COVID-19

2024-03-03

Problem

With COVID-19 case and deaths numbers on the rise, we would like to analyze countries with the smallest and the biggest population.

Data Description

The data used for the analysis is COVID19 dataset from the Johns Hopkins Github site.

Import Data

The data is initially imported allowing it to be analyzed.

```
## Get current data in the four files
url_in<-"https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid
file_names<- c("time_series_covid19_confirmed_US.csv",
               "time_series_covid19_confirmed_global.csv",
               "time_series_covid19_deaths_US.csv",
               "time_series_covid19_deaths_global.csv",
               "time_series_covid19_recovered_global.csv")
urls<-str_c(url_in, file_names)
uid_lookup_url <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/
```

Read csv and save datasets.

```
global_cases_raw<-read_csv(urls[2])
global_deaths_raw <- read_csv(urls[4])
US_cases_raw <- read.csv(urls[1])
US_deaths_raw <- read.csv(urls[3])
```

Tidy Data

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

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

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)

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

head(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>           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
```

```
US_cases <- US_cases_raw %>%
  pivot_longer(cols = -c('UID':'Combined_Key'),
               names_to = "date",
               values_to = "cases") %>%
  select(Admin2:cases) %>%
  mutate(date = mdy(gsub("X", "", date))) %>%
  select(-c(Lat, Long_))

US_deaths <- US_deaths_raw %>%
  pivot_longer(cols = -(UID:Population),
               names_to = "date",
               values_to = "deaths") %>%
  select(Admin2:deaths) %>%
  mutate(date = mdy(gsub("X", "", date))) %>%
  select(-c(Lat, Long_))

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

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

```
tail(US)
```

```
## # A tibble: 6 x 8
##   Admin2 Province_State Country_Region Combined_Key  date       cases Population
##   <chr>  <chr>          <chr>          <chr>         <date>     <int>      <int>
## 1 Weston Wyoming        US             Weston, Wyom~ 2023-03-04  1905       6927
## 2 Weston Wyoming        US             Weston, Wyom~ 2023-03-05  1905       6927
## 3 Weston Wyoming        US             Weston, Wyom~ 2023-03-06  1905       6927
## 4 Weston Wyoming        US             Weston, Wyom~ 2023-03-07  1905       6927
## 5 Weston Wyoming        US             Weston, Wyom~ 2023-03-08  1905       6927
## 6 Weston Wyoming        US             Weston, Wyom~ 2023-03-09  1905       6927
## # i 1 more variable: deaths <int>
```

I would like to analyze cases and deaths in different countries, so let's sum total population of each country, find percentage of cases and deaths per population.

```
global_sum <- global %>%
  filter(Population > 0) %>%
  group_by(Country_Region, date) %>%
  summarise(Total_Population = sum(Population), Total_Cases = sum(cases), Total_Deaths = sum(deaths)) %
  mutate(Pers_Cases_of_Pop = Total_Cases*100/Total_Population, Pers_Deaths_of_Pop = Total_Deaths*100/To
```

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

```
head(global_sum)
```

```
## # A tibble: 6 x 7
## # Groups:   Country_Region [1]
```

```
##   Country_Region date       Total_Population Total_Cases Total_Deaths
##   <chr>          <date>                <dbl>       <dbl>        <dbl>
## 1 Afghanistan    2020-02-24         38928341           5            0
## 2 Afghanistan    2020-02-25         38928341           5            0
## 3 Afghanistan    2020-02-26         38928341           5            0
## 4 Afghanistan    2020-02-27         38928341           5            0
## 5 Afghanistan    2020-02-28         38928341           5            0
## 6 Afghanistan    2020-02-29         38928341           5            0
## # i 2 more variables: Pers_Cases_of_Pop <dbl>, Pers_Deaths_of_Pop <dbl>
```

Find country with the smallest population.

```
min_population <- global_sum[global_sum$Total_Population == min(global_sum$Total_Population),]
min_population
```

```
## # A tibble: 1,099 x 7
## # Groups:   Country_Region [1]
##    Country_Region date       Total_Population Total_Cases Total_Deaths
##    <chr>          <date>                <dbl>       <dbl>        <dbl>
##  1 Holy See       2020-03-06              809           1            0
##  2 Holy See       2020-03-07              809           1            0
##  3 Holy See       2020-03-08              809           1            0
##  4 Holy See       2020-03-09              809           1            0
##  5 Holy See       2020-03-10              809           1            0
##  6 Holy See       2020-03-11              809           1            0
##  7 Holy See       2020-03-12              809           1            0
##  8 Holy See       2020-03-13              809           1            0
##  9 Holy See       2020-03-14              809           1            0
## 10 Holy See       2020-03-15              809           1            0
## # i 1,089 more rows
## # i 2 more variables: Pers_Cases_of_Pop <dbl>, Pers_Deaths_of_Pop <dbl>
```

The result is **Holy See**.

Find country with the largest population.

```
max_population <- global_sum[global_sum$Total_Population == max(global_sum$Total_Population),]
max_population
```

```
## # A tibble: 1,135 x 7
## # Groups:   Country_Region [1]
##    Country_Region date       Total_Population Total_Cases Total_Deaths
##    <chr>          <date>                <dbl>       <dbl>        <dbl>
##  1 China          2020-01-30       1417925054        8141          171
##  2 China          2020-01-31       1417925054        9802          213
##  3 China          2020-02-01       1417925054       11891          259
##  4 China          2020-02-02       1417925054       16630          361
##  5 China          2020-02-03       1417925054       19716          425
##  6 China          2020-02-04       1417925054       23707          491
##  7 China          2020-02-05       1417925054       27440          563
##  8 China          2020-02-06       1417925054       30587          633
##  9 China          2020-02-07       1417925054       34110          718
## 10 China          2020-02-08       1417925054       36814          805
```

```
## # i 1,125 more rows
## # i 2 more variables: Pers_Cases_of_Pop <dbl>, Pers_Deaths_of_Pop <dbl>
```

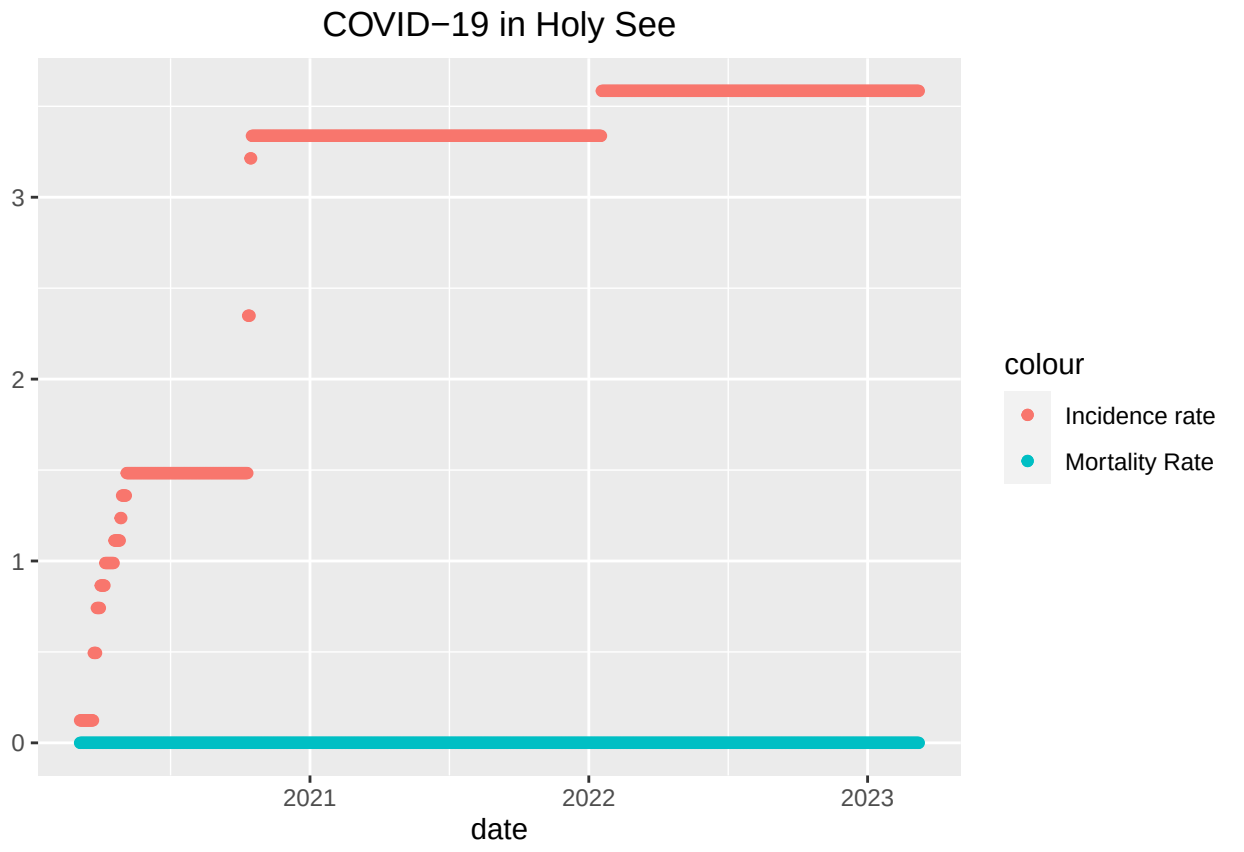
The result is **China**.

Visualization

Visualize how COVID-19 was going in Holy See (Vatican City).

Given the small size of Holy See, it might be challenging to analyze COVID-19 cases and deaths using a logarithmic scale. Logarithmic scales are effective for visualizing trends when dealing with large datasets. So I will use raw data to visualize.

```
ggplot(min_population, aes(x=date, y=Pers_Cases_of_Pop)) +
  geom_point(aes(color = "Incidence rate")) +
  geom_point(aes(y = Pers_Deaths_of_Pop, color = "Mortality Rate")) +
  theme(plot.title = element_text(hjust = 0.5)) +
  labs(title = "COVID-19 in Holy See", y = NULL)
```

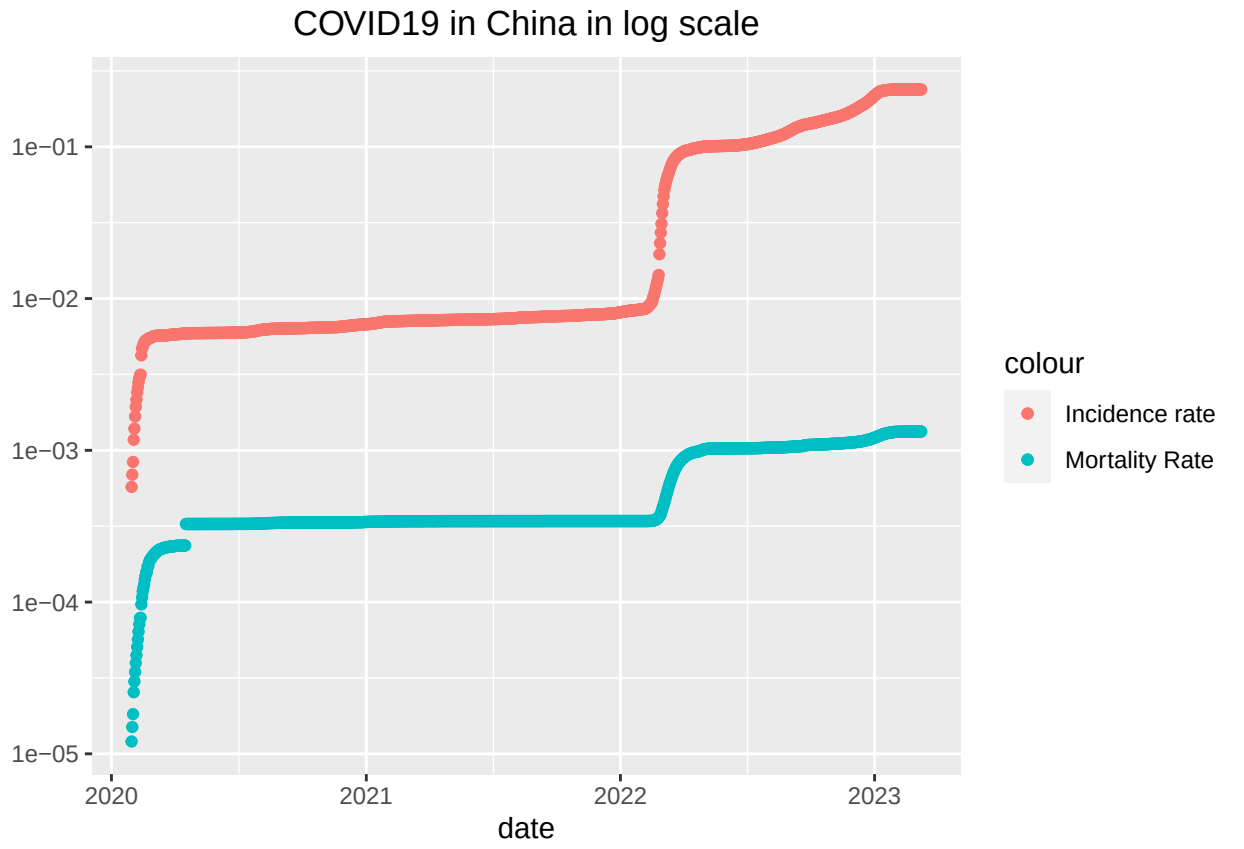


The small population size of Holy See makes it difficult to draw statistically significant conclusions from a limited number of cases. Strict measures implemented by the Holy See, including early lockdowns and vaccinations for the Pope and top officials, likely contributed to the low number of cases. And as we can see, there are no deaths were detected because of COVID-19 virus.

Visualize how COVID-19 was going in China.

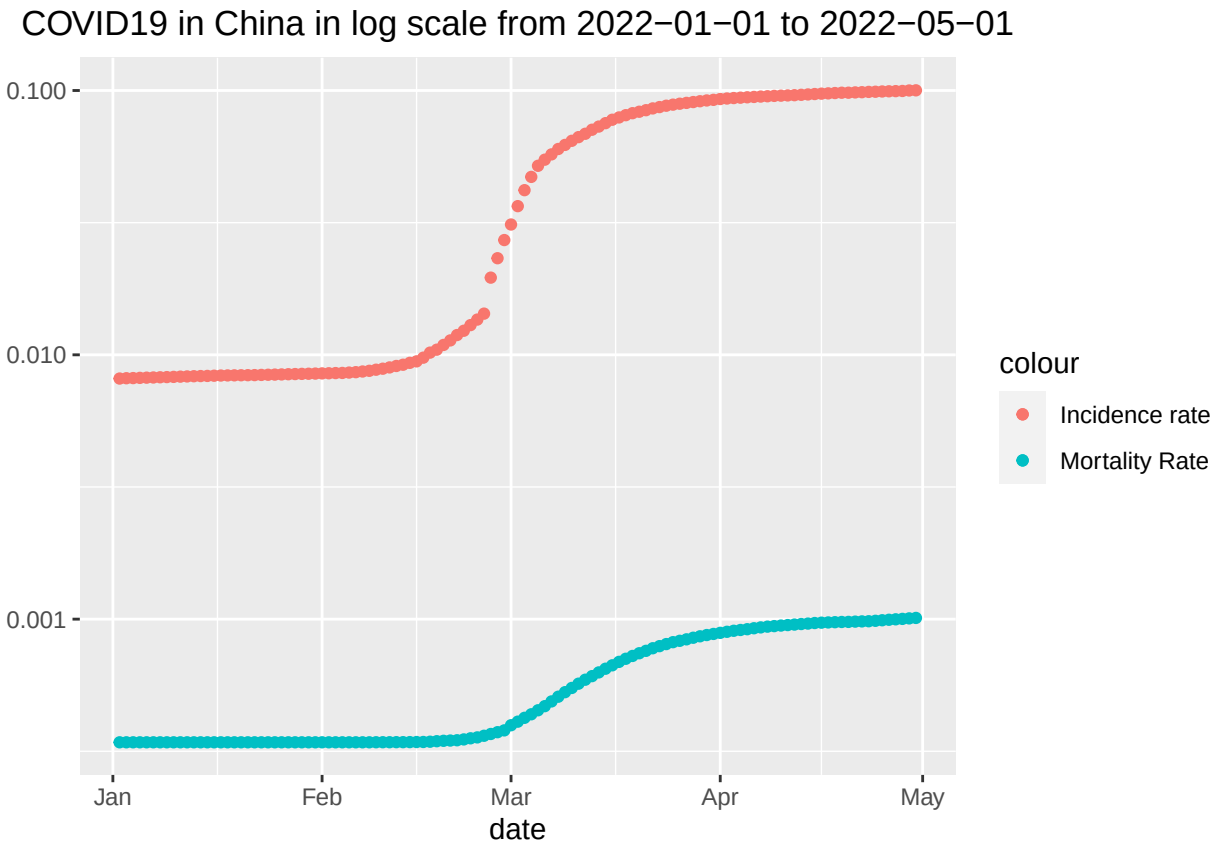
Analyzing COVID-19 cases and deaths in China on a logarithmic scale can reveal trends in the spread of the virus that might be obscured when looking at raw numbers.

```
ggplot(max_population, aes(x=date, y=Pers_Cases_of_Pop)) +
  geom_point(aes(color = "Incidence rate")) +
  geom_point(aes(y = Pers_Deaths_of_Pop, color = "Mortality Rate")) +
  scale_y_log10() +
  theme(plot.title = element_text(hjust = 0.5)) +
  labs(title = "COVID19 in China in log scale", y = NULL)
```



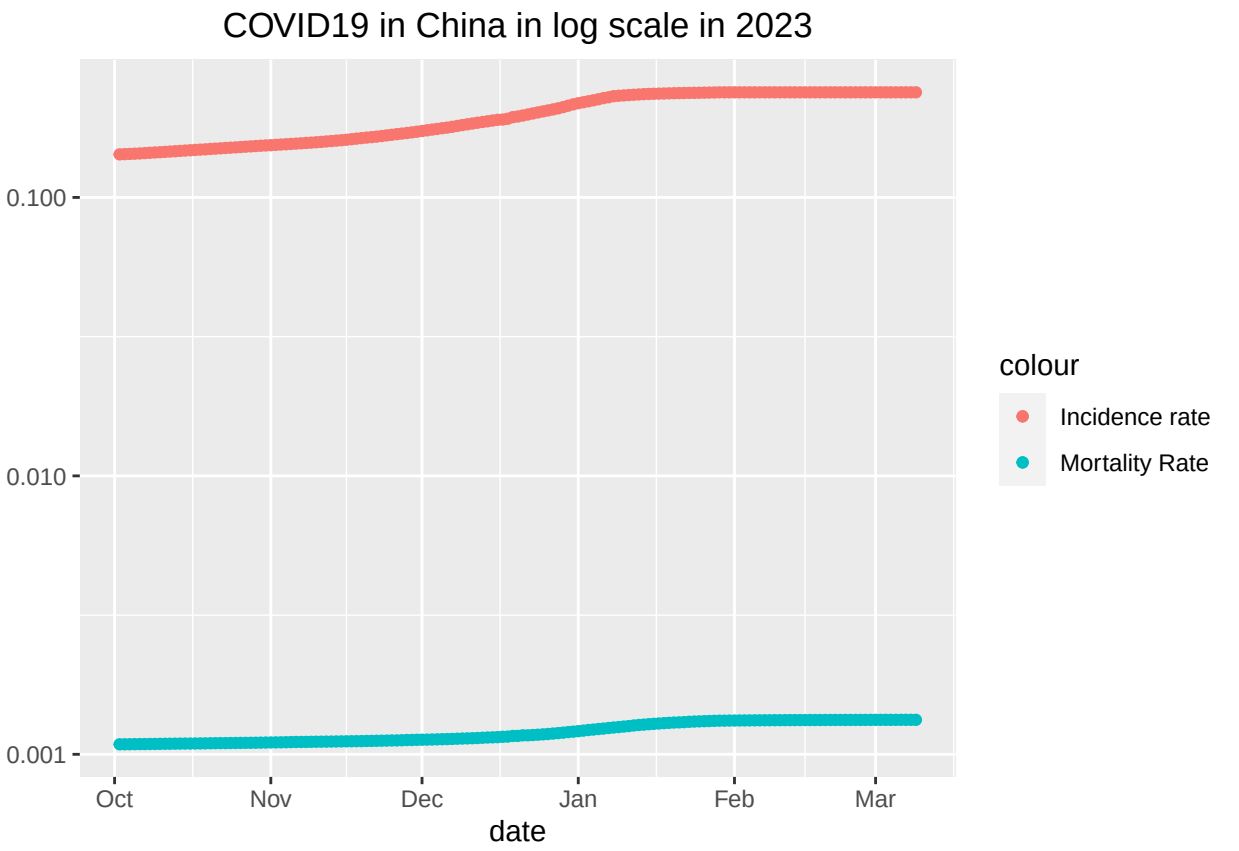

```
max_population_2022 <- max_population %>%
  filter(date > "2022-01-01", date < "2022-05-01")

ggplot(max_population_2022, aes(x=date, y=Pers_Cases_of_Pop)) +
  geom_point(aes(color = "Incidence rate")) +
  geom_point(aes(y = Pers_Deaths_of_Pop, color = "Mortality Rate")) +
  scale_y_log10() +
  theme(plot.title = element_text(hjust = 0.5)) +
  labs(title = "COVID19 in China in log scale from 2022-01-01 to 2022-05-01", y = NULL)
```




```
max_population_2023 <- max_population %>%
  filter(date > "2022-10-01")

ggplot(max_population_2023, aes(x=date, y=Pers_Cases_of_Pop)) +
  geom_point(aes(color = "Incidence rate")) +
  geom_point(aes(y = Pers_Deaths_of_Pop, color = "Mortality Rate")) +
  scale_y_log10() +
  theme(plot.title = element_text(hjust = 0.5)) +
  labs(title = "COVID19 in China in log scale in 2023", y = NULL)
```



And what we can see.

Trends in Cases:

- Steep rise in cases at the beginning of the outbreak. This indicates exponential growth.
- We can see a flattening of the curve after lockdown measures are implemented.
- 2 new peaks in cases indicate subsequent waves of infection.

Trends in Deaths:

- Generally, deaths follow a similar trend to cases, but with a time lag because COVID-19 can take weeks to become fatal in severe cases.

Model

Create a model to explain the deaths per population based on the cases per population, likely using data from the max_population (China) dataset.

```
mod <- lm(Pers_Deaths_of_Pop ~ Pers_Cases_of_Pop, data = max_population)

summary(mod)
```

```
##
## Call:
```

```
## lm(formula = Pers_Deaths_of_Pop ~ Pers_Cases_of_Pop, data = max_population)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -3.003e-04 -1.189e-05 -6.341e-06 -3.721e-06  2.172e-04
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       3.095e-04  3.580e-06   86.44   <2e-16 ***
## Pers_Cases_of_Pop 4.970e-03  4.052e-05  122.66   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.743e-05 on 1133 degrees of freedom
## Multiple R-squared:   0.93,  Adjusted R-squared:  0.9299
## F-statistic: 1.505e+04 on 1 and 1133 DF,  p-value: < 2.2e-16
```

Lastly, I will be using my results to make predictions.

```
max_population %>% slice_min(Pers_Cases_of_Pop)
```

```
## # A tibble: 1 x 7
## # Groups:   Country_Region [1]
##   Country_Region date       Total_Population Total_Cases Total_Deaths
##   <chr>          <date>                <dbl>       <dbl>        <dbl>
## 1 China          2020-01-30       1417925054        8141          171
## # i 2 more variables: Pers_Cases_of_Pop <dbl>, Pers_Deaths_of_Pop <dbl>
```

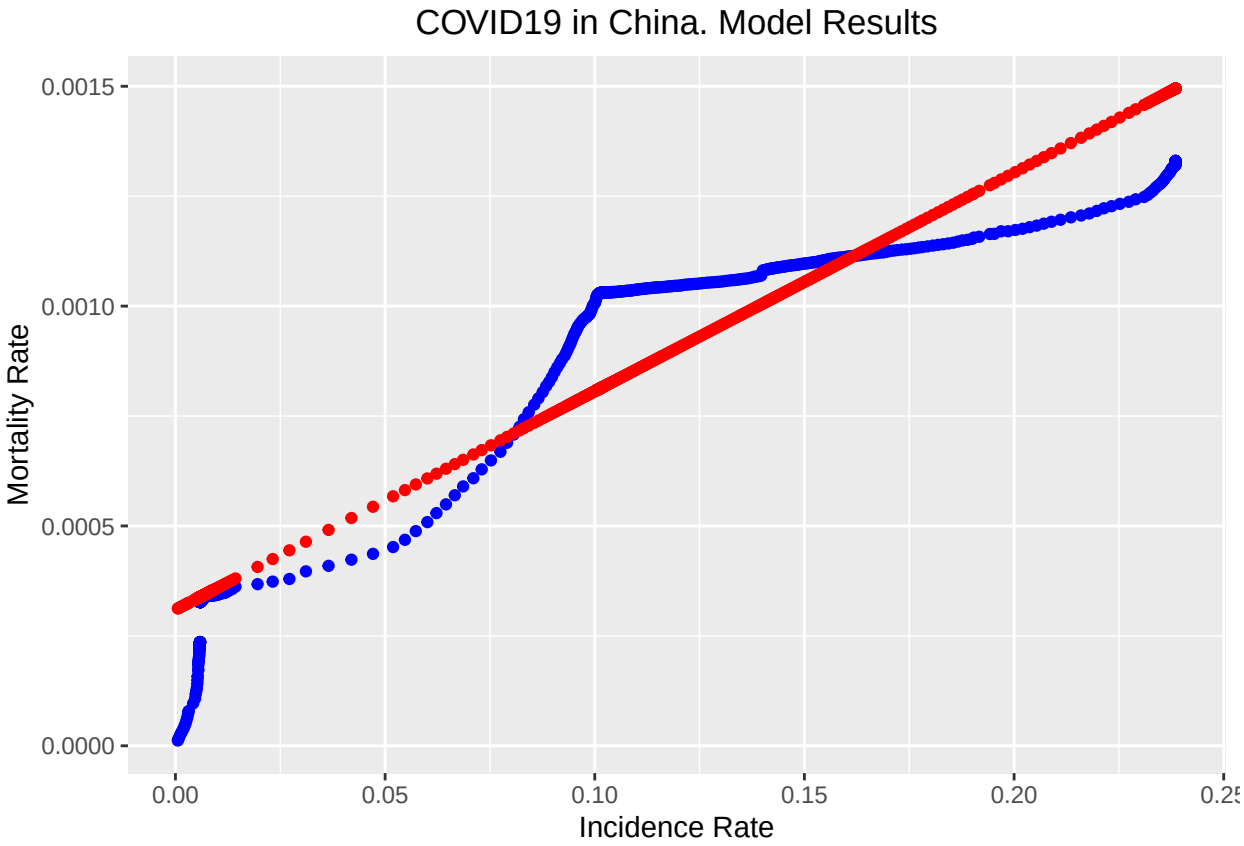
```
max_population %>% slice_max(Pers_Cases_of_Pop)
```

```
## # A tibble: 13 x 7
## # Groups:   Country_Region [1]
##    Country_Region date       Total_Population Total_Cases Total_Deaths
##    <chr>          <date>                <dbl>       <dbl>        <dbl>
##  1 China          2023-02-25       1417925054     3381708        18845
##  2 China          2023-02-26       1417925054     3381708        18847
##  3 China          2023-02-27       1417925054     3381708        18847
##  4 China          2023-02-28       1417925054     3381708        18853
##  5 China          2023-03-01       1417925054     3381708        18856
##  6 China          2023-03-02       1417925054     3381708        18856
##  7 China          2023-03-03       1417925054     3381708        18857
##  8 China          2023-03-04       1417925054     3381708        18858
##  9 China          2023-03-05       1417925054     3381708        18859
## 10 China          2023-03-06       1417925054     3381708        18860
## 11 China          2023-03-07       1417925054     3381708        18860
## 12 China          2023-03-08       1417925054     3381708        18860
## 13 China          2023-03-09       1417925054     3381708        18861
## # i 2 more variables: Pers_Cases_of_Pop <dbl>, Pers_Deaths_of_Pop <dbl>
```

```
max_pred <- max_population %>% mutate(pred = predict(mod))
```

Visualize real data and prediction.

```r
ggplot(max_pred) +
  geom_point(aes(x = Pers_Cases_of_Pop, y = Pers_Deaths_of_Pop), color = "blue") +
  geom_point(aes(x = Pers_Cases_of_Pop, y = pred), color = "red") +
  theme(plot.title = element_text(hjust = 0.5)) +
  labs(title = "COVID19 in China. Model Results", y = "Mortality Rate", x="Incidence Rate")
```



Based on the statistical significance of the coefficients and the high R-squared values, the model seems to fit the data well and suggests a positive association between cases and deaths per population.

Bias

Here's a breakdown of potential biases to consider when analyzing COVID-19 data in Holy See (Vatican City) and China:

Holy See:

Selection Bias: Due to the small population size, reported cases might not be representative of the entire population. Testing strategies or limited resources might prioritize certain groups.

Information Bias: Limited data availability due to the small scale can make it difficult to get a complete picture.

Reporting Bias: There's a possibility of underreporting due to limited testing or asymptomatic cases not being captured.

China:

Reporting Bias: Concerns exist about the accuracy of official data reported by the Chinese government. There might be underreporting of cases and deaths.

Testing Bias: Testing strategies in China might have changed throughout the pandemic, making comparisons over time difficult.

Confirmation Bias: There's a possibility of focusing on data that aligns with the government's narrative and downplaying information that contradicts it.

Censorship Bias: Restrictions on information access and media censorship can hinder a complete understanding of the situation.

Conclusion

Holy See

Due to its small population size, statistically significant conclusions are difficult to draw from the limited number of COVID-19 cases. The provided data shows no deaths attributed to COVID-19.

China:

The logarithmic scale reveals trends in cases and deaths that might not be evident in raw data. The data suggests:

- A steep rise in cases at the beginning of the outbreak, followed by a flattening due to lockdown measures.
- Subsequent waves of infection resulting in new peaks in cases.
- Deaths generally following a similar trend to cases, with a time lag.
- The linear model indicates a significant positive correlation between cases per population and deaths per population (higher cases lead to a higher number of deaths, with a moderate R-squared value of 0.93).

Predictions:

The prediction model provides an estimated number of deaths per population based on the cases per population. However, it's important to remember that this is just an estimate, and the actual number of deaths could be higher or lower due to various factors not accounted for in the model.

While the analysis provides insights into COVID-19 trends in Holy See and China, it's crucial to consider the limitations and potential biases present in the data and the model. Further analysis with more comprehensive data and addressing potential biases could provide a more accurate understanding of the situation.