R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

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

First we are going to load the necessary packages for our analysis of the COVID19 dataset.

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.2     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- 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 errors
```

Next we are going to create the necessary URL's needed for the four data sets we will be using. By creating these URL's we can have r Studio pull the required datasets from the internet.

```
urls_in <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_co

file_names <- c("time_series_covid19_confirmed_global.csv", "time_series_covid19_deaths_global.csv", "t

urls <- str_c(urls_in, file_names)

urls
```

```
## [1] "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_
## [2] "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_
## [3] "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_
## [4] "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_
```

Next we are going to assign each data set a variable.

```
global_cases <- read_csv(urls[1])
```

```
## 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.
```

```
global_deaths <- 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.
```

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

```
## 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.
```

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

```
## 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.
```

For our next step we will want to being cleaning the data we pulled.

```
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_cases
```

```
## # 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
```

We are going to clean the gloabl deaths and the global clean datasets. Once we clean both we can join each dataset together.

```
library(lubridate)


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

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, cases)`
```

If we look at our new variable "global" we will see that our date column is now a date format, we have cases and death, and we have also renamed variables.

```
global
```

```
## # A tibble: 638,166 x 4
##    Province_State Country_Region date       cases
##    <chr>          <chr>          <date>     <dbl>
##  1 <NA>           Afghanistan    2020-01-22     0
##  2 <NA>           Afghanistan    2020-01-23     0
##  3 <NA>           Afghanistan    2020-01-24     0
##  4 <NA>           Afghanistan    2020-01-25     0
##  5 <NA>           Afghanistan    2020-01-26     0
##  6 <NA>           Afghanistan    2020-01-27     0
##  7 <NA>           Afghanistan    2020-01-28     0
##  8 <NA>           Afghanistan    2020-01-29     0
##  9 <NA>           Afghanistan    2020-01-30     0
## 10 <NA>           Afghanistan    2020-01-31     0
## # i 638,156 more rows
```

now we can look at the summary of the data to see if we have any problems

```
summary(global)
```

```
##  Province_State     Country_Region          date                cases
##  Length:638166      Length:638166      Min.   :2020-01-22   Min.   :        0
##  Class :character   Class :character   1st Qu.:2020-11-22   1st Qu.:       46
##  Mode  :character   Mode  :character   Median :2021-08-31   Median :     1852
##                                        Mean   :2021-08-28   Mean   :   503521
##                                        3rd Qu.:2022-06-06   3rd Qu.:    35610
##                                        Max.   :2023-03-09   Max.   :103802702
```

Since we have rows with 0 cases, we will filter out these rows to get dates where we only have positive numbers.

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

we can then resummarize data.

```
summary(global)
```

```
##  Province_State     Country_Region          date                cases
##  Length:581859      Length:581859      Min.   :2020-01-22   Min.   :        1
##  Class :character   Class :character   1st Qu.:2020-12-26   1st Qu.:      153
##  Mode  :character   Mode  :character   Median :2021-09-28   Median :     3118
##                                        Mean   :2021-09-21   Mean   :   552247
##                                        3rd Qu.:2022-06-22   3rd Qu.:    48980
##                                        Max.   :2023-03-09   Max.   :103802702
```

We can now see that our minimum is 1 and our max is 100 million. We can then take another step to double check that the 100 million is not a typo.

These are some measure you should take when doing an analysis.

```
global %>% filter(cases > 100000000)
```

```
## # A tibble: 80 x 4
##    Province_State Country_Region date       cases
##    <chr>          <chr>          <date>     <dbl>
##  1 <NA>           US             2022-12-20 100050937
##  2 <NA>           US             2022-12-21 100233060
##  3 <NA>           US             2022-12-22 100329204
##  4 <NA>           US             2022-12-23 100368433
##  5 <NA>           US             2022-12-24 100374955
##  6 <NA>           US             2022-12-25 100378169
##  7 <NA>           US             2022-12-26 100390601
##  8 <NA>           US             2022-12-27 100501536
##  9 <NA>           US             2022-12-28 100614880
## 10 <NA>           US             2022-12-29 100718983
## # i 70 more rows
```

The data looks correct so far because we have multiple data points in the US that surpass 100 million. Therefore, our max is not an outlier. Now our goal is to do an analysis on the us_cases.

```
US_cases
```

```
## # 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>, ...
```

Now we are going to being cleaning up our data sets that contain data rearding the United States.

```
US_cases %>%
  pivot_longer(cols = -(UID:Combined_Key),
               names_to = "date",
               values_to = "cases")
```

```
## # A tibble: 3,819,906 x 13
##         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 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
##  3 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
##  4 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
##  5 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
##  6 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
##  7 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
##  8 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
##  9 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
## 10 84001001 US    USA     840  1001 Autauga Alabama        US              32.5
## # i 3,819,896 more rows
## # i 4 more variables: Long_ <dbl>, Combined_Key <chr>, date <chr>, cases <dbl>
```

```
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: 3,819,906 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
## # i 3,819,896 more rows
```

Now that we have cleaned our dataset recodring US cases we are going to repeat the above steps for our US deaths dataset.

```
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_deaths
```

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

Now since we have both US cases and US deaths cleaned we can join both data sets together.

```
US <- US_cases %>%
  full_join(US_deaths)
```

```
## 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>
```

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

```
## # A tibble: 581,859 x 5
##    UniteCombined_Key Province_State Country_Region date       cases
##    <chr>             <chr>          <chr>          <date>     <dbl>
##  1 Afghanistan       <NA>           Afghanistan    2020-02-24     5
##  2 Afghanistan       <NA>           Afghanistan    2020-02-25     5
##  3 Afghanistan       <NA>           Afghanistan    2020-02-26     5
##  4 Afghanistan       <NA>           Afghanistan    2020-02-27     5
##  5 Afghanistan       <NA>           Afghanistan    2020-02-28     5
##  6 Afghanistan       <NA>           Afghanistan    2020-02-29     5
##  7 Afghanistan       <NA>           Afghanistan    2020-03-01     5
##  8 Afghanistan       <NA>           Afghanistan    2020-03-02     5
##  9 Afghanistan       <NA>           Afghanistan    2020-03-03     5
## 10 Afghanistan       <NA>           Afghanistan    2020-03-04     5
## # i 581,849 more rows
```

WE need the UID and join with global data set

```
uid_lookup_url <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/U
```

```
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.
```

{r join_UID, include = TRUE} 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) global

Next we would like to filter the data set so that we can view the number of deaths in each state.

```
US_by_state <- US %>%
  group_by(Province_State, Country_Region, date) %>%
  summarize(cases = sum(cases), deaths = sum(deaths),
            Population = sum(Population)) %>%
  mutate(death_per_mill = deaths * 1000000 / Population) %>%
  select(Province_State, Country_Region, date, cases, deaths, death_per_mill, Population) %>%
  ungroup()
```

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

```
US_by_state
```

```
## # A tibble: 66,294 x 7
##    Province_State Country_Region date       cases deaths death_per_mill
##    <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>
```

We will also be including a dataset of the deaths per million in the entire country.

```
US_totals <- US_by_state %>%
  group_by(Country_Region, date) %>%
  summarize(cases = sum(cases), deaths = sum(deaths),
            Population = sum(Population)) %>%
  mutate(death_per_mill = deaths *1000000/ Population) %>%
  select(Country_Region, date,
         cases, deaths, death_per_mill, Population) %>%
  ungroup()
```

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

```
US_totals
```

```
## # A tibble: 1,143 x 6
##    Country_Region date       cases deaths death_per_mill Population
##    <chr>          <date>     <dbl>  <dbl>          <dbl>      <dbl>
##  1 US             2020-01-22     1      1        0.00300  332875137
##  2 US             2020-01-23     1      1        0.00300  332875137
##  3 US             2020-01-24     2      1        0.00300  332875137
##  4 US             2020-01-25     2      1        0.00300  332875137
##  5 US             2020-01-26     5      1        0.00300  332875137
##  6 US             2020-01-27     5      1        0.00300  332875137
##  7 US             2020-01-28     5      1        0.00300  332875137
##  8 US             2020-01-29     6      1        0.00300  332875137
##  9 US             2020-01-30     6      1        0.00300  332875137
## 10 US             2020-01-31     8      1        0.00300  332875137
## # i 1,133 more rows
```

This will help us view the end of our data set to see how the beginning and end differ.

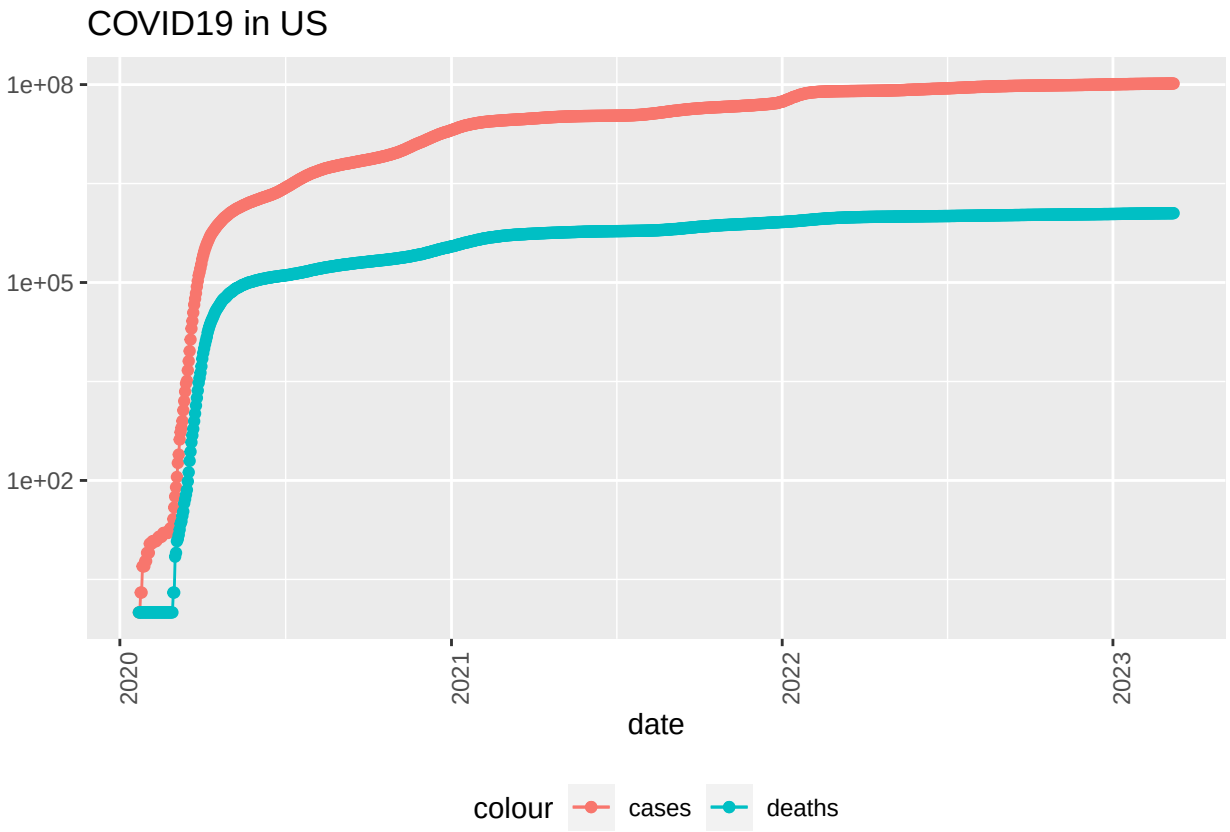
```
tail(US_totals)
```

```
## # A tibble: 6 x 6
##   Country_Region date           cases  deaths death_per_mill 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
```

Our next step is to create some visualizations of our analysis. We are creating a plot of the total number of cases versus the total number of deaths in the United States. Be careful with the valyes because the y valyes are scaled by a log function.

```
US_totals %>%
  filter(cases >0) %>%
  ggplot(aes( x = date, y = cases)) +
  geom_line(aes(color = "cases")) +
```

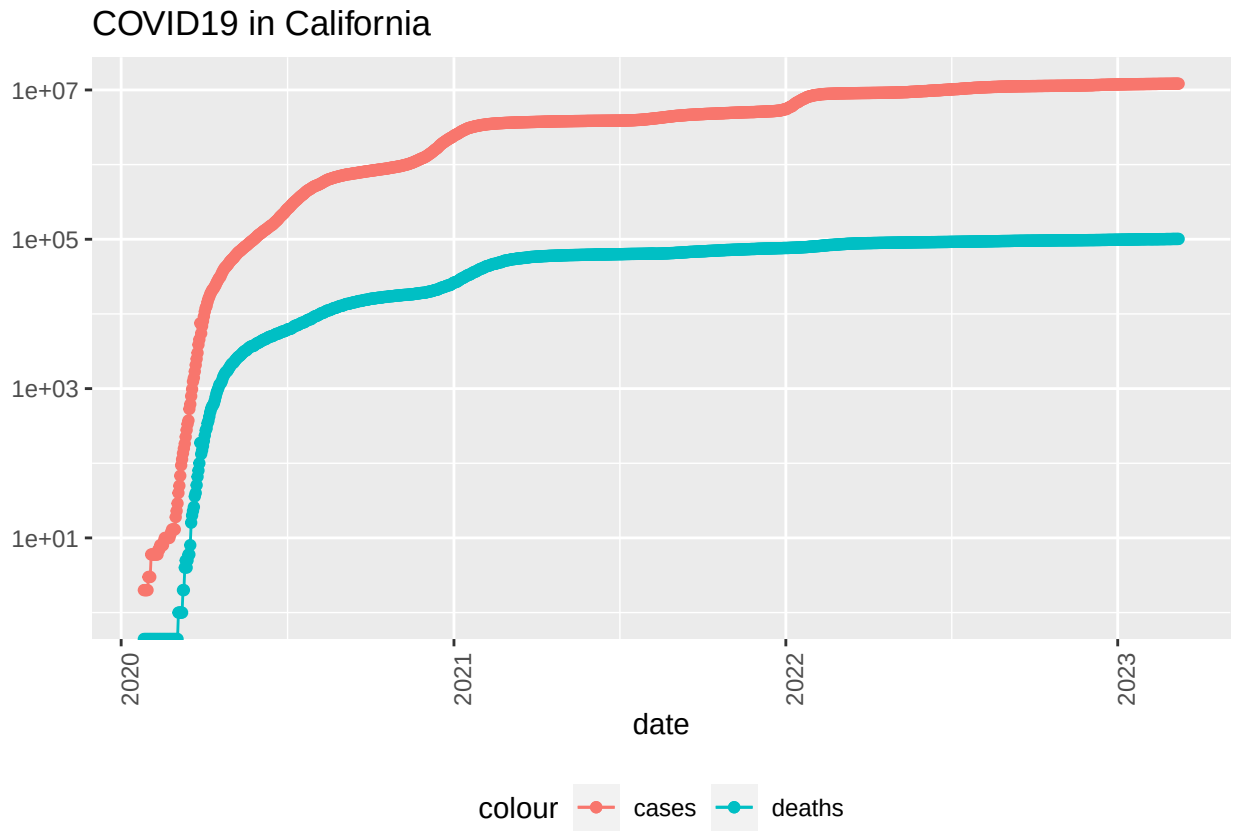
```
  geom_point(aes(color = "cases")) +
  geom_line(aes(y= deaths, color = "deaths")) +
  geom_point(aes(y=deaths, color = "deaths")) +
  scale_y_log10() +
  theme(legend.position = "bottom" ,
        axis.text.x = element_text(angle = 90)) +
  labs(title = "COVID19 in US" , y = NULL)
```



On top of our visualization of the United States data, we will be viewing a specific state in the United States to see the trends in the cases vs total deaths, similarly to the graph above. I will be looking at two states both "California", "Colorado" and "Arizona".

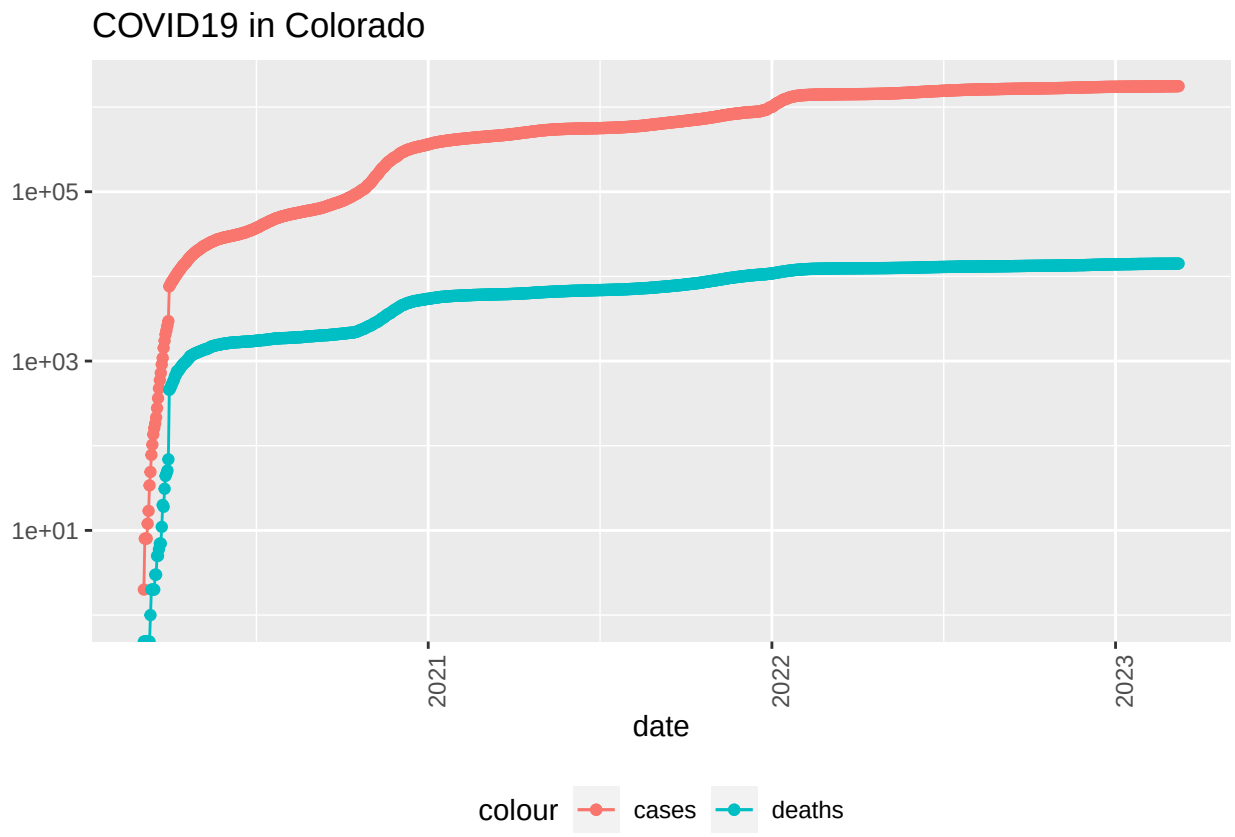
```
state <- "California"

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
```



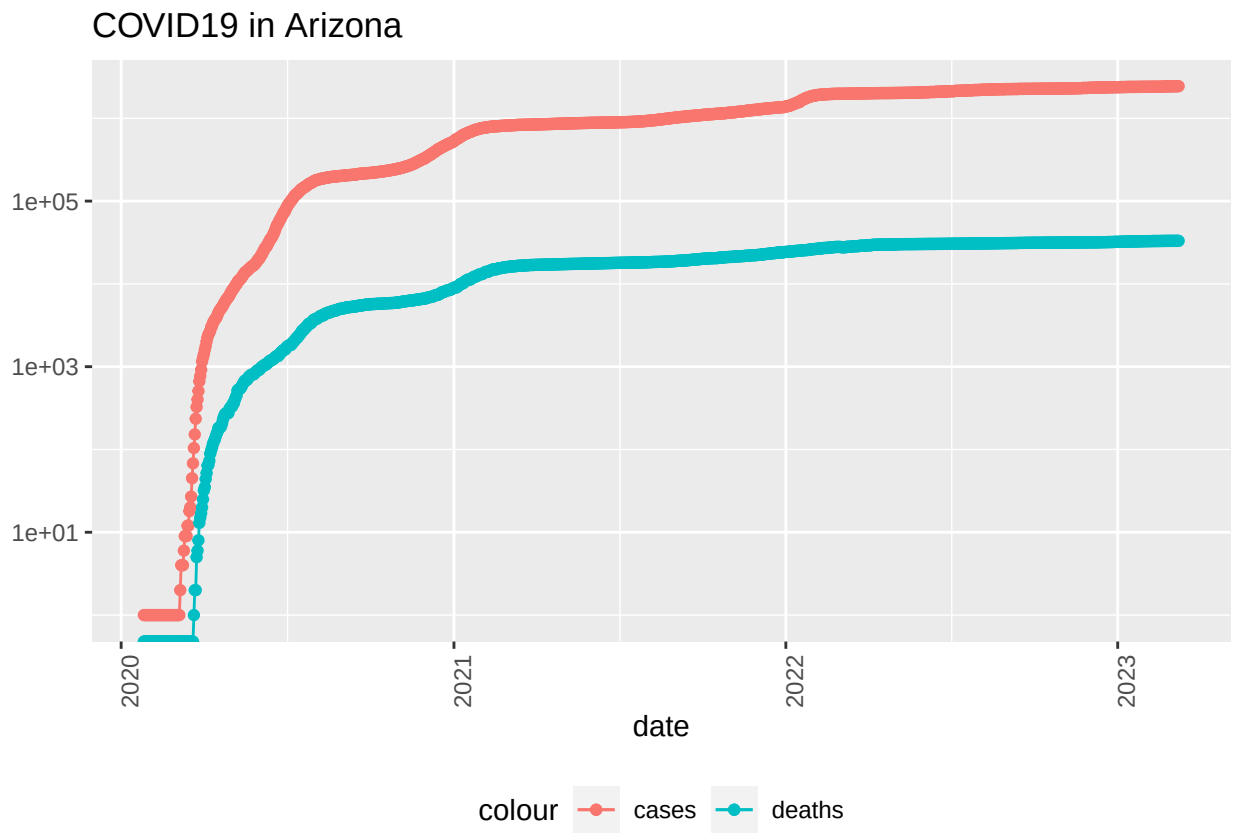

```
state <- "Colorado"

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
```




```
state <- "Arizona"

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
```



Nex in our analysis we need to find the max number of deaths and the end date of our data set.

```
max(US_totals$date)
```

```
## [1] "2023-03-09"
```

```
max(US_totals$deaths)
```

```
## [1] 1123836
```

Next we are going to calculate the about of new cases and new deaths in each state and throughout the country as a whole.

```
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))
```

Print out the new data set of US_total. The total of new cases and new deaths will be displayed.

```
tail(US_totals)
```

```
## # A tibble: 6 x 8
##   Country_Region date           cases deaths death_per_mill Population new_cases
##   <chr>          <date>         <dbl>  <dbl>          <dbl>      <dbl>     <dbl>
## 1 US             2023-03-04 103650837 1.12e6          3371.  332875137      2147
## 2 US             2023-03-05 103646975 1.12e6          3371.  332875137     -3862
## 3 US             2023-03-06 103655539 1.12e6          3371.  332875137      8564
## 4 US             2023-03-07 103690910 1.12e6          3372.  332875137     35371
## 5 US             2023-03-08 103755771 1.12e6          3374.  332875137     64861
## 6 US             2023-03-09 103802702 1.12e6          3376.  332875137     46931
## # i 1 more variable: new_deaths <dbl>
```

We can reorganize the table so that the total for new cases and new deaths are displayed first.

```
tail(US_totals %>% select(new_cases, new_deaths, everything()))
```

```
## # A tibble: 6 x 8
##   new_cases new_deaths Country_Region date           cases deaths death_per_mill
##       <dbl>      <dbl> <chr>          <date>         <dbl>  <dbl>          <dbl>
## 1      2147          7 US             2023-03-04 103650837 1.12e6          3371.
## 2     -3862        -38 US             2023-03-05 103646975 1.12e6          3371.
## 3      8564         47 US             2023-03-06 103655539 1.12e6          3371.
## 4     35371        335 US             2023-03-07 103690910 1.12e6          3372.
## 5     64861        730 US             2023-03-08 103755771 1.12e6          3374.
## 6     46931        590 US             2023-03-09 103802702 1.12e6          3376.
## # i 1 more variable: Population <dbl>
```

Next I will create a graph of the number of new cases and the total number of deaths throughout the United States.

```
US_totals %>%
  filter(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= 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 = "COVID19 in US" , y = NULL)
```

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

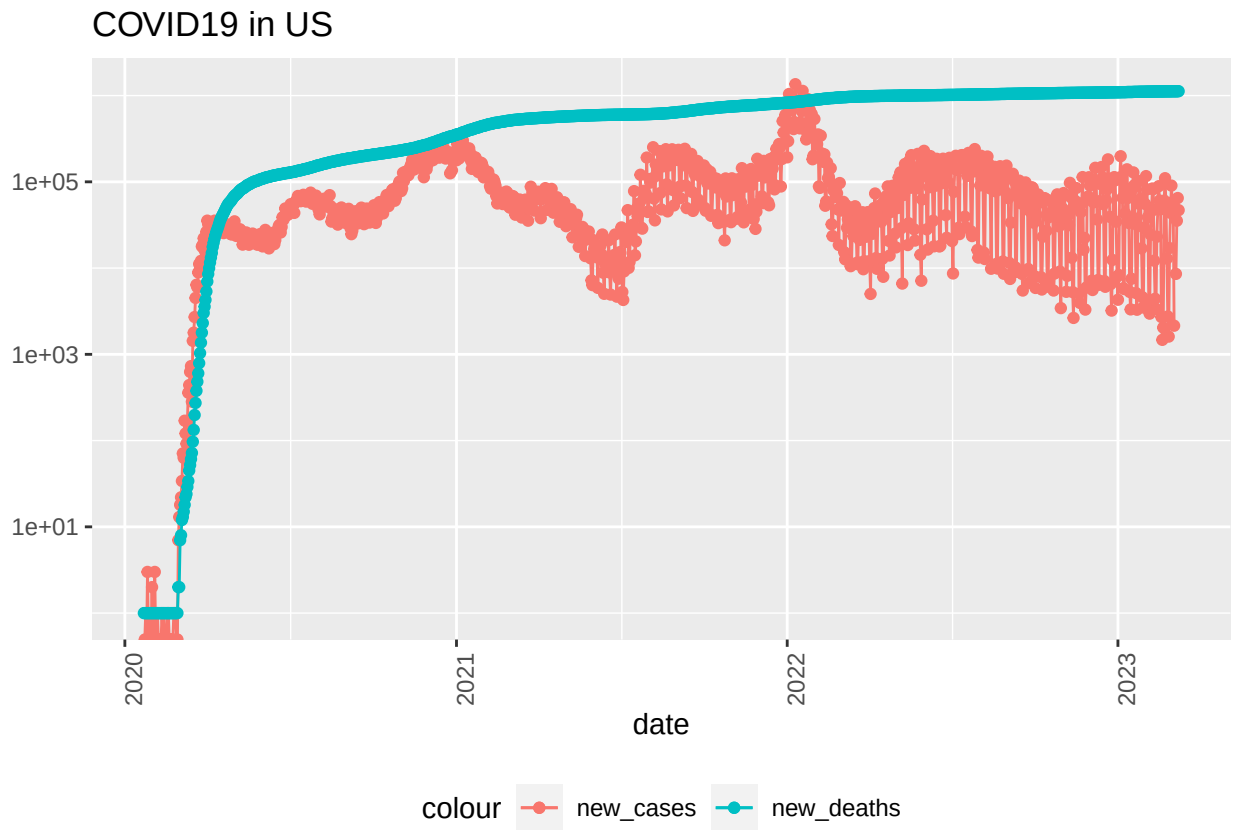
```
## Warning: Transformation introduced infinite values in continuous y-axis
```

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

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

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

```
## Warning: Removed 2 rows containing missing values (`geom_point()`).
```



I will also create a similar visualization for California, Colorado and Arizona.

```
state <- "California"

US_by_state %>%
  filter(Province_State == state) %>%
  filter(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= 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 = str_c("COVID19 in ", state) , y = NULL)
```

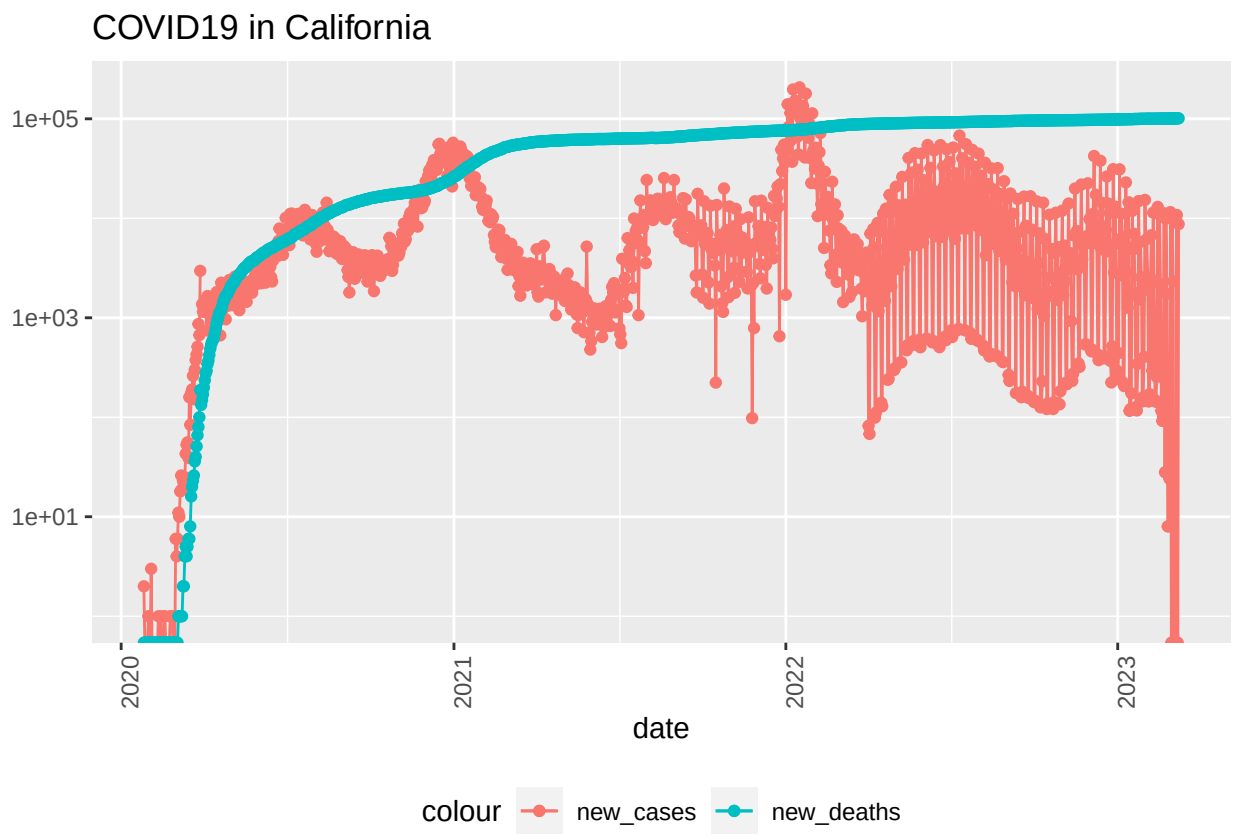
```
## Warning in self$trans$transform(x): NaNs produced
```

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

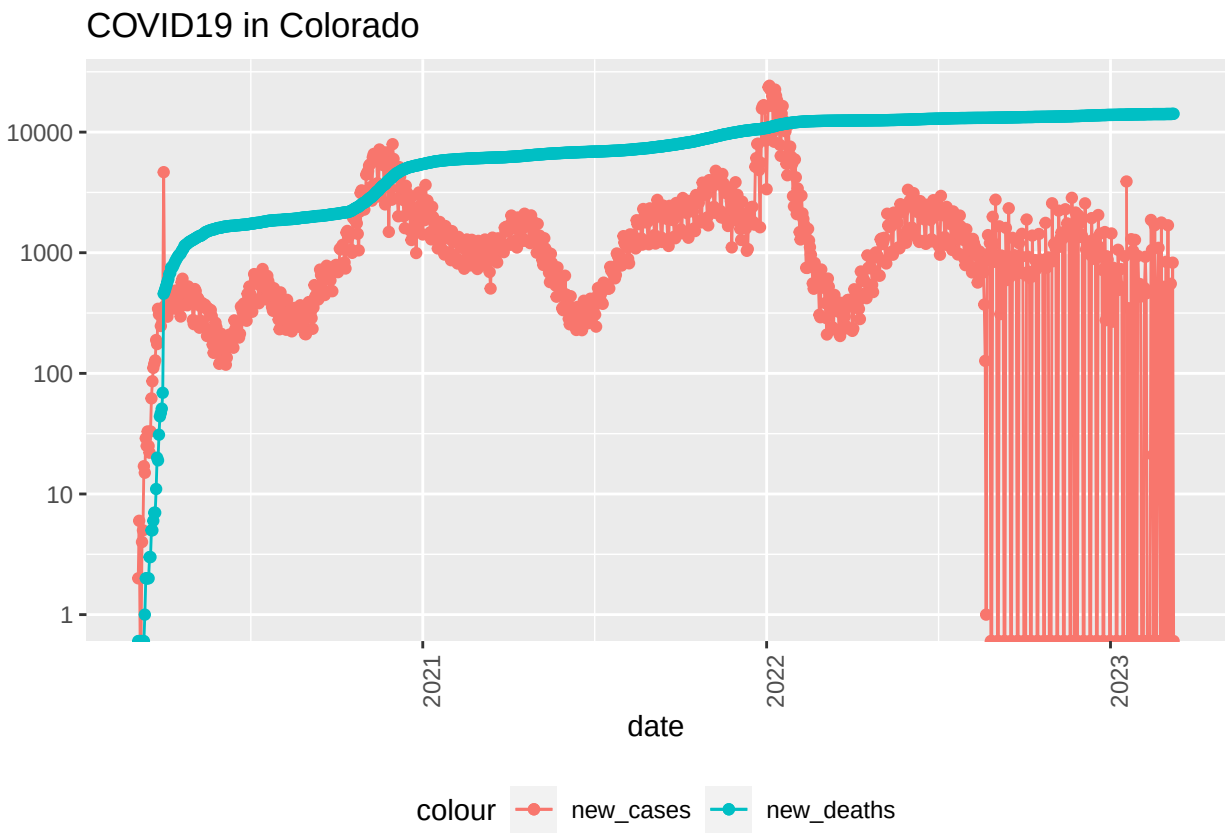
```
## Warning: Removed 2 rows containing missing values (`geom_point()`).
```




```
state <- "Colorado"

US_by_state %>%
  filter(Province_State == state) %>%
  filter(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= 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 = str_c("COVID19 in ", state) , y = NULL)
```

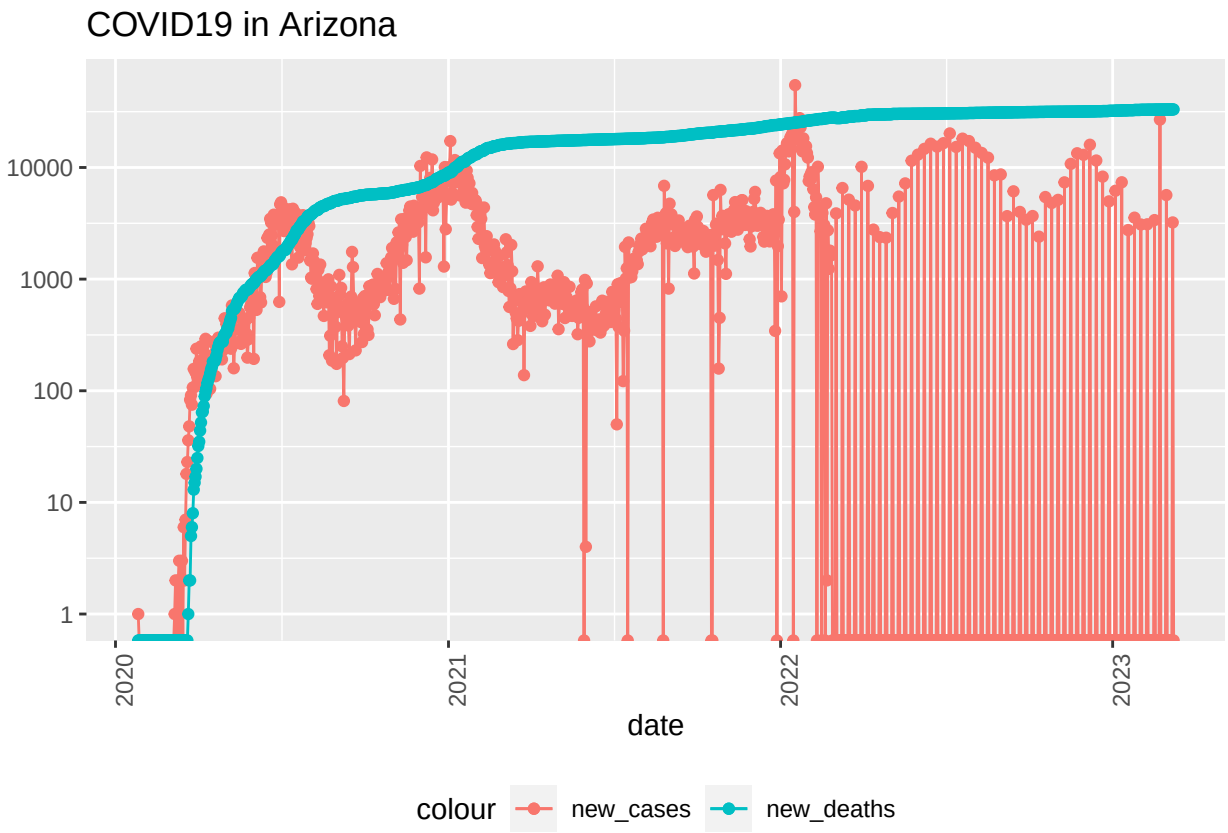
```
## Warning in self$trans$transform(x): NaNs produced
```

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

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




```r
state <- "Arizona"

US_by_state %>%
  filter(Province_State == state) %>%
  filter(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= 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 = str_c("COVID19 in ", state) , y = NULL)
```

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



Next in the analysis, I will be determining which states have the highest and least infection rate throughout the United States.

```
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)
```

This section of code will produce the areas with the lowest fatality rate.

```
US_state_totals %>%
  slice_min(deaths_per_thou, n=10) %>%
  select(deaths_per_thou, cases_per_thou, everything())
```

```
## # A tibble: 10 x 6
##    deaths_per_thou cases_per_thou Province_State        deaths  cases population
##              <dbl>          <dbl> <chr>                  <dbl>  <dbl>      <dbl>
```

```
##  1           0.611           150. American Samoa            34 8.32e3      55641
##  2           0.744           248. Northern Mariana Isl~     41 1.37e4      55144
##  3           1.21            231. Virgin Islands           130 2.48e4     107268
##  4           1.30            269. Hawaii                  1841 3.81e5    1415872
##  5           1.49            245. Vermont                  929 1.53e5     623989
##  6           1.55            293. Puerto Rico             5823 1.10e6    3754939
##  7           1.65            340. Utah                    5298 1.09e6    3205958
##  8           2.01            415. Alaska                  1486 3.08e5     740995
##  9           2.03            252. District of Columbia    1432 1.78e5     705749
## 10           2.06            253. Washington             15683 1.93e6    7614893
```

This section of code will produce the section with the highest fatality rate.

```
US_state_totals %>%
  slice_max(deaths_per_thou, n=10) %>%
  select(deaths_per_thou, cases_per_thou, everything())
```

```
## # A tibble: 10 x 6
##    deaths_per_thou cases_per_thou Province_State deaths   cases population
##              <dbl>          <dbl> <chr>           <dbl>   <dbl>      <dbl>
##  1            4.55           336. Arizona         33102 2443514    7278717
##  2            4.54           326. Oklahoma        17972 1290929    3956971
##  3            4.49           333. Mississippi     13370  990756    2976149
##  4            4.44           359. West Virginia    7960  642760    1792147
##  5            4.32           320. New Mexico       9061  670929    2096829
##  6            4.31           334. Arkansas        13020 1006883    3017804
##  7            4.29           335. Alabama         21032 1644533    4903185
##  8            4.28           368. Tennessee       29263 2515130    6829174
##  9            4.23           307. Michigan        42205 3064125    9986857
## 10            4.06           385. Kentucky        18130 1718471    4467673
```

Lastly, we will create a model of our data. THe code chuck below will create a model by comparing the deaths and cases per thousand from the US_State_total data frame.

```
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 -0.5978  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
```

This oce chuck will find our max and min for the number of cases.

```
US_state_totals %>% slice_min(cases_per_thou)
```

```
## # A tibble: 1 x 6
##   Province_State deaths cases population cases_per_thou deaths_per_thou
##   <chr>           <dbl> <dbl>      <dbl>          <dbl>           <dbl>
## 1 American Samoa     34  8320      55641           150.           0.611
```

```
US_state_totals %>% slice_max(cases_per_thou)
```

```
## # A tibble: 1 x 6
##   Province_State deaths  cases population cases_per_thou deaths_per_thou
##   <chr>           <dbl>  <dbl>      <dbl>          <dbl>           <dbl>
## 1 Rhode Island     3870 460697    1059361           435.            3.65
```

Now based off our information we will create a linear model that best fit our data.

```
US_state_totals %>% mutate(pred = predict(mod))
```

```
## # A tibble: 56 x 7
##    Province_State  deaths  cases population cases_per_thou deaths_per_thou  pred
##    <chr>            <dbl>  <dbl>      <dbl>          <dbl>           <dbl> <dbl>
##  1 Alabama          21032 1.64e6    4903185           335.            4.29   3.44
##  2 Alaska            1486 3.08e5     740995           415.            2.01   4.34
##  3 American Samoa      34 8.32e3      55641           150.            0.611  1.33
##  4 Arizona          33102 2.44e6    7278717           336.            4.55   3.44
##  5 Arkansas         13020 1.01e6    3017804           334.            4.31   3.42
##  6 California      101159 1.21e7   39512223           307.            2.56   3.12
##  7 Colorado         14181 1.76e6    5758736           306.            2.46   3.11
##  8 Connecticut      12220 9.77e5    3565287           274.            3.43   2.74
##  9 Delaware          3324 3.31e5     973764           340.            3.41   3.49
## 10 District of Co~   1432 1.78e5     705749           252.            2.03   2.49
## # i 46 more rows
```

Precition code continued.

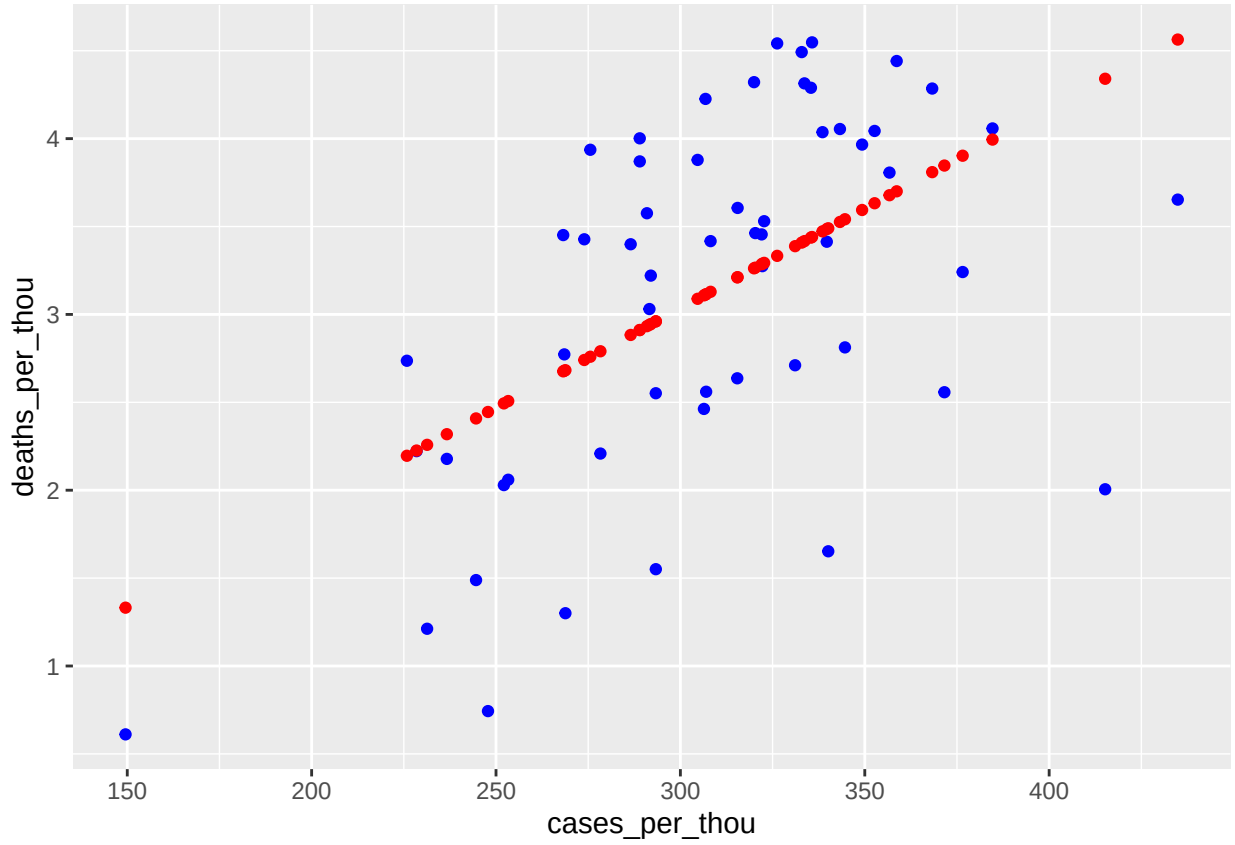
```
US_tot_w_pred <- US_state_totals %>% mutate(pred = predict(mod))
US_tot_w_pred
```

```
## # A tibble: 56 x 7
##    Province_State  deaths  cases population cases_per_thou deaths_per_thou  pred
##    <chr>            <dbl>  <dbl>      <dbl>          <dbl>           <dbl> <dbl>
##  1 Alabama          21032 1.64e6    4903185           335.            4.29   3.44
##  2 Alaska            1486 3.08e5     740995           415.            2.01   4.34
##  3 American Samoa      34 8.32e3      55641           150.            0.611  1.33
```

```
##  4 Arizona          33102 2.44e6    7278717          336.          4.55   3.44
##  5 Arkansas         13020 1.01e6    3017804          334.          4.31   3.42
##  6 California      101159 1.21e7   39512223          307.          2.56   3.12
##  7 Colorado         14181 1.76e6    5758736          306.          2.46   3.11
##  8 Connecticut      12220 9.77e5    3565287          274.          3.43   2.74
##  9 Delaware          3324 3.31e5     973764          340.          3.41   3.49
## 10 District of Co~   1432 1.78e5     705749          252.          2.03   2.49
## # i 46 more rows
```

Final visualization of out model and our data set comparing deaths per thousand and cases per thousand.

```
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")
```



Concluding my report, I noticed that bias may have occurred throughout my analysis. The top form of bias can have occurred due to the accuracy of reporting. In some district resources may have not been as readily available. Therefore, this may have led to a significant number of cases not being accurately reported due to insufficient testing and resources. This could lead to inaccuracy of maximum and minimum cases reported in a day for the country or individual states.