Analysis of Covid Cases and Deaths

ADP

2023-07-24

Import Necessary Libraries

Be sure tidyverse and lubridate are installed before beginning the analysis, as they will be needed to run the code properly.

Statement of Interest

The goal of this analysis is to better understand the impact of Covid both globally and in the United States. We look at variables such as cases and deaths and break down by region as well. Hopefully this analysis can provide some incremental insight into the impact of Covid.

Part 1 Importing the Data

The data comes from the Github page of the Center for Systems Science and Engineering (CSSE) at Johns Hopkins University (JHU), which collected data from the beginning of the crisis through March 10, 2023 (when they stopped collecting the data). Data after March 2023 can be found from the WHO and the CDC. They source their data from a variety of sources, which are clearly listed on their github page (https://github.com/CSSEGISandData/COVID-19).

Start by getting all the data in from the JHU Github **here.**

The data is in four separate files: US deaths, US cases, global deaths, and global cases. Each file lists locations and then the cases or deaths by date. This has all the information we need, although it's not in an ideal format. We will address that in a later step.

First step is to import the data directly from Github. I have broken the url into two parts for the sake of space. The two parts combined form the front part of the url for each file.

We then combine this front part with the endings for each of the files to get the specific url values for each of the four different files

```
# all files begin the same way
url_part1 <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/"
url_part2 <- "csse_covid_19_data/csse_covid_19_time_series/"
url_in    <- str_c(url_part1,url_part2)

## endings of the four different files
file_names <- c("time_series_covid19_confirmed_global.csv",
"time_series_covid19_deaths_global.csv",
"time_series_covid19_confirmed_US.csv",
"time_series_covid19_deaths_US.csv")
```

```
## full paths to the files combine the start with the endings
urls <- str_c(url_in, file_names)

## read in the files
global_cases  <- read_csv(urls[1], show_col_types = FALSE)
global_deaths <- read_csv(urls[2], show_col_types = FALSE)
us_cases      <- read_csv(urls[3], show_col_types = FALSE)
us_deaths     <- read_csv(urls[4], show_col_types = FALSE)
```

Part 2 Cleaning Up, Tidying, and Transforming the Data

We then clean up the global files to put them in a better format for R. Both files are very wide (horizontal) because each separate date is a column. We pivot to get the values into a vertical format for ease of use. We then combine the two into one data frame called "global" and filter to only include where cases are non-zero.

```
## make vertical rather than horizontal
global_cases <- global_cases %>%
  pivot_longer(cols = -c('Province/State',
                         'Country/Region', Lat, Long),
               names_to = "date",
               values_to = "cases") %>%
  select(-c(Lat,Long))

global_deaths <- global_deaths %>%
  pivot_longer(cols = -c('Province/State',
                         'Country/Region', Lat, Long),
               names_to = "date",
               values_to = "deaths") %>%
  select(-c(Lat,Long))

# combine global cases and deaths into one variable and remove dates with 0 cases
global <- global_cases %>%
  full_join(global_deaths) %>%
  rename(Country_Region = 'Country/Region',
         Province_State = 'Province/State') %>%
  mutate(date = mdy(date))
```

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

```
# only include where cases are greater than 0
global <- global %>% filter(cases>0)
```

Do the same analysis but for the US.

```
## make US more friendly for R
us_cases <- us_cases %>%
  pivot_longer(cols = -(UID:Combined_Key),
               names_to = "date",
               values_to = "cases") %>%
  select(Admin2:cases) %>%
  mutate(date = mdy(date)) %>%
```

```
  select(-c(Lat,Long_))

us_deaths <- us_deaths %>%
  pivot_longer(cols = -(UID:Population),
               names_to = "date",
               values_to = "deaths") %>%
  select(Admin2:deaths) %>%
  mutate(date = mdy(date)) %>%
  select(-c(Lat,Long_))

## combine us deaths and cases
US <- us_cases %>%
  full_join(us_deaths)
```

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

```
## US_test <- US %>% filter(cases>0)
## Note that this filter is NOT applied in the video analysis HERE but it is applied later
```

We add a combined key now to help match the global data set to the US data set. We then get population data from another file on the JHU Github page.

```
## add combined key for global to better match the US data
global <- global %>%
  unite("Combined_Key",
         c(Province_State, Country_Region),
         sep = ", ",
         na.rm = TRUE,
         remove = FALSE)

## get population data added to global data
u1 <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/"
u2 <- "csse_covid_19_data/UID_ISO_FIPS_LookUp_Table.csv"
uid_lookup_url <- str_c(u1,u2)

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

global <- global %>%
  left_join(uid, by = c("Province_State","Country_Region")) %>%
  select(-c(UID, FIPS)) %>%
  select(Province_State, Country_Region, date, cases, deaths, Population, Combined_Key)
```

Part 3 Visualizing Data

We first group the US by state in order to facilitate further analysis, then use this data (properly grouped) for the US as a whole. We also create a few variables such as deaths_per_mill to show the proportional impact rather than just an absolute value.

```
## do a state by state analysis of the US
US_by_state <- US %>%
  group_by(Province_State, Country_Region, date) %>%
  summarize(cases = sum(cases), deaths = sum(deaths), Population = sum(Population)) %>%
  mutate(deaths_per_mill = deaths *1000000/Population) %>%
  select(Province_State, Country_Region, date,
         cases, deaths, deaths_per_mill, Population) %>% ungroup
```

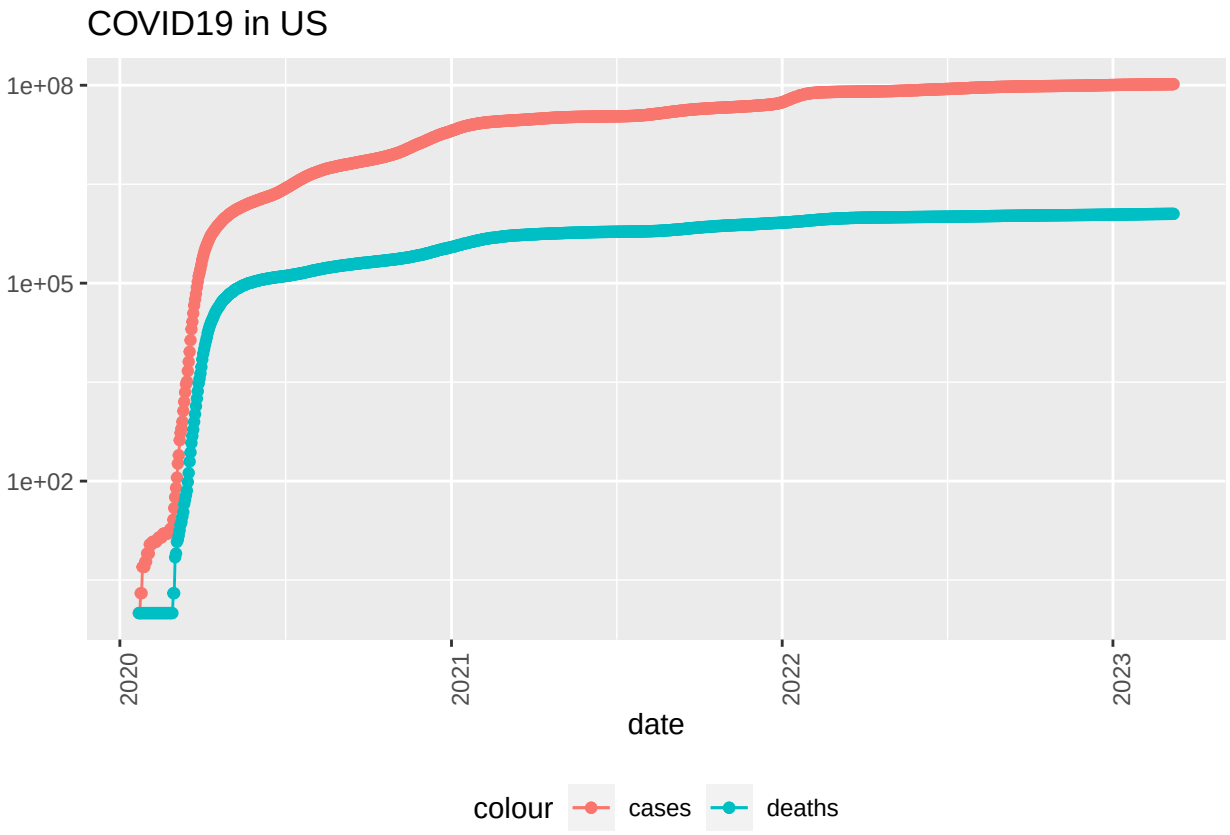
```
## `summarise()` has grouped output by 'Province_State', 'Country_Region'. You can
## override using the `.groups` argument.
```

```
## now get that into US total form
US_totals <- US_by_state %>%
  group_by(Country_Region, date) %>%
  summarize(cases=sum(cases), deaths=sum(deaths), Population=sum(Population)) %>%
  mutate(deaths_per_mill=deaths*1000000/Population) %>%
  select(Country_Region, date, cases, deaths, deaths_per_mill, Population) %>%
  ungroup()
```

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

Next we graph the US totals over time, filtering out where there weren't any cases.

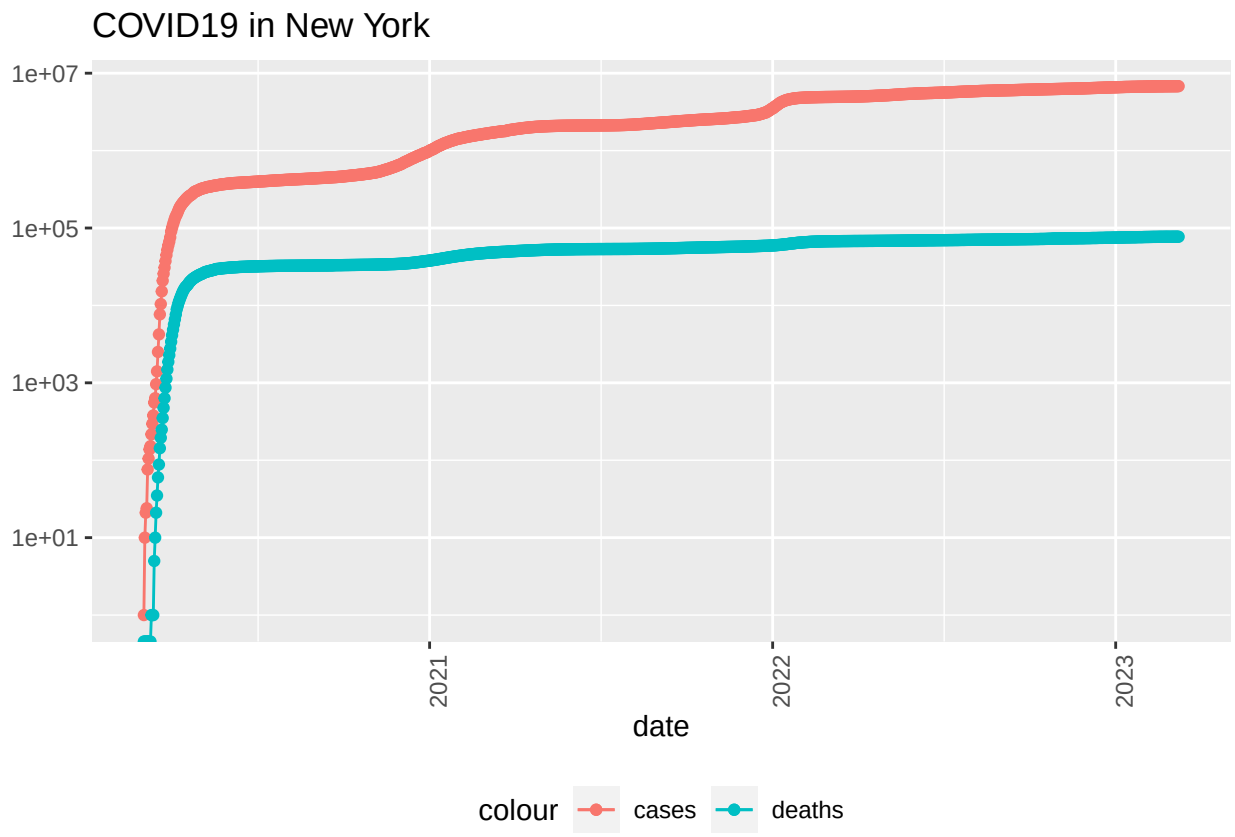
```
## graph the data
US_totals %>%
  filter(cases>0) %>%
  ggplot(aes(x=date,y=cases)) +
  geom_line(aes(color="cases")) +
  geom_point(aes(color="cases")) +
  geom_line(aes(y=deaths, color="deaths")) +
  geom_point(aes(y=deaths, color="deaths")) +
  scale_y_log10() +
  theme(legend.position="bottom",
        axis.text.x = element_text(angle=90)) +
  labs(title="COVID19 in US",y=NULL)
```



Next we graph for a particular state. While I have chosen New York, you are welcome to change the code in the first line to another state to see how that looks

```
## graph by state using NY as the example
state <- "New York"
US_by_state %>%
  filter(Province_State == state) %>%
  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
```



Part 4 Analyzing Data

Now that we have imported, processed, cleaned, transformed, and visualized the data, our next task is to perform some new analyses on it. The graphs seem to suggest a leveling in terms of total cases, so we now look at the incremental new cases or new deaths on both a total US basis and a state by state basis. We start by creating the new variables.

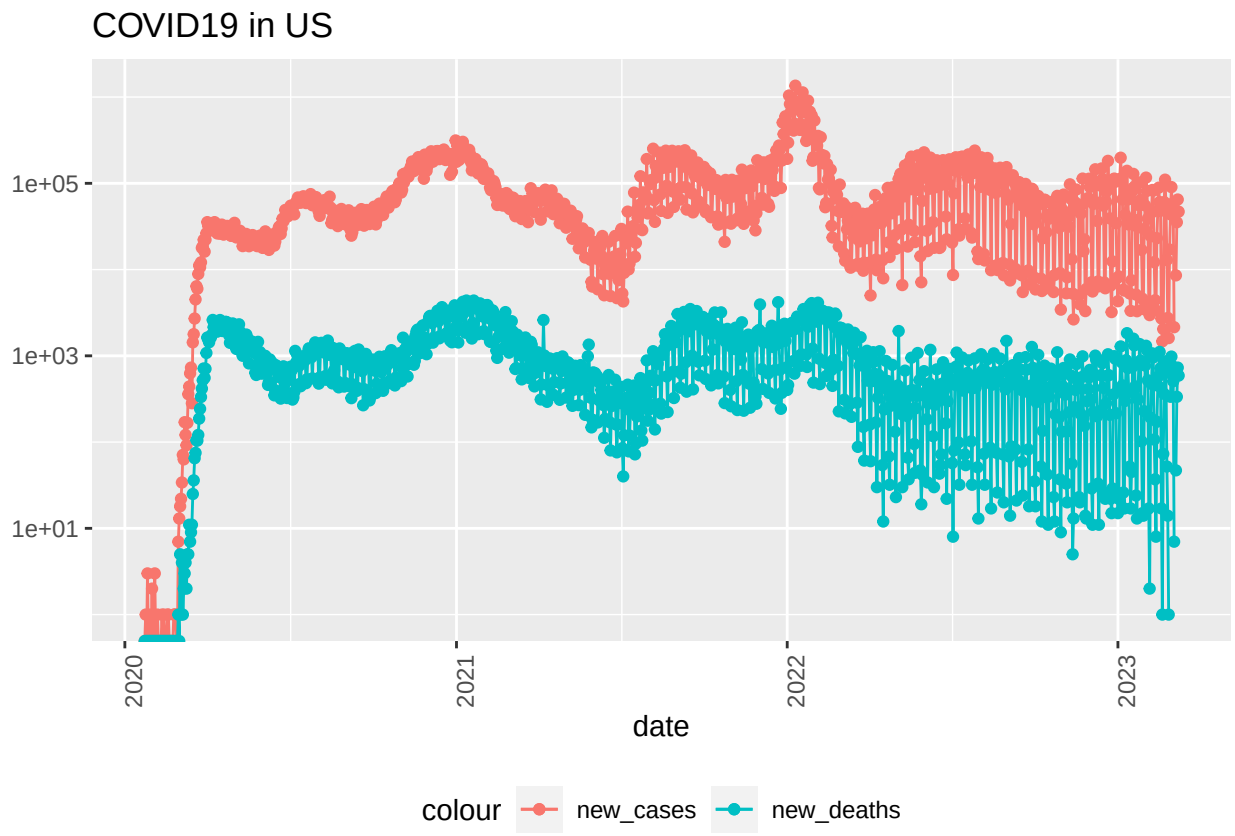
```
## new cases and deaths by date
US_by_state <- US_by_state %>%
  mutate(new_cases = cases - lag(cases),
         new_deaths = deaths - lag(deaths))

US_totals <- US_totals %>%
  mutate(new_cases = cases - lag(cases),
         new_deaths = deaths - lag(deaths))
```

Now we graph the new cases and new deaths for the US. Note that new deaths and new cases have both come down somewhat but still remain quite high. There is also a large amount of variability inherent in the data. Note that there are some dates where new cases or new deaths fall below 0, which suggests there are at least a few data points in the set with quality issues.

Note that I have removed warnings for the sake of space and clarity but would strongly advise anyone altering this code to include them initially to make sure that nothing is wrong with the data.

```
US_totals %>%
  ggplot(aes(x=date,y=new_cases)) +
  geom_line(aes(color="new_cases")) +
  geom_point(aes(color="new_cases")) +
  geom_line(aes(y=new_deaths, color="new_deaths")) +
  geom_point(aes(y=new_deaths,color="new_deaths")) +
  scale_y_log10() +
  theme(legend.position="bottom",
        axis.text.x = element_text(angle=90)) +
  labs(title="COVID19 in US",y=NULL)
```

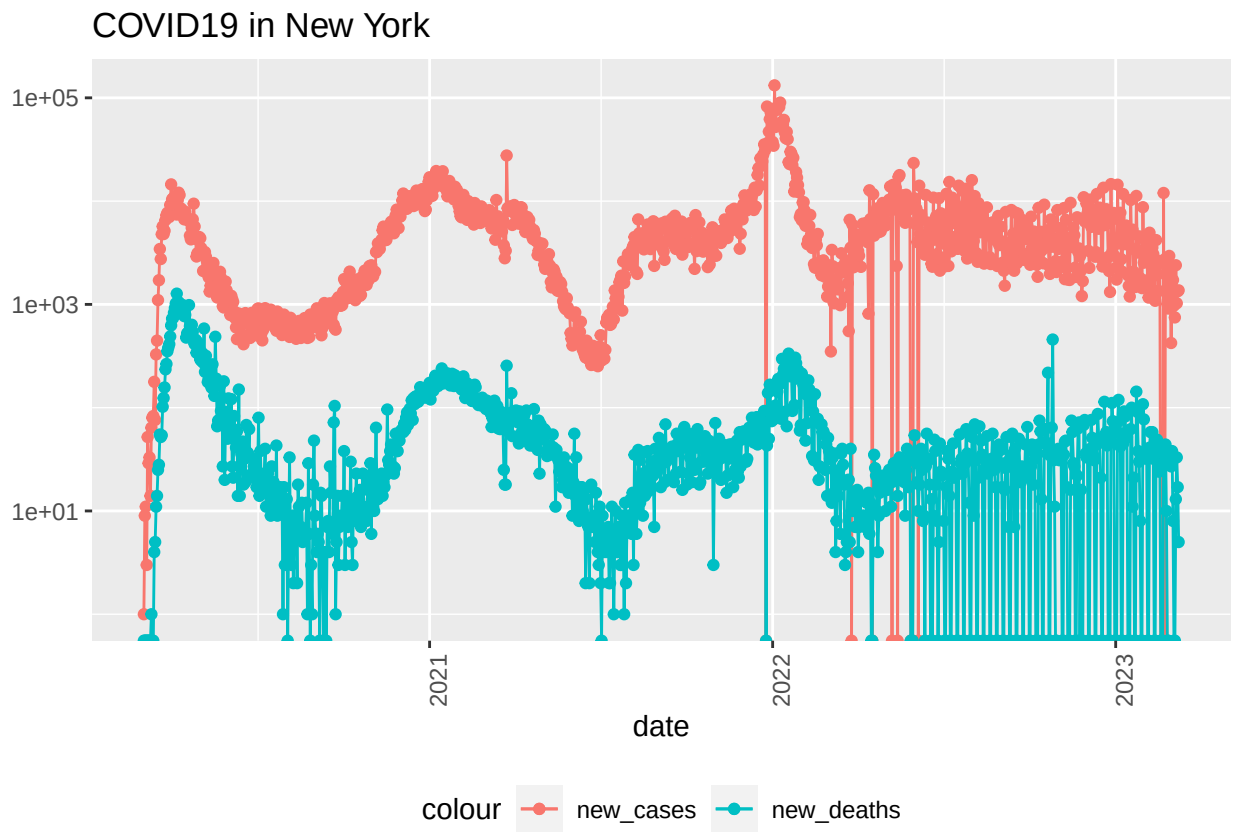



```
## US_totals[US_totals$new_cases<0, ] or US_totals[US_totals$new_deaths<0, ]
## There are a few dates where new cases or new deaths are below 0
## This suggests some quality or adjustment issues with data
```

We now look at the new cases and new deaths on an individual state basis. I have again used NY but you are welcome to change the state to whatever entity is relevant for you. While cases have come down somewhat, the number of new deaths has frequently approached 0 in late 2022 and early 2023.

```
state <- "New York"
US_by_state %>%
  filter(Province_State == state, cases>0) %>%
  ggplot(aes(x=date,y=new_cases)) +
  geom_line(aes(color="new_cases")) +
  geom_point(aes(color="new_cases")) +
```

```
  geom_line(aes(y=new_deaths, color="new_deaths")) +
  geom_point(aes(y=new_deaths,color="new_deaths")) +
  scale_y_log10() +
  theme(legend.position="bottom",
        axis.text.x = element_text(angle=90)) +
  labs(title=str_c("COVID19 in ",state), y=NULL)
```



Additional Analysis on Ireland (Not in Video)

First we take the global variable and group by country. We then create the new deaths and new cases variables as we did for the US and its states. We then filter out so that only Ireland remains and graph in the same way as above. The data shows a similar pattern with deaths and new cases peaking around the end of 2021 before decreasing substantially in 2022 and early 2023 (approaching and reaching zero in many cases but with extremely high variability). The main difference is scale, as Ireland has a much smaller population than either the US or even New York state.

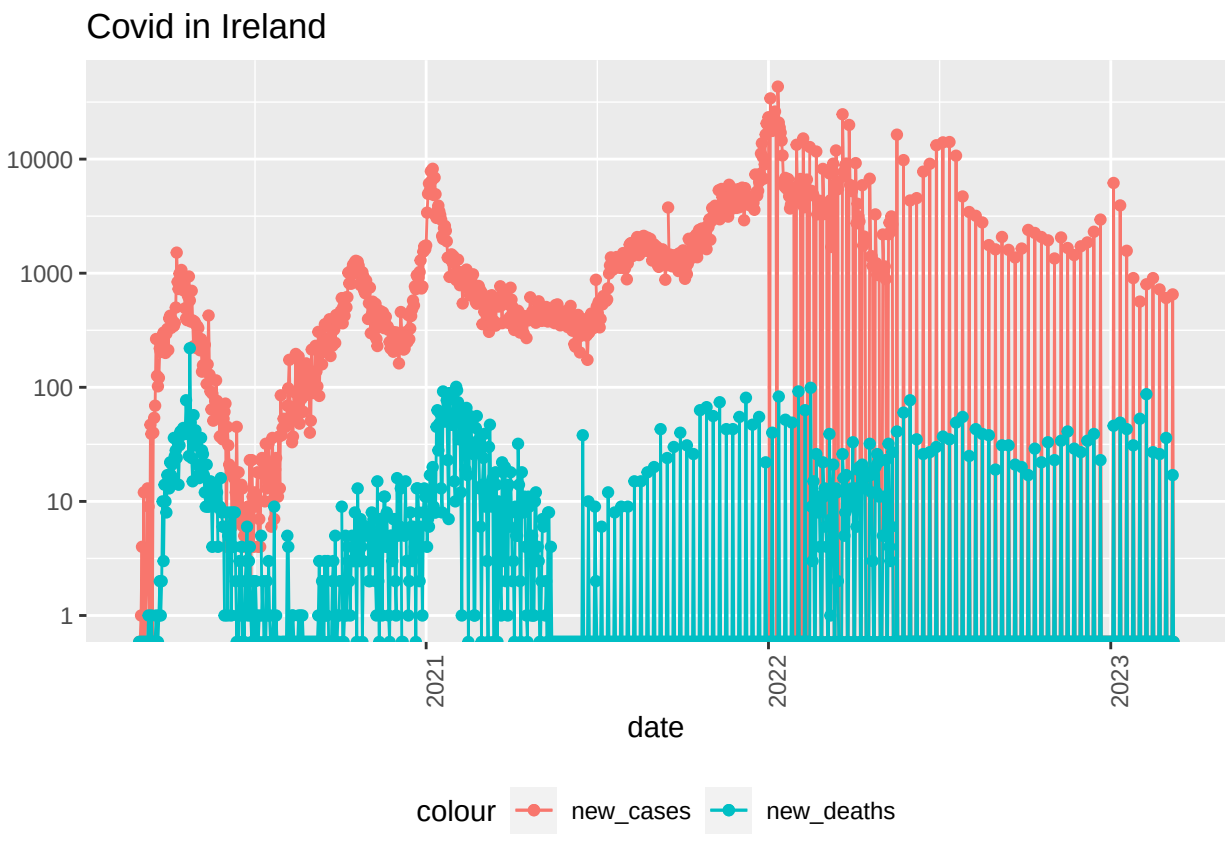
```
global_by_country <- global %>%
  group_by(Country_Region, date) %>%
  summarize(cases=sum(cases),
            deaths=sum(deaths),
            Population=sum(Population)) %>%
  mutate(deaths_per_mill=deaths*1000000/Population) %>%
  select(Country_Region, date, cases, deaths, deaths_per_mill, Population) %>%
  ungroup()
```

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

```r
## new cases and deaths by date
global_by_country <- global_by_country %>%
  mutate(new_cases = cases - lag(cases),
         new_deaths = deaths - lag(deaths))

## only focus on the country of Ireland
ireland <- global_by_country %>% filter(Country_Region == "Ireland", cases>0)

## graph the cases and deaths in Ireland
ireland %>%
  ggplot(aes(x=date,y=new_cases)) +
  geom_line(aes(color="new_cases")) +
  geom_point(aes(color="new_cases")) +
  geom_line(aes(y=new_deaths, color="new_deaths")) +
  geom_point(aes(y=new_deaths,color="new_deaths")) +
  scale_y_log10() +
  theme(legend.position="bottom",
        axis.text.x = element_text(angle=90)) +
  labs(title="Covid in Ireland", y=NULL)
```



Part 5 Highest and Lowest Death Rates

Now we want to see which states have the highest and lowest death rates calculated as deaths_per_thou in the code below. In order to do this we first group everything by state. We then find the maximum values for deaths and cases (which would be the final values representing the total numbers of deaths and cases at the end of the data set). We then calculate the death rate and case rate based on the state population.

The charts below are the regions with the lowest and the highest death rates. The lowest death rates appear to be the most isolated states or regions, while those in the south or southwest appear to have the highest death rates. This analysis could be furthered by including information regarding population density, connectivity to other regions (perhaps those with more transportation hubs are more likely to experience high deaths), and differences in state-wide Covid policies.

```
## What state is worst? What's the most apt measure?
US_state_totals <- US_by_state %>%
  group_by(Province_State) %>%
  summarize(deaths = max(deaths), cases = max(cases),
            population = max(Population),
            cases_per_thou = 1000*cases/population,
            deaths_per_thou = 1000*deaths/population) %>%
  filter(cases>0, population>0)

## states with lowest death rates
US_state_totals %>%
  slice_min(deaths_per_thou, n=10) %>%
  select(Province_State, cases_per_thou, deaths_per_thou)
```

```
## # A tibble: 10 x 3
##    Province_State           cases_per_thou deaths_per_thou
##    <chr>                             <dbl>           <dbl>
##  1 American Samoa                     150.           0.611
##  2 Northern Mariana Islands           248.           0.744
##  3 Virgin Islands                     231.           1.21
##  4 Hawaii                             269.           1.30
##  5 Vermont                            245.           1.49
##  6 Puerto Rico                        293.           1.55
##  7 Utah                               340.           1.65
##  8 Alaska                             415.           2.01
##  9 District of Columbia               252.           2.03
## 10 Washington                         253.           2.06
```

```
## states with highest death rates
## add everything() to the select clause to autofill the other cols
US_state_totals %>%
  slice_max(deaths_per_thou, n=10) %>%
  select(Province_State, cases_per_thou, deaths_per_thou)
```

```
## # A tibble: 10 x 3
##    Province_State cases_per_thou deaths_per_thou
##    <chr>                   <dbl>           <dbl>
##  1 Arizona                  336.            4.55
##  2 Oklahoma                 326.            4.54
##  3 Mississippi              333.            4.49
```

```
##  4 West Virginia            359.           4.44
##  5 New Mexico               320.           4.32
##  6 Arkansas                 334.           4.31
##  7 Alabama                  335.           4.29
##  8 Tennessee                368.           4.28
##  9 Michigan                 307.           4.23
## 10 Kentucky                 385.           4.06
```

Part 6 Modeling the Data

The final part of our exercise is to model out the data. Our model will try to predict death rate based on the case rate, which seems to be a logical hypothesis. Deviations from this prediction line imply that more or fewer deaths occurred than would have been expected given the number of cases.

The model is highly significant, as cases per thousand is a significant predictor of deaths per thousand

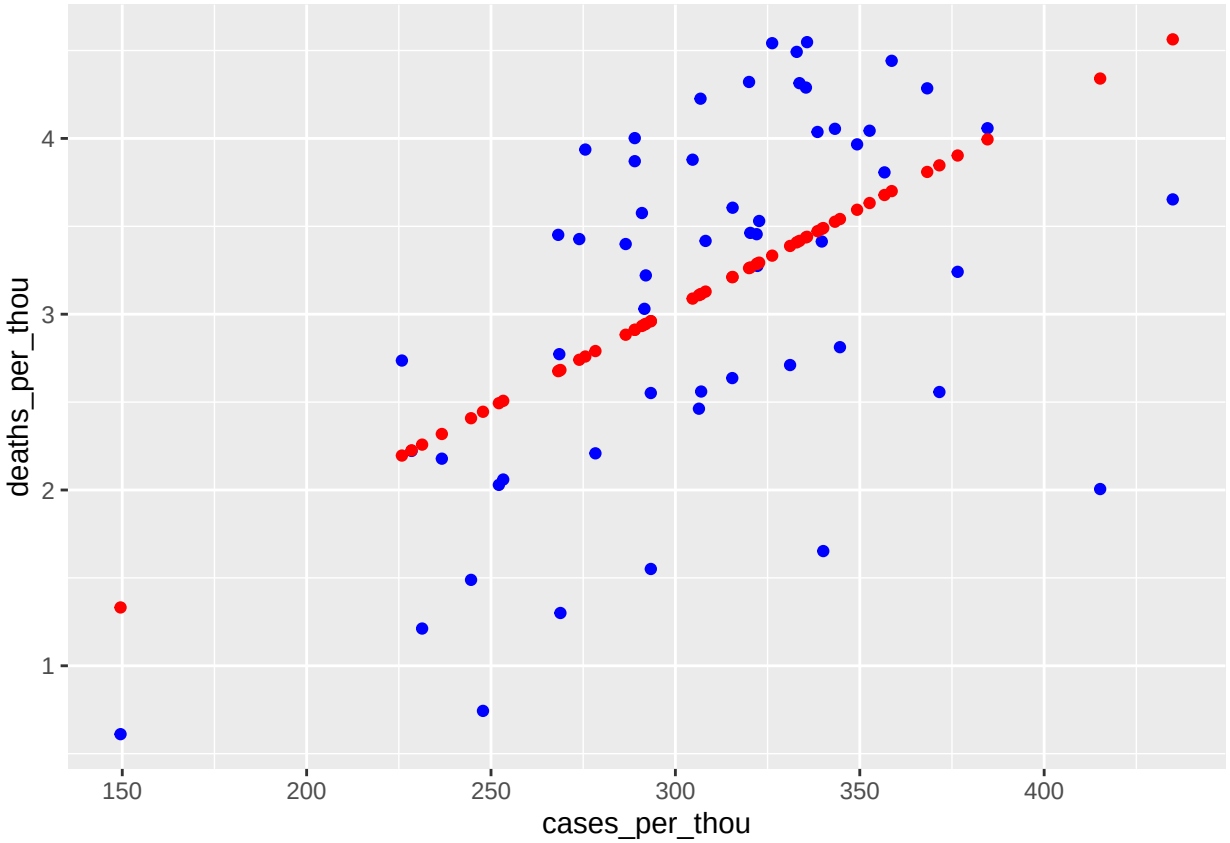
```
## introduction to modeling
mod = lm(deaths_per_thou ~ cases_per_thou, data = US_state_totals)
summary(mod)
```

```
##
## Call:
## lm(formula = deaths_per_thou ~ cases_per_thou, data = US_state_totals)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.3352 -0.5979  0.1491  0.6535  1.2086
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -0.36167    0.72480  -0.499     0.62
## cases_per_thou  0.01133    0.00232   4.881 9.76e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8615 on 54 degrees of freedom
## Multiple R-squared:  0.3061, Adjusted R-squared:  0.2933
## F-statistic: 23.82 on 1 and 54 DF,  p-value: 9.763e-06
```

We end the modeling portion by adding the predictions to our data set and graphing the results, which shows that the model is a good predictor, but significant deviations from the trend line are readily identified.

```
## adding predicted values to data set
US_tot_w_pred <- US_state_totals %>% mutate(pred = predict(mod))

## graph the dataset
US_tot_w_pred %>%
  ggplot() +
  geom_point(aes(x=cases_per_thou, y=deaths_per_thou), color="blue") +
  geom_point(aes(x=cases_per_thou, y=pred), color="red")
```



Part 7 Conclusion

The goal of this report was to explore data about Covid cases and deaths both globally and in the United States in particular (including on a state-by-state basis). We started by importing the data from JHU's Github. We then cleaned and transformed the data as necessary to better work with it in R. We also created visualizations to help us understand the data.

We then created new variables that reflect the new cases and new deaths by subtracting the lag from the present value to get a change over time. The data shows that while total cases and total deaths continue to rise, new cases and new deaths began to drop in 2022 and continued to get lower through early 2023. This trend held true in NY state and in Ireland as well (which I added as a supplement to the usual analysis).

We also looked at the states with the highest and lowest death rates of Covid, with the lowest being found in the most isolated regions while the highest rates were in the South and Southwest. We also did a linear regression model that found rate of cases to be a statistically significant but imperfect predictor of death rate.

While this analysis was interesting, it could be hampered by several sources of bias. First, the data was aggregated from a number of different sources, including different states, countries, and regions, all of which might have an incentive to distort the data (for example, some countries may have intentionally underreported infection rates to placate their citizens, assuage fears about public health, or to prevent panic).

Another source of bias could be how data categories are defined. Even if we assume all regions are 100% accurate, imagine a situation where an individual has an underlying condition that is exacerbated by Covid. If the person passes away, is it counted as a Covid death? Similarly, what about an individual that suffered from one condition but was unable to get treatment during the pandemic due to the prevalence of Covid

- would that person be counted as a Covid death? They weren't directly killed by the disease but it did indirectly cause their death.

Finally, we need to consider variables that would be helpful and likely statistically significant but were not in the data set. For example, population density was not in the original data sets but could be significant. Kazakhstan and the NY metro area have similar populations despite the fact that Kazakhstan is ~22 times the size. Understanding how dense a country or region is could help predict the outcomes better.

Overall, the research was quite interesting, but Covid remains a controversial and complex topic that is deserving of more thorough study.

```
sessionInfo()
```

```
## R version 4.3.0 (2023-04-21 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 11 x64 (build 22621)
##
## Matrix products: default
##
##
## locale:
## [1] LC_COLLATE=English_United States.utf8
## [2] LC_CTYPE=English_United States.utf8
## [3] LC_MONETARY=English_United States.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.utf8
##
## time zone: America/New_York
## tzcode source: internal
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] lubridate_1.9.2 forcats_1.0.0   stringr_1.5.0   dplyr_1.1.2
##  [5] purrr_1.0.1     readr_2.1.4     tidyr_1.3.0     tibble_3.2.1
##  [9] ggplot2_3.4.2   tidyverse_2.0.0
##
## loaded via a namespace (and not attached):
##  [1] bit_4.0.5        gtable_0.3.3     highr_0.10       crayon_1.5.2
##  [5] compiler_4.3.0   tidyselect_1.2.0 parallel_4.3.0   scales_1.2.1
##  [9] yaml_2.3.7       fastmap_1.1.1    R6_2.5.1         labeling_0.4.2
## [13] generics_0.1.3   curl_5.0.0       knitr_1.42       munsell_0.5.0
## [17] pillar_1.9.0     tzdb_0.3.0       rlang_1.1.1      utf8_1.2.3
## [21] stringi_1.7.12   xfun_0.39        bit64_4.0.5      timechange_0.2.0
## [25] cli_3.6.1        withr_2.5.0      magrittr_2.0.3   digest_0.6.31
## [29] grid_4.3.0       vroom_1.6.3      rstudioapi_0.14  hms_1.1.3
## [33] lifecycle_1.0.3  vctrs_0.6.2      evaluate_0.20    glue_1.6.2
## [37] farver_2.1.1     fansi_1.0.4      colorspace_2.1-0 rmarkdown_2.23
## [41] tools_4.3.0      pkgconfig_2.0.3  htmltools_0.5.5
```