Preprint typeset using LaTeX style emulateapj v. 12/16/11

DISEASE MODELING

Joakim Flatby, Linus Ekstrm
https://github.com/jflatby/fys3150

ABSTRACT

The spread of disease in a population has been studied using the SIRS differential equations. We simulate the behaviour of the system using fourth order Runge-Kutta and with discrete Monte Carlo algorithm.

We studied the system with recovery rates $b \in [1, 2, 3, 4]$. The simulations show that the critical point for the survival of the disease is when the recovery rate is equal to the transmission rate. Subsequently, we added vital dynamics to our simulation, as well as seasonal variation of infection rate. Here we introduced oscillations into the transmission rate, resulting in a system which did not reach a steady state over our simulation time. This models real life diseases more accurately than with just a static transmission rate. And finally, we studied the effects of a vaccination campaign in the population.

The result was a final model which enlightened us on all the different variables connected to an epidemic in a population and their importance.

1. INTRODUCTION

How a disease spreads from one person to another is intricate science and hard to predict. However, on a larger scale we can try to model how the disease would spread throughout a population of N people. In such a model we don't need to know the details on how the disease spreads, and can generalize the behaviour of the disease into three basic numbers, the rate of transmission, the rate of recovery and the rate of immunity loss. Using these three variables we construct a set of differential equations(SIRS model), and simulate it using python. This method, however, assumes that the number of people is a continuous variable which is of course not the case in real life. We therefore move on to a Monte Carlo simulation by only looking at probabilities of transitioning between the different groups in the model, and making sure we only move one person at a time. Throughout this paper we will be adding more variables, taking more and more into account to make the model as realistic as possible.

The work done in this study is compiled as follows, first we present the necessary theory for our Python implementation. Next, we describe using words how we implemented the various alterations to our SIRS model in the Method section. Following this, we present our results, mainly consisting of graphs of our differential equation system and stochastic models. These results are also discussed in the results section. And finally, a brief conclusion summarizing our work and connecting it to real world applications.

2. THEORY

2.1. *Differential equations*

The SIRS model is a popular way of simulating the development of a disease in a given isolated population of N people. The population is divided into three groups, Susceptible(S), Infected(I) and Recovered(R). S are those who can get the disease, they don't have it and don't have an immunity. I are the infected individuals, and R are the ones that have recovered and currently have an immunity to the disease. As time progresses, people move between these three groups. We assume that we are working in a time frame short enough so that we can ignore birth and death rates, and that

$$N = S(t) + I(t) + R(t) \tag{1}$$

so that the total number of people N is constant. With these assumptions, the change in the three groups over time can be written as the following differential equations

$$\begin{aligned} S' &= cR - \frac{aSI}{N} \\ I' &= \frac{aSI}{N} - bI \\ R' &= bI - cR \end{aligned} \tag{2}$$

where a is the rate of transmission, b is the of recovery and c is the rate of immunity loss. S, I and R are the amount of people in their respective groups. This is the classical SIRS model, and what we are basing our further improvements of the model on. We don't have analytic solutions for this set, but if we solve Eq. (1) for R and substitute this into the equation for S' in Eq. (2), we see that the R' is no longer required. At equilibrium there should be no change in the groups, so we set the two remaining equations equal to zero

$$\begin{aligned} 0 &= c(N - S - I) - \frac{aSI}{N} \\ 0 &= \frac{aSI}{N} - bI \end{aligned} \tag{3}$$

Which gives the following distribution between S, I and R at equilibrium

$$\begin{aligned} s^* &= \frac{b}{a} \\ i^* &= \frac{1 - \frac{b}{a}}{1 + \frac{b}{c}} \\ r^* &= \frac{b}{c}\frac{1 - \frac{b}{a}}{1 + \frac{b}{c}} \end{aligned} \tag{4}$$

We can use these to check our results by seeing how the population is distributed at the end of a simulation. If we are confident the simulation has reached an equilibrium, and there are 600 people out of 1000 who are Susceptible, we should also get $s^* = 0.6$.

2.2. *Fourth Order Runge-Kutta*

In order to solve the SIRS differential equations described above, as well as our subsequent modified models, we opted for the fourth order Runge-Kutta method. This is by far the most common method of the Runge-Kutta family, so much so that it is generally referred to as *the* Runge-Kutta method. The method is structured as follows:

let an initial value problem be specified as

$$\frac{dy}{dt} = f(t, y), \quad y(t_0) = y_0 \tag{5}$$

where y is an unknown vector or scalar function of time, which we are approximating. $\frac{dy}{dt}$ is the rate which y changes with respect to the simulation time, which is a function of t and y. At the initial time t_0 the corresponding y-value is y_0. We use this value to push our solution along to the next time step. To do this we define a step size, according to our desired accuracy, $h > 0$ and define

$$\begin{aligned} y_{n+1} &= y_n + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4) \\ t_{n+1} &= t_n + h \end{aligned} \tag{6}$$

for $n = 0, 1, 2, \ldots$ where the K_1 through K_4 are terms describing four different sample increments for our approximation.

$$\begin{aligned} K_1 &= hf(t_n, y_n) \\ K_2 &= hf\big(t_n + \frac{h}{2}, y_n + \frac{K_1}{2}\big) \\ K_3 &= hf\big(t_n + \frac{h}{2}, y_n + \frac{K_2}{2}\big) \\ K_4 &= hf(t_n + h, y_n + K_3) \end{aligned} \tag{7}$$

- K_1 is the increment based on the slope at the beginning of the sample interval, using y. This is what is done in Euler's method;
- K_2 is the increment based on the slope at the midpoint of the interval, using y and K_1;
- K_3 is again the increment based on the slope at the midpoint, but now using y and K_2;
- K_4 is the increment based on the slope at the end of the sample interval, using y and K_3

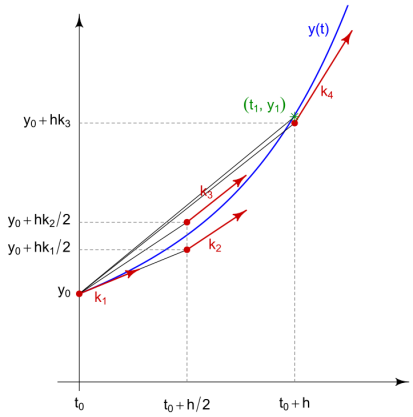

FIG. 1.— Graphic describing the sample points in the fourth order Runge-Kutta method. Picture generated for the Runge-Kutta article on wikipedia by user: HilberTraum (1)

when averaging the four increments more weight is given to the midpoint increments. This is to capture slope details which otherwise would be missed in I.E Euler's method. The RK4 method is a fourth-order method, meaning that the total accumulated error is on the order $O(h^4)$.

2.3. *Monte Carlo simulation*

The model we have so far does a good job of simulating the development of a disease, but it's very unrealistic in the sense that the variables S, I and R are used as continuous variables. In the real world 0.3 people cant get a disease, so we need to change our model to a system where 1 person is either moved from one group to the next or not, no fractions. To do this we need to ensure that our simulation uses sufficiently small Δt so that at most one person makes the move from one group to the next. Looking at Eq. (2) We see that the amount of people that move from S to I is approximately $\frac{aSI}{N}\Delta t$. We can easily see that the maximum value would occur if half the population was in S, and the other half in I, so we substitute in $\frac{N}{2}$ for both and get

$$\max\Big\{\frac{aSI}{N}\Delta t\Big\} = \frac{aN}{4}\Delta t \tag{8}$$

We apply the same technique to the transitions from I to R and from R to S and get

$$\max\Big\{bI\Delta t\Big\} = bN\Delta t \tag{9}$$

and

$$\max\Big\{cR\Delta t\Big\} = cN\Delta t \tag{10}$$

So the time step we will use in our simulation will be calculated as

$$\Delta t = \min\Big\{\frac{4}{aN}, \frac{1}{bN}, \frac{1}{cN}\Big\} \tag{11}$$

With the simulation set up this way, we can regard $\frac{aSI}{N}\Delta t$, $bI\Delta t$, and $cR\Delta t$ as the probabilities of each respective transition to occur

$$
\begin{aligned}
P(S \to I) &= \frac{aSI}{N}\Delta t, \\
P(I \to R) &= bI\Delta t, \\
P(R \to S) &= cR\Delta t.
\end{aligned} \tag{12}
$$

2.4. *Vital Dynamics*

Expanding our model can be done by adding terms for vital dynamics, allowing the spread of disease to be modelled over longer stretches of time. If b is the birth rate, d is the death rate, and d_I is the death rate of the infected, then our model reads the following

$$
\begin{aligned}
S' &= cR - \frac{aSI}{N} - dS + eN \\
I' &= \frac{aSI}{N} - bI - dI - d_I I \\
R' &= bI - cR - dR
\end{aligned} \tag{13}
$$

Note that this model assumes that all babies are born susceptible to the disease which is modelled. These equations are then implemented in our code to simulate the disease behaviour. More on this in the method section.

2.5. *Seasonal Diseases*

If we wish to model diseases such as influenza[1], where the rate of transmission depends heavily on the time of year. This seasonal dependence is not well understood and is hypothesized to be caused by a range of different factors. Indoor crowding, fluctuations in host immune response, relative humidity, temperature and UV-radiation are some of the candidates. Further reading: (2). Nevertheless, this behaviour can be modelled as a simple oscillation function

$$
a(t) = A\cos(\omega t) + a_0 \tag{14}
$$

where a_0 is the average transmission rate, A is the maximum deviation from a_0, and ω is the frequency of the transmission oscillation.

2.6. *Vaccination*

In the current year the topic of vaccination is somehow a subject which is 'debated'. Even though published scientific evidence for their effectiveness date all the way back to Edward Jenner in 1796, further reading: (3). Further, the tradition of vaccination is thought to have originated in India around AD 1000. Adding a vaccination term to our model breaks the cyclic nature by allowing susceptible individuals to move directly into the resistant category. Thus, the system of equations reads

$$
\begin{aligned}
S' &= cR - \frac{aSI}{N} - f \\
I' &= \frac{aSI}{N} - bI \\
R' &= bI - cR + f
\end{aligned} \tag{15}
$$

This version of the SIRS model assumes that a susceptible individual's choice to become vaccinated does not depend on how many other susceptible individuals are vaccinated. Note that this version lets us model the disease spread whether the vaccination rate is constant or some form of oscillation. To bring it back to influenza, which in temperate climates oscillates seasonally. It is reasonable to assume that the influenza vaccination rate oscillates in a similar manner, although slightly off-set because one commonly wishes to have worked up a resistance before the influenza season sets in.

[1] Which we as Norwegians have a much to familiar relationship with this time of year.

3. METHOD

3.1. *Differential equations*

To solve our differential equations we create a python script using the fourth order Runge-Kutta method(Section 2.2). We set the total population $N = 400$ people, 300 who are initially susceptible and the rest are infected from the start. We set a fixed rate of transmission $a = 4$ and a fixed rate of immunity loss $c = 0.5$, since these are variables that the human population can do little about. We try 4 different values for the rate of recovery, $b = 1, 2, 3, 4$, and see how the simulation differs. We use these initial conditions to loop over the 4 different populations. The differential equations for each population is defined and sent to the Runge-Kutta method to solve over time. The Runge-Kutta function returns a multidimensional matrix u containing the approximated values for S, I and R in each timestep. We plot these vs. time in days and compare the plots of the 4 different populations. Results are shown in section 4.1.

3.2. *Monte Carlo simulation*

To avoid continuous variables and discretize our calculations we put the Runge-Kutta method aside and consider the transition probabilities derived in section 2.3. We create a function which takes all the components of our equations in (12), N, a, b, c, S0, I0 and R0. We set Δt according to equation 11, and total time $T = 30$days. The function loops over the total amount of timesteps, creates a random number between 0 and 1, and if the number is lower than the transition probability, one person is moved according to the corresponding transition ($S \to I, I \to R, R \to S$). This is done for all three possible transitions in each timestep. After the loop is done, a multidimensional array u is returned in the same way as the Runge-Kutta method did, and the results are plotted in the same way.

3.3. *Vital Dynamics*

To implement vital dynamics in the differential equation solver we added the extra terms for the birth rate, death rate and infected death rate described in (2.6) to our SIR-vector function. The total population is no longer constant, $N = S(t) + I(t) + R(t)$, and is updated every time step.

For our Monte Carlo method there are now a couple if-statements in order to capture the behaviour caused by birth and death rates. These rates respectively add or remove individuals from their corresponding categories. This means that for each iteration we had to update the optimal time step, because the time step is dependent on the total population.

3.4. *Seasonal Variation*

As discussed in section (2.6) the infection probability of diseases can often vary seasonally. This is modelled as a cosine function. In order to implement this behaviour we simply added the cosine function into our simulation loops. For a realistic simulation of Influenza's seasonal variation one would have a peak in the transmission rate once a year, during the early winter. However, we felt our implementation was better highlighted with multiple peaks per year. We decided to simulate a fictional disease which had higher transmission rates twice a year with intervals of lower transmission in between.

3.5. *Vaccination*

Adding the vaccination to our code was as simple as defining a variable f, and adding it to our differential equations in the Runge-Kutta code, according to equation (15). Implementing it in the Monte-Carlo simulation was jut a matter of creating another if-statement, treating the vaccination rate fdt as a transition probability from S to R.

4. RESULTS

4.1. *Differential equations*

The simulation was run with the following configurations

Rate	A	B	C	D
a	4	4	4	4
b	1	2	3	4
c	0.5	0.5	0.5	0.5

Where a, b, c are the rates of respectively transmission, recovery and immunity loss. A, B, C and D are the four different set-ups and resulting plots are shown in Fig. 2.

As we can see from the results, each different value of the rate of recovery had a different equilibrium state, as expected. The higher the rate of recovery, the more people end up in the Susceptible group at equilibrium, and when the recovery rate goes as high as the infection rate ($b = a = 4$), the disease is eradicated within 15 days, meaning the entire population ends up in S.

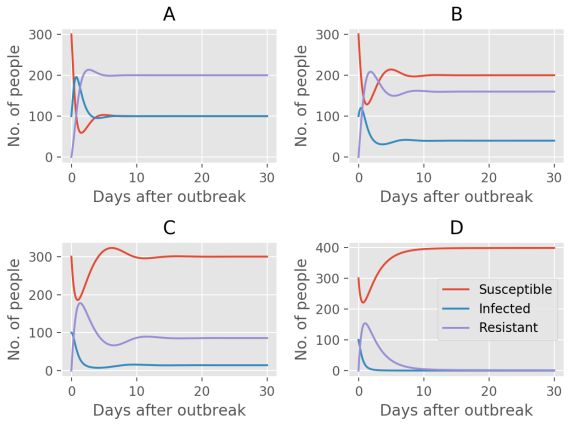

FIG. 2.— Plots of the population of the three different groups S, I, R, versus time in days for four different values of the recovery rate a. A, B, C and D have recovery rates 1, 2, 3 and 4, respectively.

In Table 1 we see that the equilibrium values reached by our simulation is identical to the values we get from calculating the steady state values using Equation (4) for A, B and C, and very close for D. This could be a great result when you are working with something else, however we want to have a more realistic simulation where we don't end up with one sixth of a person having the disease. This is, as we talked about in section 2.3, caused by us considering S, I and R as continuous variables.

A		**B**	
equilibrium	calculated	equilibrium	calculated
S=100	S=100	S=200	S=200
I=100	I=100	I=40	I=40
R=200	R=200	R=160	R=160
C		**D**	
equilibrium	calculated	equilibrium	calculated
S=300	S=300	S=398.51	S=400
I=14.29	I=14.29	I=0.16	I=0
R=85.71	R=85.71	R=1.33	R=0

TABLE 1
TABULATED VALUES OF THE EQUILIBRIUM VALUES REACHED BY OUR SIMULATION VS. THE VALUES CALCULATED USING THE STEADY STATE FOUND IN SECTION 2.1

4.2. *Monte Carlo simulation*

Changing the simulation to a Monte Carlo-based algorithm made a huge difference in that only one whole person can be transferred from one group to another at any time step. As we can see in Fig. 3, the plots follow the same tendency and equilibrate at approximately the same values. However, they are no longer smooth curves because of the limitation of one person moving at a time, and the randomness of the transition probabilities. We also see that the disease in simulation C has had a stroke of bad luck and died out. This is because of a combination of the randomness of moving between the different groups and the fact that C's equilibrium state only has 14.29 people in the Infected group.

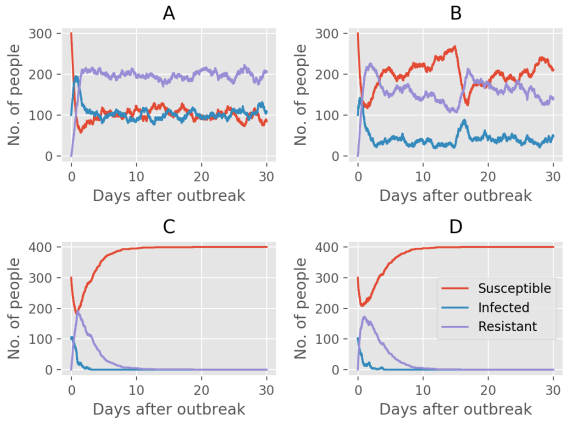

FIG. 3.— Amount of people in S, I and R vs. time in days using the Monte Carlo Algorithm. Disease died out in C and D, which was to be expected when not using continuous variables.

In Table 2 we see that we now have more what you would expect to see from data taken from a real life situation compared to a model. We see that the numbers are approximately at the calculated steady state values, but not exactly, and not with fractions of a person split up.

A		B	
equilibrium	calculated	equilibrium	calculated
S=99	S=100	S=191	S=200
I=91	I=100	I=42	I=40
R=210	R=200	R=167	R=160
C		**D**	
equilibrium	calculated	equilibrium	calculated
S=400	S=300	S=400	S=400
I=0	I=14.29	I=0	I=0
R=0	R=85.71	R=0	R=0

TABLE 2
EQUILIBRIUM VALUES REACHED BY THE MONTE CARLO SIMULATION. THESE WILL OF COURSE NEVER FULLY EQUILIBRIATE BECAUSE OF THE RANDOM NATURE OF THE SIMULATION, BUT THIS IS TO BE EXPECTED AND IS ACTUALLY DESIRED TO MORE CLOSELY MIMIC REAL LIFE.

4.3. *Vital dynamics*

By adding equal birth and general death rates, and a separate death rate for people infected with the disease, we can simulate how a deadly disease would kill the population over time. To simulate this we used the following initial values(And still a population of 400 with $S_0 = 300$ and $I_0 = 100$)

Rate	A	B
a	4	4
b	1	2
c	0.5	0.5
d	0.01	0.01
e	0.01	0.01
d_I	0.02	0.02

These values are not very realistic, but exaggerating the birth and death rates let's us demonstrate that the simulation has realistic properties.

As we can see from the result in Fig. 4, the differential equation version of this simulation is not very exciting. We let the simulation run for longer to show the decrease in the population, which is of course very smooth since exactly $0.02 \cdot I$ people die each time step. They both converge towards zero, slower and slower since N is decreasing. Over the first couple of days the death rate doesn't affect much, and all the groups reach approximately the same equilibrium values as in section 4.3 but they also slowly bleed off people. Even though people only die from the infected group($d = e$), we see that all three groups seem to be going down at the same rate. This is simply because the change in population in the Infected-group in each time step(-0.02I) Throws the system out of equilibrium, and the two other groups have to go down as well to compensate and reach a new equilibrium. This happens continuously throughout the simulation after equilibrium is reached.

We spent a lot of time trying to figure this out because we initially used an infected death rate d_I that was too high, resulting in the disease always getting eradicated in the first couple of days since the rate of transmission from S to I couldn't keep up with the deaths of people in the infected group.

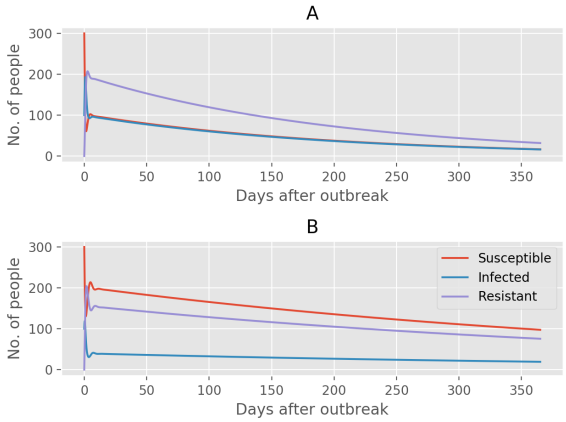

FIG. 4.— Differential equation simulation with added death rates. Each group reaches equilibrium about the same as they did in Fig. 2, but slowly lose people to the infected death rate.

In Fig. 5 we see that the Monte Carlo based simulation has some more interesting results. Since the added death and birth rates e and d are equal, the Runge-Kutta-simulation is not affected by it at all, the exact same number of people will be born and die naturally each time step, and we only see the effect of the infected death rate. In the Monte Carlo simulation the rates correspond to probabilities instead of fractions of a person, so we see the effects of randomness as the population of each group fluctuates drastically throughout the year. We also see that the infected death rate is slowly dragging the population down just like in Fig. 4, however we now reach a point where a combination of too few infected individuals, too high infected deathrate and "bad luck", leads to the last infected person to die and the remaining population can live on. We see that they still fluctuate, but are now in equilibrium since the birth and death rate is the same.

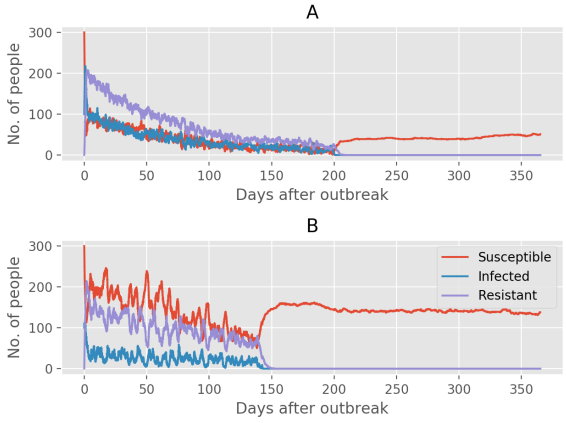

FIG. 5.— Monte Carlo simulation with added birth and death rates. Big fluctuations caused by general birth and death rates, and slow decline over time caused by infection death rate.

4.4. *Seasonal variation*

The addition of a rate of transmission that changes over time adds another layer of realism to this simulation. As we can see in Fig. 6 the different populations will increase and decrease to account for the new equilibrium

state in each step from the changing rate a. Due to the way we have designed the cosine function it repeats twice a year. Here we used an initial rate of transmission $a_0 = 4$, and a rate of recovery $b = 1$. We still include the vital dynamics, however we have lowered the rates to $(e, d, d_I) = (0.001, 0.001, 0.002)$ so that the people and the disease survives throughout the year.

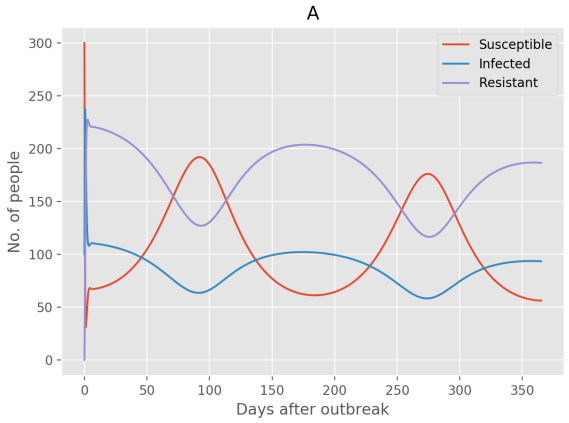

FIG. 6.— Seasonal variation added to the Runge-Kutta based simulation. The cosine function was defined so that the pattern repeats 2 times per year

adding seasons to our Monte Carlo simulation gave great results seen in Fig. 7. We can easily see that all groups S, I, R follow the general pattern we saw in Fig. 6, but with the added randomness of transmission probabilities, birth and death rates, and the slow decline from the infection death rate(Which is also seen in the runge-kutta-version).

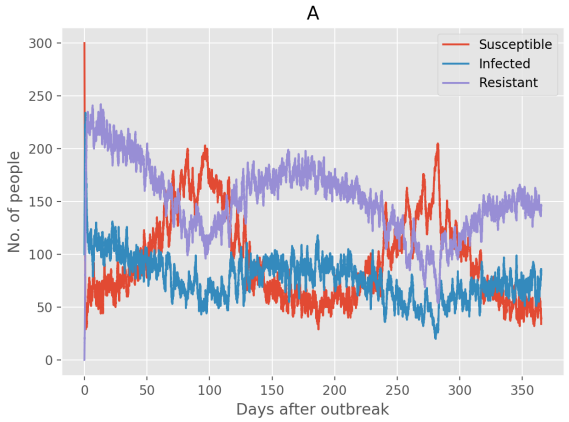

FIG. 7.— Monte Carlo simulation with added seasonal variations. One can still see the effects of transition probabilities, birth and death rates and the infected death rate, as well as the repeating pattern now acquired from adding seasons.

4.5. *Vaccination Campaign*

When adding vaccination to our model, we see that the rate of vaccination changes the equilibrium state of the system. In Fig. 8 we have vaccination rate $f = 5$, and we can see that there is a small deviation from the steady state previously achieved with these same initial values. This is the same system as A in the previous exercises with $a = 4$, $b = 1$ and $c = 0.5$. This system usually converged towards an equilibrium at $S, I, R = (100, 100, 200)$, but now S and I don't seem equal anymore.

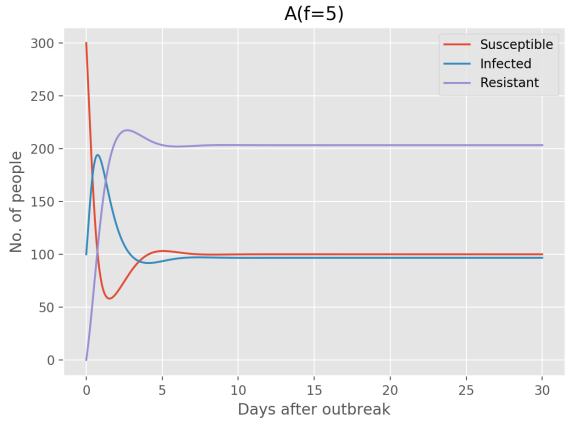

FIG. 8.— Runge-Kutta simulation of the basic SIRS-model with added vaccination rate f

If we take a look at Fig. 9 where we have $f = 50$, it becomes more clear what is happening. Increasing the vaccination rate f makes it so that the system reaches an equilibrium with a lower amount of infected people, and a higher amount of resistant people. The equilibrium value of susceptible people stays the same.

Lastly we have the Monte Carlo simulation in which, because of the random fluctuations, it's hard to tell what's different for $f = 5$, but in Fig. 10 we can see that for $f = 50$ it follows the same behaviour as our Runge Kutta model (9).

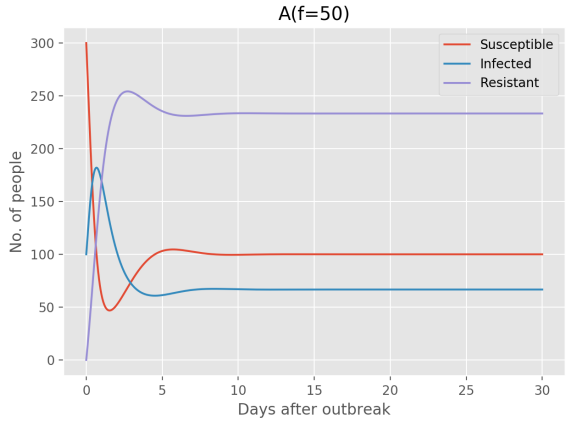

FIG. 9.— Runge-Kutta simulation, f has been increased to 50 so we can see how the steady state of the system depends on f

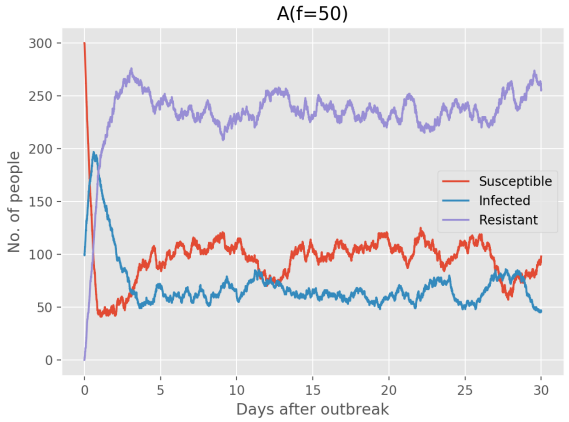

FIG. 10.— Monte-Carlo simulation of the basic SIRS model with vaccination rate.

5. CONCLUSION

The study of the SIR model allows us to understand the spread of disease in different kinds of populations. This information is invaluable for the betterment of living conditions across the Earth. There are many organizations devoted to studies similar to the one in this paper. One of these was used for reference (5) when developing our own model, along with the disease modelling text provided by H.J, Morten (4). In order to eradicate diseases worldwide vaccinations and preventive measures, I.E mosquito nets against malaria, have to be utilized. In order to do this effectively it is important to understand the effects of these factors on the transmission and recovery rates, and subsequently on the S, I, R partitions of the population. Our work does highlight some very important facts about the spread of disease, such as the the effectiveness of vaccination 9 and the critical point for the recovery rate 2, which does depend on biotechnological advances. These effects have been studied in this paper, although it is possible to further improve upon our final model.

Vaccination campaigns focused on eradicating an epidemic before it takes hold in the populations, I.E having an oscillating vaccination rate could be a possible improvement. Alternatively one could have a vaccination function which is either linear or exponential to signify the effect of the public getting educated and spreading the word between each other. An other improvement to our model would be using the extended SEIR and SEIRS model, explained very concisely at (5) (second link).

During this project we both learned a lot about simulations using differential equations as well as stochastic processes. We did run into some problems initially on task c). This was because we wanted to simulate the black plague in Europe, even though the result was as expected it did make us doubt our implementation. However, after playing around with the birth and death rate parameters we got to a result which we felt showed the details of our implementation.

Our results showed that when simulating diseases it is a lot more realistic to use a stochastic process, because else there can be weird effects present caused by the continuousness of the differential equation model. Such as, a third of an infected person infecting someone. This results in a longer epidemic time, whereas when simulating using our Monte Carlo method the disease would have died out. Once we added the seasonal variation of the transmission rate we saw exactly what we expected: the steady state of the system was now disturbed and the disease survived in a much more realistic way. Finally, after adding the vaccination term to our model we observed what we already knew. If the goal is to eradicate disease then vaccinations help a whole lot.

REFERENCES

[1]HilberTraum (2017) "Slopes used by the classical Runge-Kutta method" https://en.wikipedia.org/wiki/Runge%E2%80%93Kutta_methods#/media/File:Runge-Kutta_slopes.svg

[2]Lowen AC, Mubareka S, Steel J, Palese P (2007) Influenza Virus Transmission Is Dependent on Relative Humidity and Temperature. PLoS Pathog 3(10): e151. https://doi.org/10.1371/journal.ppat.0030151

[3]Riedel, Stefan. Edward Jenner and the history of smallpox and vaccination Proceedings (Baylor University. Medical Center) vol. 18,1 (2005): 21-5. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1200696/

[4]Hjorth-Jensen, Morten (2018). "Disease Modelling for FYS3150 course fall semester"

[5]https://institutefordiseasemodeling.github.io/Documentation/general/model-sir.html https://institutefordiseasemodeling.github.io/Documentation/general/model-seir.html