COVID-Data

Anonymous

2024-02-28

```r
# First thing is to library in the tidyverse packages as standard fare.
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

Intro and Data

For the demonstration, we are looking at COVID data from Johns Hopkins. So we want to import data. This is all from the same Github, so the initial part of the URL will be the same.

```r
url_base <- "https://github.com/CSSEGISandData/COVID-19/raw/master/csse_covid_19_data/csse_covid_19_time

filenames <- c(
  "time_series_covid19_confirmed_US.csv",
  "time_series_covid19_confirmed_global.csv",
  "time_series_covid19_deaths_US.csv",
  "time_series_covid19_deaths_global.csv"
  # "time_series_covid19_recovered_global.csv"
  )

# Concatenate in the filenames to the base url
urls <- str_c(url_base, filenames)

# Now read the data into variables
cases_US <- read_csv(urls[1])
```

```
## Rows: 3342 Columns: 1154
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr    (6): iso2, iso3, Admin2, Province_State, Country_Region, Combined_Key
## dbl (1148): UID, code3, FIPS, Lat, Long_, 1/22/20, 1/23/20, 1/24/20, 1/25/20...
```

```
##
## 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.
```

```
cases_global <- read_csv(urls[2])
```

```
## Rows: 289 Columns: 1147
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr    (2): Province/State, Country/Region
## dbl (1145): Lat, Long, 1/22/20, 1/23/20, 1/24/20, 1/25/20, 1/26/20, 1/27/20,...
##
## 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.
```

```
deaths_US <- read_csv(urls[3])
```

```
## Rows: 3342 Columns: 1155
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr    (6): iso2, iso3, Admin2, Province_State, Country_Region, Combined_Key
## dbl (1149): UID, code3, FIPS, Lat, Long_, Population, 1/22/20, 1/23/20, 1/24...
##
## 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.
```

```
deaths_global <- read_csv(urls[4])
```

```
## Rows: 289 Columns: 1147
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr    (2): Province/State, Country/Region
## dbl (1145): Lat, Long, 1/22/20, 1/23/20, 1/24/20, 1/25/20, 1/26/20, 1/27/20,...
##
## 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.
```

```
# Look at the data briefly. What is in it? What can be ignored or is not useful?
cases_global
```

```
## # A tibble: 289 x 1,147
##    `Province/State` `Country/Region`   Lat   Long `1/22/20` `1/23/20` `1/24/20`
##    <chr>            <chr>            <dbl>  <dbl>     <dbl>     <dbl>     <dbl>
##  1 <NA>             Afghanistan       33.9  67.7          0         0         0
##  2 <NA>             Albania           41.2  20.2          0         0         0
##  3 <NA>             Algeria           28.0   1.66         0         0         0
##  4 <NA>             Andorra           42.5   1.52         0         0         0
##  5 <NA>             Angola           -11.2  17.9          0         0         0
##  6 <NA>             Antarctica       -71.9  23.3          0         0         0
##  7 <NA>             Antigua and Bar~  17.1 -61.8          0         0         0
##  8 <NA>             Argentina        -38.4 -63.6          0         0         0
##  9 <NA>              Armenia           40.1  45.0          0          0          0
## 10 Australian Capit~ Australia        -35.5 149.           0          0          0
## # i 279 more rows
## # i 1,140 more variables: `1/25/20` <dbl>, `1/26/20` <dbl>, `1/27/20` <dbl>,
## #   `1/28/20` <dbl>, `1/29/20` <dbl>, `1/30/20` <dbl>, `1/31/20` <dbl>,
## #   `2/1/20` <dbl>, `2/2/20` <dbl>, `2/3/20` <dbl>, `2/4/20` <dbl>,
## #   `2/5/20` <dbl>, `2/6/20` <dbl>, `2/7/20` <dbl>, `2/8/20` <dbl>,
## #   `2/9/20` <dbl>, `2/10/20` <dbl>, `2/11/20` <dbl>, `2/12/20` <dbl>,
## #   `2/13/20` <dbl>, `2/14/20` <dbl>, `2/15/20` <dbl>, `2/16/20` <dbl>, ...
```

Tidying and combining tables

Global data

```
# Latitude and longitude are not necessary for our analysis, so that's one aspect we can tidy up.
# We may also want to tidy up the country/region and province/state.
# We can also tidy up in the sense of one observation per row - namely the dates reported.
# Recall PIPES

cases_global <- cases_global %>%
  # Turn cols into rows - everything "except" province/state, country/region, lat, long
  pivot_longer(cols = -c('Province/State', 'Country/Region', 'Lat', 'Long'),
               names_to = "Date",
               values_to = "Cases"
               ) %>%
  # And now drop the lat/long columns entirely
  select(-c("Lat", "Long"))

# Similar for the global deaths
deaths_global <- deaths_global %>%
  pivot_longer(cols = -c("Province/State", "Country/Region", "Lat", "Long"),
               names_to = "Date",
               values_to = "Deaths"
               ) %>%
  select(-c("Lat", "Long"))
```

```
# Looks like the data has been super updated since the course one, 3x as long lol
cases_global
```

```
## # A tibble: 330,327 x 4
##    `Province/State` `Country/Region` Date    Cases
##    <chr>            <chr>            <chr>   <dbl>
##  1 <NA>             Afghanistan      1/22/20     0
##  2 <NA>             Afghanistan      1/23/20     0
##  3 <NA>             Afghanistan      1/24/20     0
##  4 <NA>             Afghanistan      1/25/20     0
##  5 <NA>             Afghanistan      1/26/20     0
##  6 <NA>             Afghanistan      1/27/20     0
##  7 <NA>             Afghanistan      1/28/20     0
##  8 <NA>             Afghanistan      1/29/20     0
##  9 <NA>             Afghanistan      1/30/20     0
## 10 <NA>             Afghanistan      1/31/20     0
## # i 330,317 more rows
```

```r
deaths_global
```

```
## # A tibble: 330,327 x 4
##    `Province/State` `Country/Region` Date    Deaths
##    <chr>            <chr>            <chr>    <dbl>
##  1 <NA>             Afghanistan      1/22/20      0
##  2 <NA>             Afghanistan      1/23/20      0
##  3 <NA>             Afghanistan      1/24/20      0
##  4 <NA>             Afghanistan      1/25/20      0
##  5 <NA>             Afghanistan      1/26/20      0
##  6 <NA>             Afghanistan      1/27/20      0
##  7 <NA>             Afghanistan      1/28/20      0
##  8 <NA>             Afghanistan      1/29/20      0
##  9 <NA>             Afghanistan      1/30/20      0
## 10 <NA>             Afghanistan      1/31/20      0
## # i 330,317 more rows
```

```r
# They look good, now to join them.
global <- cases_global %>%
  full_join(deaths_global) %>%
  # I guess for ease of use later - makes it easier when combining with US data?
  rename(Country_Region = `Country/Region`,
         Province_State = `Province/State`) %>%
  mutate(Date = mdy(Date))
```

```
## Joining with `by = join_by(`Province/State`, `Country/Region`, Date)`
```

```r
global
```

```
## # A tibble: 330,327 x 5
##    Province_State Country_Region Date       Cases Deaths
##    <chr>          <chr>          <date>     <dbl>  <dbl>
##  1 <NA>           Afghanistan    2020-01-22     0      0
##  2 <NA>           Afghanistan    2020-01-23     0      0
##  3 <NA>           Afghanistan    2020-01-24     0      0
##  4 <NA>           Afghanistan    2020-01-25     0      0
##  5 <NA>           Afghanistan    2020-01-26     0      0
##  6 <NA>           Afghanistan    2020-01-27     0      0
##  7 <NA>           Afghanistan    2020-01-28     0      0
##  8 <NA>           Afghanistan    2020-01-29     0      0
##  9 <NA>           Afghanistan    2020-01-30     0      0
## 10 <NA>           Afghanistan    2020-01-31     0      0
## # i 330,317 more rows
```

```r
# Don't care about rows with no cases.
global <- global %>%
  filter(Cases > 0)

# At this point we could also check the numbers in the high range and make sure
# they exist later in the data just as a brief manual verification.
# Just a sanity check. Looks fine in the console, so not worth showing in here.
```

US data

Now let's look at the US-specific data.

```
cases_US
```

```
## # A tibble: 3,342 x 1,154
##         UID iso2  iso3  code3  FIPS Admin2   Province_State Country_Region   Lat
##       <dbl> <chr> <chr> <dbl> <dbl> <chr>    <chr>          <chr>          <dbl>
##  1 84001001 US    USA     840  1001 Autauga  Alabama        US              32.5
##  2 84001003 US    USA     840  1003 Baldwin  Alabama        US              30.7
##  3 84001005 US    USA     840  1005 Barbour  Alabama        US              31.9
##  4 84001007 US    USA     840  1007 Bibb     Alabama        US              33.0
##  5 84001009 US    USA     840  1009 Blount   Alabama        US              34.0
##  6 84001011 US    USA     840  1011 Bullock  Alabama        US              32.1
##  7 84001013 US    USA     840  1013 Butler   Alabama        US              31.8
##  8 84001015 US    USA     840  1015 Calhoun  Alabama        US              33.8
##  9 84001017 US    USA     840  1017 Chambers Alabama        US              32.9
## 10 84001019 US    USA     840  1019 Cherokee Alabama        US              34.2
## # i 3,332 more rows
## # i 1,145 more variables: Long_ <dbl>, Combined_Key <chr>, `1/22/20` <dbl>,
## #   `1/23/20` <dbl>, `1/24/20` <dbl>, `1/25/20` <dbl>, `1/26/20` <dbl>,
## #   `1/27/20` <dbl>, `1/28/20` <dbl>, `1/29/20` <dbl>, `1/30/20` <dbl>,
## #   `1/31/20` <dbl>, `2/1/20` <dbl>, `2/2/20` <dbl>, `2/3/20` <dbl>,
## #   `2/4/20` <dbl>, `2/5/20` <dbl>, `2/6/20` <dbl>, `2/7/20` <dbl>,
## #   `2/8/20` <dbl>, `2/9/20` <dbl>, `2/10/20` <dbl>, `2/11/20` <dbl>, ...
```

```
deaths_US
```

```
## # A tibble: 3,342 x 1,155
##         UID iso2  iso3  code3  FIPS Admin2   Province_State Country_Region   Lat
##       <dbl> <chr> <chr> <dbl> <dbl> <chr>    <chr>          <chr>          <dbl>
##  1 84001001 US    USA     840  1001 Autauga  Alabama        US              32.5
##  2 84001003 US    USA     840  1003 Baldwin  Alabama        US              30.7
##  3 84001005 US    USA     840  1005 Barbour  Alabama        US              31.9
##  4 84001007 US    USA     840  1007 Bibb     Alabama        US              33.0
##  5 84001009 US    USA     840  1009 Blount   Alabama        US              34.0
##  6 84001011 US    USA     840  1011 Bullock  Alabama        US              32.1
##  7 84001013 US    USA     840  1013 Butler   Alabama        US              31.8
##  8 84001015 US    USA     840  1015 Calhoun  Alabama        US              33.8
##  9 84001017 US    USA     840  1017 Chambers Alabama        US              32.9
## 10 84001019 US    USA     840  1019 Cherokee Alabama        US              34.2
## # i 3,332 more rows
## # i 1,146 more variables: Long_ <dbl>, Combined_Key <chr>, Population <dbl>,
## #   `1/22/20` <dbl>, `1/23/20` <dbl>, `1/24/20` <dbl>, `1/25/20` <dbl>,
## #   `1/26/20` <dbl>, `1/27/20` <dbl>, `1/28/20` <dbl>, `1/29/20` <dbl>,
## #   `1/30/20` <dbl>, `1/31/20` <dbl>, `2/1/20` <dbl>, `2/2/20` <dbl>,
## #   `2/3/20` <dbl>, `2/4/20` <dbl>, `2/5/20` <dbl>, `2/6/20` <dbl>,
## #   `2/7/20` <dbl>, `2/8/20` <dbl>, `2/9/20` <dbl>, `2/10/20` <dbl>, ...
```

We have a lot more granular information here.
Similar to the global data, we'll want to keep a hold of the dates to pivot.
Admin2 also looks like it may be useful, but not much before it.

```
# Like we did with the global info, pivot the dates.
cases_US <- cases_US %>%
  pivot_longer(cols = -(UID:Combined_Key),
               names_to = "Date",
               values_to = "Cases") %>%
  # Make sure the dates are date objects
  mutate(Date = mdy(Date)) %>%
  # Select everything from Admin2 onward
  select(Admin2:Cases) %>%
  # but then drop the Lat/Lon
  select(-c(Lat, Long_))

# Be sure to check the formats of both sets - they may be different.
# Deaths has a population metric, for example.
# But we do want the death counterpart to match up
deaths_US <- deaths_US %>%
  pivot_longer(cols = -(UID:Population),
               names_to = "Date",
               values_to = "Deaths") %>%
  mutate(Date = mdy(Date)) %>%
  select(Admin2:Deaths) %>%
  select(-c(Lat, Long_))
```

```
US <- cases_US %>%
  full_join(deaths_US)
```

```
## Joining with ‘by = join_by(Admin2, Province_State, Country_Region,
## Combined_Key, Date)‘
```

```
US
```

```
## # A tibble: 3,819,906 x 8
##    Admin2 Province_State Country_Region Combined_Key Date       Cases Population
##    <chr>  <chr>          <chr>          <chr>        <date>     <dbl>      <dbl>
##  1 Autau~ Alabama        US             Autauga, Al~ 2020-01-22     0      55869
##  2 Autau~ Alabama        US             Autauga, Al~ 2020-01-23     0      55869
##  3 Autau~ Alabama        US             Autauga, Al~ 2020-01-24     0      55869
##  4 Autau~ Alabama        US             Autauga, Al~ 2020-01-25     0      55869
##  5 Autau~ Alabama        US             Autauga, Al~ 2020-01-26     0      55869
##  6 Autau~ Alabama        US             Autauga, Al~ 2020-01-27     0      55869
##  7 Autau~ Alabama        US             Autauga, Al~ 2020-01-28     0      55869
##  8 Autau~ Alabama        US             Autauga, Al~ 2020-01-29     0      55869
##  9 Autau~ Alabama        US             Autauga, Al~ 2020-01-30     0      55869
## 10 Autau~ Alabama        US             Autauga, Al~ 2020-01-31     0      55869
## # i 3,819,896 more rows
## # i 1 more variable: Deaths <dbl>
```

Matching the data

The population data is interesting, but we don't have that for the global data set, so we'll want to find that. We'll also want to mutate it a bit more to match up with the columns of the US data set.

```
# CSV for global population info. Same repository.
uid_lookup_url <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/

uid <- read_csv(uid_lookup_url) %>%
  select(-c(Lat, Long_, Combined_Key, code3, iso2, iso3, Admin2))
```

```
## Rows: 4321 Columns: 12
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (7): iso2, iso3, FIPS, Admin2, Province_State, Country_Region, Combined_Key
## dbl (5): UID, code3, Lat, Long_, Population
##
## 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.
```

```
uid
```

```
## # A tibble: 4,321 x 5
##      UID FIPS  Province_State Country_Region      Population
##    <dbl> <chr> <chr>          <chr>                    <dbl>
##  1     4 <NA>  <NA>           Afghanistan           38928341
##  2     8 <NA>  <NA>           Albania                2877800
##  3    10 <NA>  <NA>           Antarctica                  NA
##  4    12 <NA>  <NA>           Algeria               43851043
##  5    20 <NA>  <NA>           Andorra                  77265
##  6    24 <NA>  <NA>           Angola                32866268
##  7    28 <NA>  <NA>           Antigua and Barbuda      97928
##  8    32 <NA>  <NA>           Argentina             45195777
##  9    51 <NA>  <NA>           Armenia                2963234
## 10    40 <NA>  <NA>           Austria                9006400
## # i 4,311 more rows
```

```
global <- global %>%
  unite("Combined_Key",
        c(Province_State, Country_Region),
        sep=", ",
        na.rm = TRUE,
        remove = FALSE)
```

```
# Now joining the global and uid tables.
# Feels like the class method was a little off? Oh well, end result worked.
global <- global %>%
  left_join(uid, by = c("Province_State", "Country_Region")) %>%
  # Like here, could have just excluded these in the previous cell.
  select(-c(UID, FIPS)) %>%
  select("Province_State", "Country_Region", "Date", "Cases", "Deaths", "Population", "Combined_Key")

global
```

```
## # A tibble: 306,827 x 7
##    Province_State Country_Region Date       Cases Deaths Population Combined_Key
##    <chr>          <chr>          <date>     <dbl>  <dbl>      <dbl> <chr>
##  1 <NA>           Afghanistan    2020-02-24     5      0   38928341 Afghanistan
##  2 <NA>           Afghanistan    2020-02-25     5      0   38928341 Afghanistan
##  3 <NA>           Afghanistan    2020-02-26     5      0   38928341 Afghanistan
##  4 <NA>           Afghanistan    2020-02-27     5      0   38928341 Afghanistan
##  5 <NA>           Afghanistan    2020-02-28     5      0   38928341 Afghanistan
##  6 <NA>           Afghanistan    2020-02-29     5      0   38928341 Afghanistan
##  7 <NA>           Afghanistan    2020-03-01     5      0   38928341 Afghanistan
##  8 <NA>           Afghanistan    2020-03-02     5      0   38928341 Afghanistan
##  9 <NA>           Afghanistan    2020-03-03     5      0   38928341 Afghanistan
## 10 <NA>           Afghanistan    2020-03-04     5      0   38928341 Afghanistan
## # i 306,817 more rows
```

Visualize

At this point, it doesn't exactly match up - US data still has the Admin2 column, but the last time I put a file together in a way that made sense to me that deviated from the tutorial, it came back to haunt me. So, noting that I think it's an issue, let's continue without fixing it for now!

Check out the US by state to make use of group by and summarize.

```r
US_by_state <- US %>%
  group_by(Province_State, Country_Region, Date) %>%
  summarize(Cases = sum(Cases), Deaths = sum(Deaths),
            Population = sum(Population)) %>%
  mutate(Deaths_Per_Mil = Deaths * 1000000 / Population) %>%
  select(Province_State, Country_Region, Date, Cases, Deaths,
         Deaths_Per_Mil, Population) %>%
  ungroup()
```

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

```r
US_by_state
```

```
## # A tibble: 66,294 x 7
##    Province_State Country_Region Date       Cases Deaths Deaths_Per_Mil
##    <chr>          <chr>          <date>     <dbl>  <dbl>          <dbl>
##  1 Alabama        US             2020-01-22     0      0              0
##  2 Alabama        US             2020-01-23     0      0              0
##  3 Alabama        US             2020-01-24     0      0              0
##  4 Alabama        US             2020-01-25     0      0              0
##  5 Alabama        US             2020-01-26     0      0              0
##  6 Alabama        US             2020-01-27     0      0              0
##  7 Alabama        US             2020-01-28     0      0              0
##  8 Alabama        US             2020-01-29     0      0              0
##  9 Alabama        US             2020-01-30     0      0              0
## 10 Alabama        US             2020-01-31     0      0              0
## # i 66,284 more rows
## # i 1 more variable: Population <dbl>
```

Now check out the total for the US.

```r
US_totals <- US_by_state %>%
  group_by(Country_Region, Date) %>%
  summarize(Cases = sum(Cases), Deaths = sum(Deaths),
            Population = sum(Population)) %>%
  # Nice rounding, bro.
  mutate(Deaths_Per_Mil = Deaths * 1000000 / Population) %>%
  select(Country_Region, Date, Cases, Deaths, Deaths_Per_Mil,
         Population) %>%
  ungroup()
```

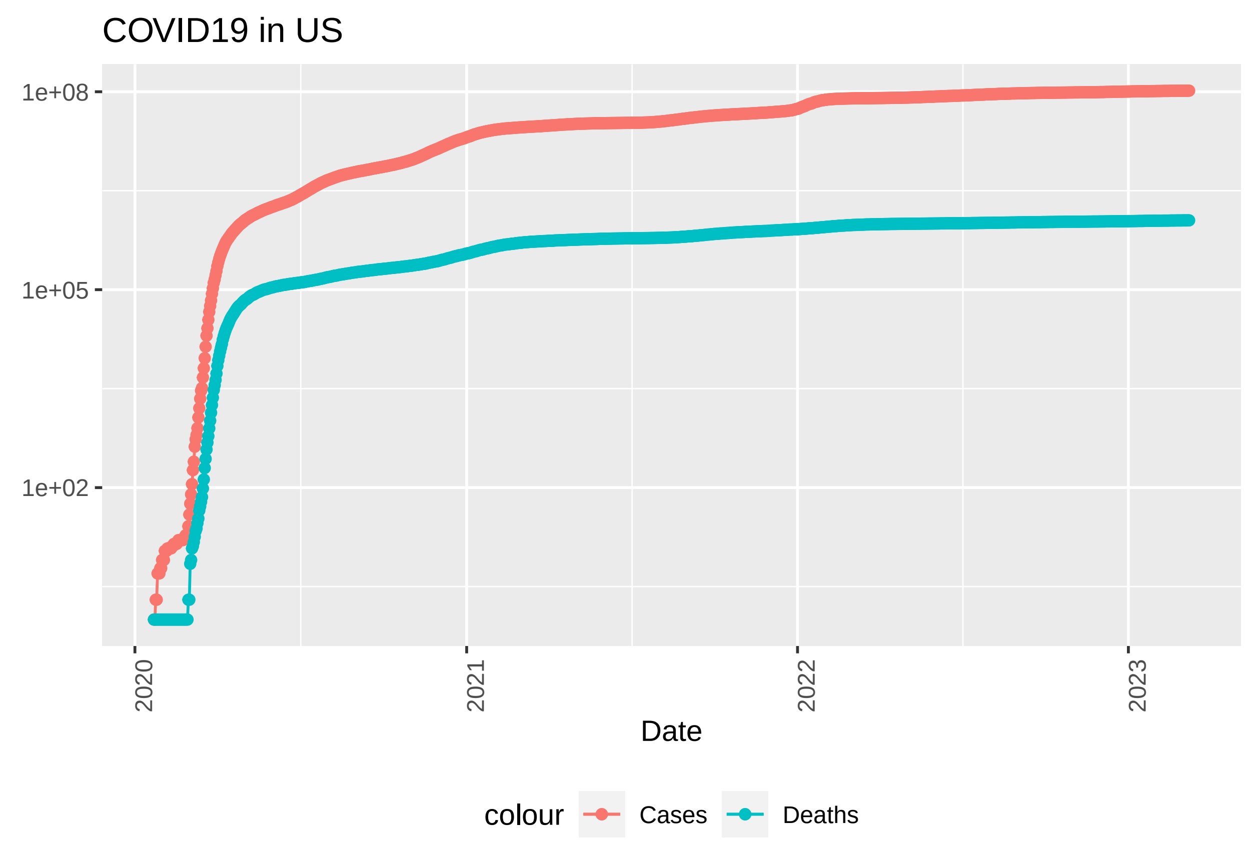
```
## `summarise()` has grouped output by 'Country_Region'. You can override using
## the `.groups` argument.
```

```r
tail(US_totals)
```

```
## # A tibble: 6 x 6
##   Country_Region Date           Cases  Deaths Deaths_Per_Mil Population
##   <chr>          <date>         <dbl>   <dbl>          <dbl>      <dbl>
## 1 US             2023-03-04 103650837 1122172          3371.  332875137
## 2 US             2023-03-05 103646975 1122134          3371.  332875137
## 3 US             2023-03-06 103655539 1122181          3371.  332875137
## 4 US             2023-03-07 103690910 1122516          3372.  332875137
## 5 US             2023-03-08 103755771 1123246          3374.  332875137
## 6 US             2023-03-09 103802702 1123836          3376.  332875137
```

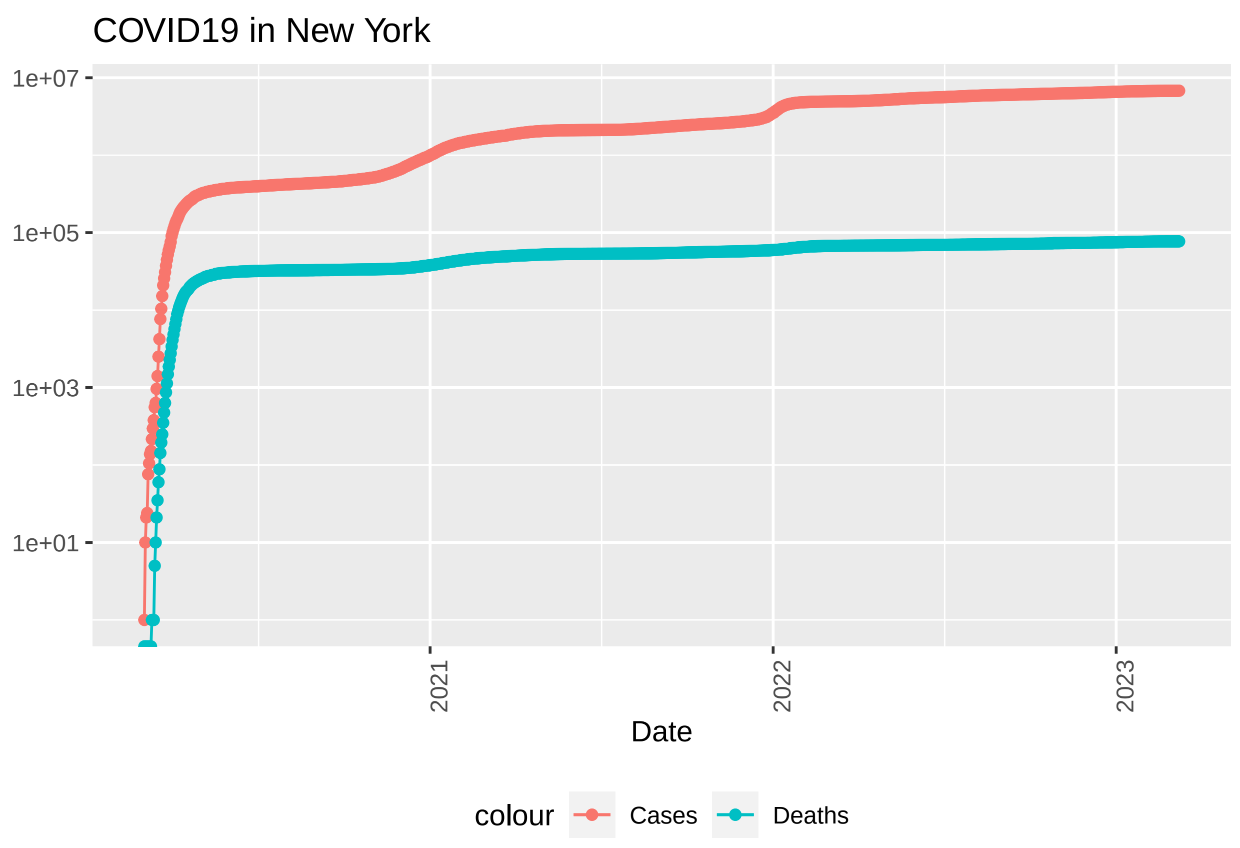
Now for the actual visualization part.
This is a simple one *from the demo*.

```r
# Overall
US_totals %>%
  # Based on revised numbers, this filter turns out to be useless
  filter(Cases > 0) %>%
  ggplot(aes(x = Date, y = Cases)) +
  geom_line(aes(color = "Cases")) +
  geom_point(aes(color = "Cases")) +
  geom_line(aes(y = Deaths, color = "Deaths")) +
  geom_point(aes(y = Deaths, color = "Deaths")) +
  # Given the huge scale diff, pretty mandatory
  scale_y_log10() +
  theme(legend.position = "bottom",
        axis.text.x = element_text(angle = 90)) +
  labs(title = "COVID19 in US", y = NULL)
```



```r
# By state
state <- "New York"
US_by_state %>%
  filter(Province_State == state) %>%
  # Not necessarily useless this time (but maybe)
  filter(Cases > 0) %>%
  ggplot(aes(x = Date, y = Cases)) +
  geom_line(aes(color = "Cases")) +
  geom_point(aes(color = "Cases")) +
  geom_line(aes(y = Deaths, color = "Deaths")) +
  geom_point(aes(y = Deaths, color = "Deaths")) +
  scale_y_log10() +
  theme(legend.position = "bottom",
        axis.text.x = element_text(angle = 90)) +
  labs(title = str_c("COVID19 in ", state), y = NULL)
```

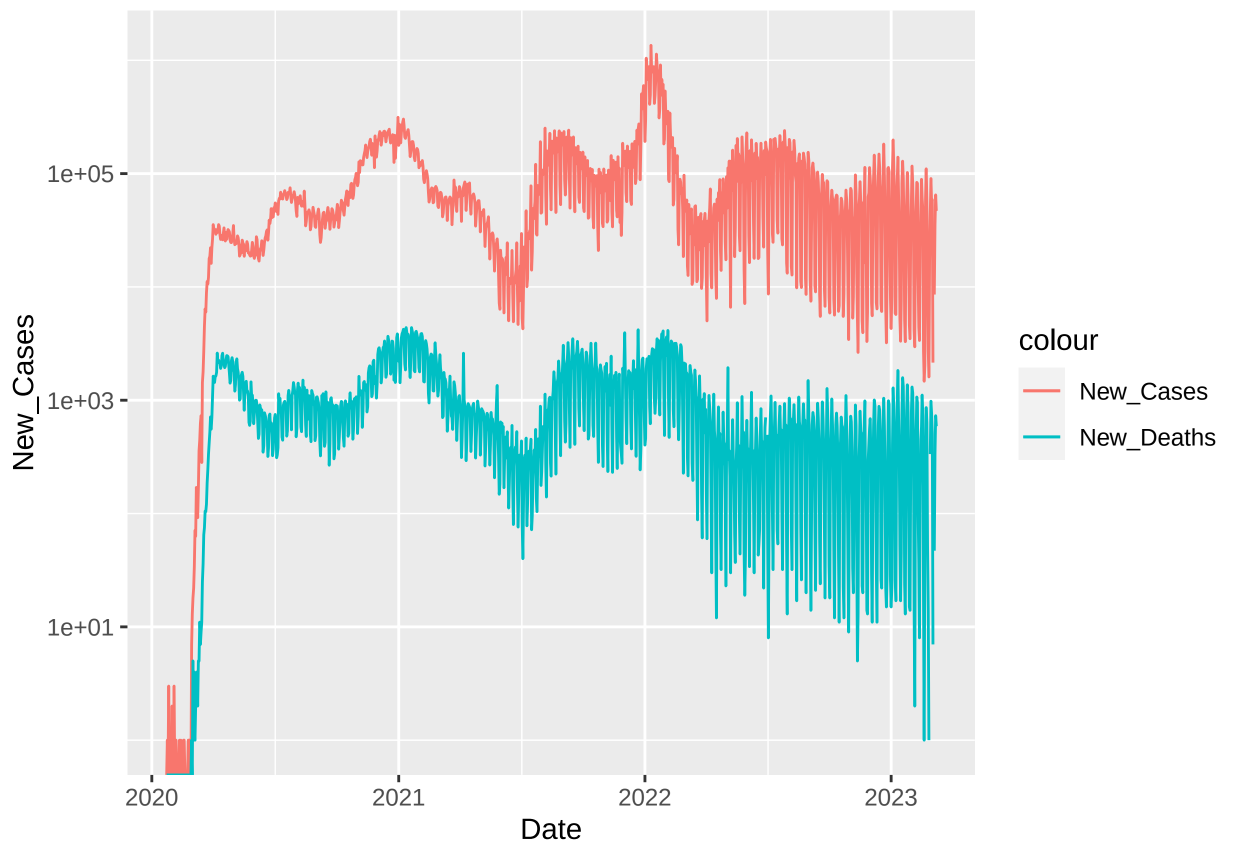
```
## Warning: Transformation introduced infinite values in continuous y-axis
## Transformation introduced infinite values in continuous y-axis
```



Questions can arise such as how many deaths total, have the deaths been leveling out, and so on. To answer these, add different data to make it look better!

```
US_by_state <- US_by_state %>%
  mutate(New_Cases = Cases - lag(Cases),
         New_Deaths = Deaths - lag(Deaths))

US_totals <- US_totals %>%
    mutate(New_Cases = Cases - lag(Cases),
           New_Deaths = Deaths - lag(Deaths))

US_totals %>%
  ggplot(aes(x = Date, y = New_Cases)) +
  geom_line(aes(color = "New_Cases")) +
  geom_line(aes(y = New_Deaths, color = "New_Deaths")) +
  scale_y_log10()
```

```
## Warning in self$trans$transform(x): NaNs produced

## Warning: Transformation introduced infinite values in continuous y-axis

## Warning in self$trans$transform(x): NaNs produced

## Warning: Transformation introduced infinite values in continuous y-axis
## Warning: Removed 1 row containing missing values ('geom_line()').
## Removed 1 row containing missing values ('geom_line()').
```



Zig-zag. Looks like the average is slowing down, but not petering out like the earlier visualization seemed to imply.

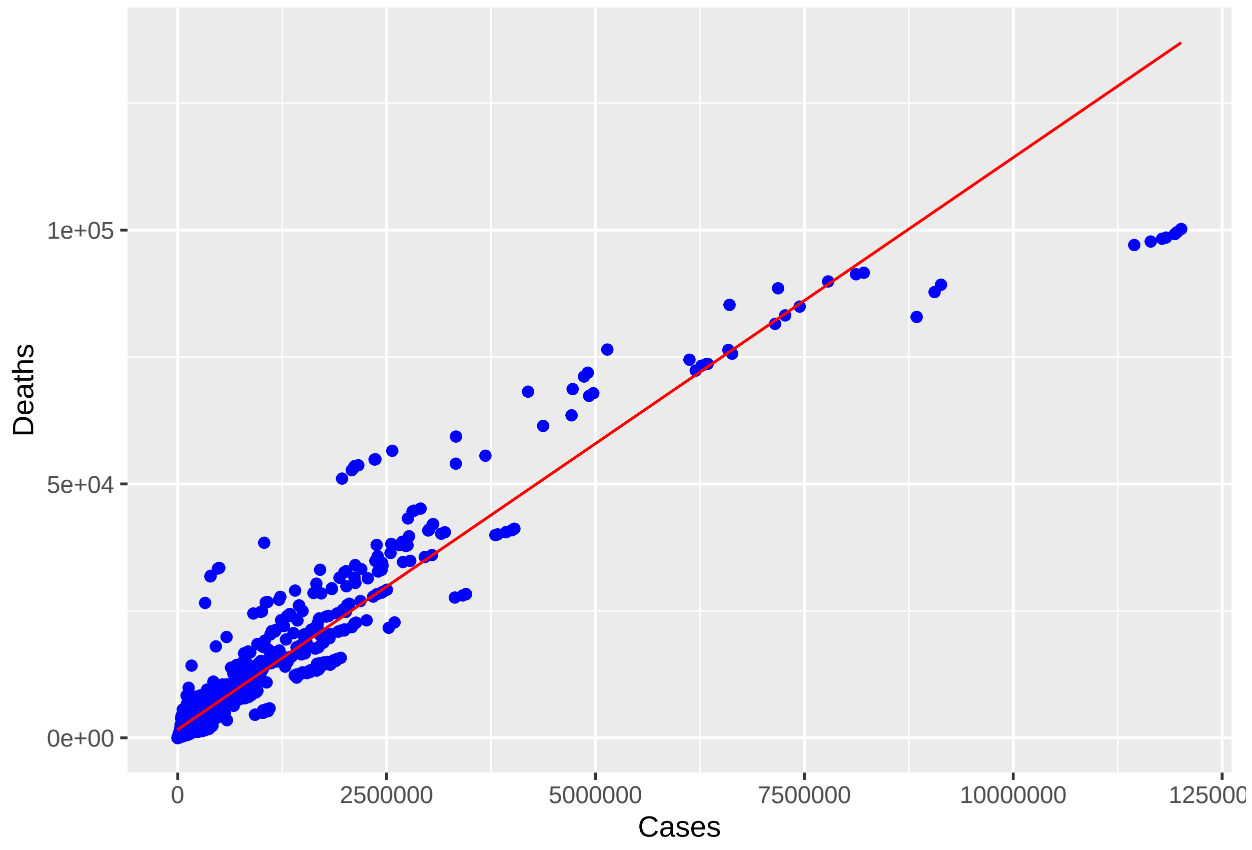
Model

Now linearly model it.

```r
model <- lm(Deaths ~ Cases, data = US_by_state)
summary(model)
```

```
##
## Call:
## lm(formula = Deaths ~ Cases, data = US_by_state)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -37215  -1627  -1280    871  28029
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.624e+03  2.263e+01   71.77   <2e-16 ***
## Cases       1.127e-02  1.382e-05  815.07   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5060 on 66292 degrees of freedom
## Multiple R-squared:  0.9093, Adjusted R-squared:  0.9093
## F-statistic: 6.643e+05 on 1 and 66292 DF,  p-value: < 2.2e-16
```

```
US_State_Predictions <- US_by_state %>%
  mutate(Predicted_Deaths = predict(model))

# It's plottin' time
US_State_Predictions %>%
  sample_n(1000) %>%
  ggplot(aes(x = Cases, y = Deaths)) +
  geom_point(aes(x = Cases, y = Deaths), color = "blue") +
  geom_line(aes(x = Cases, y = Predicted_Deaths), color = "red")
```

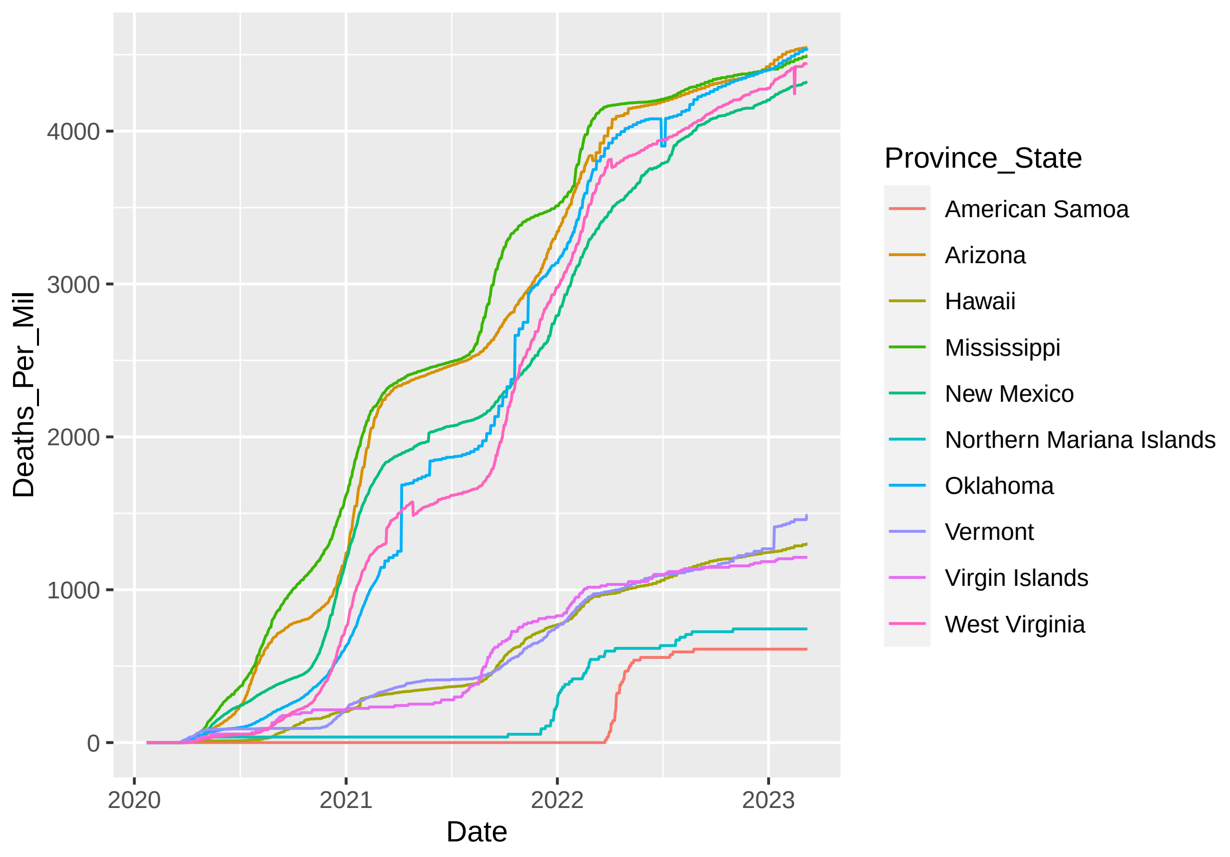


All over the place. The correlation is clear, but especially as cases increase, the prediction goes notably off. So there are probably other factors in play, such as lockdown measures working, vaccines being rolled out, whatever, and we could hunt down such information.

Extra Visualizations and Analysis

I'm interested in how different states can be pitted against each other in terms of proportional deaths. This results in an awful looking graph, however. Entirely too busy and such, so we look instead at the top and bottom few instead.

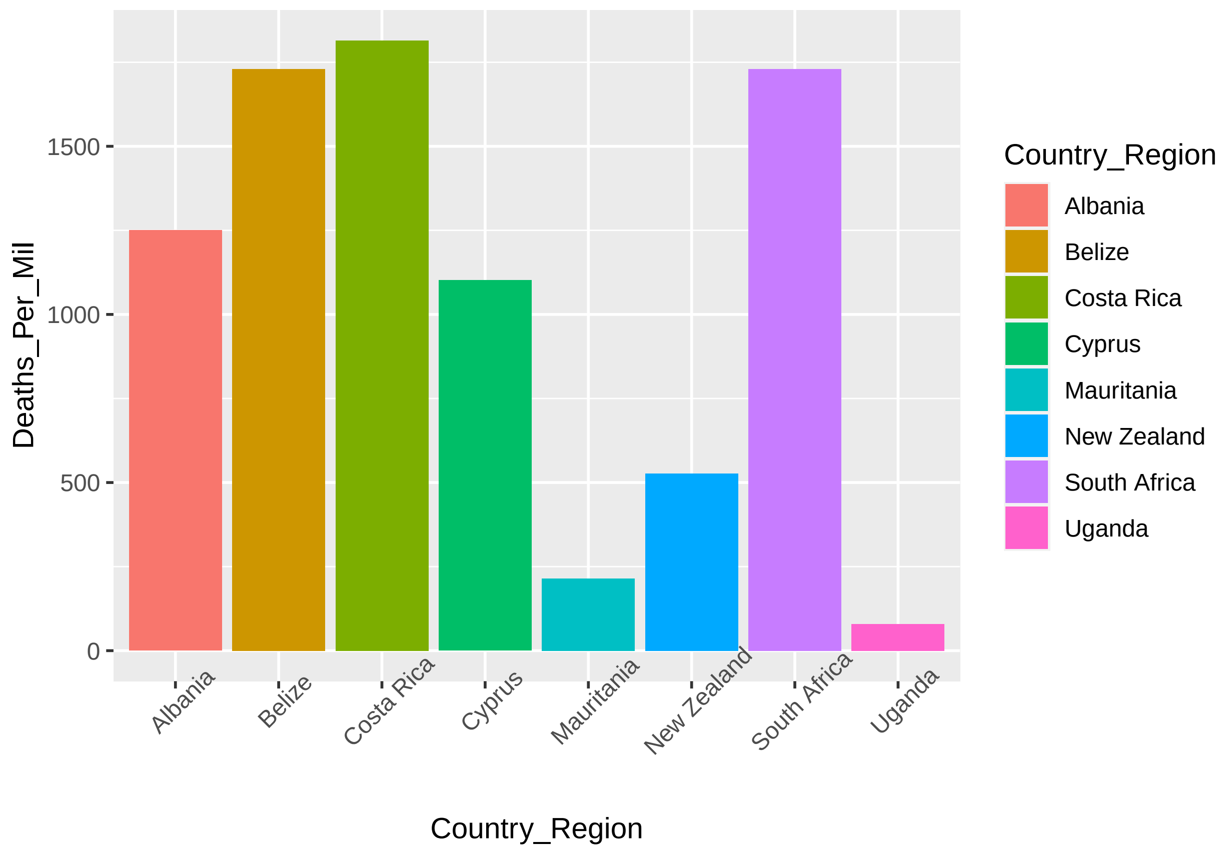
```r
# In case the data set changes in the future
latest <- US_by_state$Date %>%
  max(na.rm = TRUE)

# top and bottom X values, just here to change quickly
number_from_each <- 5

# Actually find the states of interest
least <- US_by_state %>%
  filter(Date == latest) %>%
  arrange(Deaths_Per_Mil) %>%
  head(number_from_each) %>%
  select(Province_State)

most <- US_by_state %>%
  filter(Date == latest) %>%
  # Get rid of those pesky cruise ships
  filter(Population > 0) %>%
  # Hm, more efficient to have a base object that's already sorted?
  arrange(Deaths_Per_Mil) %>%
  tail(number_from_each) %>%
  select(Province_State)

states_of_interest <- bind_rows(most, least)




# Visualize
US_by_state %>%
  filter(Province_State %in% states_of_interest$Province_State) %>%
  select(Province_State, Date, Deaths_Per_Mil) %>%
  group_by(Province_State) %>%

  ggplot(aes(x = Date, y = Deaths_Per_Mil, group = Province_State)) +
  geom_line(aes(color = Province_State))
```



In terms of analysis with the top/bottom 5 states/territories, we surprisingly cannot tell much.
The gap between the extremes is large, to be sure, but the territories at the bottom are mostly isolated locations - islands, so this makes perfect sense. The one stand-out is Vermont, though this could be a fluke. Some biases that could creep in from the perspective of a non-American is to see that the states with the worst death rates appear to be "American Conservative", while Vermont has a more "American Liberal" reputation.
This doesn't necessarily mean anything - and I might not even be right about how the political leanings line up, but to confirm, a good next step would be to maybe randomly sample states instead (as well as, you know, look up the political reaction to dealing with things).

And now a similar thing for worldwide data!
This time, I'll just look at the end result instead of the steady increase.
Also, I'll be taking a random sampling instead of top/bottom, since based on reporting that might not be that interesting.

```
latest <- max(global$Date)
number_to_sample <- 8

global_deaths_per_mil <- global %>%
  # Don't care about things other than the country as a whole
  select(-c(Combined_Key, Province_State)) %>%
  group_by(Country_Region) %>%
  filter(Date == latest) %>%
  # For combining cases with different "Province_State" vals
  summarize(Cases = sum(Cases), Deaths = sum(Deaths), Population = sum(Population)) %>%
  mutate(Deaths_Per_Mil = Deaths * 1000000 / Population) %>%
  # The only parts I care about
  select(Country_Region, Deaths_Per_Mil) %>%
  drop_na()

# Visualize time
global_deaths_per_mil %>%
  sample_n(number_to_sample) %>%
  ggplot(aes(x = Country_Region, y = Deaths_Per_Mil)) +
  geom_col(aes(fill = Country_Region)) +
  theme(axis.text.x = element_text(angle = 45))
```



Running the above code a few times has given me the impression that, above all, countries that (I am definitely biased in this) are more likely to - or have the reputation of being more likely to - report things accurately have consistently higher numbers.
This could just be because those countries are more globally connected in terms of consistent travel, like the UK and France, or it could simply be me justifying a pattern I vaguely noticed after the fact.
An interesting exception that caught my eye was Japan's low death per million count, though it is less surprising when you read up about how they dealt with the pandemic.

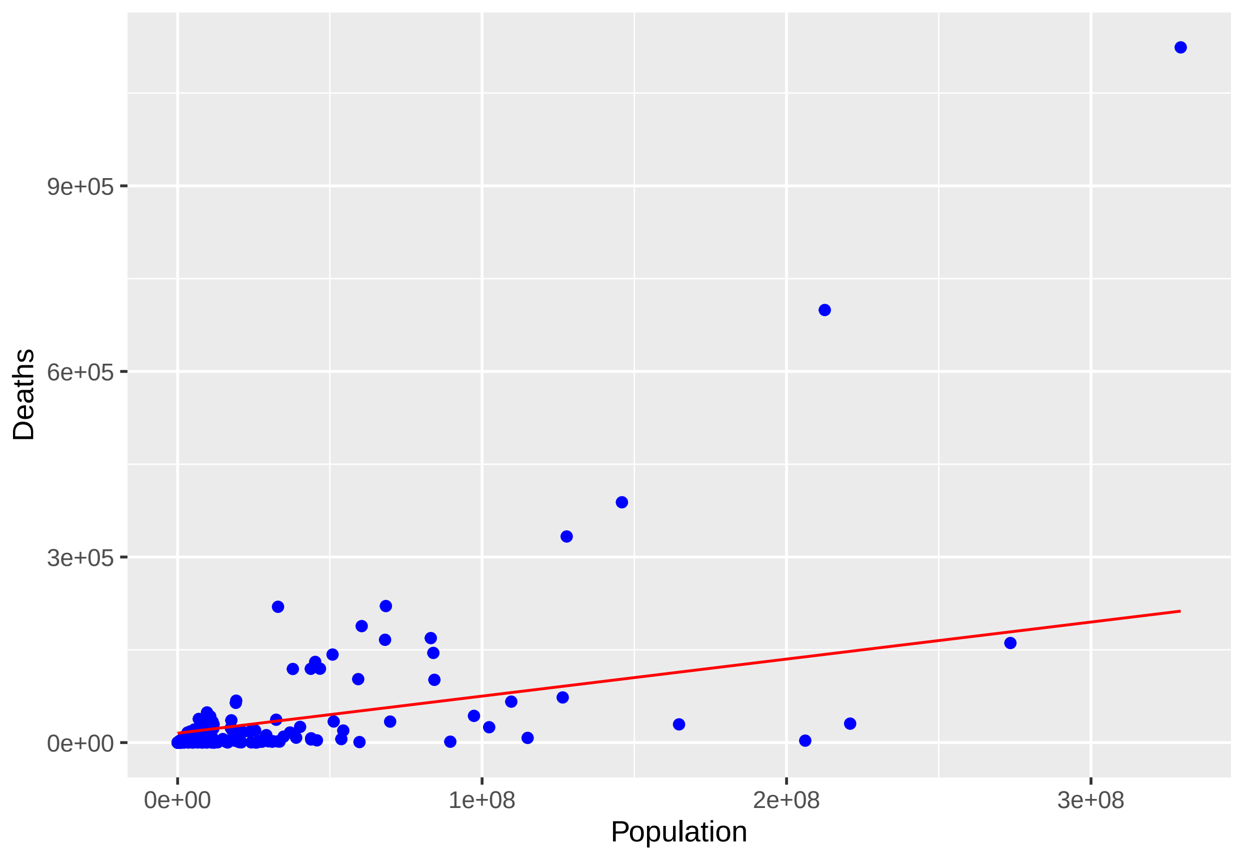
Extra Model

Instead of correlating cases to deaths, I want to look at the total population and how that corresponds to deaths.
I have an inkling that this won't be super obvious in outcome, which could lead to more interesting questions down the line.

```r
global_predictions <- global %>%
  group_by(Country_Region) %>%
  # Keeping it simple so there aren't 360k values to plot
  filter(Date == latest) %>%
  summarize(Cases = sum(Cases), Deaths = sum(Deaths), Population = sum(Population)) %>%
  drop_na()

global_model <- lm(Deaths ~ Population, data = global_predictions)

summary(global_model)
```

```
##
## Call:
## lm(formula = Deaths ~ Population, data = global_predictions)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -310760  -19666  -15153   -9095  911317
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.525e+04  6.926e+03   2.202   0.0289 *
## Population  5.988e-04  6.153e-05   9.731   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 92360 on 192 degrees of freedom
## Multiple R-squared:  0.3303, Adjusted R-squared:  0.3268
## F-statistic: 94.69 on 1 and 192 DF,  p-value: < 2.2e-16
```

```r
global_predictions <- global_predictions %>%
  mutate(Predicted_Deaths = predict(global_model))

global_predictions %>%
  # Literally just filtering out India because in this table it skews everything
  filter(Country_Region != "India") %>%
  # sample_n(10) %>%
  ggplot(aes(x = Population, y = Deaths)) +
  geom_point(aes(x = Population, y = Deaths), color = "blue") +
  geom_line(aes(x = Population, y = Predicted_Deaths), color = "red")
```



There doesn't seem to be a single angle or range or number of samples from which this relation fits a linear model.
This seems to, at least on the surface, tell us that the population of a place doesn't really have a bearing on how well they handled the pandemic.
This makes enough sense. Future questions would then be to narrow down why population itself is not a good indicator. My first guess is that there are just so many low-population isolated areas that are messing with the model, though based on the random sampling spree I went through, this didn't seem to change the accuracy.
My next hunch is that the nation's wealth, which may be tied in part to population, would be the primary factor at play. Especially since these data are from 2023, so we're not exactly in the heyday of the pandemic.