

Covid19

CU Student

6/23/2021

Importing nessary library and download CSV data from the Github Data Repository by the Center for Systems Science and Engineering (CSSE) at Johns Hopkins University (https://github.com/CSSEGISandData/COVID-19/tree/master/csse_covid_19_data/csse_covid_19_time_series).

```
library(tidyverse)

## -- Attaching packages ----- tidyverse 1.3.1 --

## v ggplot2 3.3.3     v purrr    0.3.4
## v tibble   3.1.2     v dplyr     1.0.6
## v tidyr    1.1.3     v stringr   1.4.0
## v readr    1.4.0     vforcats  0.5.1

## -- Conflicts ----- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()   masks stats::lag()

library(lubridate)

##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
## 
##     date, intersect, setdiff, union

uid_lookup_url <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/
url_in <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_time_series.csv"

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

##
## -- Column specification -----
## cols(
##   UID = col_double(),
```

```

##    iso2 = col_character(),
##    iso3 = col_character(),
##    code3 = col_double(),
##    FIPS = col_character(),
##    Admin2 = col_character(),
##    Province_State = col_character(),
##    Country_Region = col_character(),
##    Lat = col_double(),
##    Long_ = col_double(),
##    Combined_Key = col_character(),
##    Population = col_double()
##  )

file_names <- 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")
urls <- str_c(url_in, file_names)

US_cases <- read_csv(urls[1])

##
## -- Column specification -----
## cols(
##   .default = col_double(),
##   iso2 = col_character(),
##   iso3 = col_character(),
##   Admin2 = col_character(),
##   Province_State = col_character(),
##   Country_Region = col_character(),
##   Combined_Key = col_character()
## )
## i Use `spec()` for the full column specifications.

global_cases <- read_csv(urls[2])

##
## -- Column specification -----
## cols(
##   .default = col_double(),
##   'Province/State' = col_character(),
##   'Country/Region' = col_character()
## )
## i Use `spec()` for the full column specifications.

US_deaths <- read_csv(urls[3])

##
## -- Column specification -----
## cols(
##   .default = col_double(),

```

```

##   iso2 = col_character(),
##   iso3 = col_character(),
##   Admin2 = col_character(),
##   Province_State = col_character(),
##   Country_Region = col_character(),
##   Combined_Key = col_character()
## )
## i Use 'spec()' for the full column specifications.

```

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

```

##
## -- Column specification -----
## cols(
##   .default = col_double(),
##   'Province/State' = col_character(),
##   'Country/Region' = col_character()
## )
## i Use 'spec()' for the full column specifications.

```

Create and tidy Dataframes with R frendly colum names and drop coloums not needed for this assignment. All the of the final dataframes are created in this code block, with required coloums and data points.

```

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))

global <- global_cases %>%
  full_join(global_deaths) %>%
  rename(Country_Region = 'Country/Region',
         Province_State = 'Province/State') %>%
  mutate(date = mdy(date)) %>%
  filter(cases > 0) %>%
  unite("Combined_Key",
        c(Province_State, Country_Region),
        sep = ", ")

```

```

    na.rm = TRUE,
    remove = FALSE) %>%
left_join(uid, by = c("Province_State", "Country_Region")) %>%
select(-c(UID, FIPS)) %>%
select(Province_State, Country_Region, date, cases,
       deaths, Population, Combined_Key)

```

```
## Joining, by = c("Province/State", "Country/Region", "date")
```

```

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_))

```

```

US <- US_cases %>%
  full_join(US_deaths)

```

```
## Joining, by = c("Admin2", "Province_State", "Country_Region", "Combined_Key", "date")
```

```

US_by_state <- US %>%
  group_by(Province_State, Country_Region, date) %>%
  summarise(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 ` `
```

```

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

```

```

US_totals <- US_by_state %>%
  group_by(Country_Region, date) %>%
  summarise(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.

By_country <- global %>%
  group_by(Province_State, Country_Region, date) %>%
  summarise(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 '.

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

US_date <- US_by_state %>%
  mutate(Month = str_sub(date,6,7), # Separating the Month
        Day = str_sub(date,9,10), # Separating the Day
        Year = str_sub(date,1,4)) # Separating the year

global <- global %>%
  mutate(new_cases = cases - lag(cases),
        new_deaths = deaths - lag(deaths),
        Month = str_sub(date,6,7),
        Day = str_sub(date,9,10),
        Year = str_sub(date,1,4))

```

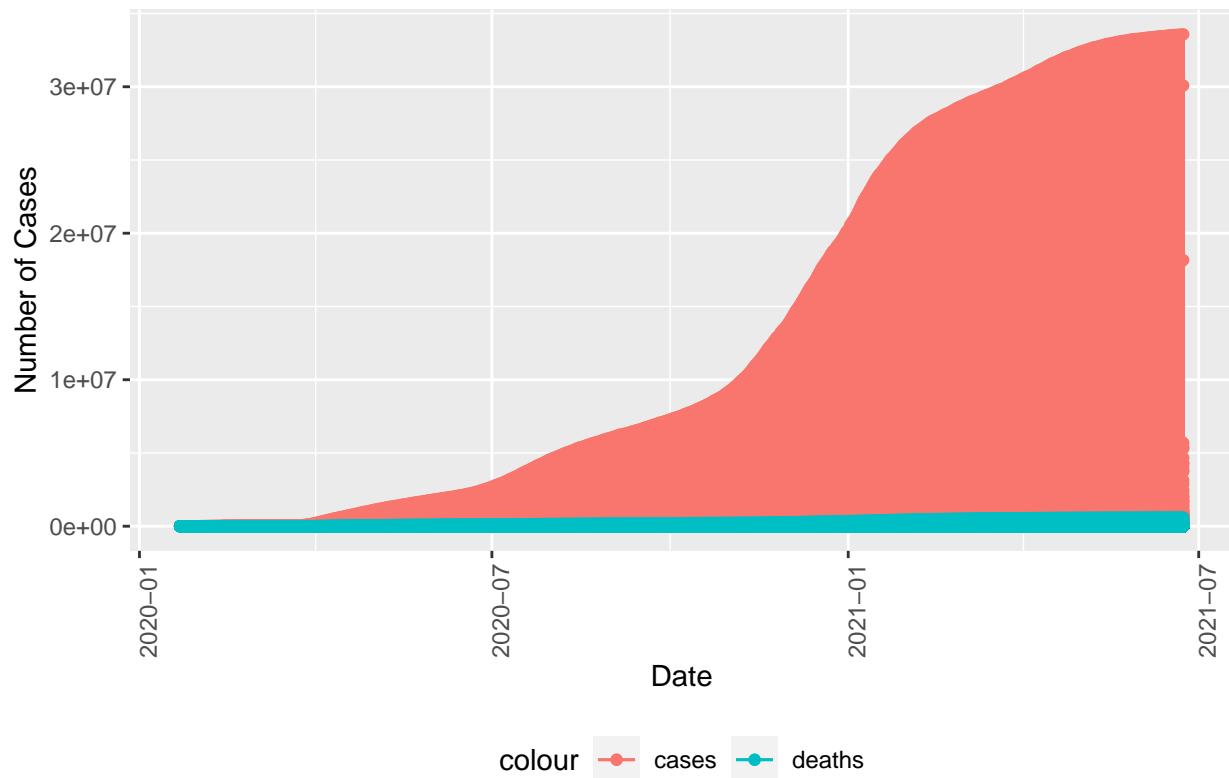
Plotting global and US COVID19 cases and deaths graphs. I made an effort to graph each individual US states, however due to fixed graph size it's barely readable, breaking it down country wise (using global dataframe) is every worse (I am unable to figure out how to scale the graph size). The facet_wrap() R function is not very useful here, since we have lot of individual sub-graphs. This worked very nicely in NYPD shooting dataset where data was facet_wrap() with yearly sub-graphs.

```

global %>%
  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")) +
  theme(legend.position="bottom",
        axis.text.x = element_text(angle = 90)) +
  labs(title = "Global COVID19 cases and deaths", x = "Date", y = "Number of Cases")

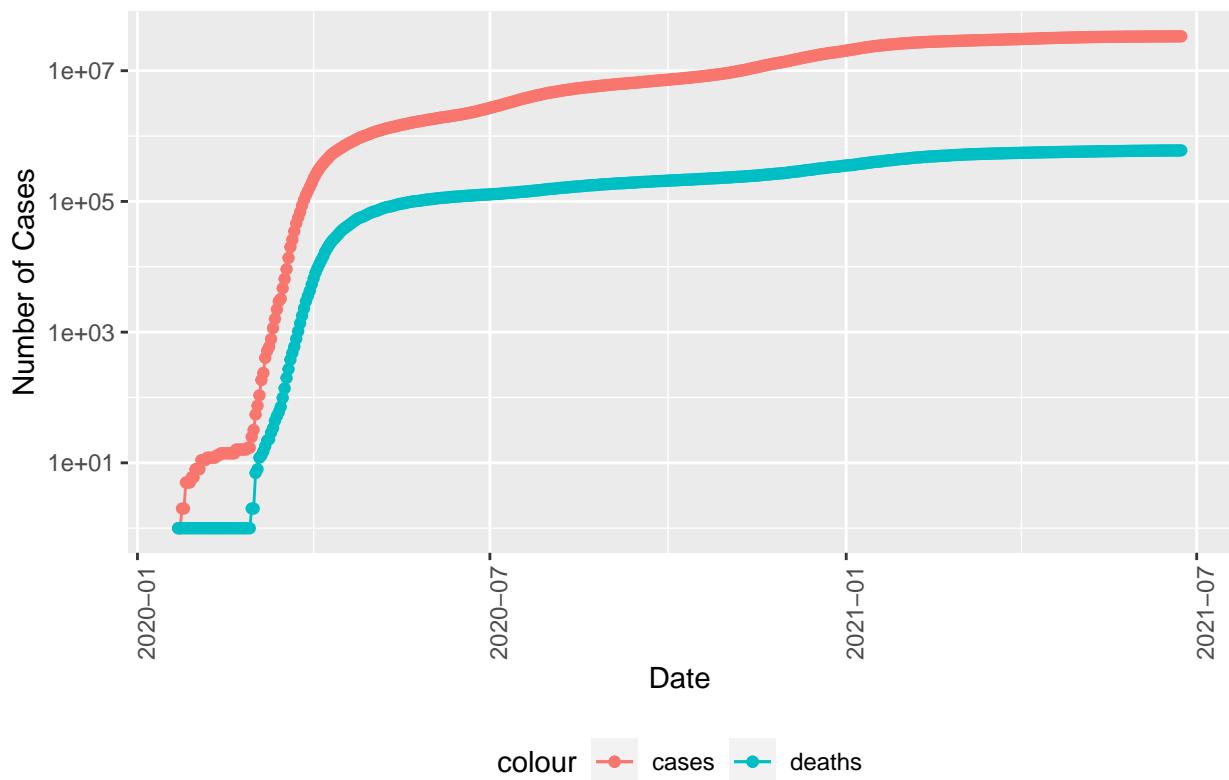
```

Global COVID19 cases and deaths



```
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", x = "Date", y = "Number of Cases")
```

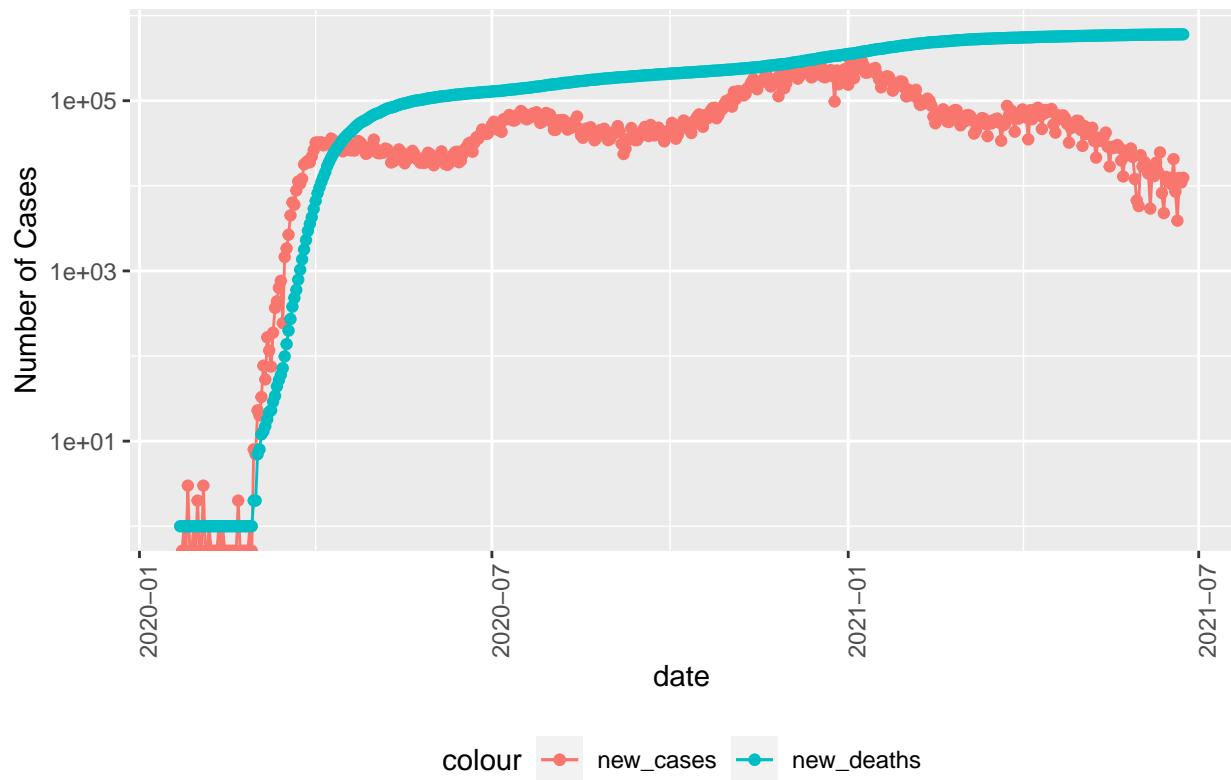
COVID19 in US



```
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 = deaths, color = "new_deaths")) +
  geom_point(aes(y = deaths, color = "new_deaths")) +
  scale_y_log10() +
  theme(legend.position="bottom",
        axis.text.x = element_text(angle = 90)) +
  labs(title = "New COVID19 cases/deaths in US", y = "Number of Cases")
```

```
## Warning: Transformation introduced infinite values in continuous y-axis
## Warning: Transformation introduced infinite values in continuous y-axis
## Warning: Removed 1 row(s) containing missing values (geom_path).
## Warning: Removed 1 rows containing missing values (geom_point).
```

New COVID19 cases/deaths in US

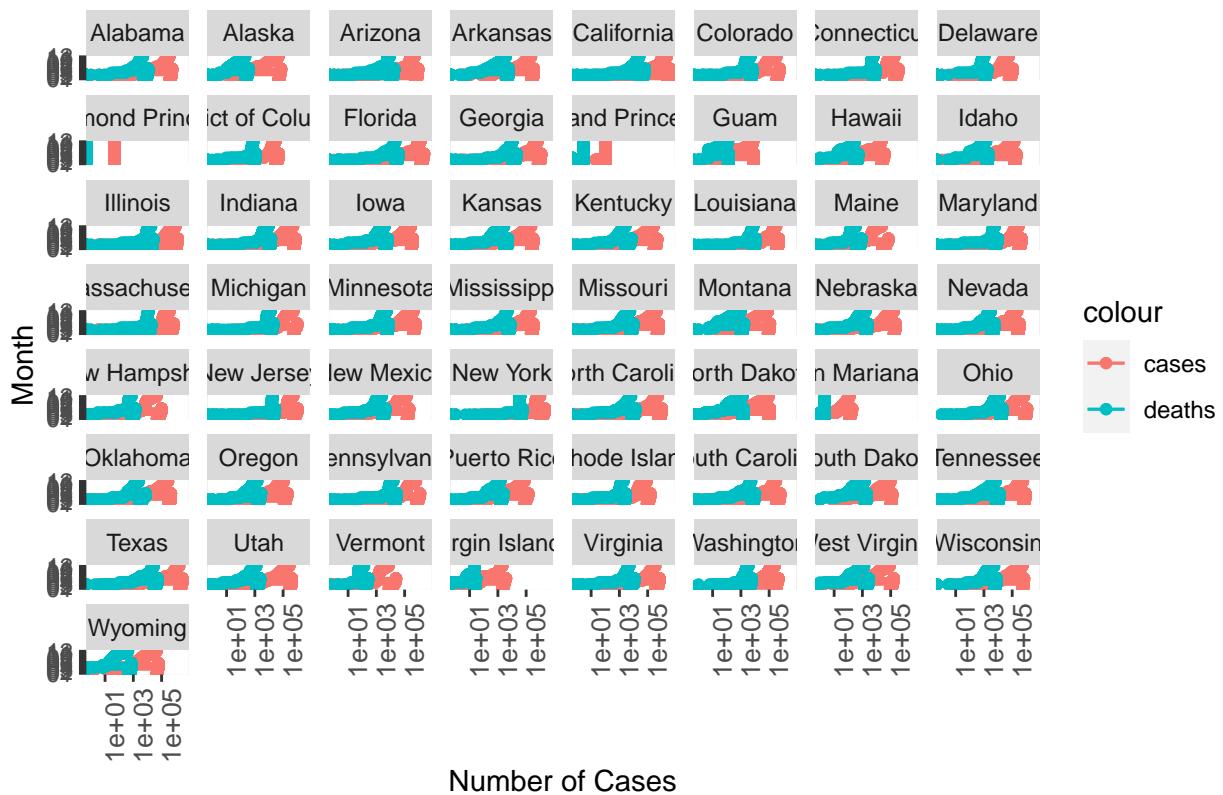


```
US_date %>%
  filter(cases > 0) %>%
  ggplot(aes(x = Month, y = cases)) +
  geom_line(aes(color = "cases")) +
  geom_point(aes(color = "cases")) +
  geom_line(aes(y = deaths, color = "deaths")) +
  geom_point(aes(y = deaths, color = "deaths")) +
  facet_wrap(~ Province_State) +
  coord_flip() +
  scale_y_log10() +
  theme(plot.title = element_text(hjust = 0.3),
  axis.text.x=element_text(angle=90, hjust=1, size=10),
  panel.spacing.x=unit(0.5, "lines")) +
  labs(x = "Month",
       y = "Number of Cases",
       title = "COVID19 by US states")
```

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

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

COVID19 by US states

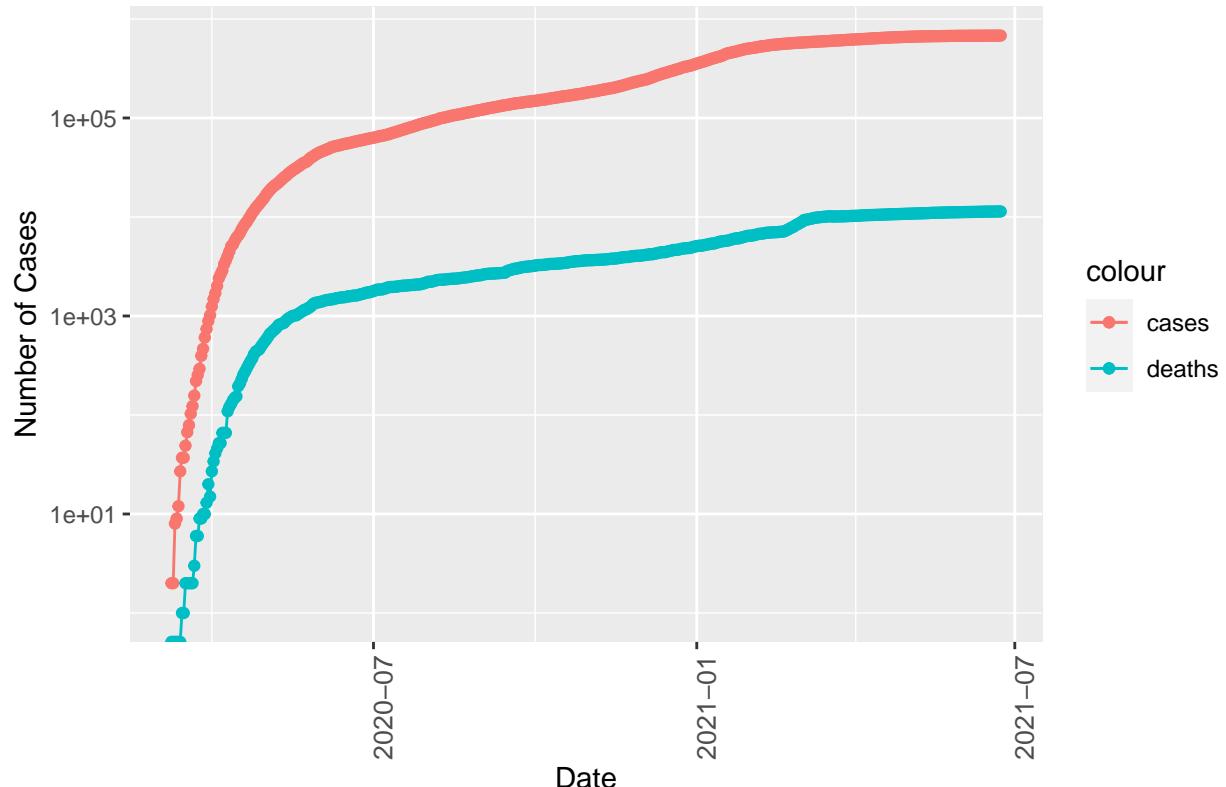


```
US_by_state %>%
  filter(Province_State == "Virginia") %>%
  filter(cases > 0) %>%
  ggplot(aes(x = date, y = cases)) +
  geom_line(aes(color = "cases")) +
  geom_point(aes(color = "cases")) +
  geom_line(aes(y = deaths, color = "deaths")) +
  geom_point(aes(y = deaths, color = "deaths")) +
  scale_y_log10() +
  theme(plot.title = element_text(hjust = 0.3),
  axis.text.x=element_text(angle=90, hjust=1, size=10),
  panel.spacing.x=unit(0.5, "lines")) +
  labs(title = "COVID19 cases/deaths in Virginia", x = "Date", y = "Number of Cases")
```

Warning: Transformation introduced infinite values in continuous y-axis

Warning: Transformation introduced infinite values in continuous y-axis

COVID19 cases/deaths in Virginia



```

US_by_state %>%
  filter(Province_State == "Virginia") %>%
  ggplot(aes(x = date, y = new_cases)) +
  geom_line(aes(color = "new_cases")) +
  geom_point(aes(color = "new_cases")) +
  geom_line(aes(y = deaths, color = "new_deaths")) +
  geom_point(aes(y = deaths, color = "new_deaths")) +
  scale_y_log10() +
  theme(legend.position="bottom",
        axis.text.x = element_text(angle = 90)) +
  labs(title = "New COVID19 cases/deaths in Virginia", x = "Date", y = "Number of Cases")

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

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

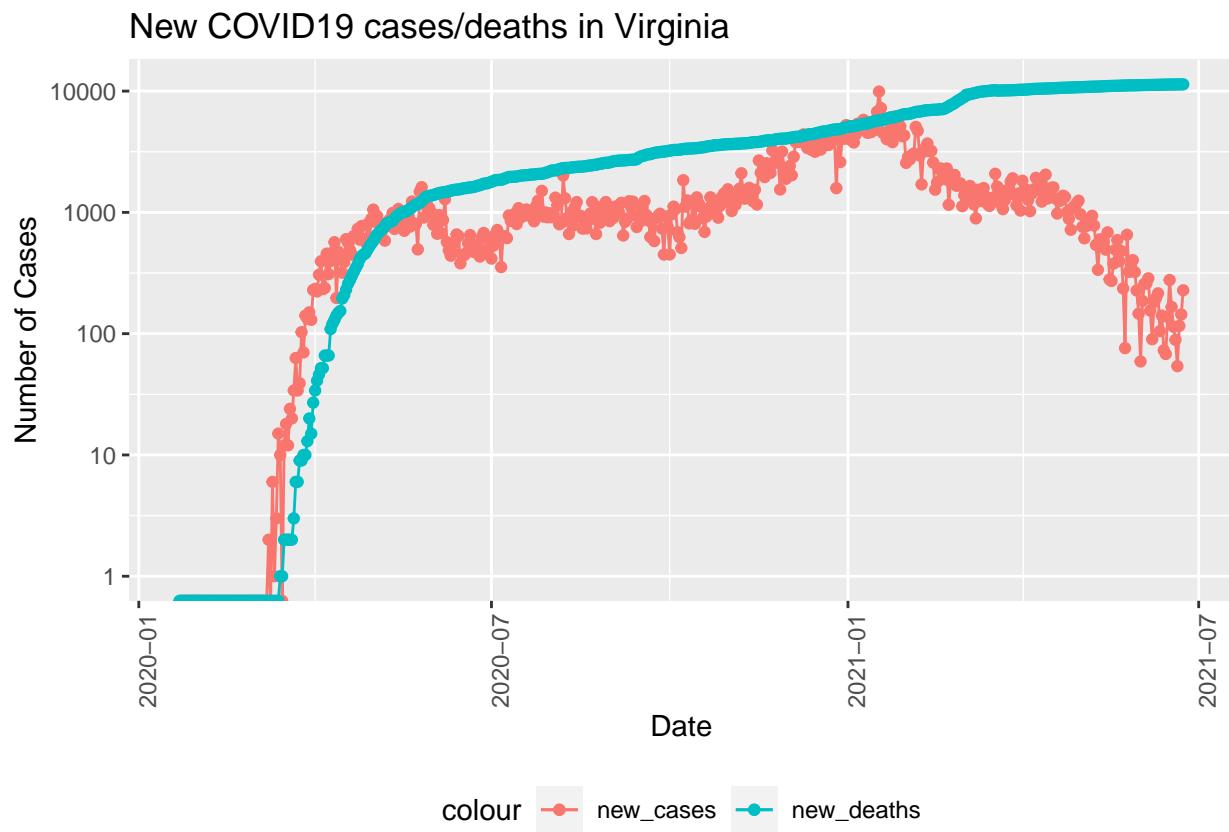
## Warning: Transformation introduced infinite values in continuous y-axis

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

```

```
## Warning: Removed 1 row(s) containing missing values (geom_path).
```

```
## Warning: Removed 1 rows containing missing values (geom_point).
```



Using a linear model to predict the cases per thousand as a function of deaths per thousand. The same model is applied for global dataset as well. This is a very simple model to fit linear models. We can see there are many outliers that are very far from the prediction line.

```
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)

global_totals <- global %>%
  group_by(Country_Region) %>%
  summarize(deaths = max(deaths), cases = max(cases),
            Population = max(Population),
            cases_per_thou = 1000* cases / Population,
            deaths_per_thou = 1000* deaths / Population) %>%
```

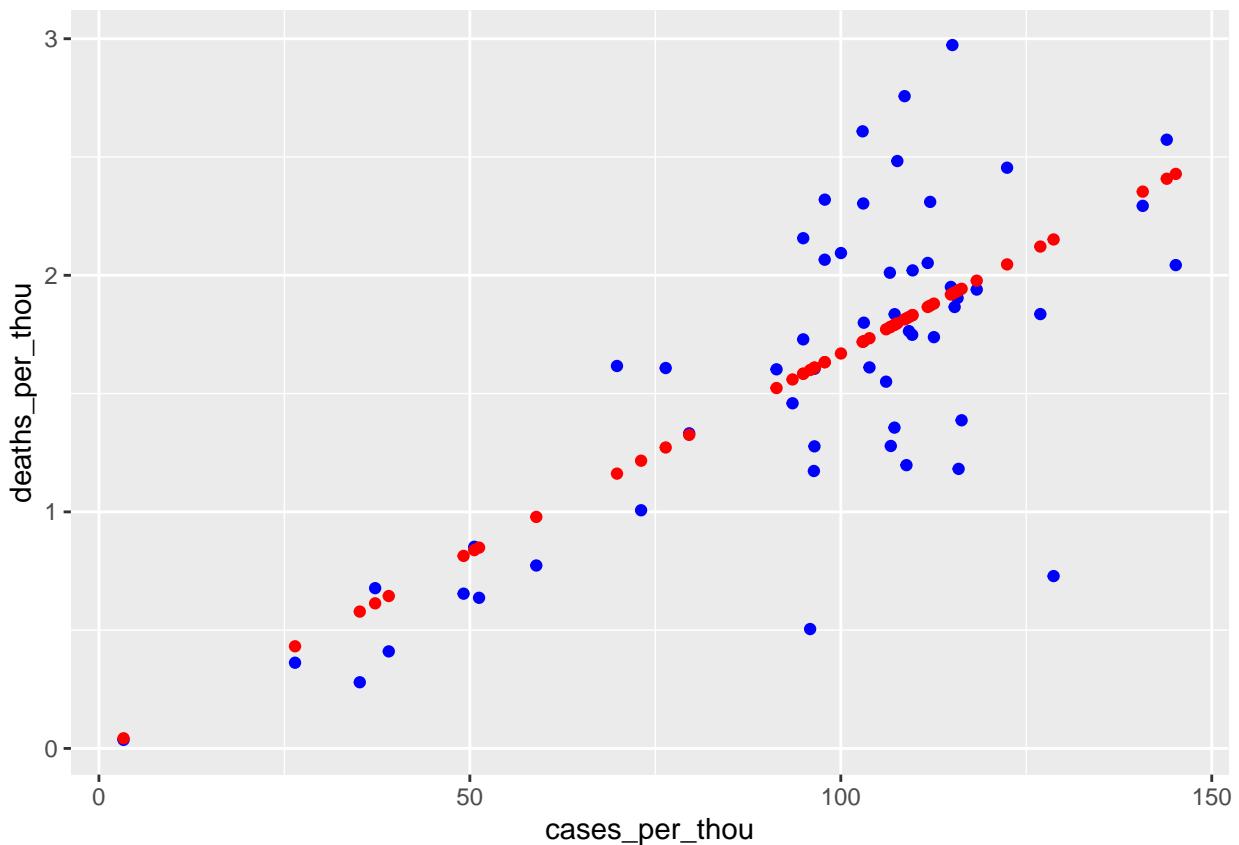
```

filter(cases > 0, Population > 0)

mod <- lm(deaths_per_thou ~ cases_per_thou, data = US_state_totals)
US_pred <- US_state_totals %>%
  mutate(pred = predict(mod))

US_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")

```

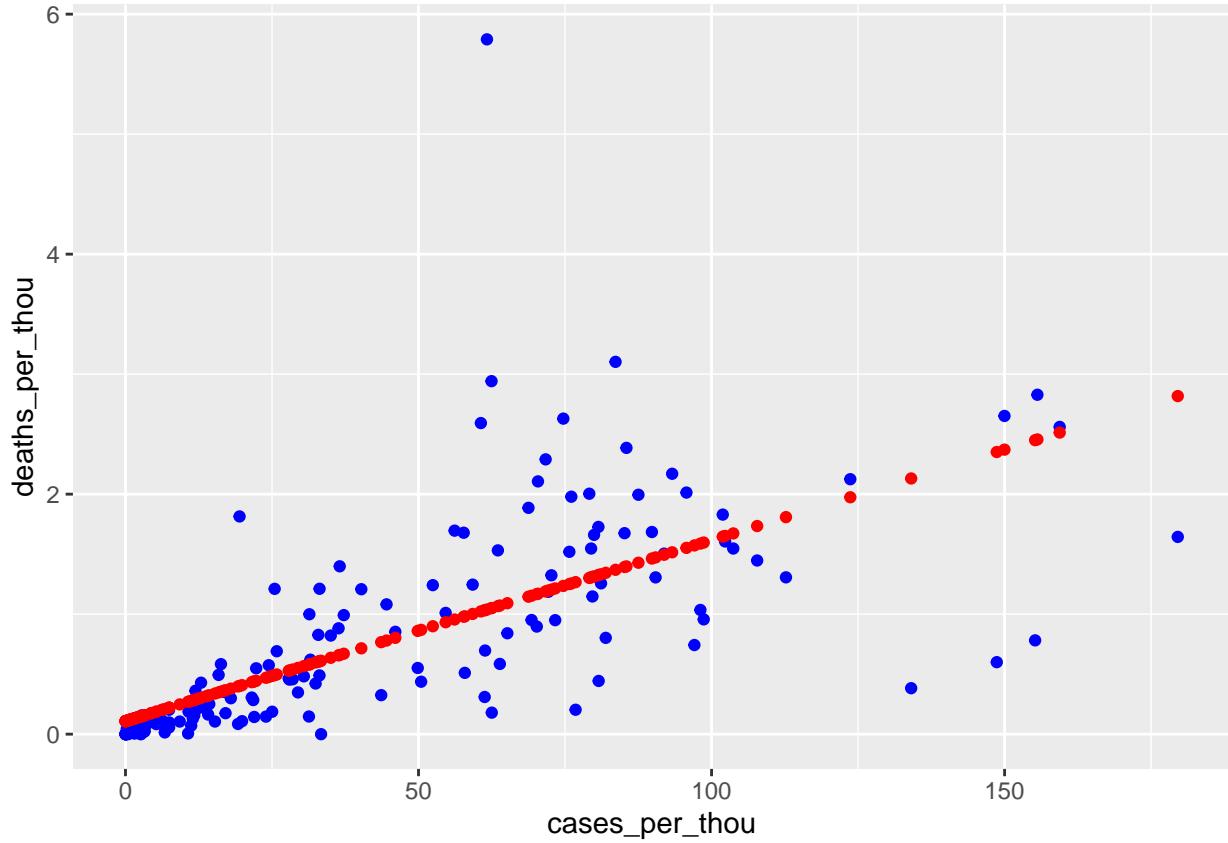


```

global_mod <- lm(deaths_per_thou ~ cases_per_thou, data = global_totals)
global_pred <- global_totals %>%
  mutate(pred = predict(global_mod))

global_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")

```



What are some of the general trends of the Covid19 pandemic, a large picture view?

We see that Covid19 recoveries are higher than the deaths. The same trend is observed in global and US datasets (visually the US dataset scales logarithmically in Y axes). We also see the same in the state of Virginia. The trend generally follows the same for US and some of its most populous states with new cases drops in US the states follow.

In conclusion, we downloaded the raw data from Systems Science and Engineering (CSSE) at Johns Hopkins University Github repository. The dataframe was then cleaned and tidy to do further data analysis. Plotting the data from dataframe, we see the total cases vs deaths and new cases and deaths for both US and global datasets. An attempt was made to do with `facet_wrap()` function to create sub-plots of US states, however it's not very readable since the plot size does not scale. Finally, linear model was used to predict the cases per thousand as a function of deaths per thousand. Since this is a linear model, it's will not be very useful for prediction for this kind of datasets. As we know the real world is not linear and virus behave in a different manner, there are lot of factors at play, from individual immunity to factors such as lockdown, public behavior and sensitivity to the pandemic, etc. The model does not account of any of this, it just does linear calculation based on the dataset. We can clearly see there are lot of outliers that are significant distance away from the prediction line.