Novel Coronavirus (COVID-19) Cases in China: Past, Present, and Future

6/14/2021

Coronavirus disease (COVID-19) is an infectious disease caused by a newly discovered coronavirus. Most people who fall sick with COVID-19 will experience mild to moderate symptoms and recover without special treatment. I import, tidy and analyze the COVID19 dataset from the Johns Hopkins github site. COVID-19 Data Repository by the Center for Systems Science and Engineering (CSSE) at Johns Hopkins University includes a complete list of all sources ever used in the data set, since January 21, 2010, for example, World Health Organization (WHO), European Center for Disease Prevention and Control (ECDC), and BNO News etc. This is a global pandemic problem and the governments across the globe are still trying hard to overcome and prevent the spread of COVID-19. Today, I will focus on my analysis in China and demonstrate several key insights derived from the COVID-19 data.

Research Questions

1. Which province in China has the most COVID-19 deaths? What's the best way to visually represent the information?
2. Which province in China has the most impressive COVID-19 recovery rate? Is there any reason for such a bounce back?
3. How do COVID-19 cases evolve over time in different China provinces? What should policymakers do to address such issue?
4. Can policymakers predict the number of COVID-19 deaths (t+1) in Hubei based on the historical data CSSE collected?

Step 0: Import Library

```
# Special functions
applySum <- function(df, ...) {
  assertthat::assert_that(...length() > 0, msg = "one or more column indexes are required")
  mutate(df, Sum = apply(as.data.frame(df[, c(...)]), 1, sum))
}

# Force R not to use exponential notation (e.g. e+10)
options(scipen = 999)
```

Step 1: Load Data

- `read_csv()` reads comma delimited files, read_csv2() reads semicolon separated files (common in countries where , is used as the decimal place), read_tsv() reads tab delimited files, and read_delim() reads in files with any delimiter.
- `glimpse()` gets a glimpse of your data.

```
# Get the main directory for URLs
url_in <- "https://github.com/CSSEGISandData/COVID-19/raw/master/csse_covid_19_data/csse_covid_19_time_
file_names <- c("time_series_covid19_confirmed_global.csv",
                "time_series_covid19_deaths_global.csv",
                "time_series_covid19_recovered_global.csv")
urls <- str_c(url_in, file_names)
urls
```

```
## [1] "https://github.com/CSSEGISandData/COVID-19/raw/master/csse_covid_19_data/csse_covid_19_time_seri
## [2] "https://github.com/CSSEGISandData/COVID-19/raw/master/csse_covid_19_data/csse_covid_19_time_seri
## [3] "https://github.com/CSSEGISandData/COVID-19/raw/master/csse_covid_19_data/csse_covid_19_time_seri
```

```
# Load data into R
global_cases <- read_csv(urls[1])
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   .default = col_double(),
##   `Province/State` = col_character(),
##   `Country/Region` = col_character()
## )
## i Use `spec()` for the full column specifications.
```

```
global_deaths <- read_csv(urls[2])
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   .default = col_double(),
##   `Province/State` = col_character(),
##   `Country/Region` = col_character()
## )
## i Use `spec()` for the full column specifications.
```

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

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   .default = col_double(),
##   `Province/State` = col_character(),
##   `Country/Region` = col_character()
## )
## i Use `spec()` for the full column specifications.
```

Step 2: Tidy and Transform Data

The key activities in this step are:

1. Filter only rows related to China according to proposed research questions.
2. Remove missing values from the data (NULL, NA, Unknown.)
3. Summarize the COVID-19 cases, deaths, and recovery by province/state in China.
4. Check the data type after data preprocessing.
5. Identify extreme values with summary statistics.
6. Gather to unpivot data to a long format for Heatmap visualization.

```
# Filter only rows related to China
# Remove missing values from the data (NULL, NA, Unknown.)
# Summarize the COVID-19 cases, deaths, and recovery by province/state in China.
china_cases = global_cases %>%
  filter(`Country/Region` == "China") %>%
  drop_na() %>%
  applySum(5:515) %>%
  select(1:4, 516) %>%
  rename(total_cases = Sum)

china_deaths = global_deaths %>%
  filter(`Country/Region` == "China") %>%
  drop_na() %>%
  applySum(5:515) %>%
  select(1:4, 516) %>%
  rename(total_deaths = Sum)

china_recover = global_recovery %>%
  filter(`Country/Region` == "China") %>%
  drop_na() %>%
  applySum(5:515) %>%
  select(1:4, 516) %>%
  rename(total_recovery = Sum)
```

```
# Get a glimpse of China COVID-19 cases
glimpse(china_cases)
```

```
## Rows: 33
## Columns: 5
## $ `Province/State` <chr> "Anhui", "Beijing", "Chongqing", "Fujian", "Gansu", "~
## $ `Country/Region` <chr> "China", "China", "China", "China", "China", "China",~
## $ Lat              <dbl> 31.8257, 40.1824, 30.0572, 26.0789, 35.7518, 23.3417,~
## $ Long             <dbl> 117.2264, 116.4142, 107.8740, 117.9874, 104.2861, 113~
## $ total_cases      <dbl> 493426, 430087, 292268, 221749, 82573, 938488, 129804~
```

```
summary(china_cases)
```

```
##  Province/State     Country/Region          Lat             Long
##  Length:33          Length:33          Min.   :19.20   Min.   : 85.24
##  Class :character   Class :character   1st Qu.:27.61   1st Qu.:107.87
##  Mode  :character   Mode  :character   Median :31.83   Median :113.55
##                                        Mean   :32.89   Mean   :111.79
##                                        3rd Qu.:37.90   3rd Qu.:117.32
##                                        Max.   :47.86   Max.   :127.76
##   total_cases
```

```
##  Min.   :     503
##  1st Qu.:  111823
##  Median :  292268
##  Mean   : 1371344
##  3rd Qu.:  493426
##  Max.   :33483557
```

```
# Get a glimpse of China COVID-19 deaths
glimpse(china_deaths)
```

```
## Rows: 33
## Columns: 5
## $ `Province/State` <chr> "Anhui", "Beijing", "Chongqing", "Fujian", "Gansu", "~
## $ `Country/Region` <chr> "China", "China", "China", "China", "China", "China",~
## $ Lat              <dbl> 31.8257, 40.1824, 30.0572, 26.0789, 35.7518, 23.3417,~
## $ Long             <dbl> 117.2264, 116.4142, 107.8740, 117.9874, 104.2861, 113~
## $ total_deaths     <dbl> 2945, 4290, 2951, 482, 987, 3857, 982, 981, 2931, 309~
```

```
summary(china_deaths)
```

```
##  Province/State     Country/Region          Lat             Long
##  Length:33          Length:33          Min.   :19.20   Min.   : 85.24
##  Class :character   Class :character   1st Qu.:27.61   1st Qu.:107.87
##  Mode  :character   Mode  :character   Median :31.83   Median :113.55
##                                        Mean   :32.89   Mean   :111.79
##                                        3rd Qu.:37.90   3rd Qu.:117.32
##                                        Max.   :47.86   Max.   :127.76
##   total_deaths
##  Min.   :      0
##  1st Qu.:    482
##  Median :   1399
##  Mean   :  67200
##  3rd Qu.:   3092
##  Max.   :2109308
```

```
# Get a glimpse of China COVID-19 recoveries
glimpse(china_recover)
```

```
## Rows: 33
## Columns: 5
## $ `Province/State` <chr> "Anhui", "Beijing", "Chongqing", "Fujian", "Gansu", "~
## $ `Country/Region` <chr> "China", "China", "China", "China", "China", "China",~
## $ Lat              <dbl> 31.8257, 40.1824, 30.0572, 26.0789, 35.7518, 23.3417,~
## $ Long             <dbl> 117.2264, 116.4142, 107.8740, 117.9874, 104.2861, 113~
## $ total_recovery   <dbl> 474602, 398823, 278254, 208173, 79026, 891628, 123158~
```

```
summary(china_recover)
```

```
##  Province/State     Country/Region          Lat             Long
##  Length:33          Length:33          Min.   :19.20   Min.   : 85.24
##  Class :character   Class :character   1st Qu.:27.61   1st Qu.:107.87
```

```
##  Mode  :character   Mode  :character   Median :31.83   Median :113.55
##                                        Mean   :32.72   Mean   :111.71
##                                        3rd Qu.:37.58   3rd Qu.:117.32
##                                        Max.   :47.86   Max.   :127.76
##  total_recovery
##  Min.   :     490
##  1st Qu.:  102716
##  Median :  278254
##  Mean   : 1242852
##  3rd Qu.:  474602
##  Max.   :29941394
```

The summary statistics shows that of all the three datasets, the maximum value is far greater than the mean and minimum value in the total COVID-19 cases, deaths, and recovery by province/state in China.

```
# Sort and show top 5 provinces with total COVID-19 cases
china_cases %>%
  arrange(desc(total_cases)) %>%
  head(5)
```

```
## # A tibble: 5 x 5
##   `Province/State` `Country/Region`   Lat  Long total_cases
##   <chr>            <chr>            <dbl> <dbl>       <dbl>
## 1 Hubei            China             31.0  112.    33483557
## 2 Hong Kong        China             22.3  114.     2858353
## 3 Guangdong        China             23.3  113.      938488
## 4 Zhejiang         China             29.2  120.      643508
## 5 Henan            China             37.9  115.      641308
```

Last but not least, for Heatmap purpose, I gather to unpivot data to a long format, a critical step in getting to the answer towards the research question #3. I aggregate the data by month and year for visualizing the evolution of COVID-19 cases in a more convenient way.

```
china_cases_long <- global_cases %>%
  filter(`Country/Region` == "China") %>%
  drop_na() %>%
  select(-2:-4) %>%
  gather(key = date,
         value = cases,
         `1/22/20`:`6/15/21`) %>%
  mutate(date = mdy(date),
         month = month(date, label = TRUE),
         year = year(date)) %>%
  group_by(`Province/State`, year) %>%
  summarise(total_cases = sum(cases)) %>%
  mutate(year = as.factor(year))
```

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

```
head(china_cases_long)
```

```
## # A tibble: 6 x 3
## # Groups:   Province/State [3]
##   `Province/State` year  total_cases
##   <chr>            <fct>       <dbl>
## 1 Anhui            2020       328142
## 2 Anhui            2021       165284
## 3 Beijing          2020       256540
## 4 Beijing          2021       173547
## 5 Chongqing        2020       193834
## 6 Chongqing        2021        98434
```

In an attempt to answering research question #4, I prepare the data for modelling purpose below:

```
china_deaths_ts = global_deaths %>%
  filter(`Country/Region` == "China") %>%
  drop_na() %>%
  select(-2:-4) %>%
  gather(key = date,
         value = cases,
         `1/22/20`:`6/15/21`) %>%
  mutate(date = mdy(date),
         month = month(date, label = TRUE),
         year = year(date)) %>%
  filter(`Province/State` == "Hubei") %>%
  select(date, cases) %>%
  rename(ds = date, y = cases)
head(china_deaths_ts)
```

```
## # A tibble: 6 x 2
##   ds              y
##   <date>     <dbl>
## 1 2020-01-22    17
## 2 2020-01-23    17
## 3 2020-01-24    24
## 4 2020-01-25    40
## 5 2020-01-26    52
## 6 2020-01-27    76
```

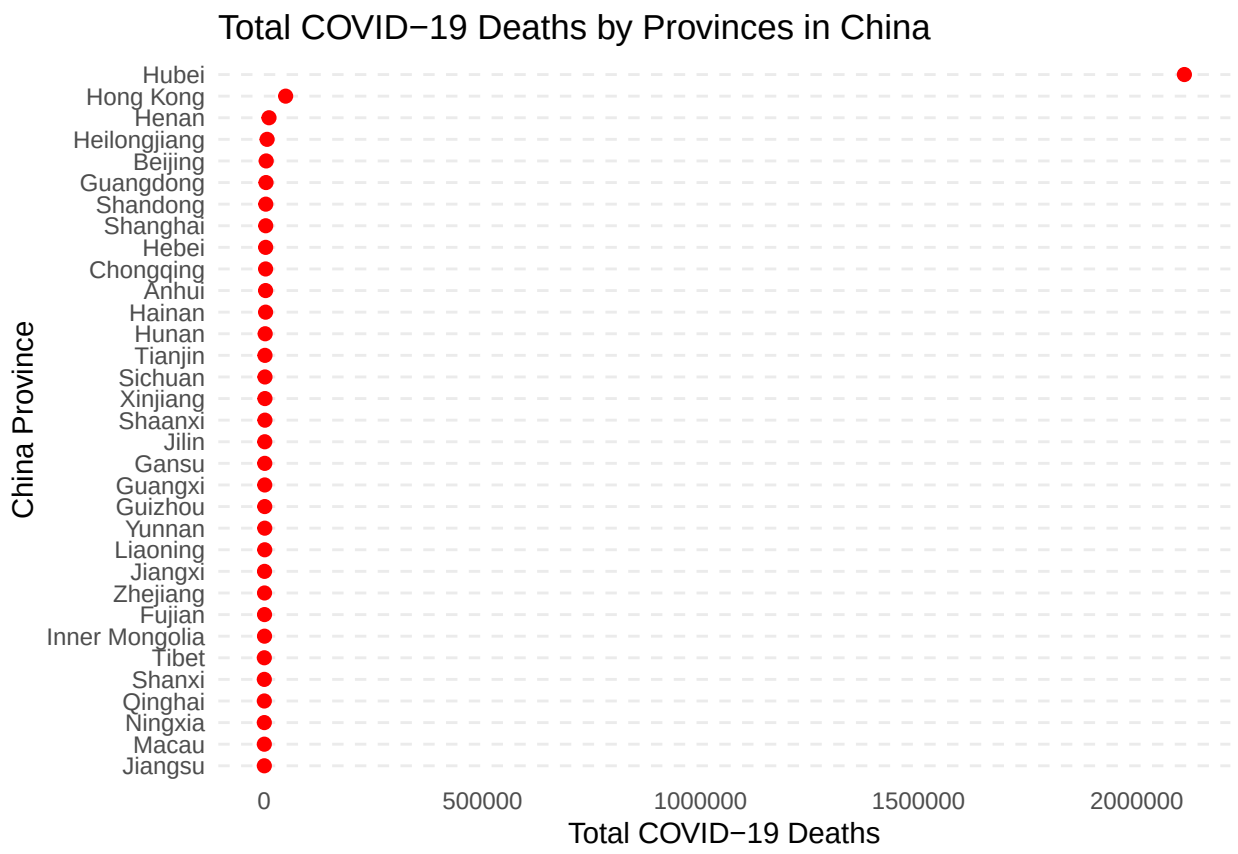
Step 3: Add Visualizations and Analysis

1. Which province in China has the most COVID-19 deaths? What's the best way to visually represent the information?

Apparently, Hubei stands out to contract the most COVID-19 deaths. . . 2,109,308 are dead due to COVID-19. This is a very concerning situation and policymakers need to investigate this issue as soon as possible! Hong Kong and Henan are placed 2nd and 3rd respectively in the number of COVID-19 deaths.

```
# Dot plot
ggplot(china_deaths, aes(x = reorder(`Province/State`, total_deaths), y = total_deaths)) +
    geom_point(
        shape = 21,
        fill = "red",
```

```
        color = "red",
        size = 2
    ) +
    labs(title = "Total COVID-19 Deaths by Provinces in China",
         x = "China Province",
         y = "Total COVID-19 Deaths") +
    coord_flip() +
    theme_minimal() +
    theme(
        panel.grid.major.x = element_blank(),
        panel.grid.minor.x = element_blank(),
        panel.grid.major.y = element_line(linetype = "dashed")
    )
```




```
# Sort and show top 5 provinces with total COVID-19 deaths
china_deaths %>%
  arrange(desc(total_deaths)) %>%
  head(5)
```

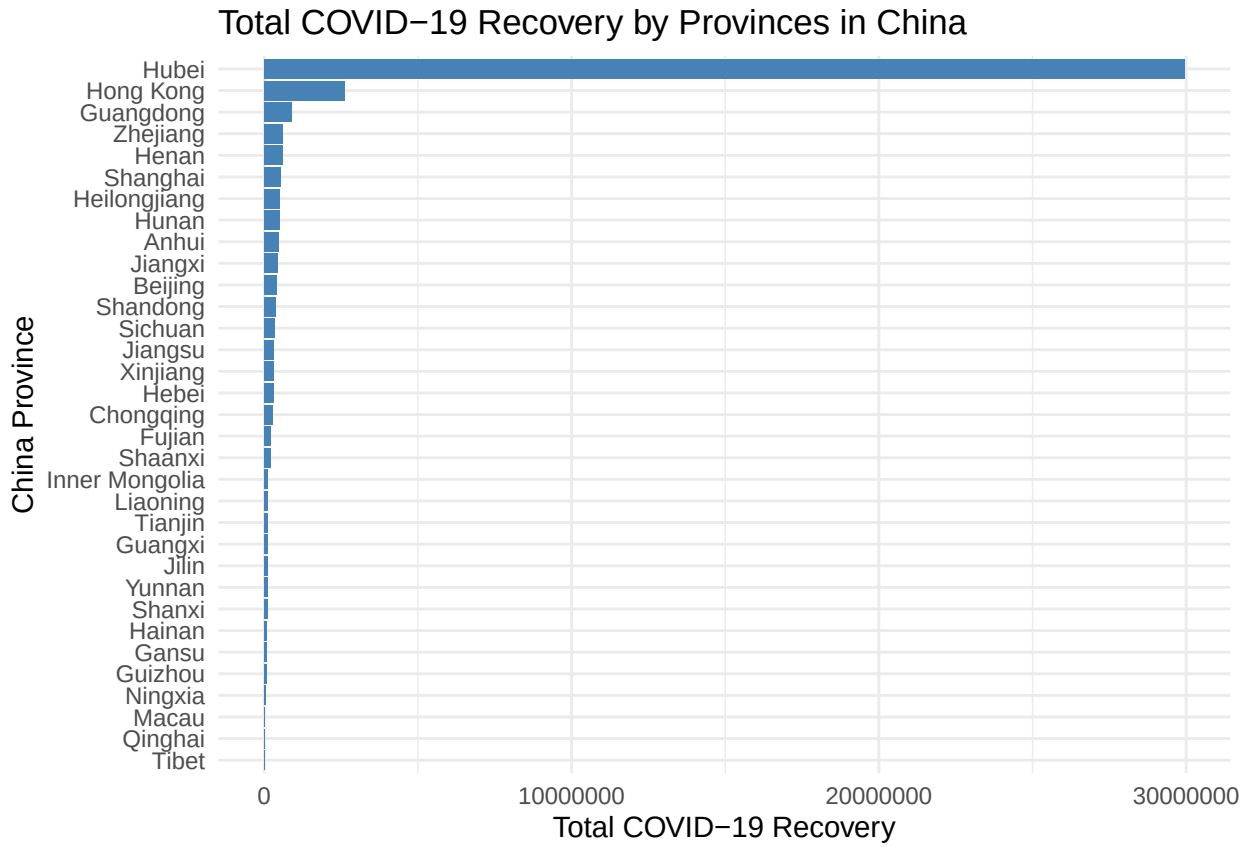
```
## # A tibble: 5 x 5
##   `Province/State` `Country/Region`   Lat  Long total_deaths
##   <chr>            <chr>            <dbl> <dbl>        <dbl>
## 1 Hubei            China             31.0  112.      2109308
## 2 Hong Kong        China             22.3  114.        48906
## 3 Henan            China             37.9  115.        10734
```

```
## 4 Heilongjiang     China              47.9  128.         6395
## 5 Beijing          China              40.2  116.         4290
```

2. Which province in China has the most impressive COVID-19 recovery rate? Is there any reason for such a bounce back?

While Hubei suffers from the most number of deaths, it does show a high recovery rate. Likewise, this pattern is found in Hong Kong. A Hubei Reborn for New Glories; A China Embracing Openness and Prosperity explained how Hubei re-emerged and embraced openness and prosperity for new glories. "Every effort has been made to make up for the time lost, and to seize the opportunities to build capacity and shore up weak areas. The pandemic and post-COVID reopening have served as an opportunity to boost economic upgrading and transformation. From post-COVID recovery to steady economic and social development, and from winning a decisive victory over extreme poverty to fully building a moderately prosperous society, Hubei has again scored impressive achievements in the major test of rebuilding and reviving development."

```
# Bar Plot with Reorder
ggplot(china_recover, aes(reorder(`Province/State`, total_recovery), total_recovery)) +
  geom_bar(stat="identity", fill="steelblue") +
  theme_minimal() +
  labs(title = "Total COVID-19 Recovery by Provinces in China",
         x = "China Province",
         y = "Total COVID-19 Recovery") +
  coord_flip()
```



```r
# Sort and show top 5 provinces with total COVID-19 cases
china_recover %>%
  arrange(desc(total_recovery)) %>%
  head(5)
```

```
## # A tibble: 5 x 5
##   `Province/State` `Country/Region`   Lat  Long total_recovery
##   <chr>            <chr>            <dbl> <dbl>          <dbl>
## 1 Hubei            China             31.0  112.       29941394
## 2 Hong Kong        China             22.3  114.        2630717
## 3 Guangdong        China             23.3  113.         891628
## 4 Zhejiang         China             29.2  120.         617520
## 5 Henan            China             33.9  114.         610666
```

3. How do COVID-19 cases evolve over time in different China provinces? What should policymakers do to address such issue?

Date-time data can be frustrating to work with and I present this information by a heatmap of COVID-19 cases over time. Hubei still shows the most number of COVID-19 cases over time, followed by Hong Kong and Guangdong!

Learn how these 3 provinces come up with COVID-19 measures here:

Hubei

- China's Response to the COVID-19 Outbreak: A Model for Epidemic Preparedness and Management - FullText - Dubai Medical Journal 2020, Vol. 3, No. 2 - Karger Publishers
- Combined measures to control the COVID-19 pandemic in Wuhan, Hubei, China: A narrative review - ScienceDirect

Hong Kong

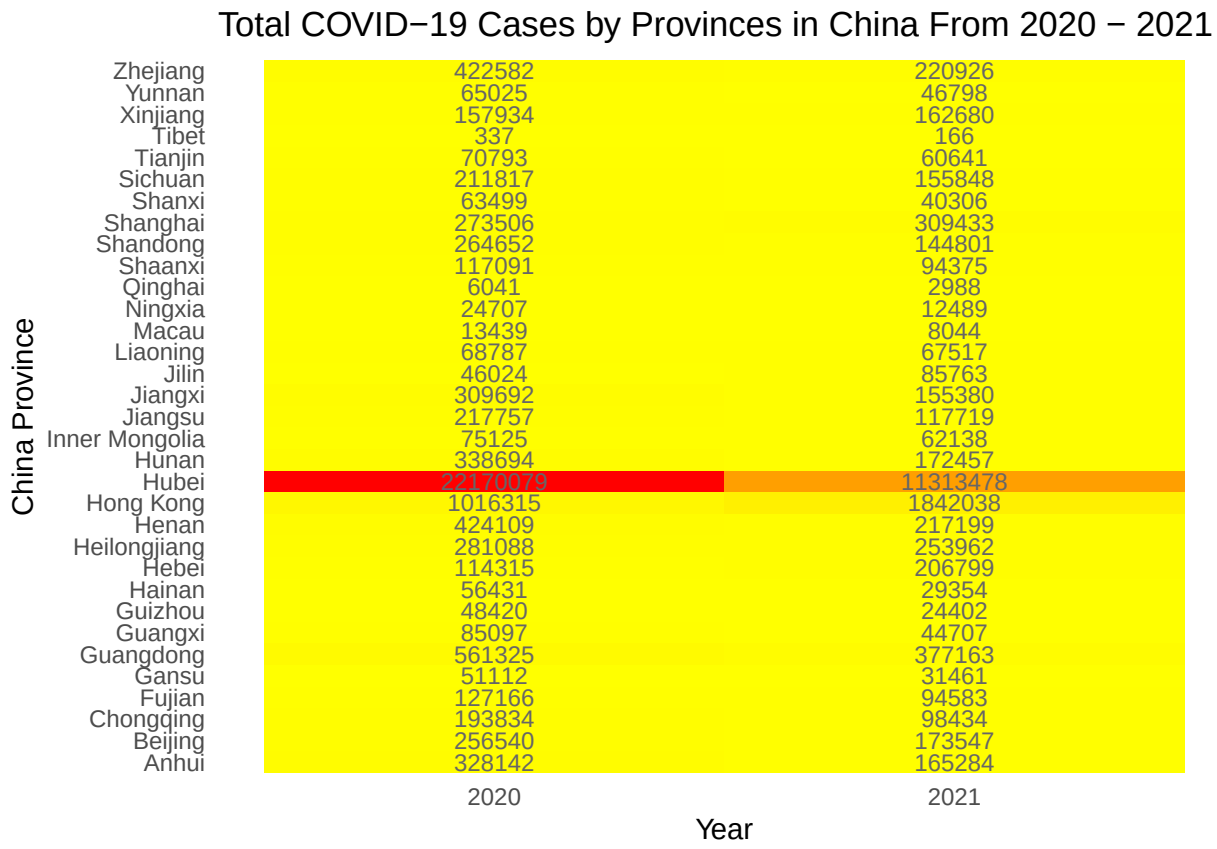
- COVID-19 Thematic Website - Together, We Fight the Virus - Inbound Travel
- Hong Kong Social Distancing And Travel Rules For Covid-19: What You Can And Can't Do | Tatler Hong Kong
- COVID-19 Thematic Website - Together, We Fight the Virus - Home

Guangdong

- China's Guangzhou covid cases rise as authorities tighten measures
- China's Guangzhou city imposes more COVID-19 measures - CNA
- China's Guangdong tightens coronavirus measures as cases persist, East Asia News & Top Stories - The Straits Times

```r
ggplot(china_cases_long, aes(x = year, y = `Province/State`, fill=total_cases)) +
    geom_tile() +
    geom_text(aes(year, `Province/State`, label=total_cases), color = "grey40", size = 3) +
    scale_fill_gradient(low = "yellow", high = "red") +
    labs(title = "Total COVID-19 Cases by Provinces in China From 2020 - 2021",
         x = "Year",
         y = "China Province") +
    theme_minimal() +
```

```
    theme(axis.line = element_blank(),
          axis.ticks = element_blank(),
          panel.grid.major = element_blank(),
          panel.grid.minor = element_blank(),
          panel.border = element_blank(),
          panel.background = element_blank(),
          legend.position="none")
```




```
# Sort and show top 5 provinces with total COVID-19 cases
china_cases_long %>%
  arrange(desc(total_cases)) %>%
  head(5)
```

```
## # A tibble: 5 x 3
## # Groups:   Province/State [3]
##   `Province/State` year  total_cases
##   <chr>            <fct>       <dbl>
## 1 Hubei            2020     22170079
## 2 Hubei            2021     11313478
## 3 Hong Kong        2021      1842038
## 4 Hong Kong        2020      1016315
## 5 Guangdong        2020       561325
```

4. Can policymakers predict the number of COVID-19 deaths (t+1) in Hubei based on the historical data CSSE collected?

Yes. In this question, I rely on Prophet, "a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects. It works best with time series that have strong seasonal effects and several seasons of historical data. Prophet is robust to missing data and shifts in the trend, and typically handles outliers well. Prophet is open source software released by Facebook's Core Data Science team. It is available for download on CRAN and PyPI." (Prophet | Forecasting at scale.)

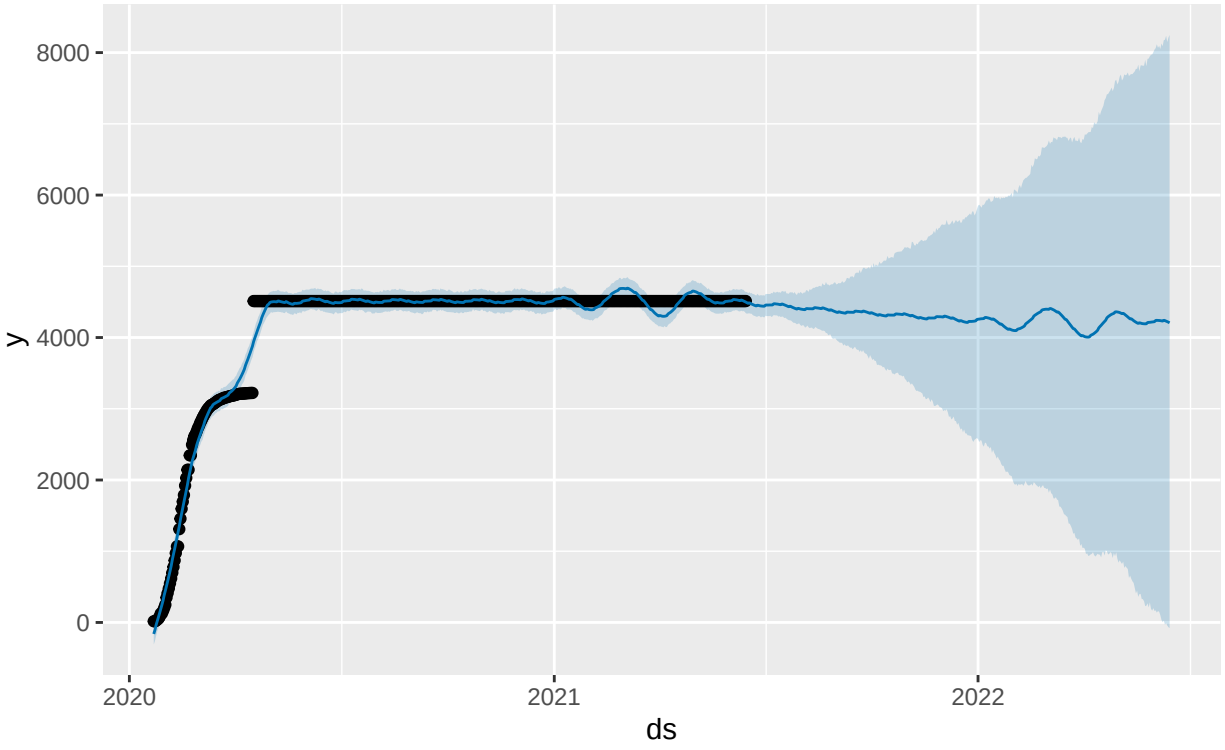
I predicted the COVID-19 deaths for the next year. The predicted value centers around 4,200 people on average as you can see from the below forecasting plot.

```
# Call the prophet function to fit the model.
m <- prophet(china_deaths_ts, yearly.seasonality = TRUE, daily.seasonality = TRUE)

# Takes the model object and a number of periods to forecast and produces a suitable dataframe. By defa
future <- make_future_dataframe(m, periods = 365)
forecast <- predict(m, future)
tail(forecast[c('ds', 'yhat', 'yhat_lower', 'yhat_upper')])
```

```
##             ds     yhat yhat_lower yhat_upper
## 871 2022-06-10 4242.194   25.67950   8173.598
## 872 2022-06-11 4237.770  -30.64854   8180.642
## 873 2022-06-12 4229.949  -11.61952   8235.894
## 874 2022-06-13 4223.081  -42.86411   8147.555
## 875 2022-06-14 4215.276  -66.96384   8210.746
## 876 2022-06-15 4206.834  -78.13508   8247.112
```

```
# Plot the forecast
plot(m, forecast)
```



Step 4: Identify Bias

As I am progressing in "Data Science as a Field" towards the end, I do not let myself fall into my personal judgement or biased experience, but I rather investigate data for the answer. I completely base my analysis on data. Hubei, Hong Kong, and Guangdong are the top 3 provinces with the most number of COVID-19 cases and deaths; however, there's a good sign of strong recovery rate likewise. Policymakers should look at this data with respect to the policies implemented in action if they positively affect the Chinese people. My finding agrees with the China's measures on the stringent measures to detain those who violate virus prevention measures and impose more restrictions on business and social activity, seeking to curb the spread of COVID-19 cases. This global pandemic is evolving and posing threats to the world, therefore policymakers need to learn the successful policies that work in other countries and apply to their country with caution, impartiality, and agility.

Additional Resources

- Coronavirus Singapore - live map tracker from Microsoft Bing
- See the latest data in your region - Johns Hopkins Coronavirus Resource Center
- COVID-19 Singapore Dashboard | UCA
- JHU CSSE – Center For Systems Science and Engineering at JHU
- An interactive web-based dashboard to track COVID-19 in real time - The Lancet Infectious Diseases
- A Hubei Reborn for New Glories; A China Embracing Openness and Prosperity
- ggplot US state and China province heatmap | Welcome to my blog