Global COVID-19 Data

G. Biehunko

2025-09-13

Setup and packages used when creating this document

```
knitr::opts_chunk$set(echo = TRUE)
#Libraries
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✓ dplyr     1.1.4     ✓ readr     2.1.5
## ✓ forcats   1.0.0     ✓ stringr   1.5.1
## ✓ ggplot2   3.5.2     ✓ tibble    3.3.0
## ✓ lubridate 1.9.4     ✓ tidyr     1.3.1
## ✓ purrr     1.1.0
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
e errors
```

```
library(lubridate)
library(plotly)
```

```
##
## Attaching package: 'plotly'
##
## The following object is masked from 'package:ggplot2':
##
##     last_plot
##
## The following object is masked from 'package:stats':
##
##     filter
##
## The following object is masked from 'package:graphics':
##
##     layout
```

Introduction

Using credible and reproducible COVID-19 data from John Hopkins University, I will plot observable data to visualize the COVID deaths per million for the most populous countries with extra detail on The United States. Then I will model the data to predict whether the US is on a better, worse, or equal path as the rest of the world, with its handling of COVID-19. (subjectively)

The data is sourced from John Hopkins University with the raw data being copied from github.

```
url_in = "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/cs
se_covid_19_time_series/"
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")
pop_url_in = "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/"
pop_file_name = "csse_covid_19_data/UID_ISO_FIPS_LookUp_Table.csv"
urls = str_c(url_in, file_names)
pop_url = str_c(pop_url_in, pop_file_name )
```

Now that my data is sourced, I read in each url into my RMD.

```
us_cases = read_csv(urls[1])
```

```
## 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.
```

```
global_cases = 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_deaths = read_csv(urls[3])
```

```
## 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.
```

```
global_deaths = read_csv(urls[4])
```

```
## 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_pop = read_csv(pop_url)
```

```
## 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.
```

I tidy the data by transforming date and time for extraction in analysis and visualization. I exclude irrelevant columns, like codes not found on each data set and longitudinal and latitudinal data.

```
#US Cases
us_cases = us_cases %>%
  pivot_longer(
    cols = -c(UID, iso2, iso3, code3, FIPS, Admin2, Province_State,
              Country_Region, Lat, Long_, Combined_Key),
    names_to = "date",
    values_to = "cases"
  ) %>%
  select(-c(UID, iso2, iso3, code3, FIPS, Lat, Long_,)) %>%
  mutate(date = mdy(date))
#Global Cases
global_cases = global_cases %>%
  pivot_longer(
    cols = -c(`Province/State`, `Country/Region`, Lat, Long),
    names_to = "date",
    values_to = "cases"
  ) %>%
  select(-c(Lat, Long)) %>%
  mutate(date = mdy(date))
#US Deaths
us_deaths = us_deaths %>%
  pivot_longer(
    cols = -c(iso2, iso3, Admin2, Province_State, Country_Region, Combined_Key, UID, code3, FIP
S, Lat, Long_, Population,),
    names_to = "date",
    values_to = "deaths"
  ) %>%
  select(-c(iso2, iso3, UID, code3, FIPS, Lat, Long_,)) %>%
  mutate(date = mdy(date))
#Global Deaths
global_deaths <- global_deaths %>%
     pivot_longer(
         cols = -c(`Province/State`, `Country/Region`, Lat, Long),
         names_to = "date",
         values_to = "deaths"
     ) %>%
     select(-c(Lat, Long)) %>%
  mutate(date = mdy(date))
```

I transform the data so that the US data set will properly join together. I do this by renaming “Country/Region” and “Province/State”. I also rename Admin2, for easier visibility on the US data and my own access. Then I create a new “combined key” and add a “population” column for the global data so that it’s uniform with the relevant columns in the US data, and I can perform better analysis with population.

```
#Merge global data sets
global <- global_cases %>%
     full_join(
         global_deaths) %>%
  rename(Country_Region = `Country/Region`,
         Province_State = `Province/State`) %>%
  filter(cases > 0)
```

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

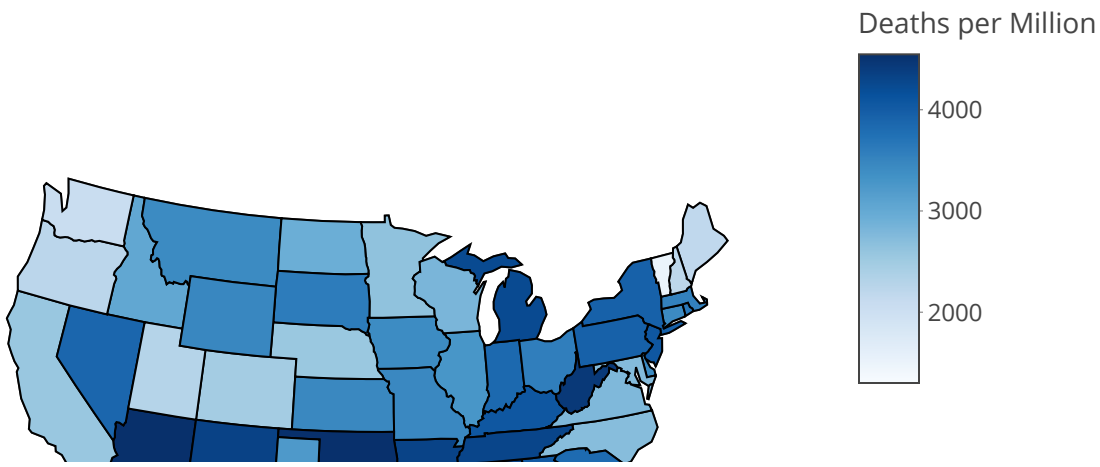
```
#Merge US data sets
us = us_cases %>%
     full_join(us_deaths) %>%
     rename(County = Admin2) %>%
     filter(cases > 0)
```

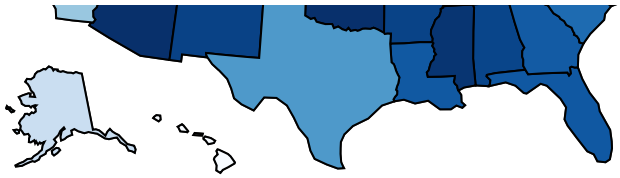
```
## Joining with `by = join_by(Admin2, Province_State, Country_Region,
## Combined_Key, date)`
```

```
#Make Combined Key column
global = global %>%
          unite("Comibed_Key", c(Province_State, Country_Region),
                sep = ",",
                na.rm = TRUE,
                remove = FALSE)
#Add population column
global = global %>%
  left_join(global_pop, by = c("Province_State", "Country_Region")) %>%
  select(-c(UID, FIPS)) %>%
  select(Province_State, Country_Region, date, cases, deaths, Population, Combined_Key)
```

Here I created an interactive plot of a US map. Each state is color coded for the deaths per million as of September 9th 2023, and the user can hover their mouse over each state showing the state name, deaths per million, total deaths, and population (as of Sep. 9, 2023). Since plotly uses the abbreviation for each US state, I had to look-up and convert each state name into the two letter abbreviation with necessary manual entry for “District of Columbia”. Hover function not accessible across all platforms, but still cool.

```
# State abbreviations
state_abbrev <- tibble(Province_State = c(state.name, "District of Columbia"),
                       state_abbr = c(state.abb, "DC"))

latest_date <- max(us$date, na.rm = TRUE)

us_states_latest <- us %>%
  group_by(Province_State) %>%
  filter(date == latest_date) %>%
  summarise(
    deaths = sum(deaths, na.rm = TRUE),
    Population = sum(Population, na.rm = TRUE),
    .groups = "drop"
  ) %>%
  mutate(deaths_per_million = (deaths / Population) * 1e6) %>%
  left_join(state_abbrev, by = "Province_State") %>%
  filter(!is.na(state_abbr))
#Make interactive choropleth
plot_us <- plot_geo(us_states_latest, locationmode = "USA-states") %>%
  add_trace(
    z = ~deaths_per_million,
    locations = ~state_abbr,
    text = ~paste0(
      Province_State,
      "<br>Deaths per Million: ", round(deaths_per_million, 1),
      "<br>Total Deaths: ", scales::comma(deaths),
      "<br>Population: ", scales::comma(Population)
    ),
    hoverinfo = "text",
    color = ~deaths_per_million,
    colors = "Blues"
  ) %>%
  colorbar(title = "Deaths per Million") %>%
  layout(title = paste("COVID-19 Deaths per Million (", latest_date, ")", sep = ""),
         geo = list(scope = "usa"))

plot_us
```





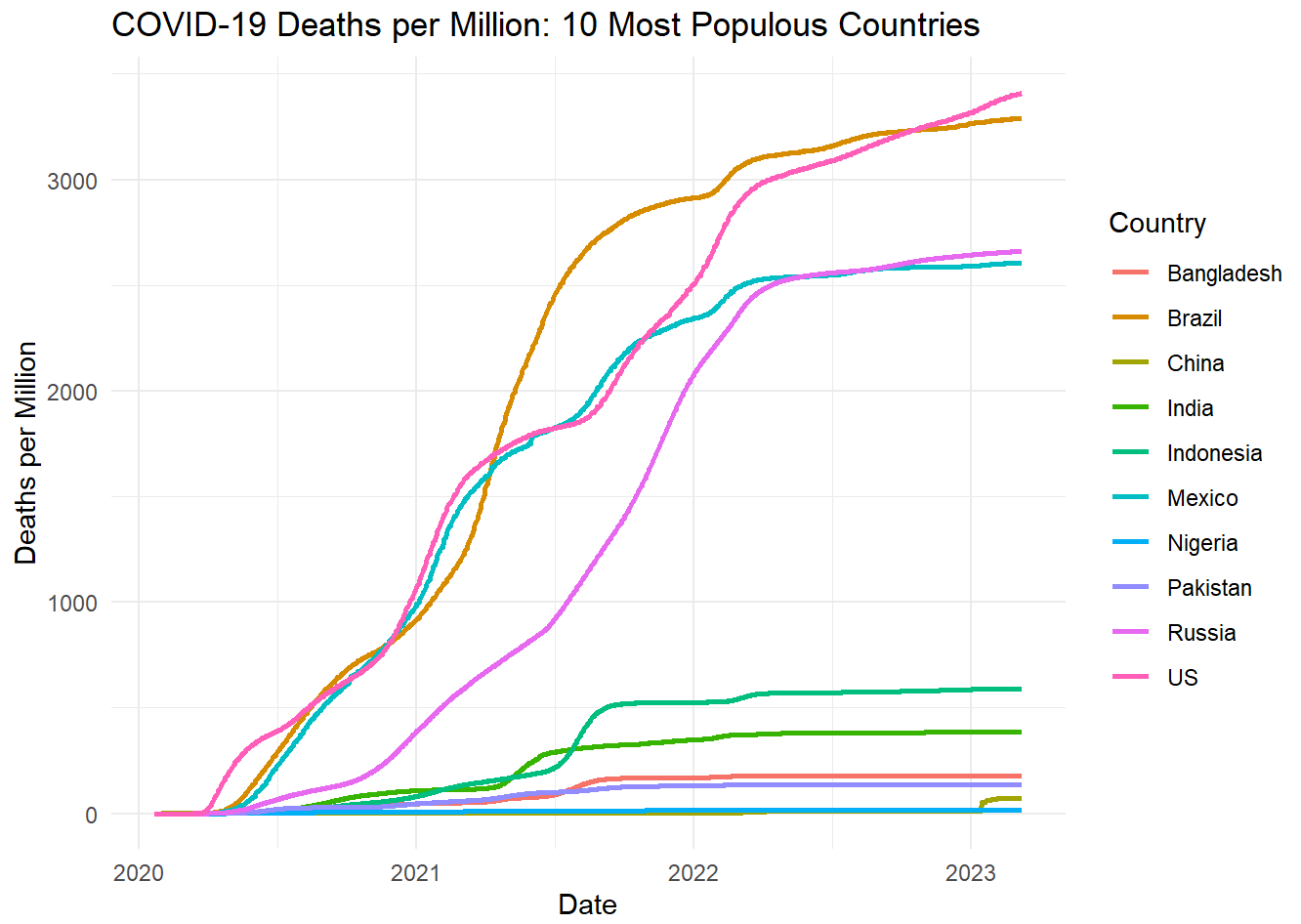
Observing and interacting with the geographic plot above allows users to observe and question data beyond population size and large metropolitan areas. Why are do Arizona and Oklahoma have such a high rate of COVID deaths per million? By observing this data, users can speculate on the social and medical standard being implemented in these states and learn from their research in order to better manage cases and case prevention of COVID-19.

In my second visualization below, I wanted to plot the deaths per million for the 10 most populous countries (as of Sept. 9, 2023) over the timeline that the data covers. I did this by identifying the 10 most populous countries on the final date documented in the data sets and making a simple line plot where each country is represented by a line with a specific color.

```
#filter top 10 countries
latest_date <- max(global$date, na.rm = TRUE)

top_countries <- global %>%
  filter(date == latest_date) %>%
  group_by(Country_Region) %>%
  summarise(population = sum(Population, na.rm = TRUE), .groups = "drop") %>%
  arrange(desc(population)) %>%
  slice_head(n = 10) %>%
  pull(Country_Region)

top_data <- global %>%
  filter(Country_Region %in% top_countries) %>%
  group_by(Country_Region, date) %>%
  summarise(
    deaths = sum(deaths, na.rm = TRUE),
    population = sum(Population, na.rm = TRUE),
    .groups = "drop"
  ) %>%
  mutate(deaths_per_million = (deaths / population) * 1e6)
#Plot data by country
plot_global = ggplot(top_data, aes(x = date, y = deaths_per_million, color = Country_Region)) +
  geom_line(size = 1) +
  labs(
    title = "COVID-19 Deaths per Million: 10 Most Populous Countries",
    x = "Date",
    y = "Deaths per Million",
    color = "Country"
  ) +
  theme_minimal() +
  theme(legend.position = "right")
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## ℹ Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
plot_global
```

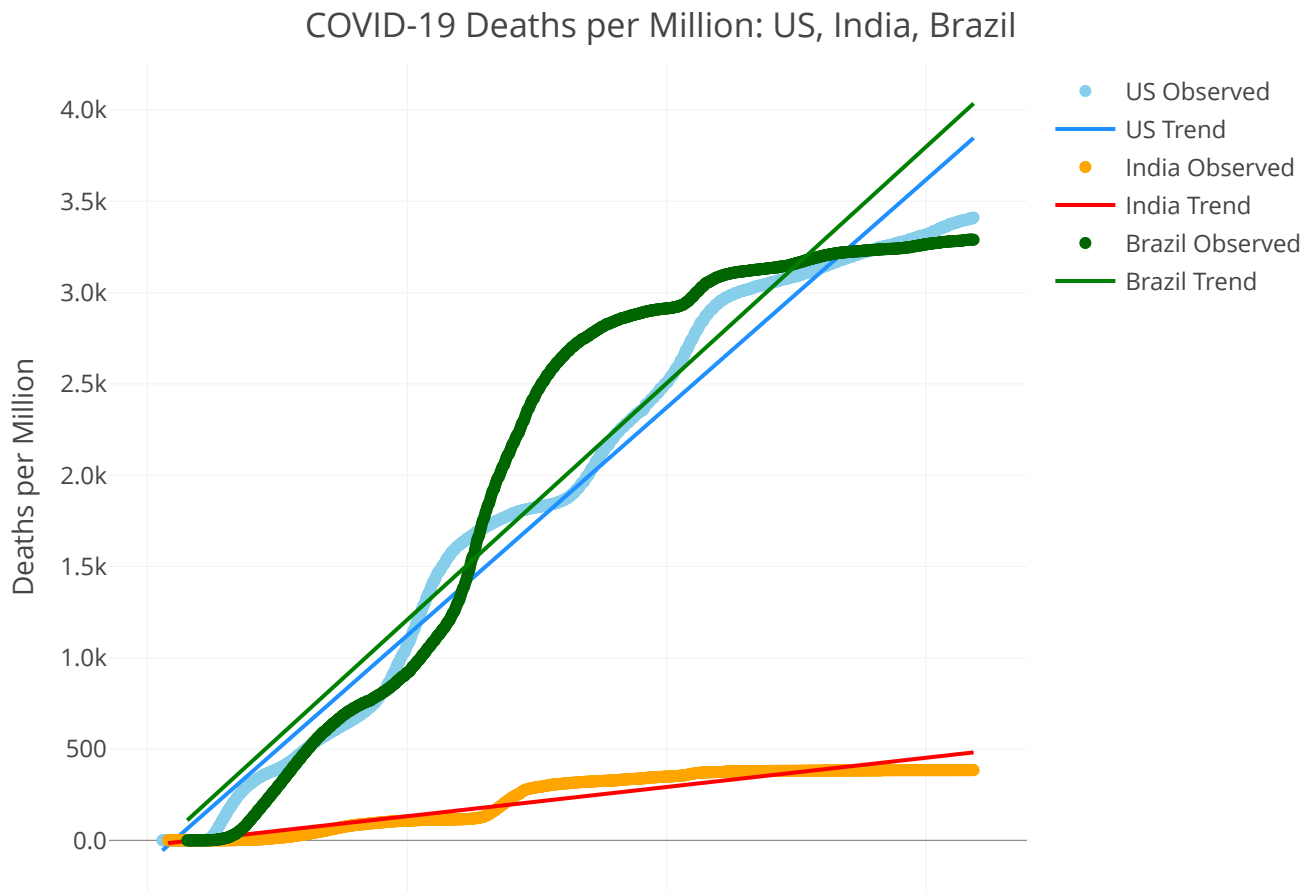


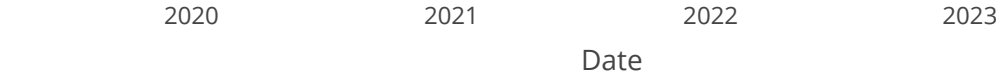
Observing the line plot above allows users to observe, compare, and question data from the 10 most populous countries in the world. What procedures were implemented in each country, and which countries tapered off the most successfully? By observing this data, users can speculate on the social and medical standard being implemented in these countries, as well as the recording of the cases and deaths in each country. Viewers of this data can hypothesize the factors of tourism, ratio of metropolitan to rural population, population density, etc. Viewers should absolutely use this graph to theorize bias on how the data is reported and documented before being archived by John Hopkins University. The official case and death counts in China were widely criticized as being under reported.

Below, I created a model to compare the observed points and the trend lines of the United States, India, and Brazil over the timeline of the data sets (roughly 3 years). To do this, I had to filter out the three countries from my global data set and transform the death and population data in a “deaths per million” value. Then, I created a predictive linear model based on the observed points in the data set. Since the points are so close together I made model interactive, so that the value of each point, and part of line pops up when the user hovers their mouse over it.

```
#Three big countries
countries <- c("US", "India", "Brazil")

three_data <- global %>%
  filter(Country_Region %in% countries) %>%
  group_by(Country_Region, date) %>%
  summarise(
    deaths = sum(deaths, na.rm = TRUE),
    population = sum(Population, na.rm = TRUE),
    .groups = "drop"
  ) %>%
  mutate(deaths_per_million = (deaths / population) * 1e6)
#Create linear model
trend_lines <- three_data %>%
  group_by(Country_Region) %>%
  do({
    mod <- lm(deaths_per_million ~ as.numeric(date), data = .)
    tibble(
      date = .$date,
      fit = predict(mod, newdata = data.frame(date = .$date))
    )
  }) %>%
  ungroup()

three_data <- three_data %>% left_join(trend_lines, by = c("Country_Region", "date"))
#Plot trends and observations
plot_three <- plot_ly() %>%
  # US
  add_markers(
    data = filter(three_data, Country_Region == "US"),
    x = ~date, y = ~deaths_per_million,
    name = "US Observed",
    marker = list(color = "skyblue"),
    text = ~paste("US<br>Date:", date, "<br>Deaths/Million:", signif(deaths_per_million, 3)),
    hoverinfo = "text"
  ) %>%
  add_lines(
    data = filter(three_data, Country_Region == "US"),
    x = ~date, y = ~fit,
    name = "US Trend",
    line = list(color = "dodgerblue")
  ) %>%
  # India
  add_markers(
    data = filter(three_data, Country_Region == "India"),
    x = ~date, y = ~deaths_per_million,
    name = "India Observed",
    marker = list(color = "orange"),
    text = ~paste("India<br>Date:", date, "<br>Deaths/Million:", signif(deaths_per_million, 3)),
    hoverinfo = "text"
  ) %>%
  add_lines(
```

```
    data = filter(three_data, Country_Region == "India"),
    x = ~date, y = ~fit,
    name = "India Trend",
    line = list(color = "red")
  ) %>%
  # Brazil
  add_markers(
    data = filter(three_data, Country_Region == "Brazil"),
    x = ~date, y = ~deaths_per_million,
    name = "Brazil Observed",
    marker = list(color = "darkgreen"),
    text = ~paste("Brazil<br>Date:", date, "<br>Deaths/Million:", signif(deaths_per_million,
3)),
    hoverinfo = "text"
  ) %>%
  add_lines(
    data = filter(three_data, Country_Region == "Brazil"),
    x = ~date, y = ~fit,
    name = "Brazil Trend",
    line = list(color = "green")
  ) %>%
  layout(
    title = "COVID-19 Deaths per Million: US, India, Brazil",
    xaxis = list(title = "Date"),
    yaxis = list(title = "Deaths per Million", tickformat = ".2s")
  )

plot_three
```





Observing and interacting with the model gives the user a comparison of the trends in COVID deaths per million in the US, Brazil, and India. Each line of actual observed data is lower than the trend lines for each country which could mean several things; The actual deaths per million are declining and making this model at the current/future date will show the trend lines lower on the graph, deaths per million were/are due for an increase to remain on trend, reporting standards for COVID have dwindled showing false numbers that remain below trend, etc. In any case, the data for India is shockingly low. Viewers could speculate that this is because having an 1.38b population is bringing down the deaths per million, but its more likely because of bias in reporting and deaths that occurred in rural areas and areas with lower quality healthcare.

Conclusion

Based on the interactive model, The United States seems to be on a similar path as a high-population country like Brazil. At the final date in my plot (March 9, 2023), The US actually had more deaths per million. Although its predictive trend line was lower, both countries had a gap between their actual observed points of data and their trend lines. This is most likely the result of each country taking affirmative action and responding to the never-before-seen COVID. Many other factors are involved in the true deaths per million, such as travel, healthcare, standards for COVID procedures, etc. There are also many factors in the reporting and documenting of COVID cases and deaths. For example, I wanted to compare China and the US in my model, but the cases and deaths for COVID in China on this data set was astoundingly low. After choosing Brazil, China, and the US, I found out India was similar in that regard. Hopefully the data gathered since March 9, 2023 is broadly more accurate and allows us to continue tapering off the curve and continuing the downward trend in COVID-19 deaths.