TMED917 2021 Set 1

Lassi Virtanen

04/2021

Task 1 - Mathematical modeling

Item a

A mathematical model of an infectious disease describes how the disease propagates through a population. This is usually done with (systems of) ordinary or partial differential equations. At the simplest, such model could try to describe and predict how many people would be infected at any time t. More robust models continue from this by considering a plethora of ''error" terms which aim to reduce the model's deviation from an underlying, ''biological" function.

Item b

The three functions $S(t)$, $I(t)$ and $R(t)$ would arguably be the key properties of SIR. In a mathematical sense the parameters β and γ are, however, as essential as the functions. These are the parameters that are usually to be fitted into data during the earliest stages of an epidemic - thus dictating the efficiency of the model. I'll give the reproductive number R_t also a special word as it has been a really important number during Covid-19.

Item c

The model's downsides are that it assumes that the population is randomly mixed, homogenous (in terms of behaviour), and large. The authors also argue that the exponential distribution is unrealistic since *e.g.*, people do not fall ill immediately. However, the model does usually a fairly good job at modeling how infections propagate since it applies well to many situations.

Item d

Vaccines, treatment of infectives, social distancing and lockdowns, and efficient politics in concert with reliable and precise health data and rigorous modeling of the disease.

Item e

The authors express many concers. First of all, even though many models may produce similar outcomes, sometimes they don't. This can lead to wrong predictions and models (and by this I mean: all models are wrong, but we should try to minimize how wrong they are). Moreover, even if the models were effective and accaptable, the data, they were formed from could have been compiled to not represent the real world. This leads to the ancient failure of creating a crap model to represent crap data. Not good. Lastly, no model can be precise on every aspect. Every model is a product of a trade-off between precision and applicability. For example, the SIR model does not produce reliable quantitative estimates about vaccines. Thus, such dilemmata should be considered with jointly evaluating a handful of other models.

Task 2 - Modeling the spread of a viral infection

Item a

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(viridis)
```

```
## Loading required package: viridisLite
```

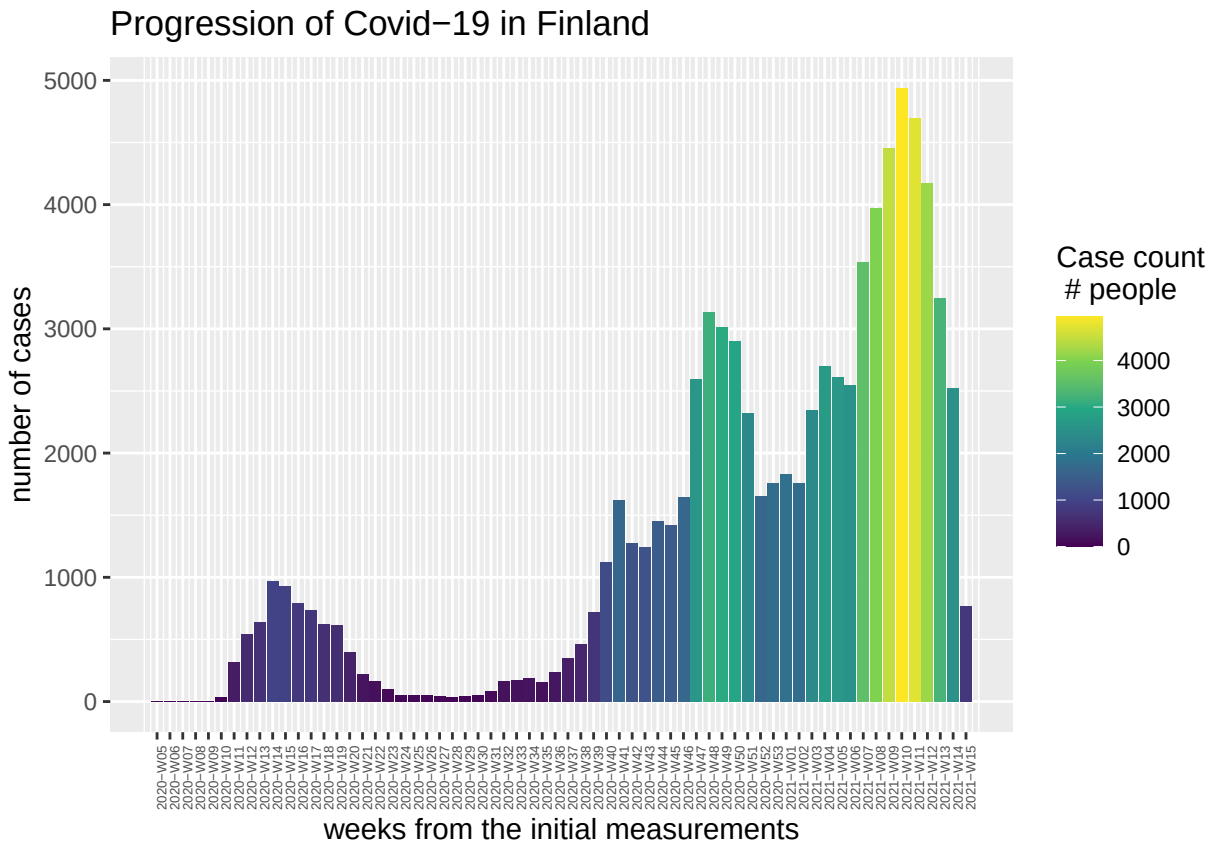
```
library(patchwork)
```

```
# Parameterize path to tsv and read the data
data.path <- "~/Documents/intro_to_systems_biology/Exercise Set 1-20210428/thl-covid.tsv"
thl.data <- read_tsv(data.path, locale = locale(encoding = "UTF-8"))
```

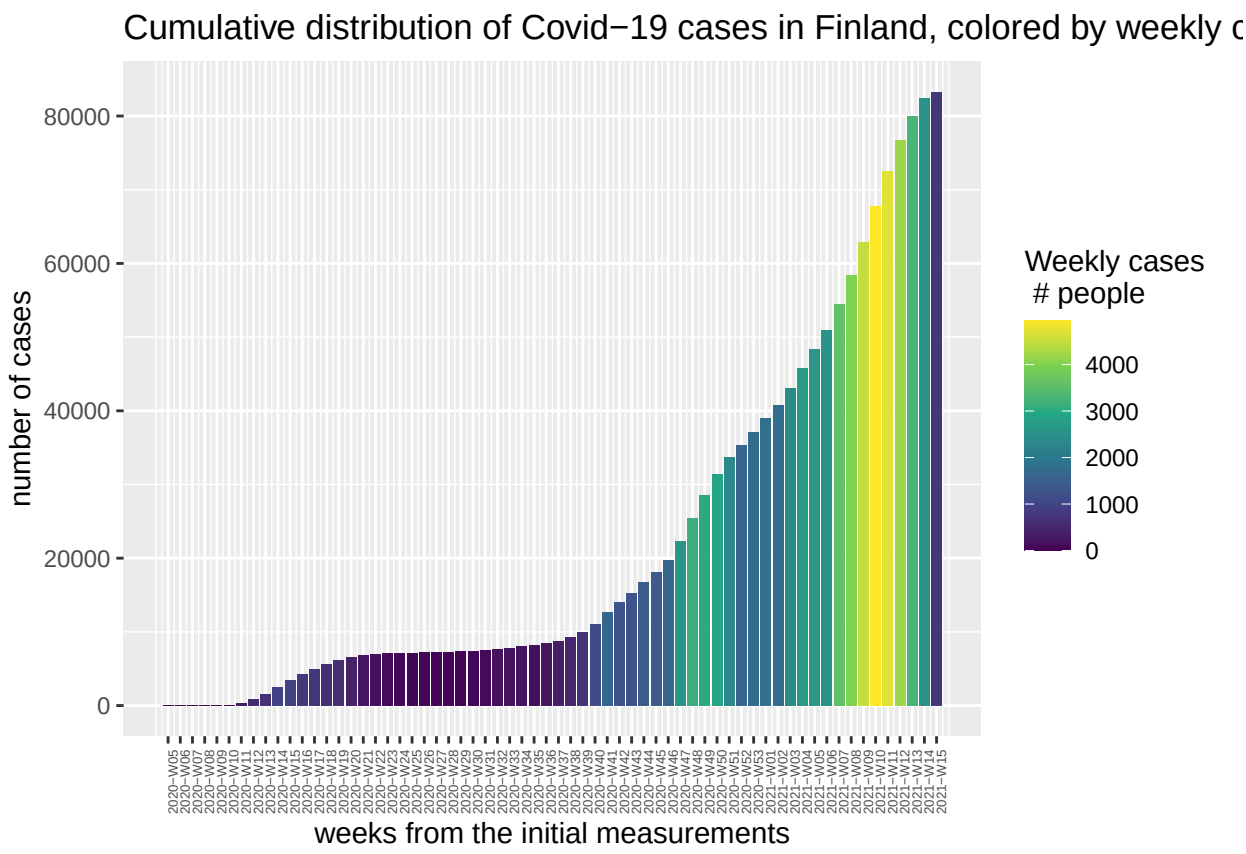
```
## Warning: Missing column names filled in: 'X1' [1]
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   .default = col_double(),
##   X1 = col_character()
## )
## i Use `spec()` for the full column specifications.
```

```
# Pick the number of days
total.weeks <- 1:nrow(thl.data)

# Compute row sums
weekly.cases <- rowSums(thl.data[,2:ncol(thl.data)])

# Plot the row sums
p1 <- tibble(total.weeks, weekly.cases) %>%
        ggplot(aes(x = total.weeks, y = weekly.cases, fill = weekly.cases)) +
        geom_bar(stat = "identity") +
        scale_fill_viridis_c() +
        scale_x_continuous(breaks = 1:length(thl.data$X1), labels = thl.data$X1) +
        theme(axis.text.x = element_text(angle = 90, size = 5)) +
        labs(fill = "Case count \n # people") +
        labs(title = "Progression of Covid-19 in Finland",
           x = "weeks from the initial measurements",
           y = "number of cases")

p1
```




```
# Cumulate cases
cum.cases <- cumsum(weekly.cases)

# Plot cumulative row sums
p2 <- tibble(total.weeks, cum.cases) %>%
        ggplot(aes(x = total.weeks, cumsum(weekly.cases), fill = weekly.cases)) +
        geom_bar(stat = "identity") +
        scale_fill_viridis_c() +
        scale_x_continuous(breaks = 1:length(thl.data$X1), labels = thl.data$X1) +
        theme(axis.text.x = element_text(angle = 90, size = 5)) +
        labs(fill = "Weekly cases \n # people") +
        labs(title = "Cumulative distribution of Covid-19 cases in Finland, colored by weekly count",
           x = "weeks from the initial measurements",
           y = "number of cases")

p2
```



Item b

```
# Functionalize the formula
infected <- function(x) {

  B = 1.598
  C = 0.836

  return(B * exp(C * x))
}
```

```
sprintf("At day 5, there would be %.3f cases", infected(5/7))
```

```
## [1] "At day 5, there would be 2.903 cases"
```

```
weeks.in.two.months <- 4 * 2

sprintf("At t = 4 weeks, there would be %.3f cases", infected(weeks.in.two.months))
```

```
## [1] "At t = 4 weeks, there would be 1282.739 cases"
```

```
# Compute the number of weeks

B = 1.598
C = 0.836

weeks.to.mayhem <- log(7.8e9 / B) / C
```

```r
sprintf("It would require %.3f weeks to spread through the whole Earth", weeks.to.mayhem)
```

```
## [1] "It would require 26.685 weeks to spread through the whole Earth"
```

Item c

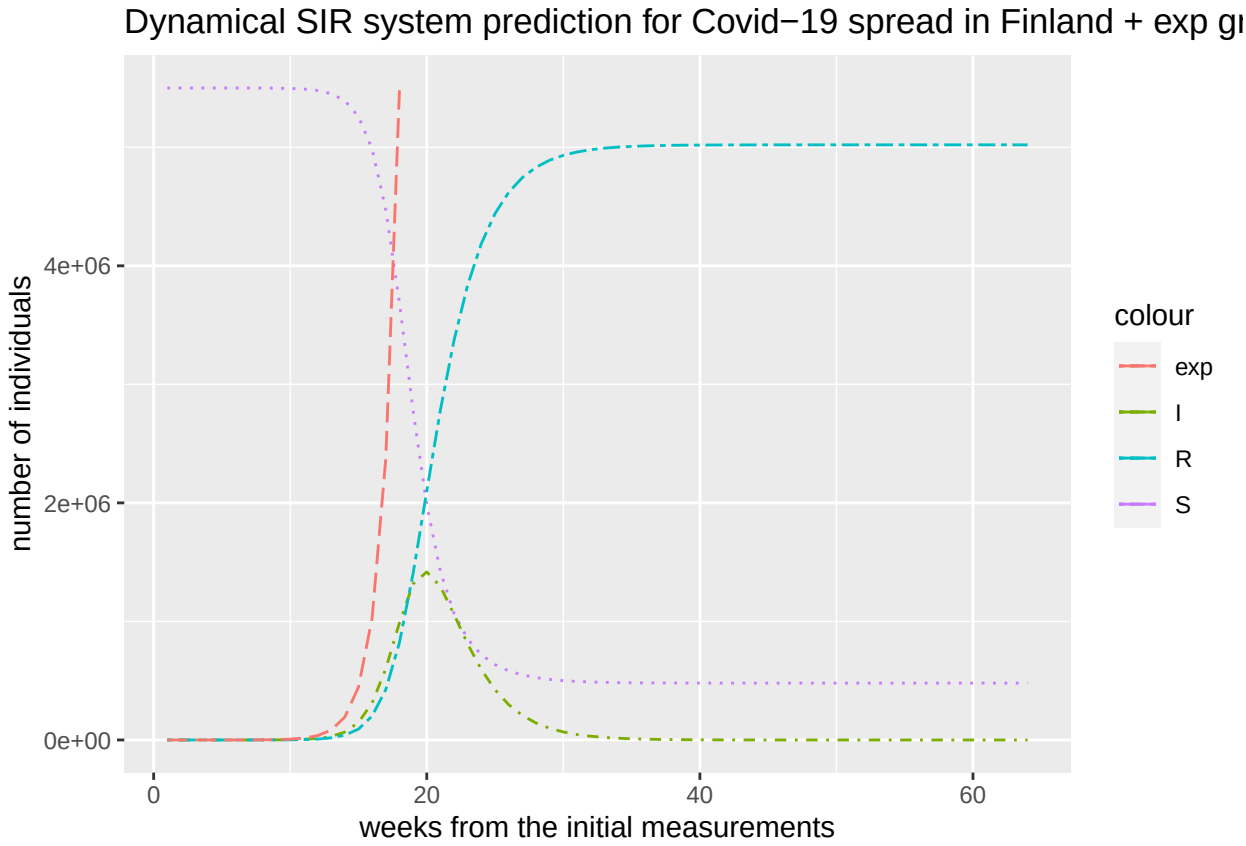
```r
# Aux function for Runge-Kutta method
source('~/Documents/intro_to_systems_biology/Exercise Set 1-20210428/rk4.R')

# Set params
population <- 5.5e6
beta <- 2.430e-7
delta_inv <- 2 # weeks

system.of.eqns <- function(t, X) {

  "
  ---------------------------------------------------------------------
  Modeling a dynamical system as follows

  s'(t) = -b*s(t)*i(t)
  i'(t) = b*s(t)*i(t) - d*i(t)
  r'(t) = d*i(t)

  where b, d = beta, delta in SIR model
  ---------------------------------------------------------------------

  params:
    X: a list containing:
      1. the number of suspectible individuals
      2. the number of initially infected people
      3. the number of initially recovered people

  returns:
    derivatives (w.r.t. t) of SIR equations at time point t

  "

  s <- X[1]
  i <- X[2]
  r <- X[3]

  ds <- -beta*s*i
  di <- beta*s*i -delta_inv^(-1)*i
  dr <- delta_inv^(-1)*i

  return(c(ds, di, dr))
}

# Time period for which to run the simulation (pick from THL data)
t_scale <- total.weeks

# Number of people infected at week 0
```

```r
initial.infected <- beta * population

# Vectorize the S, I, R model params
model.params <- c(population, initial.infected, 0) # initially 0 recovered

# Time scale + integration
rk_out <- rk4(system.of.eqns, t_scale, model.params)
```

```
## [1] "--------------------------"
## [1] "--------------------------"
```

```r
# Select the model predictions
pred <- data.frame(t(rk_out$X))

# Convert to tibble
pred <- tibble(pred) %>% rename("S" = X1, "I" = X2, "R" = X3)
pred$time <- t_scale
pred$expo <- sapply(t_scale, infected)

# Plot
ggplot(pred, aes(x = time)) +
  geom_line(aes(y = S, color = "S"), linetype = "dotted") +
  geom_line(aes(y = I, color = "I"), linetype = "dotdash") +
  geom_line(aes(y = R, color = "R"), linetype = "twodash") +
  geom_line(aes(y = expo, color = "exp"), linetype = "longdash") +
  labs(title = "Dynamical SIR system prediction for Covid-19 spread in Finland + exp growth",
           x = "weeks from the initial measurements",
           y = "number of individuals") +
  ylim(0, population)
```

```
## Warning: Removed 46 row(s) containing missing values (geom_path).
```



Item d

according to the SIR-model, the pandemic would peak at t_{max} s.t. $i(t_{max}) \approx 1.4 \cdot 10^6$. The number of people who would need treatment would of course be a function of time, but if we assume a transmission parameter β, at time t in simulatneous need of treatment would be $\beta s(t) i(t)$ [1]. In reality these numbered were not reached due to restrictions and relatively effective policy-making. The model is somewhat naive and does not take into consideration many Covid-19 specific cases, *e.g.*, the lockdowns and actual human behaviour.

As for R_0, assuming at $t = 0$ the whole population would be susceptible, we can approximate the value as follows:

$$R_0 = \frac{\beta\, s(t)}{\delta} = 2 \cdot 2.430 \cdot 10^{-7} \cdot 5.5 \cdot 10^6 = 2.673$$

Which seems relly high, but maybe not yielded via the best estimate on earth.

[1] Task 1 paper

Task 3 - Simulating Monty Hall problem

```
simulate.mhall <- function(num_iter) {

  # Self explanatory
  n_doors <- 3

  # Win count by changed strategy and initial strategy
  w.c <- 0
  w.nc <- 0

  for (i in 1:num_iter) {

    # Rng the prize door and guess good. Not mutex so replace = T
    prize <- sample(1:n_doors, 1, replace = T)
    guess <- sample(1:n_doors, 1, replace = T)

    # Opened door and a possible new guess
    host.opens <- setdiff(1:n_doors, c(prize, guess)) %>% sample(1)
    new.guess <- setdiff(1:n_doors, c(guess, host.opens)) %>%  sample(1)

    if (prize == guess) {
      w.nc <- w.nc + 1
    }

    if (prize == new.guess) {
      w.c <- w.c + 1
    }
  }

  c.freq <- w.c / (w.c + w.nc)
  nc.freq <- w.nc / (w.c + w.nc)

  sprintf("Total iter: %5.d  |  Wins by changed strategy: %.2f  |  Wins by initial strategy: %.2f",
          num_iter, c.freq, nc.freq)
}
```

Item b

```
# Simulate Monty Hall problem
iterations <- c(3, 5, 8, 1e2, 1e4)

# For nice results:
mhall.simus <- sapply(iterations, simulate.mhall)
mhall.simus
```

```
## [1] "Total iter:     3  |  Wins by changed strategy: 0.67  |  Wins by initial strategy: 0.33"
## [2] "Total iter:     5  |  Wins by changed strategy: 0.00  |  Wins by initial strategy: 1.00"
## [3] "Total iter:     8  |  Wins by changed strategy: 0.60  |  Wins by initial strategy: 0.40"
## [4] "Total iter:   100  |  Wins by changed strategy: 0.49  |  Wins by initial strategy: 0.51"
## [5] "Total iter: 10000  |  Wins by changed strategy: 0.56  |  Wins by initial strategy: 0.44"
```

We should begin to see convergence towards the correct probabilities fairly quickly, but this one did not do it. Usually when simulating the Monty Hall problem (if I remember correctly, it's been a few moments since I did the last time...), a few thousand iterations should be enough for a convergence of $\varepsilon \leq 1\%$. Well, maybe

it is the random take we had now at $n = 10000$ that just does not tell the story. The probabilities of winning the Monty Hall problem are $P(S) = \frac{2}{3}$ and $P(N) = \frac{2}{3}$ where $S = \textit{''switch''}$ and $N = \textit{''don't switch''}$. The statement can be justified with Bayesian modeling as follows:

For any door $d \in \{1, 2, 3\}$, the probability of it being the winning door is $P(\text{win} = d) = \frac{1}{3}$.

Two cases follow. Consider the winning door d_w. This car would never be opened, so we can focus on two remaining doors. The likelihoods of the remaining doors, say, d_a and d_b being opened are therefore

$$P(\text{open} = d_a \mid \text{door} = d_b) = \frac{1}{2}$$

$$P(\text{open} = d_b \mid \text{door} = d_w) = 1$$

It immediately follows that

$$P(\text{door} = d_a \mid \text{open} = d_b) = \frac{\frac{1}{2} \cdot \frac{1}{3}}{P(d_b) = \frac{1}{2}} = \frac{1}{3}$$

And

$$P(\text{door} = d_w \mid \text{open} = d_b) = \frac{1 \cdot \frac{1}{3}}{P(d_b) = \frac{1}{2}} = \frac{2}{3}$$

Where $P(d_b)$ is simply the sum $\frac{1}{2} \cdot \frac{1}{3} + \cdot \frac{1}{3}$ from the events following from

$$P(d_b) = P(\text{door} = d_a) \cdot P(\text{open} = d_b \mid \text{door} = d_a) + P(\text{door} = d_w) \cdot P(\text{open} = d_b \mid \text{door} = d_w)$$