MECH 6313 - Homework 1

Jonas Wagner

2021, February 1

1 Problem 1 - Duffing's Equation

Duffings Equation is exhibits chaotic behavior with certain parameters. It is discribed by the following:

$$\ddot{y} + \delta\dot{y} - y + y^3 = \alpha\cos(\omega_t t)$$

Problem: Simulate the equation for $\delta = 0.05, \alpha = 0.4$, and $\omega_t = 1.3$.

Solution: In matlab the nlsys class (something I have been developing to help in nonlin system simulation - https://github.com/jonaswagner2826/nlsys) was used to simulate the system and plot the following phase plots and time responses. The MATLAB code for this assignment is available in Appendix1

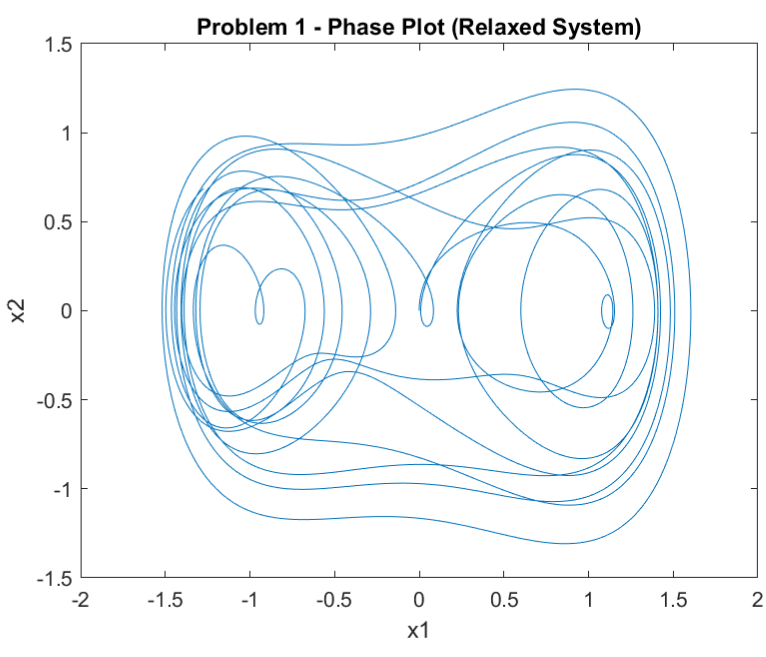

Figure 1: Phase Plot for the Relaxed System

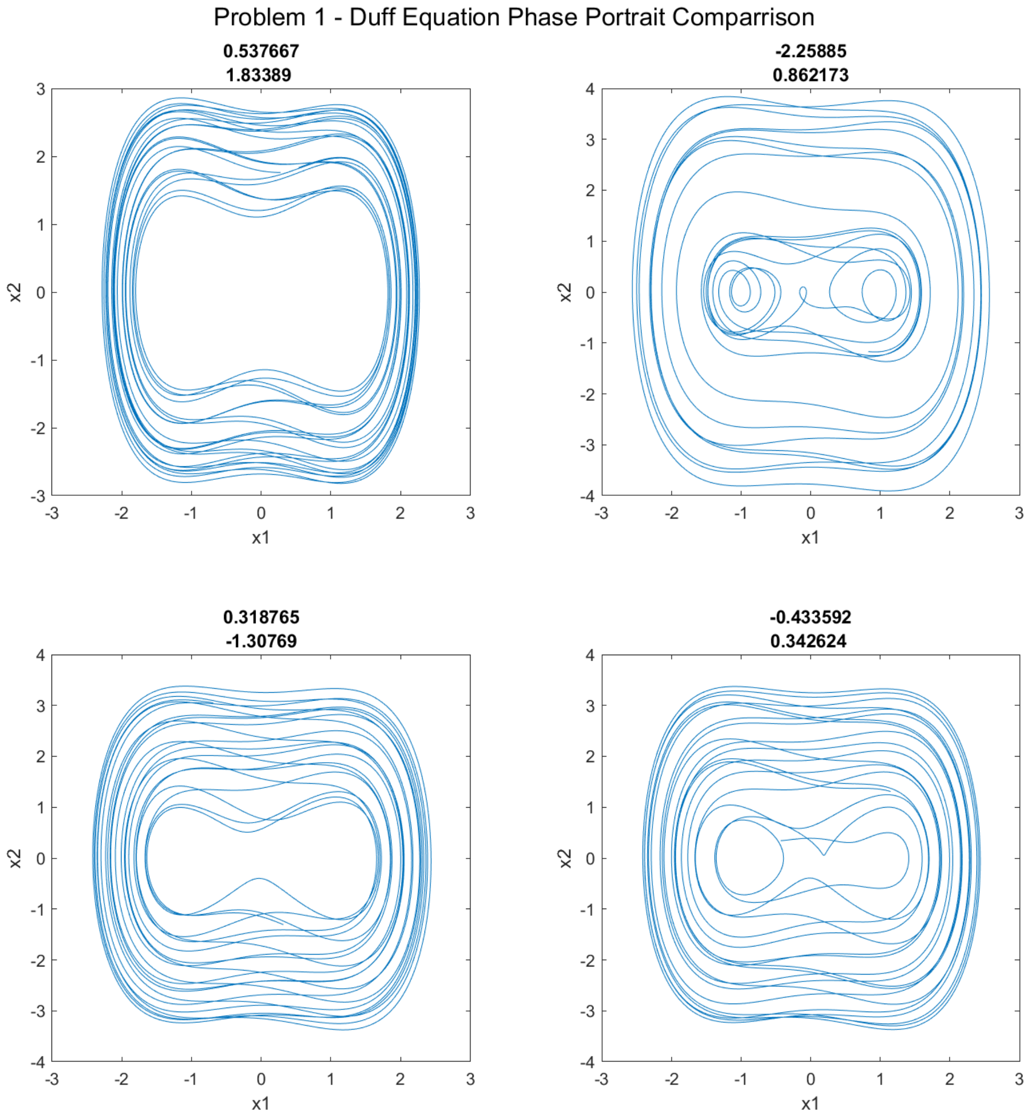

Figure 2: Phase Plot for multiple initial conditions

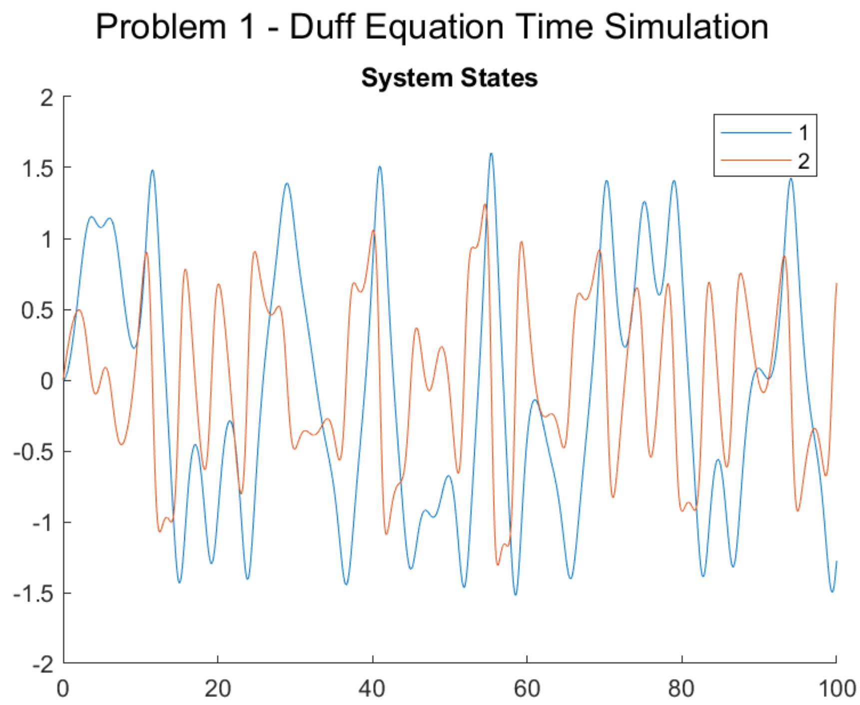

Figure 3: Plot of the relaxed system vs time

Discussion:

Both the phase portraits and time response of the Duffings equation indicate a fairly chaotic behavior, however it does appear to contentiously rotate around the origin in a periodic fashion (possibly due to the sinusoidal input). Either way, it is difficult to recognize a predictable behavior, and neither decays or explodes predictably.

2 Problem 2 - Van Der Pol Equations

Problem: The van der Pole equation is as follows:

$$\ddot{y} + \left(y^2 - 1\right) * \dot{y} + y = 0$$

Plot the phase portrait, time dependence and compare with the response of Duffing's equations.

Solution: In matlab the nlsys class (something I have been developing to help in nonlin system simulation - https://github.com/jonaswagner2826/nlsys) was used to simulate the system and plot the following phase plots and time responses. The MATLAB code for this assignment is available in Appendix1

2.1 Van Der Pol Simulation

2.1.1 Phase Portraits

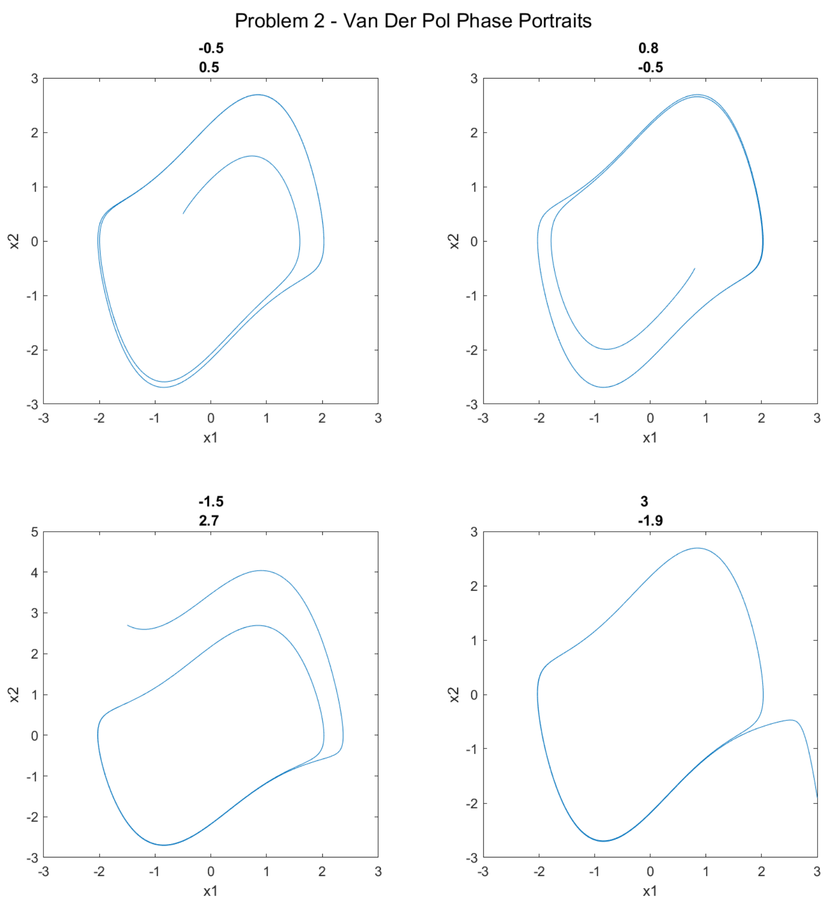

Figure 4: Phase Plot for multiple initial conditions

2.1.2 Time Response

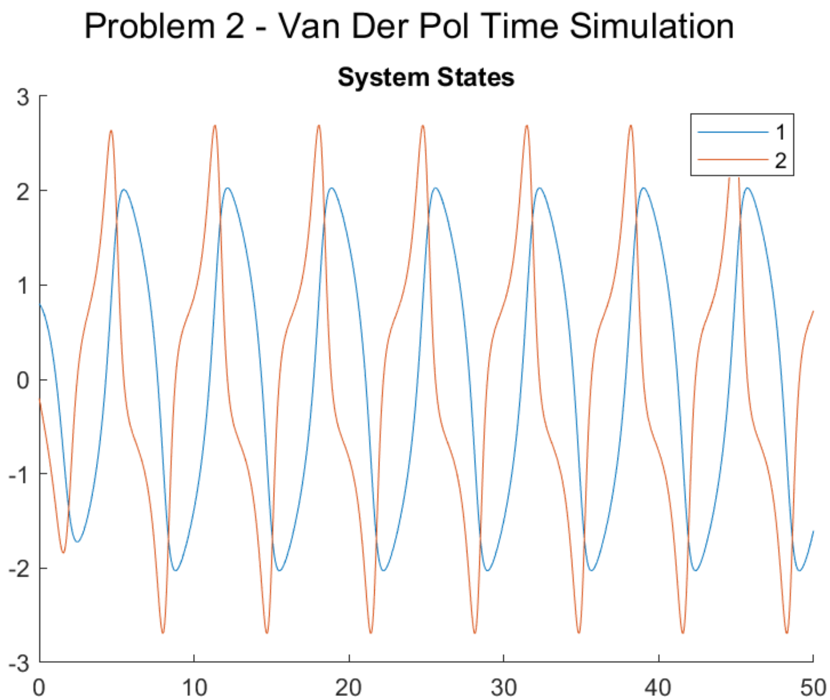

Figure 5: Phase Plot for multiple initial conditions

Discussion:

Unlike the results from the duffings equation, the Van Der Pol equations produce a very predictable response. Outside of the equilibrium point located at the origin, the unforced response from any initial equation appears to decay to the exact same periodic motion. Looking at the time response plot, the system states demonstrate a periodic response consistant with expectations from the phase portrait.

2.2 Negative Van Der Pole Equation

Modifying the nonlinear term of the van der pole equations results in the following differential equation:

$$\ddot{y} - \left(y^2 - 1\right) * \dot{y} + y = 0$$

2.2.1 System Stability at the Origin

The Van Der Pol equation can be linearized at the origin by using a taylor's series expansion of the state-space model. The first term can be found by taking the jacobian and evaluating it with $x = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$.

Letting $x_1 = y$ and $x_2 = \dot{y}$, the following state-space model is derived:

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) = \begin{bmatrix} x_2 \\ -(x_1^2 - 1)x_2 - x_1 \end{bmatrix}$$

The jacobians for the A matrix can then be found as:

$$\begin{aligned} A &= \begin{bmatrix} \frac{\mathrm{d}f_1}{\mathrm{d}x_1} & \frac{\mathrm{d}f_1}{\mathrm{d}x_2} \\ \frac{\mathrm{d}f_2}{\mathrm{d}x_1} & \frac{\mathrm{d}f_2}{\mathrm{d}x_2} \end{bmatrix}\Bigg|_{\mathbf{x}=\mathbf{x}_0} \\ &= \begin{bmatrix} 0 & 1 \\ -2x_1x_2 - 1 & -x_1^2 + 1 \end{bmatrix}\Bigg|_{\mathbf{x}=\mathbf{x}_0} \\ &= \begin{bmatrix} 0 & 1 \\ -1 & 1 \end{bmatrix} \end{aligned}$$

The stability of this matrix can then be determined by looking at the eigenvalues:

$$\Lambda\{A\} = 0.5 \pm j0.866$$

Since $\Re\{\Lambda\{A\}\} > 0$, the linearized system is said to be unstable at the origin.

2.2.2 Phase Portraits

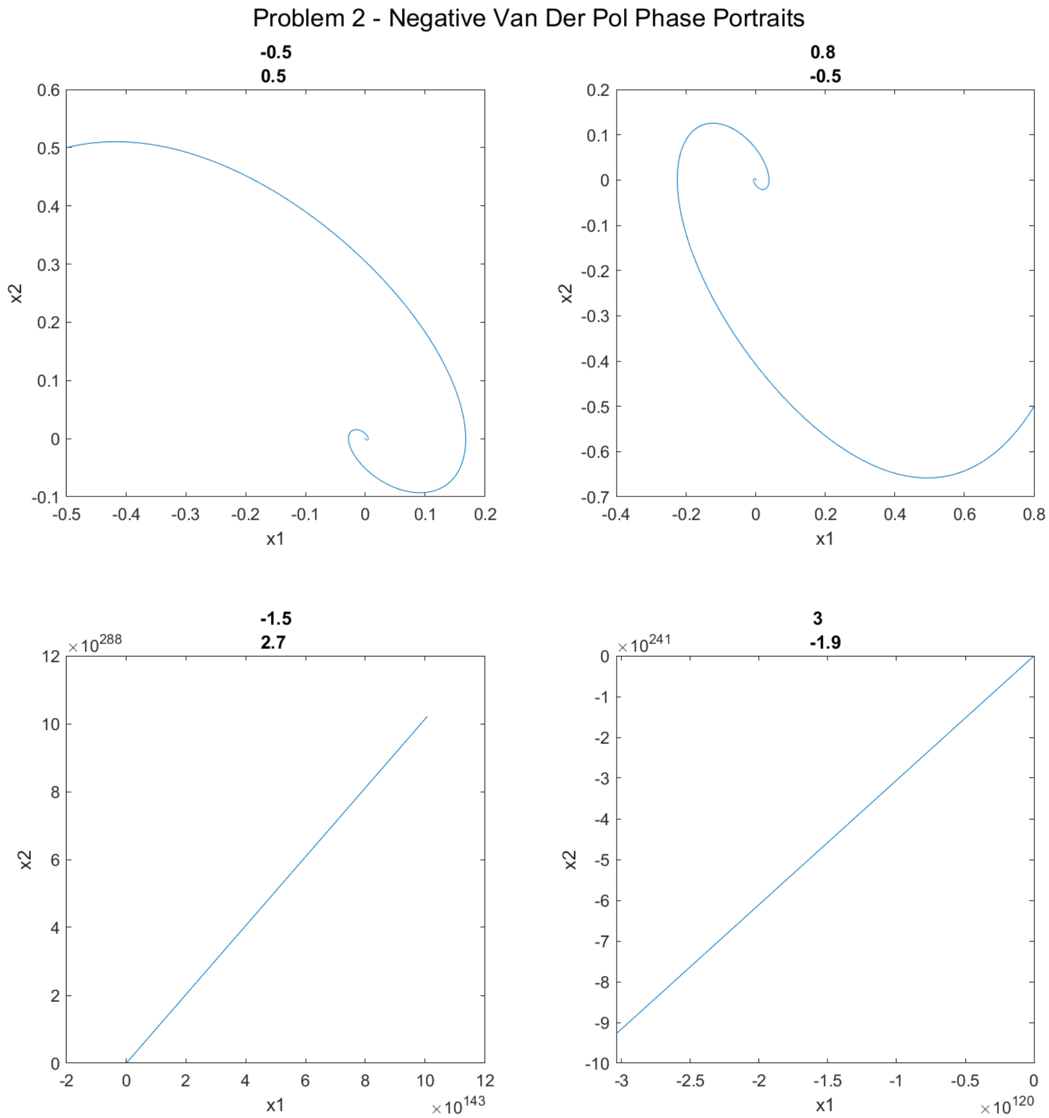

Figure 6: Phase Plot for multiple initial conditions

2.2.3 Time Response

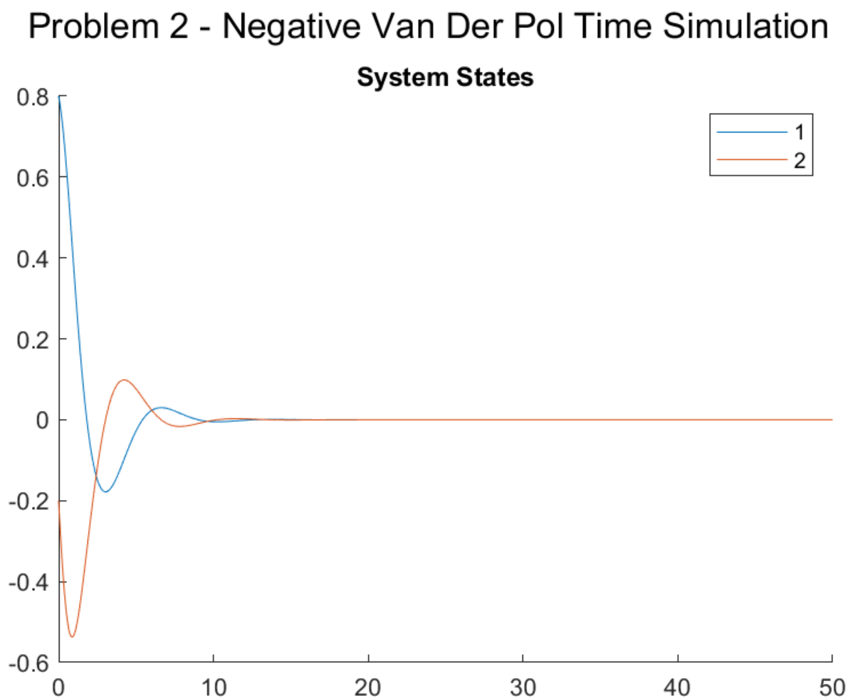

Figure 7: Phase Plot for multiple initial conditions

Discussion: Unlike in the positive van der pol equation, when the nonlinear term is set to be negative it reacts almost directly oposite of the positive one. As was discovered, the origin within the negative system is asymptotically stable, but this is not true in general. From the simulations it is apparent that the same boundary that is the steady-state response for the positve van der pol equation is the boundary of instability. The system is indead localy stable within that boundary and decays to zero, but outside it blows up. It actually doesn't blow up exactly as expected either, depending on the initial condition it won't necessarily just explode out to the extremes of its own quadrant but instead it changes with the initial condition.

3 Problem 3 - Magnetic Suspension System

An electromagnet suspends a ball and is controlled by a feedback system based on the measured postion. The equation of motion for the ball is given as:

$$m\ddot{y} = -k\dot{y} + mg + F(y,i) \tag{1}$$

where m is the mass of the ball, $y \geq 0$ is the vertical position of the ball relative to the electro magnet, k is the friction coeficent, g is the accelleration of gravity, and $F(y,i)$ is the force generated by the magnet dependent on the position and current (i).

The inductance of the electromagnet is given as a function of the ball's position as:

$$L(y) = L_1 + \frac{L_0}{1 + y/a} \tag{2}$$

where $L_1, L_0, a > 0$.

The energy stored within the electromagnet is given as a function of inductance:

$$E(y,i) = \frac{1}{2}L(y)i^2 = \frac{1}{2}\left(L_1 + \frac{L_0}{1 + y/a}\right)i^2 \tag{3}$$

The force on the ball is then calculated as the derivative of energy with respect to position:

$$F(y,i) = \frac{\partial E}{\partial y} = \frac{-L_0 i^2}{2a(1 + y/a)^2} \tag{4}$$

The electric circuit controlling the electro magnet is governed by KVL as:

$$v = \dot{\phi} + Ri \tag{5}$$

where v is the input voltage, R is the resistance and $\phi = L(y)i$.

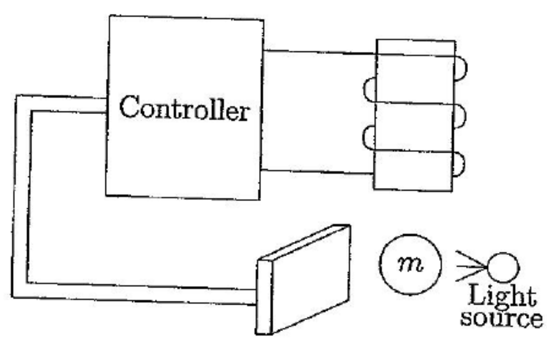

Figure 8: Magnetic suspension system diagram

3.1 State Space Model

Let the following state variables be defined:

$$x_1 = y \qquad x_2 = \dot{y} \qquad x_3 = i \qquad u = v$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x3 \end{bmatrix} = \begin{bmatrix} y \\ \dot{y} \\ i \end{bmatrix} \qquad \mathbf{u} = \begin{bmatrix} u \end{bmatrix} = \begin{bmatrix} v \end{bmatrix} \tag{6}$$

From the definition, the following state equation can be stated directly:

$$\dot{x}_1 = x_2 \tag{7}$$

A second equation of motion can be derived from (1) and (4):

$$\begin{aligned} \ddot{y} &= -\left(\frac{k}{m}\right)\, \dot{y} + g + \frac{1}{m} F(y,i) \\ &= -\left(\frac{k}{m}\right)\dot{y} + g + \frac{1}{m}\; \frac{-\,L_0(i)^2}{2a(1+y/a)^2} \\ \dot{x}_2 &= -\left(\frac{k}{m}\right) x_2 + g + \frac{-\,L_0(x_3)^2}{2am\left(1+\frac{x_1}{a}\right)^2} \end{aligned} \tag{8}$$

From (5) the following can be derived:

$$\begin{aligned} v &= \frac{\partial}{\partial t}(L(y)i) + Ri \\ &= \left(\dot{L}(y)i + L(y)\dot{i}\right) + Ri \end{aligned} \tag{9}$$

$\dot{L}(y)$ can be calculated from (4):

$$\begin{aligned} \dot{L}(y) &= \frac{\partial}{\partial t}\left(L_1 + \frac{L_0}{1+y/a}\right) \\ &= -\frac{L_0\ a\ \dot{y}}{(a+y)^2} \end{aligned} \tag{10}$$

Substituting (4) and (10) into (9), the following can be obtained:

$$\begin{aligned} v &= -\frac{L_0\ a\ \dot{y}\ i}{(a+y)^2} + \left(L_1 + \frac{L_0}{1+y/a}\right)\dot{i} + Ri \\ \left(\frac{L_1(1+y/a)+L_0}{1+y/a}\right)\dot{i} &= \frac{L_0\ a\ \dot{y}\ i}{(a+y)^2} - Ri + v \\ \dot{i} &= \left(\frac{1+y/a}{L_1(1+y/a)+L_0}\right)\left(\frac{L_0\ a\ \dot{y}\ i}{(a+y)^2} - Ri + v\right) \\ \dot{x}_3 &= \left(\frac{1+x_1/a}{L_1(1+x_1/a)+L_0}\right)\left(\frac{L_0\ a\ x_2\ i}{(a+x_1)^2} - Rx_3 + u\right) \end{aligned} \tag{11}$$

The full state-space model is given as:

$$\dot{\mathbf{x}} = \begin{bmatrix} x_2 \\ -\left(\frac{k}{m}\right)x_2 + g + \dfrac{-\,L_0(x_3)^2}{2am\left(1+\frac{x_1}{a}\right)^2} \\ \left(\dfrac{1+x_1/a}{L_1(1+x_1/a)+L_0}\right)\left(\dfrac{L_0\ a\ x_2\ i}{(a+x_1)^2} - Rx_3 + u\right) \end{bmatrix} \tag{12}$$

3.2 Steady-State Solution

The steady-state equation will occur when $\dot{x_1} = \dot{x_2} = 0$, so the state-space model can be used to find this condition at a certain position, $r > 0$.

Since $\dot{x_1} = 0$ and $y = r$, the following can be defined:

$$x_1 = y = r \tag{13}$$

$$x_2 = \dot{x_1} = 0 \tag{14}$$

$L_{ss}(r)$ is then calculated from (2):

$$L_{ss}(r) = L_1 + \frac{L_0}{1 + r/a} \tag{15}$$

Similarly, $\dot{L}_{ss}(r)$ is then calculated from (10):

$$\dot{L}_{ss}(r) = -\frac{L_0\ a\ (0)}{(a+r)^2} = 0 \tag{16}$$

Referring back to (9), v and i can be related by the following:

$$\begin{aligned} v &= \left(\dot{L}(y)i + L(y)\dot{i}\right) + Ri \\ &= \left((0)i + \left(L_1 + \frac{L_0}{1+r/a}\right)\dot{i}\right) + Ri \\ &= \left(L_1 + \frac{L_0}{1+r/a}\right)\dot{i} + Ri \end{aligned} \tag{17}$$

If you make the assumption that I_{ss} is a constant (something I don't believe),

$$V_{ss} = RI_{ss} \tag{18}$$

$$I_{ss} = \frac{V_{ss}}{R} \tag{19}$$

In this case, F_{ss} can be calculated from (4) and (19):

$$\begin{aligned} F_{ss} &= \frac{-L_0\left(\frac{V_{ss}}{R}\right)^2}{2a(1+r/a)^2} \\ &= \frac{-L_0V_{ss}^2}{2aR^2(1+r/a)^2} \end{aligned} \tag{20}$$

In order to maintain static conditions, $F_s s = -mg$, so V_{ss} can be found as:

$$F_{ss} = mg = \frac{-L_0 V_{ss}^2}{2aR^2(1 + r/a)^2} \tag{21}$$

$$V_{ss}^2 = \frac{-2amgR^2(1 + r/a)^2}{L_0} \tag{22}$$

$$V_{ss} = \sqrt{\left(\frac{-2a(mg)\left(R^2\right)}{L_0}\right)(1 + r/a)^2} \tag{23}$$

$I_s s$ is then calculated from (19) and (23):

$$\begin{aligned} I_{ss} &= \frac{1}{R}\sqrt{\left(\frac{-2a(mg)\left(R^2\right)}{L_0}\right)(1 + r/a)^2} \\ &= \sqrt{\left(\frac{-2a(mg)}{L_0}\right)(1 + r/a)^2} \end{aligned} \tag{24}$$

4 Problem 4 - Bifurcation Examples

Additional functionality was added to the nlsys class in order to introduce bifurcation plots.

4.1 Pblm 3.4.2

$$\dot{x} = rx - \sinh(x)$$

As is evident in the bifurcation diagram, the pitchfork diagram is supercritical. r_c was found by differentiating $f(x, r)$, solving for x_c, substituting back into $f(x, r)$ and then solving for $r_c = 1$.

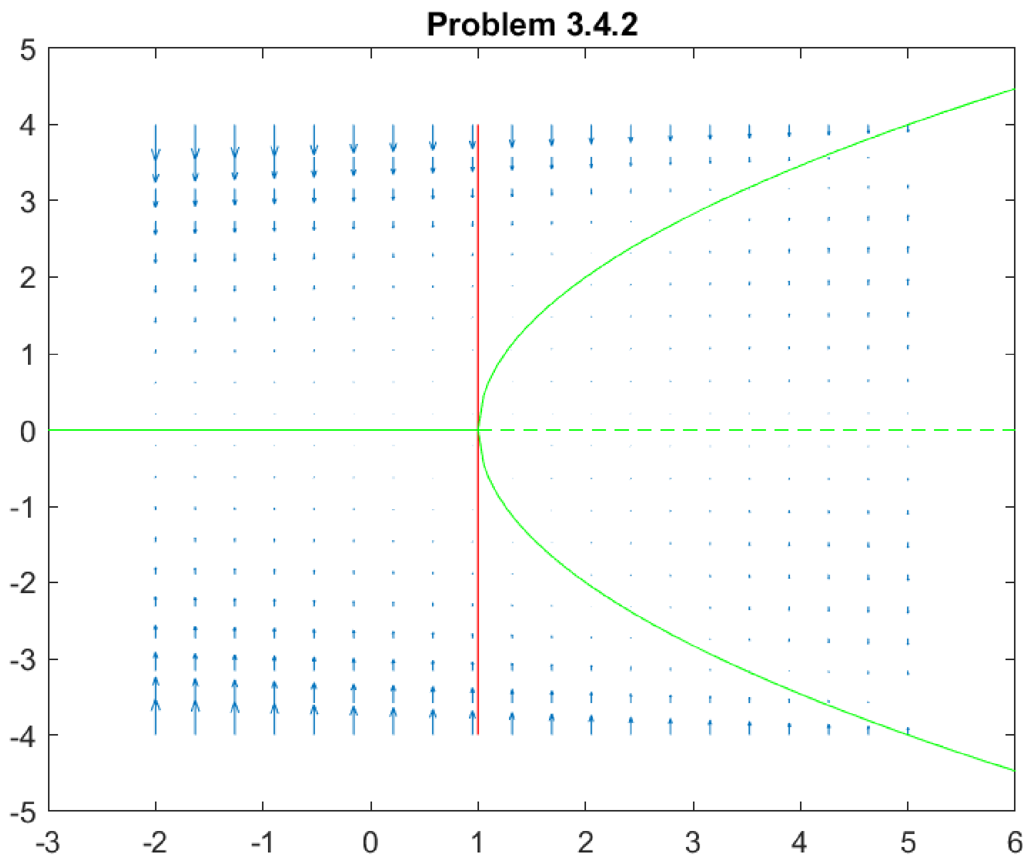

Figure 9: Bifurcation Diagram for problem 3.4.2

4.2 Pblm 3.4.4

$$\dot{x} = x + \frac{rx}{1+x^2}$$

As is evident in the bifurcation diagram, the pitchfork diagram is subcritical. r_c was found by differentiating $f(x,r)$, solving for x_c, substituting back into $f(x,r)$ and then solving for $r_c = -1$.

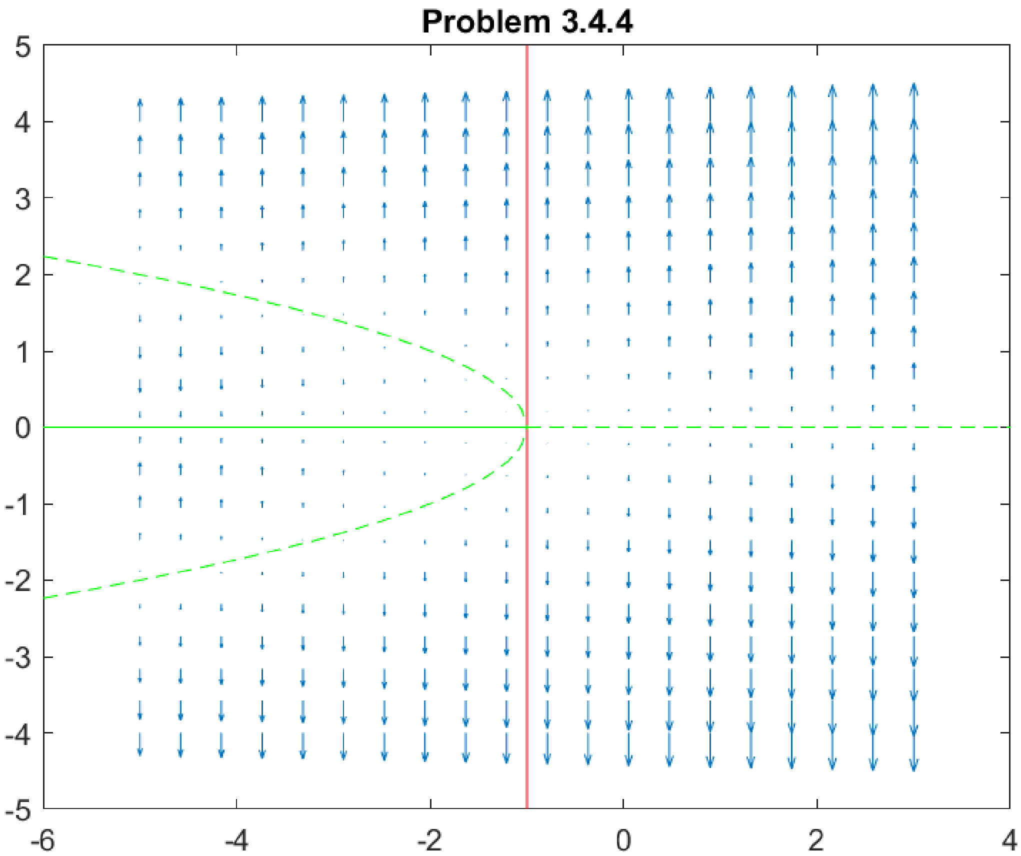

Figure 10: Bifurcation Diagram for problem 3.4.4

4.3 Pblm 3.4.7

$$\dot{x} = 5 - re^{-x^2}$$

As is evident in the bifurcation diagram, is a saddle node bifurcation with $r_c = 1$.

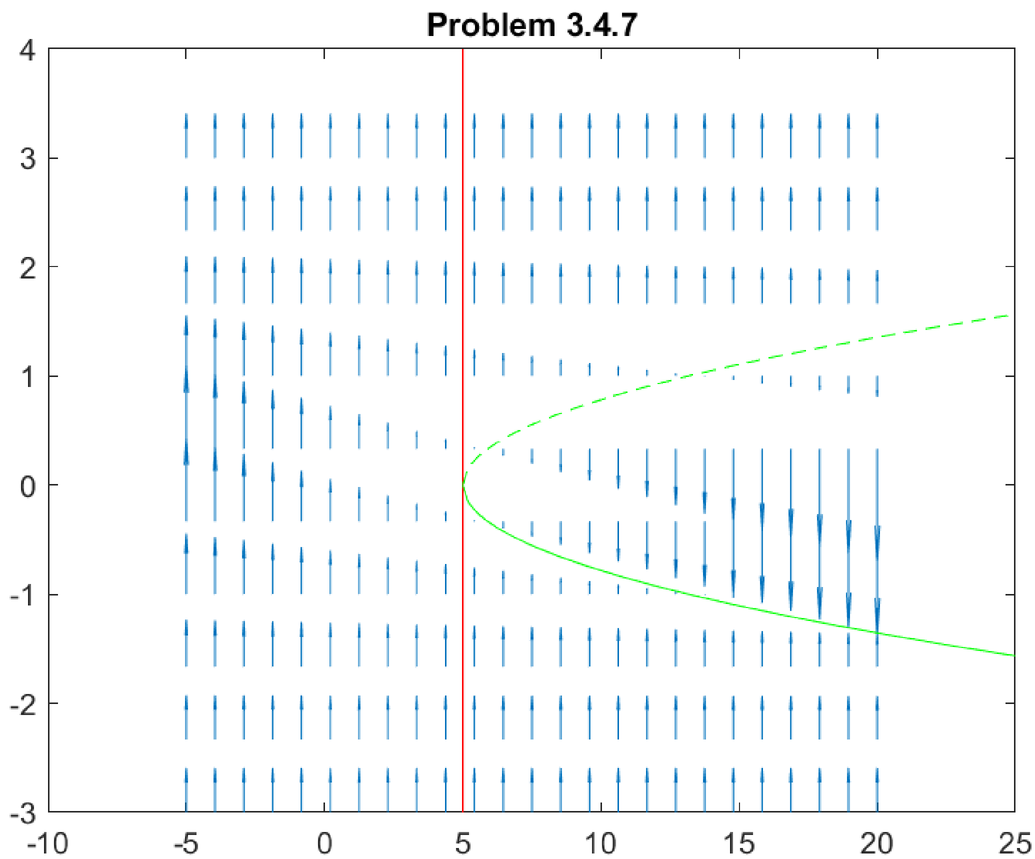

Figure 11: Bifurcation Diagram for problem 3.4.7

4.4 Pblm 3.4.9

$$\dot{x} = x + \tanh(rx)$$

As is evident in the bifurcation diagram, the pitchfork diagram is subcritical. r_c was found by differentiating $f(x,r)$, solving for x_c, substituting back into $f(x,r)$ and then solving for $r_c = -1$.

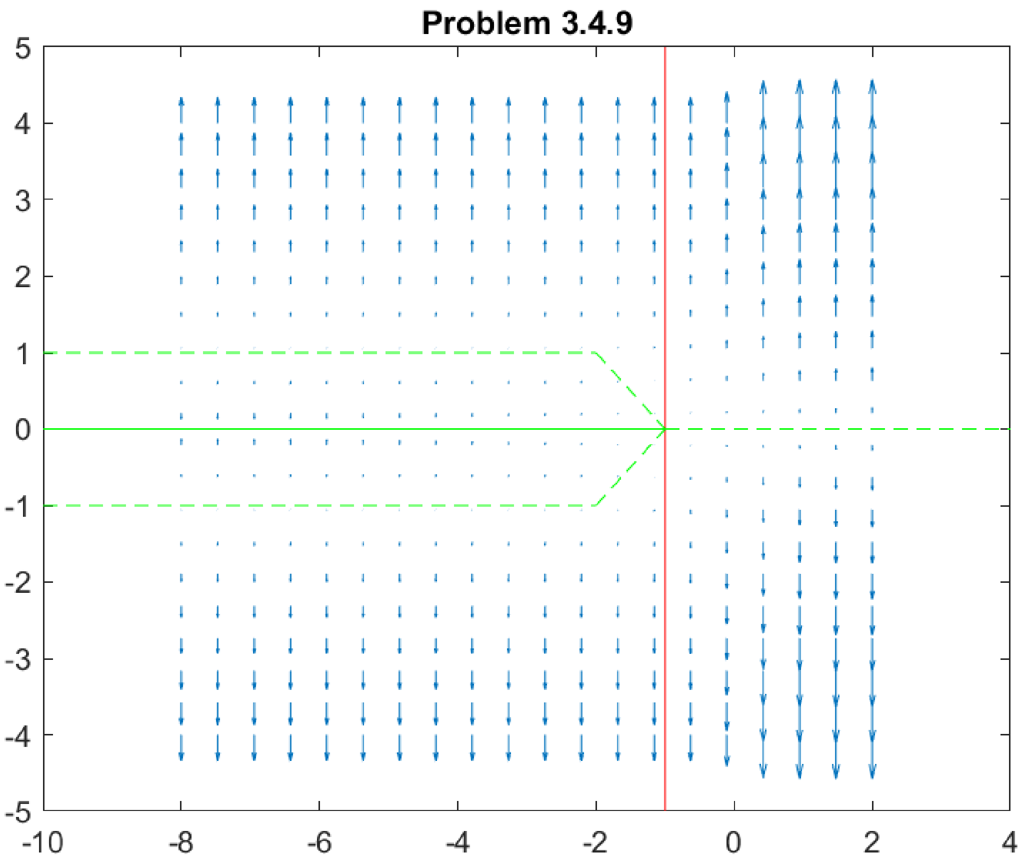

Figure 12: Bifurcation Diagram for problem 3.4.9

A MATLAB Code:

All code I write in this course can be found on my GitHub repository:
https://github.com/jonaswagner2826/MECH6313

Script 1: MECH6313_HW1

```
%% MECH6313 - HW 1
clear
close all

pblm1 = false;
pblm2 = false;
pblm4 = true;

if pblm1
%% Problem 1
% System Def
delta = 0.05;
alpha = 0.4;
omega_t = 1.3;
sys1 = nlsys(@duff_eq);

% Simulation Setup
x_0 = [0;0];

N = 1e4;
t_step = 0.01;
t_max = N * t_step - t_step;
T = reshape(0:t_step:t_max,N,1);
U = alpha * cos(omega_t * T);
SYS1 = nlsim(sys1,U,T,x_0);

% Phase Plot
fig = SYS1.phasePlot(1,2,'Problem 1 - Phase Plot (Relaxed System)');
saveas(fig,fullfile([pwd '\\' 'HW1' '\\' 'fig'],'pblm1_phase.png'))

% Phase Plot Comparrison
N = 5e3;
fig = figure('position',[0,0,1000,1000]);
for i = 1:4
    x_0 = randn(2,1);
    SYS(i) = nlsim(sys1,U,T,x_0);
    ax = subplot(2,2,i);
    SYS(i).phasePlot(1,2,x_0,fig,ax);
```

```
end
sgtitle('Problem 1 - Duff Equation Phase Portrait Comparrison')
saveas(fig,fullfile([pwd '\\' 'HW1' '\\' 'fig'],'pblm1_phase_comparrision.png'))

% Vs Time Plot
fig = SYS1.plot(-1,0,0);
sgtitle('Problem 1 - Duff Equation Time Simulation')
saveas(fig,fullfile([pwd '\\' 'HW1' '\\' 'fig'],'pblm1_vs_time.png'))

end

if pblm2
%% Problem 2
% System Def
sys2 = nlsys(@van_der_pol);

% Simulation Setup
x_0 = [0.8; -0.2];

N = 5e3;
t_step = 0.01;
t_max = N * t_step - t_step;
T = reshape(0:t_step:t_max,N,1);
U = 0 * T;
SYS2 = nlsim(sys2,U,T,x_0);

% Phase Plot
fig = SYS2.phasePlot(1,2,'Problem 2 - Phase Plot');
saveas(fig,fullfile([pwd '\\' 'HW1' '\\' 'fig'],'pblm2_phase.png'))

% Phase Plot Comparrison
fig = figure('position',[0,0,1000,1000]);
X_0 = [ -0.5,0.8, -1.5, 3;
       0.5, -0.5, 2.7, -1.9];
for i = 1:4
    SYS(i) = nlsim(sys2,U,T,X_0(:,i));
    ax = subplot(2,2,i);
    SYS(i).phasePlot(1,2,X_0(:,i),fig,ax);
end
sgtitle('Problem 2 - Van Der Pol Phase Portraits')
saveas(fig,fullfile([pwd '\\' 'HW1' '\\' 'fig'],'pblm2_phase_comparrision.png'))

% Vs Time Plot
```

```
x_0 = X_0(1);
fig = SYS2.plot(-1,0,0);
sgtitle('Problem 2 - Van Der Pol Time Simulation')
saveas(fig,fullfile([pwd '\\' 'HW1' '\\' 'fig'],'pblm2_vs_time.png'))


% Negative Equivelent
sys2_neg= nlsys(@van_der_pol,'empty',0,-1,0,0,-1);

% Negative Stability
x_0 = [0;0];
u_0 = 0;
sys2_neg_lin = sys2_neg.ss(x_0,u_0)
eig_A = eig(sys2_neg_lin.A)
is_stable = isstable(sys2_neg_lin)



%Negative Sim
x_0 = [0.8; -0.2];
SYS2_neg = nlsim(sys2_neg,U,T,x_0)


% Phase Plot Comparrison
fig = figure('position',[0,0,1000,1000]);
%X_0(i) is from last set of plot
for i = 1:4
    SYS_neg(i) = nlsim(sys2_neg,U,T,X_0(:,i));
    ax = subplot(2,2,i);
    SYS_neg(i).phasePlot(1,2,X_0(:,i),fig,ax);
end
sgtitle('Problem 2 - Negative Van Der Pol Phase Portraits')
saveas(fig,fullfile([pwd '\\' 'HW1' '\\' 'fig'],'pblm2_phase_comparrision_neg.png'))

% Vs Time Plot
x_0 = X_0(1);
fig = SYS2_neg.plot(-1,0,0);
sgtitle('Problem 2 - Negative Van Der Pol Time Simulation')
saveas(fig,fullfile([pwd '\\' 'HW1' '\\' 'fig'],'pblm2_vs_time_neg.png'))

end


```

```
if pblm4
%% Problem 4

syms f(x,r) g(x,r)

% Problem 3.4.2
sys342 = nlsys(@pblm342);


r = linspace(-2,5,20);
x = linspace(-4,4,20);
[r_c,fig] = sys342.bifurcationPlot(r,x);


xlimit = xlim;
plot([xlimit(1),r_c],[0,0],'g');
plot([r_c,xlimit(2)],[0,0],'g--');

x = linspace(r_c,xlimit(2));
y = 2 * sqrt(x-r_c);
plot(x,y,'g-')
plot(x,-y,'g-')
hold off

title('Problem 3.4.2')

saveas(fig,fullfile([pwd '\\' 'HW1' '\\' 'fig'],'pblm4_342.png'))


% Problem 3.4.4
sys344 = nlsys(@pblm344);
r = linspace(-5,3,20);
x = linspace(-4,4,20);
[r_c,fig] = sys344.bifurcationPlot(r,x);


xlimit = xlim;
plot([xlimit(1),r_c],[0,0],'g')
plot([r_c,xlimit(2)],[0,0],'g--')

x = linspace(xlimit(1),r_c);
y = 1 * sqrt(abs(x-r_c));
plot(x,y,'g--')
```

```
plot(x,-y,'g--')
hold off

title('Problem 3.4.4')

saveas(fig,fullfile([pwd '\\' 'HW1' '\\' 'fig'],'pblm4_344.png'))

% Problem 3.4.7
sys347 = nlsys(@pblm347);
r = linspace(-5,20,25);
x = linspace(-3,3,10);
[r_c,fig] = sys347.bifurcationPlot(r,x);


xlimit = xlim;
x = linspace(r_c,xlimit(2));
y = 0.35 * sqrt(x-r_c);
plot(x,y,'g--')
plot(x,-y,'g-')
hold off

title('Problem 3.4.7')

saveas(fig,fullfile([pwd '\\' 'HW1' '\\' 'fig'],'pblm4_347.png'))

% Problem 4.4.9
sys349 = nlsys(@pblm349);
r = linspace(-8,2,20);
x = linspace(-4,4,20);
[r_c,fig] = sys349.bifurcationPlot(r,x);


xlimit = xlim;
x = linspace(xlimit(1),-2);
y = 0 * x;
plot(x,y-1,'g--');
plot(x,y+1,'g--');
plot(x,y,'g-');

x = linspace(-2,r_c);
y = 0 * x;
plot(x,y-x-1,'g--');
plot(x,y+x+1,'g--');
```

```
plot(x,y,'g-');

x = linspace(r_c,xlimit(2));
y = 0*x;
plot(x,y,'g--');

hold off

title('Problem 3.4.9')

saveas(fig,fullfile([pwd '\\' 'HW1' '\\' 'fig'],'pblm4_349.png'))

end







%% Local Functions
function y = duff_eq(x,u,parms)
    % DUFF_EQ nonlin function with caotic behavior
    arguments
        x (2,1) = [0; 0];
        u (1,1) = 0;
        parms = false
    end

    if parms == false
        delta = 0.05;
    else
        delta = parms(1);
    end

    % Array sizes
    n = 2; % Number of states
    p = 1; % Number of inputs

    % State Update Equations
    y(1,1) = x(2);
    y(2,1) = - delta * x(2) + x(1) - x(1)^3 + u;

```

```
    if nargin ==0
        y = [n;p];
    end
end

function y = van_der_pol(x,u,parms)
    % VAD_DER_POL nonlin function
    arguments
        x (2,1) = [0; 0];
        u (1,1) = 0;
        parms = false
    end

    if parms == false
        a = 1;
    else
        a = parms(1);
    end

    % Array sizes
    n = 2; % Number of states
    p = 1; % Number of inputs

    % State Update Equations
    y(1,1) = x(2);
    y(2,1) = - a * (x(1)^2 -1) * x(2) - x(1) + u;


    if nargin ==0
        y = [n;p];
    end
end


function y = pblm342(x,u,parms)
    % VAD_DER_POL nonlin function
    arguments
        x (1,1) = 0;
        u (1,1) = 0;
        parms = false
    end

```

```
    if parms == false
        r = 1;
    else
        r = parms(1);
    end

    % Array sizes
    n = 1; % Number of states
    p = 1; % Number of inputs

    % State Update Equations
    y(1,1) = r * x(1) - sinh(x(1));


    if nargin ==0
        y = [n;p];
    end
end


function y = pblm344(x,u,parms)
    % VAD_DER_POL nonlin function
    arguments
        x (1,1) = 0;
        u (1,1) = 0;
        parms = false
    end

    if parms == false
        r = 1;
    else
        r = parms(1);
    end

    % Array sizes
    n = 1; % Number of states
    p = 1; % Number of inputs

    % State Update Equations
    y(1,1) = x(1) + (r*x(1))/(1+x(1)^2);


    if nargin ==0
```

```
        y = [n;p];
    end
end


function y = pblm347(x,u,parms)
    % VAD_DER_POL nonlin function
    arguments
        x (1,1) = 0;
        u (1,1) = 0;
        parms = false
    end

    if parms == false
        r = 1;
    else
        r = parms(1);
    end

    % Array sizes
    n = 1; % Number of states
    p = 1; % Number of inputs

    % State Update Equations
    y(1,1) = 5 - r * exp(-x(1)^2);


    if nargin ==0
        y = [n;p];
    end
end

function y = pblm349(x,u,parms)
    % VAD_DER_POL nonlin function
    arguments
        x (1,1) = 0;
        u (1,1) = 0;
        parms = false
    end

    if parms == false
        r = 1;
    else
```

```
        r = parms(1);
    end

    % Array sizes
    n = 1; % Number of states
    p = 1; % Number of inputs

    % State Update Equations
    y(1,1) = x + tanh(r*x(1));


    if nargin ==0
        y = [n;p];
    end
end
```