Japan's Latest Wave of Covid-19 Pandemic

Welly Wong

April 12, 2023

Load Packages

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6      v purrr   0.3.5
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
##
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
##
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(deSolve)     #ode():solve initial value problems of a system of 1st-order ordinary differential
library(incidence)   #incidence():compute incidence from dates, fit(): linear regression on log-inciden
library(epitrix)     #gamma_mucv2shapescale():Reparameterise Gamma distributions
library(distcrete)   #distcrete():Creates discretised versions of continuous distribution
library(projections) #Project Future Case Incidence
```

Load Data

Lab confirmed cases and confirmed deaths data came from COVID-19 Data Repository by the Center for Systems Science and Engineering (CSSE) at Johns Hopkins University https://github.com/CSSEGISandData/COVID-19/tree/master/csse_covid_19_data/csse_covid_19_time_series

Vaccination data came from Our World in Data Covid-19 data repository: https://github.com/owid/covid-19-data/blob/master/public/data/vaccinations/country_data/Japan.csv

```
url_folder = "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_
file_names = c("time_series_covid19_confirmed_global.csv", "time_series_covid19_deaths_global.csv")
urls = str_c(url_folder, 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.
```

```
japan_vaccine = read_csv("https://raw.githubusercontent.com/owid/covid-19-data/master/public/data/vacci
```

```
## Rows: 774 Columns: 8
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (3): location, vaccine, source_url
## dbl  (4): total_vaccinations, people_vaccinated, people_fully_vaccinated, to...
## date (1): date
##
## 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.
```

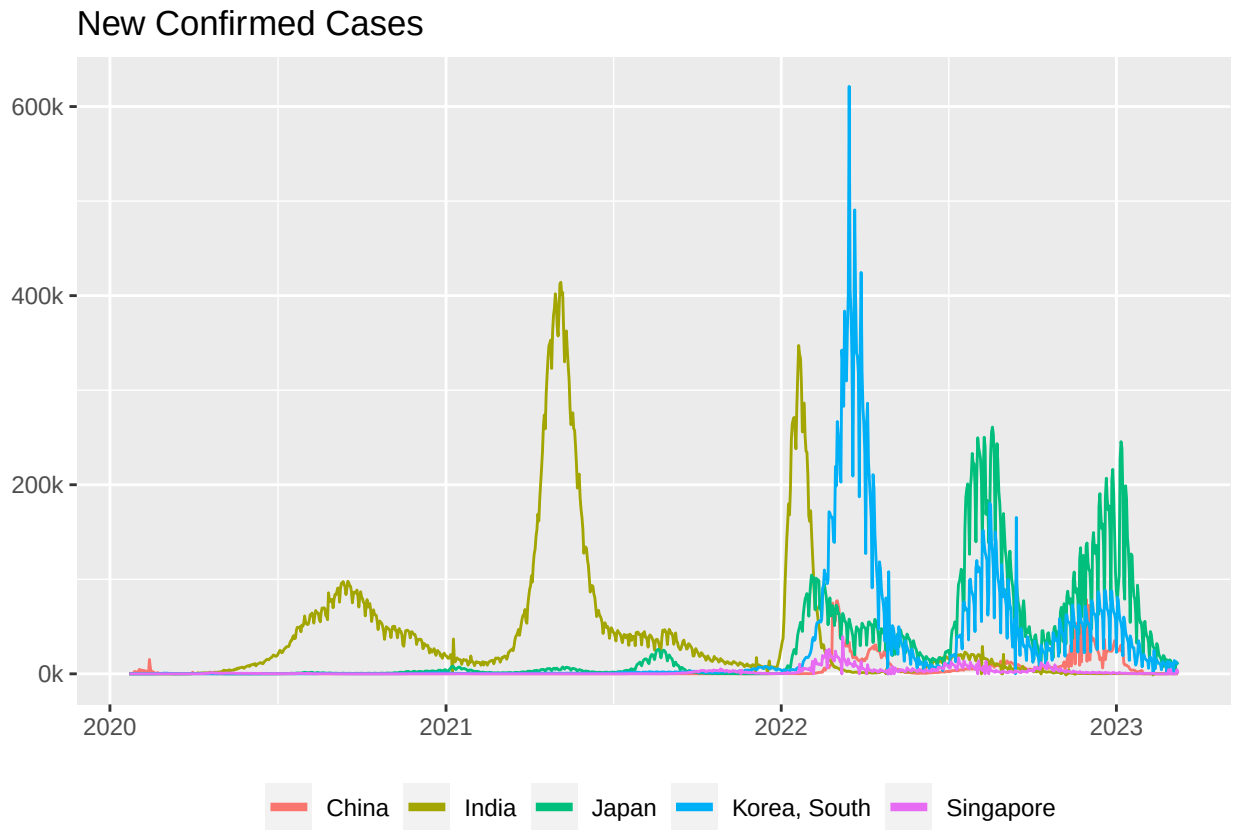
Initial Data Exploration

I picked Japan for this project since it had the highest number of confirmed new cases (in January 2023) compared to the other Asian countries such as South Korea, Singapore, China and India (see the plot below). I thought it would be more interesting to analyze.

```
global5 = global_cases %>% rename(Country_Region = "Country/Region") %>%
  filter(Country_Region %in% c("Japan", "Korea, South", "Singapore", "China", "India")) %>%
  select(-c("Province/State", Lat, Long)) %>%
  pivot_longer(-Country_Region, names_to = "Date", values_to = "Cumulative_Cases") %>%
  mutate(Date = mdy(Date)) %>% group_by(Country_Region, Date) %>%
  summarise(Cumulative_Cases = sum(Cumulative_Cases)) %>% ungroup() %>%
  arrange(Date) %>% group_by(Country_Region) %>%
  mutate(Daily_Cases = c(0, diff(Cumulative_Cases))) %>% ungroup()
```

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

```
ggplot(global5, aes(x=Date, y=Daily_Cases, group=Country_Region, color=Country_Region)) + geom_line() +
  scale_y_continuous(labels = scales::label_number(suffix="k", scale=1e-3, big.mark="")) +
  xlab(NULL) + ylab(NULL) + labs(title = "New Confirmed Cases") +
  theme(legend.title = element_blank(), legend.position = "bottom") +
  guides(color = guide_legend(override.aes = list(size = 1.5)))
```



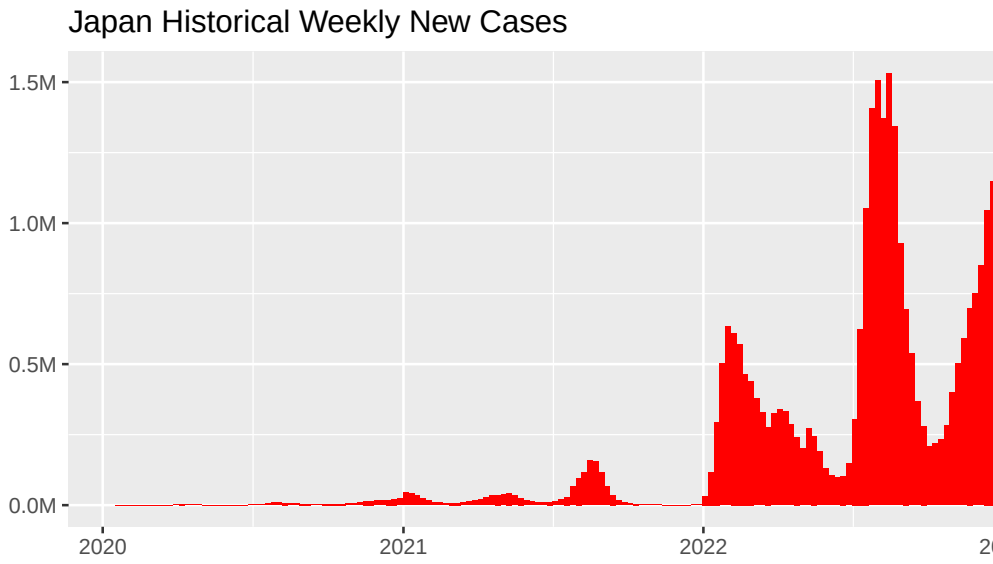
Tidying and Exploring Japan's Data

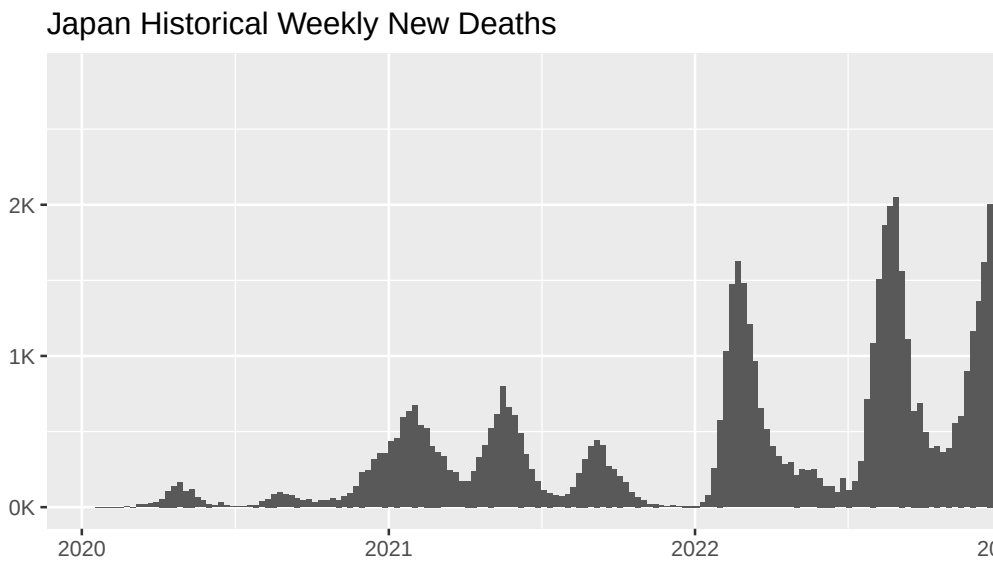
Now, since we have decided on a specific country to analyze, we will filter only rows for Japan, and get the Cumulative Cases, Cumulative Deaths, and Cumulative Doses Administered data. We will then use diff function to compute a lagged version of a time series data, shifting the time base back by 1 day of observation. The Daily Cases, Daily Deaths and Daily Doses can then be calculated as the difference between Cumulative

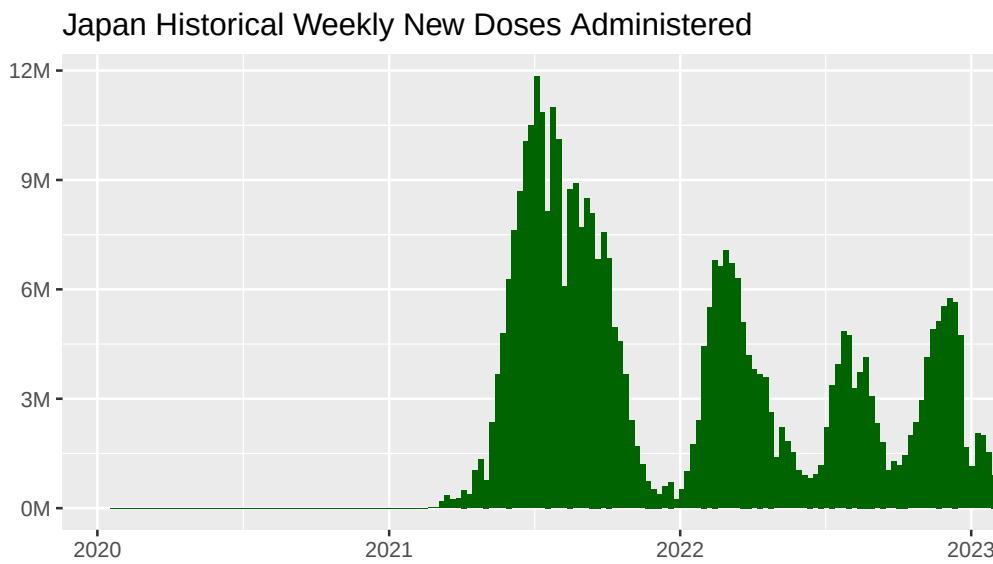
Cases and their lagged version. Because vaccine's starting date was much later than the incident's, I added zero vaccine data for the years before vaccination started in order to match the length of the other dataframes.

```
japan_cases = global_cases %>% rename(Country_Region = "Country/Region") %>%
  filter(Country_Region == "Japan") %>% select(-c("Province/State", Lat, Long)) %>%
  pivot_longer(-Country_Region, names_to = "Date", values_to = "Cumulative_Cases") %>%
  mutate(Daily_Cases = c(0, diff(Cumulative_Cases)), Date = mdy(Date))

japan_deaths = global_deaths %>% rename(Country_Region = "Country/Region") %>%
  filter(Country_Region == "Japan") %>% select(-c("Province/State", Lat, Long)) %>%
  pivot_longer(-Country_Region, names_to = "Date", values_to = "Cumulative_Deaths") %>%
  mutate(Daily_Deaths = c(0, diff(Cumulative_Deaths)), Date = mdy(Date))

japan_vaccine = japan_vaccine %>% select(c(date, total_vaccinations)) %>%
  mutate(Daily_Doses = c(0, diff(total_vaccinations))) %>% rename(Date=date)
start_date = min(japan_cases$Date)
last_date = min(japan_vaccine$Date-1)
dates = seq(start_date, last_date, "days")
df = data.frame(Date = dates, total_vaccinations = rep(0, length(dates)),
                Daily_Doses = rep(0, length(dates)))
japan_vaccine = rbind(df, japan_vaccine)
```

```
japan_cases = japan_cases %>%
  mutate(Week = as.Date(cut(Date, breaks="week")))
p1 = japan_cases %>% group_by(Week) %>% summarise(Weekly=sum(Daily_Cases)) %>%
  ggplot(aes(x=Week, y=Weekly)) +
  geom_bar(stat="identity", fill="red") +
  scale_y_continuous(labels = scales::label_number(suffix="M", scale=1e-6, big.mark="")) +
  xlab(NULL) + ylab(NULL) + labs(title = "Japan Historical Weekly New Cases") +
  guides(fill="none")

japan_deaths = japan_deaths %>%
  mutate(Week = as.Date(cut(Date, breaks="week")))
p2 = japan_deaths %>% group_by(Week) %>% summarise(Weekly=sum(Daily_Deaths)) %>%
  ggplot(aes(x=Week, y=Weekly)) +
  geom_bar(stat="identity") +
  scale_y_continuous(labels = scales::label_number(suffix="K", scale=1e-3, big.mark="")) +
  xlab(NULL) + ylab(NULL) + labs(title = "Japan Historical Weekly New Deaths")

japan_vaccine = japan_vaccine %>%
  mutate(Week = as.Date(cut(Date, breaks="week")))
p3 = japan_vaccine %>% group_by(Week) %>% summarise(Weekly=sum(Daily_Doses)) %>%
  ggplot(aes(x=Week, y=Weekly)) +
  geom_bar(stat="identity", fill="darkgreen") +
  scale_y_continuous(labels = scales::label_number(suffix="M", scale=1e-6, big.mark="")) +
  xlab(NULL) + ylab(NULL) + labs(title = "Japan Historical Weekly New Doses Administered") +
  guides(fill="none")

grid.arrange(p1, p2, p3, ncol=1)
```







Plotting Historical Weekly data

Notice that the plot shows new deaths cases began earlier than the confirmed new cases. There might have been many cases that went undetected back in 2020 when the pandemic had just begun, therefore no reports were made.

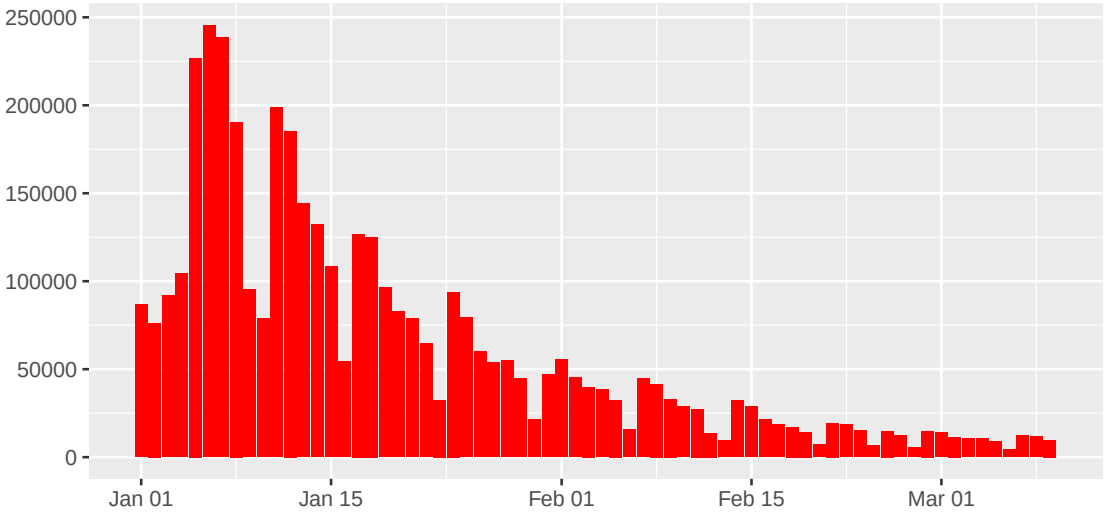
Except for the initial wave of vaccination, the number of new doses administered followed the rise and fall of the confirmed new cases. Vaccinations against the new Covid-19 variants probably coincided with that.

New Cases and New Deaths seem to have peaked at the start of January 2023. We will investigate further by looking more closely at the data in the year 2023.

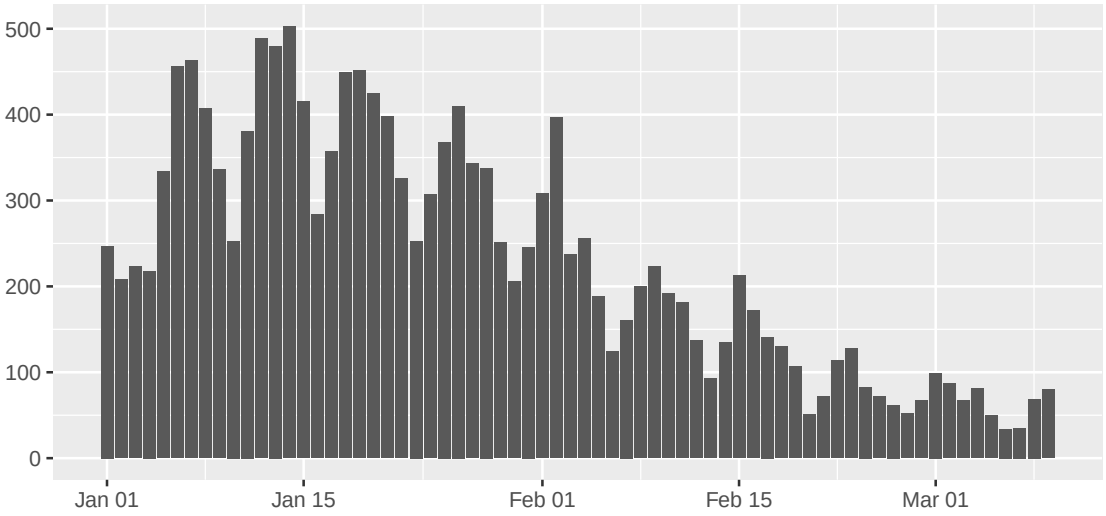
Plotting data in 2023 JHU has stopped collecting data as of 03/10/2023, and so the last date in this report is March 09, 2023. This will be our end day. We will use January 01, 2023 as our start day in the code below.

```
options(scipen = 999)
endday = max(japan_cases$Date)
startday = ymd("2023-01-01")
p4 = japan_cases %>%
  filter(between(Date, startday, endday)) %>%
  ggplot(aes(x=Date, y=Daily_Cases)) + geom_bar(stat="identity", fill="red") +
  xlab(NULL) + ylab(NULL) + labs(title = "Japan Daily New Cases in 2023") + guides(fill="none")

p5 = japan_deaths %>%
  filter(between(Date, startday, endday)) %>%
  ggplot(aes(x=Date, y=Daily_Deaths)) + geom_bar(stat="identity") +
  xlab(NULL) + ylab(NULL) + labs(title = "Japan Daily New Deaths in 2023")

p6 = japan_vaccine %>%
  filter(between(Date, startday, endday)) %>%
  ggplot(aes(x=Date, y=Daily_Doses)) + geom_bar(stat="identity", fill="darkgreen") +
  xlab(NULL) + ylab(NULL) + labs(title = "Japan Daily New Doses Administered in 2023") + guides(fill =

grid.arrange(p4, p5, p6, ncol=1)
```

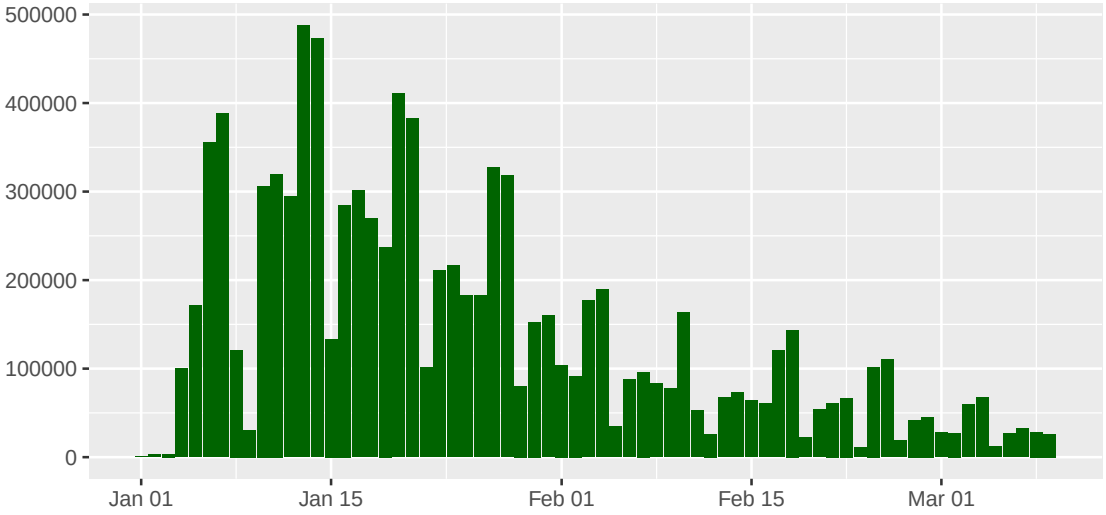
Japan Daily New Cases in 2023



Japan Daily New Deaths in 2023



Japan Daily New Doses Administered in 2023



It appears that the number of new confirmed cases peaked on January 6, 2023.

Fitting SIR Model

Data science blogger Learning Machines first posted an analysis of the Covid-19 outbreak, in which he fitted the classic SIR model to the cases in China. Tim Churches then reproduce his analysis and elaborated further the case incidence in Hubei Province, China. They used R to solve the three simultaneous differential equations which describe the classic SIR (Susceptible-Infectious-Recovered) compartmental model. For details, see: https://blog.ephorie.de/epidemiology-how-contagious-is-novel-coronavirus-2019-ncov and https://timchurches.github.io/blog/posts/2020-02-18-analysing-covid-19-2019-ncov-outbreak-data-with-r-part-1/#modelling-the-epidemic-trajectory-using-log-linear-models

The basic idea behind the SIR model for spread of disease is that each individual falls in one of the three compartments: S, I, R. Those who are healthy but susceptible to the disease (S); those that got infected, the infectious (I); and those people who have recovered (R). It then uses one of the optimizers to fit the SIR model to the observed Covid-19 cases by minimizing the residual sum of squares (RSS). This enable us to estimate the reproduction number, R0, to make predictions about the epidemic.

We will use the information from the previous exploratory data analysis to fit the SIR model. We will put our cumulative cases number back from December 01, 2022 up to January 06, 2023 (peak date) into a vector called Infected. Then we will find the values of beta and gamma that give the smallest Residual Sum of Squares (RSS) between the observed cumulative cases and the predicted cumulative cases, which represents the best fit for our data.

We will get the value of N, the initial uninfected population, from this wiki page: https://en.wikipedia.org/wiki/Demographics_of_Japan. They estimated Japan's population in 2022 to be 125,681,593. The initial value for R (Recovered) will be estimated from Japan's 2021 Recovered data, from JHU recovered data (R = 852,451).

```
SIR <- function(time, state, parameters) {
  par = as.list(c(state, parameters))
  with(par, {
    dS = -beta * I * S/N
    dI = beta * I * S/N - gamma * I
    dR = gamma * I
    list(c(dS, dI, dR))
  })}
sir_start_date = "2022-12-01"
Infected = japan_cases %>% filter(Date >= ymd("2022-12-01"), Date <= ymd("2023-01-06")) %>%
  pull(Cumulative_Cases)   #pull() -> extract a single column

# Create an increment Day vector the same length as the cases vector
Day = 1:(length(Infected))

# Specify initial values for S, I and R
N = 125681593
init = c(S = N - Infected[1], I = Infected[1], R = 852451)

# Define a function to calculate the residual sum of squares (RSS)
RSS <- function(parameters) {
  names(parameters) = c("beta", "gamma")
  out = ode(y = init, times = Day, func = SIR, parms = parameters)
  fit = out[, 3]
  sum((Infected - fit)^2)}
Opt = optim(c(0.5, 0.5), RSS, method = "L-BFGS-B", lower = c(0, 0), upper = c(1, 1))
```

```r
# check for convergence
Opt$message
```

```
## [1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```

```r
#Convergence is confirmed. Now we can examine the fitted values for beta and gamma
Opt_par = setNames(Opt$par, c("beta", "gamma"))
beta = Opt_par[1]; gamma = Opt_par[2]
Opt_par
```

```
##        beta       gamma
## 0.019619022 0.009732739
```

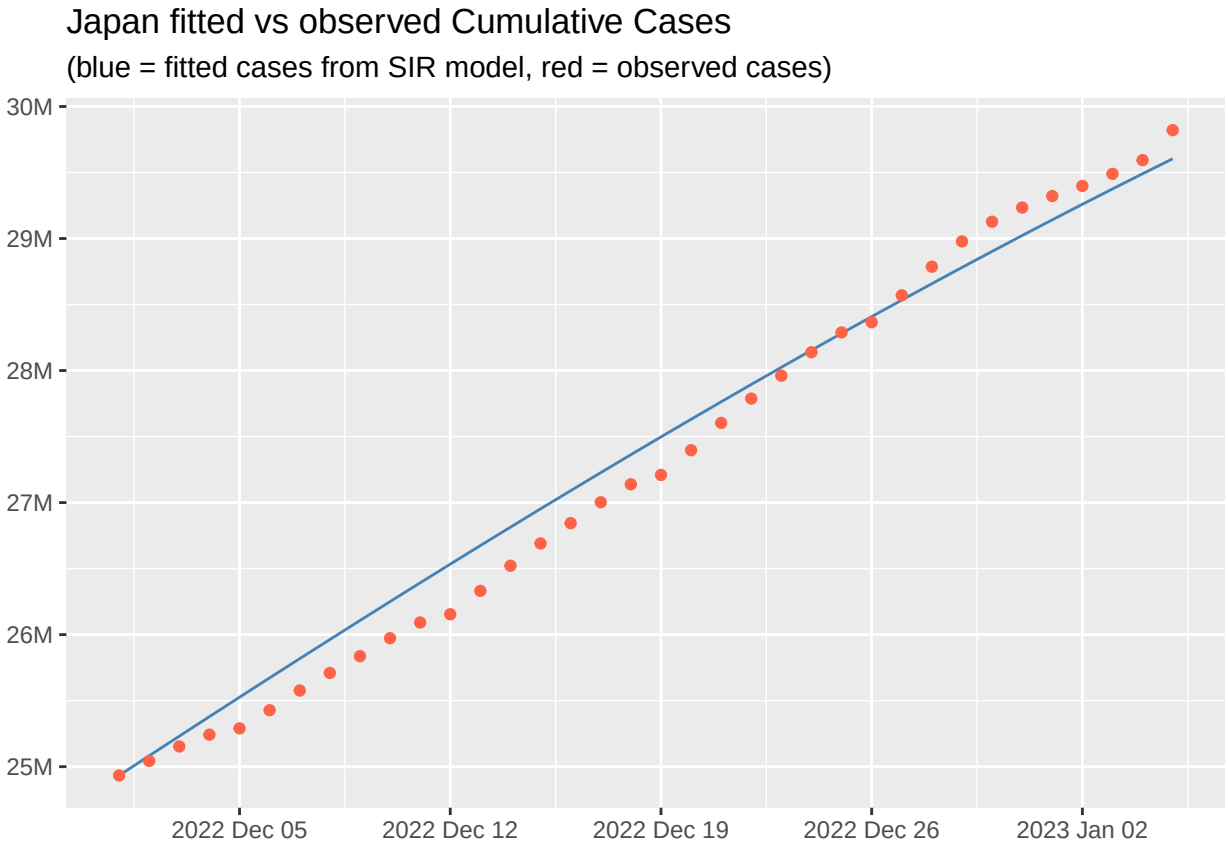
```r
t = 1:as.integer(ymd("2023-01-06") - ymd(sir_start_date))
fitted_cumulative_incidence = data.frame(ode(y = init, times = t, func = SIR, parms = Opt_par))
```

```r
japan_cases2 = japan_cases %>% select(Date, Cumulative_Cases) %>% rename(date=Date)
fitted_cumulative_incidence = fitted_cumulative_incidence %>%
  mutate(date = ymd(sir_start_date) + days(t - 1)) %>%
  left_join(japan_cases2 %>% ungroup() %>% select(date, Cumulative_Cases))
```

Plotting SIR Model

```
## Joining, by = "date"
```

```r
fitted_cumulative_incidence %>% filter(date <= ymd("2023-01-06")) %>%
  ggplot(aes(x = date)) + geom_line(aes(y = I), colour = "steelblue") +
  geom_point(aes(y = Cumulative_Cases), colour = "tomato") +
  scale_y_continuous(labels = scales::label_number(suffix="M", scale=1e-6, big.mark="")) +
  scale_x_date(date_labels = "%Y %b %d") +
  labs(y=NULL, x=NULL, title = "Japan fitted vs observed Cumulative Cases",
       subtitle = "(blue = fitted cases from SIR model, red = observed cases)")
```



As this model fits reasonably well to the observed cumulative cases data, we can use it to calculate the basic reproduction number R0, which gives the average number of susceptible people infected by each infectious individual. R0 is defined as beta over gamma (Recall that we calculated our beta and gamma earlier).

```
R0 = beta/gamma
as.numeric(R0)
```

```
## [1] 2.015776
```

An R0 of 2.0 is lower than NIH's estimate. They found the average R0 to be 3.28 and median to be 2.79

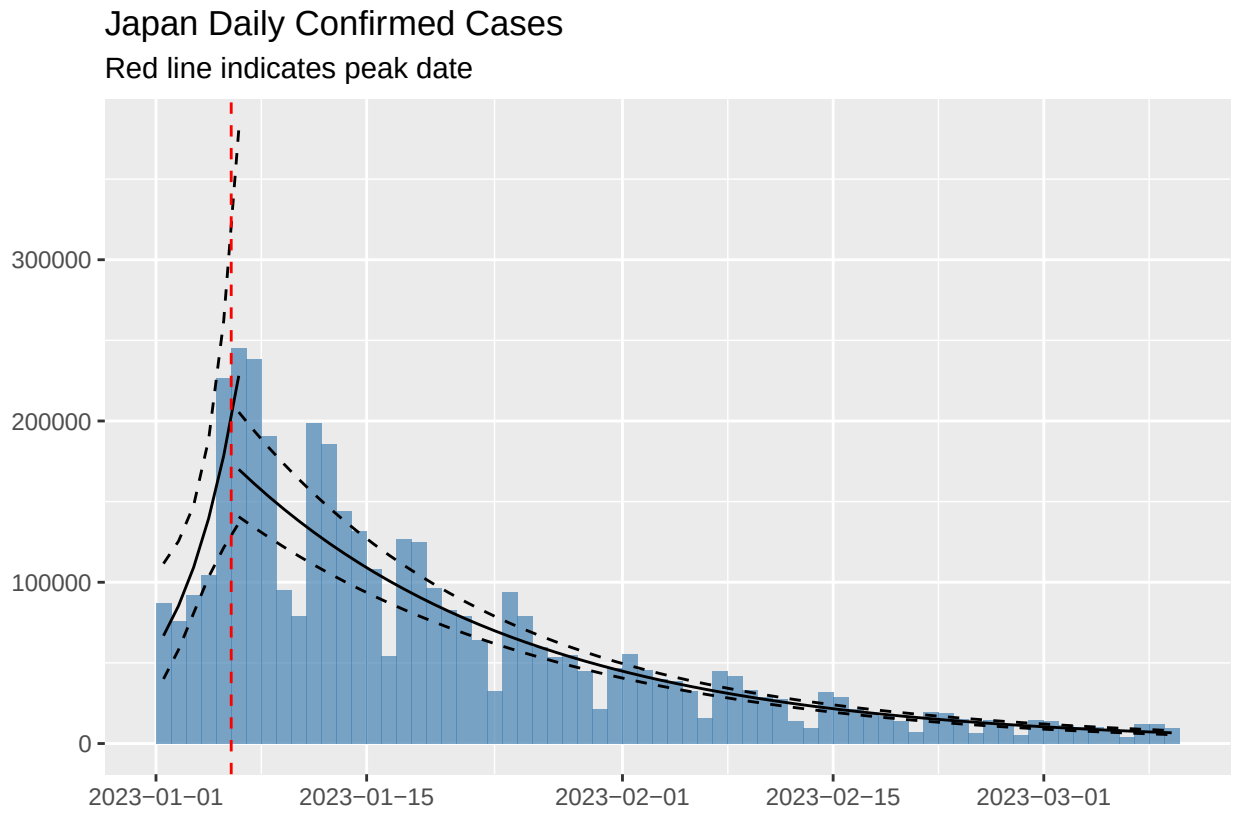
"R0 is an indication of the transmissibility of a virus, representing the average number of new infections generated by an infectious person in a totally naïve population. For R0 $>$ 1, the number infected is likely to increase, and for R0 $<$ 1, transmission is likely to die out. The basic reproduction number is a central concept in infectious disease epidemiology, indicating the risk of an infectious agent with respect to epidemic spread." https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7074654/

```
japan_incidence_function_data = japan_cases %>% select(Date, Daily_Cases) %>%
  rename(date=Date) %>% filter(date >= ymd("2023-01-01"), date <= ymd("2023-03-09")) %>%
  select(date, Daily_Cases) %>% uncount(Daily_Cases)

japan_incidence_object = incidence(japan_incidence_function_data$date)
japan_incidence_peak = find_peak(japan_incidence_object)
```

```
japan_incidence_fit = incidence::fit(japan_incidence_object, split = japan_incidence_peak)

plot(japan_incidence_object, color="steelblue") %>% add_incidence_fit(japan_incidence_fit) +
  geom_vline(xintercept = japan_incidence_peak, col = "red", lty = 2) + ylab(NULL) +
  labs(title = "Japan Daily Confirmed Cases", subtitle = "Red line indicates peak date")
```

Modeling the epidemic trajectory using log linear model

```
## Warning: It is deprecated to specify `guide = FALSE` to remove a guide. Please
## use `guide = "none"` instead.
```



From the plot above, the last wave have peaked on January 06, 2023, as printed out below.

The growth rate prior to the peak 95% Confidence Interval: (0.07563030, 0.4159337)

And the decay rate after the peak 95% Confidence Interval: (-0.05758985, -0.0469095)

```
japan_incidence_peak
```

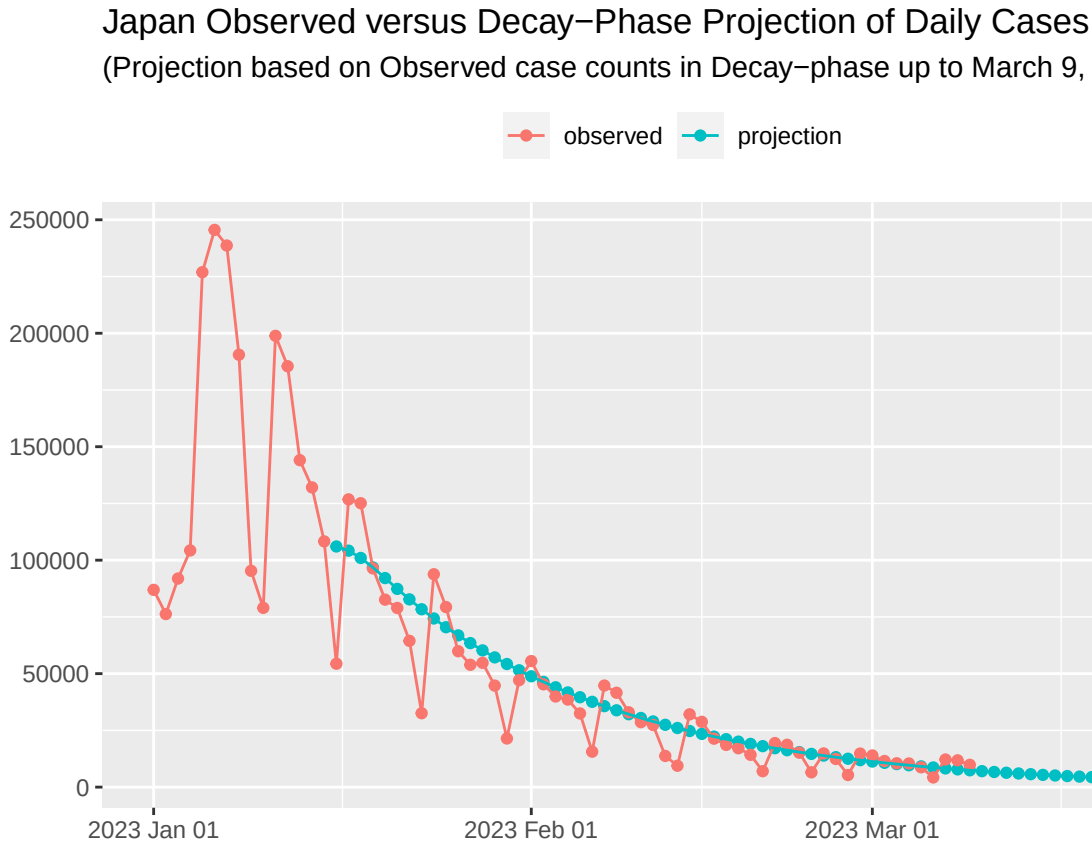
```
## [1] "2023-01-06"
```

```
get_info(japan_incidence_fit, "r.conf")
```

```
##              2.5 %      97.5 %
## before  0.07627137  0.41456934
## after  -0.05752590 -0.04697411
```

We can then use this model of the epidemic trajectory to estimate the reproduction number in the decay phase of the epidemic.

```
mu = 7.5; sigma = 3.4
param = gamma_mucv2shapescale(mu, sigma/mu)
w = distcrete("gamma", interval = 1, shape = param$shape, scale = param$scale, w = 0)

#Decay phase
decay_R0 = lm2R0_sample(japan_incidence_fit$after$model, w)
set.seed(100)
pred_days = 70
test_pred_growth = project(japan_incidence_object[1:15],
                           R = median(decay_R0), si = w, n_days = pred_days, n_sim = 1000)

# Convert test_pred_growth matrix to a data frame and get median cases for all the simulations for each
test_pred_growth_median_counts = test_pred_growth %>% as.data.frame() %>%
  pivot_longer(-dates, names_to = "simulation", values_to = "incidence") %>%
  group_by(dates) %>% summarise(incident_cases = as.integer(median(incidence))) %>%
  mutate(data_type = "projection")

test_pred_growth_median_counts %>%
  bind_rows(tibble(dates = get_dates(japan_incidence_object),
                   incident_cases = get_counts(japan_incidence_object), data_type = "observed")) %>%
  ggplot(aes(x = dates, y = incident_cases, colour = data_type)) + geom_point() + geom_line() +
  scale_x_date(date_labels = "%Y %b %d") +
  labs(x=NULL, y=NULL, title = "Japan Observed versus Decay-Phase Projection of Daily Cases",
       subtitle = "(Projection based on Observed case counts in Decay-phase up to March 9, 2023)") +
  theme(legend.position = "top", legend.title = element_blank())
```



Decay-Phase Projection

Conclusion

Japan appears to have contained the latest outbreak of Covid-19. New cases peaked on January 06, 2023. Since then, they have declined steadily. Our projections on the decay-phase, which is based on the log linear model, seems to match the observed cases reasonably well. This projection is valuable for planning purposes. Planned interventions such as border controls, public transportation restrictions, mandatory social distancing, etc., may need to be maintained for a period of time as a result of this projection. Based on our projection plot, it will take at least until April 2023 for the outbreak to 'die down'. As of April 03, 2023, the news indicates Japan will cease Covid-19 border control measures on May 8, 2023.

Using math in statistical modeling to find out the trajectory of the outbreak is one of the factor that informed policy makers. At the end of the day, it may well be heuristics, and not math, that are guiding policy-making.

Potential Bias and Mitigation If we expect to discover specific results when analyzing data, we may be biased towards data that supports our expectations.

For this reason, I tried to review data without any preconceived notions and searched for evidence that disprove my assumptions.

Session Info

```
sessionInfo()
```

```
## R version 4.2.3 (2023-03-15 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
```

```
## Running under: Windows 10 x64 (build 19044)
##
## 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
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] projections_0.6.0 distcrete_1.0.3   epitrix_0.4.0     incidence_1.7.3
##  [5] deSolve_1.35      gridExtra_2.3     lubridate_1.8.0   forcats_0.5.2
##  [9] stringr_1.4.1     dplyr_1.0.10      purrr_0.3.5       readr_2.1.3
## [13] tidyr_1.2.1       tibble_3.1.8      ggplot2_3.3.6     tidyverse_1.3.2
##
## loaded via a namespace (and not attached):
##  [1] assertthat_0.2.1    digest_0.6.29       utf8_1.2.2
##  [4] R6_2.5.1            cellranger_1.1.0    backports_1.4.1
##  [7] reprex_2.0.2        evaluate_0.17       highr_0.9
## [10] httr_1.4.4          pillar_1.8.1        rlang_1.0.6
## [13] curl_4.3.3          googlesheets4_1.0.1 readxl_1.4.1
## [16] rstudioapi_0.14     rmarkdown_2.17      labeling_0.4.2
## [19] googledrive_2.0.0   bit_4.0.4           munsell_0.5.0
## [22] broom_1.0.1         compiler_4.2.3      modelr_0.1.9
## [25] xfun_0.37           pkgconfig_2.0.3     htmltools_0.5.3
## [28] tidyselect_1.2.0    fansi_1.0.3         crayon_1.5.2
## [31] tzdb_0.3.0          dbplyr_2.2.1        withr_2.5.0
## [34] grid_4.2.3          jsonlite_1.8.2      gtable_0.3.1
## [37] lifecycle_1.0.3     DBI_1.1.3           magrittr_2.0.3
## [40] scales_1.2.1        cli_3.4.1           stringi_1.7.8
## [43] vroom_1.6.0         farver_2.1.1        fs_1.5.2
## [46] xml2_1.3.3          ellipsis_0.3.2      generics_0.1.3
## [49] vctrs_0.4.2         tools_4.2.3         bit64_4.0.5
## [52] glue_1.6.2          hms_1.1.2           parallel_4.2.3
## [55] fastmap_1.1.0       yaml_2.3.5          colorspace_2.0-3
## [58] gargle_1.2.1        rvest_1.0.3         knitr_1.40
## [61] haven_2.5.1
```