FYS3150/4150 - Project 5
Numerical simulation of SIRS-model

Aram Salihi[1], Adam Niewelgowski[1]

[1]Department of Physics, University of Oslo, N-0316 Oslo, Norway

December 18, 2018

Abstract

In this numerical study we are going to compare two very different methods to solve the famous SIRS model. The SIRS model can easily be explained by three coupled time dependent differential equations , and can be easily solved numerically. Due to the discrete and random nature of this compartment model. We have chosen to compare the solution of the differential equations, with the result from the classical Monte Carlo simulation by using basic probabilty transitions rates. Comparing the result produced from these methods we found out that Monte Carlo is quite accurate and agrees with RK4 results almost all the time. The conclusion is that Monte Carlo method is a quite reliable and accurate method to use when solving problem in SIRS model.

Contents

1 Introduction

Having the knowledge how deadly infectious spreads among a closed or open group, can be crucial in order to stop a deadly disease before its to late. In order to have this knowledge we must simply build a mathematical model which can describe this notion. In early seventeenth hundred such models were derived and used to predict of how disease behaved when introducing a vaccine. This field have expanded into mathematical epidemiology.

The most famous and commonly used is the SIR (**S**usceptible, **I**nfected, **R**ecovered) model. The philosophy behind this model is to introduce three different compartment groups into a system, where one of the group carries a infectious disease. Assuming that each individual from each group has the possibility to interact with each other. This will lead to people become ill from the disease, but also recover from the disease depending how infectious and deadly the disease is. In order to describe this notion, we must develop a set of coupled differential equations, which explains this type of behaviour in a closed or open system. From this we can develop a understanding how fast a disease can spread, but also how fast people heal and recover from a infectious disease.

Many models in mathematical epidemiology are derived from this basic idea used in SIR model. Models such as SIRS, MSIR, SEIR, and more. However this model is not only limited to this field of studies. The SIR model can also be expanded and be used in the study of traffic congestion see [Jianjun Wu and Sun, 2004]. The main22 idea is to have three different groups, where objects are moved around between groups based on system's conditions.

In this numerical study we will be studying the model SIRS which is explained in depth in the next section. We wish wish to study different events created by this model.

- We will first introduce a non deadly infectious disease, and simply assume no one dies from it. We will also assume that no one from each compartment groups dies of natural cause and gives birth. Thus simulating a closed system where the total population number is fixed.
- We will then continue our studies to a open system, where we allow people to die from natural cause and the disease itself, but also introduce a birth rate.
- We will also study how some seasonal diseases impact the model by introducing a new time depdendent osciallating transmission rate. For diseases such as influenza, the rate of transmission depends largely on the time of year we are in.
- We also introduce a vaccine for an arbitray disease.

In order to solve these coupled time dependent system we have chosen two types of approach. First we will directly solve the coupled differential equation by using fourth order Runge-Kutta method. We will then compared it with Markov Chain Monte Carlo method, where the transition from one state/group to another state/group be based on probability and random numbers. The main goal is to compare the result from the Monte Carlo simulation against the result from Runge Kutta. We are simply doing this to get a better understanding if Monte Carlo simulation is stable enough to produce good result when applying to SIRS model.

2 Theory

In this section we will present the theoretical framework and different types of model used in this numerical study. Keep in mind that the algorithm Runge-Kutta of fourth and probability transition in Monte Carlo is also derived in this section. For implementation of Monte Carlo and Runge kutta please see section (3)

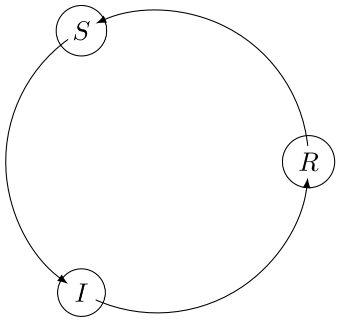
2.1 SIR-model

Consider a group of "susceptible" with population S, which has the possibility to be infected by a deadly or non-deadly disease by someone which carries this infection. The person who carries this infection is in the "infected" group with population I. We will now assume that there is some possibility (depends on the disease) that one or more from group I heals and recoveres from the disease, and becomes immune. Thus, the persons who heals moves to the group "recovered" with population R. These three variables represent the number of people in each compartment at particular time t. Keep in mind that these three group are time dependent even though the number of people in each compartment are constant. Thus the total population in this system follows

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

2.2 SIRS-model

We will now consider a bit more general model compared with previous one presented in (**??**) called SIRS. Assuming no babies born into these three group and the possibility of dying being zero. We will now consider a group of recovered people from a disease $\mathcal{D}$. We will now assume that these people will have the ability to loose immunity from this disease and become susceptible. Meaning that they can be infected by someone with disease $\mathcal{D}$. We will now have circular system. The idea behind is demonstrated in the figure down below



In other words we have consider a case with a closed system. The change in the population for this system is described by following coupled differential equations (gathered from [Hjorth-Jensen, 2018])

$$\frac{\mathrm{d}S}{\mathrm{d}t} = cR - \frac{aSI}{N} \tag{2}$$
$$\frac{\mathrm{d}I}{\mathrm{d}t} = \frac{aSI}{N} - bI \tag{3}$$
$$\frac{\mathrm{d}R}{\mathrm{d}t} = bI - cR \tag{4}$$

Where a describes the transmission rate of the disease, b describes the rate of recovary, and c describes the loss of immunity. The change described by these equations will always keep the total population $N(t)$ constant. This can be easily seen when adding S', I' and R' together.

Vital dynamics (open system) We will now consider a open system. Meaning we will have a even more realistic scenario compared with two previous models. We will now introduce a death rate which can occour in group S, I and R. The cause of death is not by the disease itself, but describing other reasons such as natural cause of death such as heart attack. We will define this parameter rate as d. Further we will introduce a death rate of people dying from the disease itself, and call this rate for d_i. To make it even more realistic we will introduce a birth rate parameter

e. This parameter explains the rate of babies added to the system. We will now add a few terms to the differential equations introduced in (4)

$$\frac{\mathrm{d}S}{\mathrm{d}t} = cR - \frac{aSI}{N} - dS + eN$$
$$\frac{\mathrm{d}I}{\mathrm{d}t} = \frac{aSI}{N} - bI - dI - d_I I$$
$$\frac{\mathrm{d}R}{\mathrm{d}t} = bI - cR - dR$$

(the equations are gathered from [Hjorth-Jensen, 2018]) We will further assume that all babies added to the system are susceptible. If we take the sum of the these equation, we will notice that

$$\frac{\mathrm{d}S}{\mathrm{d}t} + \frac{\mathrm{d}I}{\mathrm{d}t} + \frac{\mathrm{d}R}{\mathrm{d}t} = eN - dR - (d_i - d)I \tag{5}$$

From this we can see that the population number N is never constant. It will either decrease or increase depending how the factors are behaving with respect to each other. Notice that in order to keep a increase in population with this model, eN must dominate

$$eN >> dR + (d_i + d)I \tag{6}$$

Considering the other case, simulating an apocalypse with deadly disease/infection with a decreasing population, we must have

$$eN << dR + (d_i + d)I \tag{7}$$

Thus $dR + (d_i + d)I$ must dominate in order to bring the population down.

Seasonal variation As introduced in the introdcution, we also wish to study how the population is affected by oscillating diseases such as influenza. We will slightly change the transmition rate a to be time dependend

$$a(t) = A\cos(\omega t) + a_0 \tag{8}$$

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

Vaccination f(t) Here we introduce a new constant, f, which will break our circular model, and allow transition directly from **S** to **R**. The set of differential equation will then look like

$$\frac{dS}{dt} = cR - \frac{aSI}{N} - f$$
$$\frac{dI}{dt} = \frac{aSI}{N} - bI$$
$$\frac{dR}{dt} = bI - cR + f$$

Keep in mind we have removed the parameter explaining the death and the birth rate the system. Meaning we wish to investigate the maximum potential of a vaccine. In this numerical study we wish to implement a time dependent vaccination rate $f(t)$, to simulate the improvement of the vaccination as more people get vaccinated as the time goes. The function which satisfies this is

$$f(t) = f_0 \ln(1+t) + f_0 \tag{9}$$

Where f_0 is the initial rate of how many people gets treated with this type of vaccine, and t is the time.

2.3 Numerical algorithm

2.3.1 Runge-Kutta 4

Runge-Kutta of fourth order is widely used numerical algorithm for solving ordinary differential equations ,due to its precision which makes it a superior and relatively fast method. As other numerical ODE algorithms it is also based on Taylor expansion theorem. The idea behind this method is that it provides intermediate steps to calculate y_{i+1}. Consider a general first order ordinary differential on the form:

$$\frac{\mathrm{d}y}{\mathrm{d}t} = f(t, y) \tag{10}$$

We wish to solve this ODE, i.e, we want to integrate both sides with respect to t. Keep in mind that y and t can represent all kind of phenomenas.

$$y(t) = \int f(t, y)dt \tag{11}$$

The idea now is to discretize our solution interval $y \to y_i$ and our interval $t \to t_i$. We want to approximate the next solution y_{i+1} by using the previous solution y_i with a step of

$$\int_{t_i}^{t_{i+1}} f(t, y)\,\mathrm{d}t \tag{12}$$

Thus the equation we want to end up with

$$y_{i+1} = y_i + \int_{t_i}^{t_{i+1}} f(t, y)\,\mathrm{d}t \tag{13}$$

We now wish to taylor expand $f(t, y)$ to fourth order with respect to t around the point

$$t_{i+1/2} = \frac{t_{i+1} + t_i}{2} \qquad y_{i+1/2} = \frac{y_{i+1} + y_i}{2} \tag{14}$$

We are now interested in finding a good aproximate to the integral we want to evaluate by using the Taylor expansion. For the sake of simplicity we only focus on the one dimensional problem and try to find a approximation for the integral of $f(t)$. Consider the following

$$\int_{t_i}^{t_{i+1}} \left[f(t_{i+1/2}) + f'(t_{i+1/2})(t - t_{i+1/2}) + \frac{f''(t_{i+1/2})(t - t_{i+1/2})^2}{2} + \frac{f'''(t_{i+1/2})(t - t_{i+1/2})^3}{6} + \frac{f^4(\xi)(t - t_{i+1/2})^4}{24} \right] \mathrm{d}t$$

Where the local error term in the taylor expansion is $\frac{f^4(\xi)(t - t_{i+1/2})^4}{24}$. Integrating this, we will obtain the following expression

$$\left[f(t_{i+1/2})t + \frac{f'(t_{i+1/2})(t - t_{i+1/2})^2}{2} + \frac{f''(t_{i+1/2})(t - t_{i+1/2})^3}{6} + \frac{f'''(t_{i+1/2})(t - t_{i+1/2})^4}{24} + \frac{f^4(\xi)(t - t_{i+1/2})^5}{120} \right]_{t_i}^{t_{i+1}}$$

Evaluating this at the point t_i and t_{i+1}, and thereby adding the expression together, we will obtain

$$\int_{t_i}^{t_{i+1}} f(t)\,\mathrm{d}t = 2hf(t_{i+1/2}) + h^3 \frac{f''(t_{i+1/2})}{3} + h^5 \frac{f^4(\xi)}{60} \tag{15}$$

Where $h = \frac{t_{i+1} - t_i}{2}$ is the stepsize if using two points. Using an approximation for the second derivative (this is gathered from [Mørken, 2017]), we can write the right hand side as

$$2hf(t_{i+1/2}) + \frac{h^3}{3} \left(\frac{f(h + t_{i+1/2}) - 2f(t_{i+1/2}) + f(t_{i+1/2} - h)}{h^2} - \frac{h^2}{12} f^4(\xi) \right) + h^5 \frac{f^4(\xi)}{60} \tag{16}$$

Adding all the terms together, we can write

$$\int_{t_i}^{t_{i+1}} f(t)\,\mathrm{d}t = \frac{t_{i+1}-t_i}{6}\left(f(t_i)+4f(t_{i+1/2})+f(t_{i+1})\right) - h^5\frac{f^4(\xi)}{90} \tag{17}$$

Thus we the approximation we derived is called Simpsons method. This can be easily expanded to

$$\int_{t_i}^{t_{i+1}} f(t)\,\mathrm{d}t = \frac{h}{6}\left(f(t_i,y_i)+4f(t_{i+1/2},y_{i+1/2})+f(t_{i+1},y_{i+1})\right) + \mathcal{O}(h^5) \tag{18}$$

Where we now use a general stepsize h defined as $\frac{t_{i+1}-t_i}{N}$, where we have discretized with N points. Using this approximation into (13), we can write

$$y_{i+1} \approx y_i + \frac{h}{6}\left(f(t_i,y_i)+4f(t_{i+1/2},y_{i+1/2})+f(t_{i+1},y_{i+1})\right) \tag{19}$$

The problem we will now encounter is that we do not know what the value of $y_{i+1/2}$ and y_{i+1} is. In order to solve this problem we want to split this into two parts, and write

$$y_{i+1} \approx y_i + \frac{1}{6}\left(hf(t_i,y_i)+2hf(t_{i+1/2},y_{i+1/2})+2hf(t_{i+1/2},y_{i+1/2})+hf(t_{i+1},y_{i+1})\right) \tag{20}$$

The idea now is to approximate $y_{i+1/2}$ in two steps. In order to do this we first the need the value of $hf(t_i,y_i$, we can define this constant as k_1. We can then use this to calculate $y_{i+1/2}$ using Eulers method

$$y_{i+1/2} = y_i + \underbrace{\frac{h}{2}f(t_i,y_i)}_{k_1/2} \tag{21}$$

To compute

$$k_2 = hf(t_{i+1/2}, y_i + k_1/2) \tag{22}$$

Using this philosophy until y_{i+1}, we can define

$$k_3 = hf(t_{i+1/2}, y_i + k_2/2) \tag{23}$$
$$k_4 = hf(t_{i+1}, y_i + k_3) \tag{24}$$

Thus, this is giving us

$$y_{i+1} \approx y_i + \frac{1}{6}\left(k_1+2k_2+2k_3+k_4\right)) \tag{25}$$

This expression is called Runge Kutta of fourth order with a local error going as $\mathcal{O}(h^5)$. As we see this expression is quite accurate even though if we use a large stepsize h it is quite superior compared to Leap frog, Euler Cromer and Eulers midpoint. The algorithm is as follows:

- Find the first coefficient

$$k_1 = hf(t_i,y_i)$$

- Use Euler's method to find $y_{i+1/2}$, so the next coefficient is

$$k_2 = hf(t_i + h/2, y_i + k_1/2)$$

- Improve the slope by calculating by

$$k_3 = hf(t_i + h/2, y_i + k_2/2)$$

- Last coefficient is computed as following

$$k_4 = hf(t_i + h, y_i + k_3)$$

- Finally, update y_{i+1}

$$y_{i+1} = y_i + \frac{1}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right)$$

For more in depth explaination please see [Hjorth-Jensen, 2015].

2.4 Monte Carlo simulation

The philosophy behind Monte Carlo simulation is to draw a number from a probabilty transition or density, and compared it with random uniformed number in the interval $[0, 1]$. In our case we assume that S, I and R are continous, but in reality S, I and R are discrete. Using the idea of randomness, we can use the philosophy behind Monte Carlo simulation to construct this discrete movement between groups. In order to do this we must a have set of probability transitions explaining the move from $S \to I$, $I \to R$ and $R \to S$. The beauty of this is that it is easily expanded, and probability transition explaining other moves can easily be implemented. The implementation of this is explained in section (3.6). In order to make a move from a general group $j \to q$, the probabilty is then, a small timestep Δt multiplied with the number of people moving from $j \to q$. Recalling the factors used in the differential in subsection 2.2 SIRS-model, section 2.2 Vital dynamics (open system) and section 2.2 Vaccination f(t), we can define probability transition to a another compartment group as following:

- Lets assume that a susceptible person from group S interacts with a infected person from group I. Depending how large the transmission rate parameter a of some disease $\mathcal{D}$ is. The number of people moving from S to I at small time step Δt is

$$S \to I = \frac{aSI}{N} \tag{26}$$

The probabilty transition is then is defined as

$$P(S \to I) = \frac{aSI}{N}\Delta t \tag{27}$$

Using the same philosophy we can define other probabilty transition as following:

- The probabilty transition of making a move from $I \to R$ is

$$P(I \to R) = bI\Delta t \tag{28}$$

- The probabilty transition of making a move from $R \to S$ is

$$P(R \to S) = cR\Delta t \tag{29}$$

As explained earlier, the SIRS model can be easily expanded. Thus we can find the transition probabilities for section 2.2 Vital dynamics (open system) and section 2.2 Vaccination f(t) by using the same philosophy. The probabilty transition are defined as

$$P(B \to S) = eN\Delta t \tag{30}$$
$$P(S \to D) = dS\Delta t \tag{31}$$
$$P(I \to D) = (d + d_I)I\Delta t \tag{32}$$
$$P(R \to D) = dR\Delta t \tag{33}$$

Here we have defined two new subgroups B and D. Where B is a group of babies comming into to system S, and D is the group dead people when dying from the disease itself or a some other natural cause. For the case with vaccination we have

$$P(S \to R) = f\Delta t \tag{34}$$

How we update the number of people in each compartment group is explained in section (3.6).

3 Method

In this section we will expalin how the different type method are implement into our code. Keep in mind that the code is writeen in C++.

3.1 Structure of the code

The complete code consist four classes and object oriented in a more pythonic way compared with the traditional C++ object orienting. The main.cpp reads and calls different class instances.

- solver.cpp : This class contains different method which has different task to solve the problem introduced. This class has to solver methods implemented. Runge Kutta method of fourth order and a Monte Carlo function. If one desires to use another solver, it can be easily implemented since most of the functions are independent of each other. Keep in mind to call the new solver in execute_solver method. This class also calculates the standard deviation and the variance.
- read_parameter.cpp : This class reads parameters from parameters.txt
- savetofile.cpp : This class saves result into a .txt file.

3.2 Runge-Kutta

For the sake of accuracy we have chosen to implement Runge-Kutta of fourth order to achieve best possible results. Keep in mind, as explained in the previous section the code is fully object oriented and vectorized. Meaning additional numerical methods for solving the ODE is easily implemented. In order to use this method, first initialize the initial conditions, and then follow these step defined down below

- Find the first coefficient
$$k_1 = hf(t_i, y_i)$$
- Use Euler's method to find $y_{i+1/2}$, so the next coefficient is
$$k_2 = hf(t_i + h/2, y_i + k_1/2)$$
- Improve the slope by calculating by
$$k_3 = hf(t_i + h/2, y_i + k_2/2)$$
- Last coefficient is computed as following
$$k_4 = hf(t_i + h, y_i + k_3)$$
- Finally, update y_{i+1}
$$y_{i+1} = y_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

3.3 Sampling of Monte Carlo runs

When using Monte Carlo we must expect that for each run, the result will deviate by some decimals from the expected numerical result In order to minimize this deviation, we can sample the same simulation N_{sample} times until the desired simulation time is reached. The idea is to look at the average of the result when sampling N_{sample} times. This is simply done by adding the point $N_{samples}$ times for each iteration of the Monte Carlo simulation, and then divide by $N_{samples}$. From this we will get the average of the result which is signifcantly more smoother and nicer to analyze. From this we can also calculate the variance for each time point, and use this to calculate the standard deviation. The sampling method is implemented in the following way

```
  for(int n = 0; n < nsamples; n++){
    while(time(mc_step)<= final_time){
      SIR(mc_step,0) += S/double(nsamples);
      SIR(mc_step,1) += I/double(nsamples);
      SIR(mc_step,2) += R/double(nsamples);
    }
  }
```

3.4 Variance

As explained previously we are interested in the average of the result computed, this means that we can compute the variance. The variance is simply defined as squared of the deviation from the mean value, mathematically

$$Var(X) = (X - \mu)^2 \tag{35}$$

Where X is our random result from each Monte Carlo cycles and μ is the computed average of X. In our case we want to look at the average variance from each $N_{samples}$. This is simply

$$Var(X) = \frac{(X - \mu)^2}{N_{samples}} \tag{36}$$

Notice that X and μ are matrices containing information about the deviation in S I and R data with number of Monte Carlo cycles as length in each dimension. This method is implemented in the following way

```
for(int n = 0; n < nsamples; n++){
  for(int j = 0; j < length; j++){
      varS(j) += (avgS(j)-S(j))*(avgS(j)-S(j))/(nsamples-1);
      varI(j) += (avgI(j)-I(j))*(avgI(j)-I(j))/(nsamples-1);
      varR(j) += (avgR(j)-R(j))*(avgR(j)-R(j))/(nsamples-1);
    }
  }
```

Where length is the number of iteration completed in the Monte Carlo simulation until final time is reached. The goal of this is to have the variance such that we can calculate the standard deviation $\sigma = \{\sigma_S, \sigma_I, \sigma_R\}$ for each compartment group. This number simply tells us where the expected data should lie.

3.5 Standard deviation

As explained previously section (3.4) we want to estimate the standard deviation in order to have a idea of where the result should lie. The standard deviation is defined as

$$\sigma = \sqrt{\frac{\sum_i Var(X_i)}{N}} \tag{37}$$

This method is implemented in the following way

```
for(int n = 0; n < nsamples; n++){
  for(int j = 0; j < length; j++){
    sigmaS += sqrt(varS(j)/(nsamples*length));
    sigmaI += sqrt(varI(j)/(nsamples*length));
    sigmaR += sqrt(varR(j)/(nsamples*length));
  }
}
```

Where length is the number of iteration completed in Monte Carlo simulation until final time is reached.

3.6 Monte Carlo simulation

In the section of theory we explained and derived the probability transitions from one compartment group to another. Using these probability transitions we can simply draw a random uniformed number r in the inverval $[0,1]$ to check if $r \leq P_{i \to j}$. If the condition is fullfilled we then move a human from one compartment group i to j. This is done by subtract and adding 1 to temporary variables. The reason why use a temporary variables is that have several probability transitions conditions, and every event can happen at once. Thus we must draw a new number each time we check these transitions. When have checked all the conditions, we will then add the corresponding temporary variable to its compartment group. In the code, we have defined three variables as "susceptible", "infected" and recovered in the end are these added to S, I and R. A small part of the Monte Carlo method is presented down below

```
random_device rd;
mt19937_64 gen(rd());
uniform_real_distribution<double> RandomNumberGenerator(0.0,1.0);

 //probabilty transitions of moving from S to I to R
 S_to_I = (parameter[0]*S*I/N)*dt;
 I_to_R = parameter[1]*I*dt;
 R_to_S = parameter[2]*R*dt;

 susceptible = infected = recovered = 0;

 if(RandomNumberGenerator(gen) < S_to_I){
     susceptible -= 1;
     infected += 1;
 }

 if(RandomNumberGenerator(gen) < I_to_R){
     infected -= 1;
     recovered += 1;
 }

 if(RandomNumberGenerator(gen) < R_to_S){
     recovered -= 1;
     susceptible += 1;

 }
 S += susceptible;
 I += infected;
 R += recovered;
```

4 Results

In this section we will present the different result from the different type of SIRS model. Keep in mind that we have used $N_{mc} = 10000$ Monte Carlo cycles with $N_{samples} = 100$ sampling runs. In the simulation where RK4 where used, we used $N = 100000$ iterations with a $h = 1/N$. Important infomartion suc as parameter used are given in the title of the result. Keep in mind that the standard divitation $\sigma = \{\sigma_S, \sigma_I, \sigma_R\}$ for the Monte Carlo simulations are also given in the caption of the figures.

4.1 SIRS model for closed system

In this section we will present the result when using different values of the recovery rate b. We used $b = \{1,2,3,4\}$.

4.1.1 b = 1

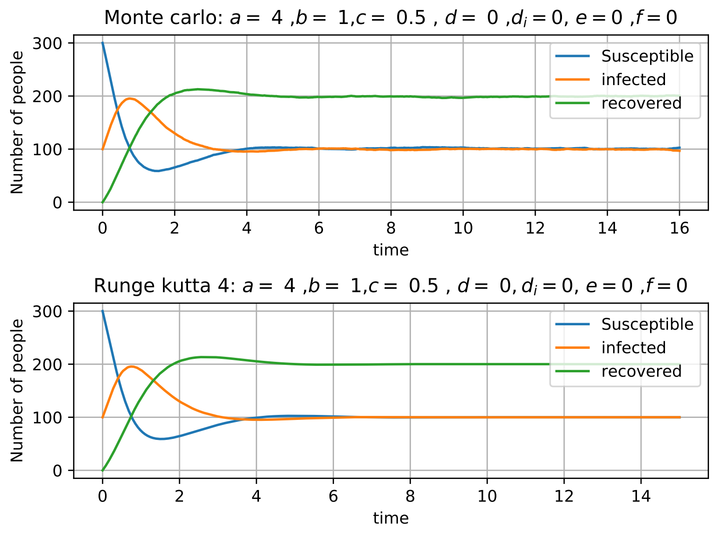


Figure 1: The standard diviation for the monte carlo simulation is: $\sigma_R = 0.00833$, $\sigma_I = 0.00868$ and $\sigma_R = 0.00655$.

4.1.2 b = 2

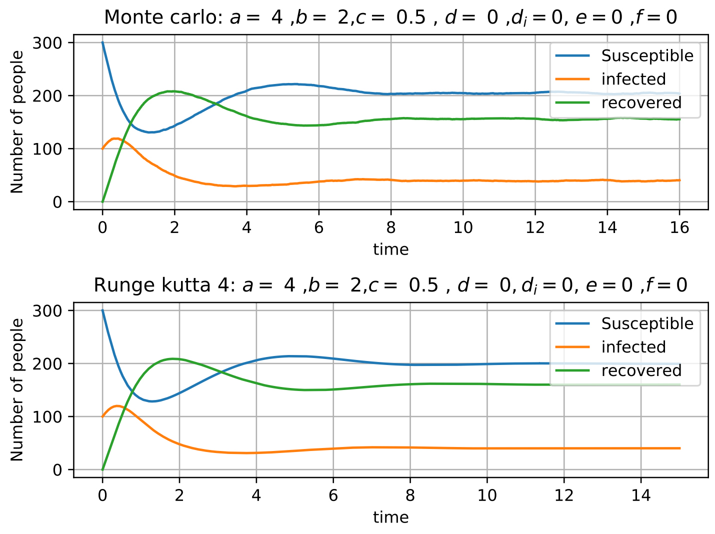


Figure 2: The standard diviation for the monte carlo simulation is: $\sigma_R = 0.04218$, $\sigma_I = 0.01513$ and $\sigma_R = 0.03091$.

4.1.3 b = 3

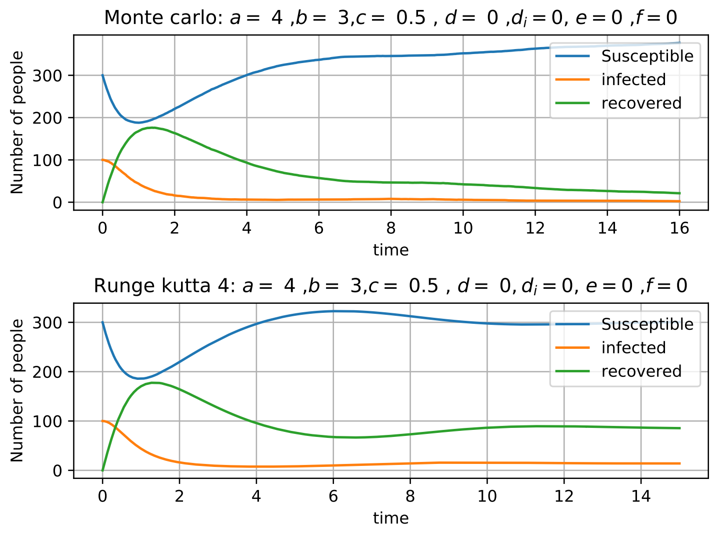


Figure 3: The standard diviation for the monte carlo simulation is: $\sigma_R = 0.02416$, $\sigma_I = 0.00577$ and $\sigma_R = 0.02070$.

4.1.4 b = 4

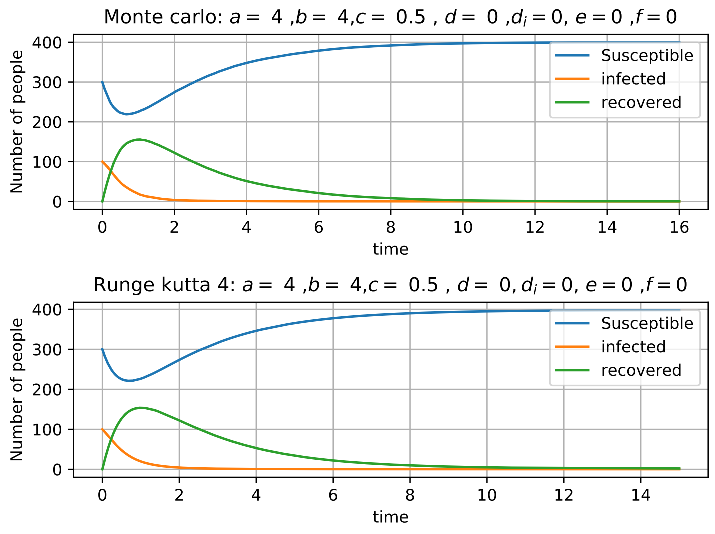


Figure 4: The standard diviation for the monte carlo simulation is: $\sigma_R = 0.0075805$, $\sigma_I = 0.0014021$ and $\sigma_R = 0.006399$.

4.1.5 SIRS model with vital dynamics:

Using the set of rate of recovery $b = \{1, 2, 3, 4\}$ as before with parameters $d = d_i = 0.1$ and $e = 0.2$.

4.2 b = 1

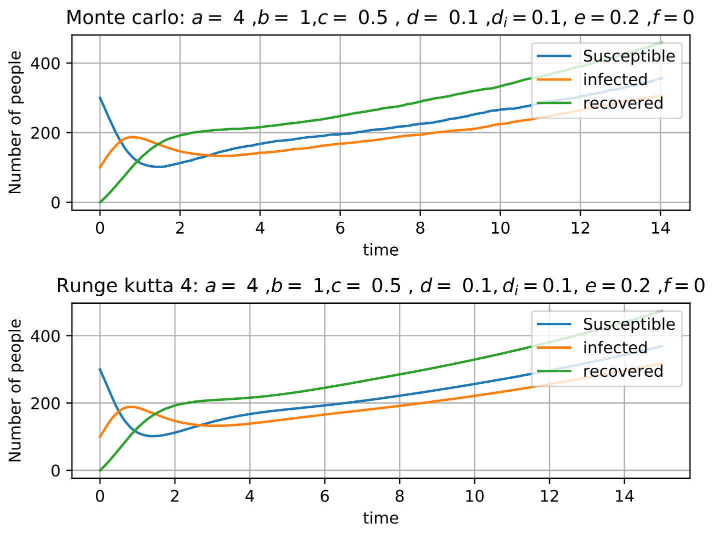


Figure 5: The standard diviation for the monte carlo simulation is: $\sigma_R = 0.01545$, $\sigma_I = 0.01513$ and $\sigma_R = 0.01291$

4.2.1 b = 2

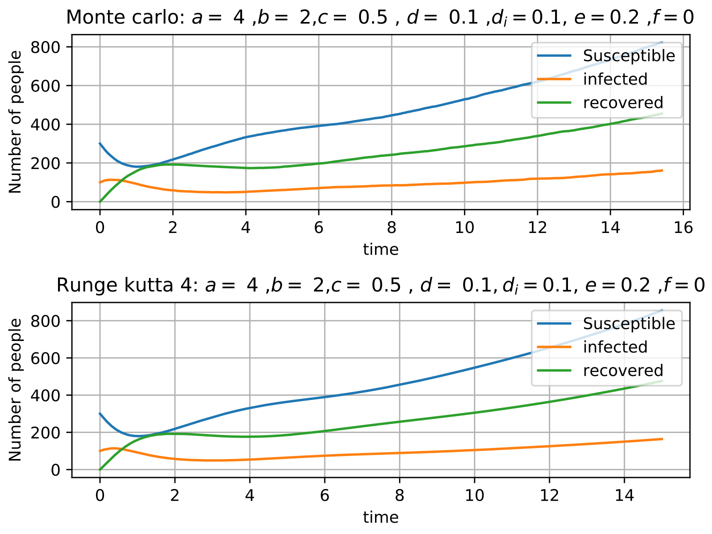


Figure 6: The standard diviation for the monte carlo simulation is: $\sigma_R = 0.03111$, $\sigma_I = 0.01448$ and $\sigma_R = 0.03010$

4.2.2 b = 3

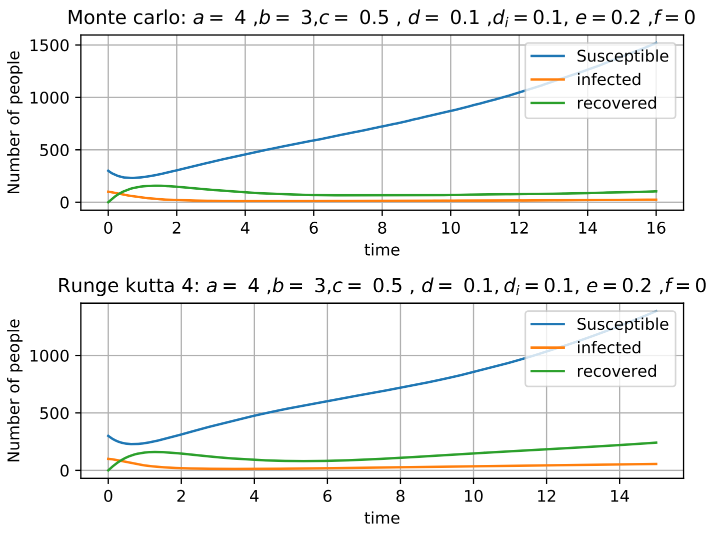


Figure 7: The standard diviation for the monte carlo simulation is: $\sigma_R = 0.01831$, $\sigma_I = 0.0.000482$ and $\sigma_R = 0.0.00241$

4.2.3 b = 4

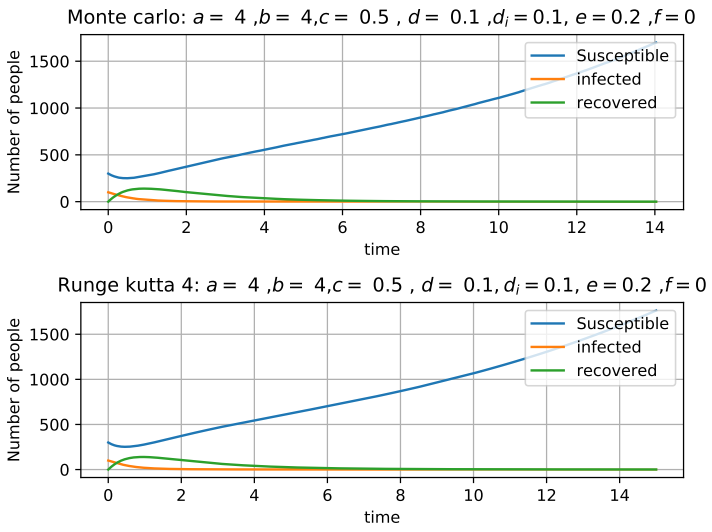


Figure 8: The standard diviation for the monte carlo simulation is: $\sigma_R = 0.05652$, $\sigma_I = 0.01513$ and $\sigma_R = 0.01291$.

4.3 Seasonal variation, no death and birth rate

4.3.1 $a = a(t)$, $b = 2$, $c = 0.5$, $d_i = d = e = f = 0$

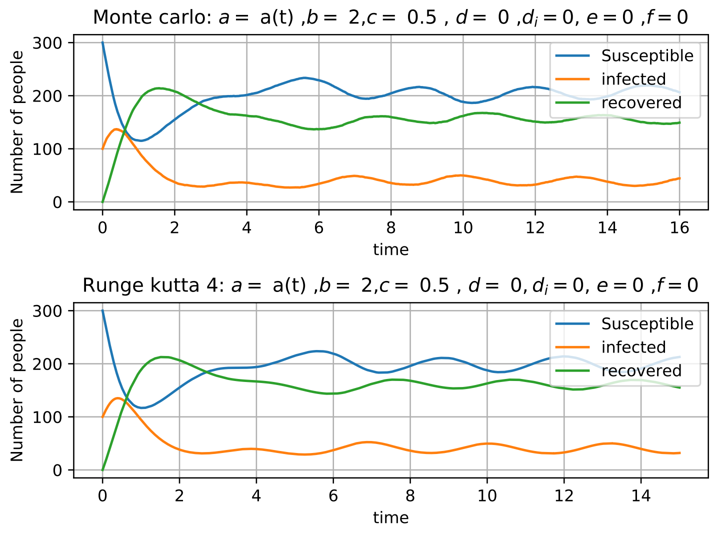


Figure 9: The standard diviation for the monte carlo simulation is: $\sigma_R = 0.014741$, $\sigma_I = 0.0076703$ and $\sigma_R = 0.010293$

4.4 Vaccination, no death and birth rate

4.4.1 $a = a(t)$, $b = 1$, $c = 0.5$, $d_i = d = e = 0, f = 0.7$

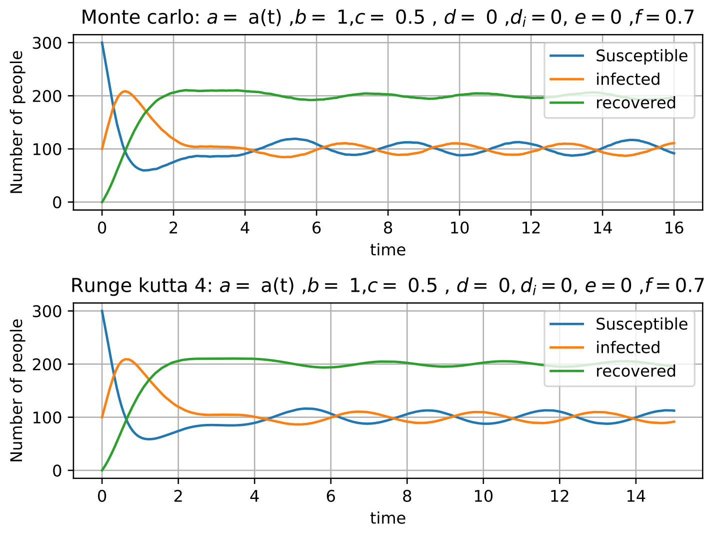


Figure 10: The standard diviation for the monte carlo simulation is: $\sigma_R = 0.008605$, $\sigma_I = 0.0081643$ and $\sigma_R = 0.0064243$

5 Discussion

5.1 SIRS closed system

5.1.1 Numerical Behaviour

Comparing the numerical behaviour of RK4 relative to Monte Carlo and vice versa, it is hard to find differences between these to results. We clearly see the same type behaviour in both methods. This can be easily seen in figure (1), (2) and (4). If we further analyze these result, we do notice a slightly odd behaviour in the Monte Carlo result for $b = 3$ in figure (3). Comparing this with RK4 we should expect a local maximum at $t \approx 6$, where as in the Monte Carlo simulation it is still increasing. The explaination behind this odd behaviour is that it could need more sampling to find this specific point (local maxima as seen in RK4). Another explaination of this behaviour is that the probabilty transition derived is a rough estimate of people transitioning from one group to another. Nevertheless when looking at the standard diviation are quite acceptable.

5.1.2 Theoretical view of closed SIRS model

As explained the theory section of closed SIRS model, we have assumed no death and birth rate. Thus, we have assumed a harmless disease, because of this we expect that N is constant. When analyzing the results, as expected the population is constant at all time. This behaviour can especially be seen when the different variables $S(t)$, $I(t)$ and $R(t)$ converges to a certain value. However the number of people in each compartments groups differs quite signifcantly. For $b = 1$ we see that the number of infected increases and peaks at time $t \approx 1.1$ with a value of $I \approx 192$,and then converges to 100 again. This can be interpeted as a semi strong flu. When having a small population with a intial infected of 100 with a relatively low rate of recovery it is expected that the disease will still be there. Increasing the value of b the number of infected drasticly decreases ,and goes towards zero for increasing value of b as expected. This type of behaviour can be interpeted as strong immune system towards this disease and recovers faster.

5.2 SIRS open system (vital dynamics)

5.2.1 Numerical Behaviour

Comparing the two method we hardly find any differences when directly analyzing the graphs given in figure (1), (2) , (3) and (4). This means when introducing vital dynamics to SIRS model the Monte Carlo simulation is quite stable and reliable. With low standard diviation, it is to be expected that the expected result lies within that area.

5.2.2 Theoretical view of open SIRS model

Analyzing the result we clearly see a large increase of the population. This means, the choice of our parameters e , d_i and d have produced a world with a increasing population. analyzing figure (5) and (6) and comparing it with figure (7) and (8). Since the number of infected in figure (5) and (6) slowly increases, we can interpret it as a small disease since we hardly see any dramatic decrease in the number of infected people. Thinking about the general death rate, we clearly see that this rate has also minimal effect on the system since the number of susceptible is always increasing. Thus the death rate is almost neglible.

Looking at the cases for $b > 2$ the number of infected dramatically decreases and converges towards zero. Here we clearly see that death rate of infected people has an big inpact of group I. A explaination to this: is that the number of people recovering from the disease is also decreasing. The disease kills the infected people fast, and has not the chance to infect susceptibles. Thus the disease dies out naturally. This can be seen in figure (8). In general, we clearly see that the factor

eN dominates eventhough the birth rate is quite low

$$eN >> dR + (d_i + d)I \tag{38}$$

This means that the system is quite sensitive for cases when adding new susceptibles into the system.

5.3 SIRS seasonal variation

5.3.1 Numerical Behaviour

Comparing the result again, we can confirm that the Monte Carlo simulation is quite reliable method when having time dependent parameters. We do notice small differences. This small difference may be fixed by increasing the number of sampling, but this is just a quick guess. After increasing the number of samples the small differences may still be there. Nevertheless, the system is quite stable and behaves in the same manner as the RK4 result. With low standard diviation, it is to be expected that the expected result lies within that area.

5.3.2 Theoretical view of seasonal variation

When analyzing figure (9) we notice that the number of infected is all time high for $t \approx 0.4$. This can be interpreted as the cold period of the year/season and many people gets the cold or the influenza. After some time the number of infected people decreases and number of recovered people increases. This can be seen as warmer period comming and the immune system get stronger. After a while the system stabilizes and the number of people in each compartment group oscillates between moving from one group to another. This is a normal behaviour, which can be seen in every day life.

5.4 SIRS time dependent vaccine and seasonal variation

5.4.1 Numerical Behaviour

We want to add one more time dependent parameter to check wether system remain stable as before. When analyzing and comparing the behaviour of the Monte Carlo simulation and RK4 in figure (10), one can hardly find any differences between these. This means that we are very close to the expected result. With standard deviation relatively small $\sigma_R = 0.008605$, $\sigma_I = 0.0081643$ and $\sigma_R = 0.0064243$, it is fair to say that Monte Carlo simulation for this scenario is pretty accurate.

5.4.2 Theoretical view of time dependent and seasonal variation

The thought of choosing a vaccine defined as

$$f(t) = f_0 \ln(1 + t) + f_0 \tag{39}$$

Was that after some time t the number of infected would go down to zero as more people get immune to disease introduced in previous section. This behaviour where not produced as the number of infected peaks at ≈ 204 at $t \approx 0.8$, but decreases and osillates around ≈ 100. The reason behind this behaviour is that the vaccine term in the differential equationa and in the probability transission is wrong. Since the function above is defined to be slowly increasing, the number of infected should also slowly decrease eventhough we have an time dependent $a(t)$ which represent the rate of how fast the disease spreads. A suggestion would be to have fR rather than f.

6 Conclusion

The main goal of this numerical study was to compare and check if Monte Carlo was stable enough to produce similar result as Runge Kutta 4 which solved SIRS directly. As we found through this study, the result produced by Monte Carlo for different SIRS scenarios was quite accurate with very small diviations when comparing with result from Runga Kutta method. With low standard diviation we can expect that we are really close to the expected result. Remember that this is for cases for a system with a low population. Unfortunately we did not manage to test algorithm with a larger population size and extreme cases for the parameters to check if the result ramain stable. Intuitively for the case of larger population size, we belive that system will still remain stable and produce accurate results we are looking at the transition one person moving to another group. Since the transission is heavily dependent on the choice of Δt and is defined to be $\Delta = \frac{1}{\alpha N}$ where α is one of the parameters defined in this study. Increasing N will not harm the transition from one group to another since it Δt adjust itself with respect to the population change. This can be done as future work to check wether if this is true or not. For the cases with extreme parameters it is hard to say if the system will stable or not.

The second goal of this numerical study was to get a better understanding of the SIRS model worked. We found out in order to have a growth in the population the term eN has to dominate. Comparing the value of e, d and d_i, the value of e has to be slightly larger than d_i and d in order to get this behaviour. Meaning that the term eN will always dominate if birth rate is slightly larger than the death rate.

7 Future work

For this numerical study there is alot that can be done for future work. One of the goal are to paralyze and make the C++ program more general such it can be applied to any Compartment disease models. Further we could increase the number of samplings to check if RK4 and Monte Carlo overlap each other. As for the study of SIRS model we could try more parameters to check extreme cases, such as introducing a deadly disease and antidote for that disease to check the behaviour and how the population changes. An another interesting case would be to increase the total pouplation N to see if RK4 and Monte Carlo still agree with each other.

References

[Hjorth-Jensen, 2018] Hjorth-Jensen, M. (2018). Project 5, disease modeling.

[Hjorth-Jensen, 2015] Hjorth-Jensen, M. (Fall 2015). Computational physics fys3150/fys4150, lecture notes.

[Jianjun Wu and Sun, 2004] Jianjun Wu, Z. G. and Sun, H. (2004). Simulation of traffic congestion with sir model. *Modern Physics Letters B*, Vol. 18, No. 30, pp. 1537-1542.

[Mørken, 2017] Mørken, K. (2017). Numerical algorithms and digital representation.