Homework One

Student: Timothy Jones
Math 723: Mathematics of Neuroscience
Professor: Medvedev
Spring 2008

1.1.

Follow the lines of the example analyzed in Sec. 3.2 and 3.3 in the handout (Elements of the theory of differential equations) to analyze the system of equations of motion of a pendulum:

$$\begin{cases} \dot{x} &= y, \\ \dot{y} &= -\sin(x) \end{cases} \qquad (1)$$

In particular,

a. Use the graph of the potential energy $U(x) = 1 - \cos(x)$ to construct the phase plane of (1).

Given a system $\ddot{x} = -\sin(x)$, we can write an equivalent system of two equations:

$$\begin{cases} \dot{x} &= y, \\ \dot{y} &= -\sin(x) \end{cases}$$

In general, when $\ddot{x} = f(x),\ x \in \mathbb{R}$, we have kinetic energy given by $T(\dot{x}) = \frac{1}{2}\dot{x}^2$ and potential energy $U(x) = -\int_{x_0}^{x} f(\zeta)d\zeta$. In our current case, $\int_{x_0}^{x} -\sin(\zeta)d\zeta = \cos(\zeta)\ |_{x_0}^{x}$. We choose $x = 0$ to be our reference point for the potential energy, whereby $U(x) = 1 - \cos(x)$ and our total energy is $E = \frac{y^2}{2} + 1 - \cos(x)$.

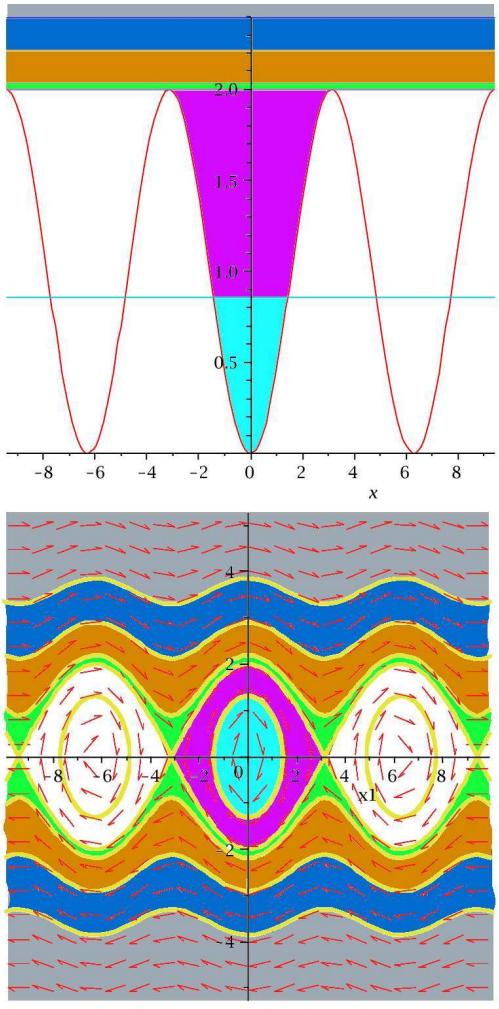


FIGURE 1.1. Top graph shows the potential energy, bottom graph the phase portrait for x verses y. The color regions correspond to the potential energies as found in the phase portrait.

Professor: Dr. Medvedev
Student: Timothy Jones

Below we include the maple code used to generate the above graphs:

```
>plot([1-cos(x), 1+(1/2)*1.7^2-cos(20), 1+(1/2)*1.8^2-cos(20), \\
 1+(1/2)*1.9^2-cos(20), 2, 1+(1/2)*1.3^2-cos(1/6)], x = -3*Pi .. 3*Pi)

> with(DEtools);
> EQ1 := diff(x1(t), t) = x2(t); EQ2 := diff(x2(t), t) = -sin(x1(t));

> DEplot([EQ1, EQ2], [x1(t), x2(t)], t = 0 .. 10, x1 = -3*Pi .. 3*Pi,\\
    x2 = -10 .. 10, title = 'Spiral*stable*point', arrows = medium)

> part1 := phaseportrait([EQ1, EQ2], [x1(t), x2(t)], t = 0 .. 150,
[[x1(0) = 5*Pi, x2(0) = -0.1e-1]], stepsize = .1, x1 = -3*Pi .. 3*Pi, x2 = -5 .. 5);
> part15 := phaseportrait([EQ1, EQ2], [x1(t), x2(t)], t = 0 .. 150,
 [[x1(0) = -5*Pi, x2(0) = 0.1e-1]], stepsize = .1, x1 = -3*Pi .. 3*Pi, x2 = -5 .. 5);
> part2 := phaseportrait([EQ1, EQ2], [x1(t), x2(t)], t = 0 .. 100,
 [[x1(0) = 20, x2(0) = -1.9]], stepsize = .1, x1 = -3*Pi .. 3*Pi, x2 = -5 .. 5);
> part3 := phaseportrait([EQ1, EQ2], [x1(t), x2(t)], t = 0 .. 100,
 [[x1(0) = 2*Pi+1, x2(0) = 1.7]], stepsize = .1, x1 = -3*Pi .. 3*Pi, x2 = -5 .. 5);
> part4 := phaseportrait([EQ1, EQ2], [x1(t), x2(t)], t = 0 .. 100,
 [[x1(0) = -20, x2(0) = 1.9]], stepsize = .1, x1 = -3*Pi .. 3*Pi, x2 = -5 .. 5);
> part5 := phaseportrait([EQ1, EQ2], [x1(t), x2(t)], t = 0 .. 100,
 [[x1(0) = -2*Pi-1, x2(0) = 1.7]], stepsize = .1, x1 = -3*Pi .. 3*Pi, x2 = -5 .. 5);
> part6 := phaseportrait([EQ1, EQ2], [x1(t), x2(t)], t = 0 .. 100,
 [[x1(0) = -1, x2(0) = 1.7]], stepsize = .1, x1 = -3*Pi .. 3*Pi, x2 = -5 .. 5);
> part7 := phaseportrait([EQ1, EQ2], [x1(t), x2(t)], t = 0 .. 100,
 [[x1(0) = 20, x2(0) = -2.7]], stepsize = .1, x1 = -3*Pi .. 3*Pi, x2 = -5 .. 5);
> part8 := phaseportrait([EQ1, EQ2], [x1(t), x2(t)], t = 0 .. 100,
 [[x1(0) = -20, x2(0) = 2.7]], stepsize = .1, x1 = -3*Pi .. 3*Pi, x2 = -5 .. 5);
> part9 := phaseportrait([EQ1, EQ2], [x1(t), x2(t)], t = 0 .. 100,
 [[x1(0) = 20, x2(0) = -3.7]], stepsize = .1, x1 = -3*Pi .. 3*Pi, x2 = -5 .. 5);
> part10 := phaseportrait([EQ1, EQ2], [x1(t), x2(t)], t = 0 .. 100,
 [[x1(0) = -20, x2(0) = 3.7]], stepsize = .1, x1 = -3*Pi .. 3*Pi, x2 = -5 .. 5);
> part11 := phaseportrait([EQ1, EQ2], [x1(t), x2(t)], t = 0 .. 100,
 [[x1(0) = 20, x2(0) = -2.7]], stepsize = .1, x1 = -3*Pi .. 3*Pi, x2 = -5 .. 5);
> part12 := phaseportrait([EQ1, EQ2], [x1(t), x2(t)], t = 0 .. 100,
[[x1(0) = 2*Pi+1/6, x2(0) = 1.3]], stepsize = .1, x1 = -3*Pi .. 3*Pi, x2 = -5 .. 5);
> part13 := phaseportrait([EQ1, EQ2], [x1(t), x2(t)], t = 0 .. 100,
 [[x1(0) = -2*Pi-1/6, x2(0) = 1.3]], stepsize = .1, x1 = -3*Pi .. 3*Pi, x2 = -5 .. 5);
> part14 := phaseportrait([EQ1, EQ2], [x1(t), x2(t)], t = 0 .. 100,
 [[x1(0) = 1/6, x2(0) = 1.3]], stepsize = .1, x1 = -3*Pi .. 3*Pi, x2 = -5 .. 5);

> display([part1, part2, part4, part7, part8, part9, part10, part11,
                                              part12, part13, part14, part15])
```


Professor: Dr. Medvedev
Student: Timothy Jones

b. Locate the fixed point(s). Linearize (1) around the fixed points.

Sketch the phase portraits for the linearized systems and compare them with the phase portrait of (1).

State if the Hartman-Grobman theorem applies to the linearized systems.

Discuss stability of the fixed points.

We write

$$\dot{\vec{x}} = \vec{F}(\vec{x}) = \begin{cases} \dot{x_1} &= x_2, \\ \dot{x_2} &= -\sin(x_1) \end{cases}$$

The fixed points are where $\vec{F}(\vec{x}) = 0$, which we write as $\vec{F}(\bar{\vec{x}}) = 0$ for the fixed points $\bar{\vec{x}}$. It is most clear that $\bar{\vec{x}}$ must take the form $(\bar{x}_1, 0$ since $\dot{x}_1 = x_2 = 0$. Our other equation, $\dot{x}_2 = -\sin(\bar{x}_1)$ is satisfied for $x_1 = n\pi$, $n \in \mathbb{Z}$. Thus we will have fixed points at all points that satisfy $\bar{x} = (n\pi, 0),\ n \in \mathbb{Z}$.

We now introduce the typical change of coordinates, $\vec{\zeta} = \vec{x} - \bar{\vec{x}}$, so that $\dot{\vec{\zeta}} = \dot{\vec{x}}$. The Taylor expansion is

$$\dot{\vec{\zeta}} = F(\bar{\vec{x}}) + DF(\bar{\vec{x}})\hat{\zeta} + \mathcal{O}(\zeta^2) \approx DF(\bar{\vec{x}})\hat{\zeta}$$

With this linearization we can compute the eigen values of the linearized matrix:

$$DF(\bar{\vec{x}}) = \begin{pmatrix} 0 & 1 \\ -\cos(n\pi) & 0 \end{pmatrix} = \begin{cases} \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} & \text{n even} \implies \lambda^2 + 1 = 0,\ \lambda = \pm i \\ \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} & \text{n odd} \implies \lambda^2 - 1 = 0,\ \lambda = \pm 1 \end{cases}$$

Thus we see that the Hartman-Grobman theorem does not apply to the n-even case, but does to the n-odd case.

For n-even we have the following eigenvalue and eigenvector combination:

$$\begin{cases} \begin{pmatrix} -i & 1 \\ -1 & -i \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad \text{for } \lambda_1 = i \ \Rightarrow\ \mathcal{E}_1 = \begin{pmatrix} 1 \\ i \end{pmatrix} \\ \begin{pmatrix} i & 1 \\ -1 & i \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad \text{for } \lambda_2 = -i \ \Rightarrow\ \mathcal{E}_2 = \begin{pmatrix} 1 \\ -i \end{pmatrix} \end{cases}$$

We can confirm these are correct by noting that $DF(\bar{\vec{x}}) = \mathcal{E}D\mathcal{E}^{-1}$, i.e.,

$$\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ i & -i \end{pmatrix} \begin{pmatrix} i & 0 \\ 0 & -i \end{pmatrix} \begin{pmatrix} \frac{1}{2} & -\frac{i}{2} \\ \frac{1}{2} & \frac{i}{2} \end{pmatrix}$$

We note that the Wronskian of $\mathcal{E}$ is not zero, and so these solutions are linearly independent. We have solutions of the form $\vec{x} = \mathcal{E}_i e^{\lambda_i t}$, specifically,

$$\mathcal{X}_1 = \begin{pmatrix} 1 \\ i \end{pmatrix} e^{it}, \quad \mathcal{X}_2 = \begin{pmatrix} 1 \\ -i \end{pmatrix} e^{-it}$$

The Wronskian is,

$$\begin{vmatrix} e^{it} & e^{-it} \\ ie^{it} & -ie^{-it} \end{vmatrix} = -i - i = -2i$$

which is never zero, so the above solutions form a fundamental set of solutions, and our general solution is

$$\mathcal{X} = c_1\mathcal{X}_1 + c_2\mathcal{X}_2 = c_1 \begin{pmatrix} 1 \\ i \end{pmatrix} e^{it} + c_2 \begin{pmatrix} 1 \\ -i \end{pmatrix} e^{-it}$$

Professor: Dr. Medvedev
Student: Timothy Jones

In order to obtain more realistic solutions, we can take one of the solutions and take its real and imaginary values:

$$\mathcal{X}_1 = \begin{pmatrix} 1 \\ i \end{pmatrix} e^{it} = \mathcal{X}_1 = \begin{pmatrix} 1 \\ i \end{pmatrix} (\cos t + i \sin t) = \begin{pmatrix} \cos t \\ -\sin t \end{pmatrix} + i \begin{pmatrix} \sin t \\ \cos t \end{pmatrix}$$

Again we apply the Wronskian test:

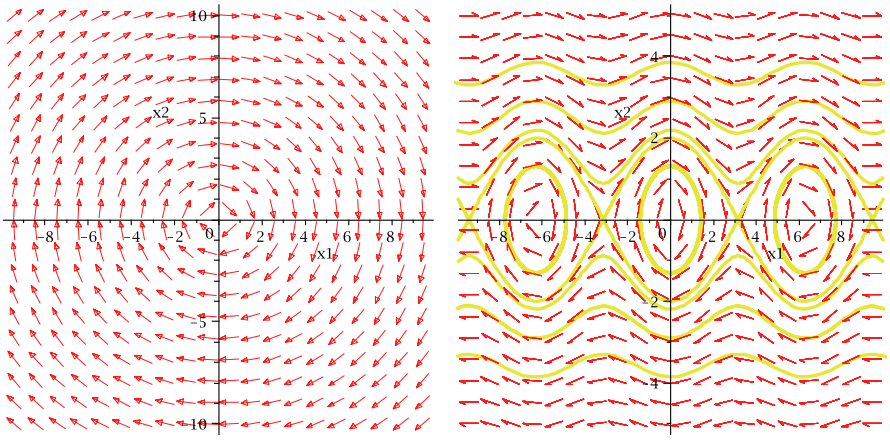
$$\begin{vmatrix} \cos t & \sin t \\ -\sin t & \cos t \end{vmatrix} = 1$$

Hence our general solution can now be written:

$$\mathcal{X} = \alpha_1 \begin{pmatrix} \cos t \\ -\sin t \end{pmatrix} + \alpha_2 \begin{pmatrix} \sin t \\ \cos t \end{pmatrix}$$

Hence we see that our fixed point is a "center". All that remains is for us to apply initial value conditions.

We can use this information to sketch the phase portraits, but instead we wish to use the method used in the previous section. With the linearized equations, we now have $U(x) = -\int_{x_0}^{x} \zeta d\zeta = \frac{-x^2+x_0^2}{2}$. The resulting graph will be quite similar to that of the original at the center, but in this case the potential is infinite and all "particles" would be trapped in a cycle.



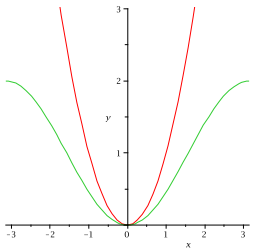
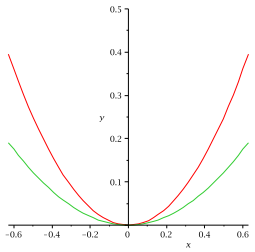
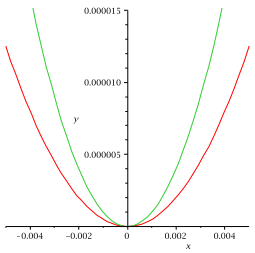
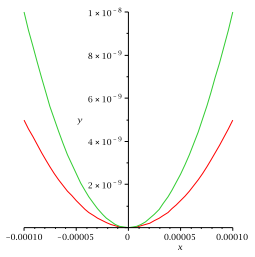





FIGURE 1.2. The linearized system (left) and the original system (right) are similar only locally for $\vec{\vec{x}} = (n\pi, 0)$, $n \in \mathbb{Z}$ even. In the margin we have compared the potential of the linearized system (red) and the original system (green) which are similar only locally for points very near $\vec{\vec{x}} = (0, 0)$.

The **odd case** follows similarly. As before we note that $DF(\vec{\vec{x}}) = \mathcal{E}D\mathcal{E}^{-1}$, i.e.,

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} \end{pmatrix}$$

where again the Wronskian is non-zero, affirming that we have a fundamental set of solutions. Furthermore, our case is made easier by the fact that these solutions are all real. Our solution thus takes the form,

$$\mathcal{X} = c_1\mathcal{X}_1 + c_2\mathcal{X}_2 = c_1 \begin{pmatrix} 1 \\ 1 \end{pmatrix} e^{t} + c_2 \begin{pmatrix} -1 \\ 1 \end{pmatrix} e^{-t}$$

This suggest an exponential growth in the direction of $(1, 1)$ and and exponential decay in the direction of $(-1, 1)$ which is indeed the case.

Finally, we include the maple code used to create these figures:

Professor: Dr. Medvedev
Student: Timothy Jones

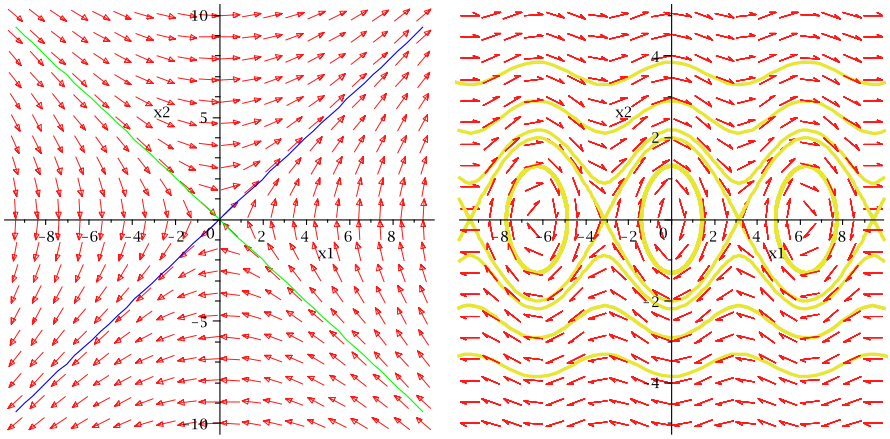


FIGURE 1.3. The linearized system (left) and the original system (right) are similar only locally for $\vec{\vec{x}} = (n\pi, 0)$, $n \in \mathbb{Z}$ odd.

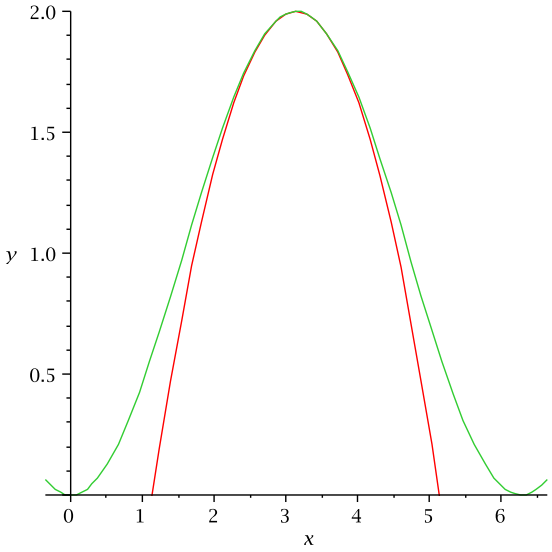


FIGURE 1.4. The linearized system (red) and the original system (green) are similar only locally for $\vec{\vec{x}} = (n\pi, 0)$, $n \in \mathbb{Z}$ odd. Here we have reset our potentials so that each match up: $1 - \cos(x)$ and $\left(\frac{\pi^2}{2} - \frac{(x-\pi)^2}{2}\right) - \frac{\pi^2}{2} + 2$.

```
> EQ1 := diff(x1(t), t) = x2(t); EQ2 := diff(x2(t), t) = -x1(t);
> with(DEtools); with(plots);
> v := 1; DEplot([EQ1, EQ2], [x1(t), x2(t)], t = 0 .. 10,
          x1 = -3*Pi .. 3*Pi, x2 = -10 .. 10, arrows = medium);
> EQ1 := diff(x1(t), t) = x2(t); EQ2 := diff(x2(t), t) = x1(t)
> plot01 := DEplot([EQ1, EQ2], [x1(t), x2(t)], t = 0 .. 10,
              x1 = -3*Pi .. 3*Pi, x2 = -10 .. 10, arrows = medium);
> plot02 := plot(x, x = -3*Pi .. 3*Pi, y = -10 .. 10, color = blue);
> plot03 := plot(-x, x = -3*Pi .. 3*Pi, y = -10 .. 10, color = green)
> display([plot01, plot02, plot03])
> plot({1-cos(x), (1/2)*Pi^2-(1/2)*(x-Pi)^2-(1/2)*Pi^2+2},
                              x = Pi-3.5 .. Pi+3.5, y = 0 .. 2)
```

Stability: Let the linearization of our system be written $\dot{x} = Ax$. We now need to apply the Lyapunov stability criteria for linearized systems. That is, if there exists two positive-definite matrices $P > 0$ and $Q > 0$ so that $A^T P + PA + Q = 0$,

then the system defined by A is stable. This is shown in many text, but we follow *Nonlinear Systems* by Shankar Sastry. The Lyaponov function we would use is $V(z) = x^T P x$. Lyaponov's direct method requires that $V(x)$ is a positive definite function and $\dot{V}(x) = \nabla V \cdot \mathbf{f} = \dot{V}(x(t))$ is a negative definite function. We have the requirement that $\frac{dV(x)}{dt} = \dot{x}^T P x + x^T P \dot{x} = (Ax)^T P x + x^T P (Ax) = x^T (A^T P + PA)x = x^T(-Q)x$, where, iff Q is positive definite, $\dot{V}$ is negative definite as required for stability.

It is shown that we can find a P given a Q, via the equation $P = \int_0^\infty (e^{A\tau})^T Q e^{A\tau} d\tau$. We will instead consider our two matrices in question in a simpler way.

Positive definite matrices of dimension two have, amongst other properties: $Q = Q^T$, $\forall\ x\ \in \mathbb{R}^n$, and $f(x) = x^T Q x$ is a positive definite function, i.e. $f(x) \geq 0$. Thus a positive definite function P has the form where

$$\begin{pmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \end{pmatrix}$$

where it can be shown that we require $p_{11} > 0$, $p_{12} = p_{21}$, $p_{22} > \frac{p_{12}^2}{p_{11}}$. Thus we have

$$\begin{aligned} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \longrightarrow A^T P + PA &= \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} p_1 & p_2 \\ p_2 & p_3 \end{pmatrix} + \begin{pmatrix} p_1 & p_2 \\ p_2 & p_3 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \\ &= \begin{pmatrix} 2p_2 & p_3 + p_1 \\ p_1 + p_3 & 2p_2 \end{pmatrix} = -Q \end{aligned}$$

In order for Q to be positive definite, we require that $-2p_2 > 0$ and $-2p_2 > \frac{(p_3+p_1)^2}{-2p_2}$, i.e.

$$Q = \begin{pmatrix} -2p_2 & -(p_1 + p_3) \\ -(p_1 + p_3) & -2p_2 \end{pmatrix}$$

But if Q is to be positive definite, as required for stability, we have that,

$$-2p_2 > \frac{(p_1 + p_3)^2}{-2p_2}, \quad \text{i.e.} \quad 2p_2 > p_1 + p_3$$

But we know that since P is itself positive definite, we require $p_1, p_3 > 0$, and so we have the coupled conditions that $2p_2 > p_1 + p_3 > 0$. If this is so, then q_1, the first entry of Q is negative, and so Q is not positive definite in contradiction to our assumption of stability.

We knew this also from the fact that this linearization had one positive eigenvalue (stability requires all eigenvalues be negative in their real parts).

Our other case is more quickly shown to not satisfy the Lyapunov conditions for stability. We have,

$$\begin{aligned} \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \longrightarrow A^T P + PA &= \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \begin{pmatrix} p_1 & p_2 \\ p_2 & p_3 \end{pmatrix} + \begin{pmatrix} p_1 & p_2 \\ p_2 & p_3 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \\ &= \begin{pmatrix} -2p_2 & -p_3 + p_1 \\ p_1 - p_3 & 2p_2 \end{pmatrix} = -Q \end{aligned}$$

We quickly see that if Q were to be positive definite, we can not have the diagonal elements to be of opposite signs or negative at all.

When we look at the eigenvalues, we find that the real parts are equal to zero for both. The Lyaponov tests do not treat this case, and as we have shown above, the system is a cycle.

Finally, out of curiosity, we show what a stable solution would look like using the Lyaponov conditions for linear systems. We consider the system whose linearized matrix is:

$$\begin{pmatrix} -1 & 0 \\ 0 & -2 \end{pmatrix} \longrightarrow A^TP + PA = \begin{pmatrix} -1 & 0 \\ 0 & -2 \end{pmatrix}\begin{pmatrix} p_1 & p_2 \\ p_2 & p_3 \end{pmatrix} + \begin{pmatrix} p_1 & p_2 \\ p_2 & p_3 \end{pmatrix}\begin{pmatrix} -1 & 0 \\ 0 & -2 \end{pmatrix}$$
$$= \begin{pmatrix} -2p_1 & -3p_2 \\ -3p_2 & -4p_3 \end{pmatrix} = -Q$$

The MATLAB lyap function solves the equation $AX + XA^T + Q = 0$, which is equivalent to the one we've been working with, $A^TP + PA + Q = 0$ since $Q^T = Q$ and and $P^T = P$ (they are positive definite) and so we can write $A \to A^T$ to obtain the correspondence we desire.

The routine is a function of the form:

```
X = lyap(A,Q)
```

For Q we chose the 2×2 positive-definite matrix

$$\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$$

The code found the corresponding $P = X$. We follow this up by testing the code with our unstable case.

Unstable case:

```
>>A=[0 1 ; 1 0]

A =

     0     1
     1     0

>>  Q=[1 1; 0 1]

Q =

     1     1
     0     1

>> X = lyap(A,Q)
??? Error using ==> sylvslv
Solution does not exist
or is not unique.
```

Stable Case:

```
>> A=[-1 0 ; 0 -2]

A =

    -1     0 \\
     0    -2

>> Q=[1 1; 0 1]

Q =

     1     1 \\
     0     1

>> X = lyap(A,Q)

X =

    0.5000    0.3333 \\
         0    0.2500

>> A'*X + X*A

ans =

    -1    -1 \\
     0    -1
```


Professor: Dr. Medvedev
Student: Timothy Jones

1.2.

Problem 2 is posted on the web (see Lecture 2, below Topics). Follow the steps described in this problem. After that, modify the parameters of the model and repeat all the steps. In your report, include your codes and the representative plots.

We present the the web-hand-out as an attachment at the end of this report. The results of following these instructions are show in the figure below.

The model used for this problem is that for a "nonlinear gain control in the retina":

$$\begin{aligned} \dot{B} &= \frac{-B + \frac{L}{1+A}}{\tau_b} \\ \dot{A} &= \frac{-A + 2B}{\tau_a}, \end{aligned}$$

where $\tau_a = 1$, $\tau_b = 1$, and $L = 1$. The initial conditions are $B = x(1)$ and $A = x(2)$. This is written in MATLAB as (from Dr. Medvedev's instructions):

```
function xdot=gain(t,x)

% This function codes the equations for the
% model of nonlinear gain control in the
% retina (Example 6, p. 37)

  tau_a=1; tau_b=1; L=1;

  B=x(1); A=x(2);

  Bdot=(-B + L/(1+A))/tau_b;
  Adot=(-A+2*B)/tau_a;

  xdot=[Bdot; Adot];
```

We compare this with Maple, where the above commands would appear:

```
tau_a:=1; tau_b:=1; L:=1;
EQ1 := diff(x1(t), t) = (-B + L/(1+A))/tau_b;
EQ2 := diff(x2(t), t) = (-A+2*B)/tau_a;
```

Now in the MATLAB command window we are to type: `diary save_your_work` which saves our work to a file called `save_your_work` after we have entered, later, `diary off`. After a perfunctory `clear all`, we are instructed to set the time span via `tspan=[0 50];` and set initial conditions `x0=[1 ; 0.5];`. Now having defined our differential equation, we are instructed to apply the numerical integration ode23 with the command `x[t,x]=ode23('gain', tspan, x0);`. This routine takes our differential equation, initial conditions, and our chosen time span for integration and returns the integrated values stored in t, a vector of discreet times, and a corresponding x which contains the pair of values (x1(t),x2(t)) for position and velocity. The following code prints out a plot of the velocity and position verses time:

```
B=x(:,1); % extract the first column with the values of B
A=x(:,2); % extract the second column for A
figure(1)  %  open a new figure
plot(t,B,'-b')      % plot B versus t

figure(2)  %  open another figure
```

```
plot(t,A)      % plot A versus t


figure(3)  % alternatively we can combine two plots

plot(t,B,'-b')      % plot B versus t
hold on              %
plot(t,A,'--r')     % plot x2 vs t
xlabel('time, t')   % supply lablels
ylabel('B (-), and A (--)')

print -deps batraces  % save this figure to file batraces.eps
```

We close our run with the command that will write our input: `diary off`. After forming another plot (along with nullclines–curves where the vector field is horizontal and vertical, these are found by finding where $\dot{A} = 0 \implies B = L/(1+A)$ and $\dot{B} = 0 \implies A = 2B$.).

Finally we are asked to the fixed points, which can be found via $-A + 2B = 0$ and $-B + L/(1 + A) = 0$:

```
>> b_bar=(-1+sqrt(1+8*1))/4; a_bar=2*b_bar;
>> DF=[-1 -1/(1+a_bar)^2; 2 -1];
>> [V, D]=eig(DF)
V =

        0 - 0.3333i        0 + 0.3333i
  -0.9428            -0.9428

D =

  -1.0000 + 0.7071i        0
        0            -1.0000 - 0.7071i

>> lambda_1=D(1,1)
lambda_2=D(2,2)
lambda_1 =

  -1.0000 + 0.7071i

lambda_2 =

  -1.0000 - 0.7071i

>> v1=V(:,1)
v2=V(:,2)
v1 =

        0 - 0.3333i
  -0.9428

v2 =

        0 + 0.3333i
  -0.9428

>> v1=v1/norm(v1); v2=v2/norm(v2);
```

Finally we modify the parameters: $\tau_a = 1.1$, $\tau_b = 1.2$, and $L = 1.3$. Having thus modified gain.m, we take our output from `save_your_work`, add the last few commands used in the instructions, and run the set of commands shown below.

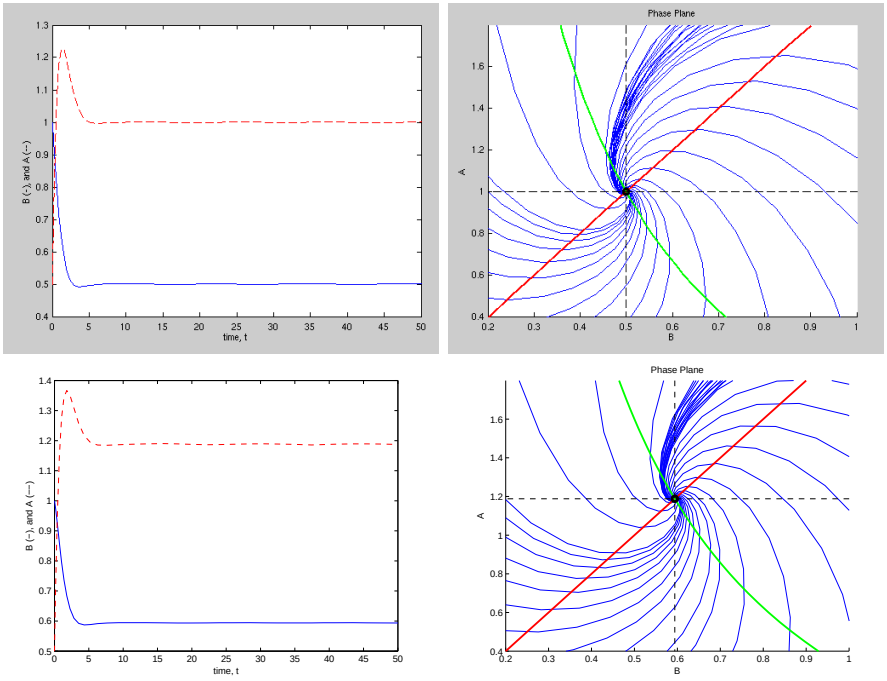


FIGURE 1.5. Output figures from following the directions given for problem two. The top row are with the original values, the bottom row are with the new values.

We present our final MATLAB worksheets below.

gain.m

```
function xdot=gain(t,x)

% This function codes the equations for the
% model of nonlinear gain control in the
% retina (Example 6, p. 37)

  tau_a=1.1; tau_b=1.2; L=1.3;

  B=x(1); A=x(2);

  Bdot=(-B + L/(1+A))/tau_b;
  Adot=(-A+2*B)/tau_a;

xdot=[Bdot; Adot];
```


Professor: Dr. Medvedev
Student: Timothy Jones

pln.m

```
figure(4)             % open another figure
hold on

tspan=[0 20];

for i=0.2:0.2:1.8

x0=[0; i];                      % defines initial conditions
[t, x]=ode23('gain', tspan, x0);
plot(x(:,1), x(:,2))
drawnow
[t, x]=ode23('gain', tspan, [i; 0]);
plot(x(:,1), x(:,2), '-b')
[t, x]=ode23('gain', tspan, [i; 2]);
plot(x(:,1), x(:,2), '-b')
[t, x]=ode23('gain', tspan, [2; i]);
plot(x(:,1), x(:,2), '-b')

drawnow
end

axis([0.2 1.0 0.4 1.8])

% Next, plot the nullclines

x1=0:0.05:2;
y1=2*x1;    % A-nullcline

plot( x1, y1, '-r', 'linewidth', 2) % plots A-nullcline
plot( 1.3./(1+x1), x1, '-g', 'linewidth', 2) % plots B-nullcline; note the dot
                                             % following in the expression of 1./(1+x1)
                                             % It indicates that the division in arrays
                                             % should be executed in term-by-term manner



% Compute the coorinates of the fixed point in the first quadrant
%

b_bar=(-1+sqrt(1+8*1.3))/4; a_bar=2*b_bar;

plot(b_bar, a_bar, 'ok', 'linewidth', 3) % indicate the fixed point on the plot


% Compute the Jacobian:

DF=[-1 -1/(1+a_bar)^2; 2 -1]; % ; indicates  end of row

%Find the eigenvalues and the eigenvectors of DF

[V, D]=eig(DF); %  eig is a matlab function which computes eigenvalues and eigenvectors
```


Professor: Dr. Medvedev
Student: Timothy Jones

```
                % see http://www.mathworks.com/access/helpdesk/help/techdoc
                %            /index.html?/access/helpdesk/help/techdoc/ref/eig.html

% The eigenvalues are complex. Thus, the eigenvectors can be taken complex conjugate
% We shall need the real and imaginary part of one of them

v1=real(V(:,1));
v2=imag(V(:,1));   % extract the eigenvectors

v1=v1/norm(v1); v2=v2/norm(v2);  % normalize

s=-1:1;

plot(b_bar+v1(1)*s, a_bar+v1(2)*s, '--k', 'linewidth', 1)
% In the present case the subspaces
plot(b_bar+v2(1)*s, a_bar+v2(2)*s, '--k', 'linewidth', 1)
% spanned by the eigenvectors are
% are not as informative as they would be
% if the eigenvalues were real. But we plot them anyway.


% Finally, add the axis lables and the title


xlabel('B')
ylabel('A')
title('Phase Plane')

print -deps pplane % save the figure
```

final form of `save_your_work`

```
clear all
tspan=[0 50];
x0=[1; 0.5];
[t,x]=ode23('gain', tspan, x0);
B=x(:,1);
A=x(:,2);
figure(1)
plot(t,B,'-b')
hold on
plot(t,A,'--r')
xlabel('time, t')
ylabel('B (-), and A (--)')
print -deps batraces
pln
b_bar=(-1+sqrt(1+8*1))/4; a_bar=2*b_bar;
DF=[-1 -1/(1+a_bar)^2; 2 -1];
[V, D]=eig(DF)
lambda_1=D(1,1)
lambda_2=D(2,2)
v1=V(:,1)
v2=V(:,2)
v1=v1/norm(v1); v2=v2/norm(v2);
```

Output from final form of `save_your_work`

```
V =

        0 - 0.3333i         0 + 0.3333i
  -0.9428             -0.9428


D =

  -1.0000 + 0.7071i         0
        0             -1.0000 - 0.7071i


lambda_1 =

  -1.0000 + 0.7071i


lambda_2 =

  -1.0000 - 0.7071i


v1 =

        0 - 0.3333i
  -0.9428


v2 =

        0 + 0.3333i
  -0.9428
```

1.3.

Write a matlab code for plotting the phase portrait for the equation of pendulum in Problem 1. Compare it with your solution of Problem 1. Include the code and the phase plane plot.

We are now to use MATLAB to create the phase portrait for the equation of the pendulum we addressed in problem one. Having worked through problem two, the solution to this problem follows quite closely. We present the resulting figure and the programs written for this figure below. As one would expect, our solution in MATLAB exactly matches that found in problem one.

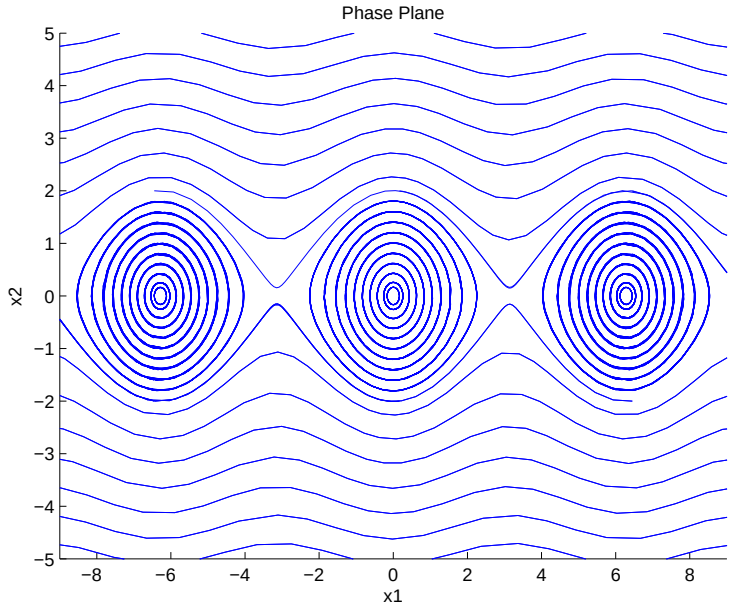


FIGURE 1.6. Phase portrait for pendulum problem of problem one, as made with MATLAB

prob3.m

```
function xdot=prob3(t,x)
xx=x(1);
yy=x(2);
xxdot=yy;
yydot=-sin(xx);
xdot=[xxdot; yydot];
```

prob3pp.m

```
figure(5)
hold on
tspan=[0 30];
for i=0:0.5:7
x0=[20;i];
[t,x]=ode23('prob3', tspan, x0);
plot(x(:,1),x(:,2))
drawnow
[t,x]=ode23('prob3', tspan, [20,-i]);
plot(x(:,1), x(:,2), '-b')
```

```
drawnow
[t,x]=ode23('prob3', tspan, [-20,i]);
plot(x(:,1), x(:,2), '-b')
drawnow
end
for i=-2:0.2:2
[t,x]=ode23('prob3', tspan, [2*pi+1/6,i]);
plot(x(:,1), x(:,2), '-b')
drawnow
[t,x]=ode23('prob3', tspan, [0*pi+1/6,i]);
plot(x(:,1), x(:,2), '-b')
drawnow
[t,x]=ode23('prob3', tspan, [-2*pi-1/6,i]);
plot(x(:,1), x(:,2), '-b')
drawnow
end
axis([-9 9 -5 5])
xlabel('x1')
ylabel('x2')
title('Phase Plane')
print -deps prob3pp
```

Finally in the command box we enter `prob3pp` and the figure is formed.

Note to self: command used to merge pdf files is:

```
gs -q -dNOPAUSE -dBATCH -sDEVICE=pdfwrite -sOutputFile=s.pdf \\
                                first.pdf hw1.pdf gain_control.pdf
```

Homework Two

Student: Timothy Jones
Math 723: Mathematics of Neuroscience
Professor: Medvedev
Spring 2008

2.1.

Plot nullclines for the ODEs (2.5.2 of Eckhoff and Holmes' notes), find all the equilibria and determine their stability types for the parameter values specified above (you may additionally take $\tau = 20$: how does this time constant affect stability?). Repeat for $K_1 = 130, K_2 = 110$ and for $K_1 = 150, K_2 = 90$ with the other parameters unchanged. In each case plot nullclines and solution trajectories for five different initial conditions. In the decision-making context, what is significant about the third case?

Let $S(P) = \frac{MP^N}{\sigma^N + P^N}$ for $P \geq 0$ and $S(P) = 0$ for $P < 0$. The ODE for this problem describes "a simple tow neuron network with the 'winner-take-all property"':

$$\begin{aligned} \frac{dE_1}{dt} &= \frac{1}{\tau}\left(-E_1 + S(K_1 - 3E_2)\right) \\ \frac{dE_2}{dt} &= \frac{1}{\tau}\left(-E_2 + S(K_2 - 3E_1)\right) \end{aligned}$$

E_1 represents the spike rate of neuron 1 receiving external input K_1, inhibited (negative sign) by the spike rate E_2 of neuron 2. This is:

$$\begin{aligned} \frac{dE_1}{dt} &= \frac{1}{\tau}\left(-E_1 + \frac{M(K_1 - 3E_2)^N}{\sigma^N + (K_1 - 3E_2)^N}\right) \\ \frac{dE_2}{dt} &= \frac{1}{\tau}\left(-E_2 + \frac{M(K_2 - 3E_1)^N}{\sigma^N + (K_2 - 3E_1)^N}\right) \end{aligned}$$

We can do a phase plot with maple:

```
> restart;
> with(DEtools); with(plots);
> K1 := 120; K2 := 120; M := 100; sig := 120; N := 2; tau := 20;
> EQ1 := diff(E1(t), t) = (-E1(t)+M*(K1-3*E2(t))^N/(sig^N+(K1-3*E2(t))^N))/tau;
> EQ2 := diff(E2(t), t) = (-E2(t)+M*(K2-3*E1(t))^N/(sig^N+(K2-3*E1(t))^N))/tau;
> DEplot([EQ1, EQ2], [E1(t), E2(t)], t = 0 .. 100, E1 = -10 .. 60,
                          \ E2 = -10 .. 60, title = 'Choice', arrows = medium);
```

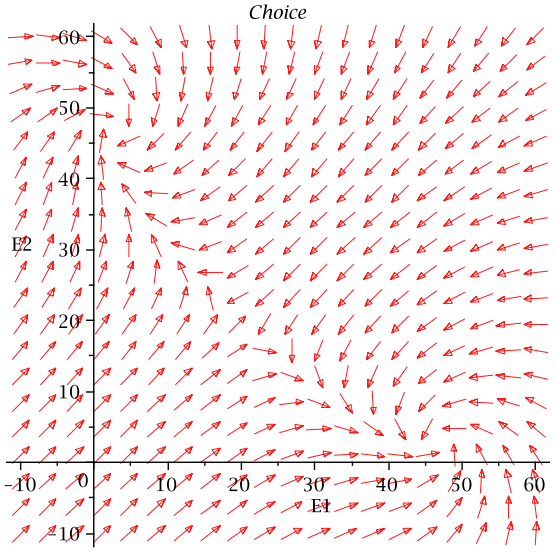


FIGURE 2.1. Phase portraits for this function as compiled in Maple. Here $K_1 = K_2 = 120$.

Professor: Dr. Medvedev
Student: Timothy Jones

We can plot these functions in Matlab as well:

```
function xdot=choice(t,x)

%Function for problem one of homework 2

K1=120; K2=120; M=100; sig=120; N=2; tau=20;

E1=x(1); E2=x(2);

E1dot=(-E1 + (M*(K1-3*E2)^N)/(sig^N + (K1-3*E2)^N))/tau;
E2dot=(-E2 + (M*(K2-3*E1)^N)/(sig^N + (K2-3*E1)^N))/tau;
xdot=[E1dot; E2dot];

////////////////////////////////////////////////////

clear all
tspan=[0 300];
x0=[1; 0.5];
[t,x]=ode23('choice', tspan, x0);
E1=x(:,1);
E2=x(:,2);
figure(1)
plot(t,E1,'-b')
hold on
plot(t,E2,'--r')
xlabel('time, t')
ylabel('E1 (-), and E2 (--)')
print -deps batraces
```

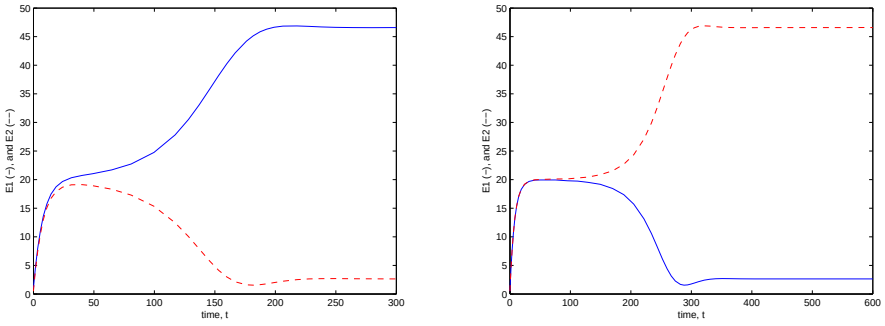


FIGURE 2.2. Matlab output with initial conditions $x_0 = (1, 0.5)$ (left) $x_0 = (0.49, 0.51)$ (right). The winner take all result is present, though when the initial conditions are close, the separation in results takes longer to achieve.

Now we wish to find its equilibria point. We disagree with the text that the fixed points are $(0, 50)$ and $(50, 0)$, though are results agree closely. We find fixed points at $(20, 20)$, $(46.58901055, 2.64175867)$, and $(2.64175867, 46.58901055)$:

Maple code to find the fixed points:

```
> with(LinearAlgebra):
> f := (x, y) --> (-x+M*(K1-3*y)^N/(sig^N+(K1-3*y)^N))/tau;
> g := (x, y) --> (-y+M*(K2-3*x)^N/(sig^N+(K2-3*x)^N))/tau;
> K1 := 120; K2 := 120; M := 100; sig := 120; N := 2; tau := 20;
> sol := solve([f(x, y) = 0, g(x, y) = 0], [x, y]);
> alias(R1 = RootOf(_Z^2-4*_Z+5, label = _L4));
> alias(R2 = RootOf(13*_Z^2-16*_Z+1, label = _L8));
> sol;
  [                                          [             640           ]]
  [[x = 20, y = 20], [x = 40 R1, y = 40 R1], [x = -40 R2 + ---, y = 40 R2]]
  [                                          [             13            ]]
> soln1 := (i) --> allvalues(R1)[i];
> soln2 := (i) --> allvalues(R2)[i];

> soln1(1);
                                    2 + I
> soln1(2);
                                    2 - I
> soln2(1); p := evalf(%);
                                8    1    (1/2)
                                -- - -- 51
                                13   13
                                 0.0660439670
> soln2(2); q := evalf(%);
                                8    1    (1/2)
                                -- + -- 51
                                13   13
                                  1.164725264
> evalf(-40*p+640/13);
                                  46.58901055
> evalf(40*p);
                                  2.641758680
> evalf(-40*q+640/13);
                                  2.64175867
> evalf(40*q);
                                  46.58901056
```

Just by looking at the phase plot we can guess that the fixed point at (20,20) is a saddle, and the the other two are spiral sinks. However, this isn't good enough, and we do a linearization analysis as follows (in Maple):

```
 > restart; K1 := 120; K2 := 120; M := 100; sig := 120; N := 2; tau := 20;
> v1 := (-x+M*(K1-3*y)^N/(sig^N+(K1-3*y)^N))/tau;
> v2 := (-y+M*(K2-3*x)^N/(sig^N+(K2-3*x)^N))/tau;
> with(LinearAlgebra); with(linalg);
> v := Vector(2, [v1, v2]);
> Vector[column](%id = 21087208)
> jacobian(v, [x, y]);
%Here I extracted the functions of the jacobian matrix for more control:
> mtopright := (y) -> -(3600-90*y)/(14400+(120-3*y)^2)-5*(120-3*y)^2*(-720+18*y)
                                                                  \ /(14400+(120-3*y)^2)^2;
> mbotleft := (x)-> -(3600-90*x)/(14400+(120-3*x)^2)-5*(120-3*x)^2*(-720+18*x)
```


Professor: Dr. Medvedev
Student: Timothy Jones

```
                                                        \ /(14400+(120-3*x)^2)^2;
> alph := matrix([[-1/20, mtopright(46.58901056)], [mbotleft(2.614175867), -1/20]]);
> bet := matrix([[-1/20, mtopright(2.614175867)], [mbotleft(46.58901056), -1/20]]);
> eigenvalues(alph);
     -0.05000000000 + 0.05097393436 I, -0.05000000000 - 0.05097393436 I
> eigenvalues(bet);
     -0.05000000000 + 0.05097393436 I, -0.05000000000 - 0.05097393436 I
> gam := matrix([[-1/20, mtopright(20)], [mbotleft(20), -1/20]]);
> eigenvalues(gam);
                                  3   -13
                                 ---, ---
                                 100  100
```

We find that the fixed point at the center has one positive eigenvalue and one negative eigenvalue. It is thus hyperbolic and obeys the Hartman-Grobman theorem. We can thus conclude that this point is a saddle point.

The other two fixed points are also hyperbolic. There Jacobians numerically work out to be:

$$\begin{pmatrix} -\frac{1}{20} & 0.039034 \\ -0.06656 & -\frac{1}{20} \end{pmatrix}, \begin{pmatrix} -\frac{1}{20} & -0.06656 \\ 0.039034 & -\frac{1}{20} \end{pmatrix}$$

There eigenvalues work out to be both

$$-0.05000000000 + 0.05097393436i, -0.05000000000 - 0.05097393436i$$

Thus the eigenvectors will work out to be similar, but with opposite orientation. We can see this clearly on the phase portrait plotted earlier. Since the real values of the eigenvalues are negative, both points are attracting stable points.

Next we are to find the nullclines for this function. We use both Maple and Matlab for this. In Maple we plot the nullclines against the phase portrait, and in Matlab we plot the nullclines and numerous (more than five) solution trajectories. In Maple we have:

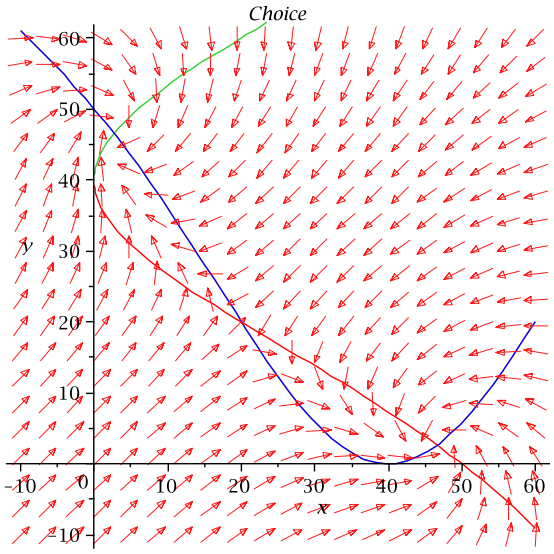


FIGURE 2.3. Nullclines as compiled in Maple.

Professor: Dr. Medvedev
Student: Timothy Jones

```
> restart; K1 := 120; K2 := 120; M := 100; sig := 120; N := 2; tau := 20;
> a := (x, y) -> (-x+M*(K1-3*y)^N/(sig^N+(K1-3*y)^N))/tau;
> b := (x, y) -> (-y+M*(K2-3*x)^N/(sig^N+(K2-3*x)^N))/tau;
> solve(a(x, y) = 0, y);
> solyleft := op(1, op(%)); solve(a(x, y) = 0, y); solyright := op(2, op(%));
> solve(b(x, y) = 0, x);
> solxleft := op(1, op(%)); solve(b(x, y) = 0, x); solxright := op(2, op(%));
> solve(solxright = x, y);
> solve(solxleft = x, y);
> fig1 := plot([solyleft, solyright, (100*(-80*x+1600+x^2))/(3200-80*x+x^2),
                (100*(-80*x+1600+x^2))/(3200-80*x+x^2)], x = -10 .. 60, y = -10 .. 60);
> with(DEtools); with(plots);
> EQ1 := diff(E1(t), t) = (-E1(t)+M*(K1-3*E2(t))^N/(sig^N+(K1-3*E2(t))^N))/tau;
> EQ2 := diff(E2(t), t) = (-E2(t)+M*(K2-3*E1(t))^N/(sig^N+(K2-3*E1(t))^N))/tau;
> fig2 := DEplot([EQ1, EQ2], [E1(t), E2(t)], t = 0 .. 100, E1 = -10 .. 60,
                                  E2 = -10 .. 60, title = 'Choice', arrows = medium);
> display([fig1, fig2]);
```

In Matlab we produce the following plot:

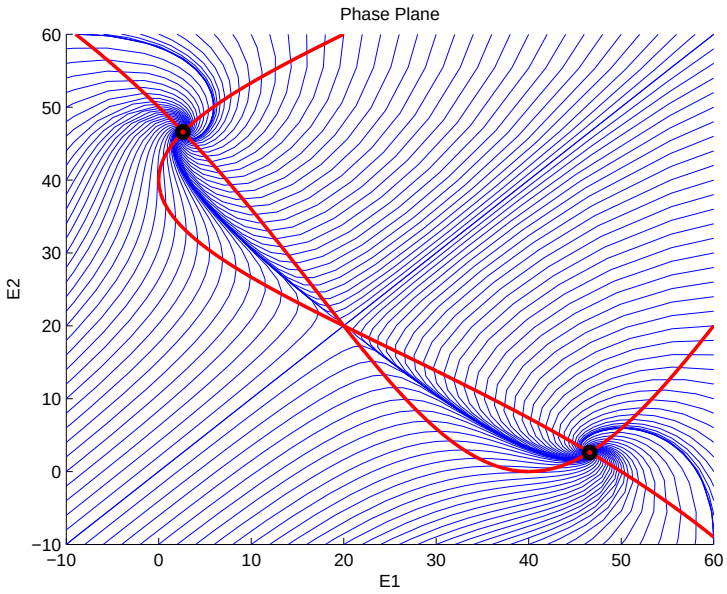


FIGURE 2.4. Nullclines as compiled in Maple.

```
figure(4)              % open another figure
hold on
tspan=[0 160];
for i=-10:2:60
x0=[-10; i];                        % defines initial conditions
[t, x]=ode23('choice', tspan, x0);
plot(x(:,1), x(:,2))
drawnow
[t, x]=ode23('choice', tspan, [i; -10]);
plot(x(:,1), x(:,2), '-b')
[t, x]=ode23('choice', tspan, [i; 60]);
```


Professor: Dr. Medvedev
Student: Timothy Jones

```
plot(x(:,1), x(:,2), '-b')
[t, x]=ode23('choice', tspan, [60; i]);
plot(x(:,1), x(:,2), '-b')
drawnow
end
axis([-10 60 -10 60])
% Next, plot the nullclines
x1=0:0.05:60;
y1=40*(x1-100+sqrt(x1*100-x1.^2))./(x1-100);    % dotx=0-nullcline
y2=40*(x1-100-sqrt(x1*100-x1.^2))./(x1-100);
plot( x1, y1, '-r', 'linewidth', 2) % plots dotx-nullcline
plot( x1, y2, '-r', 'linewidth', 2) % plots dotx-nullcline
% The result for the other case (doty) is just the interpose of x<->y
plot( y1, x1, '-r', 'linewidth', 2) % plots doty-nullcline
plot( y2, x1, '-r', 'linewidth', 2) % plots doty-nullcline
% Compute the coorinates of the fixed point in the first quadrant
plot(2.614175867, 46.5890156, 'ok', 'linewidth', 3) % indicate the fixed point
plot(46.5890156, 2.614175867, 'ok', 'linewidth', 3) % indicate the fixed point
% Finally, add the axis lables and the title
xlabel('E1')
ylabel('E2')
title('Phase Plane')
print -deps pplane % save the figure
```

We are now to repeat this analysis for two more cases. Our next case is $K_1 = 130$ and $K_2 = 110$.

```
> restart; with(LinearAlgebra); with(linalg);
> K1 := 130; K2 := 110; M := 100; sig := 120; N := 2; tau := 20;
> f := (x, y)-> (-x+M*(K1-3*y)^N/(sig^N+(K1-3*y)^N))/tau;
> g := (x, y)-> (-y+M*(K2-3*x)^N/(sig^N+(K2-3*x)^N))/tau;
> sol := solve([f(x, y) = 0, g(x, y) = 0], [x, y]);
> alias(R1 =
    RootOf(-5294258+12995789*_Z-7876248*_Z^2+1901718*_Z^3-200286*_Z^4+8181*_Z^5,
                                                                    label = _L2));
> op(2, op(1, op(sol)));
> op(2, op(2, op(sol)));
> funcx := (z) ->
    -(1318475/93528)*z-(179645/62352)*z^2+1304765/23382-(505/6928)*z^4+(5485/5196)*z^3;
> funcy := (z) -> 10*z;
> soln1 := (i)->allvalues(R1)[i];
> for i to 5 do a := evalf(soln1(i)); x = funcx(a); y = funcy(a) end do;
```

This results in an equation with five roots, only three of which are real. The complex roots do not interest us for this current problem.

We apply the linearization procedure, again using Maple; here we only show the results-the procedure was the same as :

```
> restart; K1 := 130; K2 := 110; M := 100;
...
> alph := matrix([[-1/20, mtopright(5.902851517)],
                  [mbotleft(46.68516637), -1/20]]);
> gam := matrix([[-1/20, mtopright(45.32481166)],
                    [mbotleft(.24726129), -1/20]]);
> bet := matrix([[-1/20, mtopright(29.35629200)],
```

The results:
a = 0.5902851517
x = 46.68516637
y = 5.902851517
a = 2.935629200
x = 10.88126817
y = 29.35629200
a = 4.532481166
x = 0.24726129
y = 45.32481166
a =
8.211726333 + 3.868115832i
x =
77.7983097 + 41.4277878i
y =
82.11726333 + 38.68115832i
a =
8.211726333 - 3.868115832i
x =
77.7983097 - 41.4277878i
y =
82.11726333 - 38.68115832i

Professor: Dr. Medvedev
Student: Timothy Jones

```
                    [mbotleft(10.88126817), -1/20]]);
> eigenvalues(alph);
     -0.05000000000 + 0.06071815859 I, -0.05000000000 - 0.06071815859 I
> eigenvalues(bet);
                          0.02469938912, -0.1246993891
> eigenvalues(gam);
     -0.05000000000 + 0.02903015478 I, -0.05000000000 - 0.02903015478 I
```

Again, the outer fixed points are spiral sinks; the central point is a saddle point.

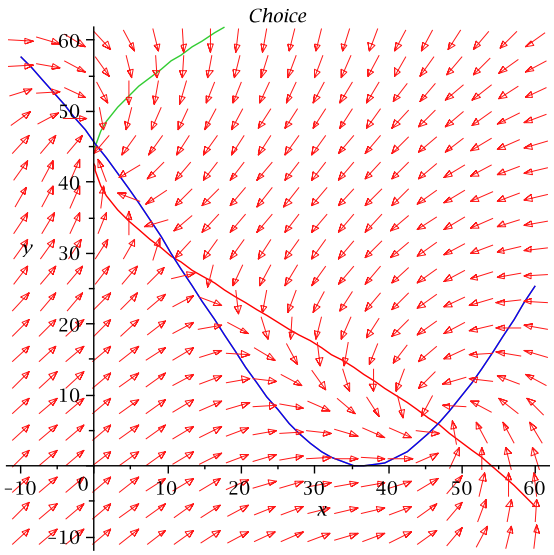


FIGURE 2.5. Nullclines as compiled in Maple.

The mapping of the nullcline in Matlab procedes as in the previous case.

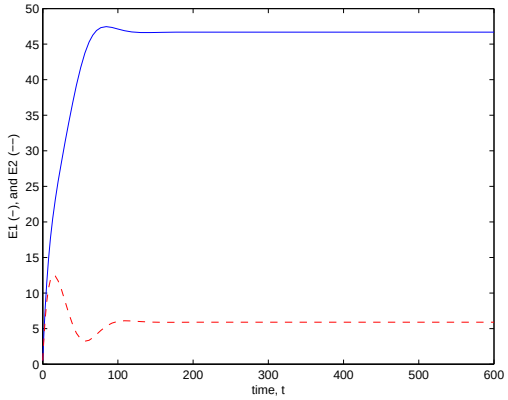


FIGURE 2.6. Function plot as compiled in Matlab. Initial conditions: (0.51,0.49)

Professor: Dr. Medvedev
Student: Timothy Jones

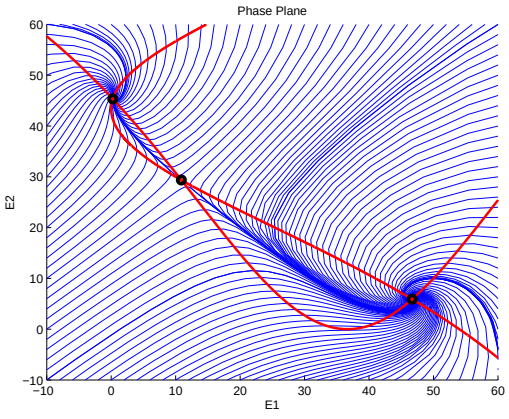


FIGURE 2.7. Nullclines as compiled in Matlab.

Our last case is $K_1 = 150$ and $K_2 = 90$. The result for this problem is that we only find one non-complex fixed point at $(45.67963035, 13.31910503)$. The Jacobian for this fixed point works out to be:

$$\begin{pmatrix} -\frac{1}{20} & -0.067646 \\ 0.073631 & -\frac{1}{20} \end{pmatrix}$$

with eigenvalues:

$$-0.05000000000 + 0.07057550471i, -0.05000000000 - 0.07057550471i$$

resulting in a single spiral sink.

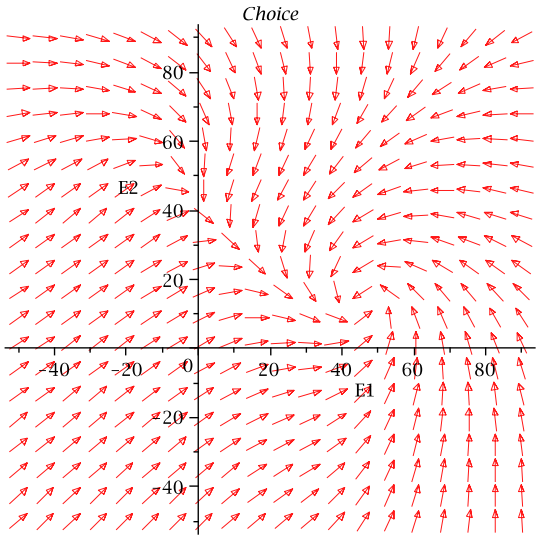


FIGURE 2.8. Phase portrait as compiled in Maple.

This last case causes one neuron to completely dominate the other:

Professor: Dr. Medvedev
Student: Timothy Jones

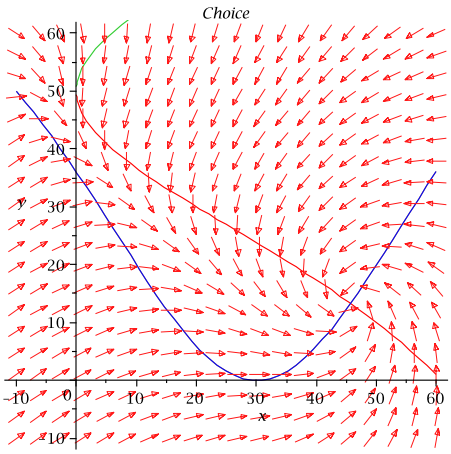


FIGURE 2.9. Nullclines as compiled in Maple.

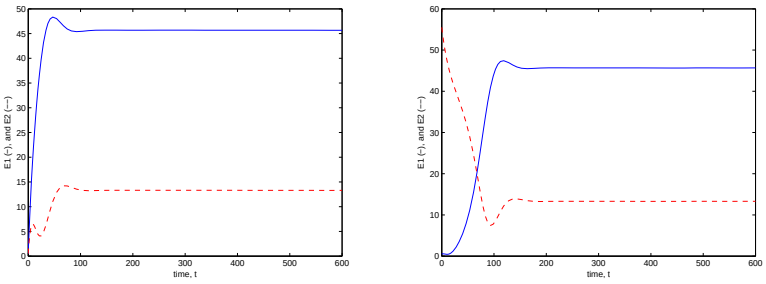


FIGURE 2.10. Nullclines as compiled in Matlab. Left: Initial condtions: (0.51,0.49). Right: Initial condtions (0.51,55.51), yet E_1 still dominates after a short time period.

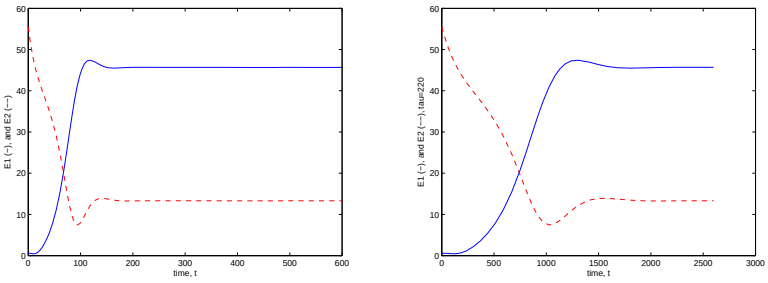


FIGURE 2.11. Nullclines as compiled in Matlab. Initial conditions: (0.51,55.51). $\tau = 20$ on the left and $\tau = 220$ on the right. As can be seen, τ's effect is to regulate the rate at which equilibrium is achieved.

Professor: Dr. Medvedev
Student: Timothy Jones

2.2.

a. Consider $\mathbf{f}(\mathbf{x}) = \begin{cases} -y - x^3 \\ x - y^3 \end{cases} \quad (1)$

As in the previous homework, linearization produces a Jacobian which is non-hyperbolic (it's eigenvalue's real parts are zero) at the fixed point $(0,0)$:

$$\mathbf{J}(\mathbf{f}(\mathbf{x})) = \begin{pmatrix} -2x^2 & -1 \\ 1 & -3y^2 \end{pmatrix}\Big|_{(0,0)} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$$

Thus the Hartman-Grobman theorem does not apply, and we can not use linearization to determine the stability of this function. For the sake of curiosity, we temporarily following the linearization path. We have the following eigenvalue and eigenvector combination:

Linearization results will only give us something qualitatively suggestive of the local behavior, but not necessarily quantitatively accurate.

$$\begin{cases} \begin{pmatrix} -i & -1 \\ 1 & -i \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad \text{for } \lambda_1 = i \;\Rightarrow\; \mathcal{E}_1 = \begin{pmatrix} 1 \\ -i \end{pmatrix} \\ \begin{pmatrix} i & -1 \\ 1 & i \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad \text{for } \lambda_2 = -i \;\Rightarrow\; \mathcal{E}_2 = \begin{pmatrix} 1 \\ i \end{pmatrix} \end{cases}$$

We will not go into much detail here (this was covered fully in the previous homework), but our result will be similar.

In order to obtain more realistic solutions, we can take one of the solutions and take its real and imaginary values:

$$\mathcal{X}_1 = \begin{pmatrix} 1 \\ i \end{pmatrix} e^{-it} = \mathcal{X}_1 = \begin{pmatrix} 1 \\ i \end{pmatrix}(\cos t - i \sin t) = \begin{pmatrix} \cos t \\ \sin t \end{pmatrix} + i \begin{pmatrix} -\sin t \\ \cos t \end{pmatrix}$$

We apply the Wronskian test:

The Wronskian tests for linear independence of solutions.

$$\begin{vmatrix} \cos t & -\sin t \\ \sin t & \cos t \end{vmatrix} = 1$$

Hence our general solution can now be written:

$$\mathcal{X} = \alpha_1 \begin{pmatrix} \cos t \\ \sin t \end{pmatrix} + \alpha_2 \begin{pmatrix} -\sin t \\ \cos t \end{pmatrix}$$

Hence we see that our fixed point is a "center". All that remains is for us to apply initial value conditions.

We can compare this to the solution for the cycle in the previous homework and note that the difference will be that this cycle flows counterclockwise, whereas in the previous problem, in which the Jacobian had the opposite sign, the flow was clockwise.

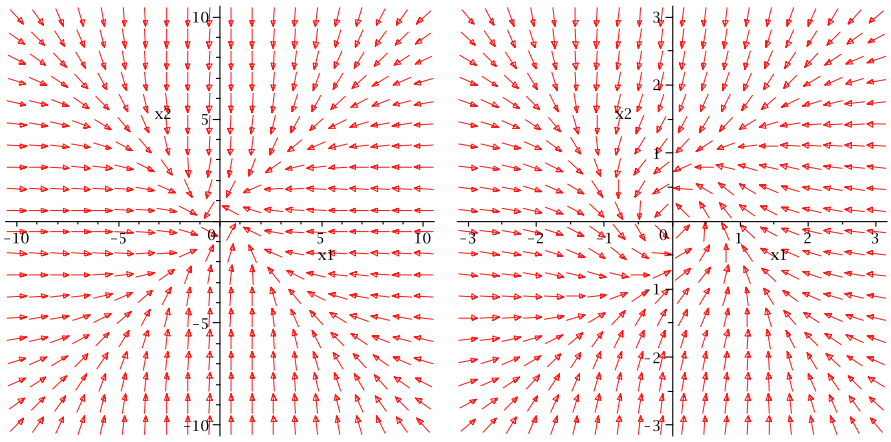


FIGURE 2.12. Phase portraits for the full function of this problem (close up of equilbrium point shown in right figure).

Professor: Dr. Medvedev
Student: Timothy Jones

Using the same Maple codes as in the previous problem, we compute the phase portraits for this system (in full, not the linearization) and find that the equilibrium at the center is indeed a "center", and seems to be attracting. That it is indeed attracting will be shown via the Lyapunov test. Now we are to show that $V(x, y) =$

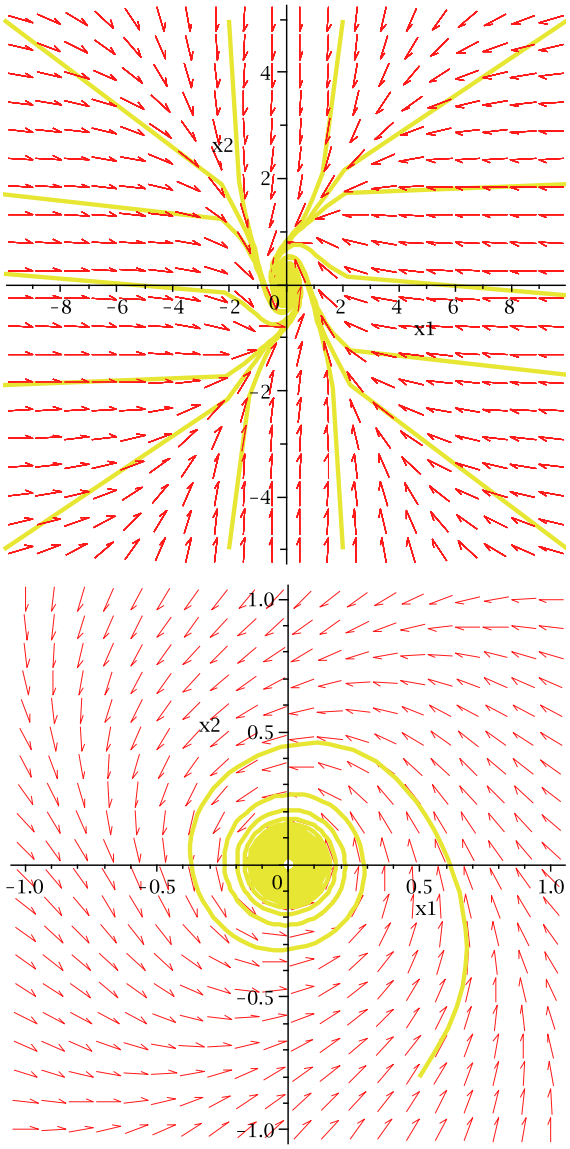


FIGURE 2.13. Phase portraits for the full function of this problem.

$\frac{1}{2}\left(x^2 + y^2\right)$ is a Lyapunov function. In order for $V(x, y)$ to be a Lyapunov function for this particular system, we require that

$$\frac{dV}{dt} = \nabla V \cdot \mathbf{f} \leq 0 \text{ (is positive definite) on some domain } D \ni \mathbf{0}$$

We have $\nabla V = x\mathbf{i} + y\mathbf{j}$ and so

$$\frac{dV}{dt} = (x\mathbf{i} + y\mathbf{j}) \cdot \left((-y - x^3)\mathbf{i} + (x - y^3)\mathbf{j}\right) = -xy - x^4 + xy - y^4 = -(x^4 + y^4) \leq 0 \ \forall x, y$$

We are thus assured that it is asymptotically stable (and Liapunov stable).

Professor: Dr. Medvedev
Student: Timothy Jones

b. Follow the same procedure for the Lorenz system.

The Lorenz system is given by the equation,

$$\mathbf{f}(\mathbf{x}) = \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{pmatrix} = \begin{pmatrix} \sigma(y-x) \\ rx - y - xz \\ xy - bz \end{pmatrix}$$

To prove stability for $0 \leq r \leq 1$, we seek a **Lyapunov function** in the form:

$$L(x, y, z) = \alpha x^2 + \beta y^2 + \gamma z^2$$

This functions needs to be positive definite, whereby we require $\alpha, \beta, \gamma, \sigma, r, b > 0$. We have further requirements in that it also must satisfy the condition that dV/dt is negative definite, that is, we require:

$$\frac{dV}{dt} = \nabla V \cdot \mathbf{f} \leq 0$$

We have:

$$\nabla V = 2\alpha x\mathbf{i} + 2\beta y\mathbf{j} + 2\gamma z\mathbf{k}$$

Thus:

$$\frac{dV}{dt} = \left(2\alpha\sigma xy - 2\alpha\sigma x^2 + 2\beta yrx - 2\beta y^2 - 2\beta yxz + 2\gamma zxy - 2\gamma bz^2\right)$$

We must show that $\dot{V} \leq 0$ always in order to prove stability. We reorganize the above as

$$\frac{dV}{dt} = -\left(2\alpha\sigma x^2 + 2\beta y^2 + 2\gamma bz^2\right) + (2\alpha\sigma + 2\beta r)xy + (2\gamma - 2\beta)xyz$$

We first attempt to simplify by letting $\gamma = \beta$ so that,

$$\frac{dV}{dt} = -\left(2\alpha\sigma x^2 + 2\beta y^2 + 2\beta bz^2\right) + (2\alpha\sigma + 2\beta r)xy$$

We which to sync the x and y terms so that we can write the above result in terms of the negative of a square. We are thus motivated to let $\alpha = 1$ and $\beta = \sigma$ whereby,

$$\frac{dV}{dt} = -2\sigma\left(x^2 + y^2 - (1+r)xy\right) - 2\sigma bz^2$$

For $r = 1$,

$$\frac{dV}{dt} = -2\sigma\left(x - y\right)^2 - 2\sigma bz^2 \leq 0$$

and we are done. For $0 \leq r < 1$, we have that $(1 + r) < 2$ and for $xy > 0$,

$$x^2 + y^2 - (1+r)xy > x^2 + y^2 - 2xy = (x-y)^2 > 0$$

whereas if $xy < 0$ then the quantity in question is positive anyway and we have shown stability.

We can use **linearization** to show the stability or instability for the Lorenz system in terms of $r \neq 1$ as follows. The Jacobian for the Lorenz system is:

$$D(f(\mathbf{x})) = \begin{pmatrix} -\sigma & \sigma & 0 \\ r - (z=0) & -1 & -x = 0 \\ y = 0 & x = 0 & -b \end{pmatrix}$$

We use the algebraic program Maple to compute the eigenvalues (which we could do on paper, but Maple promises accuracy):

$$\mathbf{E} = \{\frac{1}{2}\left(\sqrt{1 - 2\sigma + \sigma^2 + 4\sigma r} - \sigma - 1\right),\ -\frac{1}{2}\left(\sigma + 1 + \sqrt{1 - 2\sigma + \sigma^2 + 4\sigma r}\right), -b\}$$

Here it is clear that all the real-parts are nonzero and thus we have a hyperbolic case for $0 \leq r < 1$ (and as long as b is nonzero). At $r = 1$ the real parts vanish for the first two cases since $\sqrt{1 - 2\sigma + \sigma^2 + 4\sigma} = \sqrt{(1+\sigma)^2} = (1 + \sigma)$. However, we

have used a Lyapunov function to demonstrate stability for $r = 1$. We also see that for $r < 1$, the eigenvalues are all negative, indicating the existence of asymptotic stability. The first eigenvalue would be the only one we need worry about being positive, but we note that the first eigenvalue can be written:

$$\frac{1}{2}\left(\sqrt{\sigma^2 - 2\sigma + 1 + 4\sigma r} - (\sigma + 1)\right)$$

and since

$r < 1$	$\sigma^2 - 2\sigma + 1 + 4\sigma r < \sigma^2 + 2\sigma + 1 = (\sigma + 1)^2$
$r > 1$	$\sigma^2 - 2\sigma + 1 + 4\sigma r > \sigma^2 + 2\sigma + 1 = (\sigma + 1)^2$

we see that we asymptotic stability for $r < 1$ and instability for $r > 1$. It is clear that a bifurcation occurs at $r = 1$, and our next task is to identify this. We read in Dr. Medvedev's notes that:

> Two simplest (and typical) cases when the equilibrium becomes non-hyperbolic are when one (and only one) real eigenvalue becomes 0 and when a pair of complex conjugate eigenvalues crosses the imaginary axis. The first case, under some additional conditions, is called a saddle-node (SN) bifurcation and the second case is called an Andronov-Hopf (AH) bifurcation.

The bifurcation here is Sadle-Node.

2.3.

Follow the steps of analysis of the saddle node bifurcation in the handout (section 4.1) to discuss the transcritical and pitchfork bifurcations for $\dot{x} = rx - x^2$ and $\dot{x} = rx - x^3$ respectively.

We briefly summarize the analysis of the handout. We consider a one-parater family of flows in $\mathbb{R}^1$ of the form $\dot{x} = f(x, \mu), x \in \mathbb{R}^1$ under the additional (though unnecessary) assumption that $f(0, 0) = 0$. Further requirements are that $f_x(0, 0) = 0$ (giving that for $\mu = 0$, the equilibirum at $x = 0$ is nonhyperbolic), $f_{xx}(0, 0) \neq 0$ and $f_\mu(0, 0) \neq 0$ for nondegeneracy (verifices the family of vecotr fields is typical [generic]) and transversality (forcing the function to pass through the bifurcation).

The Taylor expansion of our dynamical equation gives:

$$\begin{aligned}\dot{x} &= f(0,0) + f_x(0,0)x + f_\mu(0,0)\mu + f_{xx}(0,0)x^2 + f_{x\mu}(0,0)x\mu + f_{\mu\mu}(0,0)\mu^2 + \mathcal{O}(3) \\ &= a\mu + bx^2 + cx\mu + d\mu^2 + \mathcal{O}(3)\end{aligned}$$

Assume $a > 0$ and $b > 0$ and set the right hand side of the above equation to zero to find the equilibria for small values of μ and x. Taking the lowest order terms, we have:

$$bx^2 = -a\mu + h.o.t. \rightarrow x_0^{\pm}(\mu) = \pm\sqrt{\frac{-a\mu}{b}} + h.o.t.$$

where obviously we would need $\mu \leq 0$. Dr. Medvedev writes "A pair fo equilbiria which exists for negative μ ($|\mu|$ small) collides at $\mu = 0$ and disappears as μ becomes positive."

The stability of the equilibria is determined as follows:

$$f'_x(x_0^{\pm}, \mu) = 2bx^{\pm}(\mu) + h.o.t. = \pm 2b\sqrt{\frac{a\mu}{b}} + h.o.t = \pm 2\sqrt{ba\mu} + h.o.t.$$

It is seen here that $x_0^+(\mu)$ is unstable and $x_0^-(\mu)$ is stable for $\mu < 0$.

Now we look at the **transcritical case**, $\dot{x} = \mu x - x^2 + h.o.t.$ In this case we would have a result along the lines of $\dot{x} = \mu x - x^2 + h.o.t.$ and so our equilibria condition is $x = 0$ and $x = \mu$. Here the equilibria doesn't collapse after collision. We see that

$$f'_x(x_0^{0,\mu}, \mu) = \mu - 2x$$

Stability for the $x = 0$ case changes with the sign of μ, i.e. goes from stable ($\mu < 0$) to unstable for ($\mu > 0$). The case $x = \mu$ has the opposite case. Thus stability is switched "transcritically".

For the **pitchfork** case we have $\dot{x} = \mu x - x^3 + h.o.t. \rightarrow x_0 = 0, \sqrt{\mu} : \mu > 0$. For $\mu < 0$ there is only one fixed point $x_0 = 0$. At $\mu = 0$ we have a split into three equilibria points. To find stability we consider: $f' = \mu - 3x^2 + h.o.t.$ Obviously for the $x_0 = 0$ case the stability changes with the sign of μ, i.e. it is stable for $\mu < 0$ and unstable for $\mu > 0$. For the other two cases we have $f' = \mu - 3(\pm\sqrt{\mu})^2 = -2\mu$ and both are stable.

Professor: Dr. Medvedev
Student: Timothy Jones

2.4.

The matlab code for the numerical experiments with the HH system of the previous subsection is given in the Appendix to this lecture. Modify this code to study the voltage responses in the ML system. For the ML system, use (V0, n0) = (60.855, 0.01495) as the initial condition. Find the values for the amplitude and the duration of stimulation which yield sub- and superthreshold responses, as well as trains of AP. Repeat this numerical experiment for $\phi = 0.02$. Describe the effect of changing ϕ on the trains of AP generated 10 for prolonged stimulation. To make sure that you entered the parameters correctly, compare your numerics with that in Figure 6 for the same values of parameters.

Our modified Matlab code is:

```
function ML_stimulate(intensity,duration,delaytime)

global howstrong howlong delay

howstrong = intensity;  % intensity of applied current
howlong = duration;     % duration of applied current
delay = delaytime;      % delaytime (or beginning time) of applied current

T_MAX = 250;
step = 0.05;
tspan=0:step:T_MAX;

x0 =[-60.855,0.01495];     % steady state values after transient time 10ms
%x = [v n m h];

[t,x] = ode15s(@ml_syst, tspan, x0);

v=x(:,1);
n=x(:,2);

figure(1)

subplot(2,1,1)
plot(t,v);
axis([0 T_MAX -100 50]);
title(sprintf('ML MODEL with I (applied current)', num2str(howstrong)))
ylabel ('V (mV)')


for i=1:length(t)
    current(i)=inp(t(i));
end

subplot(2,1,2)
plot(t, current)
axis([0 T_MAX -100 max(current)+1]);
xlabel ('t (ms)')
ylabel ('I (\muA/cm^2)')
```


Professor: Dr. Medvedev
Student: Timothy Jones

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function xdot = ml_syst(t,x)

I = inp(t);

v=x(1);
n=x(2);

gca=4.4;
gl=2
v2=18
Eca=120;
El=-60;
v3=2;
gk=8;
C=20;
v4=30;
Ek=-84;
v1=-1.2;
phi=0.04;

m=0.5*(1 + tanh((v-v1)/v2));
ninf=0.5*(1 + tanh((v-v3)/v4));
tau=1/(cosh((v-v3)/(2*v4)));

dv=(-gca*m*(v-Eca) - gk*n*(v-Ek)-gl*(v-El)+I)/C;
dn=phi*((ninf-n)/tau);

xdot=[dv dn]';  %column vector: '


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function I=inp(t)

global howstrong howlong delay

I = 100;

t_end  = delay+howlong;

if (t >= delay) & (t<=t_end)
   I = howstrong;
end;
```

For intensity = 10, duration = 1, and delay = 5 we plot v against t to make sure that it resembles Figure 6 of the handout (though since we don't know the exact intensity/duration/delay used to produce figure six, they won't be exactly the same).

We find that for $I = 0$, $I = 75$ and $I = 100$ we find null, subthreshold, and superthreshold behavior (trains of AP) respectively, in agreement with Figure 6 of the hand out.

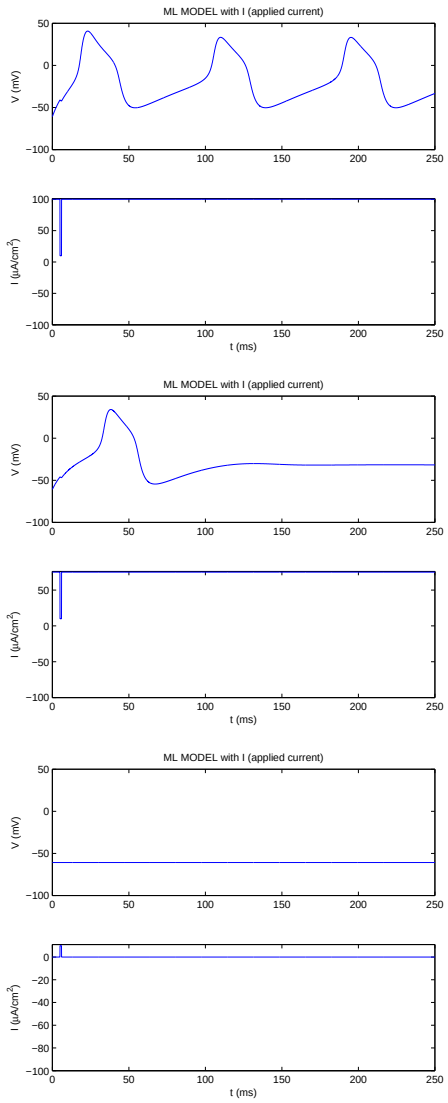


FIGURE 2.14. Voltage and current plots for I=100 (top), I=75 (middle) and I=0 (bottom figure) at $\phi = 0.04$.

For comparison, stimulation for I=75 of (95,10,200), that is 20 above the baseline I, 10 time units, 200 seconds after initiation, we get subcritical behavior:

Using the same units as above, but increasing the duration to 13 seconds (anything below $\approx$ 13 seconds won't work) we get another AP, supracritical behavior.

Increasing the stimulation intensity by 5, while keeping the durration at 10 has the same effect.

Professor: Dr. Medvedev
Student: Timothy Jones

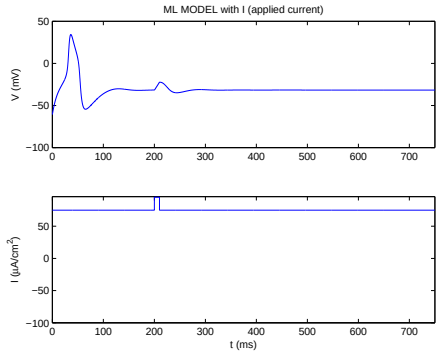


FIGURE 2.15. For voltage and current plots for I=75 and input of (95,10,200), that is 20 above the baseline I, 10 time units duration, 200 seconds after initiation, we get subcritical behavior.

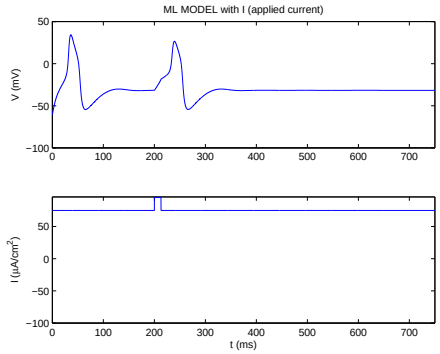


FIGURE 2.16. For voltage and current plots for I=75 and input of (95,13,200), that is 20 above the baseline I, 13 time units duration, 200 seconds after initiation, we get additional supercritical behavior.

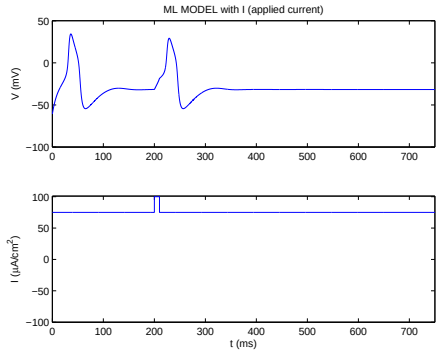


FIGURE 2.17. For voltage and current plots for I=75 and input of (100,10,200), that is 25 above the baseline I, 10 time units duration, 200 seconds after initiation, we get additional supercritical behavior.

Professor: Dr. Medvedev
Student: Timothy Jones

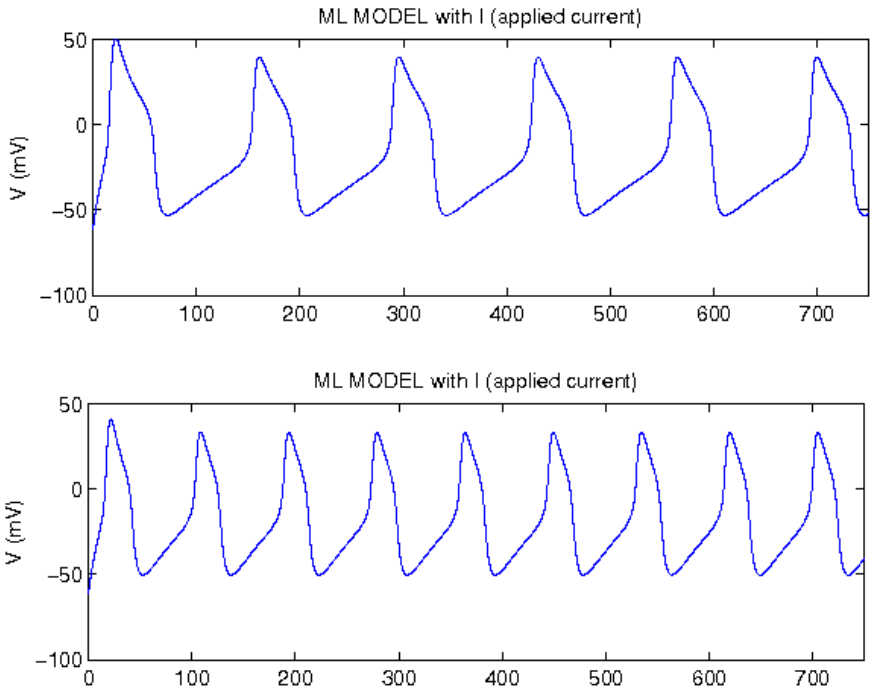


FIGURE 2.18. For voltage and current plots for I=100 and input of (10,1,200). The top graph is for $\phi = 0.02$ and the bottom is for $\phi = 0.04$. As can be seen in the graph, phi decreases the frequency of the AP train and widens the spikes.

Professor: Dr. Medvedev
Student: Timothy Jones

Homework Three

Student: Timothy Jones
Math 723: Mathematics of Neuroscience
Professor: Medvedev
Spring 2008

3.1.

Saddle-node on an invariant circle bifurcation. Consider a system of equations written in polar coordinates:

$$\dot{\rho} = \rho(1-\rho^2)$$
$$\dot{\theta} = 1+\mu-\sin\theta, \quad p \geq 0, \ \theta \in S^1$$

Sketch (by hand) the phase portraits for positive and negative values of μ near 0 and for $\mu = 0$. for small $\mu > 0$, estimate the period of the limit cycle as a function of μ.

Obviously, ρ can't be expanded in terms of small μ, but $\dot{\theta}$ can. Our expansion will be in the form of

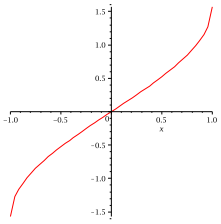
$$\begin{aligned}\dot{\theta} &= f(\theta,0) + f_\mu(\theta,0)\mu + f_{\mu\mu}(\theta,0)\mu^2 + \mathcal{O}(3) \\ &= 1-\sin\theta+\mu\end{aligned}$$

Obviously in this case the expansion takes on the same form as the function, and so we are not helped with this method. It is obvious, though, that we have fixed points at $\rho_f = \{0,1\}$ and $\theta_f = \{\arcsin(1+\mu), \pi - \arcsin(1+\mu)\}$ where the latter follows from the fact that $\sin(A-B) = \sin A\cos B - \cos A \sin B$.

We note that $f_\theta(\theta_f,\mu) = -\cos\theta_f$ so that $\arcsin(1+\mu)$ is stable ($f_x < 0$) until $\mu = 0$ and $\pi - \arcsin(1+\mu)$ is unstable ($f_\theta > 0$) until $\mu = 0$. At $\mu = 0$ the two fixed points crash into each other. For $\mu > 0$ the results are no longer purely real and we no longer have fixed points. The fixed point for $\rho = 0$ is pallid, so we only focus on $\rho = 1$.

Formally, we note that,

$$Df(\theta,\mu) = \begin{pmatrix} 1-3\rho^2 & 0 \\ 0 & -\cos\theta \end{pmatrix} \overbrace{\rightarrow}^{\text{fixed points}} \left\{ \begin{pmatrix} -2 & 0 \\ 0 & -\cos(\arcsin(1+\mu)) \end{pmatrix} \begin{pmatrix} -2 & 0 \\ 0 & -\cos(\pi-\arcsin(1+\mu)) \end{pmatrix} \right)$$



For only the range $\mu \in [-1,1]$ this function has real values. Write $\zeta = \{\cos(\arcsin(1+\mu)), \pi - \cos(\arcsin(1+\mu))\}$ then we have $\lambda = \frac{-(2+\zeta)\pm\sqrt{(2+\zeta)^2-8\zeta}}{2}$. The graph of arcsin is shown in the margin. For $x \in (-1,0)$, arcsin returns values from $[-\pi/2, 0]$, and for corresponding positive values of x it returns values from $[0,\pi/2]$. The first fixed point Jacobian above will thus have one positive and one negative diagonal component for $\mu \in [-1,-1]$. The second will have both negative diagonal components for the same values. Thus the Jacobians range ($\mu \in [-1,0]$) as:

$$Df(\theta,\mu) = \begin{pmatrix} 1-3\rho^2 & 0 \\ 0 & -\cos\theta \end{pmatrix} \overbrace{\rightarrow}^{\text{fixed points}} \left\{ \begin{pmatrix} -2 & 0 \\ 0 & -(1...0) \end{pmatrix} \begin{pmatrix} -2 & 0 \\ 0 & -(-1...0) \end{pmatrix} \right)$$

All eigenvalues will be real under these conditions (thus hyperbolic), and so the Hartman-Grobman theorem applies. The top Jacobian yields, at maximum, the secular equation $\lambda^2 + 3\lambda + 2 = 0$ which gives two negative eigenvalues and is thus fully stable. The second Jacobian gives one positive and one negative eigenvalue, and is thus a saddle point. The plots below show this clearly. For $\mu > 1$ the stable points vanish. We have animated the evolution of the bifurcation for $\mu \in (-1,1)$ in both Cartesian and polar coordinates and have posted the animations on our webpage:

http://www.physics.drexel.edu/~tim/programs/snic_neuroscience/

We present a few of our graphs below.

We estimate the period by restricting ourselves to $\rho = 1$ and calculating:

Professor: Dr. Medvedev
Student: Timothy Jones

$$\frac{d\theta}{dt} = 1 + \mu - \sin\theta \;\rightarrow\; \int_{-\pi/2}^{\pi/2} \frac{d\theta}{1+\mu-\sin\theta} = \int_0^T dt$$

With the help of Maple (this is a difficult integral), we have (next page),

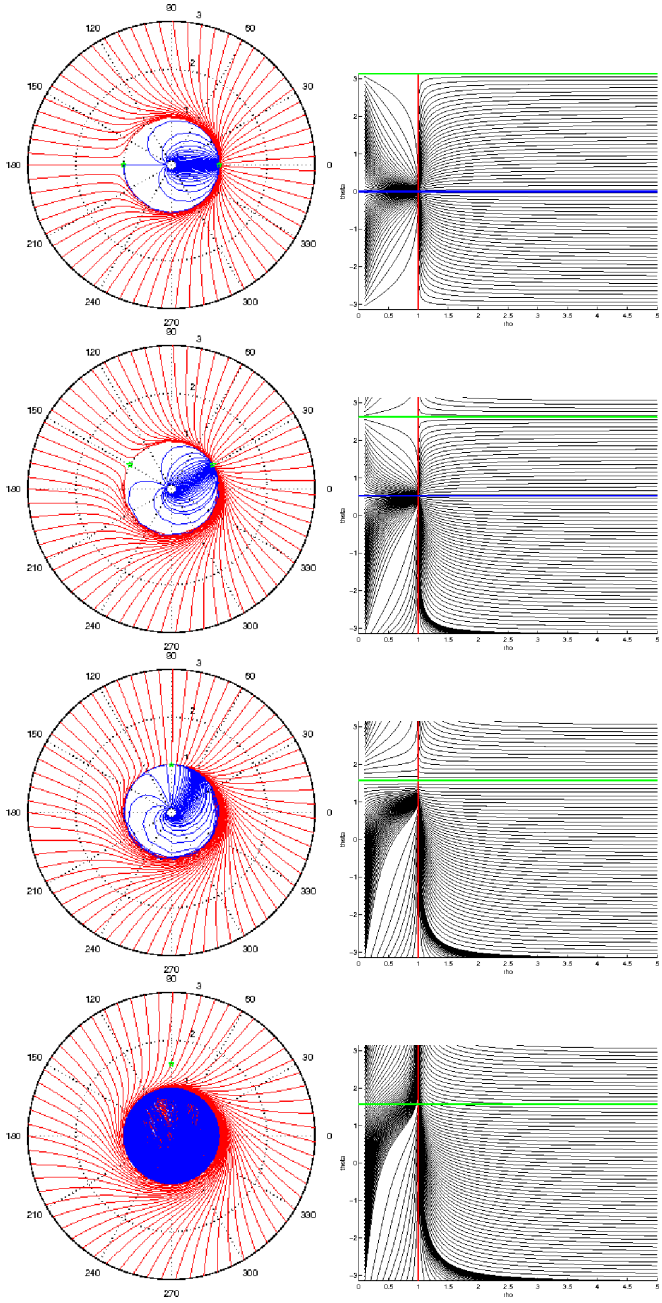


FIGURE 3.1. Phase portraits for this function as compiled in Matlab in polar and Cartesian plot mode. In order we see $\mu = -1$, $\mu = -0.5$, $\mu = 0$ and $\mu = 0.5$. The full animation of these plots can be found at the cited webpage.

[3]

Professor: Dr. Medvedev
Student: Timothy Jones

$$\int_0^{2\pi} \frac{d\theta}{1+\mu-\sin\theta} = \frac{2i\left[\ln\left(\frac{-i(1+\mu)}{\sqrt{\mu(2+\mu)}}\right) - \ln\left(\frac{i(1+\mu)}{\sqrt{\mu(2+\mu)}}\right)\right]}{\sqrt{\mu(2+\mu)}}$$

We use the fact that $e^{ix} = \cos x + i\sin x$, $\ln ax = \ln a + \ln x$, $\ln\frac{1}{x} = -\ln x$, and $\ln x^n = n\ln x$ to rewrite one piece as:

$$2i\left[-\frac{i\pi}{2} + \ln(1+\mu) - \frac{1}{2}(2\mu+\mu^2) - \frac{i\pi}{2} - \ln(1+\mu) + \frac{1}{2}(2\mu+\mu^2)\right]$$

Whereby,

$$T = \int_0^{2\pi} \frac{d\theta}{1+\mu-\sin\theta} = \frac{2\pi}{\sqrt{\mu(2+\mu)}}$$

This is undefined for $\mu \leq 0$, as should be the case from what we learned earlier.

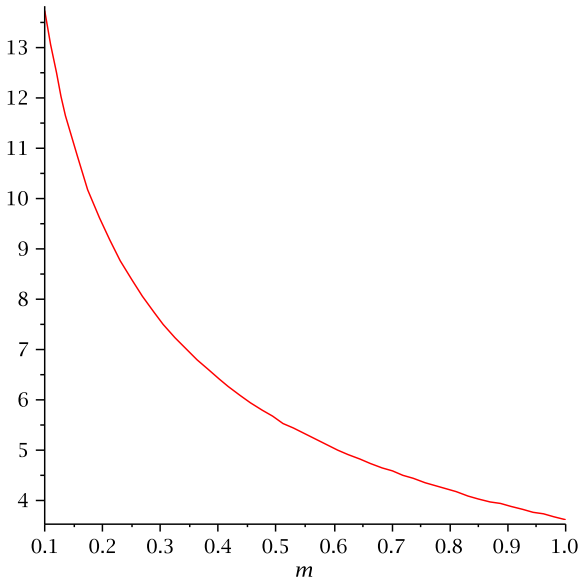


FIGURE 3.2. The estimated period of the limit cycle as a function of μ, in time units (y-axis). Here we plot for m (μ) from 0.1 to 1. As μ goes beyond one, the estimated period asymptotically tends towards zero.

Isolated periodic trajectories are called **limit cycles**. In this case, for $\mu \in [-1, 0)$ there is a stable point on the limit cycle and so

MATLAB code for this solution:

```
function STNI_run(muin)

global mu;
mu=muin;

tspan=[0 100];
figure(1)
axis([0 5 -pi pi])
hold on

for i=0:0.1:5
```


Professor: Dr. Medvedev
Student: Timothy Jones

```
    x0=[i;-3.1415];
    [t,x]=ode23(@SNIC,tspan,x0);
    rho=x(:,1);
    the=x(:,2);
    plot(rho,the, '-black')
    drawnow
end

for i=-3.1415:0.1:3.1415

    x0=[5;i];
    [t,x]=ode23(@SNIC,tspan,x0);
    rho=x(:,1);
    the=x(:,2);
    plot(rho,the, '-black')
    drawnow
end

for i=-pi:0.1:pi

    x0=[0.1;i];
    [t,x]=ode23(@SNIC,tspan,x0);
    rho=x(:,1);
    the=x(:,2);
    plot(rho,the, '-black')
    drawnow
end

y=-3.1415:0.001:3.1415;
plot(1, y, '-r', 'LineWidth', 2)
x=0:0.001:5;
plot(x,asin(1+mu), '-b', 'LineWidth', 2);
plot(x,pi-asin(1+mu), '-g', 'LineWidth', 2);

xlabel('rho');
ylabel('theta');
print -dpng 'three'

function xdot=SNIC(t,x)

global mu

r=x(1);
t=x(2);

dr=r*(1-r^2);
dt=1 + mu - sin(t);
xdot=[dr dt]';  %COLUMN VEC '
```


Professor: Dr. Medvedev
Student: Timothy Jones

3.2.

The matlab code for the numerical experiments with the HH system of the previous subsection is given in the Appendix to this lecture. Modify this code to study the voltage responses in the ML system. For the ML system, use (V0, n0) = (60.855, 0.01495) as the initial condition. Find the values for the amplitude and the duration of stimulation which yield sub- and superthreshold responses, as well as trains of AP. Repeat this numerical experiment for $\phi = 0.02$. Describe the effect of changing ϕ on the trains of AP generated 10 for prolonged stimulation. To make sure that you entered the parameters correctly, compare your numerics with that in Figure 6 for the same values of parameters.

Our modified Matlab code is:

```
function ML_stimulate(intensity,duration,delaytime)

global howstrong howlong delay

howstrong = intensity;  % intensity of applied current
howlong = duration;     % duration of applied current
delay = delaytime;      % delaytime (or beginning time) of applied current

T_MAX = 250;
step = 0.05;
tspan=0:step:T_MAX;

x0 =[-60.855,0.01495];     % steady state values after transient time 10ms
%x = [v n m h];

[t,x] = ode15s(@ml_syst, tspan, x0);

v=x(:,1);
n=x(:,2);

figure(1)

subplot(2,1,1)
plot(t,v);
axis([0 T_MAX -100 50]);
title(sprintf('ML MODEL with I (applied current)', num2str(howstrong)))
ylabel ('V (mV)')


for i=1:length(t)
    current(i)=inp(t(i));
end

subplot(2,1,2)
plot(t, current)
axis([0 T_MAX -100 max(current)+1]);
xlabel ('t (ms)')
ylabel ('I (\muA/cm^2)')
```


Professor: Dr. Medvedev
Student: Timothy Jones

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function xdot = ml_syst(t,x)

I = inp(t);

v=x(1);
n=x(2);

gca=4.4;
gl=2
v2=18
Eca=120;
El=-60;
v3=2;
gk=8;
C=20;
v4=30;
Ek=-84;
v1=-1.2;
phi=0.04;

m=0.5*(1 + tanh((v-v1)/v2));
ninf=0.5*(1 + tanh((v-v3)/v4));
tau=1/(cosh((v-v3)/(2*v4)));

dv=(-gca*m*(v-Eca) - gk*n*(v-Ek)-gl*(v-El)+I)/C;
dn=phi*((ninf-n)/tau);

xdot=[dv dn]';  %column vector: '


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function I=inp(t)

global howstrong howlong delay

I = 100;

t_end  = delay+howlong;

if (t >= delay) & (t<=t_end)
   I = howstrong;
end;
```

For intensity = 10, duration = 1, and delay = 5 we plot v against t to make sure that it resembles Figure 6 of the handout (though since we don't know the exact intensity/duration/delay used to produce figure six, they won't be exactly the same).

We find that for $I = 0$, $I = 75$ and $I = 100$ we find null, subthreshold, and superthreshold behavior (trains of AP) respectively, in agreement with Figure 6 of the hand out.

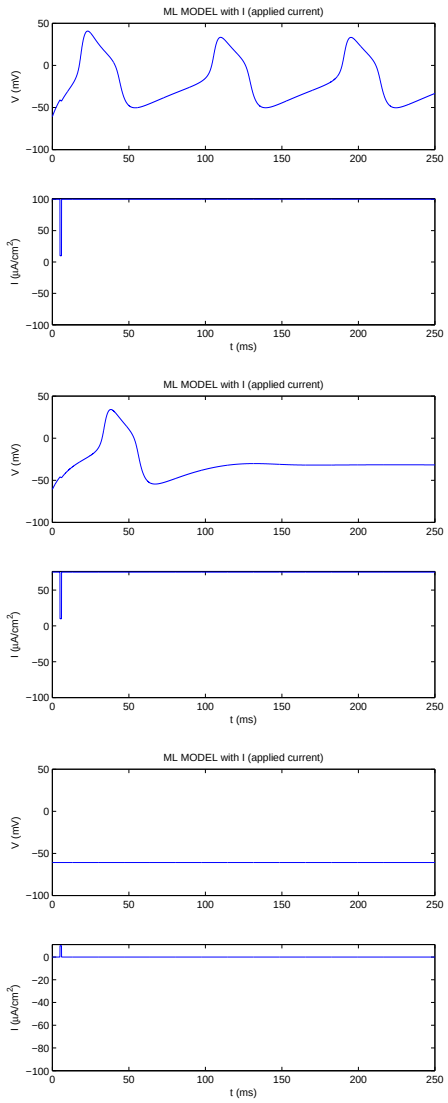


FIGURE 3.3. Voltage and current plots for I=100 (top), I=75 (middle) and I=0 (bottom figure) at $\phi = 0.04$.

For comparison, stimulation for I=75 of (95,10,200), that is 20 above the baseline I, 10 time units, 200 seconds after initiation, we get subcritical behavior:

Using the same units as above, but increasing the duration to 13 seconds (anything below ≈ 13 seconds won't work) we get another AP, supracritical behavior.

Increasing the stimulation intensity by 5, while keeping the duration at 10 has the same effect.

Professor: Dr. Medvedev
Student: Timothy Jones

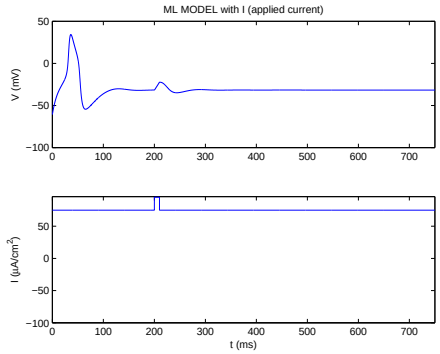


FIGURE 3.4. For voltage and current plots for I=75 and input of (95,10,200), that is 20 above the baseline I, 10 time units duration, 200 seconds after initiation, we get subcritical behavior.

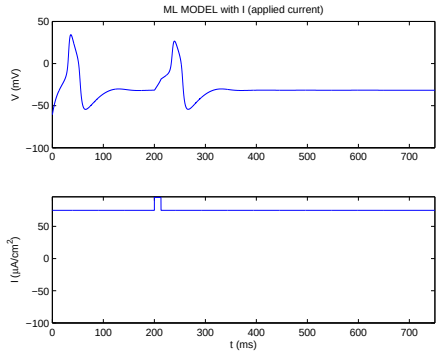


FIGURE 3.5. For voltage and current plots for I=75 and input of (95,13,200), that is 20 above the baseline I, 13 time units duration, 200 seconds after initiation, we get additional AH behavior.

Finally we note that changing ϕ alters the frequency of the AP train.

Professor: Dr. Medvedev
Student: Timothy Jones

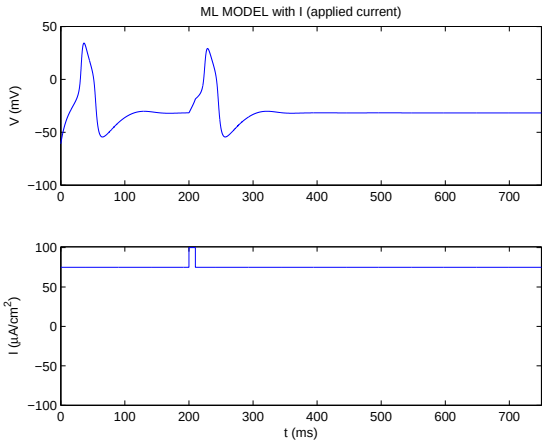


FIGURE 3.6. For voltage and current plots for I=75 and input of (100,10,200), that is 25 above the baseline I, 10 time units duration, 200 seconds after initiation, we get additional AH behavior.

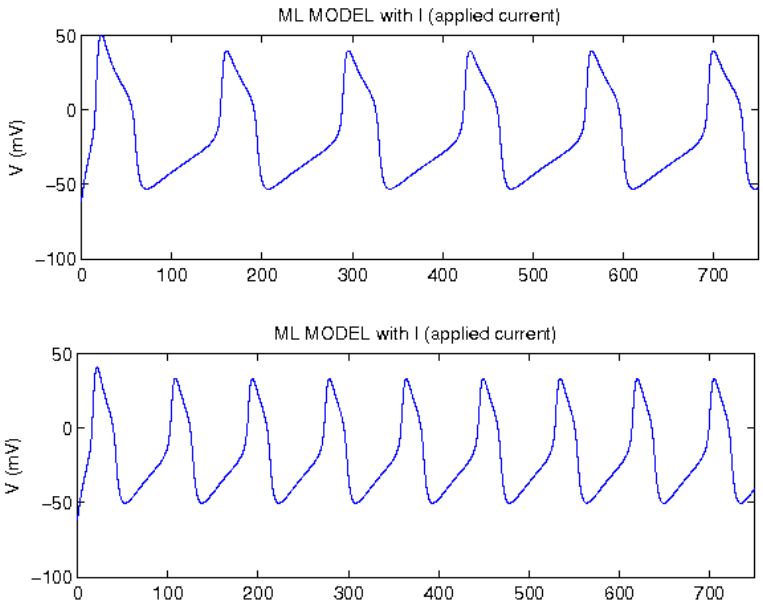


FIGURE 3.7. For voltage and current plots for I=100 and input of (10,1,200). The top graph is for $\phi = 0.02$ and the bottom is for $\phi = 0.04$. As can be seen in the graph, phi decreases the frequency of the AP train and widens the spikes.

3.3.

Professor: Dr. Medvedev
Student: Timothy Jones

The homework project for Lecture 5 "The Hodgkin-Huxley model" (see notes for this lecture).

a. Plot the phase portrait for the ML model. Determine whether it is a type I or type II model. Change the value(s) of some parameter(s) in this model to make it of a different type.

b. Use phase plane analysis to discuss two mechanisms for the action potential generation (excitability) in the Morris-Lecar model.

The Morris-Lecar model was formulated to "explain different patterns of electrical activity observed in the barnacle muscle fiber". It consists of only two differential equations, making it simpler than the Hodgkin-Huxley model. The equations are,

$$\begin{aligned} C\frac{dV}{dt} &= -g_{Ca}m_\infty(V-E_{Ca}) - g_K n(V-E_K) - g_L(V-E_L) + I \\ \frac{dn}{dt} &= \phi\frac{n_\infty(V)-n}{\tau(V)} \\ m_\infty(V) &= 0.5\left(1+\tanh\left(\frac{V-\nu_1}{\nu_2}\right)\right) \\ n_\infty(V) &= 0.5\left(1+\tanh\left(\frac{V-\nu_3}{\nu_4}\right)\right) \\ \tau(V) &= \frac{1}{\cosh\left(\frac{V-\nu_3}{2\nu_4}\right)} \end{aligned}$$

g_{Ca}	$4.4mS/cm^2$	E_{Ca}	120mV	g_K	$8mS/cm^2$	E_K	-84mV
g_L	$2mS/cm^2$	E_L	-60mV	C	$20\mu F/cm^2$	ν_1	-1.2mV
ν_2	18mV	ν_3	2mV	ν_4	30mV	ϕ	$0.04\text{m}S^{-1}$

We know (from having simulated this system, see:

http://www.physics.drexel.edu/~tim/programs/ml_V

for an interesting animation of this system as V is varied from 0 to 150) that this system undergoes a Andronov-Hopf Bifurcation. We pause here for a brief review of the Andronov-Hopf Bifurcation. This review will follow that of S. Wiggins in "Introduction to Applied Nonlinear Dynamical Systems and Chaos", Springer-Verlag 1990.

We begin by looking over a simple model,

$$\dot{y} = g(y,\lambda), \quad y \in \mathbb{R}^n, \quad \lambda \in \mathbb{R}^p$$

It is required that $g \in C^r$ (the Morris-Lecar model, hereafter ML, satisfies this). We take the linearization as, with fixed points y_0, λ_0,

$$\dot{\zeta} = D_y g(y_0,\lambda_0)\zeta, \quad \zeta \in \mathbb{R}^n$$

For our current case, let $C = 1$ and we have the matrix:

$$\begin{pmatrix} -\frac{1}{2\nu_2}gC_a\left(1-\tanh^2(\frac{V-\nu_1}{\nu_2})\right)(V-E_{Ca}) - g_{Ca}m_\infty - g_K n - g_L & -g_k(V-E_k) \\ \frac{1}{2\nu_4}\left(\left(1-\tanh^2\left(\frac{V-\nu_3}{\nu_4}\right)\right)\cosh\left(\frac{V-\nu_3}{\nu_4}\right) + \frac{1}{2}\left(1\tanh\left(\frac{V-\nu_3}{\nu_4}\right)+2n\right)\sinh\left(\frac{V-\nu_3}{2\nu_4}\right)\right) & -1 \end{pmatrix}$$

It would be tedious and wasteful of time to not use Maple at this point for such a calculation, and we do so in order to find the Eigenvalues. These are plotted, and the program written for this is given later.

The discussion of the deep details of the AH bifurcation are beyond the call of this assignment. It involves use of the "Center Manifold" on which the dynamics of a fixed point can be reduced for qualitative purposes, and the "Normal Form" of

the equations in which the equations are cast in a coordinate system which provides their 'simplest' manifestation.

The key result is that at the conditions of an AH bifurcation, the dynamics result in a defining equation that is induced when the real parts of the eigenvalue go to zero and a complex conjugate set of imaginary parts does not:

$$(r(t), \theta(t)) = \left(\sqrt{\frac{-ud}{a}}, \left[\omega + \left(c - \frac{bd}{a} \right) \mu \right] t + \theta_0 \right)$$

We use Maple to calculate the eigenvalues (we don't know whether or not an analytical solution would be possible, though we think it would be quite involved). We note in the plot (figure below) that the curve of the real part of the eigenvalue very clearly has a positive derivative as it crosses the zero axis, and thus $d > 0$ as predicted from the phase plot.

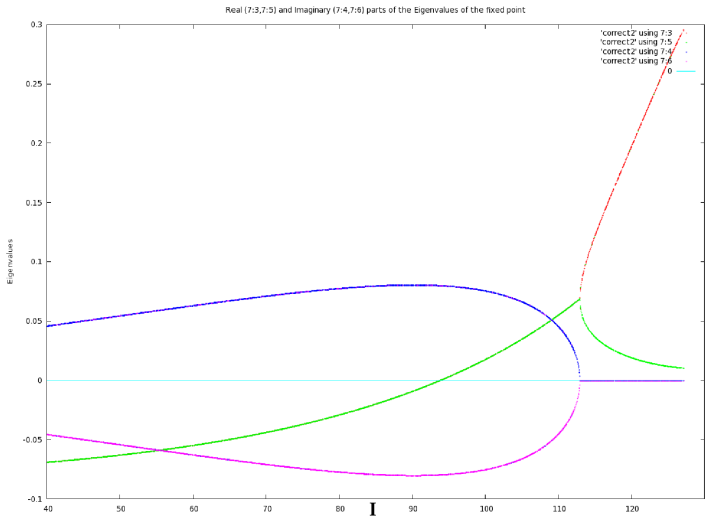


FIGURE 3.8. The real and imaginary parts of the eigenvalues verses I. Notice that at nearly I=93.85, the real part of the eigenvalues crosses the Imaginary axis (go to zero) while the pre-existing imaginary parts do not vanish; under conditions satisfied by this system, this is the cause of the AH bifurcation.

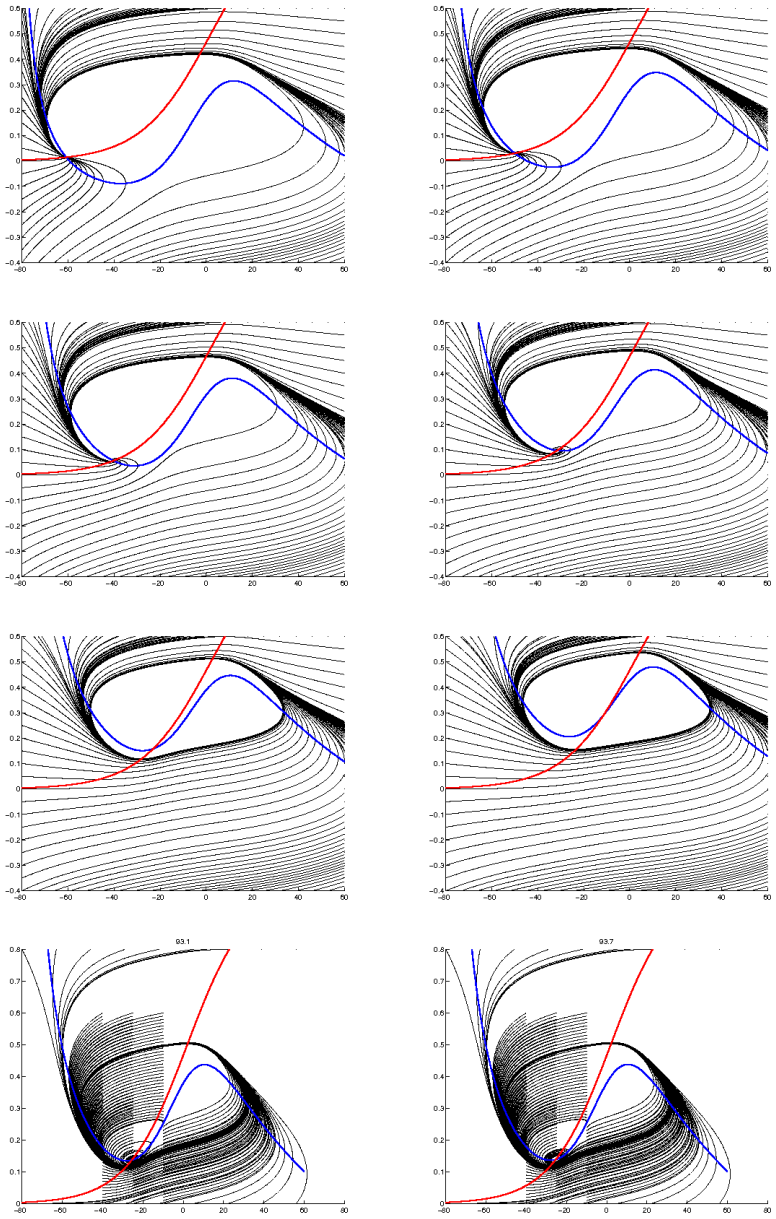


FIGURE 3.9. Left to right and top to bottom except last row: Phase plot for $I = 0, 25, 50, 75, 100, 125$. Notice the AH Bifurcation between $I = 75$ and $I = 100$. Further examination and analysis of the eigenvalues will reveal exactly where this bifurcation occurs. Last row: we focus in right before and after the real part of the eigenvalues cross the zero axis.

Professor: Dr. Medvedev
Student: Timothy Jones

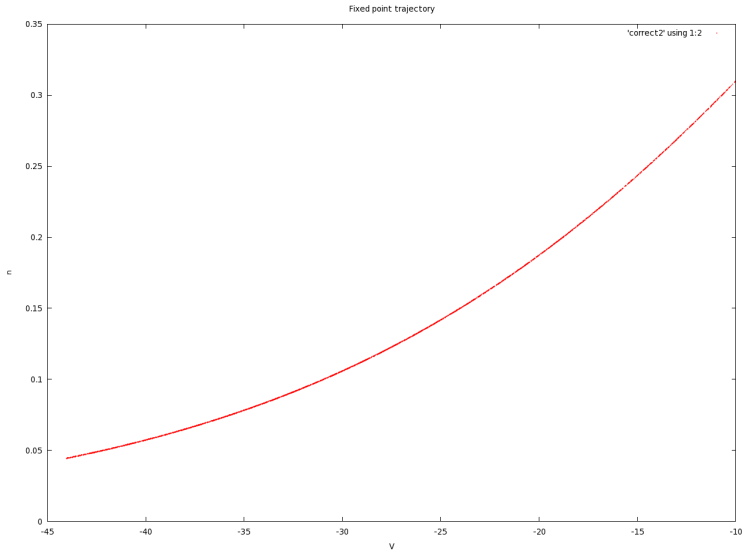


FIGURE 3.10. By using the maple code quoted here and solving for the fixed points, we find a smooth trajectory for the fixed point in the v verses n plane, in agreement with our animations of the phase plot.

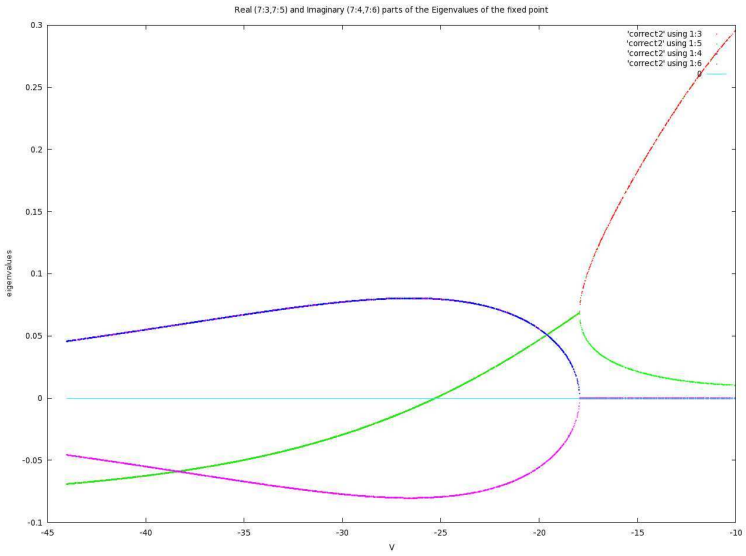


FIGURE 3.11. The real and imaginary parts of the eigenvalues verses V.

Professor: Dr. Medvedev
Student: Timothy Jones

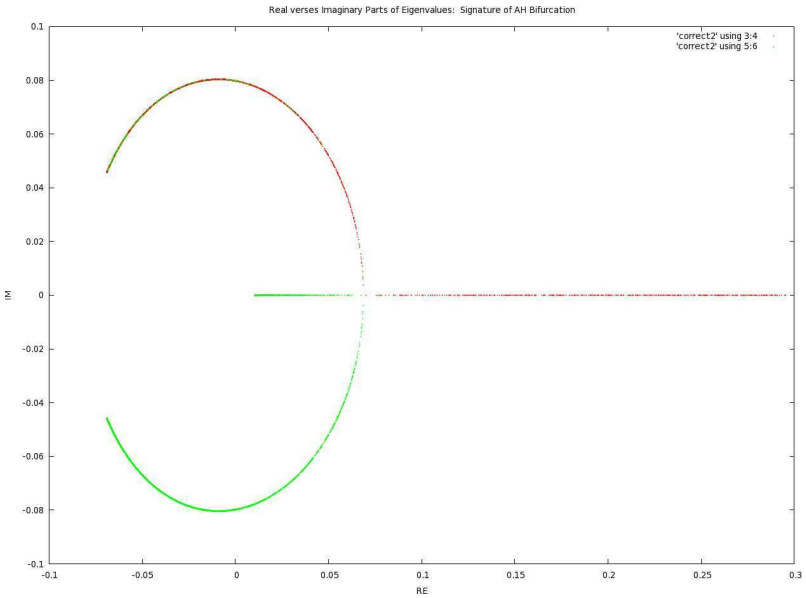


FIGURE 3.12. Here we plot the eigenvalues, real part (x-axis) verses imaginary part (y-axis). The resulting shape is an obvious signature of an AH bifurcation.

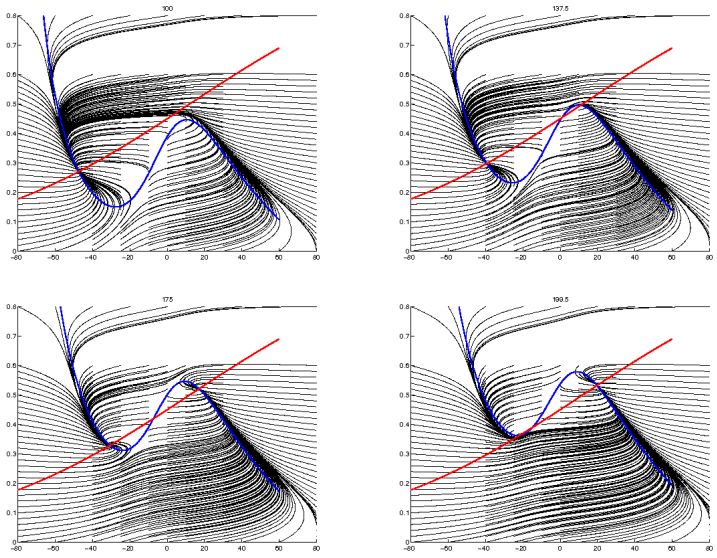


FIGURE 3.13. Left to right and top to bottom: Phase plot for $I = 100, 137.5, 175, 200$. The bifurcation type is now SN, TYPE I. We changed ν_3 from 2 to 12 and ν_4 from 30 to 120.

Professor: Dr. Medvedev
Student: Timothy Jones

Matlab code for phase plot of ML system:

```
function ML_stimulate(intensity,duration,delaytime,curr)

global howstrong howlong delay I


howstrong = intensity;  % intensity of applied current
howlong = duration;     % duration of applied current
delay = delaytime;      % delaytime (or beginning time) of applied current
I=curr;

T_MAX = 750;
step = 0.05;
tspan=0:step:T_MAX;


%x = [v n m h];
figure(1)
axis([-80 60 -0.4 0.6])

hold on

for i=-80:7:60
x0=[i;-0.4];
[t,x] = ode15s(@ml_syst, tspan, x0);
v=x(:,1);
n=x(:,2);
plot(v,n,'-black')
drawnow
x0=[i;0.6];
[t,x] = ode15s(@ml_syst, tspan, x0);
v=x(:,1);
n=x(:,2);
plot(v,n,'-black')
drawnow
end
for i=-0.4:0.05:0.7
x0=[-80;i];
[t,x] = ode15s(@ml_syst, tspan, x0);
v=x(:,1);
n=x(:,2);
plot(v,n,'-black')
drawnow
x0=[60;i];
[t,x] = ode15s(@ml_syst, tspan, x0);
v=x(:,1);
n=x(:,2);
plot(v,n,'-black')
drawnow
end
```

```
%I=70;

gca=4.4;
gl=2;
v2=18;
Eca=120;
El=-60;
v3=2;
gk=8;
C=20;
v4=30;
Ek=-84;
v1=-1.2;
phi=0.04;

vv=-80:0.01:60;

m=0.5*(1 + tanh((vv-v1)/v2));
ninf=0.5*(1 + tanh((vv-v3)/v4));
tau=1./(cosh((vv-v3)/(2*v4)));


%null1=(-gca*m*(vv-Eca) - gl*(vv-El)+I)./(gk*(vv-Ek));
null1=(I-gl*(vv-El)-gca.*m.*(vv-Eca))./(gk*(vv-Ek));
null2=ninf;
plot(vv , null1, '-b', 'LineWidth', 2);
plot(vv, null2, 'r', 'LineWidth',2);
print -dpng 'ml'
```

Shell code for animating above

```
#!/bin/sh

j=1000
counter=0
while [ $counter -lt 301 ]
do
i=$counter/2
echo $i
unset DISPLAY
matlab >&! matlab.out <<EOF
ML_stimulate2(0,0,0,$i)
EOF

mv ml.png $j.png
let j=j+1
let counter=counter+1

done
```

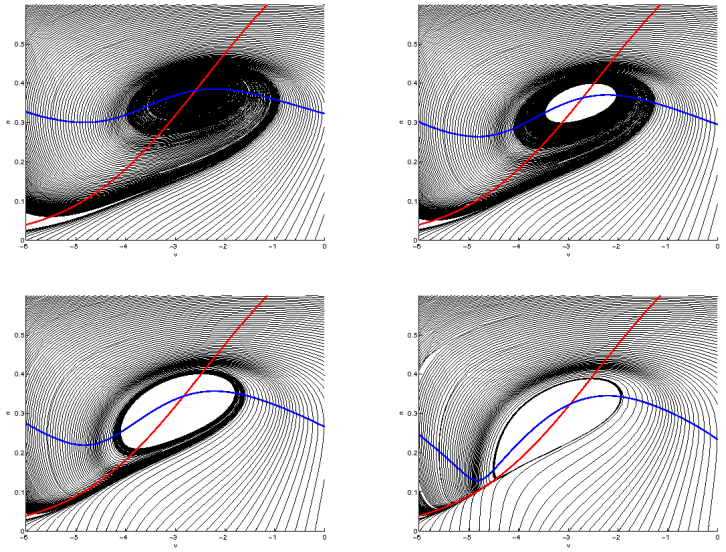
Maple code for finding fixed points and eigenvalues

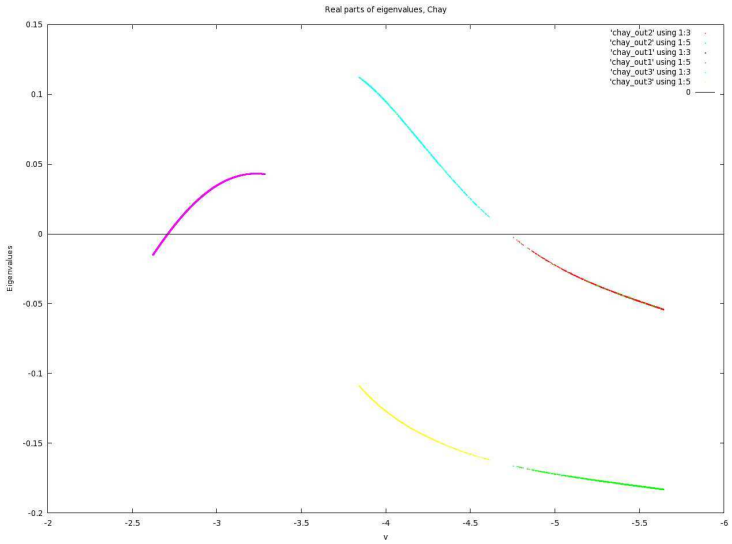
```
restart;
with(linalg); with(LinearAlgebra);
Digits := 20;
gca := 4.4; gl := 2; v2 := 18; eca := 120; el := -60;
v3 := 2; gk := 8; c := 20; v4 := 30; ek := -84;
v1 := -1.2; phi := 0.4e-1;
fd := fopen("correct2", WRITE);

for II from 40 by 0.01 to 150 do
unassign('V'); unassign('n');
mm := proc (V) options operator, arrow; .5*(1+tanh((V-v1)/v2)) end proc;
nn := proc (V) options operator, arrow; .5*(1+tanh((V-v3)/v4)) end proc;
tt := proc (V) options operator, arrow; 1/cosh((1/2)*(V-v3)/v4) end proc;
dv := proc (V, n) options operator, arrow; (-gca*mm(V)*(V-eca)-gk*n*(V-ek)-gl*(V-el)+II)/c end
dn := proc (V, n) options operator, arrow; phi*(nn(V)-n)/tt(V) end proc;
ul := diff(dv(V, n), V);
ur := diff(dv(V, n), n);
bl := diff(dn(V, n), V);
br := diff(dn(V, n), n);
poly := {dv(V, n) = 0, dn(V, n) = 0};
a := fsolve(poly);
V := op(2, op(1, a));
n := op(2, op(2, a));
A := matrix([[ul(V, n), ur(V, n)], [bl(V, n), br(V, n)]]);
eigens := {eigenvalues(A)};
eigen1 := op(1, eigens); eigen2 := op(2, eigens);
if V < -10 then
if n*n < 1 then
fprintf(fd, "%f %f %f %f %f %f %f \n", V, n, Re(eigen1),
                    Im(eigen1), Re(eigen2), Im(eigen2), II)
end if
end if;
end do;
```

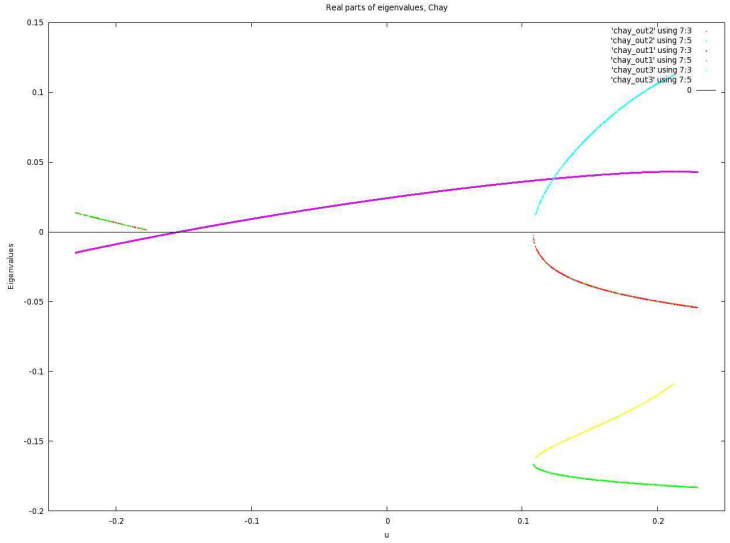

Professor: Dr. Medvedev
Student: Timothy Jones

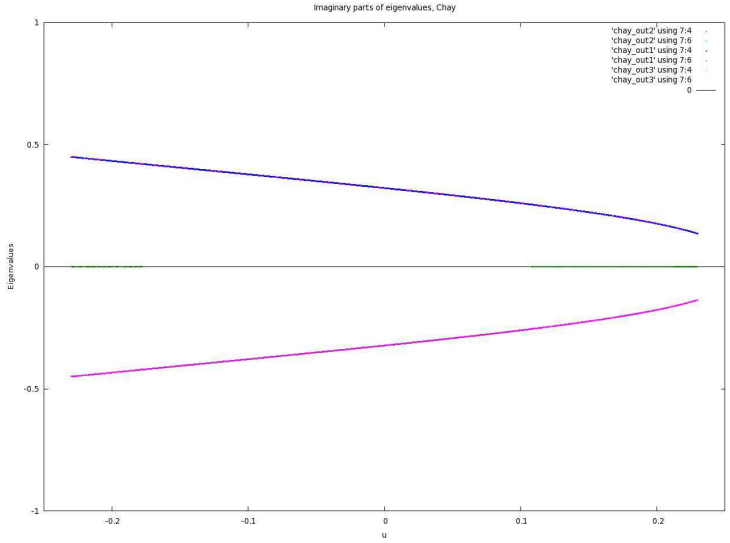
3.4. Coda: The Chay Model. The following graphs demonstrate this students attempt to recreate some of the data in the journal article "Reduction of a model of an excitable cell to a one-dimensional map" by Georgi S. Medvedev, Physica D 202 (2005) 37-59. It is obvious that our code needs refinement, but the general behavior of the system is evident.



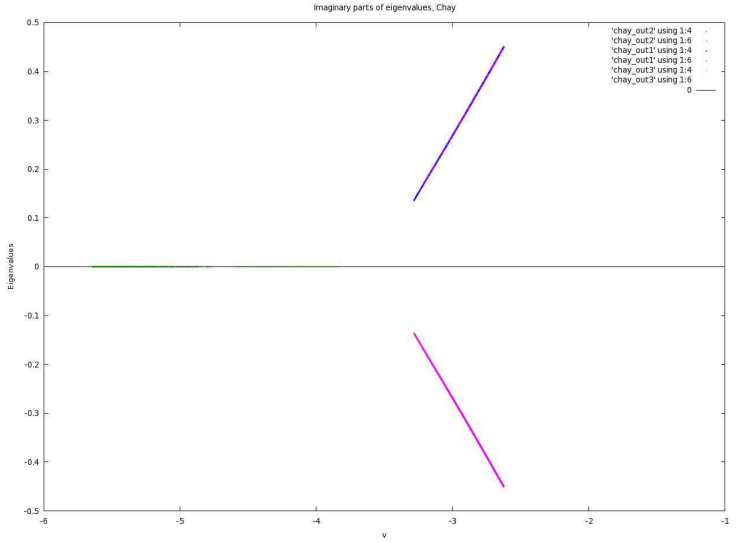


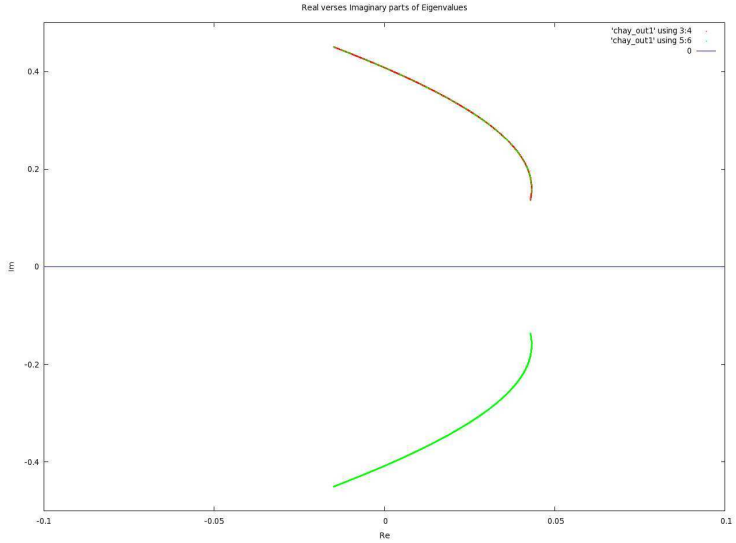
Professor: Dr. Medvedev
Student: Timothy Jones





Professor: Dr. Medvedev
Student: Timothy Jones





Professor: Dr. Medvedev
Student: Timothy Jones

REPORT ON "THE DYNAMIC STRUCTURE UNDERLYING SUBTHRESHOLD OSCILLATORY ACTIVITY AND THE ONSET OF SPIKES IN A MODEL OF MEDIAL ENTORHINAL CORTEX STELLATE CELLS"

TIMOTHY JONES

Abstract. The entorhinal cortex (EC) has been found to be a major information hub for the hippocampus [1] and seems to play an important role in memory. This part of the brain has been found to be among the first and most severely damaged by Alzheimer's disease as well as other dementing disorders; young individuals with the Alzheimer's related variant of the ApoE gene are statistically found to have thinner entorhinal cortexes than do controls [2, 3].

A common neuron found in the EC is the spiny stellate cells. These cells demonstrate, individually, subthreshold oscillation (STO) [4]. Rotstein, Opperman, White, and Kopell [5] discuss a particular conductance based model for these cells [6] that reproduces the STO phenomenon. They seek to explain the STOs in dynamical terms. The goal of this report is to summarize and simulate their results as a final project for Math 723.

1. Subthreshold Oscillations Discovered and Modeled

1.1. **Experimental results.** The thrust of this report is dynamical, not physiological, but we briefly discuss some physiological results from stellate cells of layer II in the entorhinal cortex (ECIIsc). Alonso and Llinás demonstrated these STOs in their 1989 Nature article [4] from which we present in Figure 1.1.

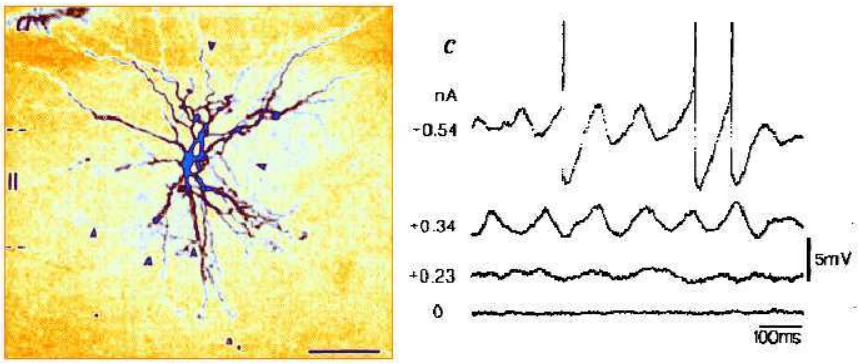


Figure 1.1. ECIIsc cell (left) under microscopic magnification. The scale bar in the lower right of the left panel is 100μm. The graph on the right represents experimental data from the neuron stimulated with current. The oscillation becomes apparent at $\tilde{5}$7mV, which the model we discuss replicates quite well. These figures are from [4]. False color introduced by this student to enhance figure.

1.2. **Conductance model.** Acker et al. [6] introduced a conductance model for the ECIIsc cells. Before going into detail about this model, we demonstrate its success in modeling the ECIIsc cells by including some graphs (Figure 1.2) of its simulation, programed by this student in MATLAB. We conclude with 3D images of the trajectory for $I_{app} = -2.5$ as modeled in the ray tracer program POV-Ray.

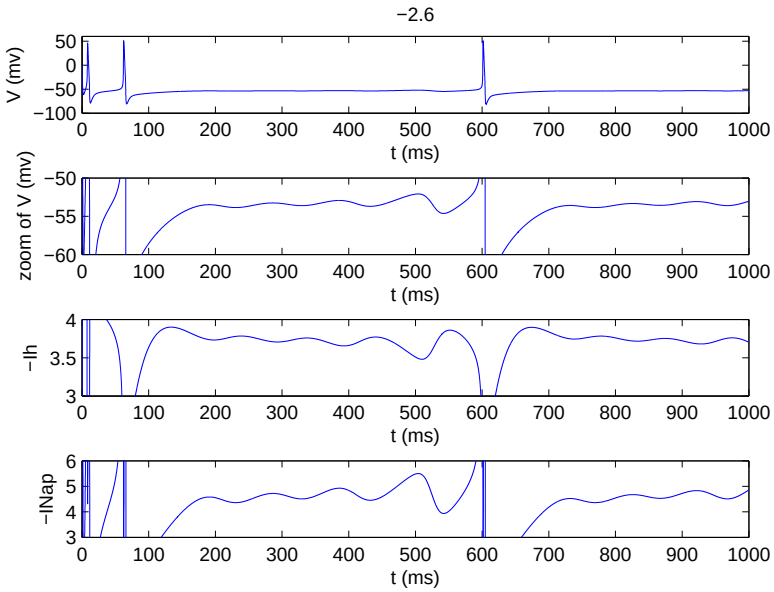


FIGURE 1.2. ECIIsc cell conductance model of Acker et al. (2003) [6].From top to bottom: Action potentials; second panel zooms in on the first one where subthreshold oscillation is apparent. The third and forth panels show the behavior of I_h and I_{Nap} whose role in the model is explored in this report. These figures were made via this student's MATLAB simulation Acker_Fig002.m (included at the end of this report) based on this model.

Channels without inactivation gates are called **persistent**, channels with such gates are called **transient**.

In a generic model, the current passing through a membrane due to a particular type of current is given as $I = \bar{g}p(V - E)$ where p is the average proportion of channels in the open state (a probability, basically), $\bar{g}$ is the maximal conductance, and E is the reverse potential of the current. In Hodgkin-Huxley type models, m (and n for K^+ and CL^- channels) represents the probability that an activation gate is in its open state, and h the probability that an inactivation gate is in the open state. See Figure 1.3 for a simple schematic of these states.

The dynamics of the activation variables such as m are covered in [7] and references therein. They are created to model experimental results. The dynamical system discussed in this report has similarly been tailored to match experimental results. The Acker et al. model is a seven dimensional system based on the interactions of the following current channels: persistent sodium (I_{Nap}), a two-component (fast and slow) hyperpolerization-activated current (I_h), and the standard Hodgkin-Huxley currents, i.e., sodium (I_{Na}), potassium (I_K) and the leak current (I_L). I_{app} is the applied bias current (DC, $\mu A/cm^2$). C is the membrane capacitance $/cm^2$. We denote the membrane potential V (mV). The remaining currents are modeled with the following equations:

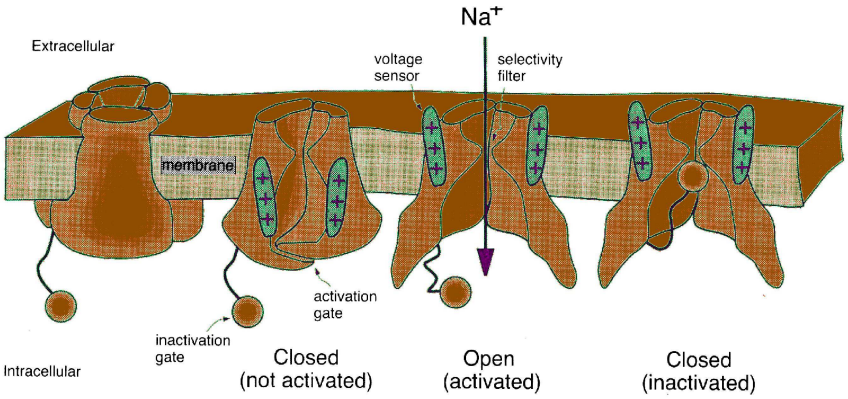


FIGURE 1.3. Figure modified from Izhikevich [7] from Armstrong and Hille 1998.

$$
\begin{aligned}
I_{Na} &= G_{Na} m^3 h (V - E_{Na}) \\
I_k &= G_K n^4 (V - E_k) \\
I_L &= G_L (V - E_L) \\
I_{Nap} &= G_p p (V - E_{Na}) \\
I_h &= G_h (0.65 r_f + 0.35 r_s)(V - E_h)
\end{aligned}
$$

The dynamical equations are:

$$C\frac{dV}{dt} = I_{app} - I_{N_a} - I_k - I_L - I_h - I_{Nap} \tag{1.1}$$

$$\frac{dm}{dt} = \frac{m_\infty(V) - m}{\tau_m(V)} \tag{1.2}$$

$$\frac{dh}{dt} = \frac{h_\infty(V) - h}{\tau_h(V)} \tag{1.3}$$

$$\frac{dn}{dt} = \frac{n_\infty(V) - n}{\tau_n(V)} \tag{1.4}$$

$$\frac{dp}{dt} = \frac{p_\infty(V) - p}{\tau_p(V)} \tag{1.5}$$

$$\frac{dr_f}{dt} = \frac{r_{f\infty}(V) - r_f}{\tau_{rf}(V)} \tag{1.6}$$

$$\frac{dr_s}{dt} = \frac{r_{s\infty}(V) - r_s}{\tau_{rs}(V)} \tag{1.7}$$

The definition of the various $_x$ and τ_x can be found in the codes included in this report, and the original papers upon which this report are based. The physiological details can be found in the literature.

2. Reduced model

To continue are discussion, we reproduce Figure 1 of our topic paper below. Rotstein et al. note, from this figure, that $\tau_{r,f}$ and $\tau_{r,s}$ are much greater than all other τ_x in the regime of the STO, what they call the subthreshold interval

(STI), (compare Figure 2.1 with Figure 2.2). The main point upon which their work hinges is the idea that in the STI, I_{Na} and I_K are nearly inactive. From Figure 2.1, left column, one can also approximate that $m_\infty\ n_\infty^4 \approx 0$. Since τ_p is so small in this regime, it quickly evolves and so one can use the approximation $p \approx p_{\infty(v)}$, and from the previous assumptions, $m \approx 0, I_{Na} \approx 0$, and $I_K \approx 0$ due to their dependence on m and n.

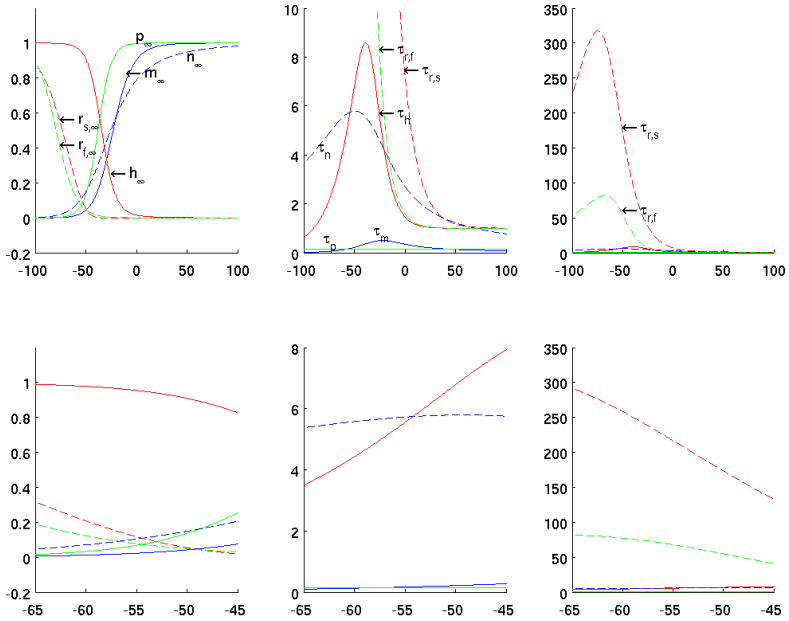


FIGURE 2.1. Recreation of Figure 1 of [5]. Ion channel dynamics for full seven dimensional SC model. Bottom row magnifies corresponding top row. First column shows activation and inactivation curves for the gating variables. Second column shows Voltage-dependent time scales. The third column shows the second column for a larger time interval. These time scales are key to the dynamical discussion of [5].

This sets up the reduced equations, given by,

$$C\frac{dV}{dt} = I_{app} - G_p p_\infty(V)(V - E_{Na}) - I_L - I_h \tag{2.1}$$

$$\frac{dr_f}{dt} = \frac{r_{f,\infty}(V) - r_f}{\tau_{r_f}(V)} \tag{2.2}$$

$$\frac{dr_s}{dt} = \frac{r_{s,\infty}(V) - r_s}{\tau_{r_s}(V)} \tag{2.3}$$

Next Rotstein et al. note that, as seen in Figure 2.1 at the third column, in the STI regime, $\tau_{r,s} >> \tau_{r,f}$ and so r_f can be taken as a slow system in the fast/slow decomposition. That is, we take r_s as a parameter rather than as an evolving variable. This brings us to our simplest possible system. In Figure 2.4, I use Maple to compute the Eigenvalues for the fixed point of the fast system as a function of

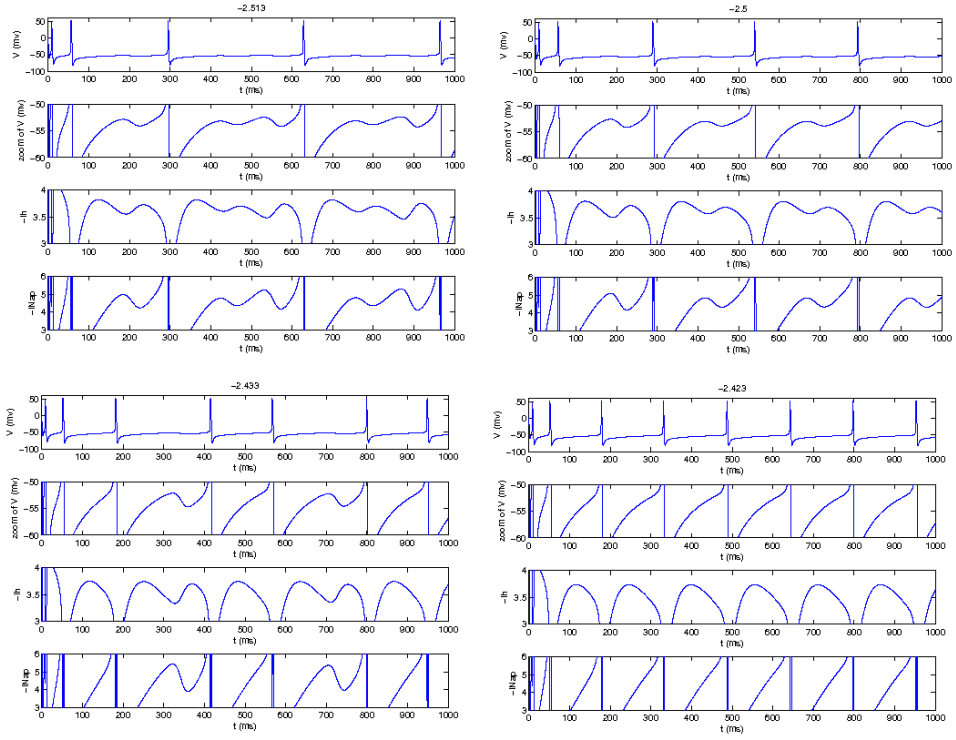


FIGURE 2.2. Approximate recreation of Figure 2 of [5] made using same MATLAB program created for Figure 1.2. Certain unknown initial conditions made an exact recreation unlikely. Full SC model with conditions given in program.

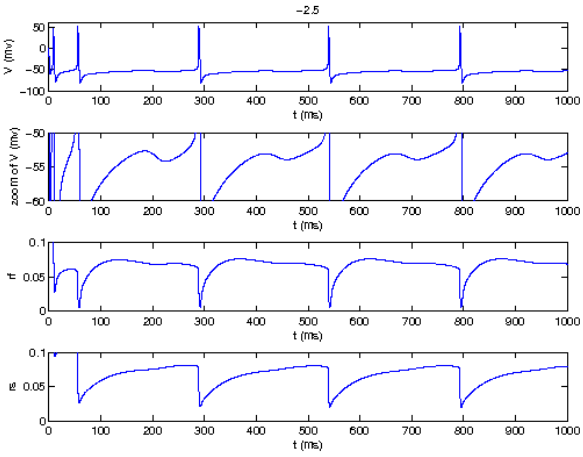


FIGURE 2.3. Approximate recreation of Figure 3 of [5] made using same MATLAB program created for Figure 1.2. Certain unknown initial conditions made an exact recreation unlikely. Full SC model with conditions given in program.

the variable of the slow system (r_s). This is done by calculating the nullclines for the two fast components of the reduced system, and solving for their intersection. Then, for each fixed point, we perform the typical localization, and then solve for the eigenvalues of the Jacobian. If the point is hyperbolic (nonzero real parts of eigenvalues) then the Hartman- Grobman theorem indicates that the results for the linearization of the system will match the results for the actual system (locally).

Center manifold theory can indicate the nature of the system at the point at which the fixed point becomes non-hyperbolic, i.e. when the real part of the eigenvalues crosses the imaginary axis. The result is the Andronov-Hopf bifurcation (sub-critical in this case).

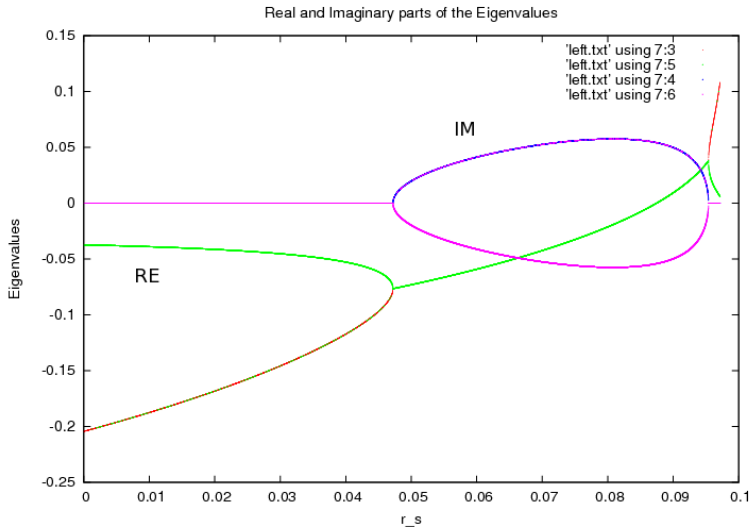


FIGURE 2.4. Eigenvalues for the fast component of the reduced SC model from [5] made using same MATLAB program created for Figure 1.2. Closer look at generating area of STO. The signature of a Hopf bifurcation is present.

The eigenvalue plot of Figure 2.4, created with the enclosed program **eigenrun.m**, suggests the following: In the figure we see the amplitude of the Eigenvalues as plotted for the r_s variable. from $r_s = 0.. \approx 0.047215$ we have two negative real eigenvalues, indicating a fixed point. At around that point, the eigenvalues collide and we now have two complex conjugate eigenvalues with negative real part. This indicates the presence of a spiral fixed point. Around $r_s \approx 0.087460$ the real part of the eigenvalues crosses the Imaginary axis, which is a signature for a Hopf Bifurcation (unstable central point). At $r_s \approx 0.095380$ the imaginary components of the eigenvalues collide and the fixed point becomes simply unstable (non-spiral) and all trajectories should avoid this fixed point. In Figure 2.5, detailed further in Figures 2.6 and 2.7, we see the dynamics indicated by the eigenvalues in the phase plane.

Obviously, the results of the analysis for the fast part of the system will not accurately describe the overall system; it does, however, give a strong indication of the dynamics that drive the STO. We now turn our attention to an analysis of the full reduced system.

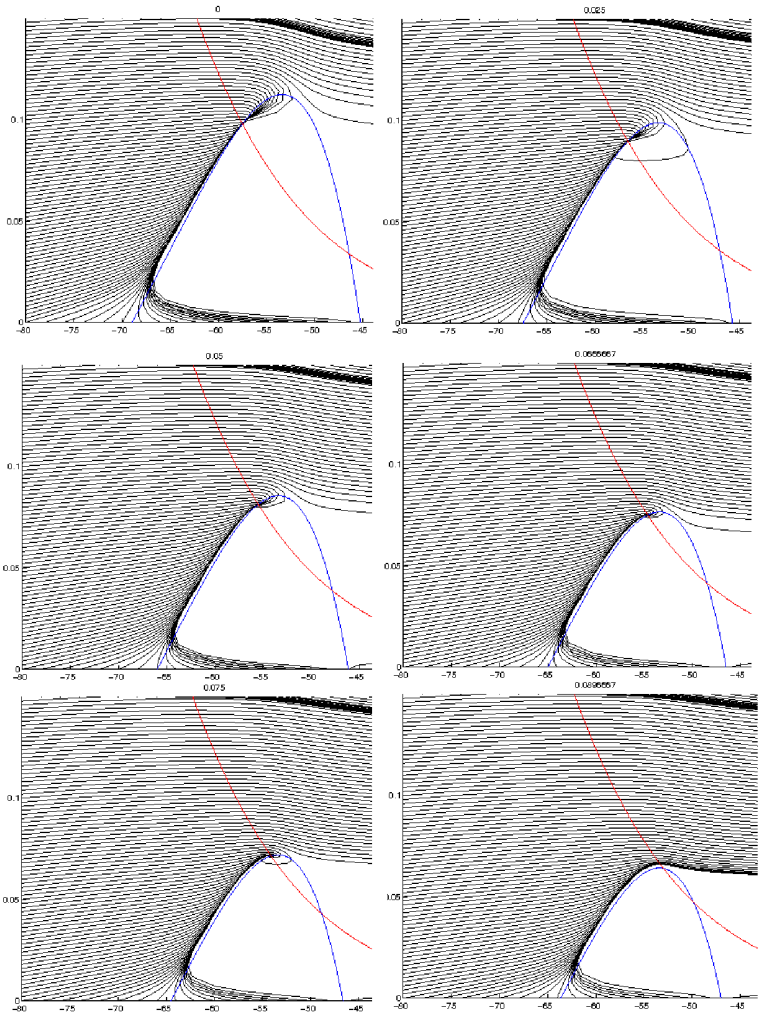


FIGURE 2.5. Approximate recreation of upper panel Figure 5 of [5] made using same MATLAB program created for Figure 1.2. Values of rs scan, from left to right and top to bottom: 0, 0.025, 0.050, 0.075, 0.0666..., and 0.089667.

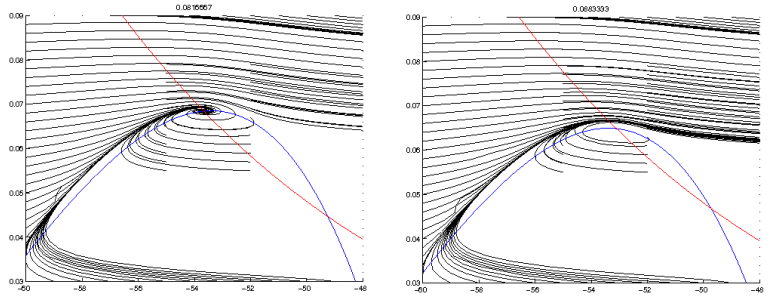


FIGURE 2.6. Approximate recreation of of Figure 5 of [5] made using same MATLAB program created for Figure 1.2. $r_s = 0.081667, 0.88333$, before and after the Hopf bifurcation indicated by the eigenvalues shown in Figure 2.4.

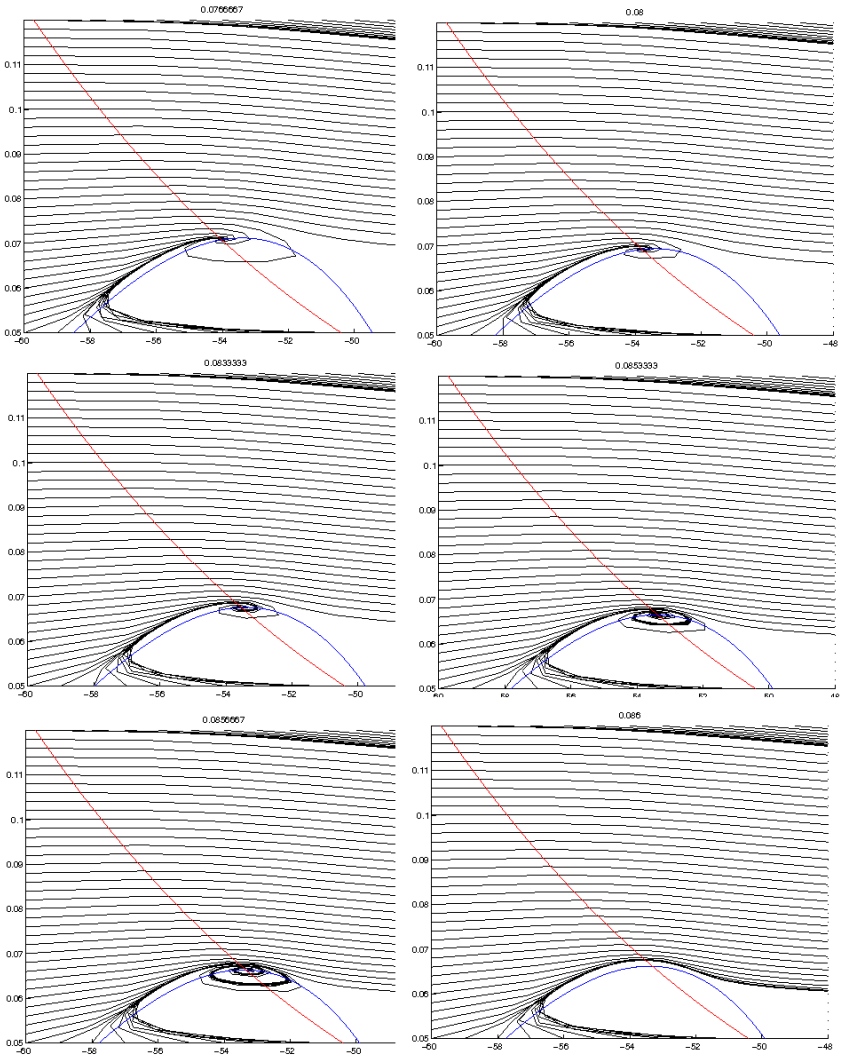


FIGURE 2.7. Approximate recreation of lower panel of Figure 5 of [5] made using same MATLAB program created for Figure 1.2. Values of rs scan, from left to right and top to bottom: 0.0766, 0.08, 0.0833, 0.0853, 0.0856, and 0.086.

3. Modeling of full reduced SC system

I wrote the enclosed C++ code to model the full three dimensional system using Runge-Kutta IV method. In the three dimensional system, the STO motion never settles to a fixed point; after a number of oscillations, the trajectory escapes (spike). The reduced model has no mechanism for return. However, following Rotstein et al., we introduce an artificial "reset" mechanism, i.e. the following line of code:

```
if(y[1]>-30){y[1]=-80; y[2]=0; y[3]=0; flag=0;}
```

This is justified in the article by the fact that under the full model, r_f and r_h reset to near zero after each action potential (see Figure 2.3). The result of our simulation of their simulation can be found in 3.1, with some additional detail.

After a certain threshold of applied current, the subthreshold oscillations can be seen to grow more prominent; we demonstrate this evolution in Figure 3.2.

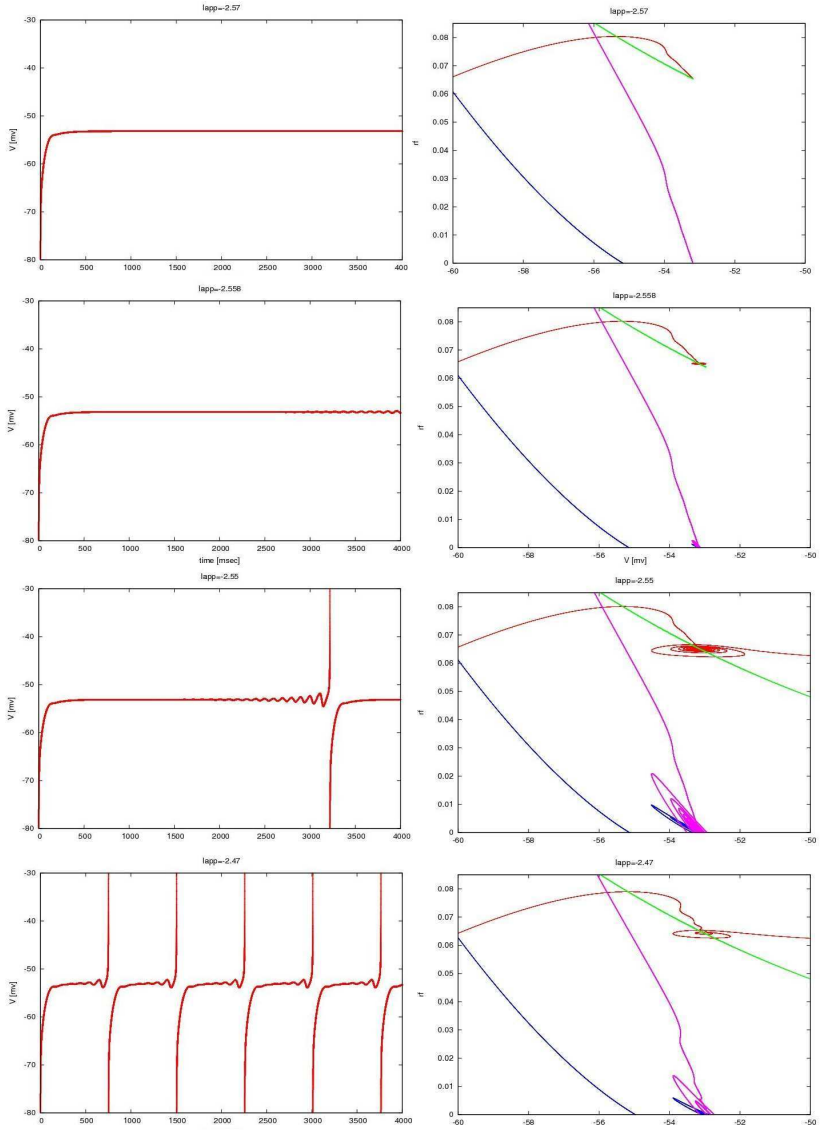


FIGURE 3.1. Approximate recreation of lower panel of Figure 6 of [5] made using same MATLAB program created for Figure 1.2. Values of I_{app} varied (from top to bottom) as -2.57, -2.558, -2.55, -2.47. The right panel is a bit abstract: the green line is the static r_f nullcline. The red line is the trajectory with initial conditions $(V, r_f, r_s) = (-80, 0, 0)$. The blue line represents the difference between the V and r_f nullcline for the given value of V, r_f, r_s that the trajectory has at the given V. The Purple line represents the difference between the V and r_s nullcline for the values of V, r_f, r_s that the trajectory has at the given V. Since the system is allowed to evolve in all three coordinates freely, no static nullclines for V can be drawn in this diagram. The "reset" in the left column is artificial, see text for more details.

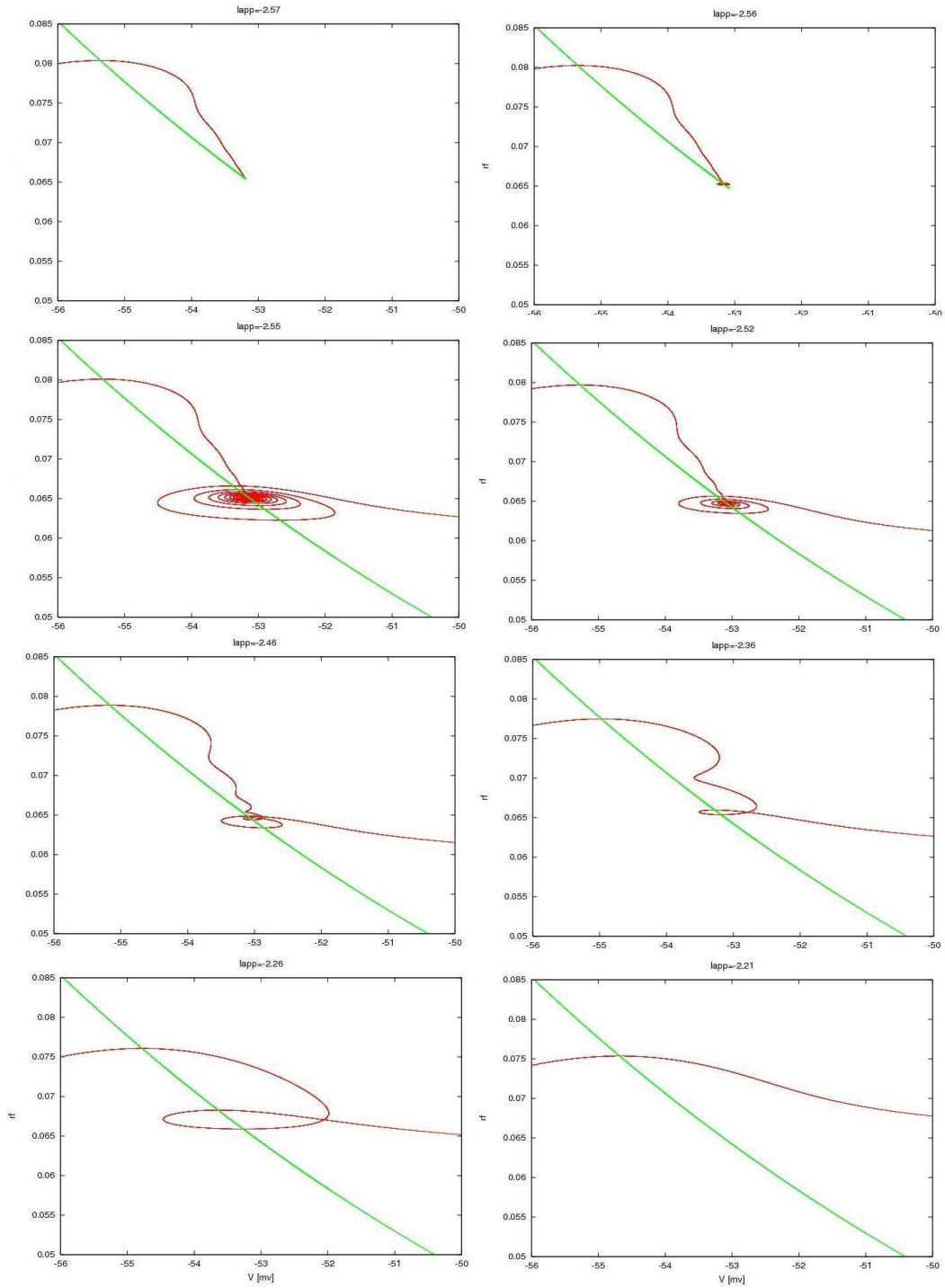


FIGURE 3.2. Approximate recreation of Figure 6 of [5] made using same MATLAB program created for Figure 1.2. Closer look at generating area of STO.

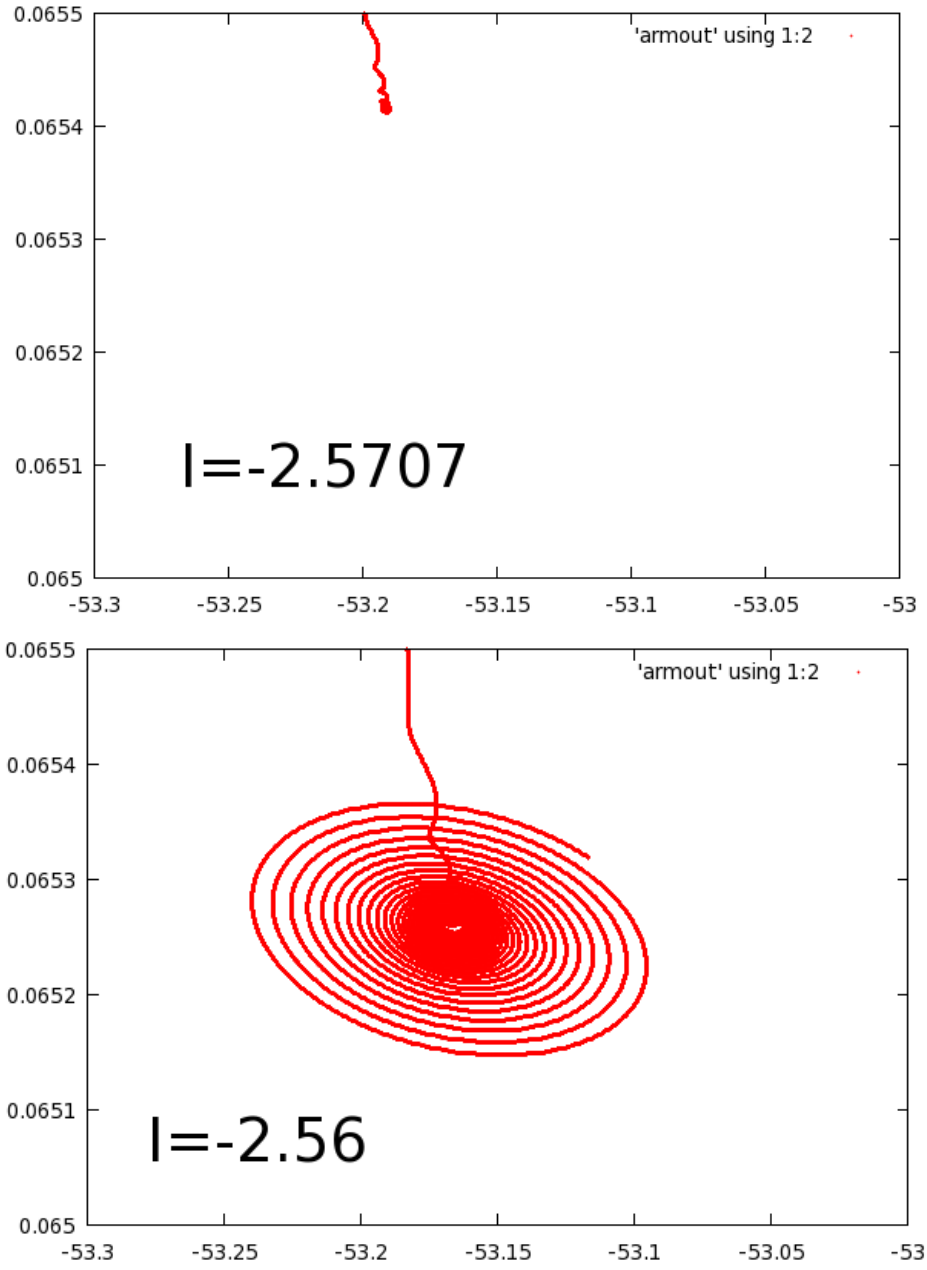


FIGURE 3.3. Close up of transitional area for foll reduced SC model.

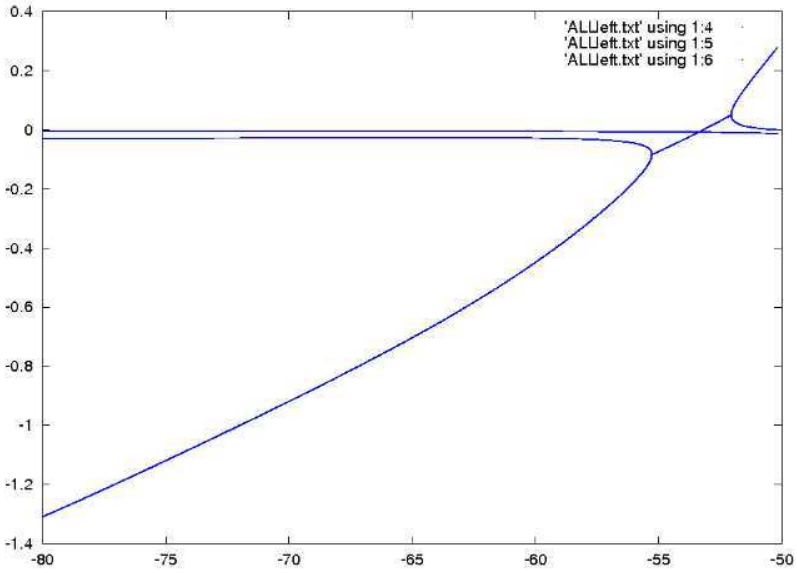


FIGURE 3.4. Eigenvalues for the fast component of the reduced SC model from [5] made using same MATLAB program created for Figure 1.2. Closer look at generating area of STO. This figure and next are recreations of Figure 12 of [5].

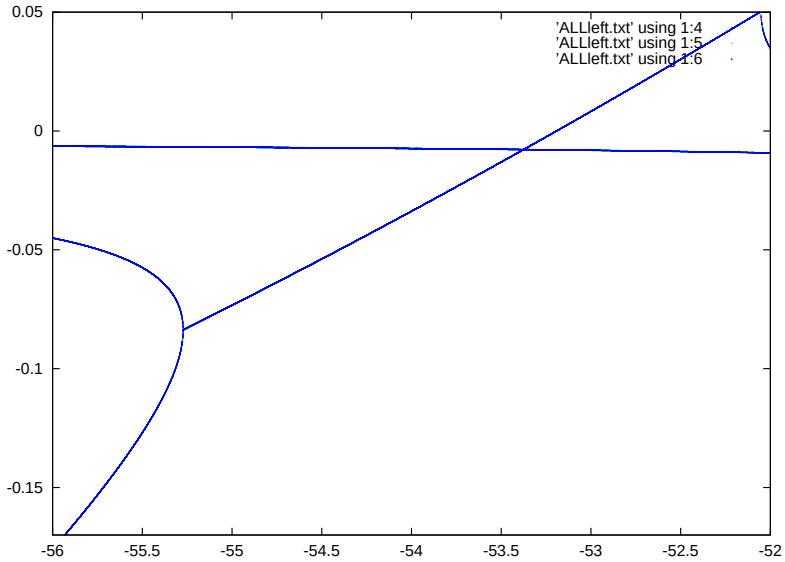
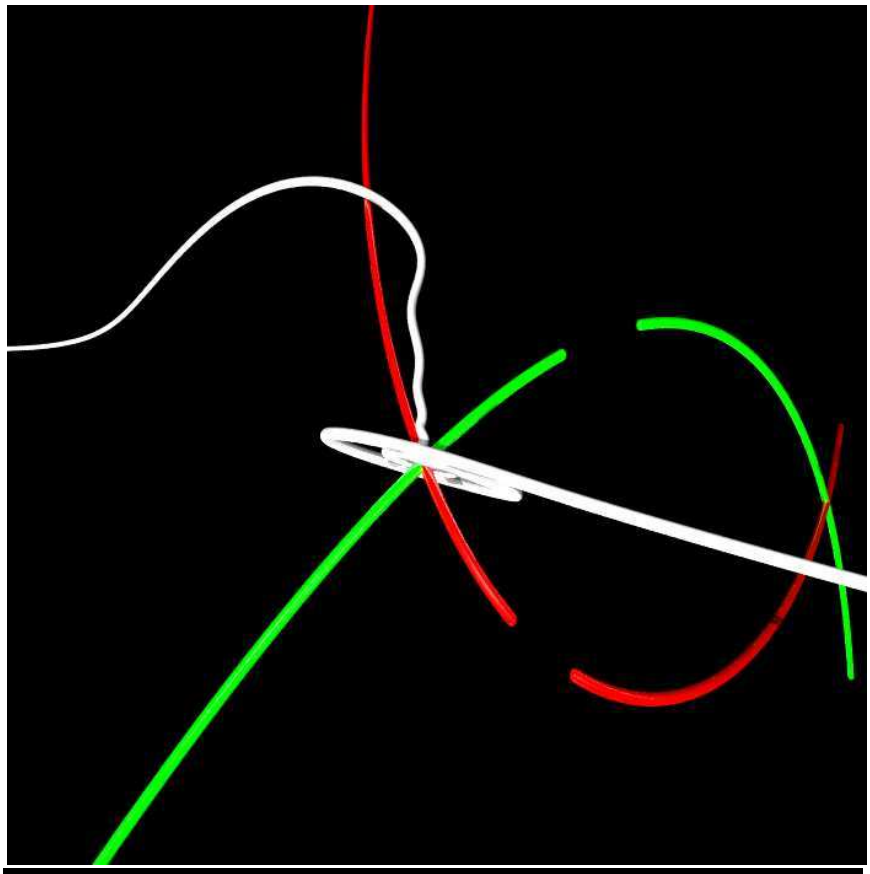
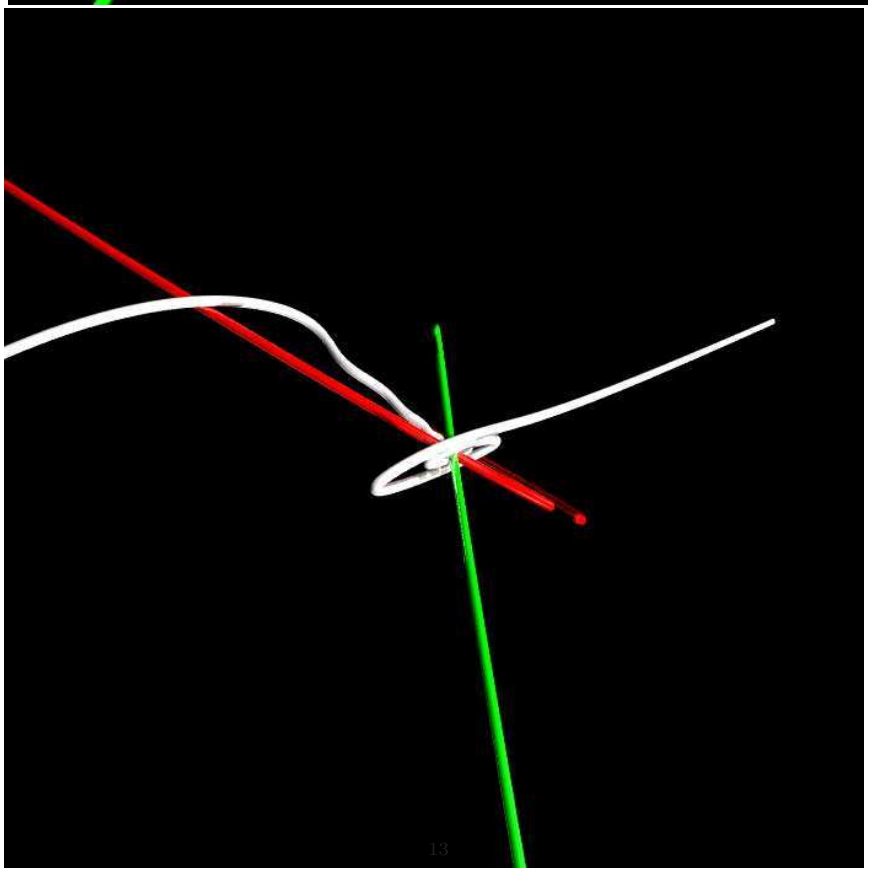
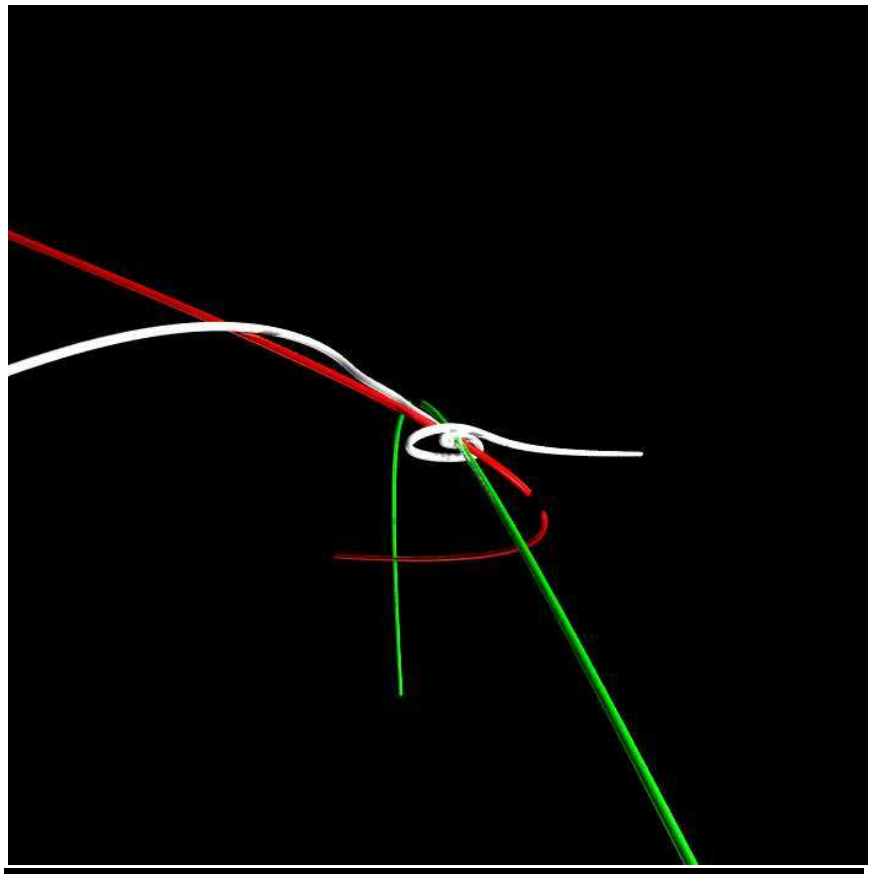
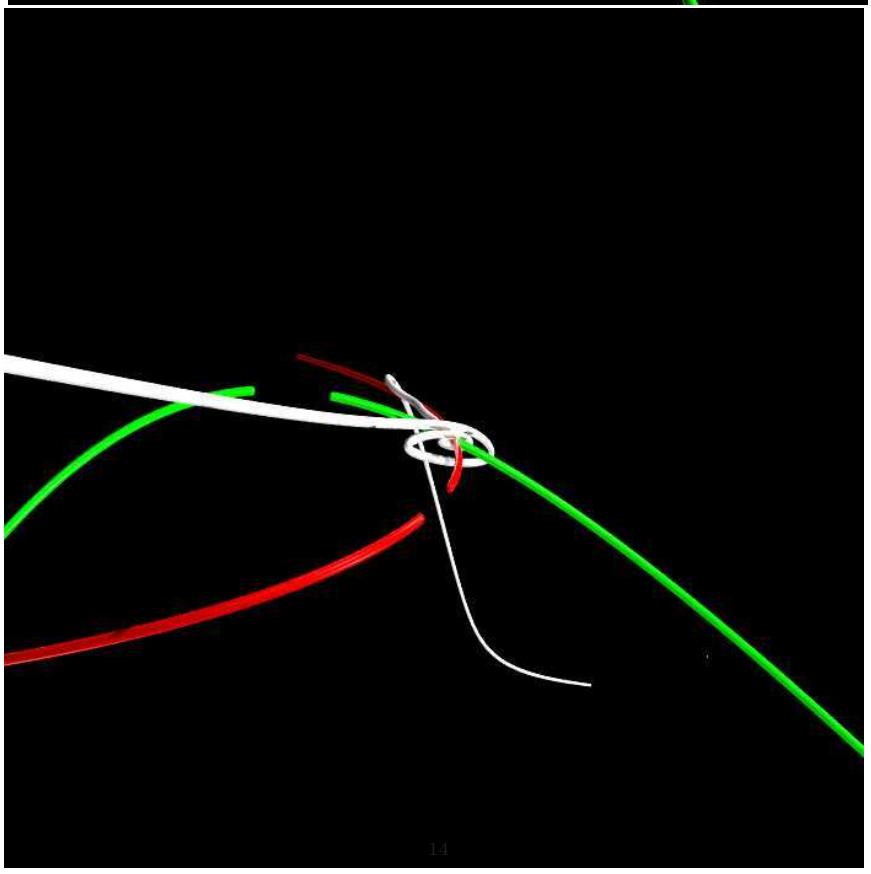


FIGURE 3.5. Zoom in of eigenvalues for the fast component of the reduced SC model from [5] made using same MATLAB program created for Figure 1.2. Closer look at generating area of STO. The signature of a Hopf bifurcation is present (imaginary values not shown).









References

[1] ALONSO, A. & KHLER, C. (1984) *A study of the reciprocal connections between the septum and the entorhinal area using anterograde and retrograde axonal transport methods in the rat brain.* J. Compar. Neurol. 225: 327343.

[2] JONES, R.S.G. (1993) *Entorhinal-hippocampal connections: A speculative view of their function.* Trends Neurosci. 16: 5864.

[3] NIH Press Release, *http://www.nimh.nih.gov/science-news/2007/cortex-area-thinner-in-youth-with-alzheimers-related-gene.shtml*

[4] Angel Alonso & Rodolfo R. Llinás (1989) *Subthreshold Na+-dependent theta-like rhythmicity in stellate cells of entorhinal cortex layer II* Nature 342, 175 - 177

[5] Horacio G. Rotstein, Tim Oppermann, John A. White, & Nancy Kopell (2006) *The dynamic structure underlying subthreshold oscillatory activity and the onset of spikes in a model of medial entorhinal cortex stellate cells* Journal of Computational Neuroscience 21:271-292

[6] Acker C, Kopell N, White J (2003) *Synchronization of strongly coupled excitatory neurons: Relating network behavior to biophysics.* Journal of Computational Neuroscience 15:71-90

[7] Eugen M. Izhikevich, (2007) *Dynamical Systems in Neuroscience*, The MIT Press, Cambridge

Acker_Fig2.m

```
function Acker(input)
global Iapp
Iapp=input;
T_MAX=1000;
step=0.05;
tspan=0:step:T_MAX;
x0=[-0.2828,  0.3208, 0.0513, 0.5841, 0.3, 0.3, 0.3]
[t,x]=ode15s(@Acker_function, tspan, x0);

v=x(:,1);
m=x(:,2);
h=x(:,3);
n=x(:,4);
p=x(:,5);
rf=x(:,6);
rs=x(:,7);

ENa=55;
EK=-90;
EL=-65;
Eh=-20;
GNa=52;
GK=11;
GL=0.5;
Gp=0.5;
Gh=1.5;
C=1;
Ih=Gh*(0.65.*rf + 0.35.*rs).*(v-Eh);
INap=Gp.*p.*(v-ENa);

figure(1)
subplot(4,1,1)
plot(t,v);
axis([0 T_MAX -100 60]);
xlabel ('t (ms)')
ylabel ('V (mv)')
title(Iapp);

subplot(4,1,2);
plot(t,v);
axis([0 T_MAX -60 -50]);
xlabel ('t (ms)')
ylabel ('zoom of V (mv)')

subplot(4,1,3);
plot(t,-Ih);
axis([0 T_MAX 3 4]);
xlabel ('t (ms)')
ylabel ('-Ih')

subplot(4,1,4);
```

```
plot(t,-INap);
axis([0 T_MAX 3 6]);
xlabel ('t (ms)')
ylabel ('-INap')
print -dpng 'acker1'
```

Acker_function (included inside the above and below Acker variations, this is the kernel of the main SC model)

```
function xdot = Acker_function(t,x)
global Iapp
v=x(1);
m=x(2);
h=x(3);
n=x(4);
p=x(5);
rf=x(6);
rs=x(7);
ENa=55;
EK=-90;
EL=-65;
Eh=-20;
GNa=52;
GK=11;
GL=0.5;
Gp=0.5;
Gh=1.5;
C=1;
am=-0.1*(v+23)/(exp(-0.1*(v+23))-1);
bm=4*exp(-(v+48)/18);
ah=0.07*exp(-(v+37)/20);
bh=1/(exp(-0.1*(v+7))+1);
an=-0.01*(v+27)/(exp(-0.1*(v+27))-1);
bn=0.125*exp(-(v+37)/80);
ap=1/(0.15*(1+exp(-(v+38)/6.5)));
bp=exp(-(v+38)/6.5)/(0.15*(1+exp(-(v+38)/6.5)));
rfi=1/(1+exp((v+79.2)/9.78));
trf=0.51/(exp((v-1.7)/10) + exp(-(v+340)/52)) + 1;
rsi=1/(1+exp((v+2.83)/15.9))^58;
trs=5.6/(exp((v-1.7)/14) + exp(-(v+260)/43)) + 1;
tp=0.15;
pinf=1/(1+exp(-(v+38)/6.5));
mi=am/(am+bm);
tm = 1/(am+bm);
dm=(mi -m)/tm;
hi=ah/(ah+bh);
th = 1/(ah+bh);
dh=(hi -h)/th;
ni=an/(an+bn);
tn = 1/(an+bn);
dn=(ni -n)/tn;
pi=ap/(ap+bp);
```

```
tp = 1/(ap+bp);
dp=(pi -p)/tp;
drf=(rfi -rf)/trf;
drs=(rsi -rs)/trs;
dv=Iapp - GNa*m^3*h*(v-ENa) - GK*n^4*(v-EK) - GL*(v-EL)
                          - Gh*(0.65*rf + 0.35*rs)*(v-Eh) - Gp*p*(v-ENa);
xdot=[dv dm dh dn dp drf drs]';
```

Acker_Fig1.m

```
function Acker(input)

global Iapp
Iapp=input;

T_MAX=1000;
step=0.05;
tspan=0:step:T_MAX;
v=-100:0.05:100;
am=-0.1.*(v+23)./(exp(-0.1.*(v+23))-1);
bm=4.*exp(-1.*(v+48)/18);
tm = 1./(am+bm);
ah=0.07.*exp(-1.*(v+37)/20);
bh=1./(exp(-0.1.*(v+7))+1);
th = 1./(ah+bh);
mi=am./(am+bm);
hi=ah./(ah+bh);

an=-0.01.*(v+27)./(exp(-0.1.*(v+27))-1);
bn=0.125.*exp(-(v+37)/80);
tn = 1./(an+bn);
ap=1./(0.15*(1+exp(-1.*(v+38)/6.5)));
bp=exp(-(v+38)/6.5)./(0.15*(1+exp(-1.*(v+38)/6.5)));
tp = 1./(ap+bp);
rfi=1./(1+exp((v+79.2)/9.78));
trf=0.51./(exp((v-1.7)/10) + exp(-1.*(v+340)/52)) + 1;
rsi=1./(1+exp((v+2.83)/15.9)).^58;
trs=5.6./(exp((v-1.7)/14) + exp(-1.*(v+260)/43)) + 1;
tp=0.15;
pinf=1./(1+exp(-(v+38)/6.5));
ni=an./(an+bn);
tn = 1./(an+bn);
ni=an./(an+bn);

miX=(-0.1*(0+23)/(exp(-0.1*(0+23))-1))/( -0.1*(0+23)/(exp(-0.1*(0+23))-1)
                                              + 4*exp(-(0+48)/18));
hiX=( 0.07*exp(-(-20+37)/20) ) / ( 0.07*exp(-(-20+37)/20)
                                          +  1/(exp(-0.1*(-20+7))+1)   );
pinfX=1/(1+exp(-(0+38)/6.5));
niX= (   -0.01.*(60+27)/(exp(-0.1*(60+27))-1)   ) / (    -0.01*(60+27)
                  /(exp(-0.1*(60+27))-1)    +    0.125*exp(-(60+37)/80) );

figure(1)
```

```
hold on

subplot(2,3,1)
hold on
axis([-100 100 -0.2 1.2]);
plot(v,mi, '-b');
text(v(:,1800),mi(:,1800),'\leftarrow m_\infty','HorizontalAlignment','left')
plot(v,hi, '-r');
text(v(:,1500),hi(:,1500),'\leftarrow h_\infty','HorizontalAlignment','left')
plot(v,pinf, '-g');
text(v(:,2000),pinf(:,2000),'p_\infty','HorizontalAlignment','left')
plot(v,ni, '--b');
text(v(:,3000),ni(:,3000)-0.07,'n_\infty','HorizontalAlignment','left')
plot(v,rsi, '--r');
text(v(:,500),rsi(:,500),'\leftarrow r_{s,\infty}','HorizontalAlignment','left')
plot(v,rfi, '--g');
text(v(:,500),rfi(:,500),'\leftarrow r_{f,\infty}','HorizontalAlignment','left')

subplot(2,3,4)

hold on
axis([-65 -45 -0.2 1.2]);
plot(v,mi, '-b');
plot(v,hi, '-r');
plot(v,pinf, '-g');
plot(v,ni, '--b');
plot(v,rsi, '--r');
plot(v,rfi, '--g');


subplot(2,3,2)
hold on
axis([-100 100 0 10]);
plot(v,tm, '-b');
text(v(:,1700),tm(:,1700)+0.4,'\tau_m','HorizontalAlignment','right')
plot(v,th, '-r');
text(v(:,1500),th(:,1500),'\leftarrow \tau_h','HorizontalAlignment','left')
plot(v,tp, '-g');
text(-80,tp+0.4,'\tau_p','HorizontalAlignment','left')
plot(v,tn, '--b');
text(v(:,300),tn(:,300)-0.07,'\tau_n','HorizontalAlignment','left')
plot(v,trs, '--r');
text(v(:,2000),trs(:,2000),'\leftarrow \tau_{r,s}','HorizontalAlignment','left')
plot(v,trf, '--g');
text(v(:,1500),trf(:,1500),'\leftarrow \tau_{r,f}','HorizontalAlignment','left')

subplot(2,3,5)

hold on
axis([-65 -45 0 8]);
plot(v,tm, '-b');
plot(v,th, '-r');
```

```
plot(v,tp, '-g');
plot(v,tn, '--b');
plot(v,trs, '--r');
plot(v,trf, '--g');

subplot(2,3,3)
hold on
axis([-100 100 0 350]);
plot(v,tm, '-b');

plot(v,th, '-r');
plot(v,tp, '-g');
plot(v,tn, '--b');
plot(v,trs, '--r');
text(v(:,1000),trs(:,1000),'\leftarrow \tau_{r,s}','HorizontalAlignment','left')
plot(v,trf, '--g');
text(v(:,1000),trf(:,1000),'\leftarrow \tau_{r,f}','HorizontalAlignment','left')

subplot(2,3,6)

hold on
axis([-65 -45 0 350]);
plot(v,tm, '-b');
plot(v,th, '-r');
plot(v,tp, '-g');
plot(v,tn, '--b');
plot(v,trs, '--r');
plot(v,trf, '--g');
```

Acker_FIG5.m: MATLAB code written to recreate Figure 5 from main journal article.

```
function ARM(input)
global rsNULL
rsNULL=input;
Iapp=-2.5;
ENa=55;
EK=-90;
EL=-65;
Eh=-20;
GNa=52;
GK=11;
GL=0.5;
Gp=0.5;
Gh=1.5;
C=1;
tspan=[0 350];
figure(2)
axis([-80 -40 0 0.15])
hold on
for i=-80:2:-40
x0=[i; 0; rsNULL];
```

```
[t,x]=ode23(@AckerRM_function, tspan, x0);
vv=x(:,1);
nn=x(:,2);
plot(vv,nn, '-black')
x0=[i; 0.15; rsNULL];
[t,x]=ode23(@AckerRM_function, tspan, x0);
vv=x(:,1);
nn=x(:,2);
plot(vv,nn, '-black')
end
for j=0:0.002:0.15
x0=[-80; j; rsNULL];
[t,x]=ode23(@AckerRM_function, tspan, x0);
vv=x(:,1);
nn=x(:,2);
plot(vv,nn, '-black')
x0=[-40; j; rsNULL];
[t,x]=ode23(@AckerRM_function, tspan, x0);
vv=x(:,1);
nn=x(:,2);
plot(vv,nn, '-black')
end
v=-80:0.1:-40;
pinf=1./(1+exp(-1.*(v+38)/6.5));
rfNULL1=(Iapp - GL.*(v-EL) - Gh*(0.35*rsNULL).*(v-Eh)
                      - Gp.*pinf.*(v-ENa))./(Gh*0.65.*(v-Eh));
rfNULL2=1./(1+exp((v+79.2)/9.78));
plot(v,rfNULL1, '-b');
plot(v,rfNULL2, '-r');
title(rsNULL);
print -dpng 'acker1'

function xdot = AckerRM_function(t,x)

global rsNULL
Iapp=-2.5;

v=x(1);
rf=x(2);
rs=x(3);

ENa=55;
EK=-90;
EL=-65;
Eh=-20;
GNa=52;
GK=11;
GL=0.5;
Gp=0.5;
Gh=1.5;
C=1;
```

```
am=-0.1*(v+23)/(exp(-0.1*(v+23))-1);
bm=4*exp(-(v+48)/18);
ah=0.07*exp(-(v+37)/20);
bh=1/(exp(-0.1*(v+7))+1);
an=-0.01*(v+27)/(exp(-0.1*(v+27))-1);
bn=0.125*exp(-(v+37)/80);
ap=1/(0.15*(1+exp(-(v+38)/6.5)));
bp=exp(-(v+38)/6.5)/(0.15*(1+exp(-(v+38)/6.5)));
rfi=1/(1+exp((v+79.2)/9.78));
trf=0.51/(exp((v-1.7)/10) + exp(-(v+340)/52)) + 1;
rsi=1/(1+exp((v+2.83)/15.9))^58;
trs=5.6/(exp((v-1.7)/14) + exp(-(v+260)/43)) + 1;
tp=0.15;
pinf=1/(1+exp(-(v+38)/6.5));
drf=(rfi -rf)/trf;
drs=0;
dv=Iapp - GL*(v-EL) - Gh*(0.65*rf + 0.35*rs)*(v-Eh) - Gp*pinf*(v-ENa);
xdot=[dv drf drs]';
```

eigenrun.m: This is a Maple program designed to solve for the fast part of the reduced SC system's eigenvalues.

```
restart: with(linalg): with(LinearAlgebra): Digits := 13:
fa := fopen("left.txt", WRITE):
fb :=fopen("right.txt",WRITE);
ENa := 55: EK := -90:
EL := -65: Eh := -20:
GNa := 52: GK := 11:
GL := .5: Gp := .5:
Gh := 1.5: C := 1:
Iapp := -2.5:

for rsN from 0 by 0.00001 to 0.1 do
pinf := proc (v) options operator, arrow; 1/(1+exp((-1)*(v+38)/6.5)) end proc:
rfNULL1 := (Iapp-GL*(v-EL)-.35*Gh*rsN*(v-Eh)-Gp*pinf(v)*(v-ENa))/(.65*Gh*(v-Eh)):
rfNULL2 := 1/(1+exp((v+79.2)/(9.78))):
vFIX1a := fsolve(rfNULL1 = rfNULL2, v = -60 .. -51):
rfFIX1a := 1/(1+exp((vFIX1a+79.2)/(9.78))):
vFIX2a := fsolve(rfNULL1 = rfNULL2, v = -51 .. -40):
rfFIX2a := 1/(1+exp((vFIX2a+79.2)/(9.78))):
rfi := 1/(1+exp((v+79.2)/(9.78))):
trf := .51/(exp((v-1.7)*1/10)+exp(-(v+340)*1/52))+1:
rsi := 1/(1+exp((v+2.83)/(15.9)))^58:
trs := 5.6/(exp((v-1.7)*1/14)+exp(-(v+260)*1/43))+1:
drf := proc (v, rf) options operator, arrow; (rfi-rf)/trf end proc:
dv := proc (v, rf) options operator, arrow; Iapp-GL*(v-EL)
               -Gh*(.65*rf+.35*rsN)*(v-Eh)-Gp*pinf(v)*(v-ENa) end proc;
ul := diff(dv(v, rf), v): f1 := unapply(ul, v, rf):
ur := diff(dv(v, rf), rf): f2 := unapply(ur, v, rf):
bl := diff(drf(v, rf), v): f3 := unapply(bl, v, rf):
br := diff(drf(v, rf), rf): f4 := unapply(br, v, rf):
A := matrix([[f1(vFIX1a, rfFIX1a), f2(vFIX1a, rfFIX1a)],
```

```
            [f3(vFIX1a, rfFIX1a), f4(vFIX1a, rfFIX1a)]]):
eigens := {eigenvalues(A)}:
eigen1 := op(1, eigens); eigen2 := op(2, eigens):
B := matrix([[f1(vFIX2a, rfFIX2a), f2(vFIX2a, rfFIX2a)],
            [f3(vFIX2a, rfFIX2a), f4(vFIX2a, rfFIX2a)]]):
eigens2 := {eigenvalues(B)}:
eigen21 := op(1, eigens2); eigen22 := op(2, eigens2):
fprintf(fa, "%f %f %f %f %f %f %f \n", vFIX1a, rfFIX1a, Re(eigen1), Im(eigen1),
                                                    Re(eigen2), Im(eigen2),rsN):
fprintf(fb, "%f %f %f %f %f %f %f \n", vFIX2a, rfFIX2a, Re(eigen21), Im(eigen21),
                                                  Re(eigen22), Im(eigen22),rsN):
end do:
```

eigenall.m: This is a Maple program designed to solve for the reduced SC system's eigenvalues.

```
restart: with(linalg): with(LinearAlgebra): Digits := 15:
fa := fopen("ALLleft.txt", WRITE):
ENa := 55: EK := -90:
EL := -65: Eh := -20:
GNa := 52: GK := 11:
GL := .5: Gp := .5:
Gh := 1.5: C := 1:


for Iapp from -58.1 by 0.0001 to 10 do
pinf := proc (v) options operator, arrow; 1/(1+exp((-1)*(v+38)/6.5)) end proc:
rfi := 1/(1+exp((v+79.2)/(9.78))):
trf := .51/(exp((v-1.7)*1/10)+exp(-(v+340)*1/52))+1:
rsi := 1/(1+exp((v+2.83)/(15.9)))^58:
trs := 5.6/(exp((v-1.7)*1/14)+exp(-(v+260)*1/43))+1:
drf := (rfi-rf)/trf:
dv := Iapp-GL*(v-EL)-Gh*(.65*rf+.35*rs)*(v-Eh)-Gp*pinf(v)*(v-ENa);
drs := (rsi-rs)/trs:
f1 := unapply(dv, (v, rf, rs)):
f2 := unapply(drf, (v, rf, rs)):
f3 := unapply(drs, (v, rf, rs)):
fixed := fsolve({f1(v, rf, rs) = 0, f2(v, rf, rs) = 0, f3(v, rf, rs) = 0},
                                        {v=-80..-50, rf = 0 .. 1, rs = 0 .. 1}):
assign(fixed):
vFIXED := v:
rfFIXED := rf:
rsFIXED := rs:
unassign('v', 'rs', 'rf'):
vec := Vector(3, [f1(v, rf, rs), f2(v, rf, rs), f3(v, rf, rs)]);
A := evalf(subs({v = vFIXED, rf = rfFIXED, rs = rsFIXED}, jacobian(vec, [v, rf, rs]))):
ev := eigenvectors(A):
fprintf(fa, "%f %f %f %f %f %f %f %f %f %f \n", vFIXED, rfFIXED, rsFIXED, Re(ev[1][1]),
             Re(ev[2][1]), Re(ev[3][1]), Im(ev[1][1]), Im(ev[2][1]), Im(ev[3][1]),Iapp ):
end do:
```

all.sh: This is a typical shell script used to create frames for films that run the above MATLAB programs.

```
#!/bin/sh

j=1000
jj=3000
counter=-2570
while [ $counter -lt -2200 ]
do
i=‘echo $counter 1000 | awk ’{ print $1/$2}’‘
echo $i
./ai $i
./null $i

echo "set terminal postscript color" > gnu_pre.dat
echo "set output ’$j.ps’" >> gnu_pre.dat
echo "set title ’Iapp=$i’" >> gnu_pre.dat

echo "set terminal postscript color" > gnu_pre2.dat
echo "set output ’$jj.ps’" >> gnu_pre2.dat
echo "set title ’Iapp=$i’" >> gnu_pre2.dat

cat gnu_pre.dat gnu.dat > gnuout.dat
cat gnu_pre2.dat gnu2.dat > gnuout2.dat

gnuplot gnuout.dat
gnuplot gnuout2.dat

mogrify -format jpg $j.ps
mogrify -format jpg $jj.ps
mogrify -rotate 90 $j.jpg
mogrify -rotate 90 $jj.jpg

rm $j.ps
rm $jj.ps


let j=j+1
let jj=jj+1
let counter=counter+1

done
```

ARM_input.c: This C code was used, in conjunction with the above shell script, to create a series of images for the full reduced SC system.

```
//An ARM iterator
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
```

```
#include "nr.h"
#include "nrutil.h"
#include "nrutil.c"
//#include "rk4.c"

void rk4(double a, double y[], double dydx[], int n, double x, double h, double yout[],
void (*derivs)(double,double, double [], double []))
{
int i;
double xh,hh,h6,*dym,*dyt,*yt;

dym=dvector(1,n);
dyt=dvector(1,n);
yt=dvector(1,n);
hh=h*0.5;
h6=h/6.0;
xh=x+hh;
for (i=1;i<=n;i++) yt[i]=y[i]+hh*dydx[i];
(*derivs)(a,xh,yt,dyt);
for (i=1;i<=n;i++) yt[i]=y[i]+hh*dyt[i];
(*derivs)(a,xh,yt,dym);
for (i=1;i<=n;i++) {
yt[i]=y[i]+h*dym[i];
dym[i] += dyt[i];
}
(*derivs)(a,x+h,yt,dyt);
for (i=1;i<=n;i++)
yout[i]=y[i]+h6*(dydx[i]+dyt[i]+2.0*dym[i]);
free_vector(yt,1,n);
free_vector(dyt,1,n);
free_vector(dym,1,n);
}


int main(int argc, char *argv[]){

  int i1,i2,i3,j,n1,n,N;  double x,yold; double h=0.01; double *y;
double *dydx; double *yout;
n=3;

 int flag=100;

 double a=atof(argv[1]);
 //  printf("%f \n", a);
FILE* fd = fopen("armout", "w");
FILE* fe = fopen("armout2", "w");

y = dvector(1,n);  dydx = dvector(1,n); yout = dvector(1,n);

 y[1]=-80;
 y[2]=0.0;
 y[3]=0.0;
```

```
 x=0.0;
 for(i2=1; i2<=400000; i2++){
    x = x + h;
    derivs(a,x,y,dydx);
    rk4(a,y,dydx,n,x,h,yout,derivs);
    for(j=1; j<=n; j++){y[j]=yout[j];}
    //    yold = y[1];
    // if(y[1]<-40){

    ///GRAB THE NULLCLINES

double Iapp=a;
//printf("%f \n", Iapp);

double v=y[1];
double rf=y[2];
double rs=y[3];

double ENa=55;
double EK=-90;
double EL=-65;
double Eh=-20;
double GNa=52;
double GK=11;
double GL=0.5;
double Gp=0.5;
double Gh=1.5;
double C=1;

double  am=-0.1*(v+23)/(exp(-0.1*(v+23))-1);
double  bm=4*exp(-(v+48)/18);
double  ah=0.07*exp(-(v+37)/20);
double  bh=1/(exp(-0.1*(v+7))+1);
double  an=-0.01*(v+27)/(exp(-0.1*(v+27))-1);
double  bn=0.125*exp(-(v+37)/80);
double  ap=1/(0.15*(1+exp(-(v+38)/6.5)));
double  bp=exp(-(v+38)/6.5)/(0.15*(1+exp(-(v+38)/6.5)));
double  rfi=1/(1+exp((v+79.2)/9.78));
double  trf=0.51/(exp((v-1.7)/10) + exp(-(v+340)/52)) + 1;
 double  rsi=pow(1/(1+exp((v+2.83)/15.9)),58);
double  trs=5.6/(exp((v-1.7)/14) + exp(-(v+260)/43)) + 1;
double  tp=0.15;
double  pinf=1/(1+exp(-(v+38)/6.5));

double rfNULL1=(Iapp - GL*(v-EL) - Gh*(0.35*rs)*(v-Eh) - Gp*pinf*(v-ENa))
                                                        /(Gh*0.65*(v-Eh));
double rfNULL2=1/(1+exp((v+79.2)/9.78));
double rsNULL1=(Iapp - GL*(v-EL) - Gh*(0.65*rf)*(v-Eh) - Gp*pinf*(v-ENa))
                                                        /(Gh*0.35*(v-Eh));
double rsNULL2=pow(1/(1+exp((v+2.83)/15.9)),58);
```

```
//double rfNULL3=pow(1/(1+exp((v+2.83)/15.9)),58);

////////////////////////////////////////////////////////////

 fprintf(fd,"%f %f %f %f \n", y[1], y[2], y[3], x);//}
 if(y[1]>-30){y[1]=-80; y[2]=0; y[3]=0; flag=0;}
 if(flag>10){ fprintf(fe,"%f %f %f %f %f %f %f \n", y[1], y[2], y[3],
                          x,rfNULL2, rfNULL2-rfNULL1,rsNULL2-rsNULL1);}
   }  //ends i2 loop
  printf("%f \n", y[1]);
}  //ends main

void derivs(double a, double x, double y[], double dydx[]){

double Iapp=a;
//printf("%f \n", Iapp);

double v=y[1];
double rf=y[2];
double rs=y[3];

double ENa=55;
double EK=-90;
double EL=-65;
double Eh=-20;
double GNa=52;
double GK=11;
double GL=0.5;
double Gp=0.5;
double Gh=1.5;
double C=1;

double  am=-0.1*(v+23)/(exp(-0.1*(v+23))-1);
double  bm=4*exp(-(v+48)/18);
double  ah=0.07*exp(-(v+37)/20);
double  bh=1/(exp(-0.1*(v+7))+1);
double  an=-0.01*(v+27)/(exp(-0.1*(v+27))-1);
double  bn=0.125*exp(-(v+37)/80);
double  ap=1/(0.15*(1+exp(-(v+38)/6.5)));
double  bp=exp(-(v+38)/6.5)/(0.15*(1+exp(-(v+38)/6.5)));
double  rfi=1/(1+exp((v+79.2)/9.78));
double  trf=0.51/(exp((v-1.7)/10) + exp(-(v+340)/52)) + 1;
 double  rsi=pow(1/(1+exp((v+2.83)/15.9)),58);
double  trs=5.6/(exp((v-1.7)/14) + exp(-(v+260)/43)) + 1;
double  tp=0.15;
double  pinf=1/(1+exp(-(v+38)/6.5));

dydx[1]=Iapp - GL*(y[1]-EL) - Gh*(0.65*y[2] + 0.35*y[3])*(y[1]-Eh) - Gp*pinf*(y[1]-ENa);
dydx[2]=(rfi -y[2])/trf;
dydx[3]=(rsi -y[3])/trs;
}
```