# Math227A HW6

Zachary DeStefano

November 21, 2016

## 1 Problem 1

### 1.1 Part A

The original set of equation is as follows

$$E' = -k_{on}ES + k_{off}C + k_{cat}C$$
$$S' = -k_{on}ES + k_{off}C$$
$$C' = k_{on}ES - k_{off}C - k_{cat}C$$
$$P' = k_{cat}C$$

Using the mass conservation equations we can say that

$$E = E_t - C$$

Which allows us to assert that $E' = -C'$. This allows us to make the original system the following

$$-C' = -k_{on}(E_t - C)S + k_{off}C + k_{cat}C$$
$$S' = -k_{on}(E_t - C)S + k_{off}C$$
$$C' = k_{on}(E_t - C)S - k_{off}C - k_{cat}C$$
$$P' = k_{cat}C$$

If we let $C = E_t c$ then $C' = E_t c'$. The first and third equations are equivalent. Mass conservation also states that $S_t = S + C + P$ which implies that $S' + C' + P' = 0$ which is true from the above. The $P'$ equation thus is equivalent to other statement. Thus our system is reduced to the following

$$S' = -k_{on}(E_t - E_t c)S + k_{off}E_t c$$
$$E_t c' = k_{on}(E_t - E_t c)S - k_{off}E_t c - k_{cat}E_t c$$

Simplifying we now have

$$S' = -E_t k_{on}(1-c)S + k_{off}E_t c$$
$$c' = k_{on}(1-c)S - k_{off}c - k_{cat}c$$

Thus the final form in terms of $c, S$ is as follows

$$S' = E_t(k_{off}c - k_{on}(1-c)S)$$
$$c' = k_{on}(1-c)S - c(k_{off} + k_{cat})$$

## 1.2 Part B

We have $E_t = \epsilon S_t$ thus the $S'$ equation becomes

$$S' = \epsilon S_t(k_{off}c - k_{on}(1-c)S)$$

Because $S'$ has the $\epsilon$ value shrinking it, it will be a slow variable. By comparison $c$ will be a fast variable.
To find convergence after a time period, set $c' = 0$ and assume $S$ is a constant for fast timescale analysis. Thus we have

$$k_{on}(1-c)S = c(k_{off} + k_{cat})$$

After doing some algebra

$$k_{on}S - k_{on}cS = c(k_{off} + k_{cat})$$

$$k_{on}S = c(k_{off} + k_{cat} + k_{on}S)$$

$$c = \frac{k_{on}S}{k_{off} + k_{cat} + k_{on}S}$$

Dividing the numerator and denominator by $k_{on}$

$$c = \frac{S}{\frac{k_{off}+k_{cat}}{k_{on}} + S}$$

If we let $K_m = \frac{k_{off}+k_{cat}}{k_{on}}$ then we have

$$c = \frac{S}{K_m + S}$$

## 1.3 Part C

In slow timescale analysis, we assume that $c$ converges to the value from $B$ right away so that $S'$ becomes the following

$$S' = E_t(k_{off}\frac{S}{K_m+S} - k_{on}(1-\frac{S}{K_m+S})S)$$

Factoring out $S$ gives us

$$S' = E_tS(\frac{k_{off}}{K_m+S} - k_{on}(1-\frac{S}{K_m+S}))$$

After more algebra

$$S' = E_tS(\frac{k_{off}}{K_m+S} - k_{on} + \frac{k_{on}S}{K_m+S}))$$

Bringing out the denominator

$$S' = \frac{E_tS}{K_m+S}(k_{off} - k_{on}(K_m+S) + k_{on}S)$$

After doing some distributing

$$S' = \frac{E_tS}{K_m+S}(k_{off} - k_{on}K_m)$$

After plugging in one of the $K_m$ values

$$S' = \frac{E_tS}{K_m+S}(k_{off} - (k_{off} + k_{cat}))$$

After further simplifying

$$S' = -\frac{E_tS}{K_m+S}(k_{cat})$$

We know that $c = \frac{S}{K_m+S}$ so we have

$$S' = -E_tc(k_{cat})$$

We know that $C = E_tc$ thus we have

$$S' = -k_{cat}C$$

We know that $P' = k_{cat}C$ thus we have shown that $P' = -S'$

Before plugging $c$ and $C$ back in we showed that

$$S' = -k_{cat}E_t\frac{S}{K_m+S}$$

Combine this with the fact that $P' = -S'$ and it holds that

$$P' = k_{cat}E_t\frac{S}{K_m+S}$$

Thus showing Michaelis Menten

We can transform the equation into the following assuming $S \neq 0$

$$P' = k_{cat}E_t\frac{1}{1+\frac{K_m}{S}}$$

Intuitively this means that the rate of product being produced will increase as the substrate concentration increases, but it will begin to increase more slowly and asymptotically approach a constant.

## 2 Problem 2

### 2.1 Part A

Here is the matlab code for a function that samples from the exponential distribution

```
function [ res ] = sampleFromDist( lambda, Nvals )
%SAMPLEFROMDIST obtain N samples from distribution

%{
For Part A, sample y from 0 to 1
then lambda * exp(-lambda * t ) = lambda*y
since the result will range from 0 to lambda

log of both sides gives
-lambda*t = log(y)
t = -log(y)/lambda
%}

yVec = rand(Nvals,1);
res = -log(yVec)./lambda;

end
```

### 2.2 Part B

Here is the code for the function that computes a timecourse given an arbitrary finite graph and starting node.

```
function [ times,states ] = computeTimecourse( qMatrix,initState,numSteps )
%COMPUTETIMECOURSE
%
%Input:
%   qMatrix - matrix for graph where entry (i,j) corresponds to the edge
%             from state i to state j
%   initState - starting state
%   numSteps - number of steps to compute
%
%Output:
%   times - time elapsed after each step
%   states - current state at each step

times = zeros(1,numSteps); %time point
states = zeros(1,numSteps); %current state at each point

states(1) = initState;
times(1) = 0;

curT = 0;
curState = initState;

for i = 2:numSteps

    %lambdaBar to use to sample time to transition
    lambdaBar = sum(qMatrix(curState,:));

    %sample time from exponential dist
    sampledT = sampleFromDist(lambdaBar,1);
    curT = curT + sampledT;

    %get uniform sample
    sampledR = rand(1);
    %put it in 0 to lambdaBar range
    sampledLambda = sampledR*lambdaBar;

    %find which state that value corresponds to
    curState = find(cumsum(qMatrix(curState,:))>sampledLambda,1,'first');

    states(i) = curState;
    times(i) = curT;

end

end
```

Figure 1 shows the arbitrary finite graph I used to test it on. My initial state was 1. I did 3 timecourses and each one consisted of 30 steps. Figure 2 shows the result. Here is the script I used to generate that plot.

```
qMatrix = [0 5 0 0;2 0 1 0;0 0 0 7;1 0 3 0];

initState = 1;
numSteps = 30;

numPlots = 3;
timesArr = cell(1,numPlots);
statesArr = cell(1,numPlots);
for jj = 1:numPlots
    [timesArr{jj},statesArr{jj}] = computeTimecourse(qMatrix,initState,numSteps);
end

figure
hold on
stairs(timesArr{1},statesArr{1},'r','LineWidth',5)
stairs(timesArr{2},statesArr{2},'g','LineWidth',3)
stairs(timesArr{3},statesArr{3},'k')
xlabel('Time elapsed');
ylabel('Current state');
legend('Timecourse 1','Timecourse 2','Timecourse 3');
axis([0 inf 0 5])
hold off
```

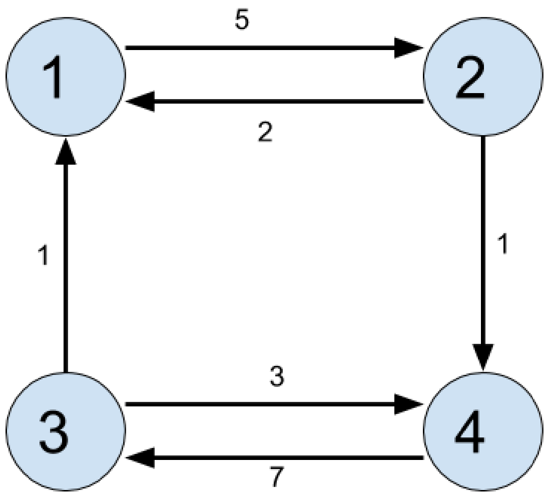

Figure 1: Finite graph with 4 states and varying transition probabilities

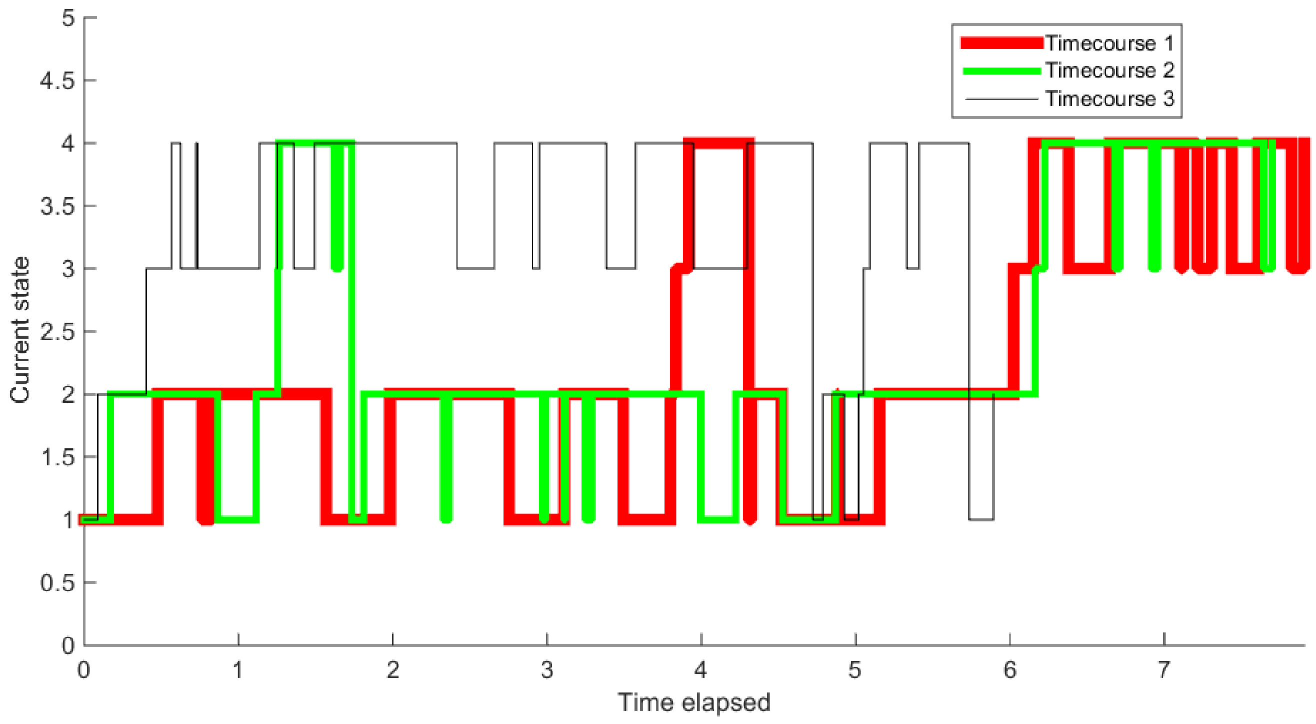

Figure 2: Graph of 3 different timecourses

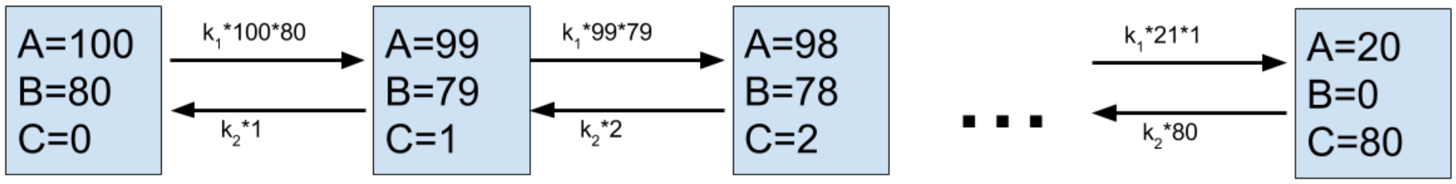

Figure 3: Graph of Reaction states and transition probabilities

## 2.3 Part C

Figure 3 shows the graph. Figure 4 shows the graph in a more symbolic form. Figure 5 shows the graph of number of chemical reaction elements over time using the graph. This is the script used in Matlab to compute that plot.

```
%k1,k2 values
k1=2;
k2=3;

%A,B,C values in each state
Avals=100:-1:20;
Bvals=80:-1:0;
Cvals=0:1:80;

%initialize transition matrix
qMatrix=zeros(length(Avals));

%assign transition values correctly
for state=1:80
    qMatrix(state,state+1)=k1*Avals(state)*Bvals(state);
    qMatrix(state+1,state)=k2*Cvals(state+1);
end

%initial state of 1, do 100 time steps
[chemTimes,chemStates]=computeTimecourse(qMatrix,1,100);

%make array for A,B,C values
Acourse = Avals(chemStates);
Bcourse = Bvals(chemStates);
Ccourse = Cvals(chemStates);

%plot the results
figure
hold on
stairs(chemTimes,Acourse,'r','LineWidth',2)
stairs(chemTimes,Bcourse,'g','LineWidth',2)
stairs(chemTimes,Ccourse,'k','LineWidth',2)
xlabel('Time elapsed');
ylabel('Number of Elements');
legend('chemical A','chemical B','chemical C');
hold off
```

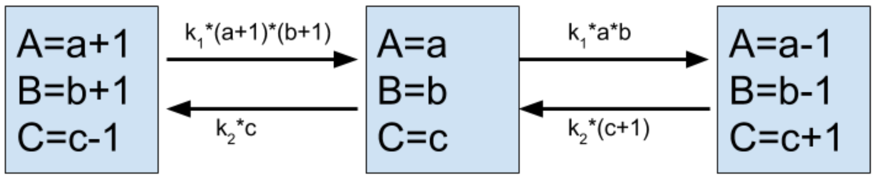

Figure 4: Graph of Reaction states and transition probabilities in symbolic form. The left most state is $(100, 80, 0)$ and the figure shows every triple in the graph until the right most state is $(20, 0, 80)$

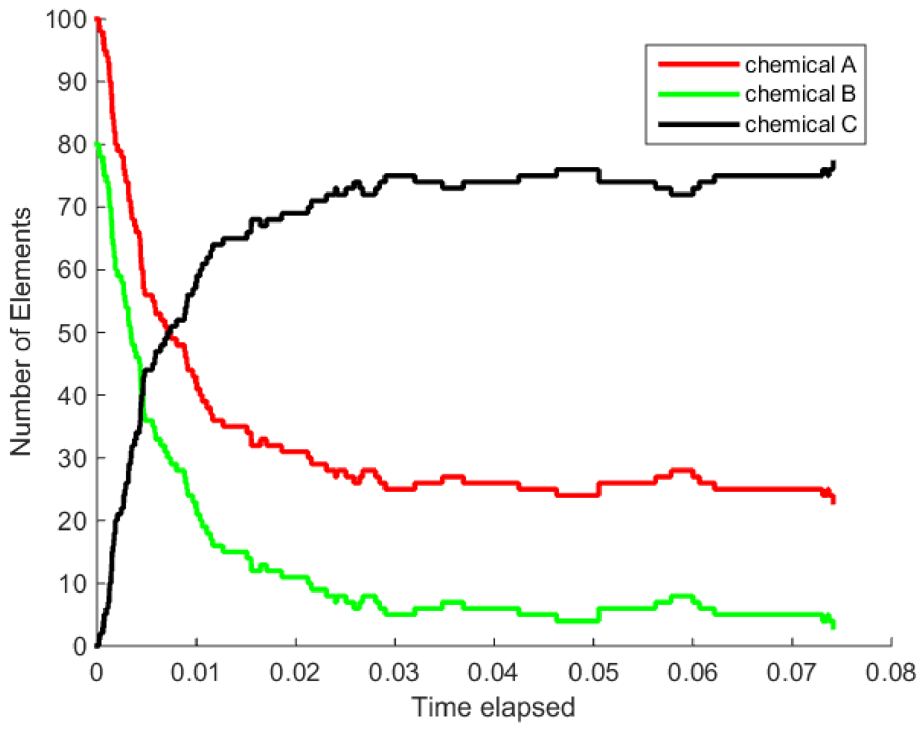

Figure 5: Graph of number of chemical elements vs time

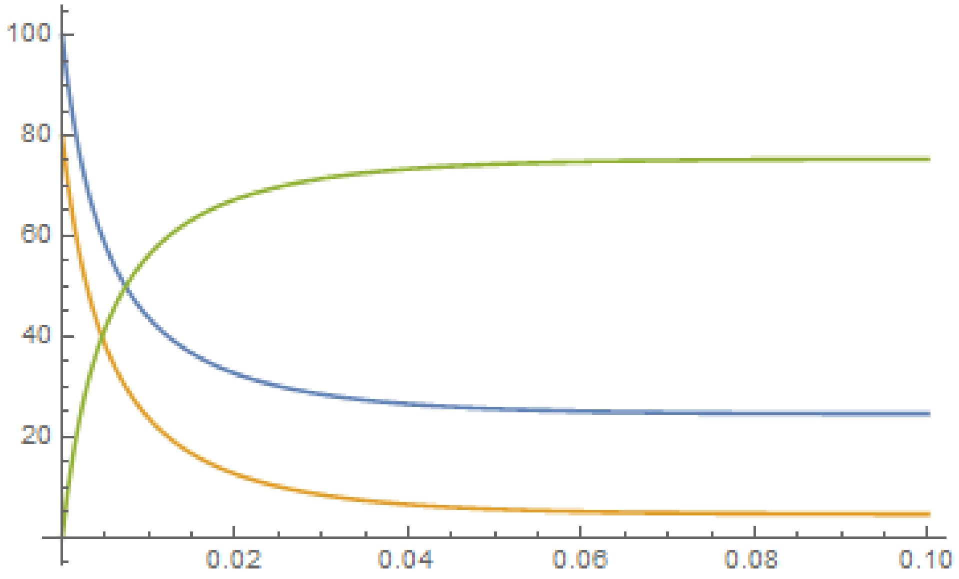

Figure 6: Graph of number of chemical elements vs time using deterministic equation. Blue is chemical A. Orange is chemical B. Green is chemical C.

## 2.4 Part D

I did a mathematical analysis of the system. We know from the reaction that

$$C' = k_1 AB - k_2 C$$

We also have the following mass conservation equations

$$A_t = A + C$$

$$B_t = B + C$$

Combining these, we have an expression for $C'$ that only includes constants and the variable $C$

$$C' = k_1(A_t - C)(B_t - C) - k_2 C$$

After we solve for $C$, we can use the mass conservation equations to deduce $A$ and $B$. In this vein, I used the following commands in Mathematica

```
k1 = 2; k2 = 3; At = 100; Bt = 80;

Cvalues =
  NDSolveValue[{CC'[t] == k1 (At - CC[t]) (Bt - CC[t]) - k2*CC[t],
    CC[0] == 0}, CC, {t, 0, .1}];

Plot[{100 - Cvalues[t], 80 - Cvalues[t], Cvalues[t]}, {t, 0, .1},
 PlotLegends -> {"Chemical A", "Chemical B", "Chemical C"}]
```

Figure 6 shows the results. It is nearly the same as the result when doing random walk, just smoother.

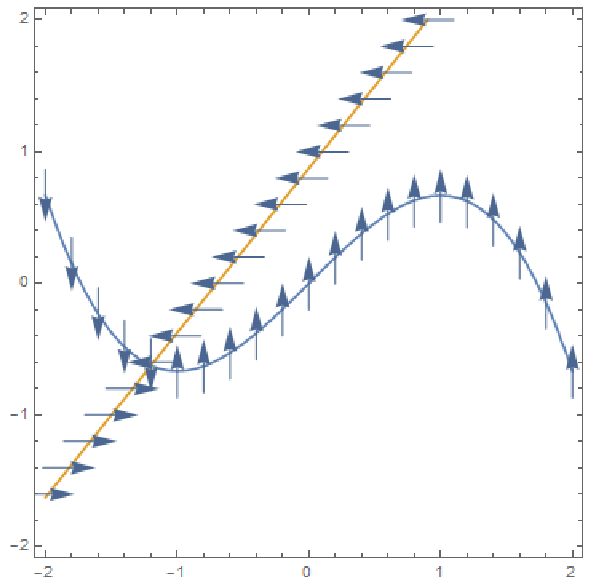

Figure 7: Graph of the null clines

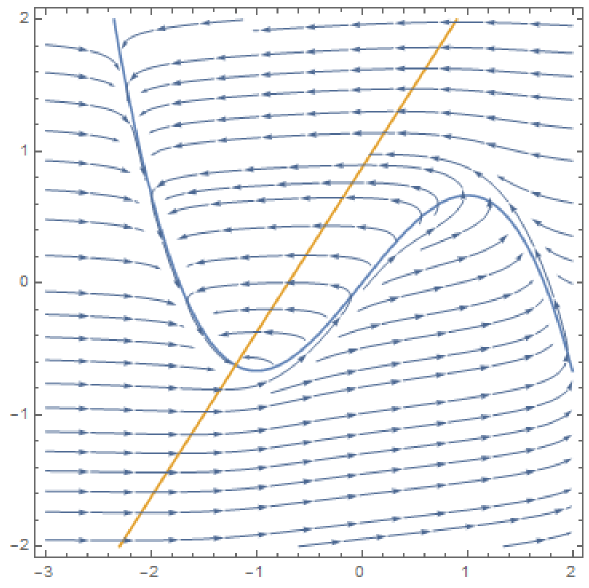

Figure 8: Graph of the stream plot

## 3 Problem 3

### 3.1 Part A

Figure 7 shows the initial nullclines. The stream plot is shown in Figure 8. Some corresponding solutions are shown in Figure 9. There seems to be oscillatory behavior around the steady state.

### 3.2 Part B

I set $I = 5$. Figure 10 shows the initial nullclines. The stream plot is shown in Figure 11. Some corresponding solutions are shown in Figure 12. It seems to converge toward the steady state and not recover. In the graphs in Part A by comparison, it seems there would be a recovery.

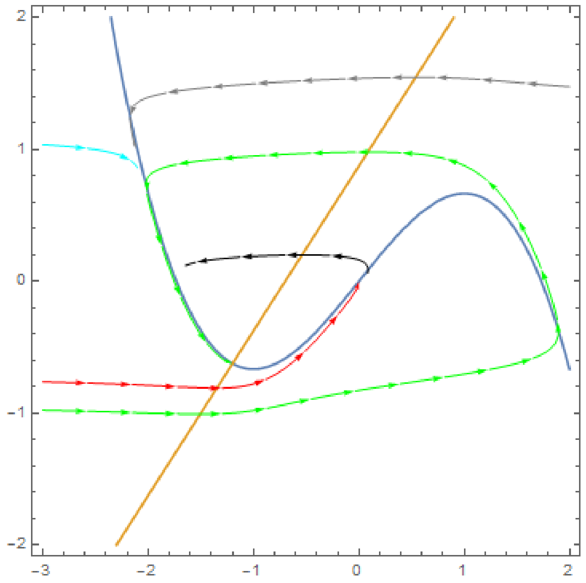

Figure 9: Graph of some sample solutions

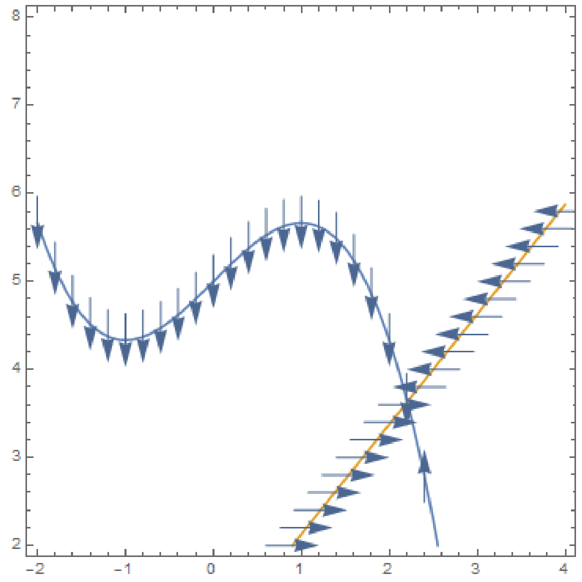

Figure 10: Graph of the null clines

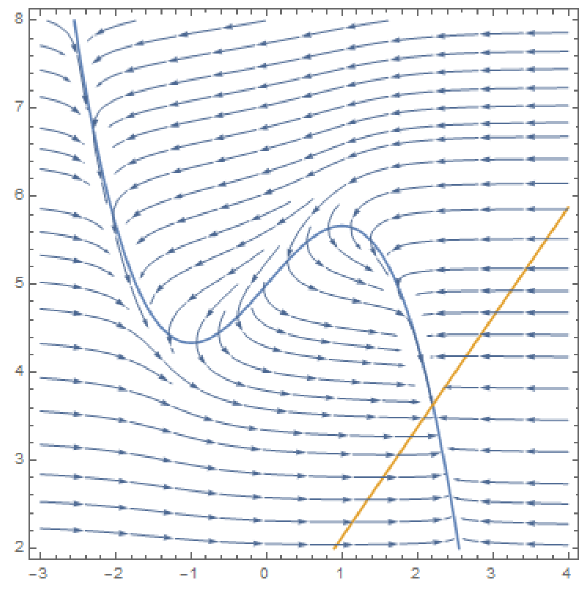

Figure 11: Graph of the stream plot

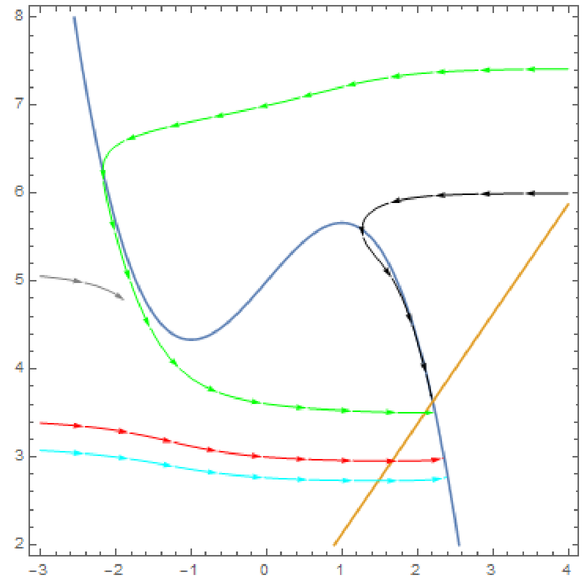

Figure 12: Graph of some sample solutions

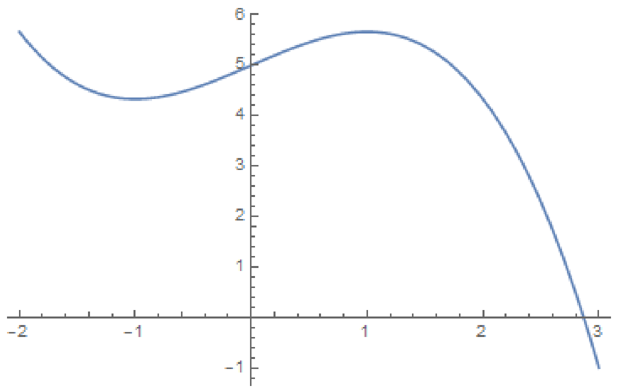

Figure 13: Graph of $v - v^3/3 = A$ when $A = 5$

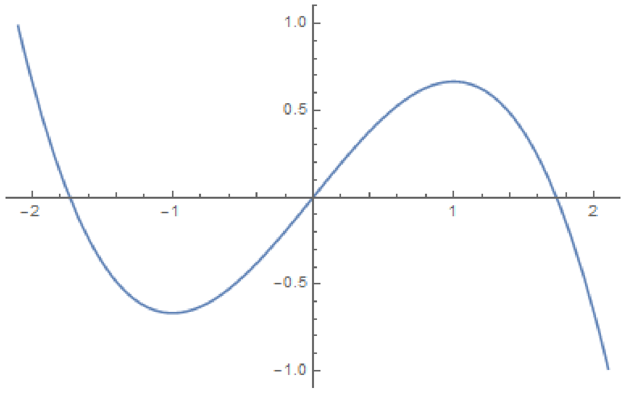

Figure 14: Graph of $v - v^3/3 = A$ when $A = 0$

## 3.3 Part C

Since $w'$ has the $\epsilon$ term, I will assume $w$ is the slow term and $v$ is the fast term.

Using fast decomposition, we want to solve

$$v - v^3/3 - w + I = 0$$

$$v - v^3/3 = w - I$$

Let $A = w - I$ be a parameter. Figure 13 shows the phase diagram if $A$ is very positive or very negative. Notice there is one stable steady state. Figure 14 shows the phase diagram if $A$ is near zero. Notice that there are three steady states, the outer ones are stable and the inner one is unstable. Figure 15 shows the resulting bifurcation graph.

For slow decomposition, we assume that $v$ has already converged to its stable state. We thus have

$$w' = -c\epsilon w + \epsilon v + \epsilon d$$

The phase diagram is thus a linear plot with a stable steady state at $w = \frac{v+d}{c}$. In the case where $v$ has only one stable steady state, we can thus say that the system will converge in $w$ too.

We will now focus on the case where $I(t)$ is near zero, because then there are three steady states. From the equation for $w'$ we can say that if $w - v < 0$ so $w < v$ then $w' < 0$. Similarly if $w > v$ then $w' > 0$. Figure 16 shows the plot of $v' = 0$ and whether or not $w' > 0$. Since we are in slow time decomposition, we assume that we are starting when $v$ has converged already, so along the $v' = 0$ line. When we are in the beige part of the plot, we travel to the right along the solid line since $w' > 0$. When we reach the unstable steady state, $w' > 0$ still holds so $w$ will increase and it will thus jump to the lower steady state in the blue region. When we are in the blue region, we travel to the left along the solid line since $w' < 0$. When we reach the unstable steady state, $w' < 0$ still holds so $w$ will decrease and it will jump to the higher steady state in the beige region. The system will thus indefinitely oscillate between the beige and blue region. Because the oscillations are periodic and not smooth like sinusoidal curves, they can be considered relaxation oscillations.

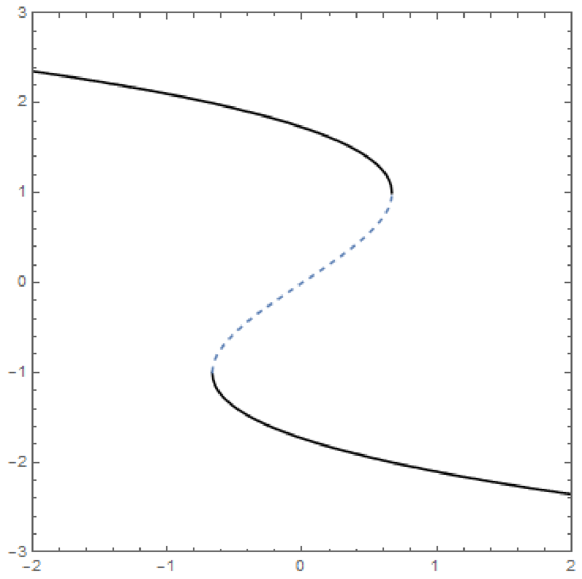

Figure 15: Bifurcation Graph of $v - v^3/3 = A$ where the x-axis is $A$ and y-axis is $v$

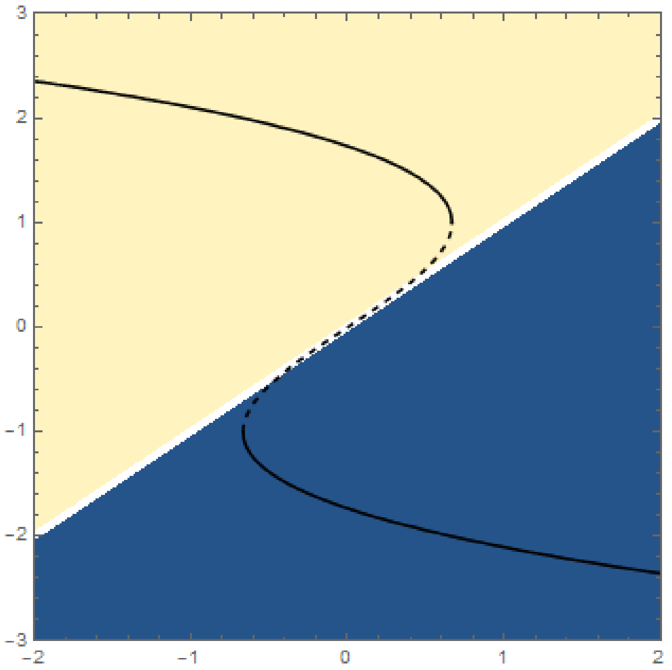

Figure 16: Graph of $v - v^3/3 = w$ where the x-axis is $w$ and y-axis is $v$. The solid line represents the stable steady states and the dashed line represents unstable steady states. The beige region is where $w' > 0$ and the blue region is where $w' < 0$

### 3.4 Part D

As we saw previously if the $w - I$ value is too high or too low, the system will converge to a steady state with $v$ or $w$ and not do much. However a small change in $I$ when it is near the border of changing stability behavior will cause the oscillatory behavior to emerge. Because this small change in input can cause different behavior, the system can be considered excitable with respect to $I$.