COVID-19 Homework

2023-06-27

COVID-19 Analysis

In this analysis based on the analysis seen in class, I will predict deaths by thousands in the COVID-19 pandemy based on the positive cases in the countries of US and Peru.

Libraries to use

```
## -- 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
```

Dataset importing

First of all, lets import the dataset of COVID-19 from source. In this case I will import the US only dataset, and the global dataset.

```
url_base = "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(url_base, file_names)
```

Next, I will import the dataframes in the in the R editor.

```
global_cases = read_csv(urls[1])
global_deaths = read_csv(urls[2])
US_cases = read_csv(urls[3])
US_deaths = read_csv(urls[4])
```

Let's explore a little bit the data.

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

```
## # A tibble: 6 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
## # i 4 more variables: Long_ <dbl>, Combined_Key <chr>, date <chr>, cases <dbl>
```

First, pivoting the table.

```
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_))
```

And then watching the result in the US Cases.

```
tail(US_cases)
```

```
## # A tibble: 6 x 6
##   Admin2 Province_State Country_Region Combined_Key        date       cases
##   <chr>  <chr>          <chr>          <chr>               <date>     <dbl>
## 1 Weston Wyoming        US             Weston, Wyoming, US 2023-03-04  1905
## 2 Weston Wyoming        US             Weston, Wyoming, US 2023-03-05  1905
## 3 Weston Wyoming        US             Weston, Wyoming, US 2023-03-06  1905
## 4 Weston Wyoming        US             Weston, Wyoming, US 2023-03-07  1905
## 5 Weston Wyoming        US             Weston, Wyoming, US 2023-03-08  1905
## 6 Weston Wyoming        US             Weston, Wyoming, US 2023-03-09  1905
```

Now, let's pivot the deaths Data Frame.

```
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_))
```

And then seeing what is the result.

```
head(US_deaths)
```

```
## # A tibble: 6 x 7
##   Admin2 Province_State Country_Region Combined_Key Population date       deaths
##   <chr>  <chr>          <chr>          <chr>             <dbl> <date>      <dbl>
## 1 Autau~ Alabama        US             Autauga, Al~      55869 2020-01-22      0
## 2 Autau~ Alabama        US             Autauga, Al~      55869 2020-01-23      0
## 3 Autau~ Alabama        US             Autauga, Al~      55869 2020-01-24      0
## 4 Autau~ Alabama        US             Autauga, Al~      55869 2020-01-25      0
## 5 Autau~ Alabama        US             Autauga, Al~      55869 2020-01-26      0
## 6 Autau~ Alabama        US             Autauga, Al~      55869 2020-01-27      0
```

Joining Cases and Deaths

```
US = US_cases %>%
  full_join(US_deaths)
```

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

And, let's see how it looks!

```
head(US)
```

```
## # A tibble: 6 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
## # i 1 more variable: deaths <dbl>
```

Overview of how are the results

```
uid_url_lookup = 'https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/U

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

```
## Rows: 4321 Columns: 12
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (7): iso2, iso3, FIPS, Admin2, Province_State, Country_Region, Combined_Key
## dbl (5): UID, code3, Lat, Long_, Population
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
uid
```

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

US Visualizing

First of all we need to formatting the deaths by million, so in the way to achieve it we need to execute the next cell.

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

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

```
head(US_by_state)
```

```
## # A tibble: 6 x 7
##   Province_State Country_Region date       cases deaths deaths_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
## # i 1 more variable: Population <dbl>
```

And, again, let's do it with the other Data Frame, of US totals.

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

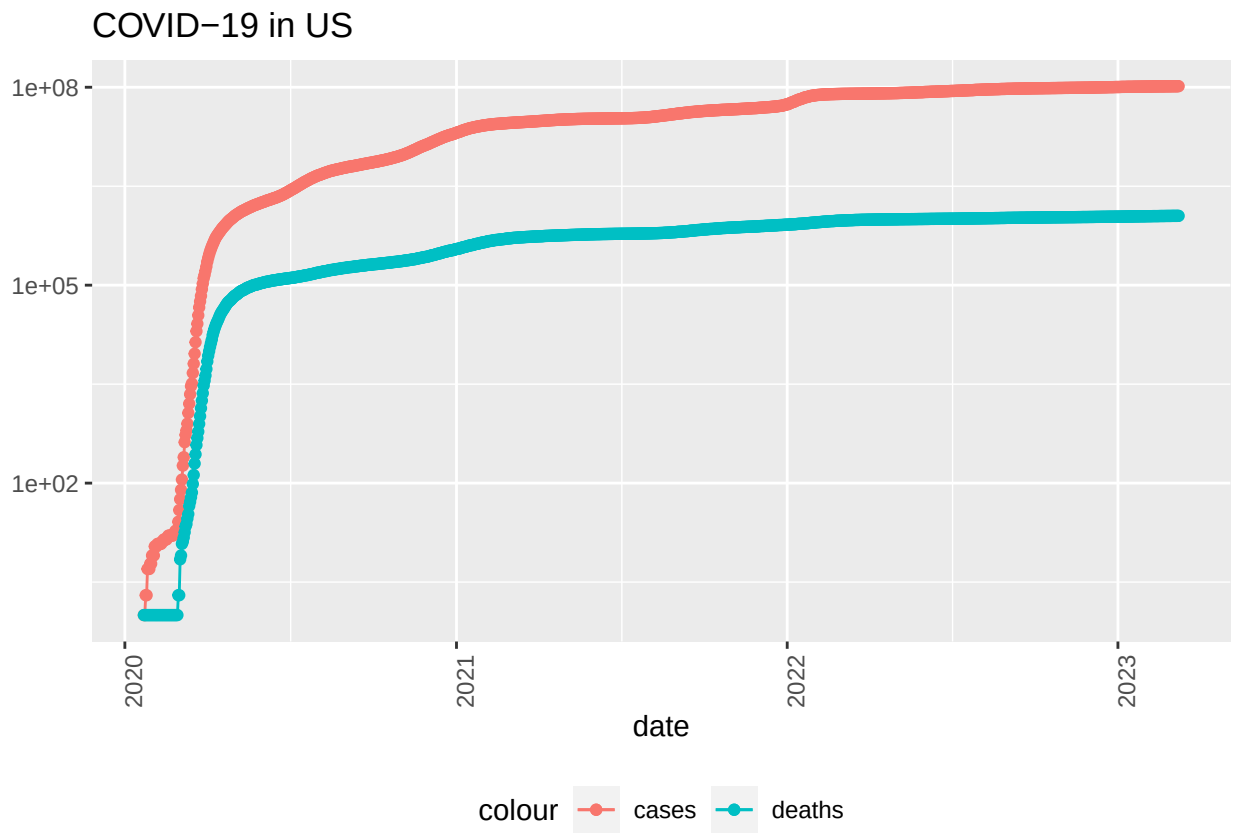
```
## `summarise()` has grouped output by 'Country_Region'. You can override using
## the `.groups` argument.
```

```
head(US_totals)
```

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

Visualizing the data First US as a country

```
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 = "COVID-19 in US", y=NULL)
```

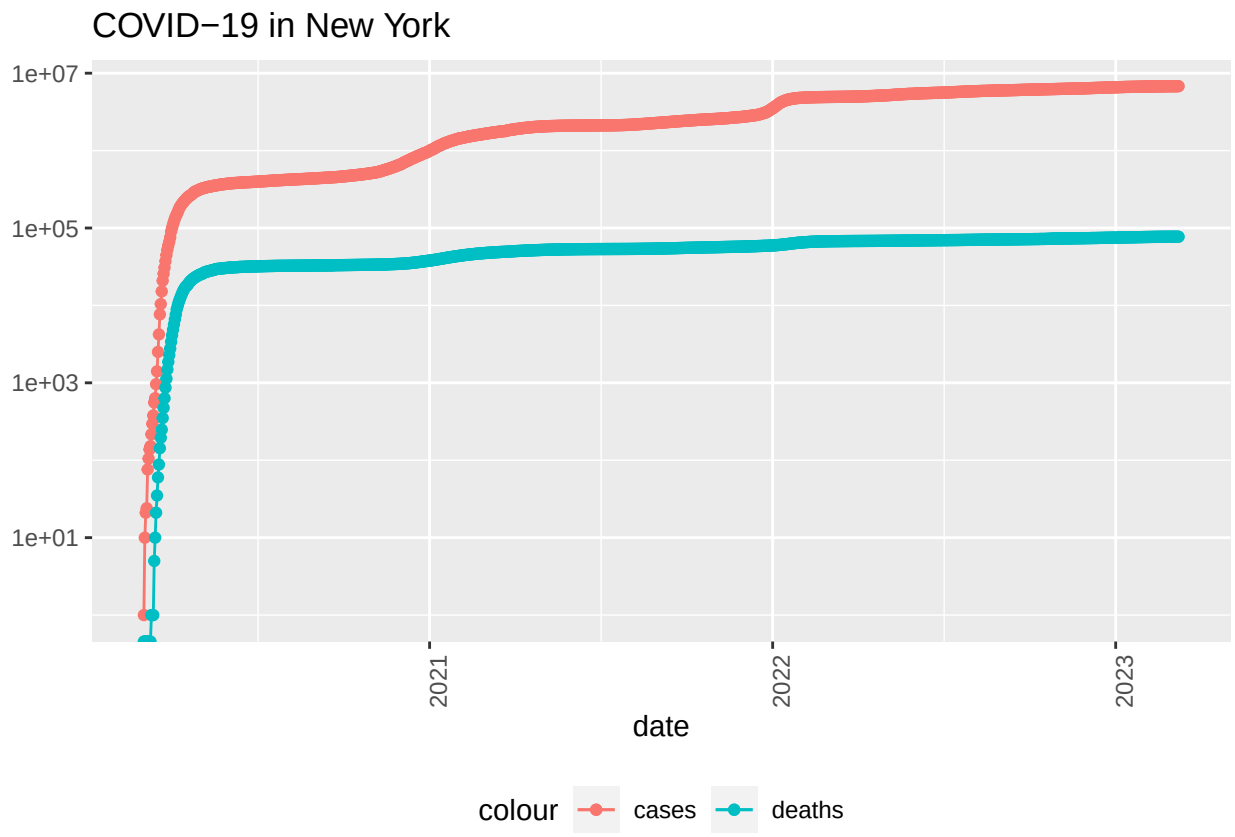


COVID-19 in New York plot Let's see some states of US individually.

First, our week 3 project city NY.

```
state = "New York"

US_by_state %>%
  filter(Province_State == state) %>%
  filter(cases > 0) %>%
  ggplot(aes(x = date, y = cases)) +
  geom_line(aes(color = "cases") ) +
  geom_point(aes(color = "cases")) +
  geom_line(aes(y = deaths, color = "deaths")) +
  geom_point(aes(y = deaths, color = "deaths")) +
  scale_y_log10() +
  theme(legend.position = "bottom",
        axis.text.x = element_text(angle = 90)) +
  labs(title = "COVID-19 in New York", y=NULL)
```

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



Interesting, but now we need the new cases. Let's make a new column with the US_by_state and US_totals Data Frames.

```
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))
```

And then seen how the data looks like.

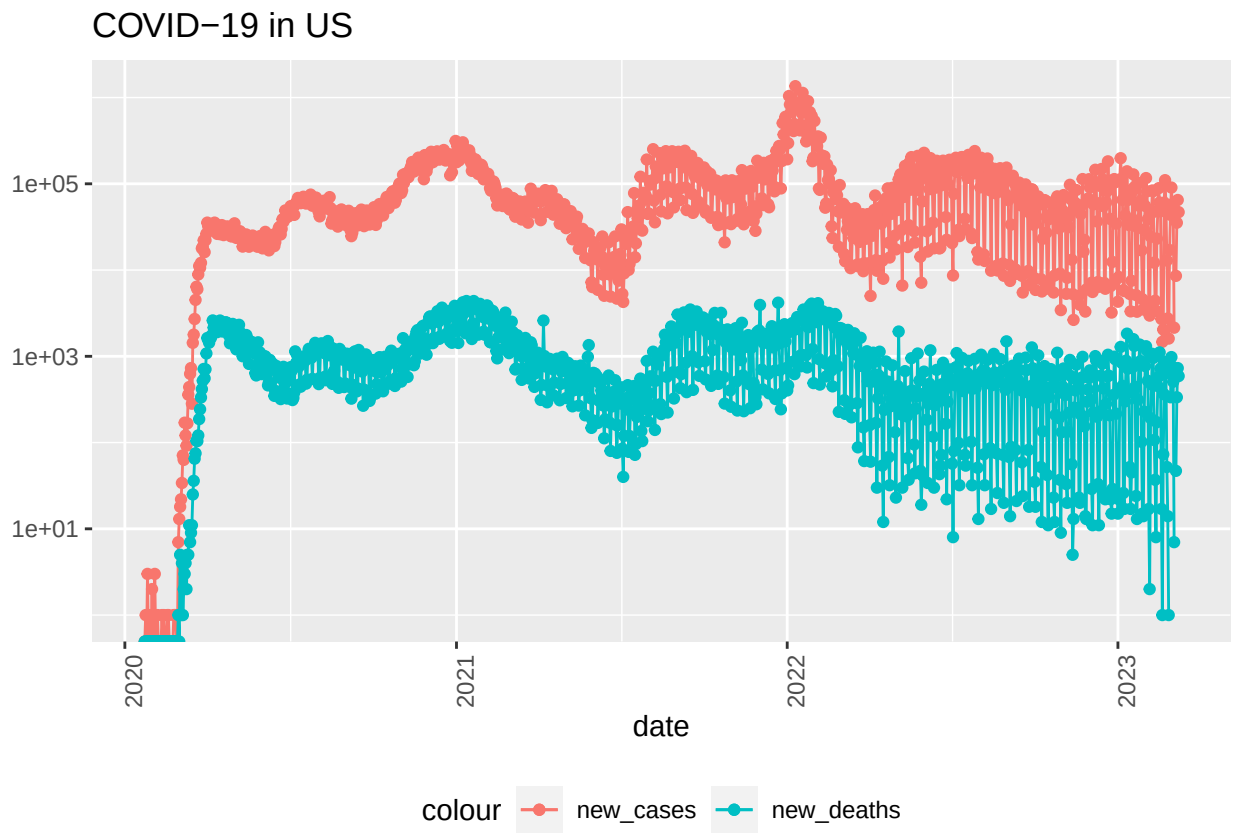
```
tail(US_totals %>% select(new_cases, new_deaths, everything()))
```

```
## # A tibble: 6 x 8
##   new_cases new_deaths Country_Region date          cases deaths deaths_per_mill
##       <dbl>      <dbl> <chr>          <date>        <dbl>  <dbl>           <dbl>
## 1      2147          7 US             2023-03-04   1.04e8 1.12e6           3371.
## 2     -3862        -38 US             2023-03-05   1.04e8 1.12e6           3371.
## 3      8564         47 US             2023-03-06   1.04e8 1.12e6           3371.
## 4     35371        335 US             2023-03-07   1.04e8 1.12e6           3372.
## 5     64861        730 US             2023-03-08   1.04e8 1.12e6           3374.
## 6     46931        590 US             2023-03-09   1.04e8 1.12e6           3376.
## # i 1 more variable: Population <dbl>
```

And now, let's watch the behavior of the new cases across the time for all the 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 = new_deaths, color = "new_deaths")) +
  geom_point(aes(y = new_deaths, color = "new_deaths")) +
  scale_y_log10() +
  theme(legend.position = "bottom",
        axis.text.x = element_text(angle = 90)) +
  labs(title = "COVID-19 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 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()`).

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

## Warning: Removed 4 rows containing missing values (`geom_point()`).
```

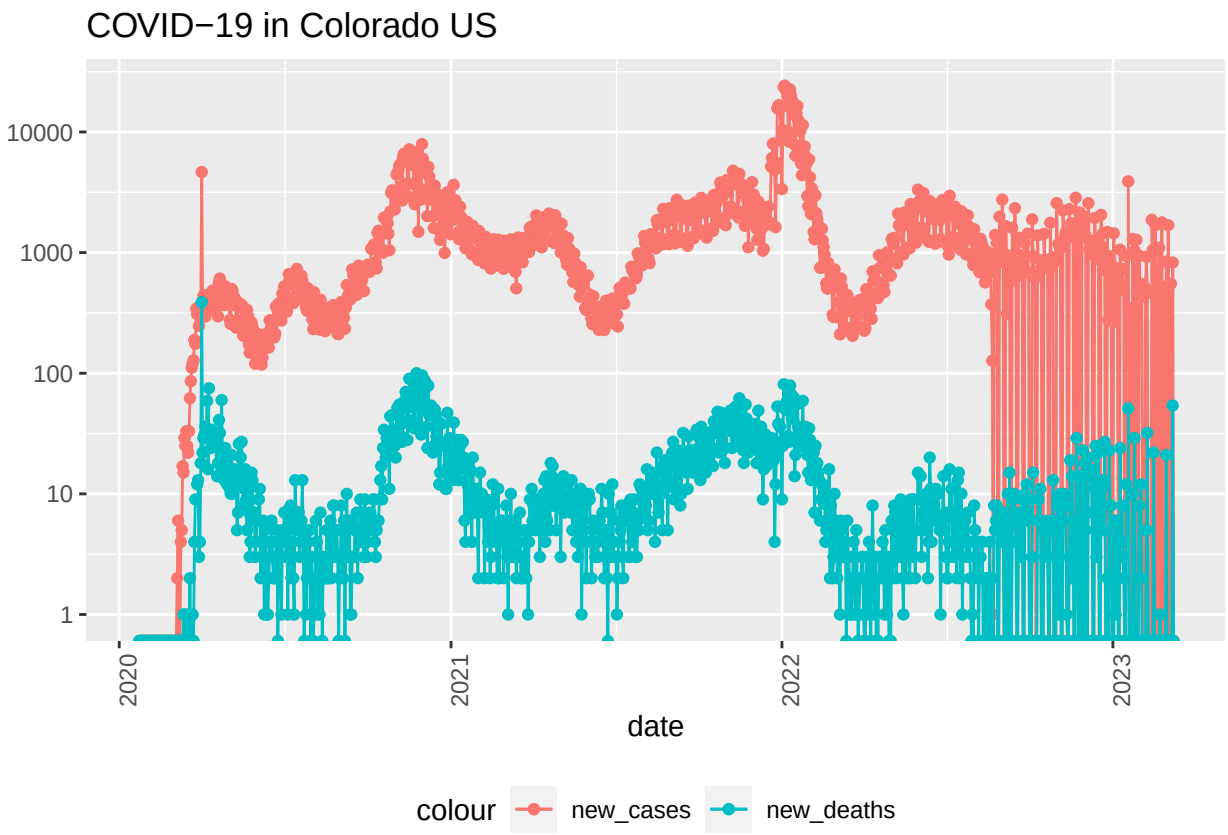


What about the state of Colorado? Let's watch it.

```
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 = "bottom",
        axis.text.x = element_text(angle = 90)) +
  labs(title = "COVID-19 in Colorado 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 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()`).

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

## Warning: Removed 5 rows containing missing values (`geom_point()`).
```



Finally, let's create new columns based on the deaths, population, cases, and cases per thousands and also deaths per thousands.

```
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)
```

And then seen the tail of this data.

```
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
```

```
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
```

Create a linear model for the US

First, let's train this model. We want to predict the deaths based on the cases per thousands.

```
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
```

Again we want to see the minimum number of cases and the maximum number of cases, because our model will work with these points.

```
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
```

And then let's predict the deaths with our model.

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

```
## # A tibble: 6 x 7
##   Province_State deaths   cases population cases_per_thou deaths_per_thou  pred
##   <chr>           <dbl>   <dbl>      <dbl>          <dbl>           <dbl> <dbl>
## 1 Virgin Islands    130   24813     107268           231.            1.21  2.26
## 2 Virginia        23666 2291951    8535519           269.            2.77  2.68
## 3 Washington      15683 1928913    7614893           253.            2.06  2.51
## 4 West Virginia    7960  642760    1792147           359.            4.44  3.70
## 5 Wisconsin       16375 2006582    5822434           345.            2.81  3.54
## 6 Wyoming          2004  185385     578759           320.            3.46  3.27
```

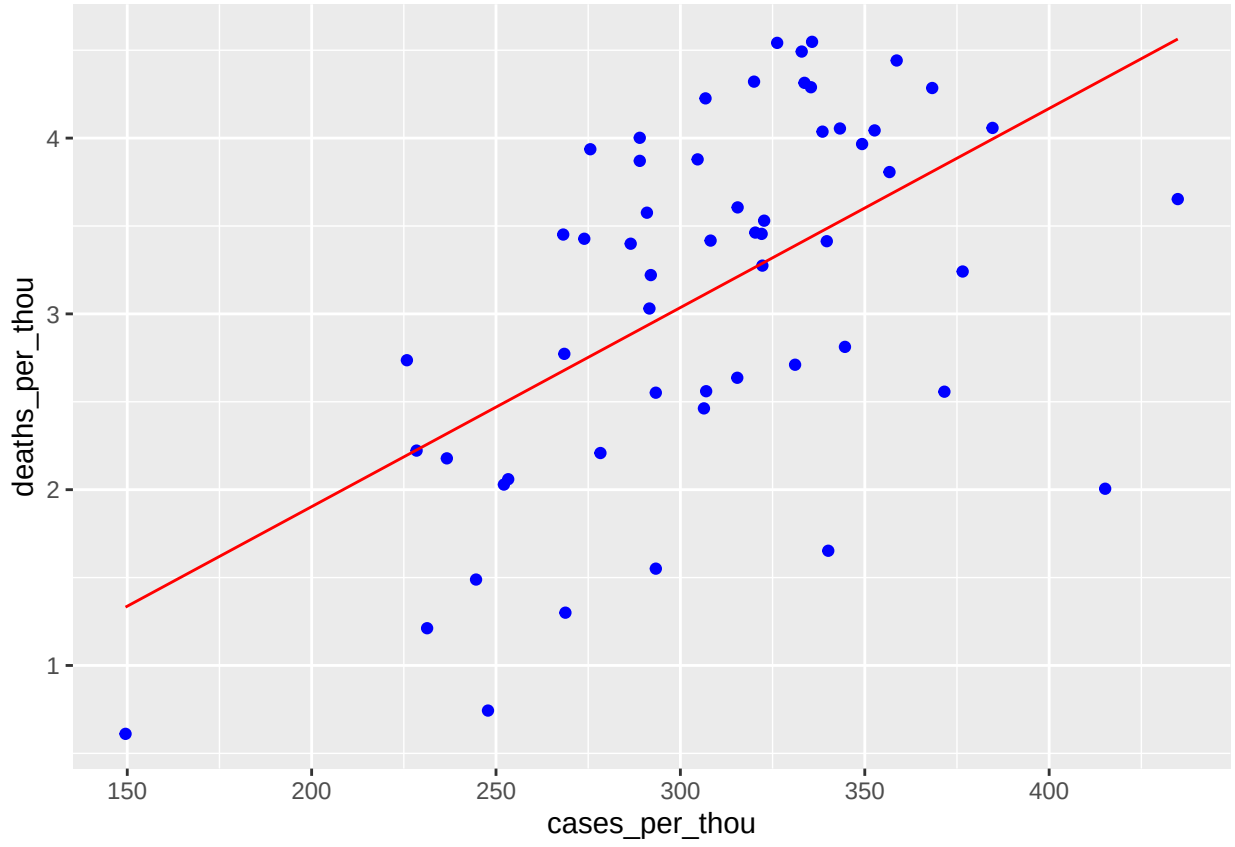
And then creating a Data Frame

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

```
## # A tibble: 6 x 7
##   Province_State deaths   cases population cases_per_thou deaths_per_thou  pred
##   <chr>           <dbl>   <dbl>      <dbl>          <dbl>           <dbl> <dbl>
## 1 Virgin Islands    130   24813     107268           231.            1.21  2.26
## 2 Virginia        23666 2291951    8535519           269.            2.77  2.68
## 3 Washington      15683 1928913    7614893           253.            2.06  2.51
## 4 West Virginia    7960  642760    1792147           359.            4.44  3.70
## 5 Wisconsin       16375 2006582    5822434           345.            2.81  3.54
## 6 Wyoming          2004  185385     578759           320.            3.46  3.27
```

Visualizing the prediction for the US

```
US_st_totals_w_pred %>%
  ggplot() +
  geom_point(aes(x = cases_per_thou, y = deaths_per_thou), color = "blue") +
  geom_line(aes(x = cases_per_thou, y = pred), color = "red")
```



Global cases - In concrete Peru

For the case of Peru, I want to make something different. I want to see the future of the possitive cases. We use the function lag in the case of US, so the main idea is: "Based in the previous week, what will be the expected possitive cases for today?"

```
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))
```

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

```
summary(global)
```

```
##  Province_State     Country_Region          date                cases
##  Length:330327      Length:330327      Min.   :2020-01-22   Min.   :        0
##  Class :character   Class :character   1st Qu.:2020-11-02   1st Qu.:      680
##  Mode  :character   Mode  :character   Median :2021-08-15   Median :    14429
##                                        Mean   :2021-08-15   Mean   :   959384
##                                        3rd Qu.:2022-05-28   3rd Qu.:   228517
##                                        Max.   :2023-03-09   Max.   :103802702
##      deaths
##  Min.   :      0
##  1st Qu.:      3
##  Median :    150
##  Mean   :  13380
##  3rd Qu.:   3032
##  Max.   :1123836
```

```
global = global %>% filter(cases > 0)
```

```
global %>% filter(cases > 28000000)
```

```
## # A tibble: 2,510 x 5
##    Province_State Country_Region date          cases deaths
##    <chr>          <chr>          <date>        <dbl>  <dbl>
##  1 <NA>           Brazil         2022-02-18 28072238 643340
##  2 <NA>           Brazil         2022-02-19 28177367 644195
##  3 <NA>           Brazil         2022-02-20 28218180 644592
##  4 <NA>           Brazil         2022-02-21 28258458 644918
##  5 <NA>           Brazil         2022-02-22 28361951 645735
##  6 <NA>           Brazil         2022-02-23 28493336 646714
```

```
##  7 <NA>           Brazil         2022-02-24 28589235 647703
##  8 <NA>           Brazil         2022-02-25 28679671 648496
##  9 <NA>           Brazil         2022-02-26 28749552 649184
## 10 <NA>           Brazil         2022-02-27 28776794 649437
## # i 2,500 more rows
```

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

```
## # A tibble: 6 x 6
##   Combined_Key Province_State Country_Region date        cases deaths
##   <chr>        <chr>          <chr>          <date>      <dbl>  <dbl>
## 1 Zimbabwe     <NA>           Zimbabwe       2023-03-04 264127   5668
## 2 Zimbabwe     <NA>           Zimbabwe       2023-03-05 264127   5668
## 3 Zimbabwe     <NA>           Zimbabwe       2023-03-06 264127   5668
## 4 Zimbabwe     <NA>           Zimbabwe       2023-03-07 264127   5668
## 5 Zimbabwe     <NA>           Zimbabwe       2023-03-08 264276   5671
## 6 Zimbabwe     <NA>           Zimbabwe       2023-03-09 264276   5671
```

```
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)
```

```
tail(global)
```

```
## # A tibble: 6 x 7
##   Province_State Country_Region date        cases deaths Population Combined_Key
##   <chr>          <chr>          <date>      <dbl>  <dbl>      <dbl> <chr>
## 1 <NA>           Zimbabwe       2023-03-04 264127   5668   14862927 Zimbabwe
## 2 <NA>           Zimbabwe       2023-03-05 264127   5668   14862927 Zimbabwe
## 3 <NA>           Zimbabwe       2023-03-06 264127   5668   14862927 Zimbabwe
## 4 <NA>           Zimbabwe       2023-03-07 264127   5668   14862927 Zimbabwe
## 5 <NA>           Zimbabwe       2023-03-08 264276   5671   14862927 Zimbabwe
## 6 <NA>           Zimbabwe       2023-03-09 264276   5671   14862927 Zimbabwe
```

We have the same problem of the US data, the possitive cases and the deaths are adding with the previous deaths. But in this case we are only interested in Peru, so we will start filtering it

```
Peru_df = global %>%
  filter(Country_Region == 'Peru')
head(Peru_df)
```

```
## # A tibble: 6 x 7
##   Province_State Country_Region date       cases deaths Population Combined_Key
##   <chr>          <chr>          <date>     <dbl>  <dbl>      <dbl> <chr>
```

```
## 1 <NA>           Peru           2020-03-06     1      1   32971846 Peru
## 2 <NA>           Peru           2020-03-07     1      1   32971846 Peru
## 3 <NA>           Peru           2020-03-08     6      2   32971846 Peru
## 4 <NA>           Peru           2020-03-09     7      2   32971846 Peru
## 5 <NA>           Peru           2020-03-10    11      2   32971846 Peru
## 6 <NA>           Peru           2020-03-11    11      2   32971846 Peru
```

Let's clean it a bit.

```
Peru_df = Peru_df %>%
  filter(!(is.nan(cases)), !(is.nan(deaths)))
head(Peru_df)
```

```
## # A tibble: 6 x 7
##   Province_State Country_Region date       cases deaths Population Combined_Key
##   <chr>          <chr>          <date>     <dbl>  <dbl>      <dbl> <chr>
## 1 <NA>           Peru           2020-03-06     1      1   32971846 Peru
## 2 <NA>           Peru           2020-03-07     1      1   32971846 Peru
## 3 <NA>           Peru           2020-03-08     6      2   32971846 Peru
## 4 <NA>           Peru           2020-03-09     7      2   32971846 Peru
## 5 <NA>           Peru           2020-03-10    11      2   32971846 Peru
## 6 <NA>           Peru           2020-03-11    11      2   32971846 Peru
```

Then getting the new cases and deaths.

```
Peru_df = Peru_df %>%
  arrange(date) %>%
  mutate(new_cases = cases - lag(cases),
         new_deaths = deaths - lag(deaths),
         previous_data = "Previous")
```

Visualizing Cases and Deaths in Peru

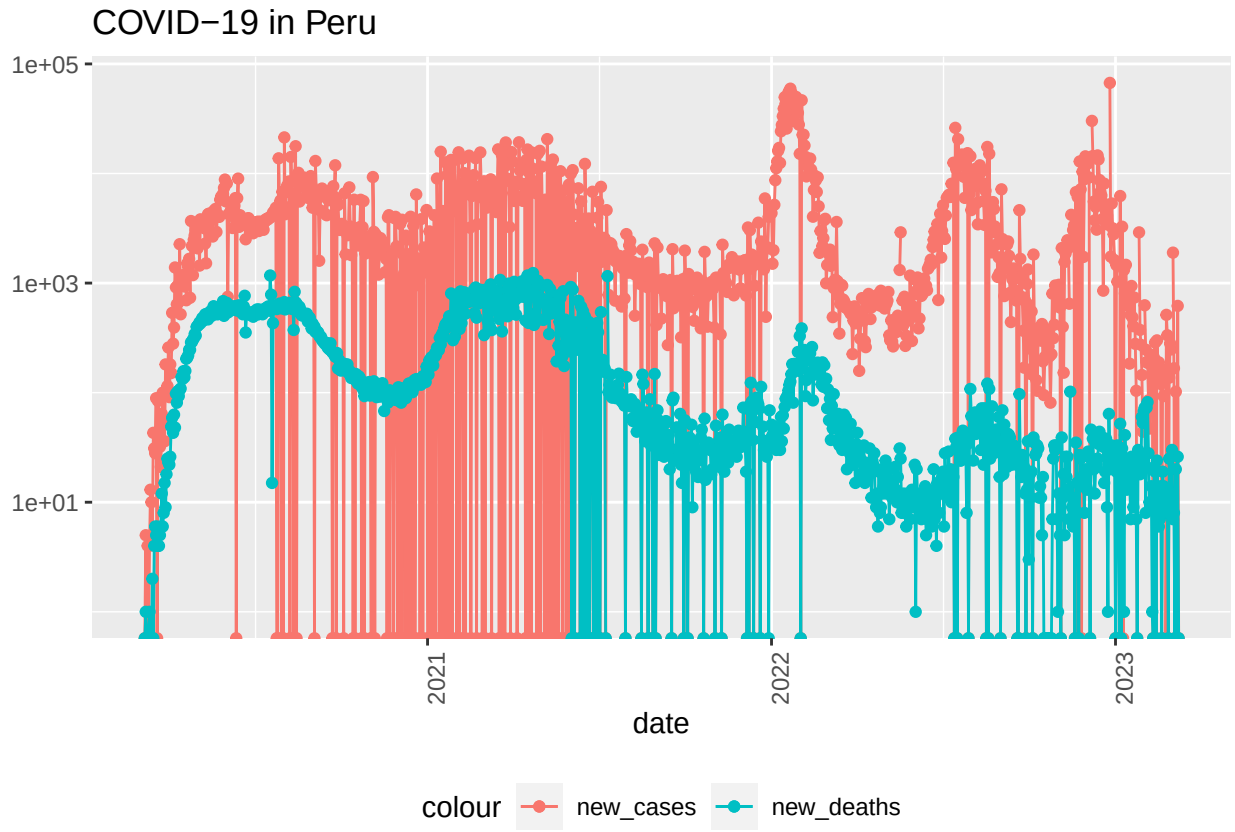
```
Peru_df %>%
  ggplot(aes(x = date, y = new_cases)) +
  geom_line(aes(color = "new_cases")) +
  geom_point(aes(color = "new_cases")) +
  geom_line(aes(y = new_deaths, color = "new_deaths")) +
  geom_point(aes(y = new_deaths, color = "new_deaths")) +
  scale_y_log10() +
  theme(legend.position = "bottom",
        axis.text.x = element_text(angle = 90)) +
  labs(title = "COVID-19 in Peru", 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 containing missing values (`geom_line()`).

## Warning: Removed 11 rows containing missing values (`geom_point()`).

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

## Warning: Removed 5 rows containing missing values (`geom_point()`).
```



In this case we dont need neither cases and deaths per thousands.

Now we need the previous seven days on each row.

```
Peru_df = Peru_df %>%
  mutate(n_deaths_1 = lag(new_deaths,1),
         n_deaths_2 = lag(new_deaths,2),
         n_deaths_3 = lag(new_deaths,3),
         n_deaths_4 = lag(new_deaths,4),
         n_deaths_5 = lag(new_deaths,5),
         n_deaths_6 = lag(new_deaths,6),
         n_deaths_7 = lag(new_deaths,7),
         n_cases_1 = lag(new_cases,1),
         n_cases_2 = lag(new_cases,2),
         n_cases_3 = lag(new_cases,3),
         n_cases_4 = lag(new_cases,4),
         n_cases_5 = lag(new_cases,5),
         n_cases_6 = lag(new_cases,6),
         n_cases_7 = lag(new_cases,7))
```

Training the COVID-19 cases and deaths models

```
mod_cases_Peru = lm(new_cases ~ n_deaths_1 +
                      n_deaths_2 +
                      n_deaths_3 +
                      n_deaths_4 +
                      n_deaths_5 +
                      n_deaths_6 +
                      n_deaths_7 +
                      n_cases_1 +
                      n_cases_2 +
                      n_cases_3 +
                      n_cases_4 +
                      n_cases_5 +
                      n_cases_6 +
                      n_cases_7, data = Peru_df)
summary(mod_cases_Peru)
```

```
##
## Call:
## lm(formula = new_cases ~ n_deaths_1 + n_deaths_2 + n_deaths_3 +
##     n_deaths_4 + n_deaths_5 + n_deaths_6 + n_deaths_7 + n_cases_1 +
##     n_cases_2 + n_cases_3 + n_cases_4 + n_cases_5 + n_cases_6 +
##     n_cases_7, data = Peru_df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -69395  -1060   -258    629  62131
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 313.471906 205.700763   1.524  0.12782
## n_deaths_1   -1.445856   1.575161  -0.918  0.35887
## n_deaths_2   -1.087851   1.574289  -0.691  0.48971
## n_deaths_3   -0.014648   1.541101  -0.010  0.99242
```

```
## n_deaths_4   -0.091149   1.547316  -0.059  0.95304
## n_deaths_5    2.250768   1.532631   1.469  0.14224
## n_deaths_6    1.038729   1.568362   0.662  0.50792
## n_deaths_7   -0.694919   1.567722  -0.443  0.65766
## n_cases_1    -0.008694   0.030751  -0.283  0.77744
## n_cases_2     0.231950   0.030253   7.667 3.93e-14 ***
## n_cases_3     0.160404   0.030965   5.180 2.65e-07 ***
## n_cases_4     0.158567   0.030955   5.122 3.57e-07 ***
## n_cases_5     0.082216   0.031046   2.648  0.00821 **
## n_cases_6     0.179158   0.030330   5.907 4.67e-09 ***
## n_cases_7     0.122605   0.030827   3.977 7.44e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4877 on 1076 degrees of freedom
##   (8 observations deleted due to missingness)
## Multiple R-squared:  0.5813, Adjusted R-squared:  0.5759
## F-statistic: 106.7 on 14 and 1076 DF,  p-value: < 2.2e-16
```

```
mod_deaths_Peru = lm(new_deaths ~ n_deaths_1 +
                       n_deaths_2 +
                       n_deaths_3 +
                       n_deaths_4 +
                       n_deaths_5 +
                       n_deaths_6 +
                       n_deaths_7 +
                       n_cases_1 +
                       n_cases_2 +
                       n_cases_3 +
                       n_cases_4 +
                       n_cases_5 +
                       n_cases_6 +
                       n_cases_7, data = Peru_df)
summary(mod_deaths_Peru)
```

```
##
## Call:
## lm(formula = new_deaths ~ n_deaths_1 + n_deaths_2 + n_deaths_3 +
##     n_deaths_4 + n_deaths_5 + n_deaths_6 + n_deaths_7 + n_cases_1 +
##     n_cases_2 + n_cases_3 + n_cases_4 + n_cases_5 + n_cases_6 +
##     n_cases_7, data = Peru_df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -968.89  -19.31   -1.50   18.86  759.29
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.140e+00  4.024e+00   0.283 0.777012
## n_deaths_1  -2.485e-01  3.081e-02  -8.064 1.95e-15 ***
## n_deaths_2   1.140e-01  3.079e-02   3.702 0.000225 ***
## n_deaths_3   2.766e-01  3.014e-02   9.177  < 2e-16 ***
## n_deaths_4   2.549e-01  3.027e-02   8.421  < 2e-16 ***
## n_deaths_5   2.439e-01  2.998e-02   8.137 1.11e-15 ***
```

```
## n_deaths_6   2.507e-01  3.068e-02   8.172 8.43e-16 ***
## n_deaths_7   8.969e-02  3.066e-02   2.925 0.003519 **
## n_cases_1   -5.307e-05  6.015e-04  -0.088 0.929705
## n_cases_2    7.874e-06  5.918e-04   0.013 0.989385
## n_cases_3    1.410e-03  6.057e-04   2.328 0.020113 *
## n_cases_4   -8.057e-04  6.055e-04  -1.331 0.183553
## n_cases_5   -7.776e-04  6.073e-04  -1.281 0.200645
## n_cases_6    1.029e-04  5.933e-04   0.173 0.862339
## n_cases_7    7.619e-04  6.030e-04   1.264 0.206680
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 95.39 on 1076 degrees of freedom
##   (8 observations deleted due to missingness)
## Multiple R-squared:  0.8662, Adjusted R-squared:  0.8645
## F-statistic: 497.6 on 14 and 1076 DF,  p-value: < 2.2e-16
```

Predicting next 100 days in COVID-19 Pandemy

First, let's make an early prediction of all Peru df.

```
Peru_df = tail(Peru_df, nrow(Peru_df) - 8)
Peru_df = Peru_df %>%
  mutate(predict_new_cases = predict(mod_cases_Peru),
         predict_new_deaths = predict(mod_deaths_Peru))
tail(Peru_df)
```

```
## # A tibble: 6 x 26
##   Province_State Country_Region date        cases deaths Population Combined_Key
##   <chr>          <chr>          <date>      <dbl>  <dbl>      <dbl> <chr>
## 1 <NA>           Peru           2023-03-04 4.49e6 219493   32971846 Peru
## 2 <NA>           Peru           2023-03-05 4.49e6 219493   32971846 Peru
## 3 <NA>           Peru           2023-03-06 4.49e6 219513   32971846 Peru
## 4 <NA>           Peru           2023-03-07 4.49e6 219513   32971846 Peru
## 5 <NA>           Peru           2023-03-08 4.49e6 219539   32971846 Peru
## 6 <NA>           Peru           2023-03-09 4.49e6 219539   32971846 Peru
## # i 19 more variables: new_cases <dbl>, new_deaths <dbl>, previous_data <chr>,
## #   n_deaths_1 <dbl>, n_deaths_2 <dbl>, n_deaths_3 <dbl>, n_deaths_4 <dbl>,
## #   n_deaths_5 <dbl>, n_deaths_6 <dbl>, n_deaths_7 <dbl>, n_cases_1 <dbl>,
## #   n_cases_2 <dbl>, n_cases_3 <dbl>, n_cases_4 <dbl>, n_cases_5 <dbl>,
## #   n_cases_6 <dbl>, n_cases_7 <dbl>, predict_new_cases <dbl>,
## #   predict_new_deaths <dbl>
```

Getting the iterator of each column

```
i = as.integer(1)
for(col in names(Peru_df)){
  print(paste(as.character(i), " " , col))
  i = i + 1
}
```

```
## [1] "1   Province_State"
```

```
## [1] "2   Country_Region"
## [1] "3   date"
## [1] "4   cases"
## [1] "5   deaths"
## [1] "6   Population"
## [1] "7   Combined_Key"
## [1] "8   new_cases"
## [1] "9   new_deaths"
## [1] "10   previous_data"
## [1] "11   n_deaths_1"
## [1] "12   n_deaths_2"
## [1] "13   n_deaths_3"
## [1] "14   n_deaths_4"
## [1] "15   n_deaths_5"
## [1] "16   n_deaths_6"
## [1] "17   n_deaths_7"
## [1] "18   n_cases_1"
## [1] "19   n_cases_2"
## [1] "20   n_cases_3"
## [1] "21   n_cases_4"
## [1] "22   n_cases_5"
## [1] "23   n_cases_6"
## [1] "24   n_cases_7"
## [1] "25   predict_new_cases"
## [1] "26   predict_new_deaths"
```

```
for(i in 1:100){
  tail_Peru_df = tail(Peru_df, 10)
  temp_df = tail(tail_Peru_df, 1)

  temp_df[10][1] = "Predictor"
  head(temp_df)
  temp_df[3][1] = max(Peru_df$date) + days(1)
  temp_df[8][1] = temp_df[25][1]
  temp_df[9][1] = temp_df[26][1]
  ## Concating
  temp_df_Peru = rbind(tail_Peru_df, temp_df)
  ## formating
  temp_df_Peru = temp_df_Peru %>%
    mutate(n_deaths_1 = lag(new_deaths,1),
          n_deaths_2 = lag(new_deaths,2),
          n_deaths_3 = lag(new_deaths,3),
          n_deaths_4 = lag(new_deaths,4),
          n_deaths_5 = lag(new_deaths,5),
          n_deaths_6 = lag(new_deaths,6),
          n_deaths_7 = lag(new_deaths,7),
          n_cases_1 = lag(new_cases,1),
          n_cases_2 = lag(new_cases,2),
          n_cases_3 = lag(new_cases,3),
          n_cases_4 = lag(new_cases,4),
          n_cases_5 = lag(new_cases,5),
          n_cases_6 = lag(new_cases,6),
          n_cases_7 = lag(new_cases,7))
  ## Getting the new last row again
```

```
  tail_of_the_first_Peru_df = tail(temp_df_Peru, 1)
  tail(tail_of_the_first_Peru_df)
  tail_of_the_first_Peru_df = tail_of_the_first_Peru_df %>%
                    mutate(predict_new_cases = predict(mod_cases_Peru, tail_of_the_first_Peru_df),
                           predict_new_deaths = predict(mod_deaths_Peru, tail_of_the_first_Peru_df))
  ## pasting to the Peru_df
  Peru_df = rbind(Peru_df, tail_of_the_first_Peru_df)
}
```

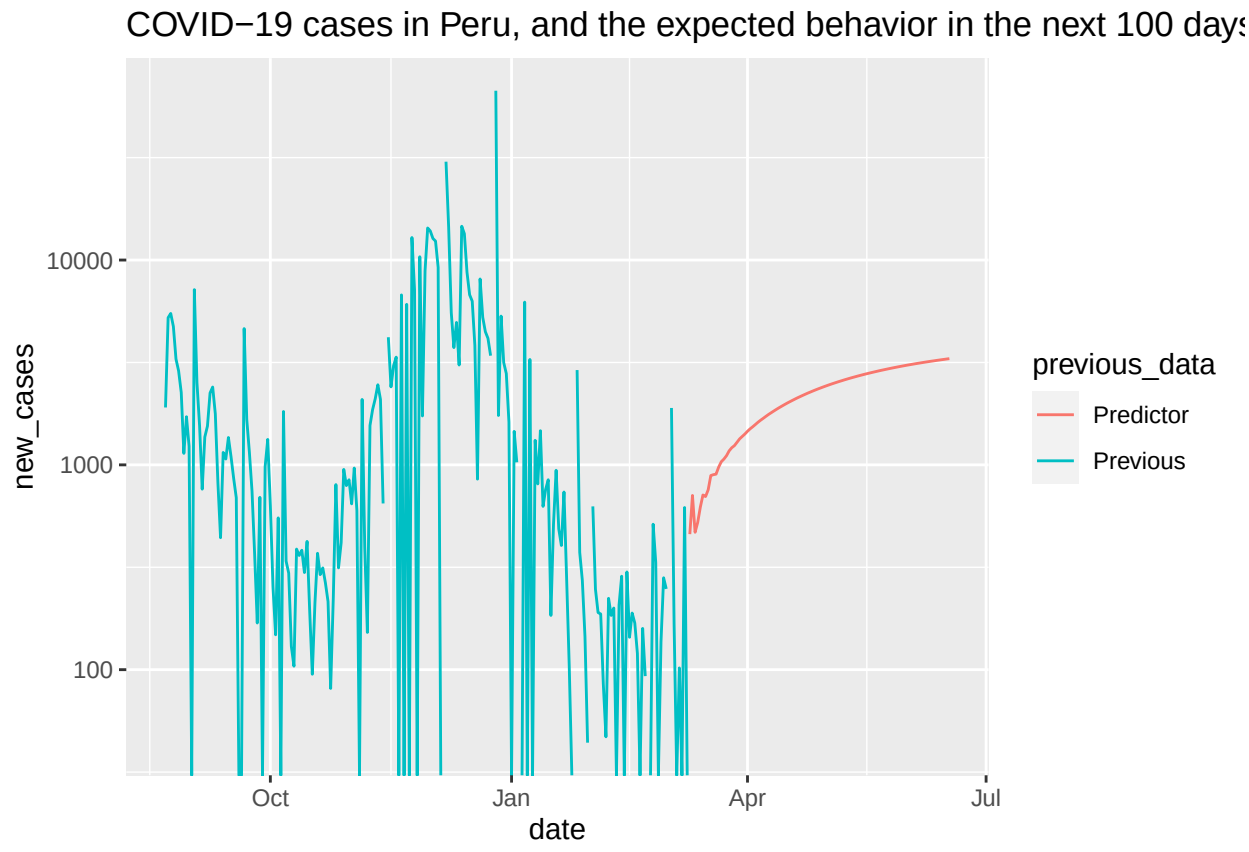
Visualizing the predictions and the data in the last 300 days

First with cases:

```
tail(Peru_df,300) %>% ggplot(aes(x=date, y = new_cases, color = previous_data)) +
  geom_line() +
  scale_y_log10() +
  ggtitle("COVID-19 cases in Peru, and the expected behavior in the next 100 days")
```

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

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

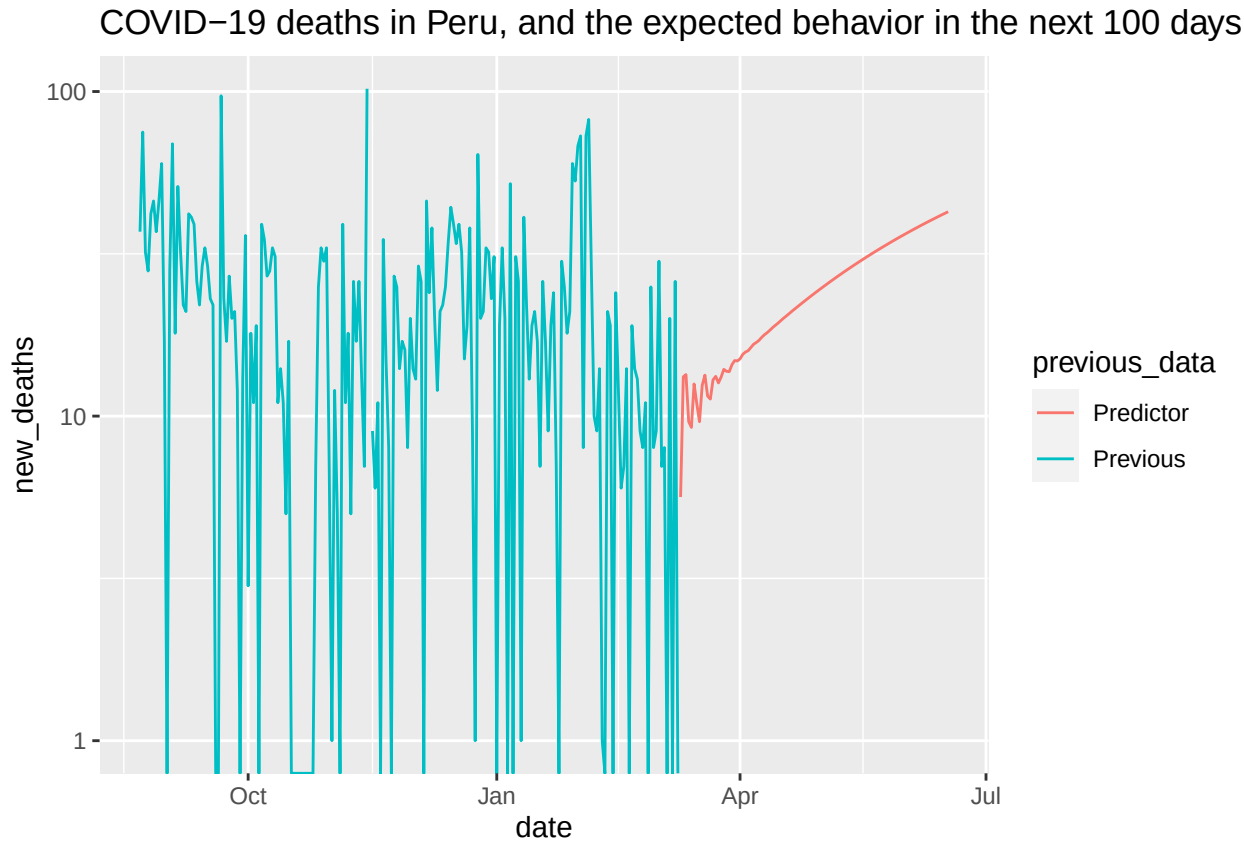


And then with deaths.

```
tail(Peru_df,300) %>% ggplot(aes(x=date, y = new_deaths, color = previous_data)) +
  geom_line() +
  scale_y_log10() +
  ggtitle("COVID-19 deaths in Peru, and the expected behavior in the next 100 days")
```

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

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



That's all folks!