Disease Modeling - FYS3150 Computational Physics

Nicholas Karlsen

This is an abstract

INTRODUCTION

THEORY, ALGORITHMS AND METHODS

SIRS Model

The SIRS model is a mathematical model from epidemiology describing how infectious disease evolves within a population, and is a part of a family of similar models in epidemiology with various different features. In the SIRS model, the total population (N) is divided into three groups

- Susceptible (S) : People who do not have the disease, and are not immune to it.
- Infected (I) : People who are infected with the disease.
- Recovered (R) : People who has recovered from the infection, and have developed immunity.

Where in the simplest case, the permitted traversal from one group to another follows a cyclical pattern $S \to I \to R \to S$, hence the name SIRS.

The rate of traversal is governed by a set of coupled differential equations,

$$\begin{aligned} S' &= cR - \frac{aSI}{N} \\ I' &= \frac{aSI}{N} - bI \\ R' &= bI - cR \end{aligned} \tag{1}$$

Where the constants a, b, c are governing the

- rate of transmission
- rate of recovery
- rate of immunity loss

respectively, with dimension inverse time. For the purposes of this report, we will not consider any particular unit of time, but rather the dynamics of the system, as the timescales at which different diseases operate on vary. However, based on data (INSERT CITATIONS) a lot of common diseases are observed to operate on a scale of days, whilst some operate on a scale of years.

We note here that each term in the system of equations is reflected in opposite, for a different group. For example, the cR term in S' is mirrored in opposite in R' as $-cR$. As defined earlier, c denotes the rate of immunity loss, so these terms essentially moves people in the R group to the S group. The result of this is that that the total population is conserved in this system, such that we have a constant $N = S(t) + I(t) + R(t)$.

As shown in Hjorth-Jensen [1], we can then substitute this into the set of equations 1 and derive equations for how the system distributes itself in a steady state

$$\begin{aligned} s^* &= \frac{b}{a} \\ i^* &= \frac{1 - \frac{b}{a}}{1 + bc} \\ r^* &= \frac{b}{c}\frac{1 - \frac{b}{a}}{1 + \frac{b}{c}} \end{aligned} \tag{2}$$

which notably only depends upon the parameters governing the system. Meaning that for some set of parameters, a, b, c the system will always converge towards the same state given enough time. Matching the behaviour of what we expect from a Markov chain governed by a regular stochastic matrix as shown in for example, Lay [2, p. 277, theorem 18].

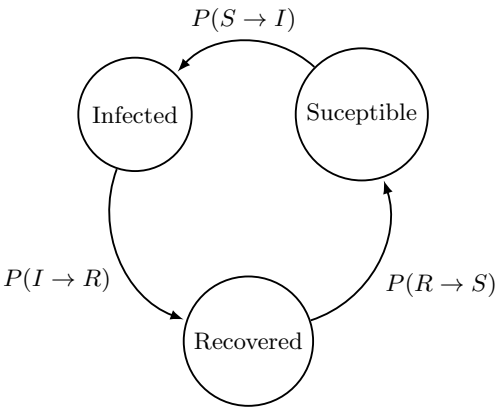

FIG. 1: Markov diagram for the basic SIRS model

Extending the SIRS model

The model described by Eqn.1, henceforth referred to as the Basic SIRS model can be extended to model other behaviours relevant to the study of an illness. In this project, we will specifically look at 3 extensions to the

model, that is: Vital dynamics, Seasonal variation and vaccinations.

Starting off, we add the effect of vital dynamics to our set of differential equations in the following way

$$\begin{aligned} S' &= cR - \frac{aSI}{N} - dS \\ I' &= \frac{aSI}{N} - bI - (d + d_I)I \\ R' &= bI - cR - dR \end{aligned} \tag{3}$$

where d denotes the death rate of the population, and d_I the added death rate of the Infected group of the population. We see here that the new terms are not mirrored in the same way as that constitute the basic model such that the system is generally speaking no longer population conservative if any of these terms are non-zero. With a possible exception being for example if $d = 0,\ a = 0,\ d_I > 0$ and $I_0 = 0$, in which case the population would be conserved.

Seasonal variation, or simply a periodic behaviour in the rate of transmission, added to the model by no longer having a constant a, but instead letting it be a function of time as

$$a(t) = A\cos(\omega t) + a_0 \tag{4}$$

Where a_0 denotes the baseline rate of transmission, A the maximum deviation from the baseline and ω the frequency of the variation.

and lastly, we look at a possible way modelling the effects of vaccinations by this time adding a term f, denoting the rate of vaccination to our basic model in the following way

$$\begin{aligned} S' &= cR - \frac{aSI}{N} - f \\ I' &= \frac{aSI}{N} - bI \\ R' &= bI - cR + f \end{aligned} \tag{5}$$

We see here that this addition of this terms allows someone in the S group to go immediately to the I group, which as intended. Further, this transition rate from $S \to I$ is not dependent on the population, or size of any population groups unlike any of the other behaviours modelled thus far. Whilst we will save the discussion on the many implications this has later, we note here a particular, significant flaw in this extension to the model. Whilst it is still population conservative if we look only at $N = S(t) + I(t) + R(t)$, it does present the possibility of $S(t) < 0$, which is entirely un-realistic.

Monte-Carlo Algorithm

We will now build a Monte-Carlo algorithm for simulating the dynamics of disease modelled by the different sets of equations we have looked at. These simulations aims to, on average match the behaviour predicted by the differential equations, but in a more realistic way by first and foremost allowing only discrete, transitions from from one group to another instead of continuous ones. And further, imposing certain restrictions to prevent unintended behaviours such as any of the groups becoming negative, which as mentioned earlier is a possibility in the system modelling the effects of vaccinations, in Eqn. 5.

In Hjorth-Jensen [1] we are methodology of translating the sets of differential equations to a set of transition probabilities for the basic model

$$\begin{aligned} P(S \to I) &= \frac{aSI}{N}\Delta t \\ P(I \to R) &= bI\Delta t \\ P(R \to S) &= cR\Delta t \end{aligned} \tag{6}$$

and a step-size

$$\Delta t = \min\left\{\frac{4}{aN}, \frac{1}{bN}, \frac{1}{cN}\right\} \tag{7}$$

where for each step we evaluate the transition probabilities against a random number $r \in [0, 1]$. If the transition probability is greater than r, the transition is made, else, it isn't. This constitutes the Monte-Carlo part of the algorithm, and in order to produce expectation values we take the average of many cycles of systems initiated in the same way, and evolved following these rules.

Shown below in Pseudo-code

SIRS Monte-Carlo

```
for each MC-Cycle:
  for each t_i:
    Compute P(S → I)
    Compute P(I → R)
    Compute P(R → S)

    if P(S → I) > r and S_i > 0:
      S_{i+1} -= 1
      I_{i+1} += 1

    if P(I → R) > r and I_i > 0:
      I_{i+1} -= 1
      R_{i+1} += 1

    if P(R → S) > r and R_i > 0:
      R_{i+1} -= 1
      S_{i+1} += 1
```

Extending this model is done quite easily by following the same model logic outlined in Hjorth-Jensen [1]. If we consider the system the addition of vital dynamics, we have three additional permitted transitions, one for each group. $S \to DEAD$, $I \to DEAD$ and $R \to DEAD$. Where probabilities of these transitions per time-step is

given by

$$\begin{aligned} P(S \to DEAD) &= dS\Delta t \\ P(I \to DEAD) &= (d + d_I)I\Delta t \\ P(R \to DEAD) &= dR\Delta t \end{aligned} \tag{8}$$

where the time step is now instead given by

$$\Delta t = \min\left\{ \frac{4}{aN}, \frac{1}{bN}, \frac{1}{cN}, \frac{1}{dN}, \frac{1}{(d+d_I)N} \right\} \tag{9}$$

Where the last entry of the set, $\frac{1}{(d+d_I)N}$ can be omitted for a slight performance increase as it will always be greater than $\frac{1}{dN}$.

Another important issue, that is dealt with in the IF statements is the potential of negative population groups. Whilst the overall net population may be conserved mathematically without these checks, we have to ensure that no groups goes below 0. This is dealt with quite simply by always checking that the group which would be reduced is populated by at least one person.

we see that the step sizes is a function of the population size, which means that separate cycles will traverse the simulation period in different ways. Therefore, for this model we resort to a linear interpolation between data points in each cycle to produce averages. For comparisons sake, it then makes the most sense to draw these interpolations at points matching the solution produced by RK4.

4th-order Runge-Kutta method

The 4th-order Runge-Kutta (RK4) is one part, of a family of methods for solving ordinary differential equations (ODE). In short, the Runge-Kutta methods consists of taking a weighted average of different finite-steps, in particular, for RK4 we have the steps

$$\begin{aligned} k_1 &= f(t_i, y_i) \\ k_2 &= f\left(t_i + \frac{h}{2}, y_i + \frac{h}{2} \cdot k_1\right) \\ k_3 &= f\left(t_i + \frac{h}{2}, y_i + \frac{h}{2} \cdot k_2\right) \\ k_4 &= f(t_i + h, y_i + h \cdot k_3) \end{aligned} \tag{10}$$

where h denotes the step-size. A weighted average of these steps are then used to compute the $i+1$'th element in the following way.

$$y_{i+1} = y_i + \frac{1}{6}h(k_1 + 2k_2 + 2k_3 + k_4) + \mathcal{O}(h^5) \tag{11}$$

where $\mathcal{O}(h^5)$ is the error. For more details, refer to Hjorth-Jensen [3, p. 250]

RESULTS AND DISCUSSIONS

Implementation

Before we look at the results, i would like to make a quick remark in regards to the code, and its structure. My initial plan was to implement the entirety of the project in python, however, upon starting to consider how i might implement the vital dynamics, with its dynamic step sizes i decided to implement this, and the subsequent models in Julia[1], which generally speaking is much faster than pure python[2]. A particular benefit of Julia being that there exists APIs that allows calling Julia functions directly from python and in reverse. These APIs allowed me to very easily offload the heavier Monte-Carlo simulations to Julia, then performing the subsequent plotting and analysis in python.

SIRS: Simple model

We start by looking at the results produced by both the RK4 solution to Eqn. 1 as well as the Monte-Carlo results for this system, for 100 cycles. The results are presented in a quite compact form in Fig. 2, where all 4 sub-plots share a common legend. In this, and subsequent plots the colours blue, red and green indicate the S, I and R groups respectively and individual MC cycles are plotted with a transparency of 5%, such that common paths will display a stronger colour. Overlaid on top of these individual MC cycles are the ODE result, as well as the Mean Monte-Carlo result. For the sake of readability the Mean and ODE results have not been colour coded with individual colours for each group, but their belonging should be quite clear from the individual cycles on which they overlay.

In these plots, the system was initiated in in the state $S_0 = 300, I_0 = 100$ and $R_0 = 0$ and the parameters $a = 4, c = 0.5$ are held constant throughout all 4 sets of data. We see that for the MC simulations, only 3 out of the 4 cases converges towards the predicted set of steady states, indicated by axis ticks on the right x-axes. In the case of $b = 3$, we observe that neither the S or R groups converge towards their respective steady states. To investigate this further, we then refer to Fig. 3, where we note how the standard deviation of the MC results seem to rise towards a peak from the beginning, until $t \approx 10$, before it drops. To make sense as to why there is such a great uncertainty in this area, we refer to $b = 1$ and $b = 2$. We observe that the standard deviation in all 3 groups are quite consistent for $b = 1$, then looking at

[1] This also meant that i had to learn Julia

[2] Python without the support of efficient APIs, i.e numpy

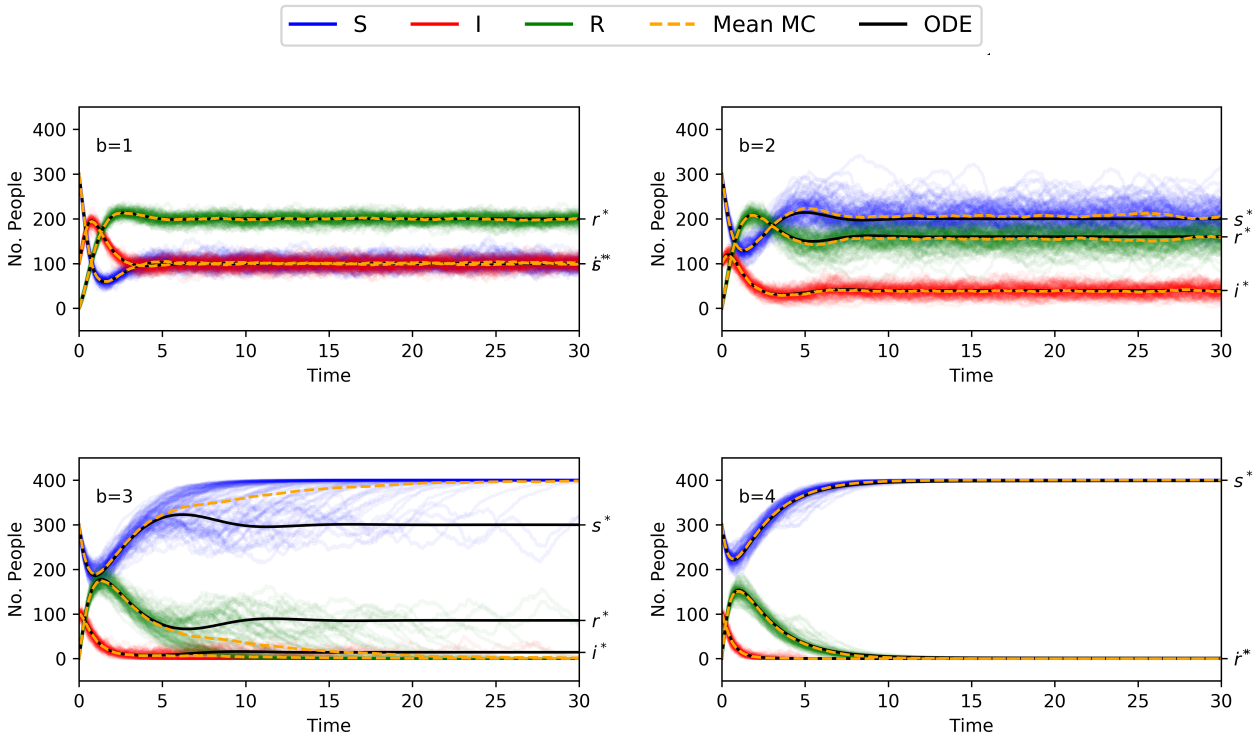

FIG. 2: The basic SIRS model simulated by the Monte-Carlo simulation, as well as the ODE solutions with initial state $S_0 = 300, I_0 = 100, R_0 = 0$, constants $a = 4, c = 0.5$ and $b = 1, 2, 3, 4$.

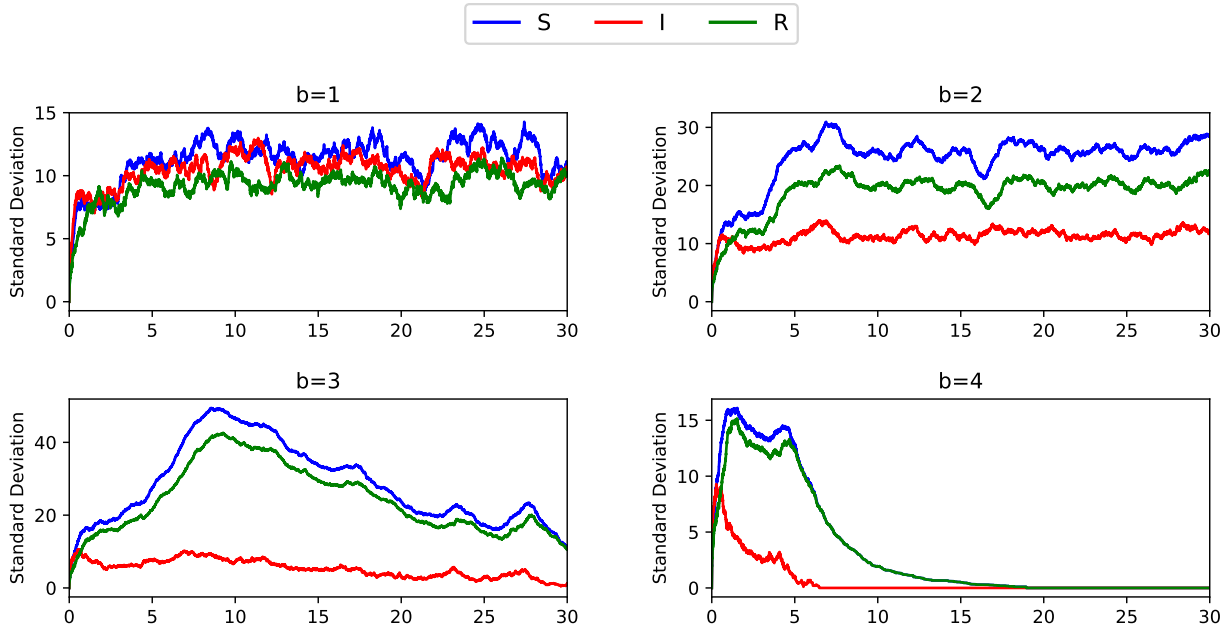

FIG. 3: The standard deviations for each b in the Monte Carlo solutions shown in Fig. 2, consisting of 100 trials each.

the standard deviation for $b = 2$ we see that standard deviation for the I group is similar to $b = 1$, but the R, and particularly S group has a significantly higher standard deviation. We observe here, what I believe to be the main cause of uncertainty, and deviance from the ODE solution for the Monte-Algorithm. That is, whenever one (or more) of the groups tends towards either the upper or lower population limits, the uncertainty of the others

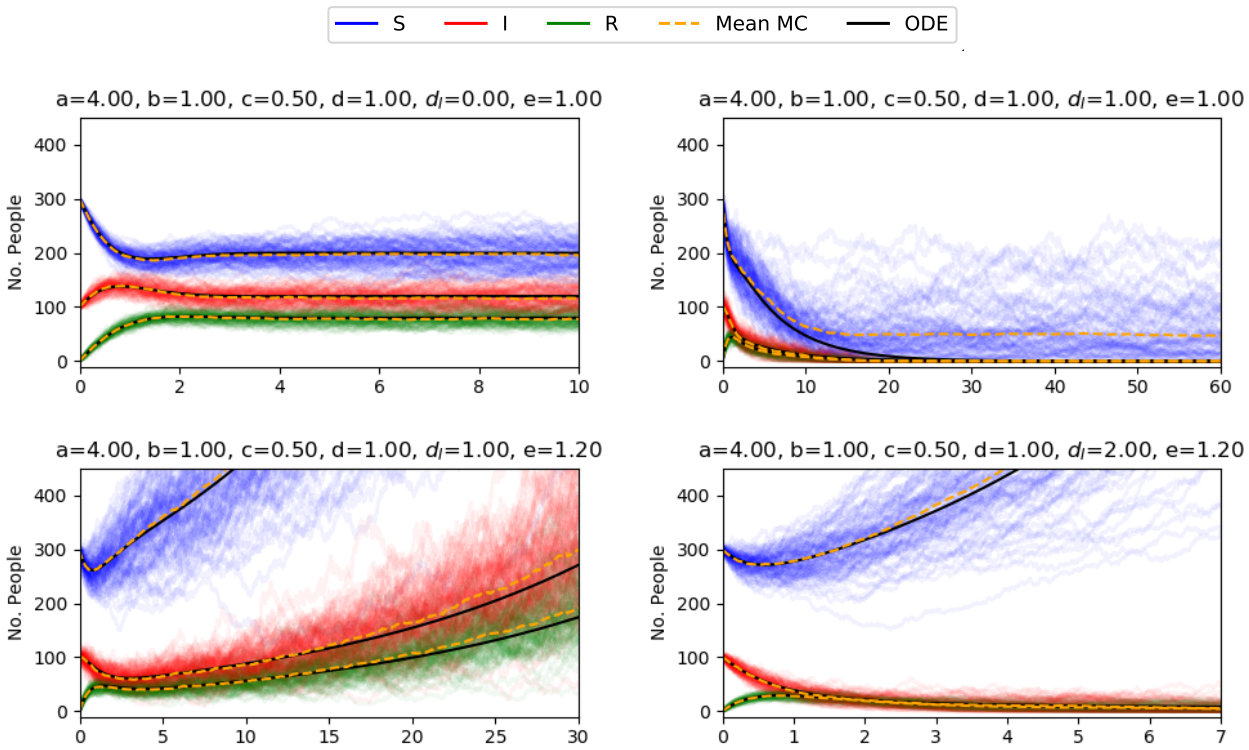

FIG. 4: The SIRS model with vital dynamics for initial state $S_0 = 300, I_0 = 100, R_0 = 0$ and constants $a = 4, b = 1, c = 0.5$.

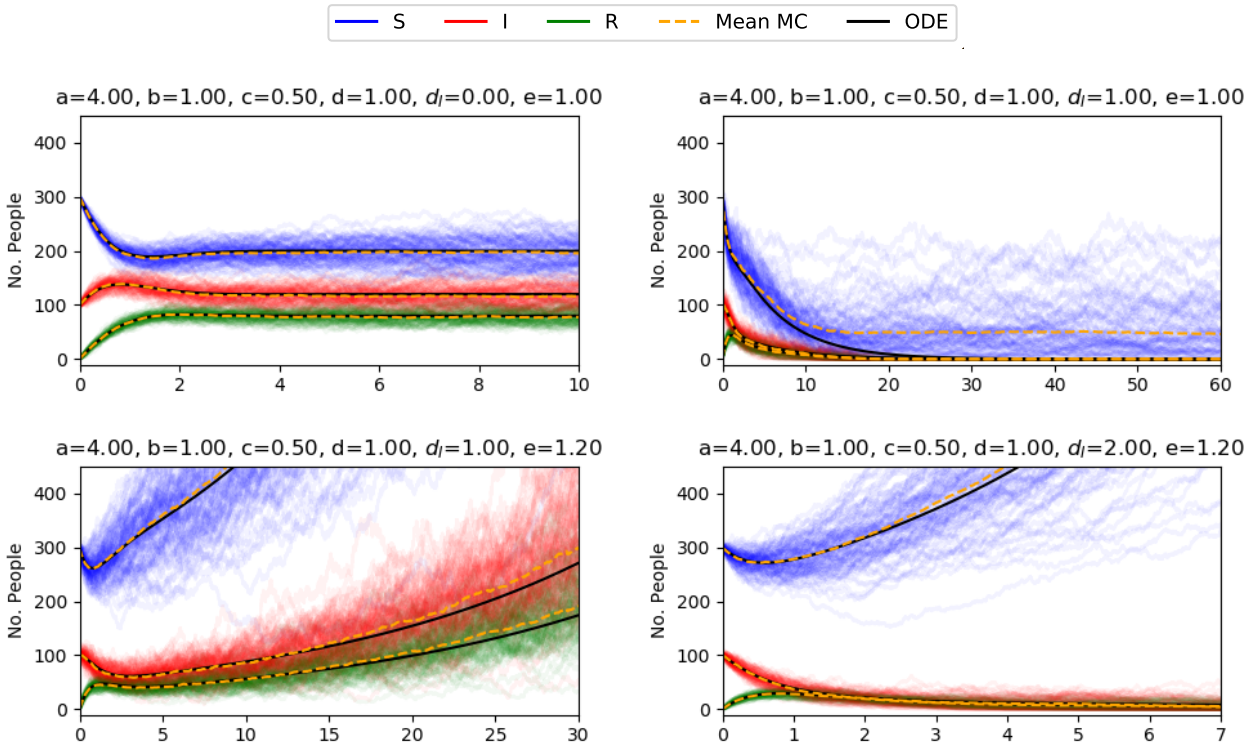

FIG. 5: The SIRS model with vital dynamics for initial state $S_0 = 300, I_0 = 100, R_0 = 0$ and constants $a = 4, b = 1, c = 0.5$.

seem to rise locally, and this error is then propagated forwards in time, even though the standard deviation in the

MC simulations goes down again. We will investigate this hypothesis more thoroughly later in the model including seasonal variation, but for now we note that the uncertainty, and deviation from the ODE in the MC solutions isn't necessarily tied to the number of MC cycles, but the way in which the system evolves, and that we need to take into account the paths of each set of MC simulations when assessing any errors on a per-simulation basis rather than a per-system or per-number of iterations basis as is often the case in numerical methods[3]

SIRS: Vital Dynamics

If we not turn to Fig. REF we see the plots for the vital dynamics system. For this model, we observe a significant mismatch between the ODE solution and and the MC solutions. Whilst this initially hits to

CONCLUSIONS

We note also, a problem in the algorithm. That is; in what order do we evaluate the probabilities? Whether to consider $P(S \to I)$ or $\mathrm{P}(R \to S)$ first is completely arbitrary, but may lead to a certain bias in smaller populations or when a group is nearing the lower and upper limits of the group sizes, which would potentially explain the local peaks of the standard deviation that we observed in these areas. A possible improvement to the algorithm can then be that we randomly select the order in which the transition probabilities are assessed, which i regretfully do not have the time to experiment right now.

It would also be interesting to experiment with the behaviour of a compound model involving all of the different extensions that we have looked at, described by

$$\begin{aligned} S' &= cR - (A\cos(\omega t) + a_0)\frac{SI}{N} - dS + eN - f \\ I' &= (A\cos(\omega t) + a_0)\frac{SI}{N} - bI - (d + d_I)I \\ R' &= bI - cR - dR + f \end{aligned} \tag{12}$$

Whilst the implementation of this model trivial[4], the analysis of it is likely a much more involved, and out of the scope of this project.

[1] M. Hjorth-Jensen, "Disease modeling, fys3150 project 5." (2018).

[2] D. C. Lay, *Linear Algebra and Its Applications*, 5th ed. (Pearson, 2016).

[3] M. Hjorth-Jensen, *Computational Physics - Lecture notes* (2015).

[3] At least in my own, limited experience.

[4] Armed with the experience of implementing, and understanding the individual features