# daf_covid_proj

Shashi

9/7/2021

```
library(stringr)
library(tidyverse)
```

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

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.4     v dplyr   1.0.7
## v tidyr   1.1.3     v forcats 0.5.1
## v readr   2.0.1
```

```
## -- 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
```

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

## We are importing the data from the csv collated by the John Hopkins University

We are directly importing the data from the github repo maintained by the university. The github repo is updated daily with new cases and deaths from the US and around the world. We then create four tibbles from the data obtained.

```
url_in <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covi
file_names <- c("confirmed_global.csv", "deaths_global.csv", "confirmed_US.csv", "deaths_US.csv")
urls <- str_c(url_in, file_names)

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)
head(global_cases)
```

```
## # A tibble: 6 x 613
##   `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>             Antigua and Barbuda  17.1 -61.8          0         0         0
## # ... with 606 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>,
## #   2/17/20 <dbl>, 2/18/20 <dbl>, 2/19/20 <dbl>, 2/20/20 <dbl>, 2/21/20 <dbl>,
## #   2/22/20 <dbl>, 2/23/20 <dbl>, 2/24/20 <dbl>, 2/25/20 <dbl>, ...
```

### Preprocessing and cleaning the data

```
global_cases <- global_cases %>%
    pivot_longer(cols = -c(`Province/State`,
                           `Country/Region`, Lat, Long),
                 names_to = "date",
                values_to = "cases") %>%
select(-c(Lat,Long))
head(global_cases)
```

```
## # A tibble: 6 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
```

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

```
## # A tibble: 6 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
```

```
global <- global_cases %>%
  full_join(global_deaths) %>%
  rename("Country_Region" = `Country/Region`,
         "Province_State" = `Province/State`) %>%
```

```
  mutate(date = mdy(date))
```

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

```
global
```

```
## # A tibble: 169,911 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
## # ... with 169,901 more rows
```

```
summary(global)
```

```
##  Province_State     Country_Region          date                cases
##  Length:169911      Length:169911      Min.   :2020-01-22   Min.   :       0
##  Class :character   Class :character   1st Qu.:2020-06-22   1st Qu.:     146
##  Mode  :character   Mode  :character   Median :2020-11-21   Median :    2318
##                                        Mean   :2020-11-21   Mean   :  288108
##                                        3rd Qu.:2021-04-22   3rd Qu.:   52404
##                                        Max.   :2021-09-21   Max.   :42410607
##      deaths
##  Min.   :     0.0
##  1st Qu.:     1.0
##  Median :    35.0
##  Mean   :  6637.6
##  3rd Qu.:   851.5
##  Max.   :678407.0
```

```
global <- global %>%
  filter(cases > 0)
```

```
global
```

```
## # A tibble: 153,895 x 5
##    Province_State Country_Region date       cases deaths
##    <chr>          <chr>          <date>     <dbl>  <dbl>
##  1 <NA>           Afghanistan    2020-02-24     5      0
##  2 <NA>           Afghanistan    2020-02-25     5      0
##  3 <NA>           Afghanistan    2020-02-26     5      0
##  4 <NA>           Afghanistan    2020-02-27     5      0
##  5 <NA>           Afghanistan    2020-02-28     5      0
##  6 <NA>           Afghanistan    2020-02-29     5      0
##  7 <NA>           Afghanistan    2020-03-01     5      0
##  8 <NA>           Afghanistan    2020-03-02     5      0
##  9 <NA>           Afghanistan    2020-03-03     5      0
## 10 <NA>           Afghanistan    2020-03-04     5      0
```

```
## # ... with 153,885 more rows
```

```
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_cases
```

```
## # A tibble: 2,035,278 x 6
##    Admin2  Province_State Country_Region Combined_Key         date       cases
##    <chr>   <chr>          <chr>          <chr>                <date>     <dbl>
##  1 Autauga Alabama        US             Autauga, Alabama, US 2020-01-22     0
##  2 Autauga Alabama        US             Autauga, Alabama, US 2020-01-23     0
##  3 Autauga Alabama        US             Autauga, Alabama, US 2020-01-24     0
##  4 Autauga Alabama        US             Autauga, Alabama, US 2020-01-25     0
##  5 Autauga Alabama        US             Autauga, Alabama, US 2020-01-26     0
##  6 Autauga Alabama        US             Autauga, Alabama, US 2020-01-27     0
##  7 Autauga Alabama        US             Autauga, Alabama, US 2020-01-28     0
##  8 Autauga Alabama        US             Autauga, Alabama, US 2020-01-29     0
##  9 Autauga Alabama        US             Autauga, Alabama, US 2020-01-30     0
## 10 Autauga Alabama        US             Autauga, Alabama, US 2020-01-31     0
## # ... with 2,035,268 more rows
```

```
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
```

```
## # A tibble: 2,035,278 x 8
##    Admin2  Province_State Country_Region Combined_Key date       cases Population
##    <chr>   <chr>          <chr>          <chr>        <date>     <dbl>      <dbl>
##  1 Autauga Alabama        US             Autauga, Al~ 2020-01-22     0      55869
##  2 Autauga Alabama        US             Autauga, Al~ 2020-01-23     0      55869
##  3 Autauga Alabama        US             Autauga, Al~ 2020-01-24     0      55869
##  4 Autauga Alabama        US             Autauga, Al~ 2020-01-25     0      55869
##  5 Autauga Alabama        US             Autauga, Al~ 2020-01-26     0      55869
##  6 Autauga Alabama        US             Autauga, Al~ 2020-01-27     0      55869
##  7 Autauga Alabama        US             Autauga, Al~ 2020-01-28     0      55869
##  8 Autauga Alabama        US             Autauga, Al~ 2020-01-29     0      55869
##  9 Autauga Alabama        US             Autauga, Al~ 2020-01-30     0      55869
## 10 Autauga Alabama        US             Autauga, Al~ 2020-01-31     0      55869
## # ... with 2,035,268 more rows, and 1 more variable: deaths <dbl>
```

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

```
global
```

```
## # A tibble: 153,895 x 6
##    combined_key Province_State Country_Region date       cases deaths
##    <chr>        <chr>          <chr>          <date>     <dbl>  <dbl>
##  1 Afghanistan  <NA>           Afghanistan    2020-02-24     5      0
##  2 Afghanistan  <NA>           Afghanistan    2020-02-25     5      0
##  3 Afghanistan  <NA>           Afghanistan    2020-02-26     5      0
##  4 Afghanistan  <NA>           Afghanistan    2020-02-27     5      0
##  5 Afghanistan  <NA>           Afghanistan    2020-02-28     5      0
##  6 Afghanistan  <NA>           Afghanistan    2020-02-29     5      0
##  7 Afghanistan  <NA>           Afghanistan    2020-03-01     5      0
##  8 Afghanistan  <NA>           Afghanistan    2020-03-02     5      0
##  9 Afghanistan  <NA>           Afghanistan    2020-03-03     5      0
## 10 Afghanistan  <NA>           Afghanistan    2020-03-04     5      0
## # ... with 153,885 more rows
```

In order to calculate the number of cases and deaths in relation to the population of that particular country, we are importing another dataset that contains the population data and joining it with our main dataset.

```
uid_url <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/UID_ISO_
uid <- read_csv(uid_url) %>%
  select(-c(Lat, Long_, Combined_Key, code3, iso2, iso3, Admin2))
```

```
## Rows: 4196 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.
```

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

```
global
```

```
## # A tibble: 153,895 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
## # ... with 153,885 more rows
```

Calculating the cases per million and deaths per million in order to do further analysis and prediction on that data.

```
country_cases_deaths_per_mil <- global %>%
  group_by(Country_Region, Province_State)%>%
  summarise(country_cases = max(cases), country_deaths = max(deaths), population = max(Population))%>%
  group_by(Country_Region)%>%
  summarise(country_cases = sum(country_cases), country_deaths = sum(country_deaths), population = sum(
  mutate(cases_per_million = country_cases*1000000/population,deaths_per_million = country_deaths*10000
```

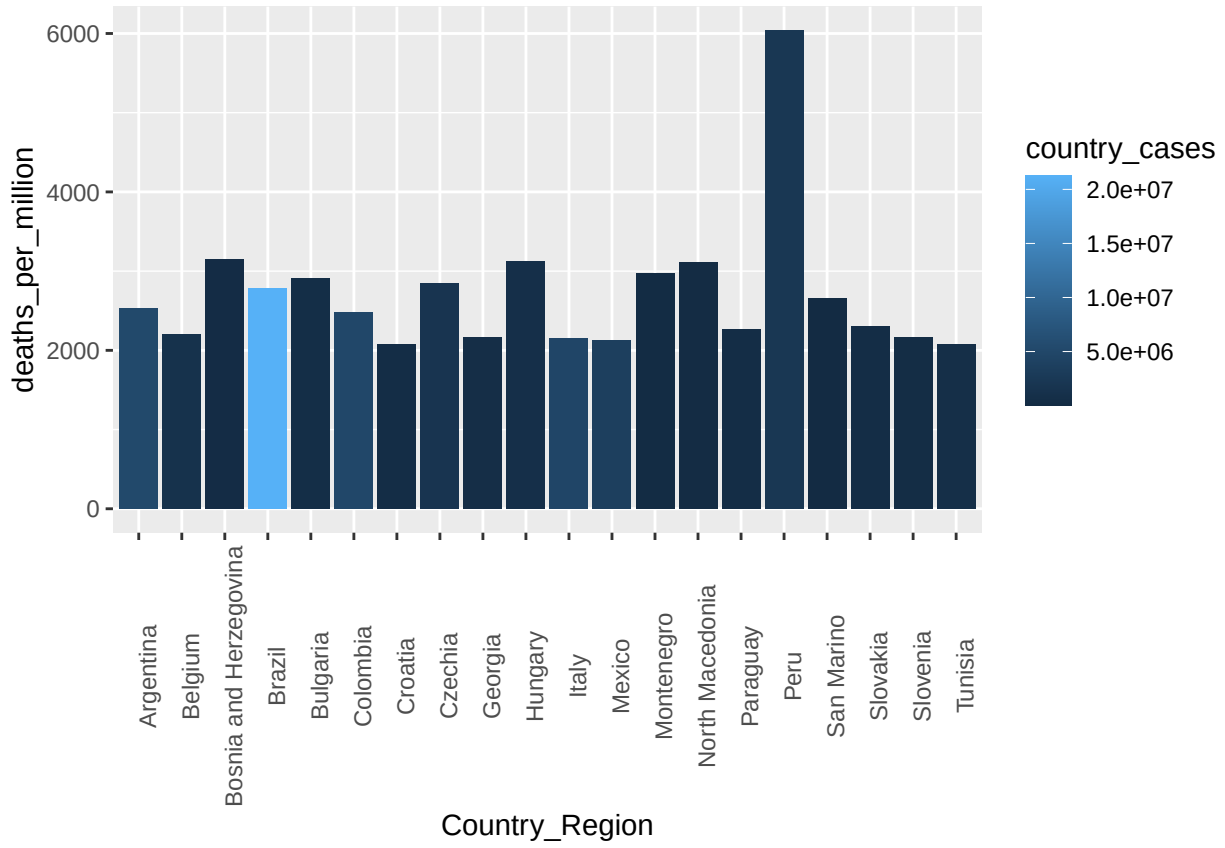
```
## `summarise()` has grouped output by 'Country_Region'. You can override using the `.groups` argument.
```

```
country_cases_deaths_per_mil
```

```
## # A tibble: 195 x 6
##    Country_Region      country_cases country_deaths population cases_per_million
##    <chr>                       <dbl>          <dbl>      <dbl>             <dbl>
##  1 Afghanistan                154712           7199   38928341             3974.
##  2 Albania                    164276           2594    2877800            57084.
##  3 Algeria                    201948           5725   43851043             4605.
##  4 Andorra                     15140            130      77265           195949.
##  5 Angola                      52968           1414   32866268             1612.
##  6 Antigua and Barbuda          2603             55      97928            26581.
##  7 Argentina                 5243231         114579   45195777           116012.
##  8 Armenia                    254436           5161    2963234            85864.
##  9 Australia                   90369           1187   25459700             3549.
## 10 Austria                    726674          10918    9006400            80684.
## # ... with 185 more rows, and 1 more variable: deaths_per_million <dbl>
```

Here we are calculating the top countries in the world by deaths per million, the bars are shaded according to the number of cases, this helps us in giving an understanding of how many absolute cases are there in that particular country.

```
plot_by_country <- country_cases_deaths_per_mil %>%
  slice_max(deaths_per_million, n=20) %>%
  ggplot(aes(x = Country_Region, y = deaths_per_million, fill = country_cases))+ geom_bar(stat = "ident
  theme(axis.text.x = element_text(angle = 90))

plot_by_country
```



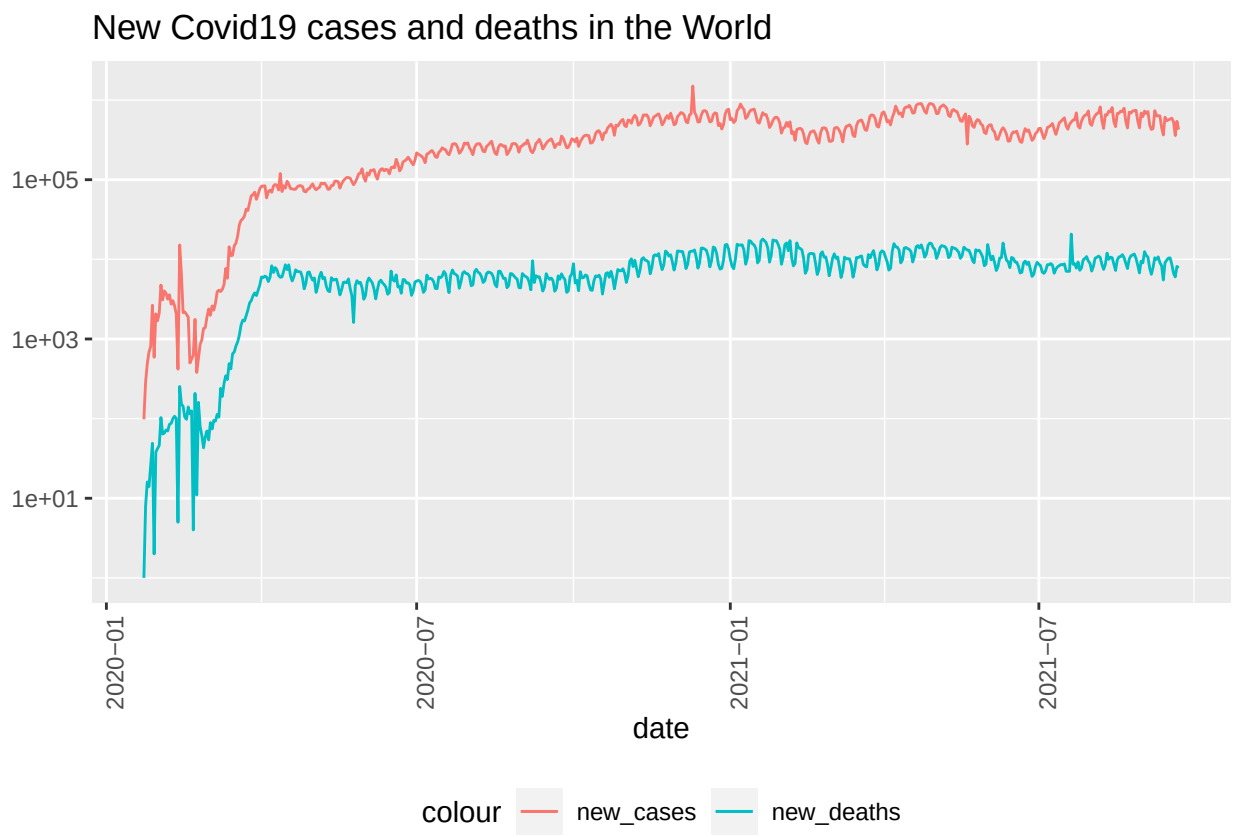

We are calculating the daily new cases and deaths in the world as a whole. We further plot this data to give us a good understanding of how this pandemic has evolved since its inception.

```
world_new_cases_deaths <- global%>%
  group_by(date)%>%
  summarise(cases = sum(cases), deaths = sum(deaths)) %>%
  mutate(new_cases = cases - lag(cases), new_deaths = deaths-lag(deaths))%>%
  ungroup()%>%
  filter(new_deaths > 0)
world_new_cases_deaths
```

```
## # A tibble: 608 x 5
##    date       cases deaths new_cases new_deaths
##    <date>     <dbl>  <dbl>     <dbl>      <dbl>
##  1 2020-01-23   655     18        98          1
##  2 2020-01-24   941     26       286          8
##  3 2020-01-25  1433     42       492         16
##  4 2020-01-26  2118     56       685         14
##  5 2020-01-27  2927     82       809         26
##  6 2020-01-28  5578    131      2651         49
##  7 2020-01-29  6167    133       589          2
##  8 2020-01-30  8235    171      2068         38
##  9 2020-01-31  9927    213      1692         42
## 10 2020-02-01 12038    259      2111         46
## # ... with 598 more rows
```

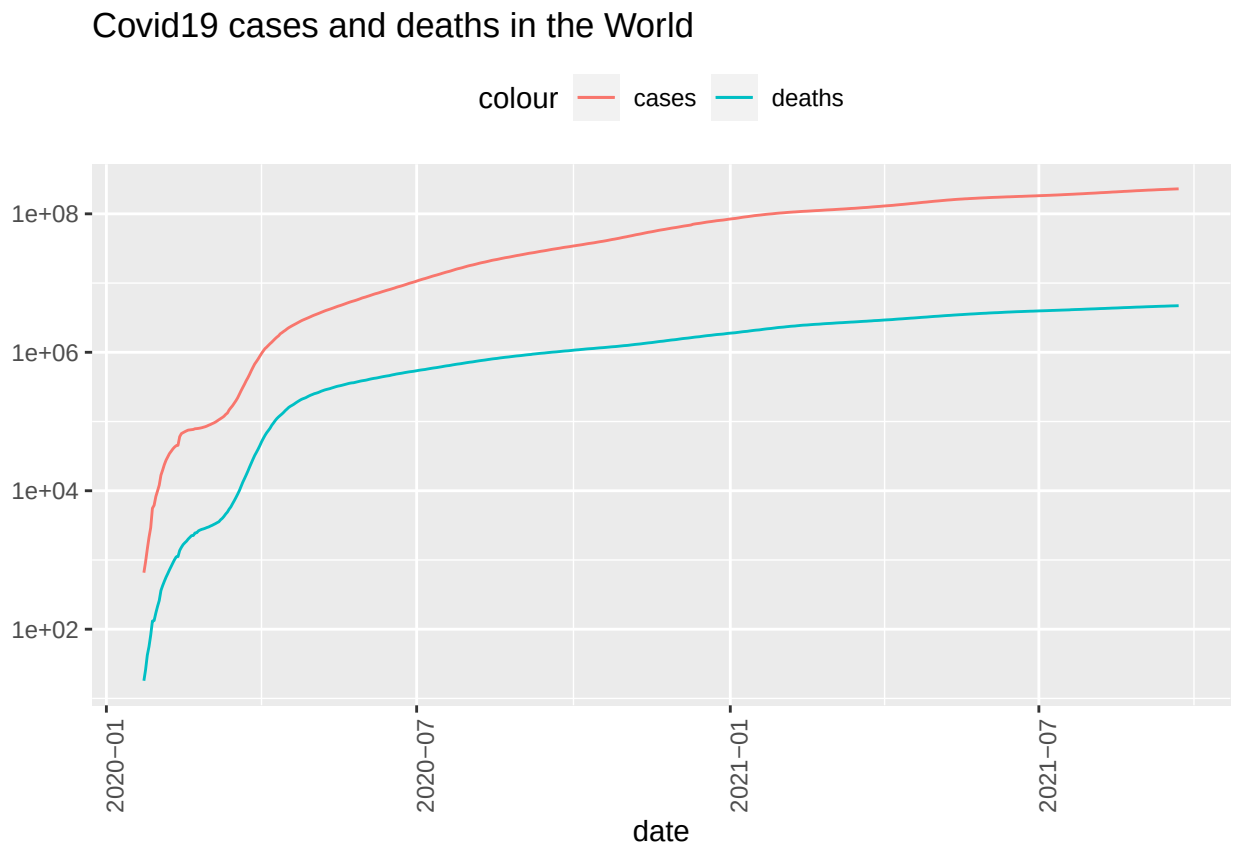
Plotting daily new cases and deaths in the world

```
plot_new_cases_deaths <- world_new_cases_deaths %>%
  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() + theme(legend.position = "bottom",
                          axis.text.x = element_text(angle = 90)) +
  labs(title = "New Covid19 cases and deaths in the World", y=NULL)
plot_new_cases_deaths
```




Plotting the cumulative cases and deaths in the world.

```
plot_world_cases_deaths <- world_new_cases_deaths %>%
  ggplot(aes(x=date,y=cases)) +
  geom_line(aes(color = "cases")) +
  geom_line(aes(y = deaths,color = "deaths")) +
  scale_y_log10() + theme(legend.position = "top",
                          axis.text.x = element_text(angle = 90)) +
  labs(title = "Covid19 cases and deaths in the World", y=NULL)
plot_world_cases_deaths
```




Building a Linear Regression model to predict new deaths every day by taking into consideration the new cases everyday.

```
world_model <- lm(new_deaths~new_cases, data = world_new_cases_deaths)
summary(world_model)
```

```
##
## Call:
## lm(formula = new_deaths ~ new_cases, data = world_new_cases_deaths)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10863.8  -1590.4   -191.9   1042.9  10853.7
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.439e+03  1.596e+02   15.28   <2e-16 ***
## new_cases   1.405e-02  3.540e-04   39.70   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2150 on 606 degrees of freedom
## Multiple R-squared:  0.7223, Adjusted R-squared:  0.7218
## F-statistic:  1576 on 1 and 606 DF,  p-value: < 2.2e-16
```

Plotting our model and predicting the increase in new deaths everyday with the increase in new cases everyday.

```
world_total_w_pred <- world_new_cases_deaths %>%
  mutate(pred = predict(world_model))
world_total_w_pred %>% ggplot() +
  geom_point(aes(x = new_cases, y = new_deaths), color = "blue")+
  geom_line(aes(x=new_cases, y = pred), color = "red")
```


```
us
```

```
## # A tibble: 2,035,278 x 8
##    Admin2  Province_State Country_Region Combined_Key date       cases Population
##    <chr>   <chr>          <chr>          <chr>        <date>     <dbl>      <dbl>
##  1 Autauga Alabama        US             Autauga, Al~ 2020-01-22     0      55869
##  2 Autauga Alabama        US             Autauga, Al~ 2020-01-23     0      55869
##  3 Autauga Alabama        US             Autauga, Al~ 2020-01-24     0      55869
##  4 Autauga Alabama        US             Autauga, Al~ 2020-01-25     0      55869
##  5 Autauga Alabama        US             Autauga, Al~ 2020-01-26     0      55869
##  6 Autauga Alabama        US             Autauga, Al~ 2020-01-27     0      55869
##  7 Autauga Alabama        US             Autauga, Al~ 2020-01-28     0      55869
##  8 Autauga Alabama        US             Autauga, Al~ 2020-01-29     0      55869
##  9 Autauga Alabama        US             Autauga, Al~ 2020-01-30     0      55869
## 10 Autauga Alabama        US             Autauga, Al~ 2020-01-31     0      55869
## # ... with 2,035,268 more rows, and 1 more variable: deaths <dbl>
```

Calculating the deaths per million in each county for each day.

```
us_by_state <- us %>%
  group_by(Province_State, Country_Region,date)%>%
```

```
  summarise(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()
```

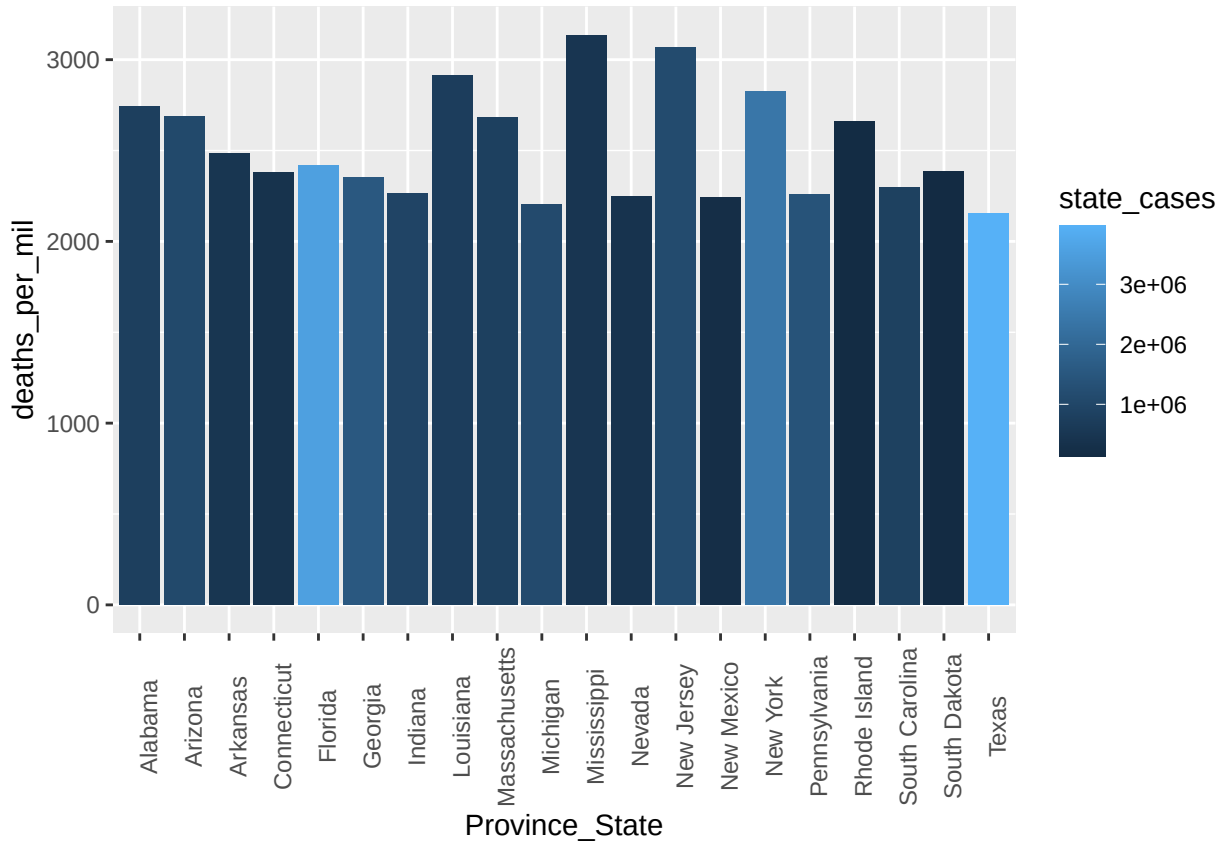
```
## `summarise()` has grouped output by 'Province_State', 'Country_Region'. You can override using the `
```

```
us_by_state
```

```
## # A tibble: 35,322 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
## # ... with 35,312 more rows, and 1 more variable: Population <dbl>
```

Plotting the top 20 states in terms of highest deaths per million, the bars are shaded according to the number of cases, this helps us in giving an understanding of how many absolute cases are there in that particular state.

```
plot_by_state <- us_by_state %>%
  group_by(Province_State) %>%
  summarise(state_cases = max(cases), state_deaths  = max(deaths), deaths_per_mil = max(deaths_per_mil)
  slice_max(deaths_per_mil, n=20) %>%
  ggplot(aes(x = Province_State, y = deaths_per_mil, fill = state_cases))+ geom_bar(stat = "identity")+
  theme(axis.text.x = element_text(angle = 90))
plot_by_state
```




```
us_total <- us_by_state %>%
  group_by(Country_Region, date)%>%
  summarise(Population = sum(Population), cases = sum(cases), deaths = sum(deaths))%>%
  mutate(deaths_per_mil = deaths*1000000/Population)%>%
  select(Country_Region, date, cases, deaths, Population, deaths_per_mil)%>%
  ungroup()
```

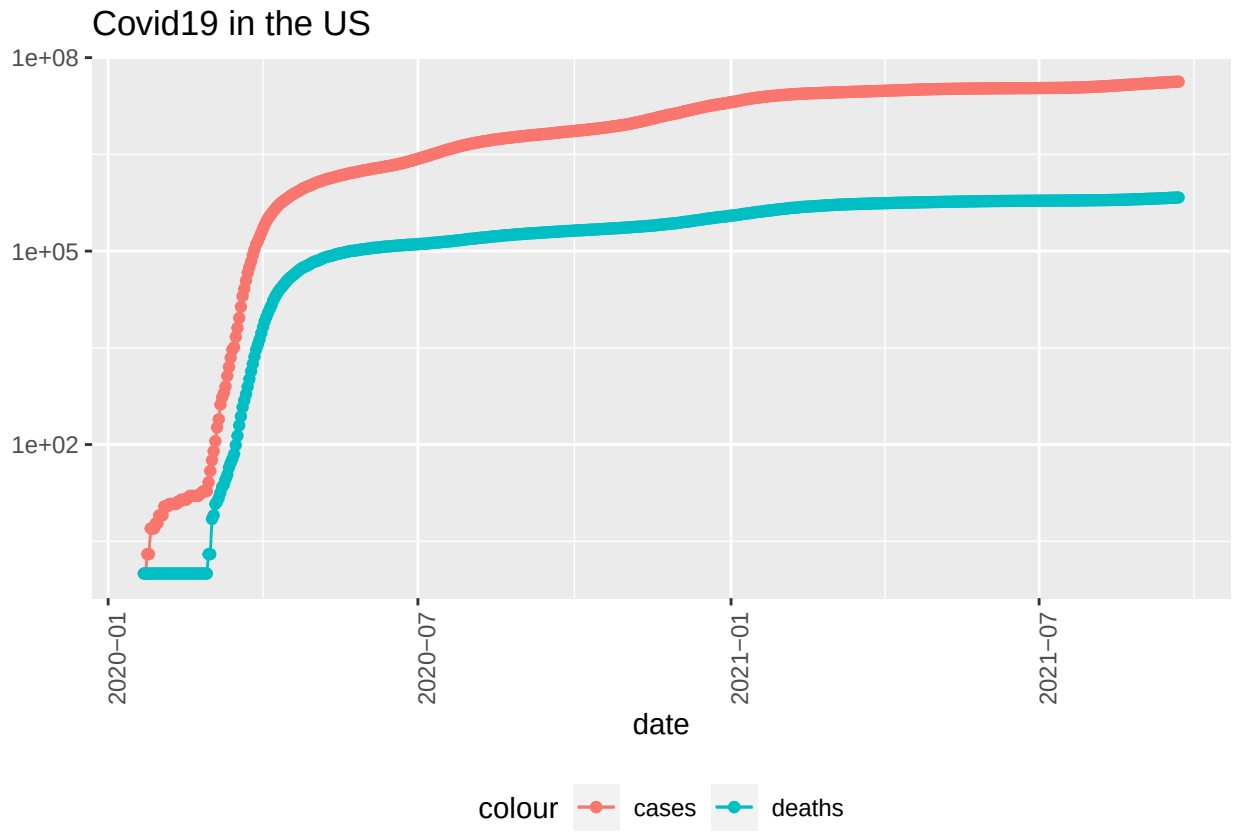
```
## `summarise()` has grouped output by 'Country_Region'. You can override using the `.groups` argument.
```

```
us_total
```

```
## # A tibble: 609 x 6
##    Country_Region date       cases deaths Population deaths_per_mil
##    <chr>          <date>     <dbl>  <dbl>      <dbl>          <dbl>
##  1 US             2020-01-22     1      1  332875137        0.00300
##  2 US             2020-01-23     1      1  332875137        0.00300
##  3 US             2020-01-24     2      1  332875137        0.00300
##  4 US             2020-01-25     2      1  332875137        0.00300
##  5 US             2020-01-26     5      1  332875137        0.00300
##  6 US             2020-01-27     5      1  332875137        0.00300
##  7 US             2020-01-28     5      1  332875137        0.00300
##  8 US             2020-01-29     6      1  332875137        0.00300
##  9 US             2020-01-30     6      1  332875137        0.00300
## 10 US             2020-01-31     8      1  332875137        0.00300
## # ... with 599 more rows
```

Plotting the evolution of cases and deaths in the US due to the Covid19 pandemic.

```
us_cases_deaths_plot<- us_total %>%
  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 the US", y=NULL)
us_cases_deaths_plot
```



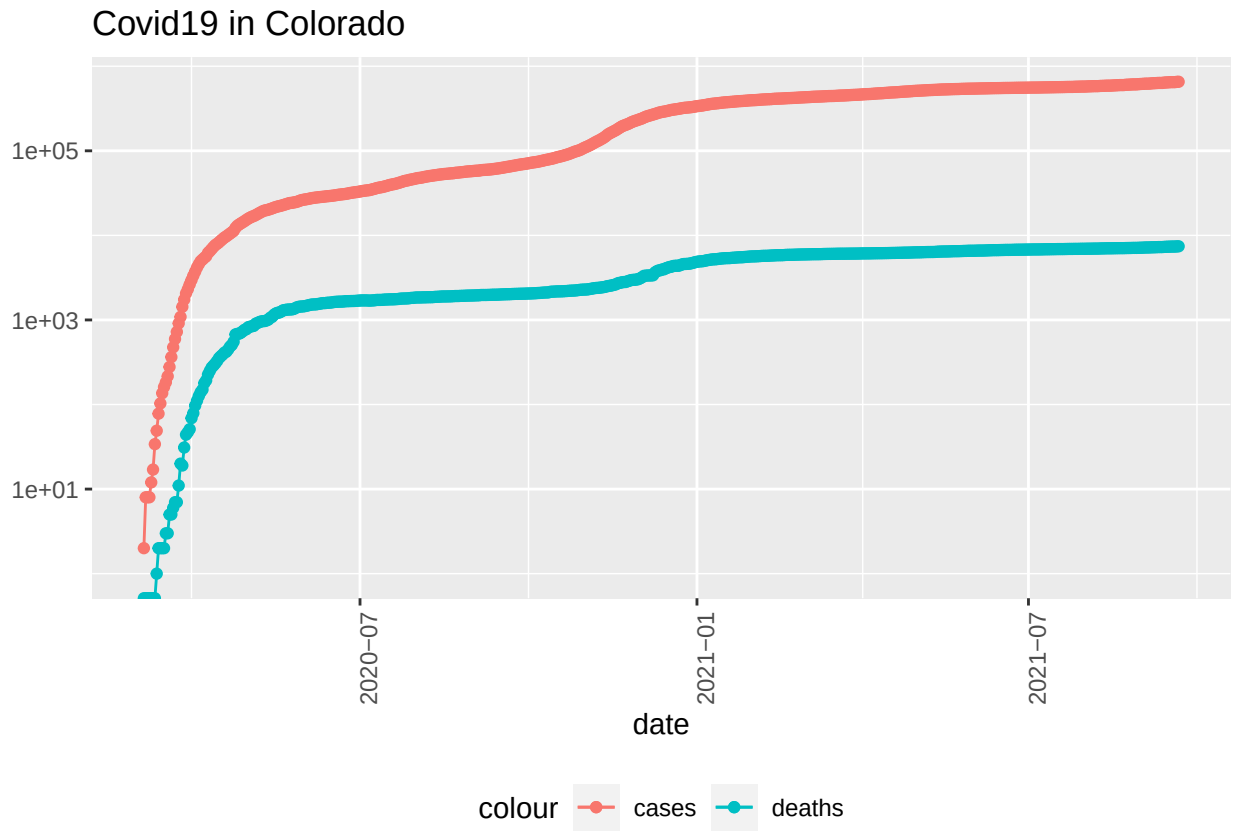

Plotting the evolution of cases and deaths in the state of Colorado due to the Covid19 pandemic.

```
state <- "Colorado"
state_cases_deaths_plot <- 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)
```

```
state_cases_deaths_plot
```

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

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




Calculating new cases and deaths in the United States everyday.

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

```
us_by_state
```

```
## # A tibble: 35,322 x 9
##    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
## # ... with 35,312 more rows, and 3 more variables: Population <dbl>,
## #   new_cases <dbl>, new_deaths <dbl>
```

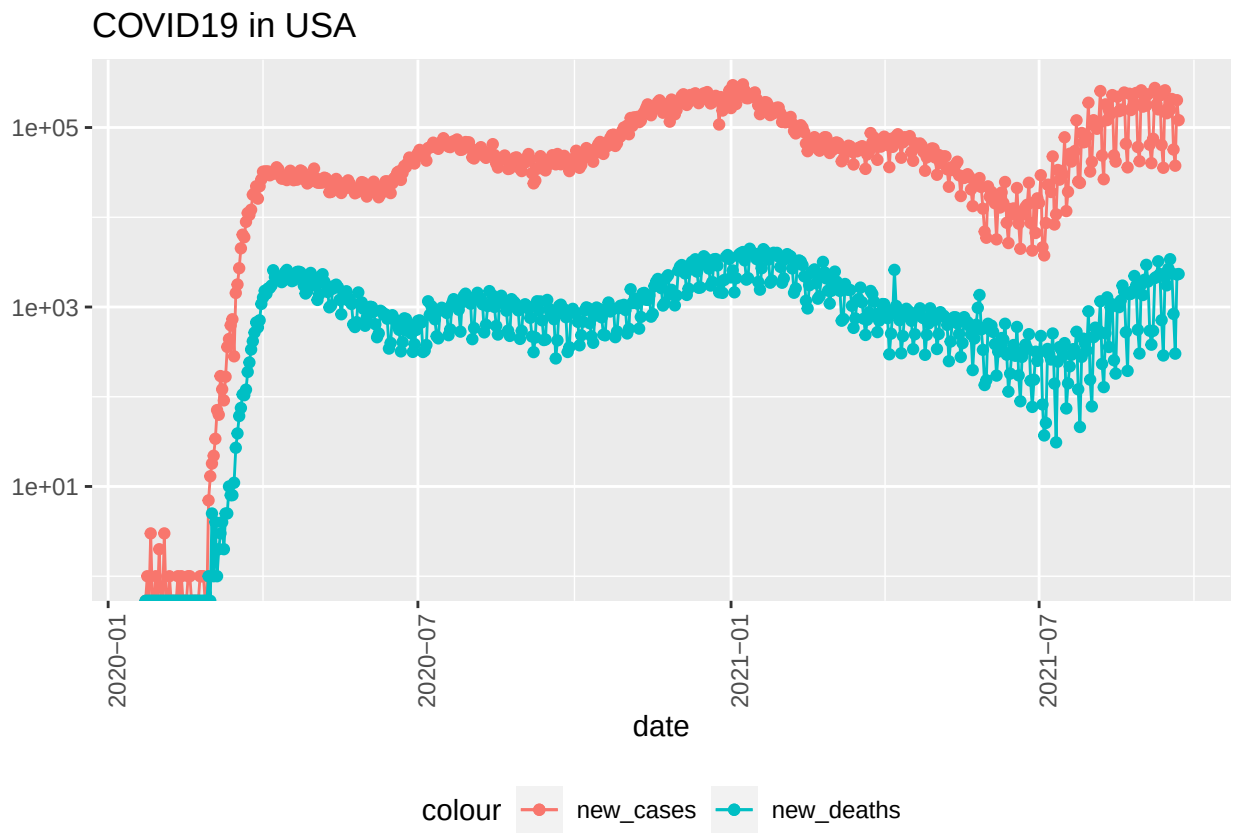
```
us_total
```

```
## # A tibble: 609 x 8
##    Country_Region date       cases deaths Population deaths_per_mil new_cases
##    <chr>          <date>     <dbl>  <dbl>      <dbl>          <dbl>     <dbl>
##  1 US             2020-01-22     1      1  332875137        0.00300        NA
##  2 US             2020-01-23     1      1  332875137        0.00300         0
##  3 US             2020-01-24     2      1  332875137        0.00300         1
##  4 US             2020-01-25     2      1  332875137        0.00300         0
##  5 US             2020-01-26     5      1  332875137        0.00300         3
##  6 US             2020-01-27     5      1  332875137        0.00300         0
##  7 US             2020-01-28     5      1  332875137        0.00300         0
##  8 US             2020-01-29     6      1  332875137        0.00300         1
##  9 US             2020-01-30     6      1  332875137        0.00300         0
## 10 US             2020-01-31     8      1  332875137        0.00300         2
## # ... with 599 more rows, and 1 more variable: new_deaths <dbl>
```

Plotting new cases and deaths in the US everyday

```
us_total %>%
  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 USA",y=NULL)
```

```
## 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: 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).

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

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

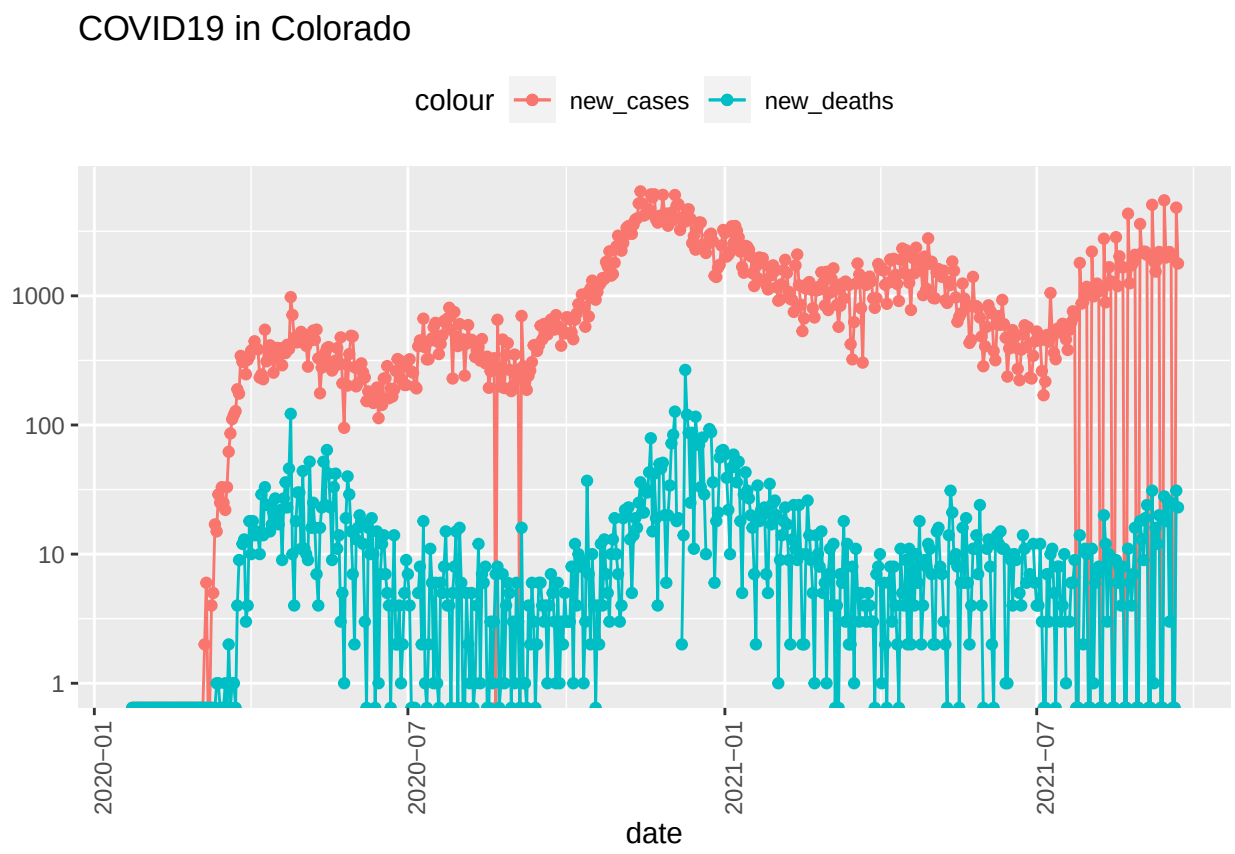



Plotting new cases and deaths in the state of Colorado everyday.

```
state <- "Colorado"
us_by_state %>%
  filter(Province_State == state) %>%
  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 = "top",
        axis.text.x = element_text(angle=90))+
  labs(title = str_c("COVID19 in ",state),y=NULL)
```

```
## 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 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(s) containing missing values (geom_path).
## Warning: Removed 1 rows containing missing values (geom_point).
## Warning: Removed 1 row(s) containing missing values (geom_path).
## Warning: Removed 12 rows containing missing values (geom_point).
```




Finding the states with the lowest deaths per thousand.

```
us_state_totals <- us_by_state %>%
  group_by(Province_State)%>%
  summarise(deaths = max(deaths), cases = max(cases), population = max(Population),
            cases_per_thousand = 1000*cases/population,
            deaths_per_thousand = 1000*deaths/population)%>%
  filter(cases>0, population>0)
us_state_totals %>%
  slice_min(deaths_per_thousand, n=10)
```

```
## # A tibble: 10 x 6
##    Province_State           deaths  cases population cases_per_thous~ deaths_per_thou~
##    <chr>                     <dbl>  <dbl>      <dbl>            <dbl>            <dbl>
##  1 Northern Mariana Islands      2    265      55144             4.81           0.0363
##  2 Vermont                     301  31911     623989             51.1            0.482
##  3 Hawaii                      714  76191    1415872             53.8            0.504
##  4 Virgin Islands               68   6516     107268             60.7            0.634
##  5 Alaska                      480 103327     740995            139.             0.648
##  6 Maine                      1002  84542    1344212             62.9            0.745
##  7 Puerto Rico                3092 179523    3754939             47.8            0.823
```

```
##  8 Oregon                     3624 314841    4217737            74.6            0.859
##  9 Utah                       2829 495704    3205958           155.             0.882
## 10 Washington                 7315 631023    7614893            82.9            0.961
```

Finding the states with the highest deaths per thousand.

```
us_state_totals %>%
  slice_max(deaths_per_thousand, n=10)
```

```
## # A tibble: 10 x 6
##    Province_State deaths   cases population cases_per_thousand deaths_per_thous~
##    <chr>           <dbl>   <dbl>      <dbl>              <dbl>              <dbl>
##  1 Mississippi      9331  477769    2976149               161.               3.14
##  2 New Jersey      27240 1137016    8882190               128.               3.07
##  3 Louisiana       13558  730099    4648794               157.               2.92
##  4 New York        54983 2382450   19453561               122.               2.83
##  5 Alabama         13460  775531    4903185               158.               2.75
##  6 Arizona         19584 1070757    7278717               147.               2.69
##  7 Massachusetts   18480  796925    6892503               116.               2.68
##  8 Rhode Island     2816  169686    1059361               160.               2.66
##  9 Arkansas         7499  486853    3017804               161.               2.48
## 10 Florida         51889 3528698   21477737               164.               2.42
```

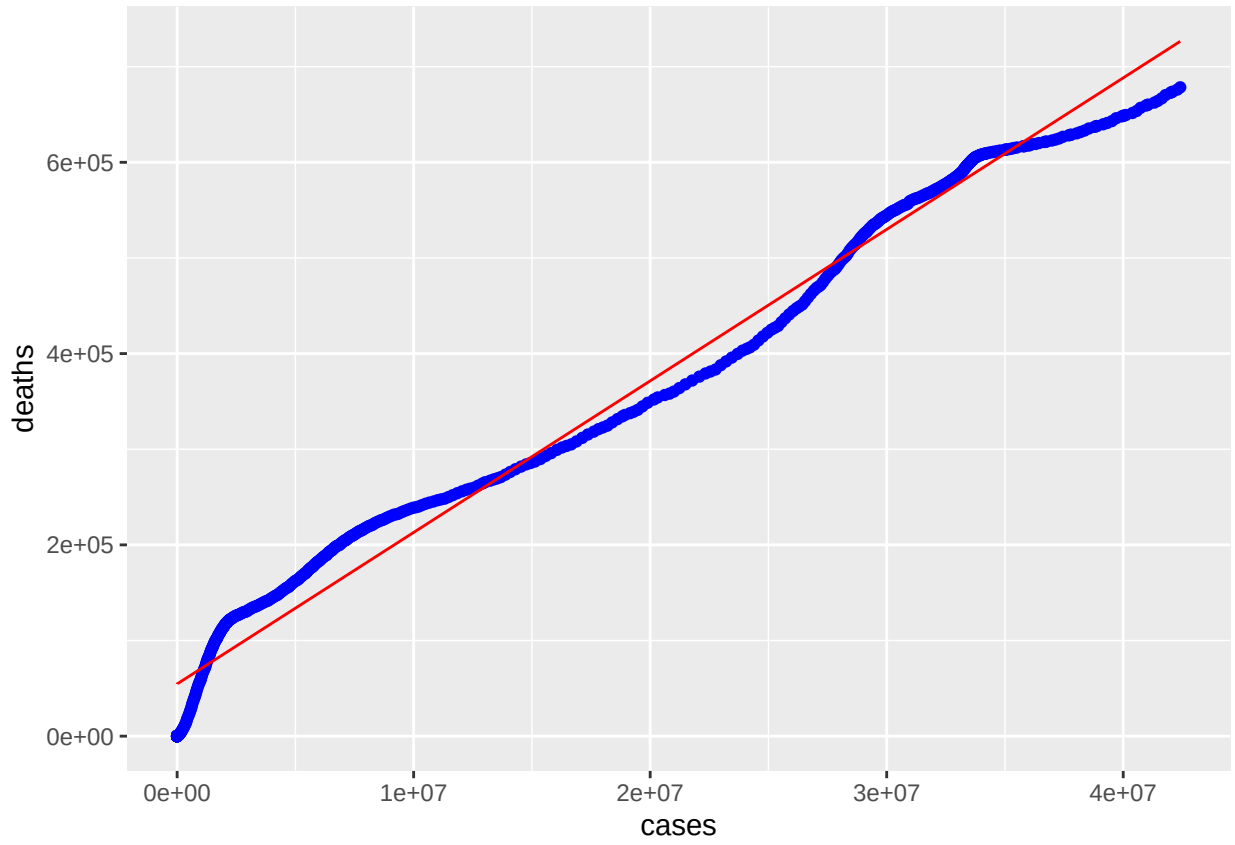
Creating a Linear Model to predict how high can deaths and cases can reach in the US depending on the rise of cases.

```
model = lm(deaths ~ cases, data = us_total)
summary(model)
```

```
##
## Call:
## lm(formula = deaths ~ cases, data = us_total)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -54582 -22063   9460  26323  37981
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 5.458e+04  1.841e+03   29.65   <2e-16 ***
## cases       1.584e-02  8.356e-05  189.59   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 29510 on 607 degrees of freedom
## Multiple R-squared:  0.9834, Adjusted R-squared:  0.9834
## F-statistic: 3.595e+04 on 1 and 607 DF,  p-value: < 2.2e-16
```

Plotting the findings we obtained through our model.

```
us_total_w_pred <- us_total %>%
  mutate(pred = predict(model))
us_total_w_pred %>% ggplot() +
  geom_point(aes(x = cases, y = deaths), color = "blue")+
  geom_line(aes(x=cases, y = pred), color = "red")
```




**Conclusion**

Whatever analysis we do depends on the quality of the data we obtain, our data is obtained from countless different sources around the world and it's quality is definitely questionable.

But from doing whatever we can do with this data we obtained some important insights, things like Mississippi having the highest deaths per thousand in the US, when it doesn't even have the highest cases or deaths.

We also tried to predict how many new deaths can occur in the world depending on the new cases and also the final number of deaths in the US depending on the final cases.

In conclusion, we can say that even though the data is not perfect, we can arrive at valuable conclusions which can then help us to stitch together a solution in the field.

**Bias**

Biases in our data

1. Wrong results corrupting the data.
2. Possibility of multiple tests for one person and no tests for asymptomatic persons.
3. Died with different disease but may counted as covid.
4. Private testing may not be included.
5. No proper availability of testing kits.
6. Delay in adding the cases to the data.
7. Different data from different countries tends to be inconsistent.

Biases in our analysis includes using the simpler linear model to predict the deaths or cases instead of more powerful models, because linear model makes our work that much easier.