

# COVID-19 Data Analysis

2024-04-04

## Introduction:

This analysis focuses on COVID-19 data sourced from the Johns Hopkins GitHub repository. These datasets provide daily updates on the total number of confirmed COVID-19 cases and deaths for each country worldwide, as well as for each state in the United States. The datasets can be found at the following URL: [https://github.com/CSSEGISandData/COVID-19/tree/master/csse\\_covid\\_19\\_data/csse\\_covid\\_19\\_time\\_series](https://github.com/CSSEGISandData/COVID-19/tree/master/csse_covid_19_data/csse_covid_19_time_series).

## Libraries

```
library(tidyverse)
library(lubridate)
library(dplyr)
library(ggplot2)
library(readr)
```

## Importing Dataset and reading data from csv file

First, we need to import the required dataset from the public repository from the following source: [https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse\\_covid\\_19\\_data/csse\\_covid\\_19\\_time\\_series/](https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_time_series/)

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

file_names <- c("time_series_covid19_confirmed_global.csv", "time_series_covid19_deaths_global.csv", "time_series_covid19_deaths_US.csv")

urls <- str_c(url_in, file_names)
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])
US_deaths <- read_csv(urls[4])
```

## TIDY AND TRANSFORM

Let us tidy the dataset and transform it into useful form so that we can use it for our further data analysis.

```
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)'
```

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

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

uid_lookup_url <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/
  covid_19_data/jsontable/uid_lookup.json"

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.
```

```
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
```

```
## # A tibble: 306,827 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
## # i 306,817 more rows
```

```
summary(global)
```

```
## Province_State      Country_Region      date      cases
## Length:306827      Length:306827      Min.   :2020-01-22      Min.   :      1
## Class :character    Class :character    1st Qu.:2020-12-12      1st Qu.:    1316
## Mode  :character    Mode  :character    Median :2021-09-16      Median :   20365
##                               Mean  :2021-09-11      Mean  :  1032863
##                               3rd Qu.:2022-06-15      3rd Qu.:  271281
##                               Max.   :2023-03-09      Max.   :103802702
##
## deaths              Population      Combined_Key
## Min.   :      0      Min.   :6.700e+01      Length:306827
## 1st Qu.:      7      1st Qu.:7.866e+05      Class :character
## Median :   214      Median :6.948e+06      Mode  :character
## Mean   :  14405      Mean   :2.890e+07
## 3rd Qu.:   3665      3rd Qu.:2.914e+07
## Max.   :1123836      Max.   :1.380e+09
##                               NA's   :6729
```

## Removing character “X” and converting date to date data type

Now we can convert the date to a date data type in the US data set, removing the unnecessarily long “X” character from the date. Also, we can rename the column as per our needs.

```
US_cases <- US_cases %>% pivot_longer(cols = -(UID:Combined_Key),
  names_to = "date",
  values_to = "cases") %>%
  select(Admin2:cases) %>%
```

```

select(-c(Lat, Long_))

US_deaths <- US_deaths %>% pivot_longer(cols = -(UID:Population),
                                       names_to = "date",
                                       values_to = "deaths") %>%

select(Admin2:deaths) %>%
select(-c(Lat, Long_))

US_cases$date <- gsub("X", "", as.character(US_cases$date))
US_deaths$date <- gsub("X", "", as.character(US_deaths$date))

US <- US_cases %>% full_join(US_deaths) %>%
mutate(date = mdy(date))

```

## Organizing the data set

Now we can organize our dataset as per our analysis by grouping all the countries in a new dataframe.

```
global_country <- global %>% group_by(Province_State, Country_Region, date) %>% summarize(cases = sum(c
```

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

```
global_country
```

```
## # A tibble: 306,827 x 7
##   Province_State Country_Region date      cases deaths deaths_per_mill
##   <chr>          <chr>      <date>    <dbl>  <dbl>         <dbl>
## 1 Alberta      Canada    2020-03-06      1      0             0
## 2 Alberta      Canada    2020-03-07      2      0             0
## 3 Alberta      Canada    2020-03-08      4      0             0
## 4 Alberta      Canada    2020-03-09      7      0             0
## 5 Alberta      Canada    2020-03-10      7      0             0
## 6 Alberta      Canada    2020-03-11     19      0             0
## 7 Alberta      Canada    2020-03-12     19      0             0
## 8 Alberta      Canada    2020-03-13     29      0             0
## 9 Alberta      Canada    2020-03-14     29      0             0
## 10 Alberta     Canada    2020-03-15     39      0             0
## # i 306,817 more rows
## # i 1 more variable: Population <dbl>
```

```
total_all <- global_country %>%
  group_by(Country_Region) %>%
  summarize(deaths = max(deaths), cases = max(cases), population = max(Population), cases_per_thou = 1000000)

total_all %>% slice_min(deaths_per_thou, n = 10)
```

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

##	1	Holy See	0	29	809	35.8	0
##	2	Tuvalu	0	2828	11792	240.	0
##	3	Korea, North	6	1	25778815	0.0000388	0.000233
##	4	Burundi	38	53631	11890781	4.51	0.00320
##	5	Chad	194	7679	16425859	0.467	0.0118
##	6	South Sudan	138	18368	11193729	1.64	0.0123
##	7	Niger	315	9508	24206636	0.393	0.0130
##	8	Tajikistan	125	17786	9537642	1.86	0.0131
##	9	Benin	163	27999	12123198	2.31	0.0134
##	10	Tanzania	846	42906	59734213	0.718	0.0142

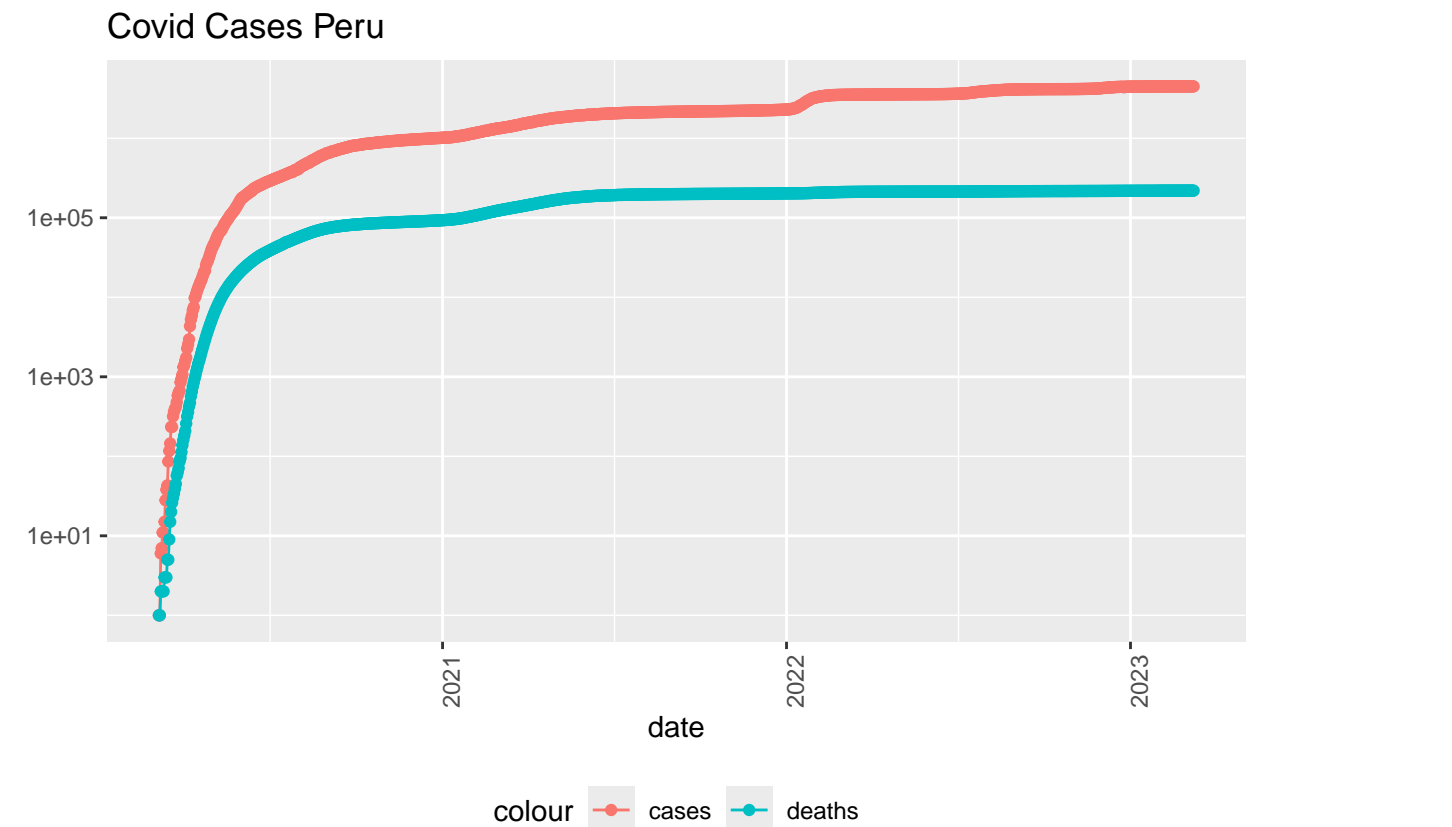
## VISUALIZATION

Data visualization is the graphical representation of information and data. First, I would like to visualize the COVID cases in each country with respect to the date.

### Covid Cases in each Country

The graph below shows the rapid rise in COVID cases in the country of Peru. Also, it shows the death rate with respect to the year.

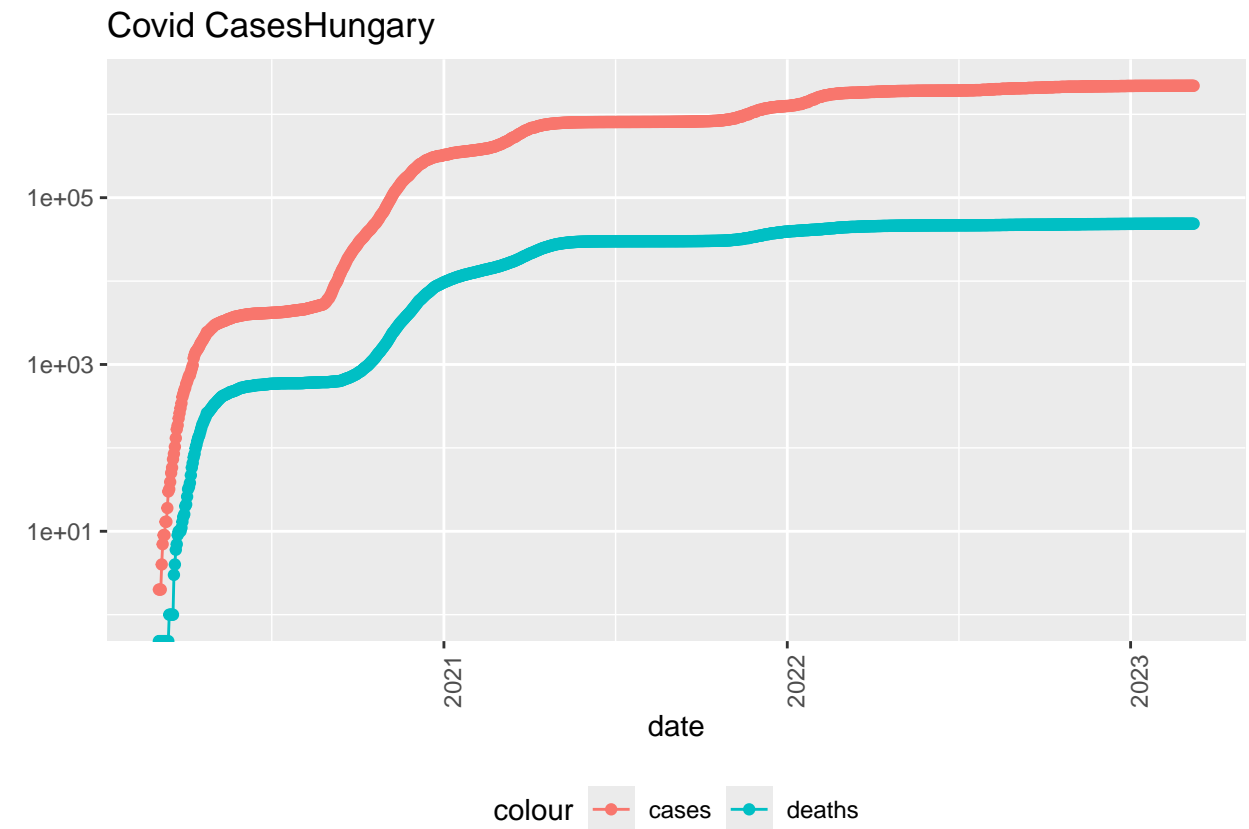
```
country <- "Peru"
global_country %>% filter(Country_Region == country) %>% filter(cases > 0) %>% ggplot(aes(x=date, y=cases))
```



Similarly, we can see the cases and deaths in Hungary and Slovakia. With this curve line, we can predict the increase and decrease in COVID cases and deaths.

```
country <- "Hungary"
global_country %>% filter(Country_Region == country) %>% filter(cases > 0) %>% ggplot(aes(x=date, y=cas
```

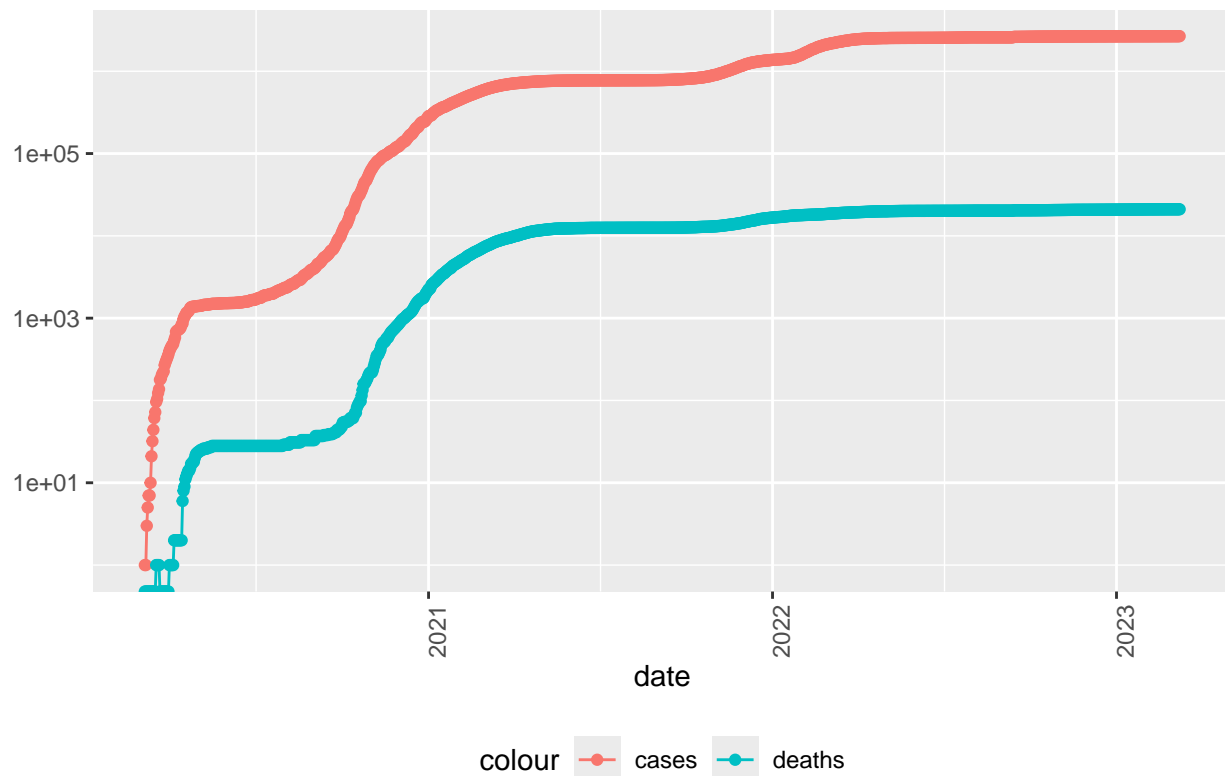
```
## Warning in scale_y_log10(): log-10 transformation introduced infinite values.
## log-10 transformation introduced infinite values.
```



```
country <- "Slovakia"
global_country %>% filter(Country_Region == country) %>% filter(cases > 0) %>% ggplot(aes(x=date, y=cas
```

```
## Warning in scale_y_log10(): log-10 transformation introduced infinite values.
## log-10 transformation introduced infinite values.
```

## Covid Cases in Slovakia

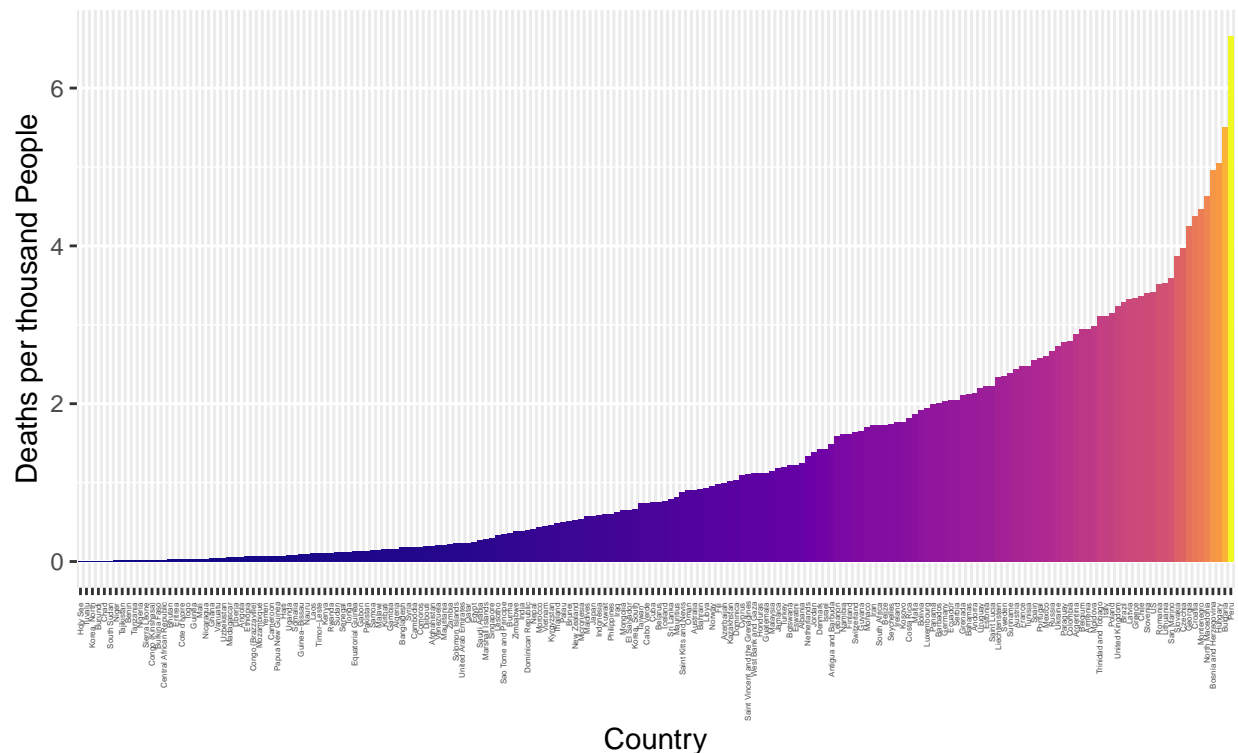


## COVID Deaths per thousand People by Country

The below bar chart shows the COVID deaths per thousand people in each country. The below bar chart shows the COVID deaths per thousand people in each country. With this chart, we can visualize the deaths that occurred in each country.

```
ggplot(data = total_all, aes(x = reorder(Country_Region, +deaths_per_thou),
                                     y = deaths_per_thou,
                                     fill = deaths_per_thou)) +
  scale_fill_viridis_c(option = "plasma") +
  geom_bar(stat = "identity") +
  labs(x = "Country",
       y = "Deaths per thousand People",
       title = "COVID Deaths per thousand People by Country",
       subtitle = "") +
  theme(legend.position="none", axis.text.x=element_text(angle=90, hjust=.98, vjust = .5, size=3))
```

## COVID Deaths per thousand People by Country



## COVID cases in each US state

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

Let us visualize the COVID cases that occurred in each state in the US from the US dataset. The below graphs show the number of cases and deaths in each US state.

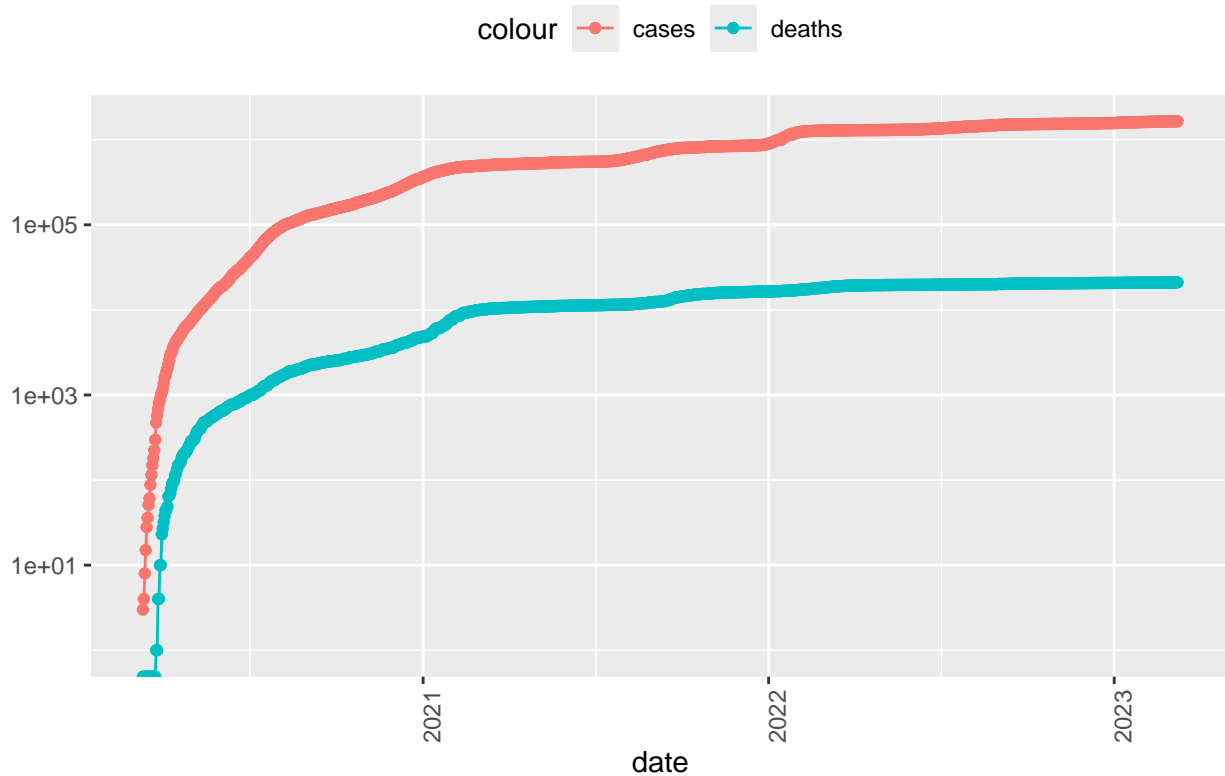
```
state <- "Alabama"
US_by_state %>% filter(Province_State == state) %>% filter(cases > 0) %>%
  ggplot(aes(x = date, y = cases)) +
  geom_line(aes(color = "cases")) +
  geom_point(aes(color = "cases")) +
  geom_line(aes(y = deaths, color = "deaths")) +
  geom_point(aes(y = deaths, color = "deaths")) +
  scale_y_log10() +
  theme(legend.position = "top",
        axis.text.x = element_text(angle = 90)) +
  labs(title = str_c("Covid Cases in US state ", state), y = NULL)
```

```
## Warning in scale_y_log10(): log-10 transformation introduced infinite values.
```



```
## log-10 transformation introduced infinite values.
```

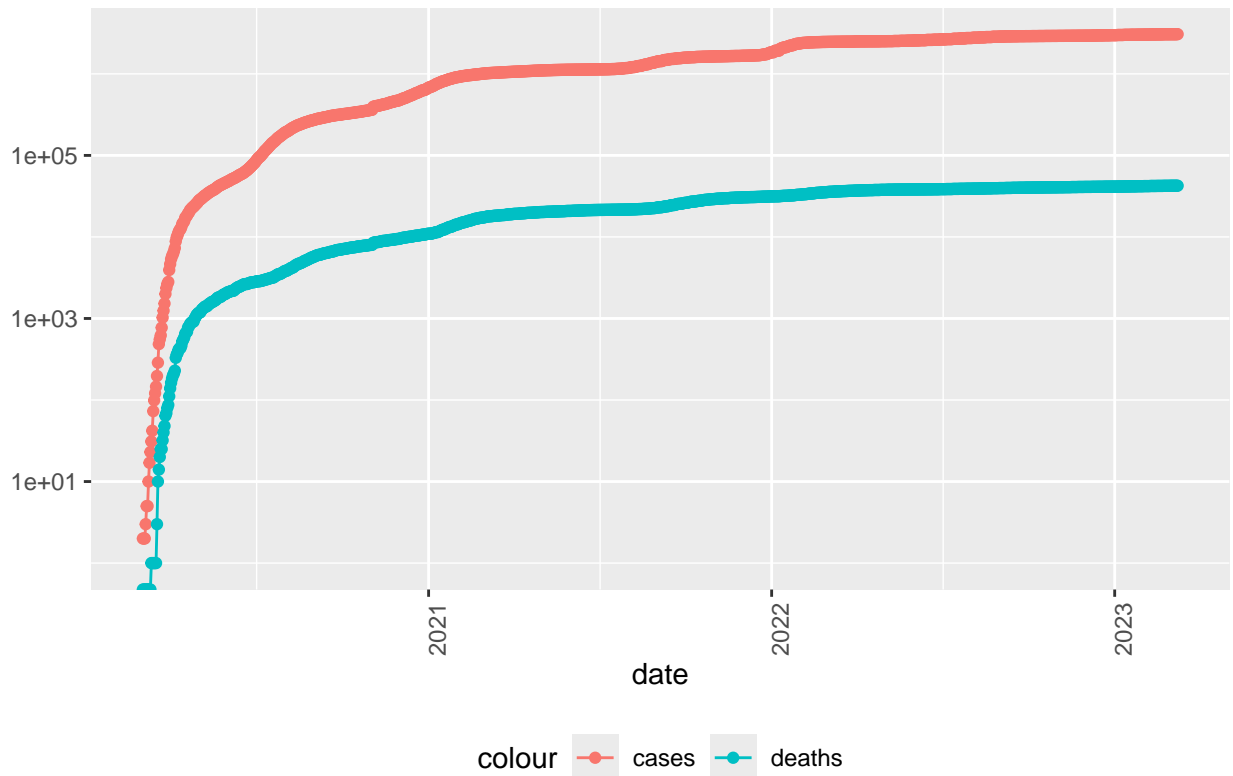
## Covid Cases in US state Alabama



```
state <- "Georgia"
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("Covid Cases in US state ", state), y = NULL)
```

```
## Warning in scale_y_log10(): log-10 transformation introduced infinite values.
## log-10 transformation introduced infinite values.
```

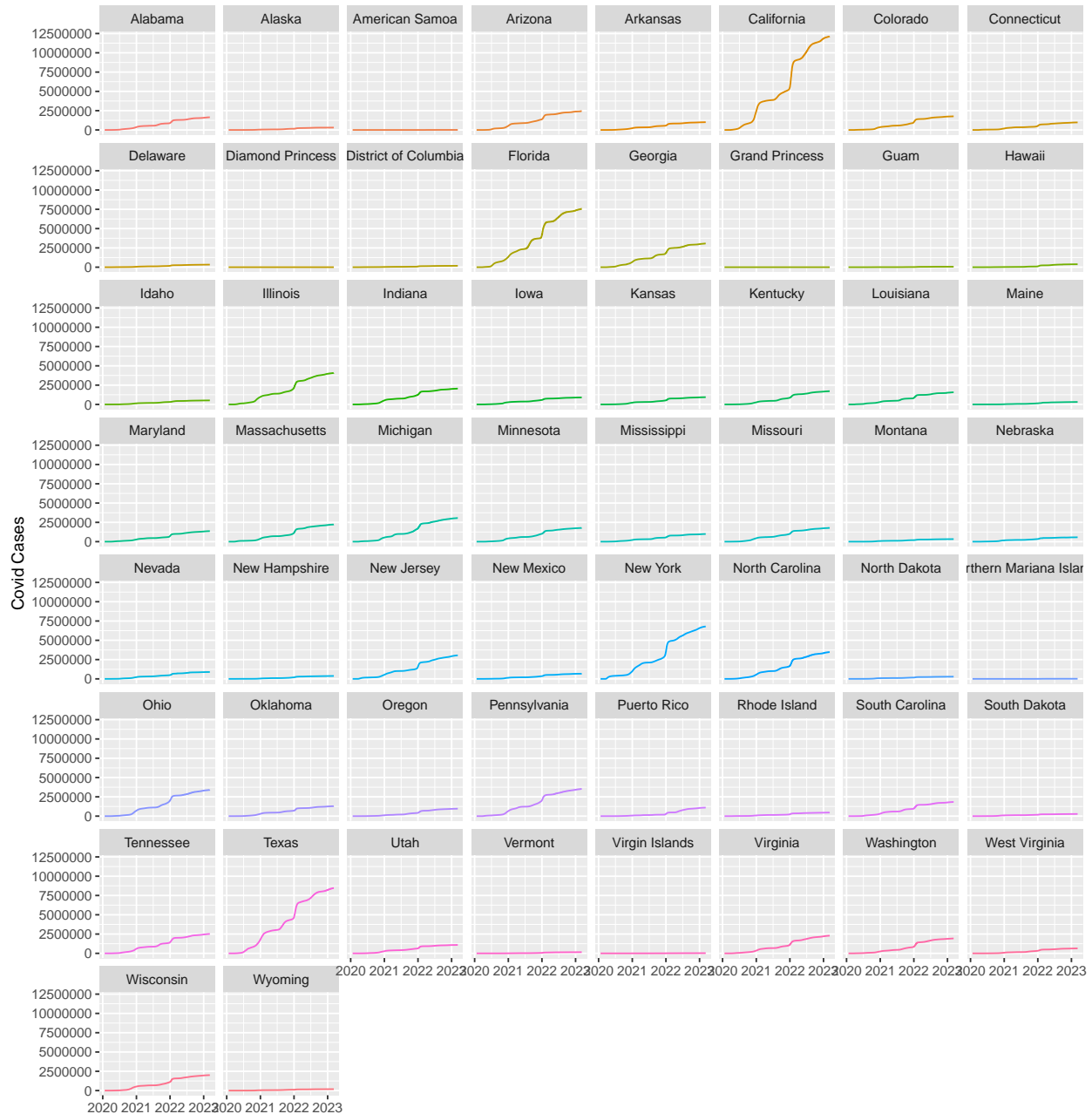
## Covid Cases in US state Georgia



Here, we can see all the US state COVID cases together. This graph shows the number of cases in each year in each state.

```
ggplot(data = US_by_state, aes(x = date, y = cases,
                                color = Province_State)) +
  geom_line() +
  facet_wrap(~Province_State) +
  labs(x = "", y = "Covid Cases", title = "COVID-19 Cases in the US",
        subtitle = "By Each state in US") +
  theme(legend.position="none")
```

## COVID-19 Cases in the US By Each state in US



## ANALYSING AND MODELING

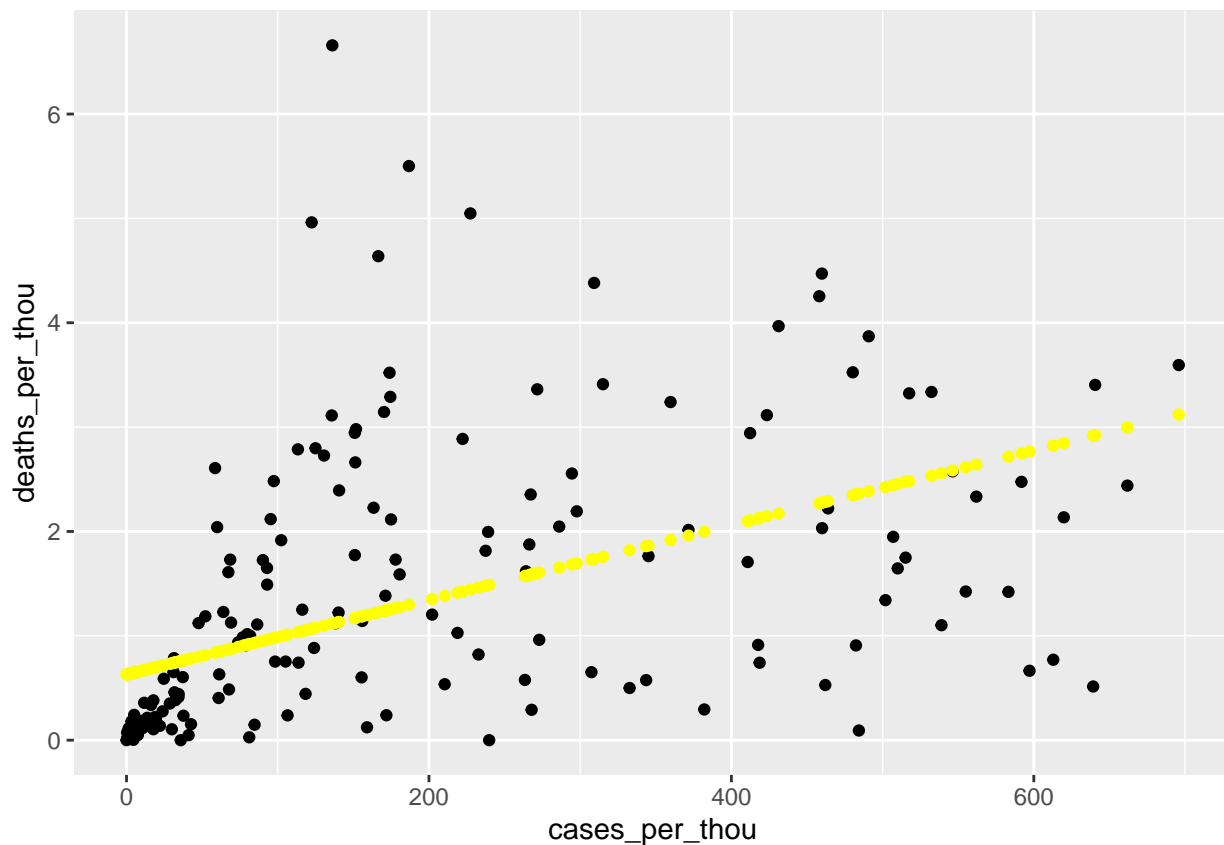
Let us now delve deeper into the data set by comparing COVID cases per thousand with respect to deaths per thousand. To do so, I used a linear model to show the correlation between them. The below model roughly follows the pandemic trend in the region. The yellow curve shows the observed deaths per thousand cases. The black one represents the linear model's estimate of cases per thousand.

```
mod <- lm(deaths_per_thou ~ cases_per_thou, data = total_all)
summary(mod)
```

```
##
## Call:
## lm(formula = deaths_per_thou ~ cases_per_thou, data = total_all)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.4034 -0.6103 -0.3892  0.4718  5.5419
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.6292479   0.1116264     5.637 6.10e-08 ***
## cases_per_thou 0.0035799   0.0004383     8.167 4.13e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.151 on 192 degrees of freedom
## Multiple R-squared:  0.2578, Adjusted R-squared:  0.254
## F-statistic: 66.7 on 1 and 192 DF, p-value: 4.132e-14
```

```
x_grid <- seq(1, 151)
new_df <- tibble(cases_per_thou = x_grid)
total_pred <- total_all %>% mutate(pred = predict(mod))
```

```
total_pred %>% ggplot() +
  geom_point(aes(x = cases_per_thou, y = deaths_per_thou), color = "black") +
  geom_point(aes(x = cases_per_thou, y = pred), color = "yellow")
```



## BIAS IDENTIFICATION:

Aggregating data from various sources introduces variability in quality, accuracy, and timeliness. The reliability of each source impacts the overall dataset. Also Inconsistent reporting of COVID cases and deaths across regions can skew the data. Varying definitions of COVID-related deaths (with or without comorbidities) add complexity. Confirmed cases may not capture all infections due to testing limitations or individual choices.

Personal Bias: Researchers and data collectors may unintentionally introduce bias based on their perspectives. Every recorded death may be directly attributable to COVID-19. Some deaths could be due to other factors, and distinguishing them accurately can be challenging. Undiagnosed or asymptomatic cases likely went unreported. This introduces uncertainty about the actual prevalence of COVID-19.

## CONCLUSION

The data John Hopkins provides is aggregated from several different sources. It is very likely that the quality, accuracy, and timeliness of the data vary depending on the source. The analysis performed on the COVID-19 data provides valuable insights into the spread and impact of the pandemic across the globe. The visualizations and summaries generated help identify countries with the highest number of cases, deaths, and recoveries, as well as the trend in active cases over time. The visualizations and summaries generated help identify countries with the highest number of cases, deaths, and recoveries, as well as the trend in active cases over time. The global ratio of recoveries to cases indicates improving medical treatment and resilience. Despite alarming overall case and death numbers, the leveling out of cases per thousand in some countries offers hope.

```
sessionInfo()
```

```
## R version 4.3.3 (2024-02-29 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 11 x64 (build 22631)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United States.utf8
## [2] LC_CTYPE=English_United States.utf8
## [3] LC_MONETARY=English_United States.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.utf8
##
## time zone: America/Los_Angeles
## tzcode source: internal
##
## attached base packages:
## [1] stats      graphics  grDevices  utils      datasets  methods    base
##
## other attached packages:
## [1] lubridate_1.9.3 forcats_1.0.0  stringr_1.5.1  dplyr_1.1.4
## [5] purrr_1.0.2    readr_2.1.5    tidyr_1.3.1    tibble_3.2.1
## [9] ggplot2_3.5.0  tidyverse_2.0.0
##
## loaded via a namespace (and not attached):
```

## [1]	utf8_1.2.4	generics_0.1.3	stringi_1.8.3	hms_1.1.3
## [5]	digest_0.6.35	magrittr_2.0.3	evaluate_0.23	grid_4.3.3
## [9]	timechange_0.3.0	fastmap_1.1.1	fansi_1.0.6	viridisLite_0.4.2
## [13]	scales_1.3.0	cli_3.6.2	rlang_1.1.3	crayon_1.5.2
## [17]	bit64_4.0.5	munsell_0.5.0	withr_3.0.0	yaml_2.3.8
## [21]	tools_4.3.3	parallel_4.3.3	tzdb_0.4.0	colorspace_2.1-0
## [25]	curl_5.2.1	vctrs_0.6.5	R6_2.5.1	lifecycle_1.0.4
## [29]	bit_4.0.5	vroom_1.6.5	pkgconfig_2.0.3	pillar_1.9.0
## [33]	gtable_0.3.4	glue_1.7.0	xfun_0.42	tidyselect_1.2.1
## [37]	highr_0.10	rstudioapi_0.15.0	knitr_1.45	farver_2.1.1
## [41]	htmltools_0.5.7	rmarkdown_2.26	labeling_0.4.3	compiler_4.3.3