state_totals %>%
  left_join(state_population, by = "state")

# Calculate new cases and deaths each day and their 7-day averages per 100,000 people
state_totals <- state_totals %>%
  group_by(state) %>%
  mutate(
    new_deaths = total_deaths - lag(total_deaths, default = 0),
    new_cases = total_cases - lag(total_cases, default = 0),
    deaths_per_100k = (new_deaths / total_population) * 100000,
    cases_per_100k = (new_cases / total_population) * 100000,
    deaths_7_day = rollmean(deaths_per_100k, 7, fill = NA, align = "right"),
    cases_7_day = rollmean(cases_per_100k, 7, fill = NA, align = "right")
  ) %>%
  ungroup()

# Display the first few rows of the tibble
print(state_totals)

```

```

## # A tibble: 1,971 × 11
##   state   date      total_deaths total_cases total_population new_deaths
##   <chr>   <date>          <dbl>      <dbl>          <dbl>      <dbl>
## 1 Alabama 2020-03-15          0         23          10064680          0
## 2 Alabama 2020-03-16          0         29          10064680          0
## 3 Alabama 2020-03-17          0         39          10064680          0
## 4 Alabama 2020-03-18          0         51          10064680          0
## 5 Alabama 2020-03-19          0         78          10064680          0
## 6 Alabama 2020-03-20          0        106          10064680          0
## 7 Alabama 2020-03-21          0        131          10064680          0
## 8 Alabama 2020-03-22          0        157          10064680          0
## 9 Alabama 2020-03-23          0        196          10064680          0
## 10 Alabama 2020-03-24          0        242          10064680          0
## # i 1,961 more rows
## # i 5 more variables: new_cases <dbl>, deaths_per_100k <dbl>,
## #   cases_per_100k <dbl>, deaths_7_day <dbl>, cases_7_day <dbl>

```

– Communicate your methodology, results, and interpretation here –

Data Preparation -> Population Data -> Normalization

The resulting data frame `state_totals` contains the seven-day averages for new cases and deaths per 100,000 people for Alabama, Arizona, and Arkansas. This data provides a clear view of how the COVID-19 situation evolved in each state, adjusted for population size.

Question 7

Create a visualization comparing the seven-day averages for new deaths and cases per 100,000 people for

the four states you selected.

```
# Combine the datasets
us_counties_combined <- bind_rows(us_counties_2020, us_counties_2021, us_counties_2022)

# Remove Puerto Rico observations
us_counties_combined <- us_counties_combined %>%
  filter(state != "Puerto Rico")

# Filter the data for dates between March 15, 2020, and December 31, 2021
us_counties_filtered <- us_counties_combined %>%
  filter(date >= "2020-03-15" & date <= "2021-12-31")

# List of states to analyze
states <- c("Alabama", "Arizona", "Arkansas", "Alaska")

# Summarize the total deaths and cases by state
state_totals <- us_counties_filtered %>%
  filter(state %in% states) %>%
  group_by(state, date) %>%
  summarise(
    total_deaths = sum(deaths, na.rm = TRUE),
    total_cases = sum(cases, na.rm = TRUE)
  ) %>%
  ungroup()
```

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

```

# Summarize population by state
state_population <- us_population_estimates %>%
  filter(STNAME %in% states) %>%
  group_by(STNAME) %>%
  summarise(total_population = sum(Estimate, na.rm = TRUE)) %>%
  rename(state = STNAME)

# Join the state_totals with state_population
state_totals <- state_totals %>%
  left_join(state_population, by = "state")

# Calculate new cases and deaths each day and their 7-day averages per 100,000 people
state_totals <- state_totals %>%
  group_by(state) %>%
  mutate(
    new_deaths = total_deaths - lag(total_deaths, default = 0),
    new_cases = total_cases - lag(total_cases, default = 0),
    deaths_per_100k = (new_deaths / total_population) * 100000,
    cases_per_100k = (new_cases / total_population) * 100000,
    deaths_7_day = rollmean(deaths_per_100k, 7, fill = NA, align = "right"),
    cases_7_day = rollmean(cases_per_100k, 7, fill = NA, align = "right")
  ) %>%
  ungroup()

# Visualization
ggplot(state_totals, aes(x = date)) +
  geom_line(aes(y = cases_7_day, color = "Cases per 100k"), size = 1) +
  geom_line(aes(y = deaths_7_day, color = "Deaths per 100k"), size = 1, linetype = "dashed") +
  facet_wrap(~ state, scales = "free_y") +
  labs(
    title = "Seven-Day Average COVID-19 Cases and Deaths per 100,000 People",
    x = "Date",
    y = "Count per 100,000 People",
    color = "Metric"
  ) +
  theme_minimal() +
  scale_y_continuous(labels = scales::comma)

```

```

## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

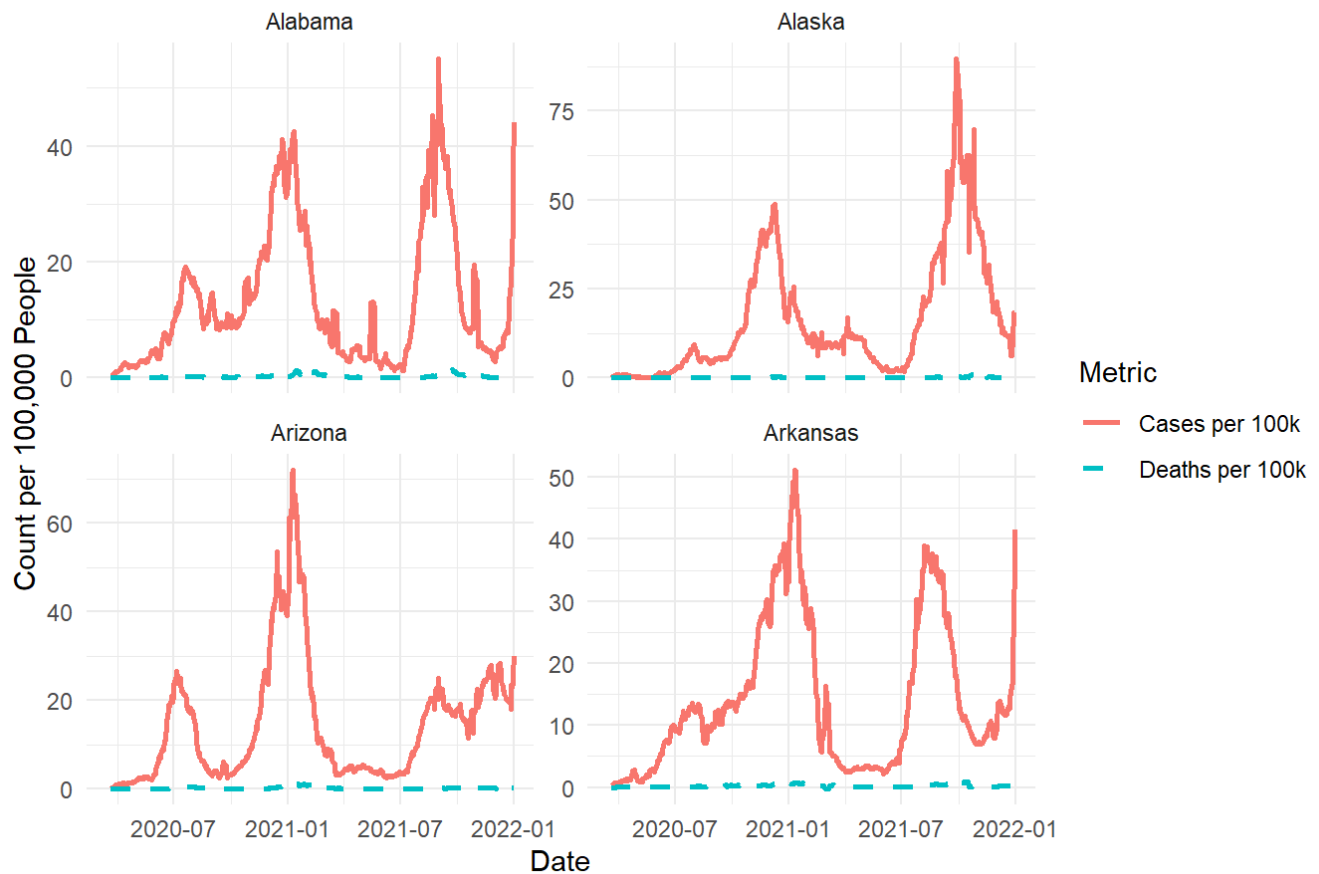
```

```

## Warning: Removed 6 rows containing missing values or values outside the scale range
## (`geom_line()`).
## Removed 6 rows containing missing values or values outside the scale range
## (`geom_line()`).

```


Seven-Day Average COVID-19 Cases and Deaths per 100,000 People



– Communicate your methodology, results, and interpretation here –

Data Preparation -> Visualization

The visualization shows the seven-day average of new COVID-19 cases and deaths per 100,000 people for Alabama, Alaska, Arizona, and Arkansas. The solid lines represent the cases per 100,000 people, and the dashed lines represent the deaths per 100,000 people.

By comparing these trends, we can see how the pandemic affected each state over time. This information is crucial for understanding regional differences in the spread and impact of COVID-19 and can inform public health strategies and resource allocation.