AMS 333 HW# 4

Ivan Tinov

110255332

Ivan Tinov 110255332

***Influenza epidemics and Stationary Points of a Simple Autoregulatory System***

**Introduction**

Population dynamics can be modeled using the S.I.R. model that consists of susceptible organism (S), infected organisms (I), and organisms that can get re-infected (R). This model is relatively simple, modeled using the forward-euler method, and can be used for infections that last a lifetime, for example like HIV. However, a lot of infections end over time and are called acute infections. Sometimes, an organism can obtain immunity after an infection has ended and an organism cannot be re-infected unless the virus has evolved, but by that point it may be an entirely new virus. If an organism doesn’t obtain immunity then it will follow the S.I.S. model, where an organism dies after being infected or doesn’t obtain immunity after the infection has passed and is susceptible for another infection. For organisms that do obtain immunity, the S.I.R. model is used and this system can be explained using three differential equations: dS/dt = −βSI, dI/dt = −βSI − γI − νI, dR/dt = γI. For the equations, S is susceptible, I is immunity, and R is the population that recovered from the disease. Likewise, β represents the infection rate which is linear in both infected and susceptible populations, γ represents the per capita recovery rate, which is dependent on infected population only. There is a mortality rate fixed at 5%, and in each iteration we lose this amount of population. Using an infection from an influenza strain with a recovery rate of 5% per individual per day with an infection rate of 5\*10^-7 individuals^2 per day. So, for a population of one million organisms that are susceptible, with only 10 infected, we would expect five organisms to be infected each run.

**Methods Analysis**

**Influenza Epidemics - S.I.R. Model**

For this analysis a population of one million organisms will be considered, with only ten organisms being infected. A time step of 0.001 days was used and the analysis can be seen in Figure 1a. The epidemic lasts around 70 days and by the end 993,020 organisms have been infected. Half of them are dead and the mortality rate = the recovery rate which are both at 5%.

Increasing the infectivity rate by 10-fold (5\*10^-6) shows the following results seen in Figure 1b. The infection will obviously occur at a much faster rate and only beginning after a single day as opposed to taking 10 days with an infection rate of 5\*10^-7. It also begins to decline more quickly and finishing by around day 50. Similarly, half of the individuals die and half of them recover.

A 10-fold increase in the recovery rate will have the results shown in Figure 1c. In this case, the infection occurs even faster, basically right away and ends in about 15 days. The recovery rate is a lot higher than the death rate with 908,990 organisms having recovered, and only 90,899 having died. So, increasing the recovery rate will reduce the death rate.

Having a 10-fold increased infectivity rate with a default recovery rate has the results seen in Figure 1d. The recovery rate will overcome the infection rate, and therefore an epidemic cannot properly occur. The population will only increase if S > (γ + v)/β, and this is a standard for a growing infection. The infection will decay to 0, leaving only 100 recovered individuals and 10 who died from the infection. Every other individual will remain susceptible, but never infected.

The S.I.R. model helps us figure out the likelihood of an epidemic to occur, and what preventions should be taken once one occurs. Now the effects of quarantine and vaccination will be considered in order to prevent the spread of infection thru a population of organisms. The best way to do this is to remove infected organisms from a population, so that the disease will not spread. It is also assumed that all infected organisms that are removed from a population will eventually recover to have a 100% recovery rate.

Implementing this quarantine method into the S.I.R. model can be seen in Figure 2. There is clearly a difference in the way an infection occurs with quarantining. There is a total death and recovery at 319,710 organisms each. However, the goal is to reduce the mortality rate, so if we were to reduce the death rate by 50%, the mortality rate will go from 0.05 to 0.025. To have the total death rate be reduced by 50%, the quarantine rate must have an effectiveness of 0.0448. For 90%, we must have an effectiveness of 0.2284, and likewise for 99%, an effectiveness of 0.3749.

Having vaccinations to prevent an organism from being infected is another way to prevent a disease from spreading. Vaccinations work by injecting organisms of a susceptible population with a dead or weakened virus that results in the generation of antibodies and immune cells that will prevent the actual infection from occurring. Therefore, these organisms can never get infected and automatically are moved into the re-infected population. The results of the S.I.R. model with vaccinations can be seen in Figure 3. Looking at the results, around 1/3 of the population still results in a large recovery rate. This also includes the immune population initially present (300,000 organisms) and almost half of this population doesn’t get infected. To actually reduce the death count by 50%, 90%, and 99%, we need to have initial vaccinated populations of respectively 309,446, 695,450, and 765,918. Vaccinations have become easier and easier to administer so these values can be obtained realistically.

**Stationary Points of a Simple Autoregulatory System**

We can analyze concentrations of biological molecules by analyzing various dynamical systems. An example of an application is the modeling of the expression of a gene. DNA contains various sequences of nucleotides represented by the letters A, T, C, and G, which pair with each other: A-T, and C-G. This code can be converted to RNA, which only differs in the sugar present; ribose as opposed to deoxyribose and one is missing an oxygen molecule while the other isn’t. Similarly, RNA contains uracil as a base instead of thymine. Transcription is then followed by translation, in which the code on the messenger RNA molecule (mRNA) is then converted into a sequence of amino acids, the building blocks of proteins inside a ribosome. These proteins are capable of then carrying out many cellular functions.

A transcription factor is a protein basically that helps with the binding of RNA polymerase to the DNA chain. Polymerase is an enzyme that is responsible for the synthesis of the mRNA strand from the DNA template. Therefore, the transcription factor serves to increase the synthesis of mRNA, which then creates more proteins. The synthesis of proteins can be modeled using this differential equation: d[Xprot]/dt = ω[Xrna] − χprot[Xprot]. Likewise, the model for the synthesis of RNA is modeled using this differential equation:

d[Xrna]/dt = μ[Xprot]^2 / (K1/2 + [Xprot]^2) - xrna[Xrna].

Examining the system analytically we can predict the behavior of the system. We are interested in finding the stationary points, which will exist at the intersection of null clines. Therefore, at these points the system will not change in concentration. Setting the first equation (synthesis of proteins) above to zero will show that the protein concentration remains constant. Likewise, setting the second equation (synthesis of RNA) will yield a cubic equation that has solutions which give stationary points for our system.

Now, analyzing the system numerically will yield various results. Our goal is to examine the stability of the various stationary points, using the forward-euler method of numerical iteration. All of the initial values for the constants in the equation are set to 1. As seen in Figure 4, there is no real solution to our equation and the only stationary point in the system is at (0, 0). The degradation and dissociation constants consume the synthesis and transcription rates and everything will eventually decay to zero. We set our initial values for protein at 0.6 and 0.4 for RNA and both decay to zero after around 10 seconds.

Next, we alter the dissociation constant to 0.5 and obtain what can be seen in Figure 5. Now, we can see that we have a stationary point at (0.5, 0.5). Both protein and RNA concentrations decay to a fixed rate of (0.5, 0.5) after about 2 seconds.

Altering the dissociation constant to 0.33, we obtain what can be seen in Figure 6. We can see two non-zero stationary points at around (0.55, 0.55) and (0.85, 0.85). This implies that the stationary points act as attractors, as time goes on.

We can iteratively examine a variety of different starting values in order to see the overall behavior of the system that we are analyzing. So, we can adjust both the initial value for the protein and RNA concentrations in order to examine the direction that the system travels in. Using the quiver function, we can display velocity vectors at each point on an x,y plot. In our case we are examining RNA concentration versus protein concentration levels with starting values that range from 0.1 to 2.0, with a dissociation constant of 0.33. The initial value for RNA concentration is set at 0.1, while we iteratively increase the protein concentration from 0.1 to 2.0, using the forward-euler method for a time period of 100 seconds. The time step is also altered from 0.01 to 1 in order to have a cleaner graph and more visible vector arrows. This entire process can be seen in Figure 7 below.

We can observe that for very low starting values of RNA and protein, the system will decay down to (0, 0) for the stationary point. However, once the system passes thru this repelling point, it reaches an attractor point at around (0.85, 0.85). Hence, this can be observed as the clumping behavior near those lines that are centered heavily around 0.85. Here the concentration of both RNA and protein has reached a stable rate and therefore growth will be slower and slower, leading to more stability.

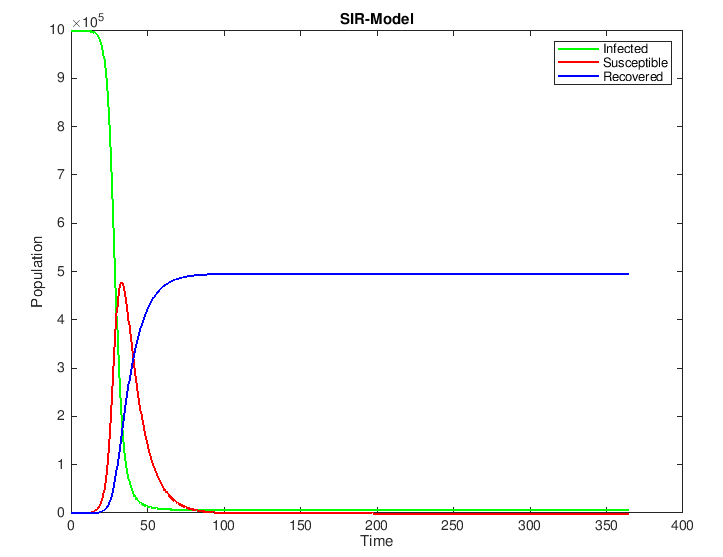
**Diagrams**

Figure 1a: S.I.R. Model

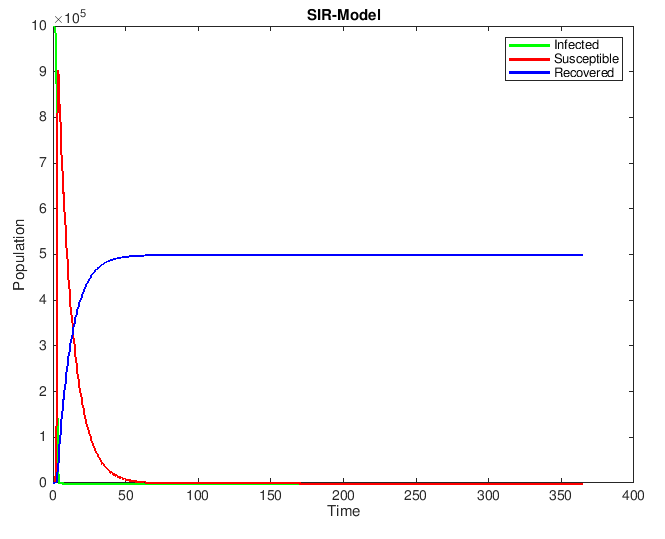


Figure 1b: S.I.R. Model with 10-fold increased infectivity

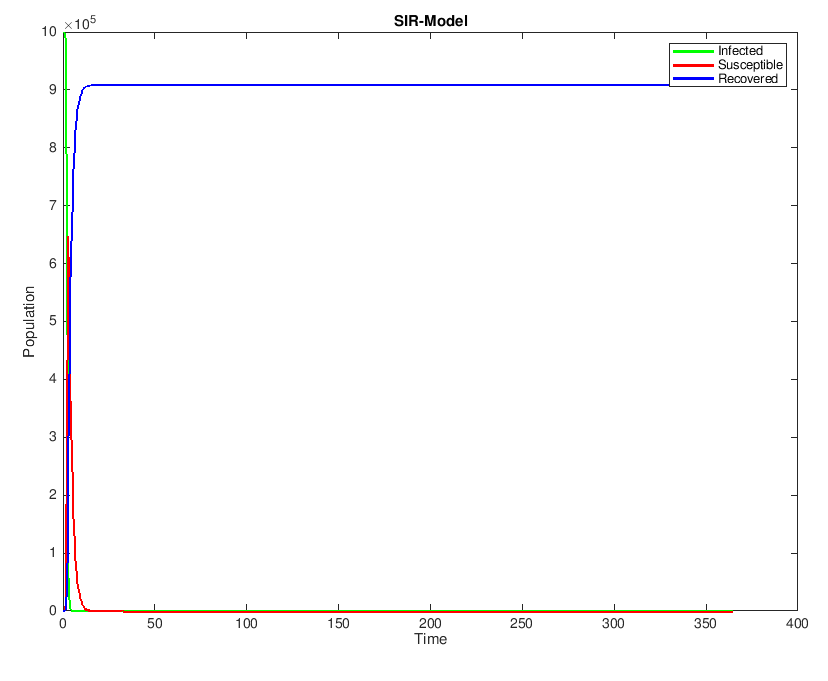


Figure 1c: S.I.R. Model with 10-fold increased infectivity and 10-fold increased recovery rate

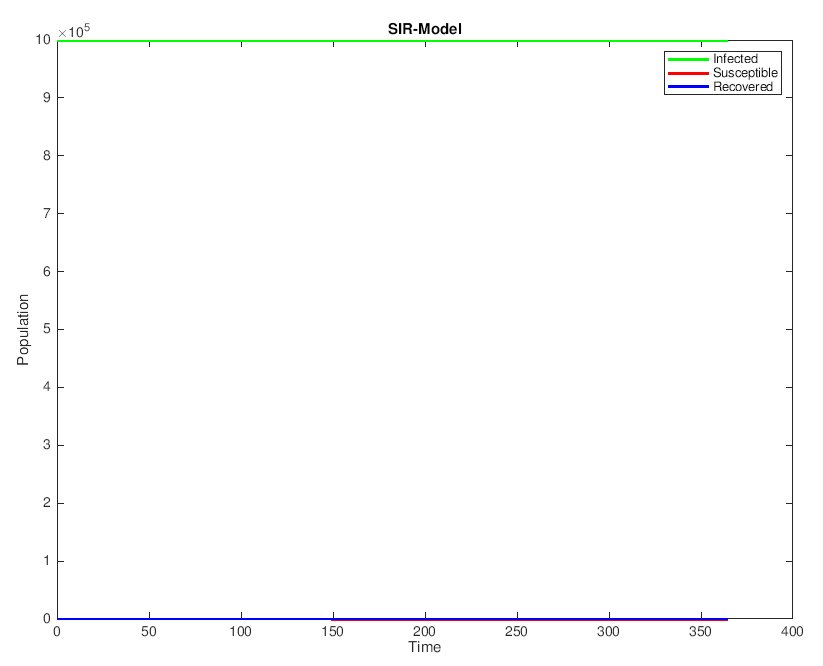


Figure 1d: S.I.R. Model with 10-fold increased infectivity and constant recovery rate

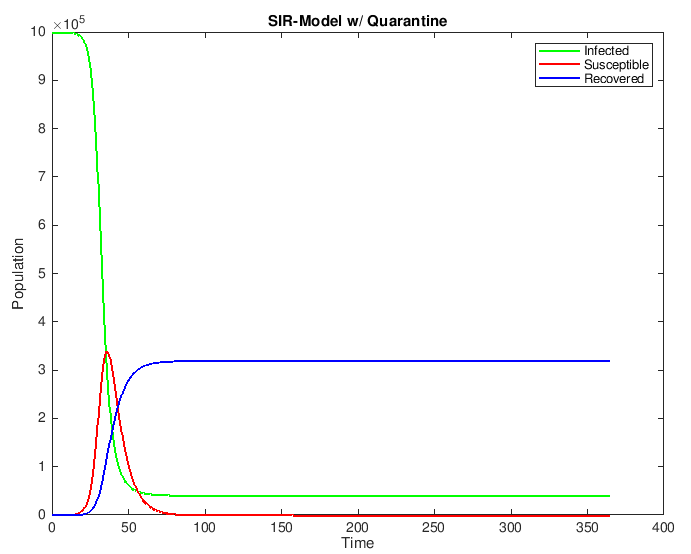


Figure 2: S.I.R. Model with Quarantine

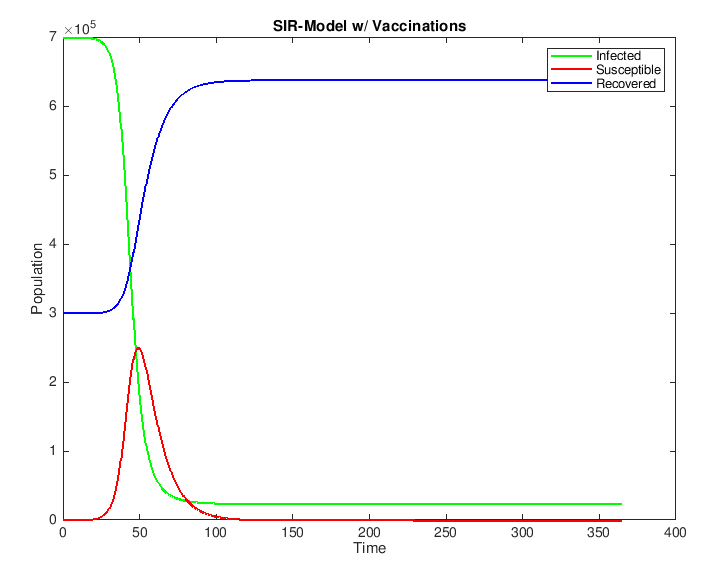
****

Figure 3: S.I.R. Model with Vaccinations

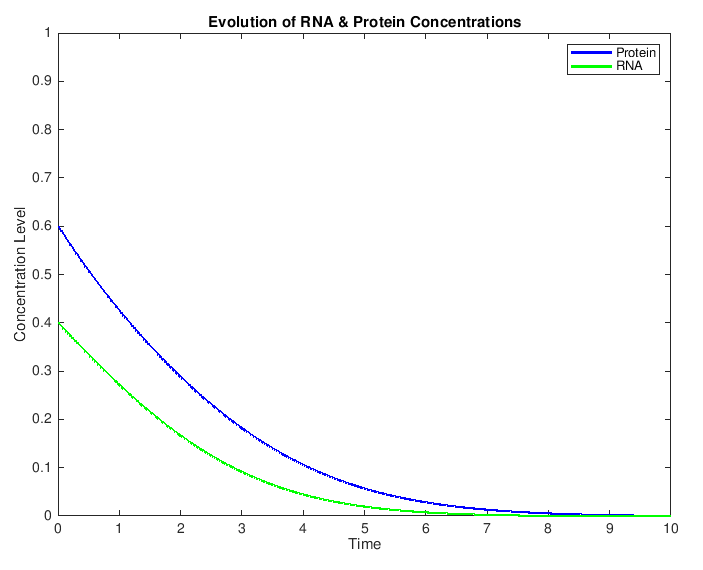
****

Figure 4: Evolution of RNA and Protein Concentrations with K1/2 = 1

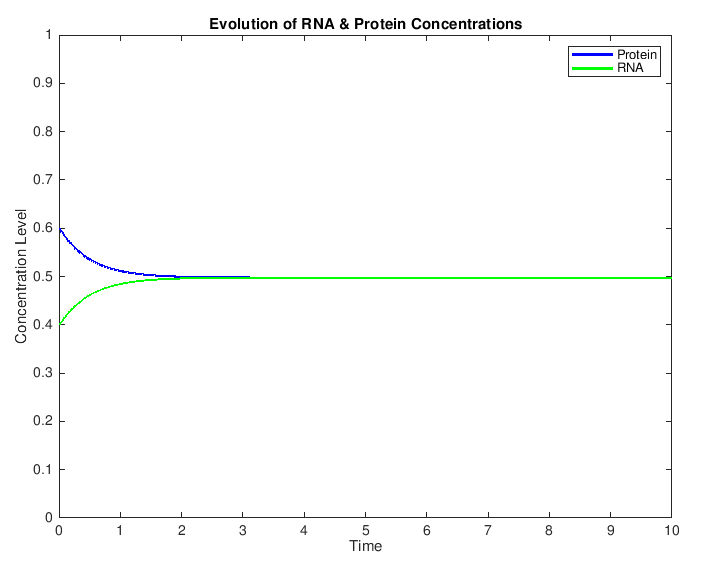


Figure 5: Evolution of RNA and Protein Concentrations with K1/2 = 0.5

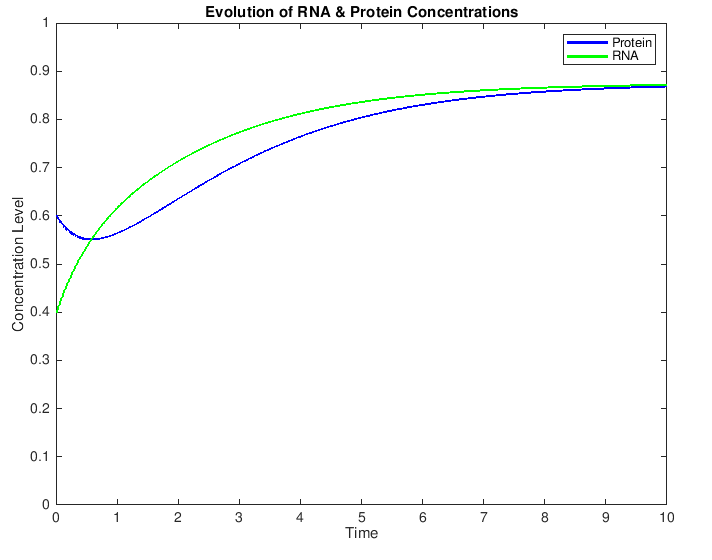


Figure 6: Evolution of RNA and Protein Concentrations with K1/2 = 0.33

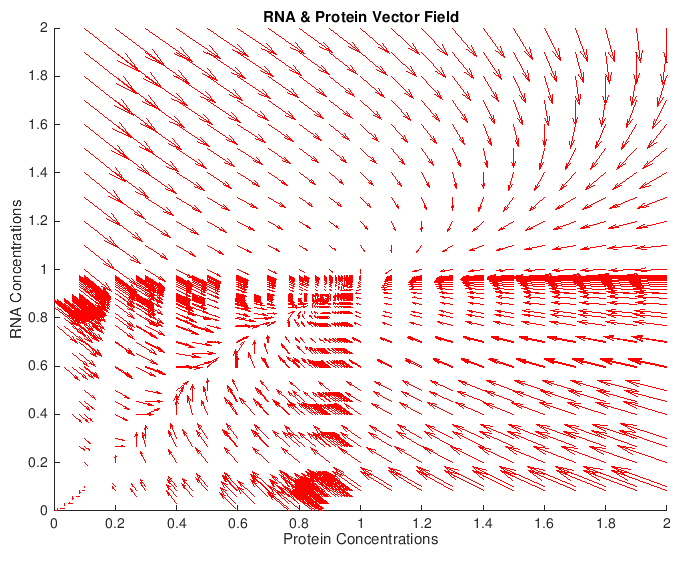


Figure 7: RNA & Protein Vector Field

**Matlab Code**

1. SIRmodel.m

S = zeros(365000,1);

I = zeros(365000,1);

R = zeros(365000,1);

D = zeros(365000,1);

S(1) = 999990;

I(1) = 10;

R(1) = 0;

D(1) = 0;

b = 5 \* 10^-7;

g = 0.05;

d = 0.05;

delt = 0.001;

t = 0:0.001:365;

for n = 1:365000

dSdt = -1 \* b \* S(n) \* I(n);

dIdt = b \* S(n) \* I(n) - g \* I(n) - d \* I(n);

dRdt = g \* I(n);

dDdt = d \* I(n);

S(n + 1) = S(n) + dSdt \* delt;

I(n + 1) = I(n) + dIdt \* delt;

R(n + 1) = R(n) + dRdt \* delt;

D(n + 1) = D(n) + dDdt \* delt;

end

plot(t,S,'g','LineWidth',2.0);

hold on

plot(t,I,'r','LineWidth',2.0);

plot(t,R,'b','LineWidth',2.0);

ylabel('Population');

xlabel('Time');

title('SIR-Model');

legend('Infected','Susceptible','Recovered');

disp('SIRmodel','­dpdf');

hold off

disp(D(365000));

disp(R(365000));

disp(I(365000));

disp(S(365000));

1. SIRmodelQ.m

S = zeros(365000,1);

I = zeros(365000,1);

R = zeros(365000,1);

D = zeros(365000,1);

S(1) = 999990;

I(1) = 10;

R(1) = 0;

D(1) = 0;

b = 5 \* 10^-7;

g = 0.05;

q = 0.05;

d = 0.05;

delt = 0.001;

t = 0:0.001:365;

for n = 1:365000

dSdt = -1 \* b \* S(n) \* I(n);

dIdt = b \* S(n) \* I(n) - g \* I(n) - q \* I(n) - d \* I(n);

dRdt = g \* I(n);

dDdt = d \* I(n);

S(n + 1) = S(n) + dSdt \* delt;

I(n + 1) = I(n) + dIdt \* delt;

R(n + 1) = R(n) + dRdt \* delt;

D(n + 1) = D(n) + dDdt \* delt;

end

plot(t,S,'g','LineWidth',1.5);

hold on

plot(t,I,'r','LineWidth',1.5);

plot(t,R,'b','LineWidth',1.5);

ylabel('Population');

xlabel('Time');

title('SIR-Model w/ Quarantine');

legend('Infected','Susceptible','Recovered');

disp('SIRmodelQ','­dpdf');

hold off

disp(D(365000));

disp(R(365000));

disp(I(365000));

disp(S(365000));

1. SIRmodelIV.m

S = zeros(365000,1);

I = zeros(365000,1);

R = zeros(365000,1);

D = zeros(365000,1);

S(1) = 699990;

I(1) = 10;

R(1) = 300000;

D(1) = 0;

b = 5 \* 10^-7;

g = 0.05;

d = 0.05;

delt = 0.001;

t = 0:0.001:365;

for n = 1:365000

dSdt = -1 \* b \* S(n) \* I(n);

dIdt = b \* S(n) \* I(n) - g \* I(n) - d \* I(n);

dRdt = g \* I(n);

dDdt = d \* I(n);

S(n + 1) = S(n) + dSdt \* delt;

I(n + 1) = I(n) + dIdt \* delt;

R(n + 1) = R(n) + dRdt \* delt;

D(n + 1) = D(n) + dDdt \* delt;

end

plot(t,S,'g','LineWidth',1.5);

hold on

plot(t,I,'r','LineWidth',1.5);

plot(t,R,'b','LineWidth',1.5);

ylabel('Population');

xlabel('Time');

title('SIR-Model w/ Vaccinations');

legend('Infected','Susceptible','Recovered');

disp('SIRmodelV','­dpdf');

hold off

disp(D(365000));

disp(R(365000));

disp(I(365000));

disp(S(365000));

1. fwdEulerGeneReg.m

Xprot=zeros(1000,1);

Xrna=zeros(1000,1);

Xprot(1)=0.6;

Xrna(1)=0.4;

delt=0.01;

K=0.33; % 0.5, 0.33

w=1;

u=1;

xprot=1;

xrna=1;

t=0:0.01:10;

for n=1:1000

dXprotdt = w \* Xrna(n) - xprot \* Xprot(n);

dXrnadt = u \* Xprot(n)^2 / (K^2 + Xprot(n)^2) - xrna \* Xrna(n);

Xprot(n+1) = Xprot(n) + dXprotdt \* delt;

Xrna(n+1) = Xrna(n) + dXrnadt \* delt;

end

plot(t,Xprot,'b', 'LineWidth', 2.0);

hold on

plot(t,Xrna,'g', 'LineWidth', 2.0);

axis([0 10 0 1]);

title('Evolution of RNA & Protein Concentrations');

xlabel('Time');

ylabel('Concentration Level');

legend('Protein','RNA');

disp('fwdEulerGeneReg','­dpdf');

hold off

1. quivPlot.m

Xprot=zeros(100,1);

Xrna=zeros(100,1);

u=zeros(101,1);

v=zeros(101,1);

hold on

for j=1:20

Xrna(1) = j/10;

for i=1:20

Xprot(1) = i/10;

delt=1;

K=0.33;

w=1;

m=1;

xprot=1;

xrna=1;

for n=1:100

dXprotdt = w \* Xrna(n) - xprot \* Xprot(n);

dXrnadt = m \* Xprot(n)^2 / (K^2 + Xprot(n)^2) - xrna \* Xrna(n);

u(n) = dXprotdt;

v(n) = dXrnadt;

Xprot(n+1) = Xprot(n) + dXprotdt \* delt;

Xrna(n+1) = Xrna(n) + dXrnadt \* delt;

end

quiver(Xprot,Xrna,u,v,'r');

end

end

axis([0 2 0 2]);

title('Protein and RNA Vector Field');

xlabel('Protein Concentration');

ylabel('RNA Concentration');

disp('quivPlot','­dpdf');

hold off