ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Lecture with Computer Exercises: Modelling and Simulating Social Systems with MATLAB

Project Report

Disease spreading

Nina Verstraete & Bokyung Kim

Zürich

May 2010

Eigenständigkeitserklärung

Hiermit erkläre ich, dass ich diese Gruppenarbeit selbständig verfasst habe, keine anderen als die angegebenen Quellen-Hilfsmittel verwenden habe, und alle Stellen, die wörtlich oder sinngemäß aus veröffentlichen Schriften entnommen wurden, als solche kenntlich gemacht habe. Darüber hinaus erkläre ich, dass diese Gruppenarbeit nicht, auch nicht auszugsweise, bereits für andere Prüfung ausgefertigt wurde.

Nina Verstraete Bokyung Kim

Agreement for free-download

We hereby agree to make our source code of this project freely available for download from the web pages of the SOMS chair. Furthermore, we assure that all source code is written by ourselves and is not violating any copyright restrictions.

Nina Verstraete Bokyung Kim

Table of content

1. Individual Contribution

The 2D model started from one of the exercises from this course. Bokyung worked at the life-long immunity model and Nina worked at the re-infection model. Otherwise, most works are equally distributed.

2. Introduction and Motivation

It is estimated that the Black Death, caused by a bacterium *Yersinia pestis*, killed about 30% of Europe's population in the 14th century [1]. In 1918 an influenza strain, called the Spanish flu, killed about 20-40 million people worldwide [2]. Nowadays, seasonal epidemics of influenza kill about 250'000 – 500'000 people every year [3]. Some infectious diseases, such as viruses causing a 'common cold', spread with a low death rate, but have a high ability for infection.

The infection rate and death rate of a certain disease are not the only factors that will determine the spread, and whether an epidemic or pandemic arises. People can recover with long lasting immunity depending on the pathogen, or immunity can induced by vaccination. Different diseases will spread with a unique pattern which is determined by above and other factors.

Modeling the spread of diseases, taking into account variable factors, will help us to understand and predict the evolution of the epidemic spread.

Predicting how a disease will spread could help us organize the community and prepare the health care sector for the amount and nature of the casualties they will get.

Modeling disease spreading could also help us understand what the best methods for intervention are. Different vaccination strategies can be modeled and compared.

The models in this paper are based on the SIR model. This model divides the population in 3 groups, relevant to an infectious disease: Susceptible, Infected and Recovered (immuned) people.

We tried to extend this model by adding an additional 'dead' compartment and including possibilities such as re-infection and vaccination.

The paper is divided in 2 sections: The 1D section models the amount of people in each compartment according to time, while the 2D model includes the spatial dimensions.

3. Description of the model

3. 1. 1D model

The basic idea of SIR model is like this [4,5].

[Figure 1] Basic SIR model

The 'S' represents the number of susceptible people, 'I' is used to describe the amount of infected people and 'R' represents the recovered population. These numbers change depending on the infection rate (beta) and recovery rate (gamma). The following equations were used in the SIR model:

$$\frac{dS}{dt} = -\beta IS$$

$$\frac{dI}{dt} = \beta IS - \gamma I$$

$$\frac{dR}{dt} = \gamma I$$

$$\frac{dS}{dt} + \frac{dI}{dt} + \frac{dR}{dt} = 0$$

$$S(t) + I(t) + R(t) = N = \text{constant}$$

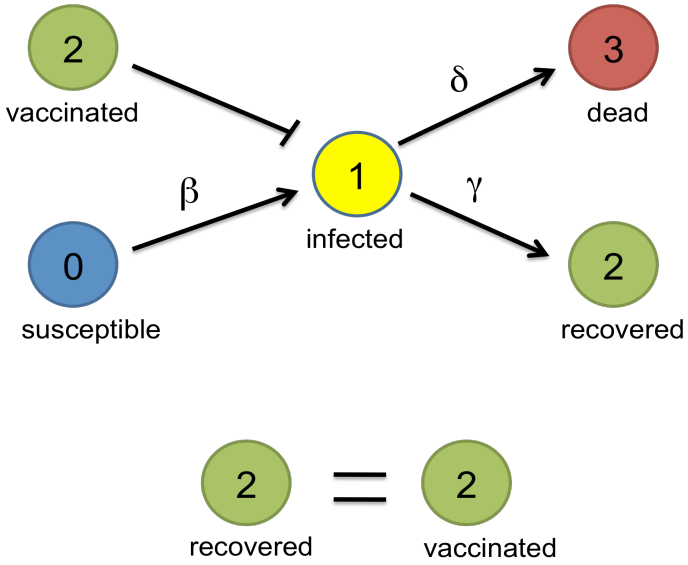
We have modified this basic SIR model by the following assumptions:

1. People die and people are born. To simplify the model, the population is assumed to be constant, which means that the number of new born and deceased people is the same.
2. New born babies are vaccinated for a certain disease.
3. Not only babies but also non-newborns are vaccinated.

3.2. 2D model

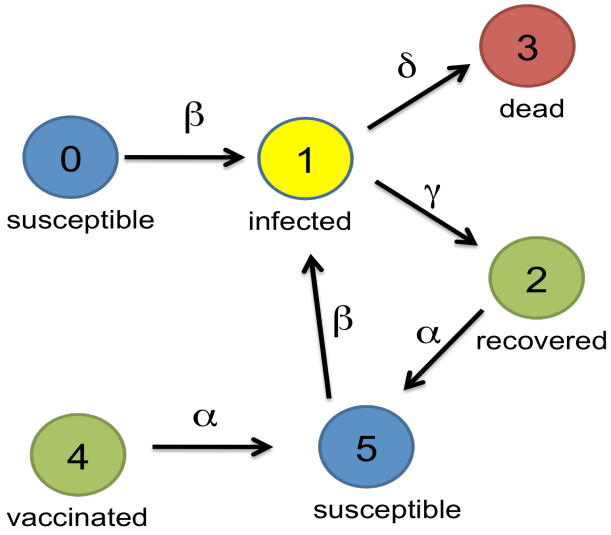
The SIR model can express only three groups; susceptible, infected and recovered. However, the situation is more complex. Some people can die from the disease and some people can get re-infected by the same disease. When getting infected by a certain pathogen for the first time, people can built up an immunity that prevents them from getting the infected a second time. Also, a vaccine can be given before people get infected. For some diseases, vaccination is not sufficient to prevent infection. For example, people get infected by the influenza virus every year since it has a high mutation rate. Even when some people were infected in the past, they can get the infection again.
We want to create a more plausible model compared to basic SIR model.

For the life-long immunity model, such as measles or chickenpox, vaccination has almost the same effect as when someone would be recovered from the disease. Once immunity for a certain disease is established, she/he will never get the disease again. In the model this means that recovered or vaccinated people never go to the infected state. To simulate this model, we need only 4 groups; susceptible (0), infected (1), recovered (2) and dead (3). The vaccinated group is included within the recovered group.



[Figure 2] Long-lasting immunity

However, this is not always the case. Sometimes, people can get infected again by the same pathogen. In this model, people can get infected even when they were vaccinated before.



[Figure 3] Re-infection

When α equals 0, this model is the same as the previous one. Depending on a 're-infection rate α' , the simulation shows a different pattern. In order to express the

population that becomes susceptible again, we added one more group compared to previous model.

4. Implementation

4. 1. 1D model

4.1.1. Constant Population

This model considers the dead and new born people, but the death rate and birth rate remain the same for simplicity. We put in a new variable μ for expressing the birth and death rate.

$S(t)+I(t)+R(t)=N=\text{constant}$

$dSdt=\mu N-\mu S-\beta SN$

$dIdt=\beta ISN-(\gamma+\mu)I$

$dRdt=\gamma I-\mu R$

In a basic SIR model, the most important variable is the basic reproduction number, which is the ratio of infection rate and recovery rate, $\beta\gamma$. However, this model also depends on the death and birth rate, μ. In this case, the basic reproduction rate is $\beta\mu+\gamma$.

4.1.2. Vaccination of Newborns

For this model, we have put in the new variable V that represents the vaccinated population. In this case, we consider that vaccination provides life-long immunity. The variable P stands for the vaccination rate.

$dSdt=\mu N(1-P)-\mu S-\beta ISN$

$dIdt=\beta ISN-(\gamma+\mu)I$

$$\frac{dV}{dt} = \mu NP - \mu V$$

4.1.3. Vaccination of Non-newborns

This model is almost similar as the previous one, except that the vaccine is given not only to newborns but also to non-newborns. The vaccination rate for non-newborns is given by ρ.

$$\frac{dS}{dt} = \mu N1 - P - \mu S - \rho S - \beta ISN$$
$$\frac{dI}{dt} = \beta ISN - (\gamma + \mu)I$$
$$\frac{dV}{dt} = \mu NP + \rho S - \mu V$$

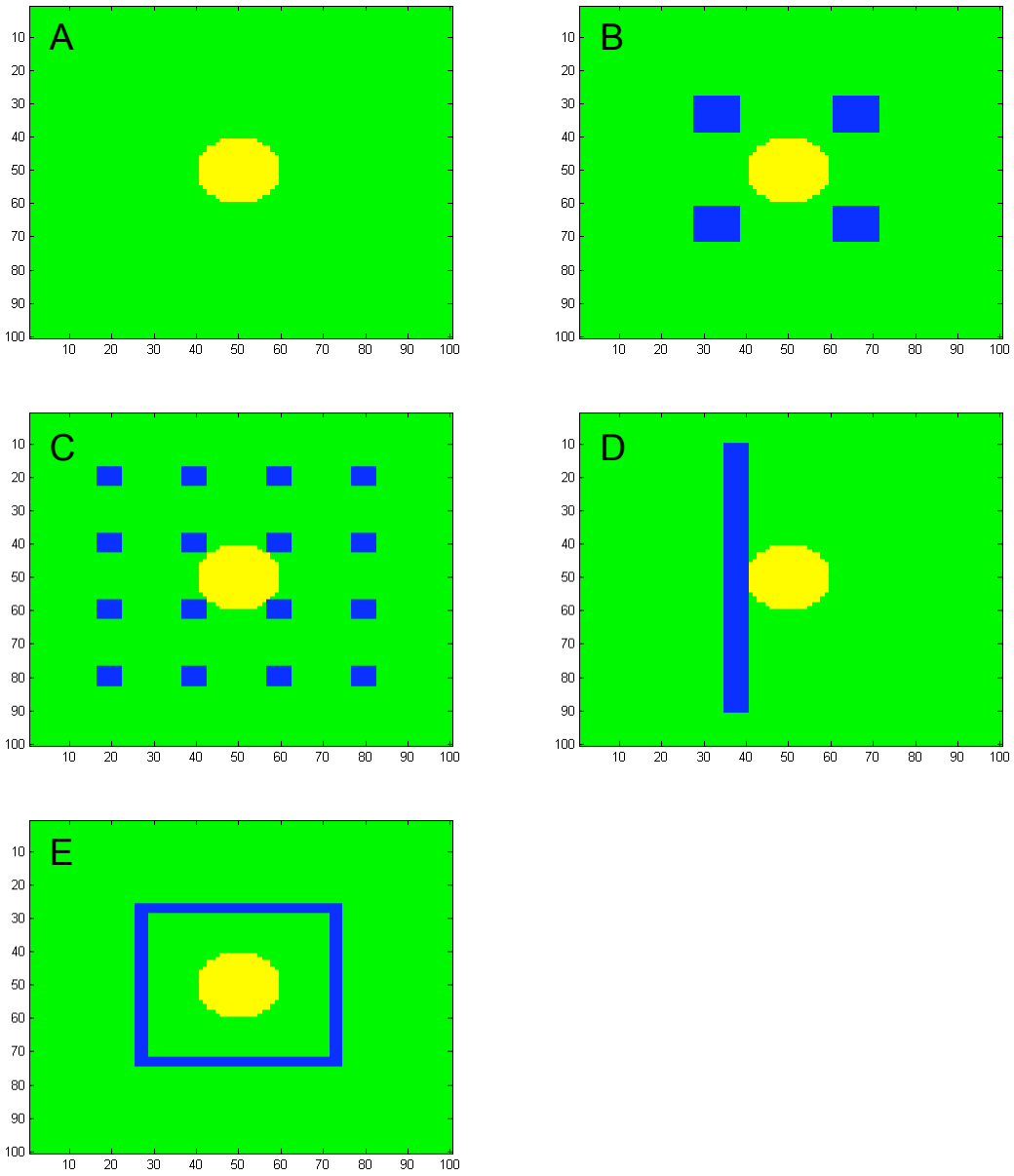
In order to easily compare the disease spreading pattern depending on the variables, we made function, called **diseasespreading1D**. The input variables for this function are infection rate (beta), recovery rate (gamma), death and birth rate (mu), new born vaccination rate (p) and non-newborn vaccination rate (rho). Also, the user can choose the pattern: the basic SIR model (basic), constant population model (constant), new born vaccination model (newborn) and non-newborn vaccination model (non-newborn). The time step for this simulation is 0.01 and it keeps continuing until 20,000 steps. The population size is 1000. At the initial point, we assumed that 80% of the population is susceptible and 20% is infected.
The changes of suspected, infected and recovered (or vaccinated) population are shown in the same graph. To distinguish the different populations, colours were used: susceptible population is black, infected is red and recovered is blue. The time is expressed on the x axis, going from 0 to 200 and the population on Y axis, going from 0 to 1000.

4. 2. 2D model

4.2.1. One-end infection

This model assumes one-end infection. Once people acquire immunity for a certain disease, they will never get the same disease again. For example, if people get measles when they are young, they will never get infected by the virus again. We simulated this idea in Matlab with a function, called **disease spreading2D_MN**.

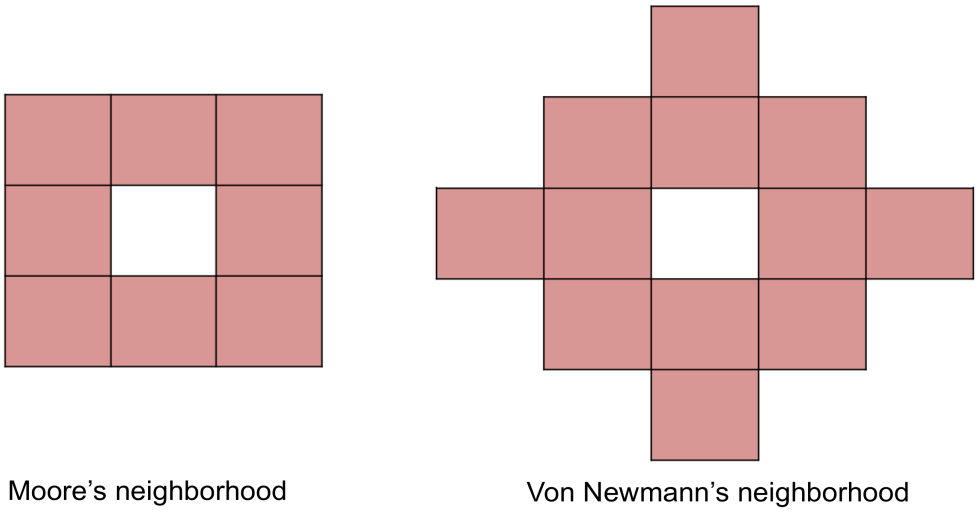
In this function, disease spreading requires infection rate (beta), recovery rate (gamma), death rate (delta), spreading pattern (neigh) and vaccination pattern (shape).
The size of the population is estimated 100x100 in this model. At the beginning, everyone is susceptible (number=0).
We assumed 4 vaccination patterns and a non-vaccinated state. The amount of vaccinated people is set up to 4 % of the population, being 400 people. The vaccinated state is set only once before the simulation and not repeatedly given.
Since this model assumes that people can acquire 'perfect immunity' after vaccination, a vaccinated person never gets infected. As a result, this vaccinated person can act as a barrier to prevent spreading of the disease. With 4 spots, spatter, rectangle and closed rectangle, we investigated whether the spreading pattern was related to the vaccination pattern.



[Figure 4] Different vaccination patterns (A: no vaccination, B: 4 spots, C: spatter, D: rectangle, E: closed rectangle)

We used different colours to represent the different states of people. As shown in the above figures, blue represents the vaccinated people, green healthy people and yellow circle stands for the infected people at time point 0. We assumed that infection starts from the centre of the grid.

To express the disease spreading, we used two models: Moore's neighbourhood and Von Neumann's neighbourhood. Moore's model considers 8 surrounding cells as his neighbourhood. Von Neumann's model considers 12 surrounding cells as his neighbourhood.



[Figure 5] Moore and Von Neumann neighbourhood

The simulation continues until no more people are infected. The number of susceptible, dead and immunized people at the end of the simulation is counted.

To determine whether a certain person gets infected or recovered at a certain time point, a random number is compared with the infection and recovery rate in the Matlab code. This can cause variability in results for same simulations. To consider this inevitable difference, each simulation was run 10 times and an average value was taken as a final result.

To repeat the same simulation, we created a function called '**counterpopulation'**. The input variables for this function are infection rate (beta), recovery rate (gamma) and death rate (delta). The user has to define vaccination pattern (shape), spreading pattern (pattern) and how many times to repeat the same simulation. The shape can be chosen: no vaccination (none), 4 spots vaccination (4spots), 16 small spots vaccination (spatter), one

long vaccination pattern in rectangular shape (rectangle) or isolated area vaccination (closedrectangle)

This function creates an array of each population and calculates the average and standard deviation of repeated simulations. A graph, including all information, appears after all simulations are ended.

4.2.2. Re-infection

In this model, people can get infected more than once. An individual can go from a recovered state back to a susceptible state with probability alpha. Re-infection can happen with viruses that cause a 'common cold'. Note that in this model a vaccination is used, while no vaccination for a common cold exists.

To assess how many people got infected how many times at the end of the simulation, a counting system was developed. Hereby a y matrix was created that added 1 to the value each time a person changed compartments. All people started with value 0. When going to infection this value was increased with 1, subsequently going to recovery another value 1 was added in the y matrix. The same procedure was followed for going from x value 5 (susceptible again) to x value 1 (infected). When dying, the y value in the matrix was converted to value -1. At the end of the simulation, the amount of y values 0 (never infected), 1 (infected once), 2 (recovered once), 3 (susceptible again), 4 (infected twice), 5 (recovered twice) and -1 (died) was counted.

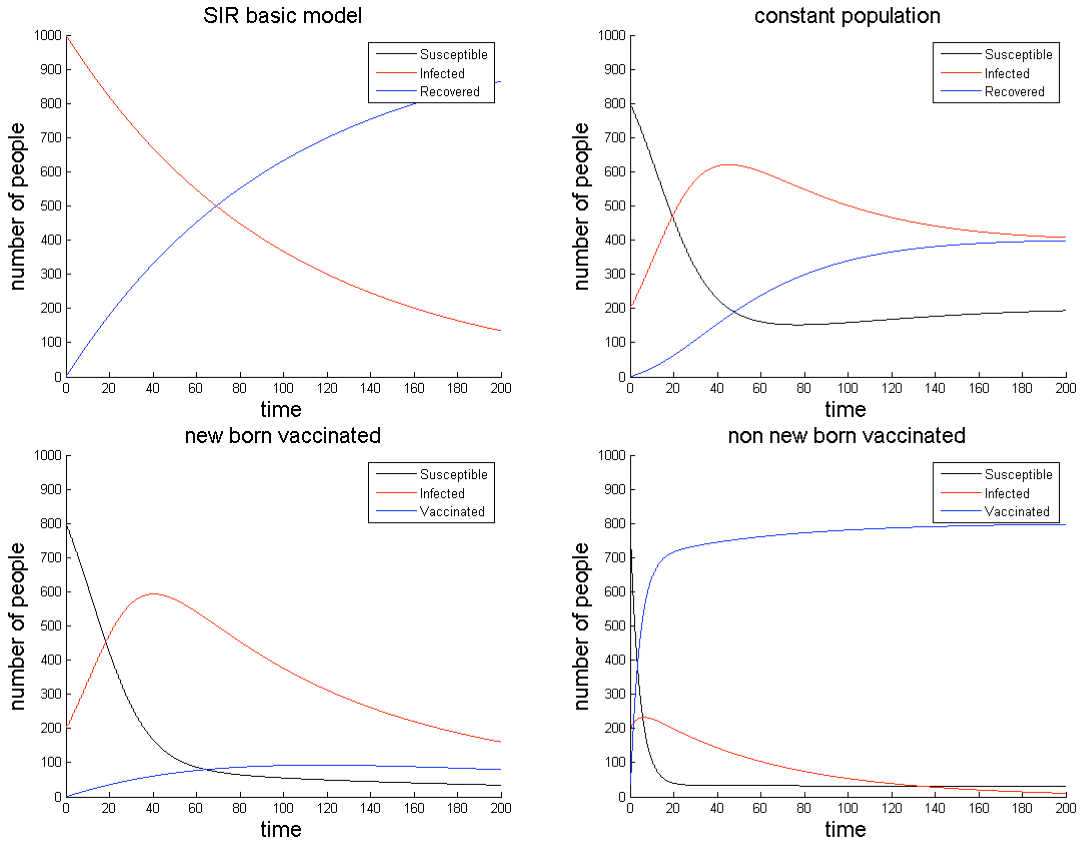
5. Simulation Results and Discussion

5. 1. 1D model

5.1.1. Infection rate > recovery rate

These simulations were done with the following values:

infection rate 0.1; recovery rate 0.01; death and birth rate 0.01, vaccination rate of newborn 0.2; vaccination rate of non newborn 0.2.



[Figure 6] High infection rate

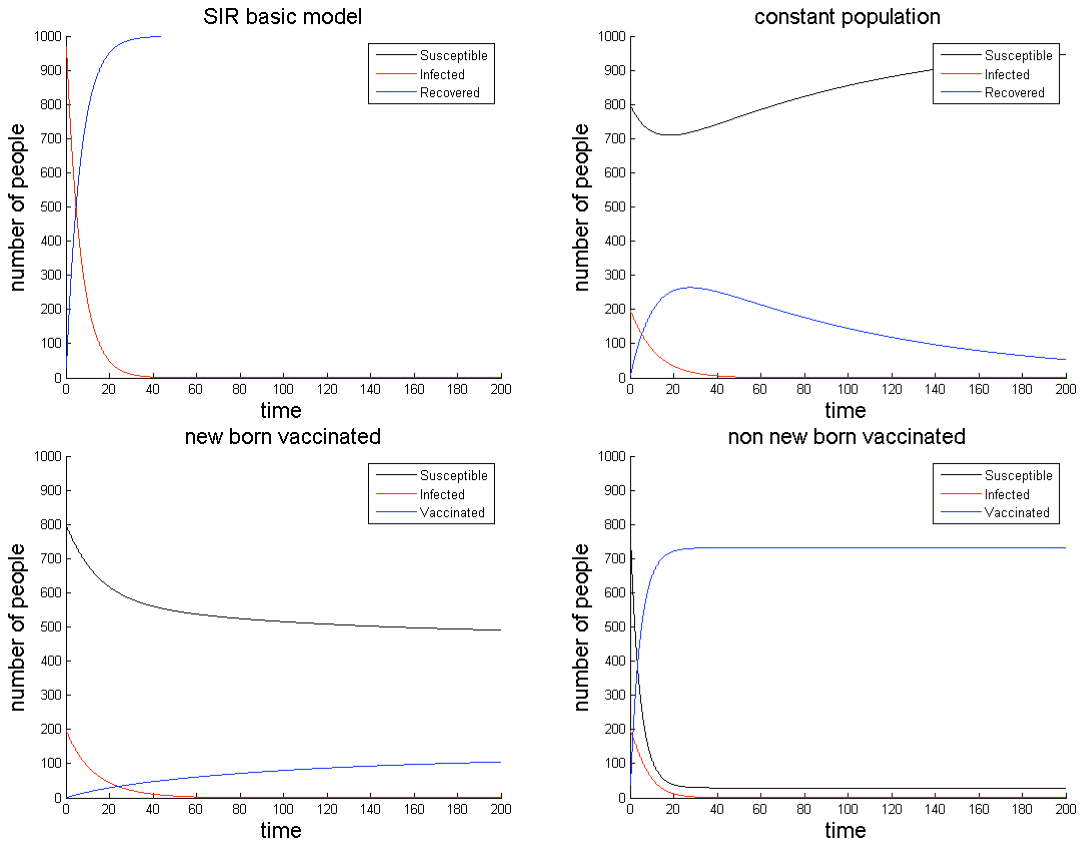
The basic reproduction number R is larger than 1, which means the infection will be able to spread in the population.

When comparing the amount of infections between the normal ‘constant population’ and the ‘new born vaccinated population’, we can see that the decrease in the amount of infected people is much more pronounced in the vaccinated population.

Increasing the vaccination to the ‘non newborn population’ as well dramatically reduces the amount of infections and the amount of susceptible people.

5.1.2. Recovery rate > infection rate

These simulations were done with the following values:
Infection rate 0.1; recovery rate 0.15; death and birth rate 0.01; vaccination rate of newborn 0.2; vaccination rate of non newborn 0.2.



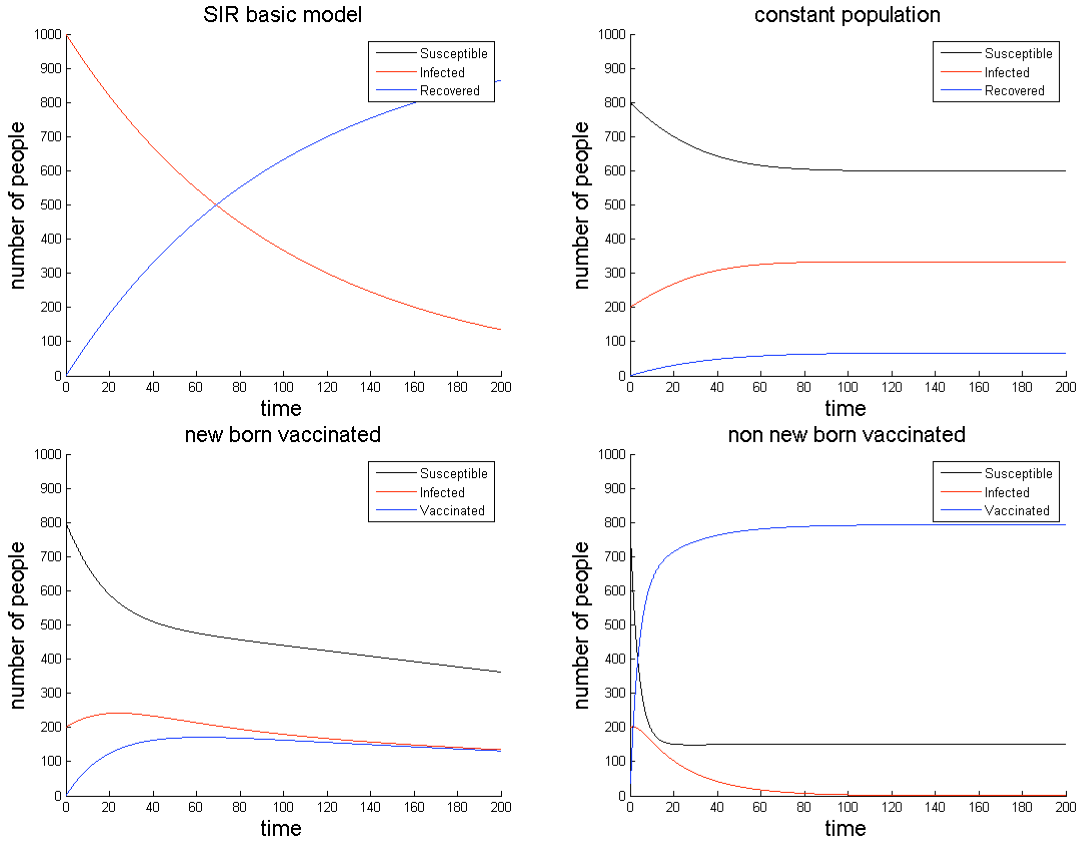
[Figure 7] High recovery rate

The basic reproduction number R is smaller than 1, which means that the infection will die out.
Looking at the 3 graphs ‘constant population’, ‘new born vaccinated’ and ‘non new born vaccinated’, there can be seen that because of the higher recovery rate, the amount of infections decreases to 0 rapidly. In the ‘constant population’, the amount of susceptible people increases over time. In the ‘new born vaccinated’ population, the amount of susceptible people decreases slowly, while in the ‘non new born vaccinated’ population,

the curve of susceptible people decreases very steeply. This is because of the very sharp increase of vaccinated people in time point 0-20. Note that this is depending on the non newborn vaccination rate (figures not shown).

5.1.3. High death & birth rate

These simulations were done with the following values:
infection rate 0.1; recovery rate 0.01; death and birth rate: 0.5; vaccination rate of newborn 0.2; vaccination rate of non newborn 0.2.



[Figure 8] High death and birth rate

The basic reproduction number R is smaller than 1, which means that the infection will die out.

Because of the high turnover of the population, the amount of susceptible, hence infected people goes to a constant level more quickly in the ‘constant population’. When comparing the ‘newborn vaccinated’ graph in this section with the graph of the same population in the ‘Infection rate > recovery rate’ section, we observe less infections when we have a high birth rate. This is because more children are born, consequently more people are vaccinated.

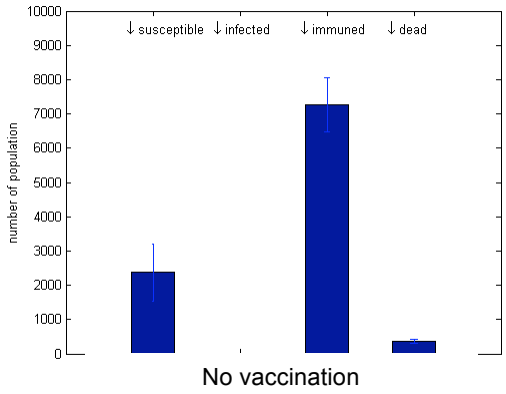
5. 2. 2D model

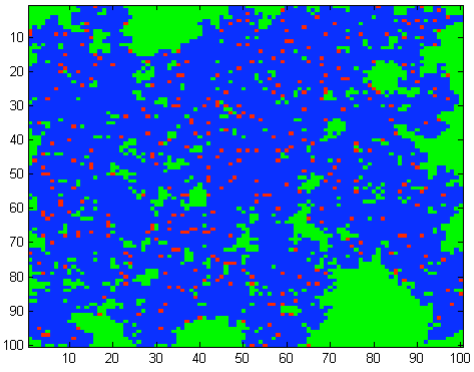
5.2.1. One-end infection

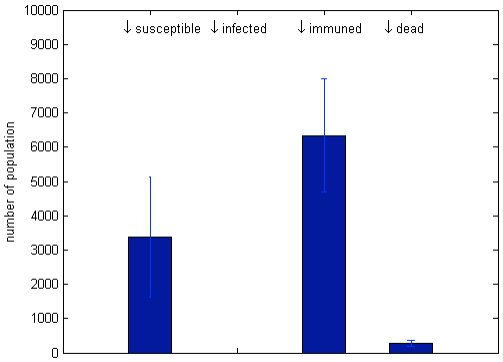
Case1. Slow infection

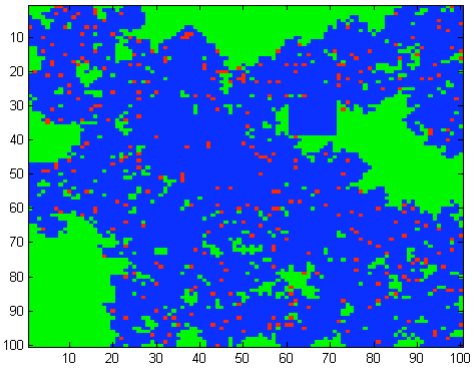
These simulations were done with the following values:

infection rate 0.01; recovery rate 0.02; death rate 0.001. To remove variation due to randomness, we repeated each simulation 10 times.

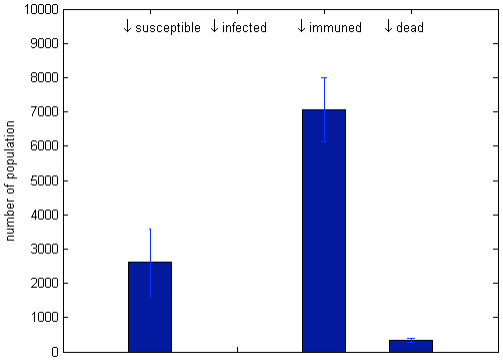


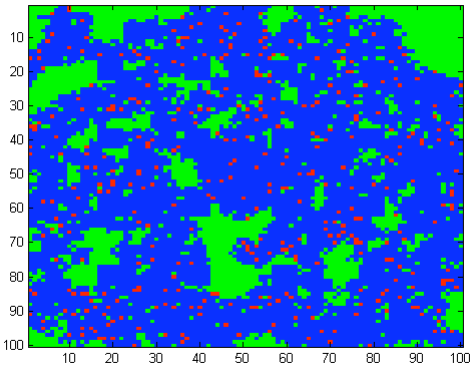




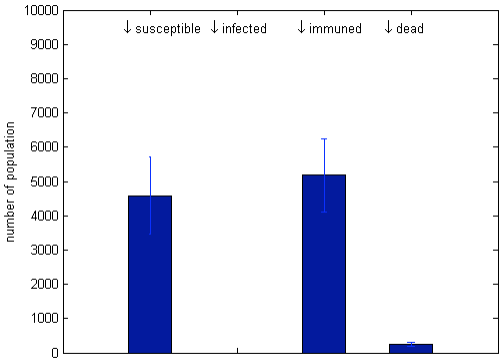


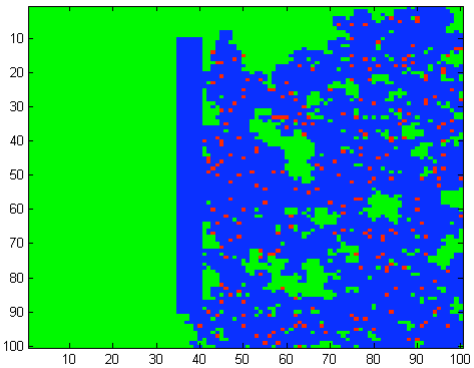
4 Spots vaccination



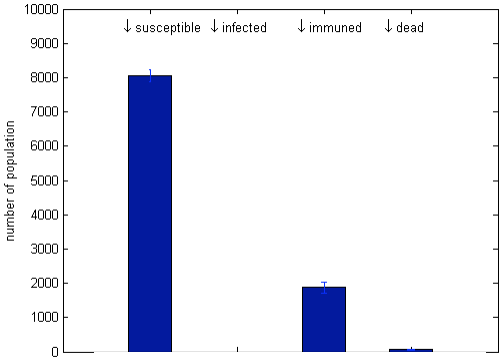


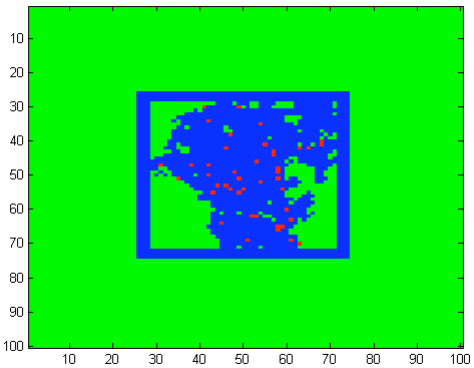
Spatter area vaccination





Rectangular vaccination





Closed area vaccination

[Figure 9] Spreading pattern depending on the vaccination area at low infection

We set 0.4% vaccination (400 people were vaccinated among 10000 people) in each case. Spattered or 4 spots vaccination was less helpful than rectangular or closed area vaccination to block the disease spreading. We compared the amount of susceptible, recovered (immunized) and dead people for each vaccination pattern.

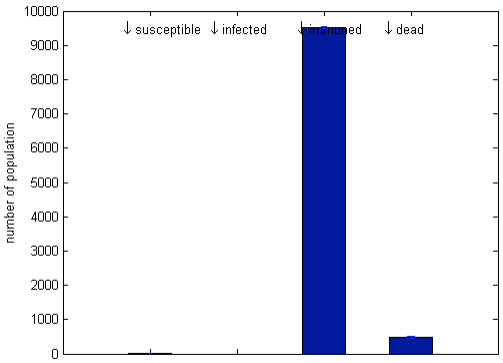
[Figure 10] Results depending on vaccination pattern, low infection rate

As seen in the graph, the change in dead population depends on the vaccination pattern. Each vaccination strategy decreased dead population compared to no vaccination. The most effective strategy is closed rectangle. However, this an idealized pattern designed so that disease cannot spread out of the closed area. Comparing the two spatter patterns, we can see that the small 16 square vaccination is more effective than the big 4 square vaccination.

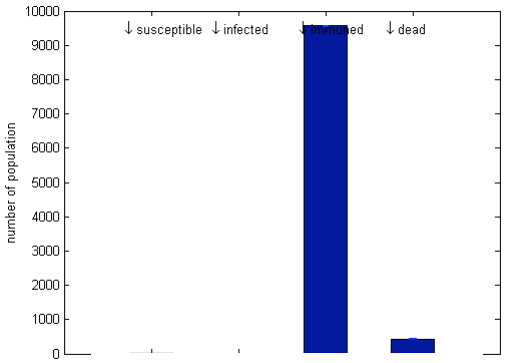
When there is no vaccination, only 20% of the population remains susceptible and 80% gets infected. In contrast to this, 80% never gets the disease in the closed vaccination pattern. Based on this, we can say that vaccination is helpful when there is a slow infection rate.

Case2. Fast infection

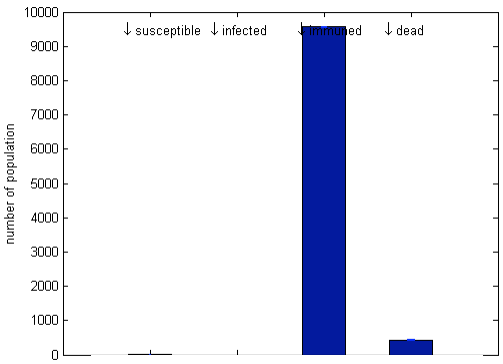
Vaccination is very helpful to block the disease spreading even when only a low percentage (0.4%) of the population is vaccinated. However, vaccination is not helpful in all cases. We increased infection rate 10 times to compare with low infection rates. These simulations were done with the following values: infection rate 0.1; recovery rate and death rate are the same as in the previous case.



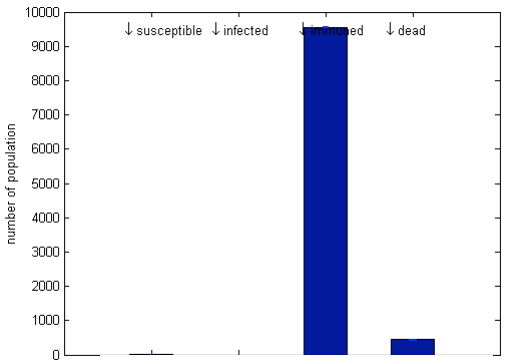
No vaccination



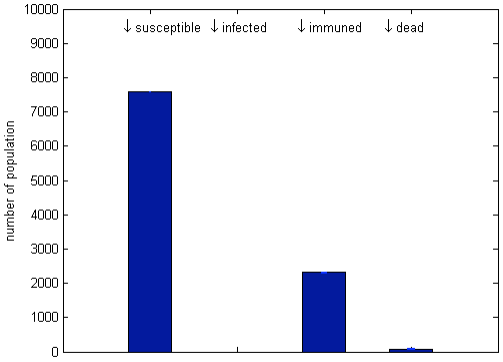
4 Spots vaccination



Spatter area vaccination



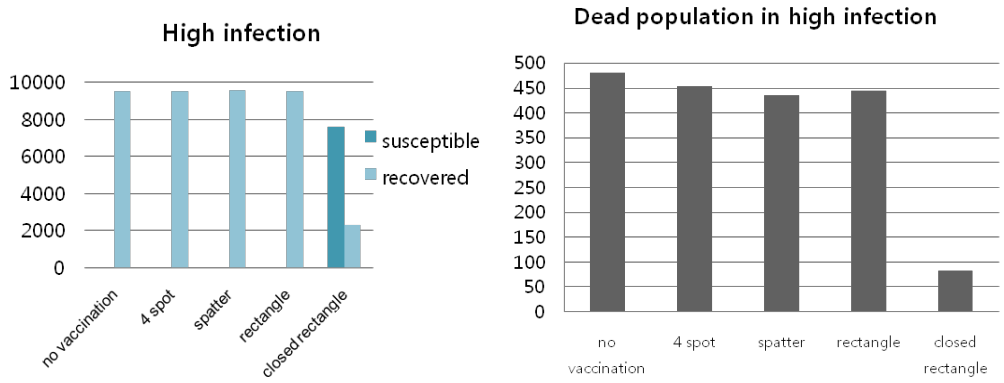
Rectangular vaccination



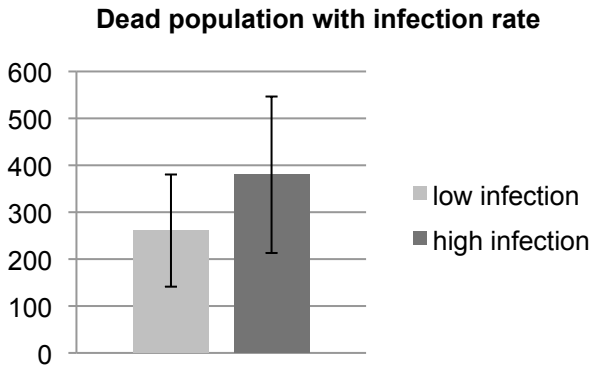
Closed area vaccination

[Figure 11] Spreading pattern depending on the vaccination area at high infection

The result is dramatically different compared to low infection. In high infection, vaccination is not helpful to prevent disease spreading. Everyone gets infected except in the closed rectangular vaccination. However, as said before, closed rectangle vaccination strategy is a too much idealized case, which can not reflect real vaccination possibilities in a community. Except for this case, the change in vaccination pattern does not strongly influence on disease spreading.



[Figure 12] Results depending on vaccination pattern, high infection rate

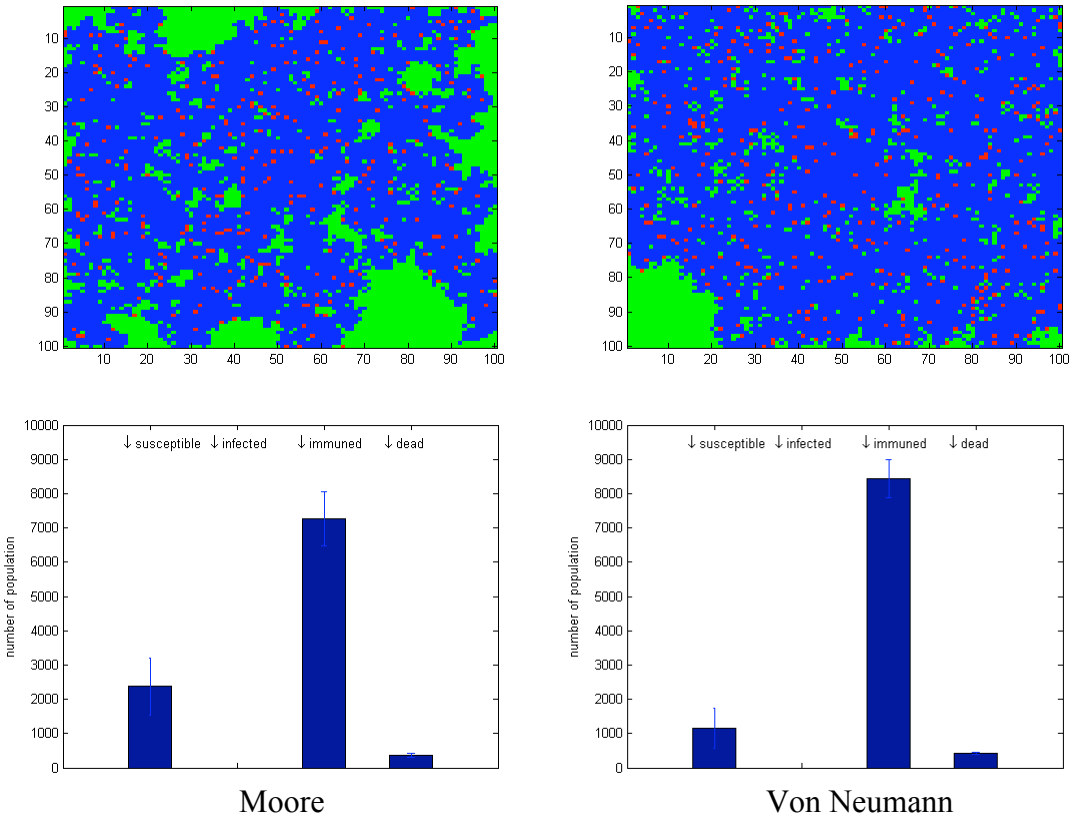


[Figure 13] Compare dead population due to infection rate

As a result, the dead population increased about 1.5 times in the high infection rate simulations, even when the recovery rate and death rate remain the same.

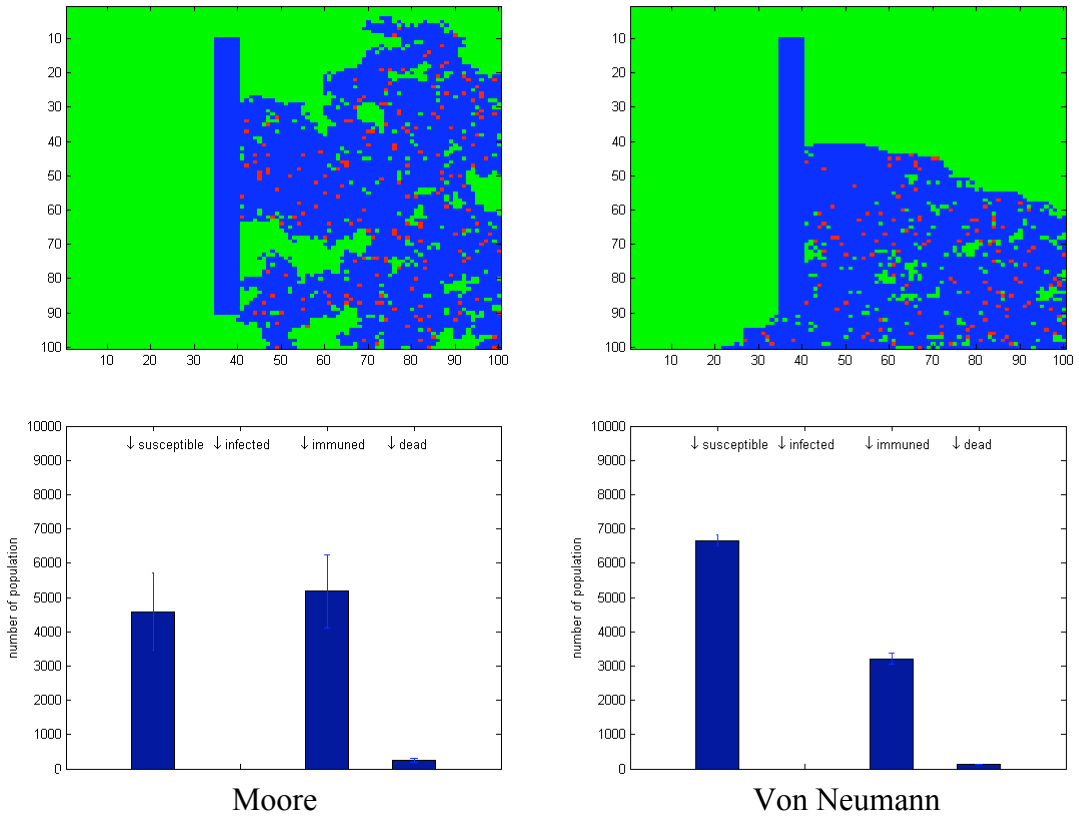
Case3. Different spreading pattern

We compared the Moore and Von Neumann neighbourhood in both the no vaccination case and the rectangle vaccination case. These simulations were done with the following values: infection rate 0.01; recovery rate 0.02; death rate 0.001 (same as in the first case). To remove variation due to randomness, the simulations were repeated 10 times.



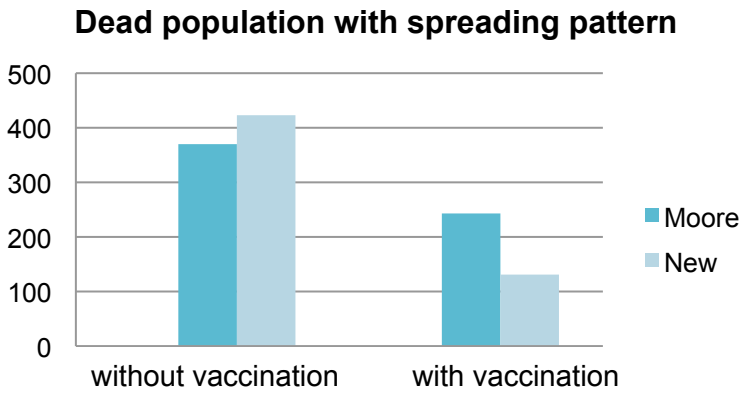
[Figure 14] Moore and Von Neumann neighbourhood (no vaccination)

In Moore's neighbourhood simulation, the amount of never infected people is 2 times higher than in the Von Neumann's neighbourhood simulation. As seen in the figure 14, the spreading pattern is denser in the Von Neumann simulation compared to the Moore simulation. As a result, more people died in the Von Neumann neighbourhood simulation. Since the Von Neumann neighbourhood has 12 surrounding cells, the infection spreads out faster. We already saw a similar phenomenon in figure 13.



[Figure 15] Moore and Von Neumann neighbourhood (0.4% vaccination)

However, the spreading pattern changed with vaccination. In the Moore's neighbourhood case, more than half of the people were infected and immunized. Compared to this, 65% people never got infected in Von Neumann neighbourhood case. Therefore, the death rate is lower in the Von Neumann case.

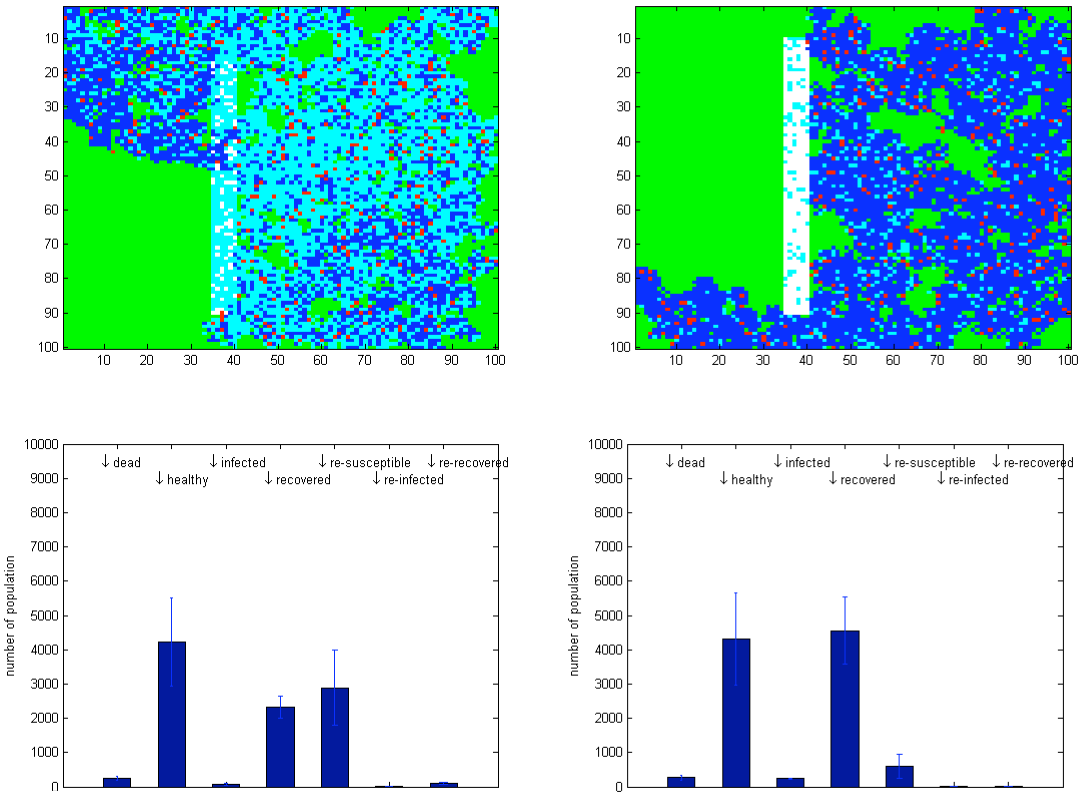


[Figure 16] Dead population depending on spreading pattern

Remarkable, the 2 patterns led to different results with or without vaccination. The spreading pattern in the Von Neumann neighbourhood was denser, but was blocked easier when there was a barrier.

5.2.2. Re-infection

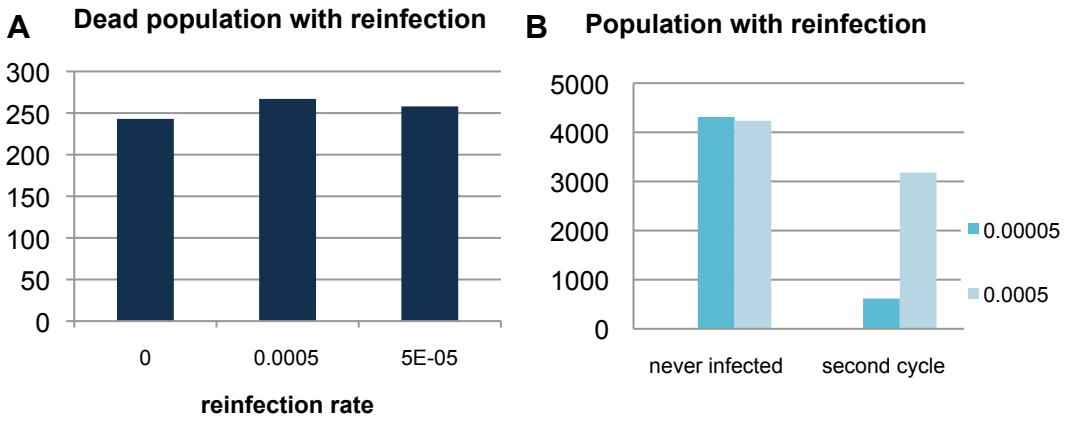
People can get infected by certain diseases even after being immunized. We simulated this kind of disease spreading, with high re-infection rate (alpha=0.0005) and low re-infection rate (alpha=0.00005), when there is a rectangle vaccination present. The other conditions are the same as in the first simulations of low infection.



High re-infection (α=0.0005)　　low re-infection (α=0.00005)

[Figure 17] Re-infection model

Even if the re-infection rate is 10 times higher, the dead population and the never infected population does not change much compared to low re-infection. There is, however, a 3 fold increase of people that go into a second infection cycle.



[Figure 18] Change of population in re-infection model

When we set the death rate quite low (0.01%), the dead population and the re-infection are not strongly related. This means, even when people get infected again, there is still a low change to die. However, there is a dramatic change in the number of re-infected people as seen in Figure 18B. 30% of the population becomes re-susceptible, re-infected and recovered again. Needless to say, if this is a disease with a high death rate, 30% of the population would be in danger again.

5.2.3. Discussion points

1) In the 2D model, no migration is considered. Every person is represented as a dot that does not change position. In today's world, people are very dynamic. It can be questioned if the vaccination models would still hold true if a dynamic model is used.
2) Instead of giving a vaccine to a selected number of people once (in the beginning of the simulation) vaccines can be given in a repeated (pulsed) manner. This

would mean that a vaccine is given to a number of (different) people at several time points during the simulation [6].

6. Summary and Outlook

The SIR model is commonly used to model the spread of diseases. In this report we extended the SIR model in both a time dependent and a spatial approach. In the 1D (time) model, we included possibilities such as vaccination of both newborns and non-newborns while considering a dynamic population (population with birth and death rate). Simulations are made with high infection rates, high recovery rates and high birth and death rates. In the 2D (spatial) model, a comparison is made between different vaccination strategies. The value of the vaccination plans is assessed by comparing the amount of never infected (susceptible) people and the total amount of dead people. Also re-infection is considered in the spatial model.

Although the simulations can be a model for a certain diseases, they do not reflect the spreading in reality. It could be interesting to get some more realistic simulations by inserting 'real' values for beta, gamma, delta,.. in the mathematical models, instead of hypothetical values. Since different diseases spread with different patterns, these numbers should not be the same for every spreading pathogen. Inserting these numbers depending on the disease will lead to a more accurate prediction of the spreading, which could lead to better planning for intervention.

7. Reference

[1] Haddock D, Kiesling L. The Black Death and Property Rights. *Journal of Legal Studies*. VOL XXXI (2002).

[2] Mills C, Robins J, Lipsitch M. Transmissibility of 1918 pandemic influenza. *Nature* 432(7019), 904-906 (2004).

[3] http://www.who.int/mediacentre/factsheets/fs211/en/index.html

[4] Kermach W, Mckendrick A. A contribution to the mathematical theory of epidemics. *Proc. R. Soc. Lond. A* 115, 700-721 (1927).

[5] http://en.wikipedia.org/wiki/SIR_Model

[6] Sulgin B, Stone L, Agur Z. Pulse vaccination strategy in the SIR epidemic model. *Bulletin of mathematical biology* 60, 1123 1148 (1998).

8. Appendix

1. 1D spreading

```
function diseasespreading1D(beta,gamma,mu,p,rho,pattern)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% function diseasespreading1D returns a graph about disease
% spreading pattern depending on time
%
% input variables:
% 1. beta:infection rate
% 2. gamma: recovery rate
% 3. mu: birth rate= death rate
% 4. p:vaccinated rate of new born
% 5. rho:vaccinated rate of non-new born
% 6. pattern: can be chosen between 'basic','constant','newborn' and
% 'nonnewborn'.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%clear all;
dt=0.01; %time step
S(1)=800; % susceptible people
I(1)=200; % infected people
R(1)=0;  % recovered people
V(1)=0; % vaccinated people
t(1)=0; %starting point
N(1)=S(1)+I(1)+R(1); % the population is constant
hold on;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if strcmp('basic',pattern)
   for i=1:20000
   dS=-beta*I(i)*S(i);
```

```
    dI=beta*I(i)*S(i)-gamma*I(i);
    dR=gamma*I(i);

    S(i+1)=S(i)+dt*dS;
    I(i+1)=I(i)+dt*dI;
    R(i+1)=R(i)+dt*dR;
    t(i+1)=t(i)+dt;
    end

    plot(t,S,'k')
    plot(t,I, 'r')
    plot(t,R,'b')
    xlabel('time','FontSize',16)
    ylabel('number of people','FontSize',16)
    legend('Susceptible','Infected','Recovered')
    title('SIR basic model','FontSize',16)
    axis([0 200 0 1000])
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if strcmp('constant',pattern)
% constant population
    for i=1:20000
        dS=mu*N(i)-mu*S(i)-beta*I(i)*S(i)/N(i);
        dI=beta*I(i)*S(i)/N(i)-(gamma+mu)*I(i);
        dR=gamma*I(i)-mu*R(i);
        S(i+1)=S(i)+dt*dS;
        I(i+1)=I(i)+dt*dI;
        R(i+1)=R(i)+dt*dR;
        N(i+1)=S(i+1)+I(i+1)+R(i+1);  % constant population
        t(i+1)=t(i)+dt;
    end

    plot(t,S,'k')
    plot(t,I,'r')
```

```
    plot(t,R,'b')
    xlabel('time','FontSize',16)
    ylabel('number of people','FontSize',16)
    legend('Susceptible','Infected','Recovered')
    title('constant population','FontSize',16)
    axis([0 200 0 1000])
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if strcmp('newborn',pattern)
% new born vaccination
    for i=1:20000
        dS=mu*N(i)*(1-p)-mu*S(i)-beta*I(i)*S(i)/N(i);
        dI=beta*I(i)*S(i)/N(i)-(gamma+mu)*I(i);
        dV=mu*p*N(i)-mu*V(i);
        S(i+1)=S(i)+dt*dS;
        I(i+1)=I(i)+dt*dI;
        V(i+1)=V(i)+dt*dV;
        N(i+1)=S(i+1)+I(i+1)+V(i+1);  % constant population
        t(i+1)=t(i)+dt;
    end

    plot(t,S,'k')
    plot(t,I,'r')
    plot(t,V,'b')
    xlabel('time','FontSize',16)
    ylabel('number of people','FontSize',16)
    legend('Susceptible','Infected','Vaccinated')
    title('new born vaccinated','FontSize',16)
    axis([0 200 0 1000])
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if strcmp('nonnewborn',pattern)
% non new born vaccination
    for i=1:20000
```

```
    dS=mu*N(i)*(1-p)-mu*S(i)-beta*I(i)*S(i)/N(i)-rho*S(i);
    dI=beta*I(i)*S(i)/N(i)-(gamma+mu)*I(i);
    dV=mu*p*N(i)-mu*V(i)+rho*S(i);
    S(i+1)=S(i)+dt*dS;
    I(i+1)=I(i)+dt*dI;
    V(i+1)=V(i)+dt*dV;
    N(i+1)=S(i+1)+I(i+1)+V(i+1);  % constant population
    t(i+1)=t(i)+dt;
  end

  plot(t,S,'k')
  plot(t,I,'r')
  plot(t,V,'b')
  xlabel('time','FontSize',16)
  ylabel('number of people','FontSize',16)
  legend('Susceptible','Infected','Vaccinated')
  title('non new born vaccinated','FontSize',16)
  axis([0 200 0 1000])
end
```

2. 2D spreading (life-long immunity)

```
function diseaseSpreading2D_MN(beta,gamma,delta,shape,neigh)

%function [susceptible,infected,immuned,dead]=diseaseSpreading2D(beta,gamma,delta,shape);
% function 'diseaseSpreading2D' returns the number of susceptible,infected, immuned and dead
population.
% This function assumes that immunity ability is quite strong.
% 1. Once vaccinated, the person never get infected.
% 2. Immuned person never get infected again.
% 3. Process keep until there is no infected person.
%------------------------------------------------------------------------
% input: 5 variances
% 1. 'beta' is a infection rate. The default is 0.01.
% 2. 'gamma' is a immunity rate. The default is 0.02.
% 3. 'delta' is the death rate. The default is 0.001.
% 4. 'shape' is the pattern of vaccination. Pattern can be chosen between
%    '4spots','rectangle','closedrectangle','spatter'and 'none'
% 5. 'neigh' is the neighborhood either 'Moore' or 'New'
%------------------------------------------------------------------------

% Simulate disease spreading on a 2D grid
%------------------------------------------------------------------------
% --------------------Putting initial values------------------------------
N=100;              % Grid size (NxN)
x = zeros(N, N);    % The grid x, is coded as:  0=susceptible,1=infected,2=immuned(including
vaccinated),3=dead

susceptible=0;  % counter for number of susceptible person
infected=0;   % counter for number of infected person
immuned=0;    % counter for number of immuned person
dead=0;       % counter for dead person
```

```
%------------------------------------------------------------------------
%------------------Setting up the vaccinated pattern---------------------

if strcmp('4spots',shape) % Case 1: give vaccine to 4 spots
x(28:38,28:38)=2;
x(28:38,61:71)=2;
x(61:71,28:38)=2;
x(61:71,61:71)=2;
end

if strcmp('rectangle',shape) % Case 2: give vaccine to certain region
x(10:90,35:40)=2;
end

if strcmp('closedrectangle',shape)% Case 3: give vaccine closed rectangular
x(26:74,26:28)=2;
x(26:74,72:74)=2;
x(26:28,26:74)=2;
x(72:74,26:74)=2;
end

if strcmp('spatter',shape) % Case 4: give vaccine to spatter area
x(17:22,17:22)=2;
x(17:22,37:42)=2;
x(17:22,57:62)=2;
x(17:22,77:82)=2;

x(37:42,17:22)=2;
x(37:42,37:42)=2;
x(37:42,57:62)=2;
x(37:42,77:82)=2;

x(57:62,17:22)=2;
x(57:62,37:42)=2;
```

```
x(57:62,57:62)=2;
x(57:62,77:82)=2;

x(77:82,17:22)=2;
x(77:82,37:42)=2;
x(77:82,57:62)=2;
x(77:82,77:82)=2;
end

if strcmp('none',shape) % Case 5: no vaccination
x=zeros(N,N);
end

%---------------------------------------------------------------------
%----------setting the infected individuals---------------------------------
%
% infection started in the center of the grid, and with a radius of 10 cells.
for i=1:N
   for j=1:N
      dx = i-N/2;
      dy = j-N/2;
      R = sqrt(dx*dx+dy*dy);
      if ( R<10 && x(i,j)~=2 ) % immuned person does not get infected
         x(i,j)=1;
      end
   end
end


if strcmp('Moore',neigh)
   neigh = [-1 -1; 0 -1; 1 -1; 1 0; 1 1; 0 1; -1 1; -1 0];
   % Define the Moore neighborhood, i.e. the 8 nearest neighbors
end
```

```
if strcmp('New',neigh)
   neigh = [-2 0; -1 1; -1 0; -1 -1; 0 2; 0 1; 0 -1; 0 -2; 1 1; 1 0;1 -1;2 0];
   % Define the Von Neumann neighborhoon
end

% Create a new figure
figure
hold on
% main loop, iterating the time variable, t
for t=1:100000
   % iterate over all cells in grid x, for index i=1..N and j=1..N
   for i=1:N
      for j=1:N
         % Iterate over the neighbors and spread the disease
         for k=1:8
            i2 = i+neigh(k, 1);
            j2 = j+neigh(k, 2);
            % Check that the cell is within the grid boundaries
            if ( i2>=1 && j2>=1 && i2<=N && j2<=N )
               % if cell is in state Susceptible and neighboring cell
               % Infected => Spread infection with probability beta
               if ( x(i,j)==0 && x(i2, j2)==1 )
                  if ( rand<beta )
                     x(i,j) = 1;
                  end
               end
            end
         end

         % If infected => Recover from disease with probability gamma
         if ( x(i,j)==1 && rand<delta)
            x(i,j) = 3; % death
         end
```

```
        if ( x(i,j)==1 && rand<gamma )
           x(i,j) = 2; % recovery
        end

     end
   end

   % Animate
   clf                          % Clear figure
   imagesc(x, [0 3])              % Display grid
   pause(0.01)                  % Pause for 0.01 s
   colormap([0 1 0; 1 1 0; 0 0 1; 1 0 0]);    % Define colors: Red, Green, Blue

   % If no more infected => Stop the simulation
   if ( sum(x==1)==0 )
      break;
   end
end

%-----------------counting each population---------------------------------

for i=1:N
   for j=1:N
        if (x(i,j)==0) susceptible=susceptible+1;
        end
        if (x(i,j)==1) infected=infected+1;
        end
        if (x(i,j)==2) immuned=immuned+1;
        end
        if (x(i,j)==3) dead=dead+1;
        end
    end
end
```

```
%----------Showing the number of each population----------

fprintf('Number of recovered/immuned are: %u\n',immuned); % directly show the number in
matlab
fprintf('Number of dead are: %u\n',dead);
fprintf('Number of susceptible are: %u\n',susceptible);
```

3. 2D spreading (repeated infection)

```
% Simulate disease spreading on a 2D grid
clear all;
% Set parameter values
N=100;              % Grid size (NxN)
beta=0.01;          % Infection rate
gamma=0.02;         % Immunity rate
delta=0.001;        % death rate
alfa=0.00005;       % susceptibility to disease

% define grid
x = zeros(N,N);    % 0=susceptible; 1=infected; 2=recoverd; 3=dead; 4= vaccinated;
5=susceptible again
y = zeros(N,N);    % The counting grid

m=1;               % counter for repetition

% Set initial values for repetition
infected_once(m)=0;
never_infected(m)=0;
number_of_death(m)=0;
recovered_once(m)=0;
infected_twice(m)=0;
susceptible_again(m)=0;
recovered_twice(m)=0;

for m=1:10
    x = zeros(N,N);
    y = zeros(N,N);
    infected_once(m)=0;
    never_infected(m)=0;
    number_of_death(m)=0;
```

```
recovered_once(m)=0;
infected_twice(m)=0;
susceptible_again(m)=0;
recovered_twice(m)=0;

% Set the initial grid, x with a circle of infected individuals in the
% center of the grid, and with a radius of 10 cells.

for i=1:N
   for j=1:N
      dx = i-N/2;
      dy = j-N/2;
      R = sqrt(dx*dx+dy*dy);
      if ( R<10 )
         x(i,j)=1;
      end
   end

x(10:90,35:40)=4; %vaccinated area

end

% Define the Moore neighborhood, i.e. the 8 nearest neighbors
neigh = [-1 -1; 0 -1; 1 -1; 1 0; 1 1; 0 1; -1 1; -1 0];

% Create a new figure
figure
hold on

% main loop, iterating the time variable, t
for t=1:100000

   % iterate over all cells in grid x, for index i=1..N and j=1..N
   for i=1:N
```

```
for j=1:N

    % Iterate over the neighbors and spread the disease
    for k=1:8
        i2 = i+neigh(k, 1);
        j2 = j+neigh(k, 2);
        % Check that the cell is within the grid boundaries
        if ( i2>=1 && j2>=1 && i2<=N && j2<=N )
            % if cell is in state Susceptible and neighboring cell infected => Spread infection
with probability beta
            if (((x(i,j)==0)||(x(i,j)==5))&& x(i2, j2)==1 )
                if ( rand<beta )
                    x(i,j) = 1;
                    y(i,j) = y(i,j)+1; %counting
                end
            end
        end
    end

    % If infected => death from disease with probability delta
    if ( x(i,j)==1 && rand<delta)
        x(i,j) = 3; % death
        y(i,j) = -1; %counting
    end
     % If infected => Recover from disease with probability gamma
    if ( x(i,j)==1 && rand<gamma )
        x(i,j) = 2; % recovery
        y(i,j) = y(i,j)+1; %counting
    end
    % If recovered => susceptible again with probability alfa
    if (((x(i,j)==2)&& rand<alfa ))
        x(i,j) = 5; % susceptible again
        y(i,j)=y(i,j)+1; %counting
    end
```

```
        % If vaccinated => susceptible again with probability alfa
        if (((x(i,j)==4))&& rand<alfa )
            x(i,j)= 5; % susceptible again
            y(i,j)=y(i,j)+3; %counting
        end
      end
    end

    % Animate
    clf                         % Clear figure
    imagesc(x, [0 5])               % Display grid
    pause(0.01)                     % Pause for 0.01 s
    colormap([0 1 0; 1 1 0; 0 0 1; 1 0 0;1 1 1;0 1 1]);    % Define colors

    % If no more infected => Stop the simulation
    if ( sum(x==1)==0 )
       break;
    end
end
%Counting every possibility
    for i=1:N
       for j=1:N
          if ((y(i,j)==0) || (x(i,j)==4))
             never_infected(m)=never_infected(m)+1;
          end
          if y(i,j)==1
             infected_once(m)=infected_once(m)+1;
          end
          if y(i,j)==2
             recovered_once(m)=recovered_once(m)+1;
          end
          if y(i,j)==3
             susceptible_again(m)=susceptible_again(m)+1;
          end
```

```
            if y(i,j)==4
                infected_twice(m)=infected_twice(m)+1;
            end
            if y(i,j)==5
                recovered_twice(m)=recovered_twice(m)+1;
            end
            if(y(i,j)==-1)
                number_of_death(m)=number_of_death(m)+1;
            end

        end
    end

end

%Making figure
figure();
Y(1)=mean(number_of_death);
Y(2)=mean(never_infected);
Y(3)=mean(infected_once);
Y(4)=mean(recovered_once);
Y(5)=mean(susceptible_again);
Y(6)=mean(infected_twice);
Y(7)=mean(recovered_twice);


bar(Y,0.5)
ylabel('number of population');

text(1,Y(1),'\downarrow dead','Position',[0.7 9500]);
text(2,Y(2),'\downarrow healthy','Position',[1.7 9000]);
text(3,Y(3),'\downarrow infected','Position',[2.7 9500]);
text(4,Y(4),'\downarrow recovered','Position',[3.7 9000]);
text(5,Y(5),'\downarrow re-susceptible','Position',[4.7 9500]);
```

```
text(6,Y(6),'\downarrow re-infected','Position',[5.7 9000]);
text(7,Y(7),'\downarrow re-recovered','Position',[6.7 9500]);
hold on;
errorbar(1,Y(1),std(number_of_death));
errorbar(2,Y(2),std(never_infected));
errorbar(3,Y(3),std(infected_once));
errorbar(4,Y(4),std(recovered_once));
errorbar(5,Y(5),std(susceptible_again));
errorbar(6,Y(6),std(infected_twice));
errorbar(7,Y(7),std(recovered_twice));
axis([0 8 0 10000])

%Display results
fprintf('Number of death are: %u\n',mean(number_of_death));
fprintf('Number of never infected are: %u\n',mean(never_infected));
fprintf('Number of infected once are: %u\n',mean(infected_once));
fprintf('Number of recoverd once: %u\n',mean(recovered_once));
fprintf('Number of infected twice: %u\n',mean(infected_twice));
fprintf('Number of recovered twice: %u\n',mean(recovered_twice));
fprintf('Number of susceptible again: %u\n',mean(susceptible_again));
```