Progress report

Kevin Chen Zixuan (Niki) Chen Lauren Dimaggio Jingxuan Fan

All script used to download and clean the data, conduct analyses, produce output, and prepare documents may be found on our GitHub repository here. The repository is organized into five main directories, whose contents reflect their names:

- `data` contains raw data as well as cleaned subsets
- `output` contains descriptive, analytic, and diagnostic results as well as intermediate objects produced over the course of work
- `references` contains scientific/statistical/technical literature as well as a BibTeX file enumerating them
- `reports` contains prepared documents for human consumption and the script used to prepare them
- `script` contains the programming script for all main tasks.

The present document describes some of the work done to-date.

Downloading and cleaning data

Data were downloaded directly from the California Health and Human Services (CalHHS) Open Data Portal. Two main datasets were used: (1) Statewide COVID-19 Cases Deaths Tests and (2) Statewide COVID-19 Vaccines Administered by County vaccination counts over time. Both datasets contain daily time series stratified by county. The script for downloading the data are saved in file `script/download_data.R` and reproduced below.

```
# download_data.R
# February 9, 2023
# Downloading data from CA Open Data portal

library(data.table)

# Case counts
```

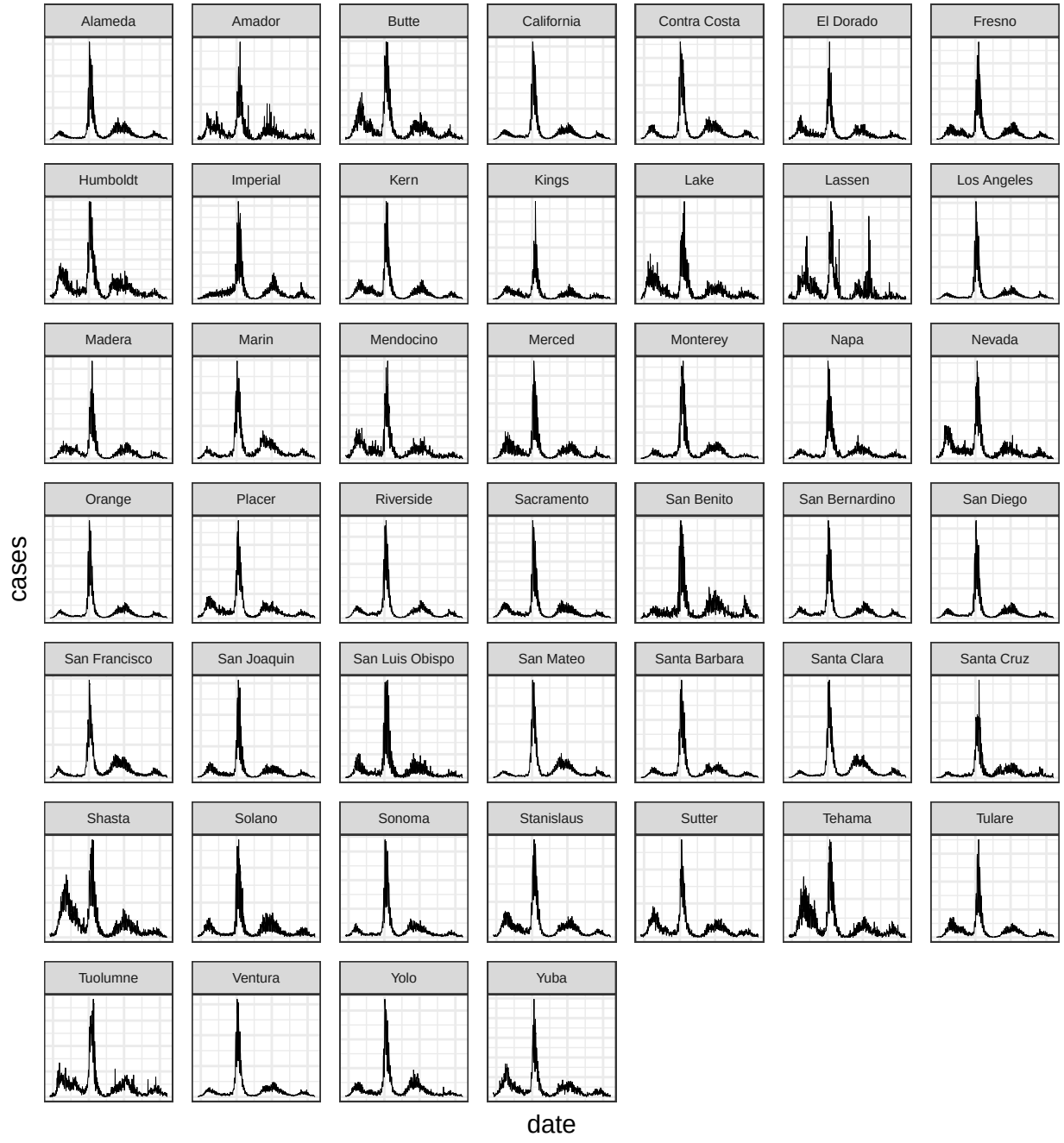
```r
cases.url <- paste0(
    "https://data.chhs.ca.gov/dataset/",
    "f333528b-4d38-4814-bebb-12db1f10f535/",
    "resource/",
    "046cdd2b-31e5-4d34-9ed3-b48cdbc4be7a/",
    "download/covid19cases_test.csv"
)
cases.dat <- fread(cases.url)
write.csv(cases.dat, 'data/cases.csv', row.names = F)

# Vaccination counts
vax.url <- paste0(
    "https://data.chhs.ca.gov/dataset/",
    "e283ee5a-cf18-4f20-a92c-ee94a2866ccd/",
    "resource/",
    "130d7ba2-b6eb-438d-a412-741bde207e1c/",
    "download/covid19vaccinesbycounty.csv"
)
vax.dat <- fread(vax.url)
write.csv(vax.dat, 'data/vax.csv', row.names = F)
```

Since the total data are small and clean, minimal processing was required. We limited the data to days on or after June 15, 2021 to restrict the analytic problem to the period of time when vaccines were widely available. The main workhorses for our analysis will be the daily new case, death, and vaccination counts. While death and vaccination counts should be accurate (difficult to mis-classify a death and vaccination requires billing), the case counts may be vulnerable to changes in reporting and testing. In the next several weeks, we will explore the use of hospitalization rates, test counts, and test positivity to construct a more accurate measure of the true case count.

Basic visualization

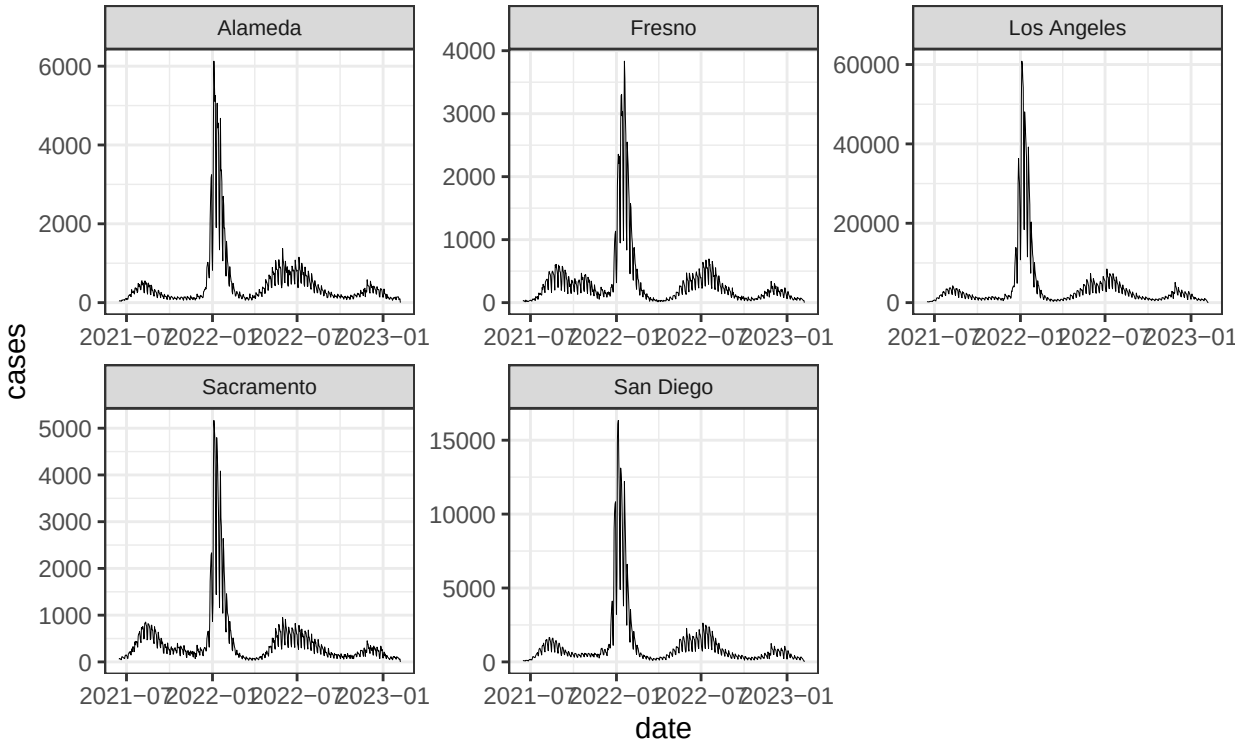
A plot of the daily case counts stratified by county is presented below. Note that for almost all of the counties that experienced at least 10 000 cases over the course of available data, there were 4-5 outbreak cycles where the case count underwent exponential growth before rapidly switching to exponential decay with what appears to be a sudden change in the velocity. Case count plots for the major counties of Alameda, San Francisco, Sacramento, Los Angeles, San Diego, and Fresno are highlighted in their own plot. Despite the differences in the magnitude of the peaks in the

epidemiologic trajectories, the overall shape of the curves is subjectively similar. The ways in which these cycles can be represented as parameters of a compartmental model are discussed below.



Basic epidemiologic principles and compartmental modeling.

The study of disease distributions and infectious disease transmission have long be central to public health research and practice. Before the wide acceptance of germ theory, early epidemiologists such as John Snow identified causes of health and disease by analyzing binary contrasts in exposure status

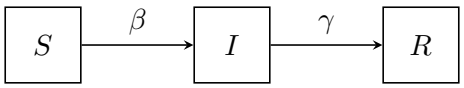


without a clear model of the biological manifestation of those exposures (Vinten-Johansen et al. 2003). By the 1910s, physicians and researchers became comfortable with the notion of the *natural history of disease* ie the progression of a disease process in an individual over time. The most basic conceptualization of the natural history of an infectious disease is an individual's progression from susceptibility to infected and finally, recovery. This model forms the basis for one of the simplest of compartmental models used for modeling disease trajectory in a population: the SIR model (Harko, Lobo, and Mak 2014). The flow of individuals through the states in a closed population is modeled as systems of ordinary differential equations:

$$\begin{cases} \frac{d}{dt}S = -\frac{\beta}{N}IS \\ \frac{d}{dt}I = \frac{\beta}{N}IS - \gamma I \\ \frac{d}{dt}R = \gamma I \end{cases}$$

This approach gives rise to parameters that are convenient for both estimation and interpretation. The transmission parameter β and the force of infection γ represent rates of change of the S and I states respectively. The transmission parameter and the force of infection are related through a constant R_0 such that $\beta = R_0 \times \gamma$ where R_0 may be interpreted as the average number of new cases (infections) that arise from contact with a case. This parameter is known as the basic reproduction number.

Figure 1: SIR model.



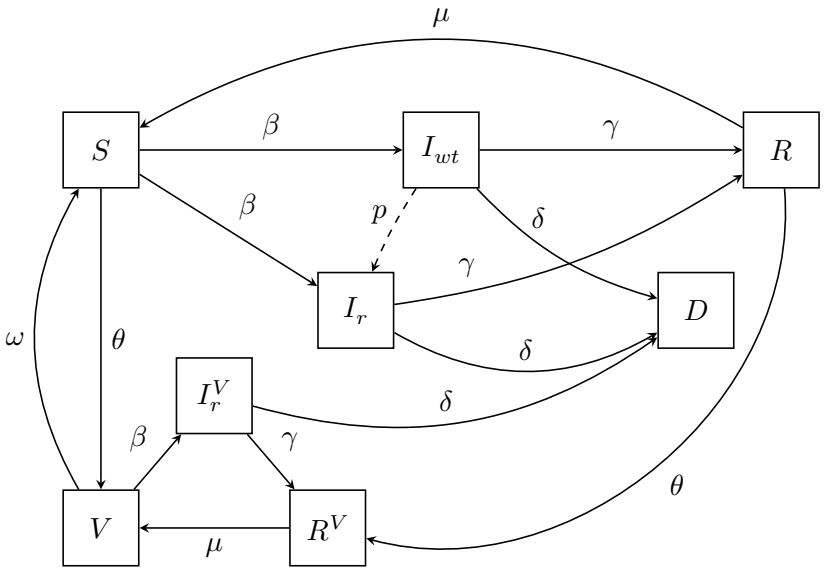
Solutions at analyst-defined parameter values

Well known extensions of the SIR model include $SIRS$, $SEIR$ and $SEIRS$ (Vynnycky and White 2010, 16). One extension presented by Rella et al. (2021) included five additional compartments to account for vaccination, transmission of a vaccine-resistant strain, and death. Under that compartmental model, emergence of a vaccine-resistant strain was governed by a rate parameter p, which served as a rate parameter for the instantaneous emergence of a vaccine-resistant strain among those infected with a wild type (non-resistant) strain. Note that setting $p = 0$ results in a four compartment $SIRD$ model with an additional compartment V for vaccination. Below, we present the ordinary differential equations for the model presented by Rella et al. (2021), but further extended to allow for flow from the vaccinated compartment back into the susceptible compartment with rate ω. Note that here, we take $\beta \times N$ to be the transmission parameter.

$$
\begin{cases}
\frac{d}{dt}S = \mu R + \omega V - \theta S - \beta(I_{wt} + I_r + I_r^V)S \\
\frac{d}{dt}I_{wt} = -(\gamma + \delta)I_{wt} + \beta S(I_{wt}) \\
\frac{d}{dt}I_r = -(\gamma + \delta)I_r + \beta S(I_r + I_r^V) \\
\frac{d}{dt}I_r^V = -(\gamma + \delta)I_r^V + \beta V(I_r + I_r^V) \\
\frac{d}{dt}R = -\mu R - \theta R + \gamma(I_{wt} + I_r) \\
\frac{d}{dt}R^V = -\mu R^V + \theta R + \gamma I_r^V \\
\frac{d}{dt}D = \delta(I_{wt} + I_r + I_r^V) \\
\frac{d}{dt}V = \mu R^V + \theta S - \beta V + (I_r + I_r^V) - \omega V
\end{cases}
$$

The waves in the disease trajectory over time may be captured by a time-varying reproduction number. Rella et al. (2021) defined two basic reproduction numbers, one for exponential growth and one for exponential decay. They theorized that the disease trajectory would begin with a period of exponential growth until the prevalence of disease reaches F_h, when the basic reproduction number would switch to one below 1. This period of exponential decay would resume until the prevalence is at the low threshold F_l. This conceptualization of the relationship between the transmission parameter

Figure 2: Compartmental model of Rella et alia (2021), extended to allow vaccinated individuals to return to being susceptible.



and time makes sense because the outbreak cycles do not appear to have smooth second derivatives. Differences in heights of the outbreak cycles may be explained by different thresholds F_h and F_l over different time periods.

Solutions at analyst-defined parameter values

We encoded the ordinary differential equations described in the previous section in the file `script/01-compartments.R` (and below). In addition to the differential equations wrapped in the function `compartmetal_model()`, we also have helper function `get_beta()`, for computing the time-varying value of the transmission parameter based on the prevalence of disease (sum of the I compartments). The value of the transmission parameter is necessary for predicting new cases on each day:

$$\text{new cases} = \beta(S(I_{wt} + I_r + I_r^V) + V(I_r + I_r^V))$$

For San Francisco ($N = 815\,201$), we can plot the solution of the model above given plausible (but arbitrarily specified) parameter values.

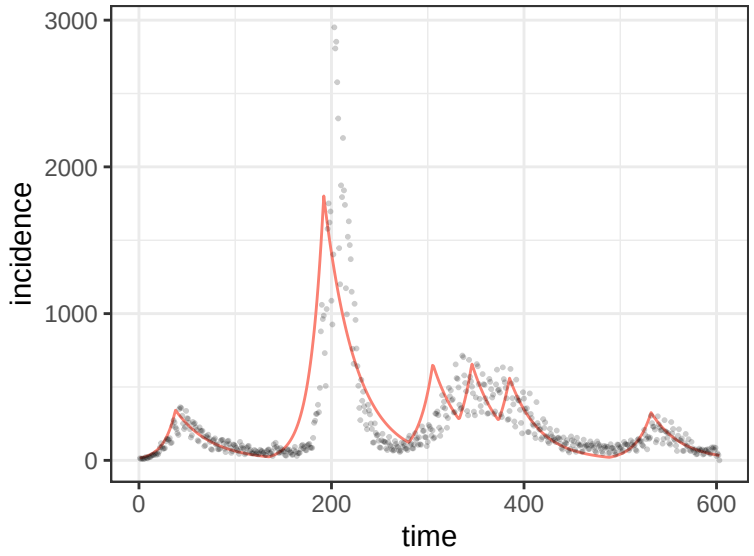
Parameter	Value	Interpretation
h	0.2	Percent of population without vaccination
k	0.01	Saturation of vaccination speed
p	0	Emergence rate

Parameter	Value	Interpretation
θ	250.6335	Vaccination rate (vaccines per day)
δ	7.3 $\times$ 10,000	Death rate (deaths per day)
t_1	160.5	Start of second wave (2021-11-21)
t_2	305.5	Start of third wave (2021-04-15)
t_3	385.5	Start of fourth wave (2021-07-04)
R_{0_h}	2.2	Basic reproduction number during growth
R_{0_l}	0.65	Basic reproduction number during decay
γ	1 / 14	Inverse disease duration (days)
μ	1 / (30 * 12)	Inverse duration of natural immunity
ω	1 / (30 * 12)	Inverse duration of vaccine protection
h_1	350	Maximum inverse prevalence in first wave
h_2	63	Maximum inverse prevalence in second wave
h_3	165	Maximum inverse prevalence in third wave
h_4	325	Maximum inverse prevalence in fourth wave
l_1	15	Fold reduction required before growth (1st)
l_2	15	Fold reduction required before growth (2nd)
l_3	2.5	Fold reduction required before growth (3rd)
l_4	17	Fold reduction required before growth (4th)

The script for plotting a deterministic solution with these parameters can be found in `script/eyeball-fit.R` (and below). Such a plot is presented below with the salmon-colored line representing the predicted incidence and the points reprenting the observed new case count.

```
-- Attaching packages --------------------------------------- tidyverse 1.3.2 --
v tibble  3.1.8     v dplyr   1.1.0
v tidyr   1.3.0     v stringr 1.5.0
v readr   2.1.3     v forcats 0.5.2
v purrr   1.0.1
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::between()   masks data.table::between()
x dplyr::filter()    masks stats::filter()
x dplyr::first()     masks data.table::first()
x dplyr::lag()       masks stats::lag()
x dplyr::last()      masks data.table::last()
x purrr::transpose() masks data.table::transpose()
```

```
# state_model.R
# February 9, 2023
# Simple ODE implementation
```




```r
library(tidyverse)
library(data.table)
library(deSolve)

# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Derivatives function for closed compartmental model:
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
compartmental_model <- function(time, state = initial_state, parameters) {
    # Parameters:
    F_h     <- parameters['F_h']
    gamma   <- 1 / parameters["disease_duration"]
    delta   <- parameters["death_rate"]
    theta0  <- parameters["vaccination_rate"]
    w       <- 1 / parameters["immune_period"]
    mu      <- 1 / parameters["recovery_period"]
    p       <- 0
    R0_high <- parameters["R0_high"]
    R0_low  <- parameters["R0_low"]
    tighten_factor1   <- parameters["tighten_factor1"]
    tighten_factor2   <- parameters["tighten_factor2"]
    tighten_factor3   <- parameters["tighten_factor3"]
    tighten_factor4   <- parameters["tighten_factor4"]
    loosen_factor1   <- parameters["loosen_factor1"]
    loosen_factor2   <- parameters["loosen_factor2"]
    loosen_factor3   <- parameters["loosen_factor3"]
```

```
loosen_factor4   <- parameters["loosen_factor4"]
# as.integer(as.Date('2021-11-21') - as.Date('2021-06-14'))
holiday_date     <- parameters["holiday_date"]
# as.integer(as.Date('2022-04-15') - as.Date('2021-06-14'))
reopening_date   <- parameters["reopening_date"]
# as.integer(as.Date('2022-07-04') - as.Date('2021-06-14'))
summer_date      <- parameters["summer_date"]
emergence_date      <- parameters["emergence_date"]
h       <- parameters["p_non_vax"]
k       <- parameters["saturation"]
dt      <- parameters["dt"]
R0     <- get('R0', envir = .GlobalEnv)
# States:
S    <- state["S"]
I_wt <- state["I_wt"]
I_r  <- state['I_r']
I_rV <- state["I_rV"]
R    <- state["R"]
RV   <- state["RV"]
D    <- state["D"]
V    <- state["V"]
N <-  S + I_wt + I_r + I_rV + R + RV + D + V
# Time-varying parameters
## Vaccination rate
theta <- (1 - h / (S + R + I_wt + I_r)) * theta0 / (S + R + k)
## Time-varying force of  infection
if (time > 0)   {F_h <- N / tighten_factor1; loosen_factor <- loosen_factor1}
if (time >= holiday_date) {
    F_h <- N / tighten_factor2; loosen_factor <- loosen_factor2}
if (time >= reopening_date) {
    F_h <- N / tighten_factor3; loosen_factor <- loosen_factor3}
if (time >= summer_date) {
    F_h <- N / tighten_factor4; loosen_factor <- loosen_factor4}
if ((I_wt + I_r + I_rV) > F_h & R0 > R0_low)                    {R0 <<- R0_low}
if ((I_wt + I_r + I_rV) < (F_h / loosen_factor) & R0 < R0_high)  {R0 <<- R0_high}
beta  <- get('R0', envir = .GlobalEnv) * gamma / N
# Resistant strain?
if ((I_r + I_rV) <= 1000 * N & time >= emergence_date) {
    p <- parameters['emergence_rate']
```

```
            if (parameters['stochastic']) {
            n_to_r  <<- rpois(1, dt * p * max(0, I_wt))
            n_to_wt <<- rpois(1, dt * p * max(0, I_r))
            } else {
                n_to_r  <<- dt * p * max(0, I_wt)
                n_to_wt <<- dt * p * max(0, I_r)
            }
        I_r  <- I_r  - n_to_wt + n_to_r
        I_wt <- I_wt + n_to_wt - n_to_r
    }
    # Derivatives:
    dS    <- mu * R + w * V - theta * S - beta * (I_wt + I_r + I_rV) * S
    dI_wt <- - (gamma + delta) * I_wt + beta * S * (I_wt)
    dI_r  <- (- (gamma + delta) * I_r  + beta * S * (I_r + I_rV))
    dI_rV <- - (gamma + delta) * I_rV + beta * V * (I_r + I_rV)
    dR    <- - mu * R - theta * R + gamma * (I_wt + I_r)
    dRV   <- - mu * RV + theta * R + gamma * I_rV
    dD    <- delta * (I_wt + I_r + I_rV)
    dV    <- mu * RV + theta * S - beta * V + (I_r + I_rV) - w * V
    return(list(c(dS, dI_wt, dI_r, dI_rV, dR, dRV, dD, dV)))
}


# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Compute what force of infection should be given state
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
get_beta <- function(traj, parameters) {
    beta <- rep(NA, nrow(traj))
    tighten_factor <- beta
    tighten_factor[traj[,'time'] > 0]    <- parameters['tighten_factor1']
    tighten_factor[traj[,'time'] >= 160] <- parameters['tighten_factor2']
    tighten_factor[traj[,'time'] >= 305] <- parameters['tighten_factor3']
    tighten_factor[traj[,'time'] >= 410] <- parameters['tighten_factor4']
    loosen_factor <- beta
    loosen_factor[traj[,'time'] > 0]    <- parameters['loosen_factor1']
    loosen_factor[traj[,'time'] >= 160] <- parameters['loosen_factor2']
    loosen_factor[traj[,'time'] >= 305] <- parameters['loosen_factor3']
    loosen_factor[traj[,'time'] >= 410] <- parameters['loosen_factor4']
    R0 <- c(beta, NA)
```

```
    R0[1] <- 0
    for (i in 1:length(beta)) {
        I <- (traj[i, 'I_wt'] + traj[i, 'I_r'] + traj[i, 'I_rV'])
        R0[i + 1] <- R0[i]
        if (I > N / tighten_factor[i] & R0[i] > parameters['R0_low']) {
            R0[i + 1] <- parameters['R0_low']}
        if (I < (N / tighten_factor[i] / loosen_factor[i]) & R0[i] < parameters['R0_high'])
            R0[i + 1] <- parameters['R0_high']}
    }
    return(R0[-1] / parameters['disease_duration'] / N)
}
```

```
# initial_values.R
# February 22, 2023
# Starting values for ODE and parameters (chosen using epidemiologic knowledge)

# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Initial conditions
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
N <- 815201
initial_state <- c(
    S = N - 100,
    I_wt = 100,
    I_r = 0,
    I_rV = 0,
    R = 0,
    RV = 0,
    D = 0,
    V = 0)

# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Parameters
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
parameters <- c(
    p_non_vax = 0.2,
    saturation = 0.01,
    emergence_rate = 0,
    vaccination_rate = 250.6335,
    death_rate = 7.3e-4,
```

```
    dt = 1,
    stochastic = 0,
    holiday_date =  160.5, # as.integer(as.Date('2021-11-21') - as.Date('2021-06-14')),
    reopening_date =  305.5, # as.integer(as.Date('2022-04-15') - as.Date('2021-06-14')),
    summer_date =  385, # as.integer(as.Date('2022-07-04') - as.Date('2021-06-14'))
    emergence_date = 161.5, # as.integer(as.Date('2021-11-30') - as.Date('2021-06-14')),
    # Random parameters
    R0_high = 2.2,
    R0_low = 0.65,
    disease_duration = 14,
    recovery_period = 30 * 12,
    immune_period = 30 * 12,
    tighten_factor1 = 350,
    tighten_factor2 = 63,
    tighten_factor3 = 165,
    tighten_factor4 = 325,
    loosen_factor1 = 15,
    loosen_factor2 = 15,
    loosen_factor3 = 2.5,
    loosen_factor4 = 17
)
```

```
# eyeball-fit.R
# February 21, 2023
# Running model with heuristic choice of parameter values

source('script/01-compartments.R')
source('script/03-initial_values.R')
san_francisco.dat <- fread("data/san_francisco.csv")
san_francisco.dat <- san_francisco.dat[date >= as.Date('2021-06-15')]
san_francisco.dat[,time := 1:.N]

# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# ODE solution
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
times <- seq(from = 1, to = 603, by = parameters['dt'])
R0 <<- 0
set.seed(222)
# parameters['emergence_rate'] <- 1/10
```

```r
# initial_state["I_r"] <- 0
# parameters['emergence_date']    <- 0
# parameters['holiday_date'] + 20
# parameters['holiday_date']    <- Inf
# parameters['reopening_date']  <- Inf
# parameters['summer_date']     <- Inf
trajectory.ode <- as.data.frame(ode(
    y = initial_state,
    times = times,
    parms = parameters,
    func = compartmental_model,
    method = "lsode"))
# rm(list = c('n_to_r', 'n_to_wt', 'R0'))

beta <- get_beta(trajectory.ode, parameters)
trajectory.ode$incidence <- beta * with(
    trajectory.ode, S * (I_wt + I_r + I_rV) + V * (I_r + I_rV))
trajectory.ode$incidence_wt <- beta * with(
    trajectory.ode, S * I_wt)
trajectory.ode$incidence_r <- beta * with(
    trajectory.ode, S * (I_r + I_rV) + V * (I_r + I_rV))

# The first few entries of the trajectory matrix:
trajectory.ode %>%
    ggplot(aes(x = time)) +
    geom_path(aes(y = incidence, col = "Total"), shape = 2, alpha = 0.6) +
    geom_path(aes(y = incidence_wt, col = "WT"), shape = 2, alpha = 0.6) +
    geom_path(aes(y = incidence_r, col = "R"), shape = 2, alpha = 0.6) +
    # geom_path(aes(y = I_wt + I_r + I_rV), shape = 2) +
    # geom_point(data = san_francisco.dat,
    #                    aes(y = cases), size = 1/.pt) +
    geom_point(data = san_francisco.dat,
                        aes(y = deaths), size = 1/.pt) +
    # hospitalization rate?
    coord_cartesian(ylim = c(0, 10)) +
    theme_bw()
```

References

Harko, Tiberiu, Francisco SN Lobo, and MK3197716 Mak. 2014. "Exact Analytical Solutions of the Susceptible-Infected-Recovered (SIR) Epidemic Model and of the SIR Model with Equal Death and Birth Rates." *Applied Mathematics and Computation* 236: 184–94.

Rella, Simon A, Yuliya A Kulikova, Emmanouil T Dermitzakis, and Fyodor A Kondrashov. 2021. "Rates of SARS-CoV-2 Transmission and Vaccination Impact the Fate of Vaccine-Resistant Strains." *Scientific Reports* 11 (1): 15729.

Vinten-Johansen, Peter, Howard Brody, Nigel Paneth, Stephen Rachman, Michael Rip, and David Zuck. 2003. *Cholera, Chloroform, and the Science of Medicine: A Life of John Snow*. Oxford University Press.

Vynnycky, Emilia, and Richard White. 2010. *An Introduction to Infectious Disease Modelling*. OUP oxford.