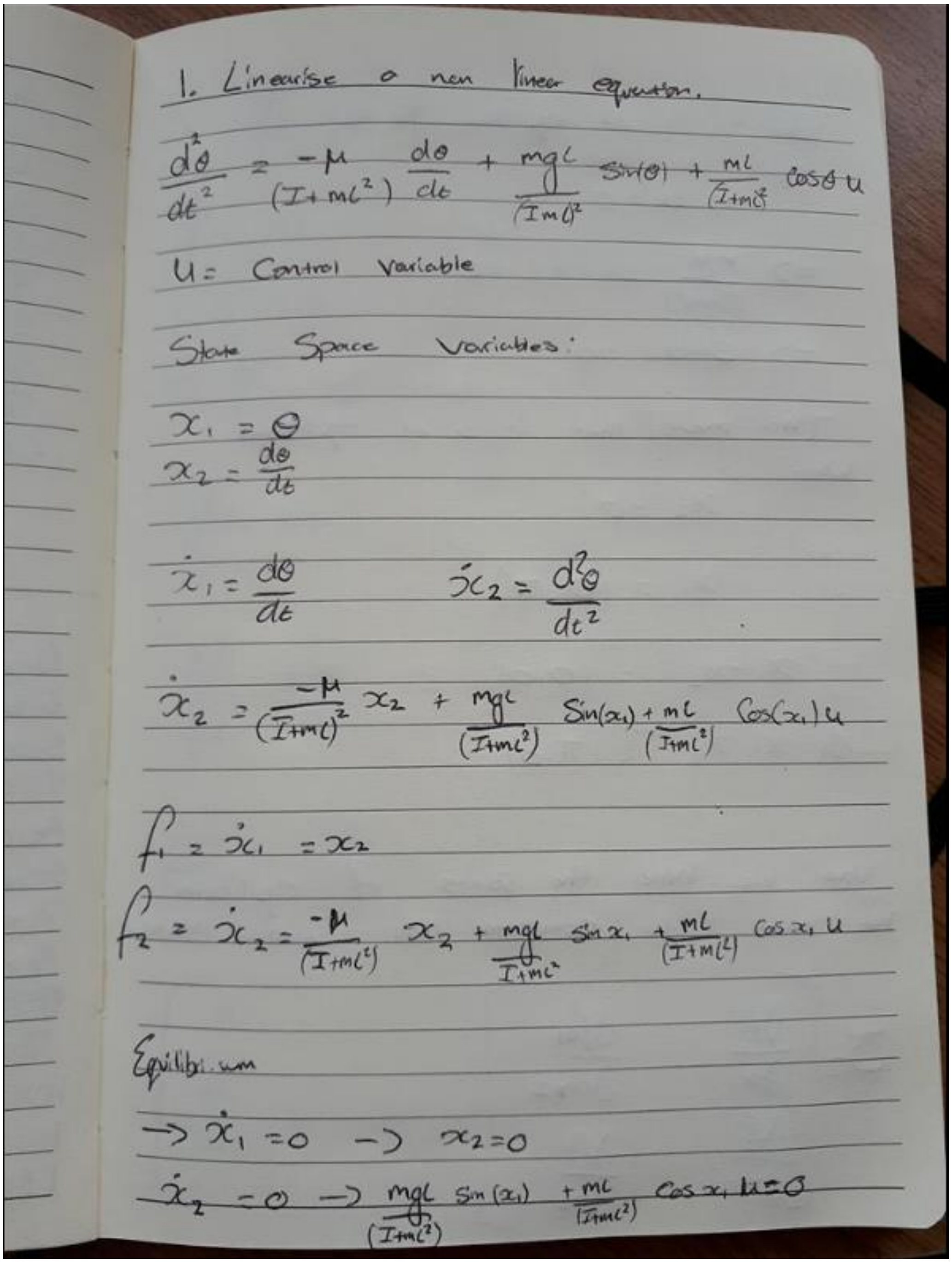
GITHUB LINK TO MATLAB FILES: https://github.com/bwickenden/ROCO223.git

1. Linearize the non-linear differential equation



1. Linearise a non linear equation.

$$\frac{d^2\theta}{dt^2} = \frac{-\mu}{(I+mL^2)}\frac{d\theta}{dt} + \frac{mgL}{(I+mL)^2}\sin(\theta) + \frac{mL}{(I+mL^2)}\cos\theta\, u$$

u = Control variable

State Space Variables:

$x_1 = \theta$

$x_2 = \frac{d\theta}{dt}$

$\dot{x}_1 = \frac{d\theta}{dt}$ $\qquad$ $\dot{x}_2 = \frac{d^2\theta}{dt^2}$

$$\dot{x}_2 = \frac{-\mu}{(I+mL)^2}x_2 + \frac{mgL}{(I+mL^2)}\sin(x_1) + \frac{mL}{(I+mL^2)}\cos(x_1)u$$

$f_1 = \dot{x}_1 = x_2$

$$f_2 = \dot{x}_2 = \frac{-\mu}{(I+mL^2)}x_2 + \frac{mgL}{I+mL^2}\sin x_1 + \frac{mL}{(I+mL^2)}\cos x_1\, u$$

Equilibrium

$\rightarrow \dot{x}_1 = 0 \rightarrow x_2 = 0$

$$\dot{x}_2 = 0 \rightarrow \frac{mgL}{(I+mL^2)}\sin(x_1) + \frac{mL}{(I+mL^2)}\cos x_1\, u = 0$$

When it is at equilibrium the input is equal to zero

$\rightarrow u = 0$

$\Rightarrow \frac{mgl}{(I+ml^2)} \sin(x_1) = 0$

$\Rightarrow \sin(x_1) = 0$

This means that it is at equilibrium when

$x_2 = 0$

and

$x_1 = (0, \pi)$

$x_1\, x_2 = (0, 0)$

Or $x_1\, x_2 = (\pi, 0)$

Now we have the points of equilibrium we need to complete the Jacobian Matrix

$$J = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{bmatrix}$$

So $\dfrac{\partial f_1}{\partial x_1} =$

$f_1 = x_2$

$f_1 = 0.x_2 + 1.x_2$

$\Rightarrow \dfrac{\partial f_1}{\partial x_1} = 0$

$$f_2 = \frac{-\mu}{(I+ml^2)} x_2 + \frac{mgl}{(I+ml^2)} \sin(x_1) + \frac{ml}{(I+ml^2)} \cos(x_1) u$$

$\dfrac{\partial f_1}{\partial x_2} = 1$

$$\frac{\partial f_2}{\partial x_1} = \frac{mgl}{(I+ml^2)} \cos(x_1) + \frac{ml}{(I+ml^2)} - \sin(x_1) u$$

$$\frac{\partial f_2}{\partial x_2} = \frac{-\mu}{(1+ml^2)}$$

$$\begin{bmatrix} 0 & 1 \\ \dfrac{mgl}{(I+ml^2)} \cos(x_1) + \dfrac{ml}{(I+ml^2)} - \sin(x_1 u) & \dfrac{-\mu}{(I+ml^2)} \end{bmatrix}$$

B

$$\begin{matrix}\frac{\partial f_1}{\partial u}\\ \frac{\partial f_2}{\partial u}\end{matrix} = \begin{bmatrix} 0 \\ \frac{ml}{I+ml^2}\ \mathrm{Sin}(x_1) \end{bmatrix}$$

$$\frac{\partial f_1}{\partial u} = \frac{x_2}{u} \rightarrow u = \frac{d^2x}{dt^2} = 0$$

$$\begin{bmatrix} 0 & 1 \\ \frac{mgl}{(I+ml^2)} & \frac{-\mu}{(I+ml^2)} \end{bmatrix} x_1 \begin{bmatrix} 0 & 1 \\ \frac{-mgl}{(I+ml^2)} & \frac{-\mu}{I+ml^2} \end{bmatrix} x_2 \begin{bmatrix} 0 \\ \frac{ml}{(I+ml^2)}\ \mathrm{Sin}\,x_1 \end{bmatrix}$$

Expand.

$$(I+ml^2)\frac{d^2\theta}{dt} + \mu\frac{d\theta}{dt} = mgl\theta + ml\frac{d^2x}{dt^2}$$

2. Write down the state space model of the system

For this task I worked through the statespace model both by hand and by using MatLab.

By hand:
Firstly I assigned my state variables
X1 and X2, along with X1 dot and X2 dot
I then rearranged the equation so the it was equal to X2 dot.
Next I let b0 = $\frac{ml}{(I+ml^2)}$ **and then continued to simplify**
Then I substituted X1 and X2 into each equation and then followd up by putting the equation in statespace form

MatLab code :
I started off by declaring the parameters for all of the components necessary for the statespace model, including the calculation of I = $\frac{1}{12} * m * l^2$
Next I added a list of coefficents, followed by the state spaces.
I then found the eigen vales for both A and A inverted.

Editor - C:\Users\Ben\Desktop\223 Referred Coursework\223 Referred Coursework\State_space.m

State_space.m

```
%parameters
u = 0.05;  %Coefficent of viciously
m = 0.314; %Mass
l = 0.64; %length the the centre of mass
g = 9.81; % Acceleration due to gravity
I = ((1/12)*m*(l^2)); %Moment of Inertial of the pendulum rod at centre

%Coefficients
a0 = 1;
a1 = (m*u)/(I+m*(l/2)^2);
a2 = (-m*g*(l/2))/(I+m*(l/2)^2);
b0 = (m*(l/2))/(I+m*(l/2)^2);
b1 = 0;

%State spaces
A = [0 1; -a2 -a1; ];
B = [b0; (b1-a1*b0);];
C = [1 0;];
D = 0;



A_inverted = [0 1; -a2 -a1; ];

A_noninverted = [0 1; a2 -a1; ];
```

Command Window

```
>> A_noninverted

A_noninverted =

         0    1.0000
  -22.9922   -0.3662

>> A_inverted

A_inverted =

         0    1.0000
   22.9922   -0.3662

fx >>
```

Roco 223 – State Space Model 4/5/18

$$(I+ml^2)\frac{d^2\theta}{dt^2} + \mu\frac{d\theta}{dt} = mgl\theta + ml\frac{d^2x}{dt^2}$$

$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$

$$\frac{d^2\theta}{dt^2} = \frac{-\mu}{(I+ml^2)}\frac{d\theta}{dt} + \frac{mgl\theta}{(I+ml^2)} + \frac{ml}{(I+ml^2)}\left(\frac{d^2x}{dt^2}\right)$$

$$x_1 = \theta \qquad \dot{x}_1 = \frac{d\theta}{dt}$$

$$x_2 = \left(\frac{d\theta}{dt} - b_0 V_c\right)$$

$$\frac{d\theta}{dt} = (x_2 + b_0 V_c)$$

$$\dot{x}_2 = \frac{d^2\theta}{dt^2} - b_0\frac{dv_c}{dt}$$

$$\Rightarrow \frac{d^2\theta}{dt^2} - b_0\frac{dv_c}{dt} = \frac{-\mu}{(I+ml^2)}\frac{d\theta}{dt} + \frac{mgl\theta}{(I+ml^2)} \cancel{+ \frac{ml}{(I+ml^2)}\frac{dv}{dt} - b_0\frac{dv_c}{dt}}$$

Let $b_0 = \frac{ml}{(I+ml^2)}$

$$\Rightarrow \left(\frac{d^2\theta}{dt^2} - \frac{ml}{(I+ml^2)} \frac{dv}{dt} \right)$$

$$\dot{x}_2 = \frac{-\mu}{(I+ml^2)} \frac{d\theta}{dt} + \frac{mgl}{(I+ml^2)} \cdot \theta$$

$$\dot{x}_2 = \frac{-\mu}{I+ml^2} \frac{d\theta}{dt} + \frac{mgl}{(I+ml^2)} \cdot x_1$$

$$\dot{x}_2 = \frac{-\mu}{(I+ml^2)} \left[x_2 + \left[\frac{ml}{(I+ml^2)} \right] V_c \right] + \frac{mgl}{I+ml^2} x_1$$

$$\dot{x}_2 = \frac{-\mu}{(I+ml^2)} x_2 + \frac{mgl}{I+ml^2} x_1 \frac{-\mu}{(I+ml^2)} \frac{ml}{(I+ml^2)} V_c$$

$$\dot{x}_2 = \frac{mgl}{I+ml^2} x_1 - \frac{\mu}{(I+ml^2)} x_2 - \frac{\mu ml}{(I+ml^2)} V_c$$

$$\dot{x}_1 = \frac{d\theta}{dt} = x_2 + \frac{ml}{(I+ml^2)} V_c$$

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \frac{mgl}{(I+ml^2)} & \frac{-\mu}{I+ml^2} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} \frac{ml}{(I+ml^2)} \\ \frac{-\mu ml}{(I+ml^2)} \end{bmatrix} V_c$$

3. Observability, controllability and stability

Once again we did completed this task both by hand and by using MatLab

Controllability

From here we can see that controllability matrix is rank 2 which is full rank which means that it is indeed controllable.

```
%Controllability
disp('controllability matrix for 2x2 A system matrix system');
AB = A * B;
MXc=[B AB ]
rank(MXc)
```

Command Window

```
controllability matrix for 2x2 A system matrix system

MXc =

    2.3438   -0.8583
   -0.8583   54.2023


ans =

     2
```

Below is us calculating this by hand and coming to the same conclusion.

Task 3 – Controllability (1)

$$A = \begin{bmatrix} 0 & 1 \\ 23.544 & -0.027 \end{bmatrix}$$

$$B = \begin{bmatrix} 2.4 \\ -0.0648 \end{bmatrix}$$

Controlability

$$M_c = \begin{bmatrix} B & AB \end{bmatrix} = \begin{bmatrix} 2.4 \\ -0.0648 \end{bmatrix} \times \begin{bmatrix} 0 & 1 \\ 23.544 & -0.027 \end{bmatrix} \begin{bmatrix} 2.4 \\ -0.027 \end{bmatrix}$$

$$AB = \begin{bmatrix} (2.4 \times 0) + (1 \times -0.0648) \\ (23.544 \times 2.4) + (\overset{-0.027}{\cancel{29.827}} \times -0.0648) \end{bmatrix}$$

$$AB = \begin{bmatrix} -0.0648 \\ 56.5073 \end{bmatrix}$$

~~B=~~

$$M_c = \begin{bmatrix} B & AB \end{bmatrix}$$

$$M_c = \begin{bmatrix} 2.4 & -0.0648 \\ -0.0648 & 56.5073 \end{bmatrix}$$

This shows that it is rank 2 which is full rank. Therefore it is Controllable

Observability

Once again we can see that the observability matrix is rank two which is full rank, which indicates that it is observable. This is also see in our hand notes.

```
%Observability, controllability and stability

disp('observability matrix for 2x2 A system matrix system');
CA = C*A;
MXo=[C; CA;]
rank(MXo)
```

Command Window

```
>> State_space
observability matrix for 2x2 A system matrix system

MXo =

     1     0
     0     1


ans =

     2
```

Task 3 Observability

$$A = \begin{bmatrix} 0 & 1 \\ 23.544 & -0.0276 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$M_o = \begin{bmatrix} C & CA \end{bmatrix}$$

$$CA = \begin{bmatrix} (0 \times 1) + (1 \times 0) \\ (23.544 \times 1) + (0 \times -0.027) \end{bmatrix} = \begin{bmatrix} 0 \\ 23.544 \end{bmatrix}$$

$$M_o = \begin{bmatrix} 1 & 0 \\ 0 & 23.544 \end{bmatrix}$$

Due to this being rank 2 this means that this is Observable as its full rank

Inverted A

$$A = \begin{bmatrix} 0 & 1 \\ -23.544 & -0.027 \end{bmatrix} \quad I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$|A - \lambda I| = \left| \begin{bmatrix} 0 & 1 \\ -23.544 & -0.027 \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right|$$

$$= \begin{bmatrix} -\lambda & 1 \\ -23.544 & -0.027-\lambda \end{bmatrix}$$

so $(-\lambda) \times (-0.027 - \lambda) - (-23.544)$

$= (-\lambda^2 - 0.027\lambda) + 23.544 = 0$

$\lambda^2 + 0.027\lambda - 23.544 = 0$

$\Rightarrow$ quadratic formulea $\Rightarrow$ shows $\lambda = 4.8387$

$\lambda = -4.8657$

one positive & one negetive eigenvalue
therefer not stable

Stability

From looking at A_inverted Eigen values we can see that it is not stable due to positive values.

```
%Inverted config
A_inverted = [0  1; -a2 -a1; ];

% eigen values of non-inverted config
eig(A_inverted)
```

```
Command Window
ans =

    4.6154
   -4.9816
```

From looking at A_nonInverted Eigen values we can see that they are't far off from the Imaginary axis and thus would be stable.

```
%Get non-inverted config
A_nonInverted = [0 1; a2 -a1; ];

%Eigen values of non-inverted config
eig(A_nonInverted)
```

```
ans =

  -0.1831 + 4.7915i
  -0.1831 - 4.7915i

fx >>
```

Task 3 – Stability

$$A = \begin{bmatrix} 0 & 1 \\ 23.544 & -0.027 \end{bmatrix}$$

$IA = A$

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \det A = \begin{bmatrix} -23.544 \end{bmatrix}$$

$$\Rightarrow \left| \begin{bmatrix} 0 & 1 \\ 23.544 & -0.027 \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right| = 0$$

$$= \begin{bmatrix} -\lambda & 1 \\ 23.544 & -0.027-\lambda \end{bmatrix} = 0$$

$= (-\lambda)\times(-0.027-\lambda) - 23.544 = 0$

$\Rightarrow 0.027\lambda - \lambda^2 - 23.544 = 0$

$-\lambda^2 - 23.571 = 0$

$\lambda^2 = -23.571$ λ = imaginary

so the system would be stable & oscillate

$= -\lambda^2 - 0.027\lambda - 23.544 = 0$

quadratic formulea $\Rightarrow$ shows 2 negtive maginary ~~no~~ numbers

~~∴ stable and is damped~~

$\lambda = -0.0135 + 4.85219i$

$\lambda = -0.0135\ \cancel{+} - 4.85219i$

4. Simulate your state space module using the Matlab ode45 function

Youtube videos Animation 1: https://www.youtube.com/watch?v=5aATIaNZaQI
Animation 2: https://www.youtube.com/watch?v=tIKLcAGnLvE

$$\dot{X} = AX + BU$$
$$Y = CX + DU$$

For this task I created a function to be used with the ode 45 function as shown below.

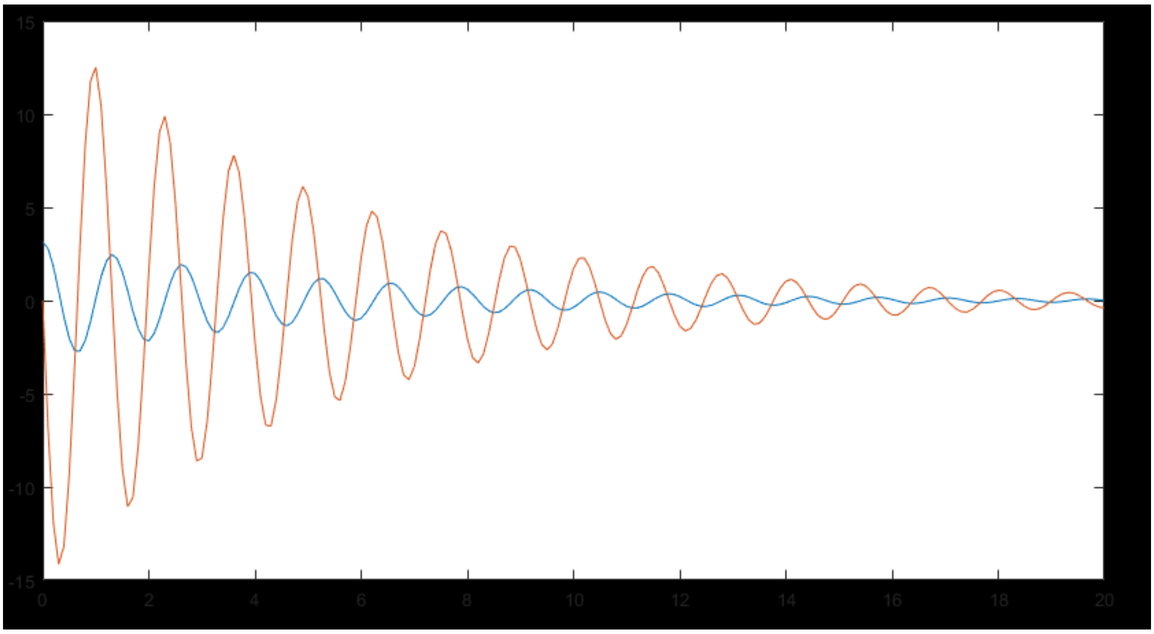
Simulate.m | Main_simFCOde45.m | FCPendOnCart.m | AnimatePendulumCart.m

```
function xDot = Simulate(A, x, B, u)

xDot = A*x + B*u;

end
```

This is the function that I used in my animation

```
% representing a force controlled pendulum on a cart
% model introduces slight amount of noise to wont stay balanced
[t,y] = ode45(@(t,y)Simulate(A, y, B, 0),tspan,y0);
```

Below is the graph from the 2nd animation, I decided to run another animation due to the first one didn't run long enough



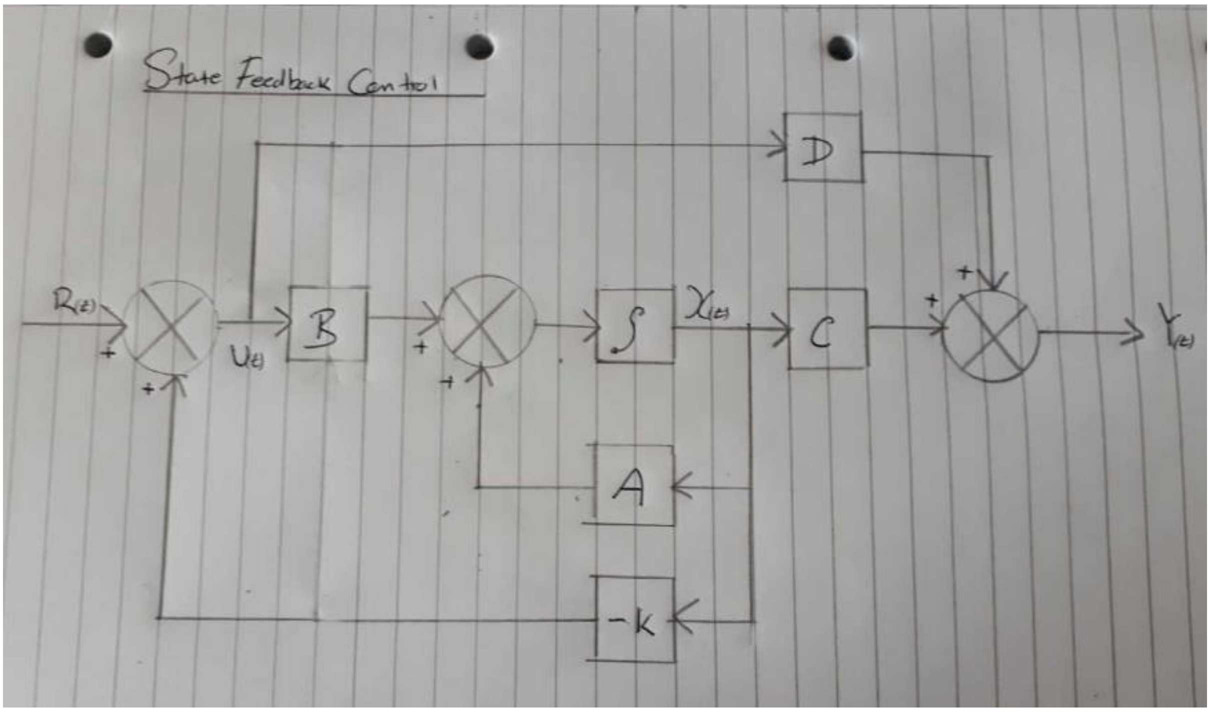
5. Design a state feedback controller

YouTube video Ode with Feedback:
https://www.youtube.com/watch?v=_GHqLaz_QUA

$$\dot{X} = AX + BU$$
$$Y = CX + DU$$

To use feedback in my system I have used state feedback where $U = -KX$
Where the matrix K is the feedback gain of the system. So the system can now be written as $\dot{X} = (A - BK)X$
We can now implement this to find the eigen values $|\lambda I - (A - BK)| = 0$
This shows us that we can influence the location of the eigen values by changing K

```
Command Window

Feedback =

    6.6896   -0.8797
```



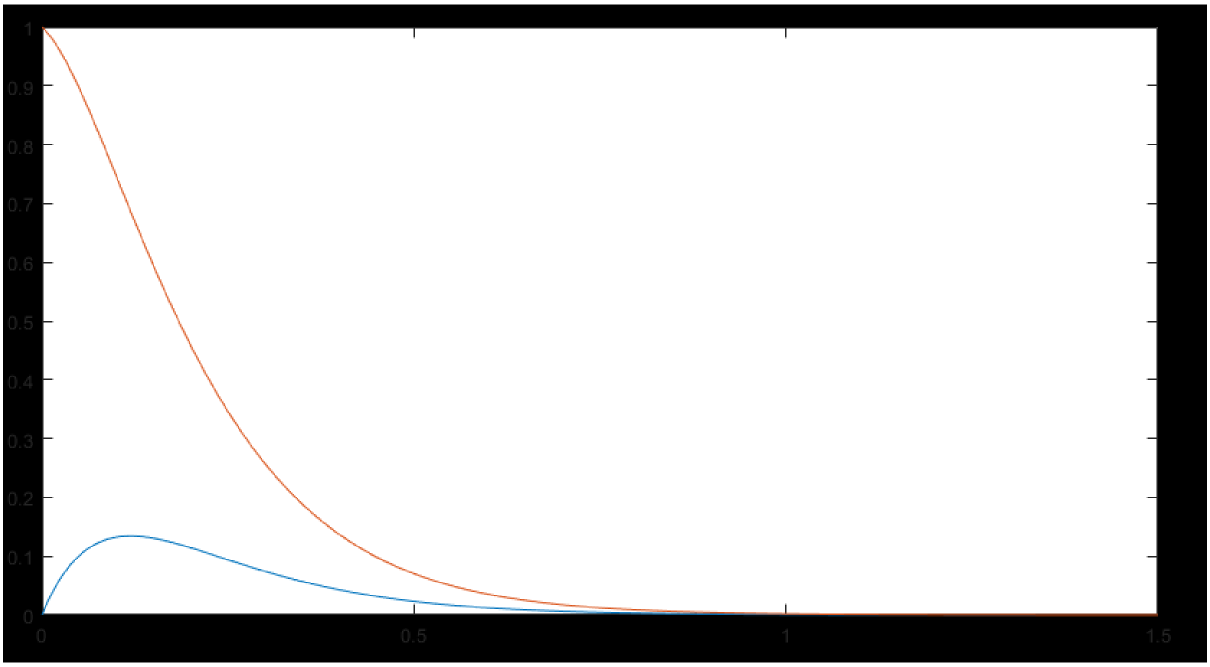
For this task a wrote a separate function from task 4

Feedback_function.m | Main_simFCOde45.m | AnimatePendulumCart.m

```
function Feedback = Feedback_function(A, x, B, u)


Feedback = A*x + B*u;
end
```

I then wrote a feedback controller and added it to the Main_simFCOde45.m along with my statespace model

```
u = 0.05;  %Coefficent of viciously
m = 0.314;% pendulum point mass
M = 2;% cart mass
L = 1; % pendulum length
l=0.64;
g = -9.81;% acceleration due to gravity
d = 1;% damping
I = ((l/12)*m*(l^2)); %Moment of Inertial of the pendulum rod at centre of mass

%Coefficients
a0 = 1;
a1 = (m*u)/(I+m*(l/2)^2);
a2 = (-m*g*(l/2))/(I+m*(l/2)^2);
b0 = (m*(l/2))/(I+m*(l/2)^2);
b1 = 0;

%State spaces
A = [0 1; -a2 -a1; ];
B = [b0; (b1-a1*b0);];
C = [1 0;];
D = 0;

% Inverted config
A_inverted = [0 1; -a2 -a1; ];

%Feedback
PX =8 *[-1 -1.1];
Feedback = place(A,B,PX);
```


1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.5
1
1.5

Task 5 – State feedback Control

$$|\lambda I - (A - BK)| = 0$$

$$A = \begin{bmatrix} 0 & 1 \\ 23.544 & -0.027 \end{bmatrix}$$

$$B = \begin{bmatrix} 2.4 \\ \cancel{-0.0648} \\ \cancel{-0.648} \end{bmatrix}$$

$$\lambda = \begin{bmatrix} 0.0135 + 4.8522i \\ -0.0135 - 4.8522i \end{bmatrix}$$

$$\dot{x} = (A - Bk)x \quad \text{and} \quad Y = (C - DK)x$$

$$\lambda I = \begin{bmatrix} 0.0135 + 4.8522i \\ -0.0135 - 4.8522i \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0.0135 + 4.8522i \\ 0 \end{bmatrix}$$

$$0.0135 + 4.8522i - (A - BK) = 0$$

$$0.0135 + 4.8522i = A - Bk$$

$$BK = \begin{bmatrix} 2.4k \\ -0.0648k \end{bmatrix}$$

$$\det \left| \lambda I - (A - BK) \right| = 0$$

$$\begin{bmatrix} 0.0135 + 4.8522i \\ 0 \end{bmatrix} - \left(\begin{bmatrix} 0 & 1 \\ 23.544 & -0.027 \end{bmatrix} - \begin{bmatrix} 2.4k \\ -0.0648k \end{bmatrix} \right) = 0$$

$$\begin{bmatrix} 0.0135 + 4.8522i \\ 0 \end{bmatrix} - \begin{bmatrix} \cancel{0} - 2.4k & -1.4k \\ 23.544 + 0.0648k & -0.027 + 0.0648k \end{bmatrix} = 0$$

$$\Rightarrow \begin{bmatrix} 0.0135 + 4.8522i + 2.4k & 0.0135 + 4.8522i + 1.4k \\ -23.544 - 0.0648k & +0.027 - 0.0648k \end{bmatrix} = 0$$

$$Bk = \begin{bmatrix} 2.4k \\ -0.0648k \end{bmatrix}$$

$$\det \left| \lambda I - (A - Bk) \right| = 0$$

$$\begin{bmatrix} 0.0135 + 4.8522i \\ 0 \end{bmatrix} - \left(\begin{bmatrix} 0 & 1 \\ 23.544 & -0.027 \end{bmatrix} - \begin{bmatrix} 2.4k \\ -0.0648k \end{bmatrix} \right) = 0$$

$$\begin{bmatrix} 0.0135 + 4.8522i \\ 0 \end{bmatrix} - \begin{bmatrix} \cancel{0} - 2.4k & -1.4k \\ 23.544 + 0.0648k & -0.027 + 0.0648k \end{bmatrix} = 0$$

$$\Rightarrow \begin{bmatrix} 0.0135 + 4.8522i + 2.4k & 0.0135 + 4.8522i + 1.4k \\ -23.544 \mp 0.0648k & +0.027 - 0.0648k \end{bmatrix} = 0$$

6. Implement the controller/observer system using Euler integration

For this task I created a new function to be used instead of the Ode45 called "SimulateSFC"

```
function [y, t, xout] = SimulateSFC(A, B, C, D, K, t, x0)


%get signal length
len = length(t);

%init output
y = zeros(1,len);
xout = zeros(2,len);

%record the initial state
xout(:, 1) = x0;
x = x0;

%calculate the command
u(1)= C(1) * x(1) + C(2) * x(2);

%calculate output from theta and thetaDot states
y(1)= C(1) * x(1) + C(2) * x(2) + D(1)* u(1);

%for all remaining data points, simulate state space model using C

for idx = 2:len

    %state feedback rule
    u(idx) = -K(1) *x(1) -K(2) * x(2);

    %get the duration between updates
    h = t(idx) - t(idx-1);

    %calculate state derivative
    xdot(1) = A(1,1) * x(1) + A(1,2) * x(2) + B(1)* u(idx);
    xdot(2) = A(2,1) * x(1) + A(2,2) * x(2) + B(2) * u(idx);

    %update the state
    x(1) = x(1) + h *xdot(1);
    x(2) = x(2) + h *xdot(2);

    %record the state
    xout(:, idx) = x;

    %calculate output from theta and thetaDot staets only
    y(idx)= C(1) * x(1) + C(2) * x(2) + D(1) * u(idx);
end
```

I then implemented the function onto my code, which replaced the Ode45 function.

```

% use ode to solve with FCPendOnCart with no control force input u
% representing a force controlled pendulum on a cart
% model introduces slight amount of noise to wont stay balanced
[y,t, xout] = SimulateSFC(A, B, C, D, Feedback, tspan, y0);

```

Unfortunately when I ran the code it came up with an error, which I was unable to solve.

```
Command Window

Feedback =

    6.6896   -0.8797

controlled nonlinear sim of FC pendulum on cart

kickLocation =

    0.5600

Index exceeds matrix dimensions.

Error in AnimatePendulumCart (line 56)
    theta = thetaIn(k);

Error in Main_simFCOde45 (line 76)
AnimatePendulumCart( pi +th,  x, L/2, tspan, range, kickFlag,
'With feedback' );

fx >>
```

6. Impliment the Controller System using Euler integration.

$\dot{x} = Ax + BU$

$y = Cx + DU$

$\Rightarrow U = -Kx$

Recurrence relationship

$\dot{x} = Ax + B(-kx)$

$y = Cx + D(-kx)$

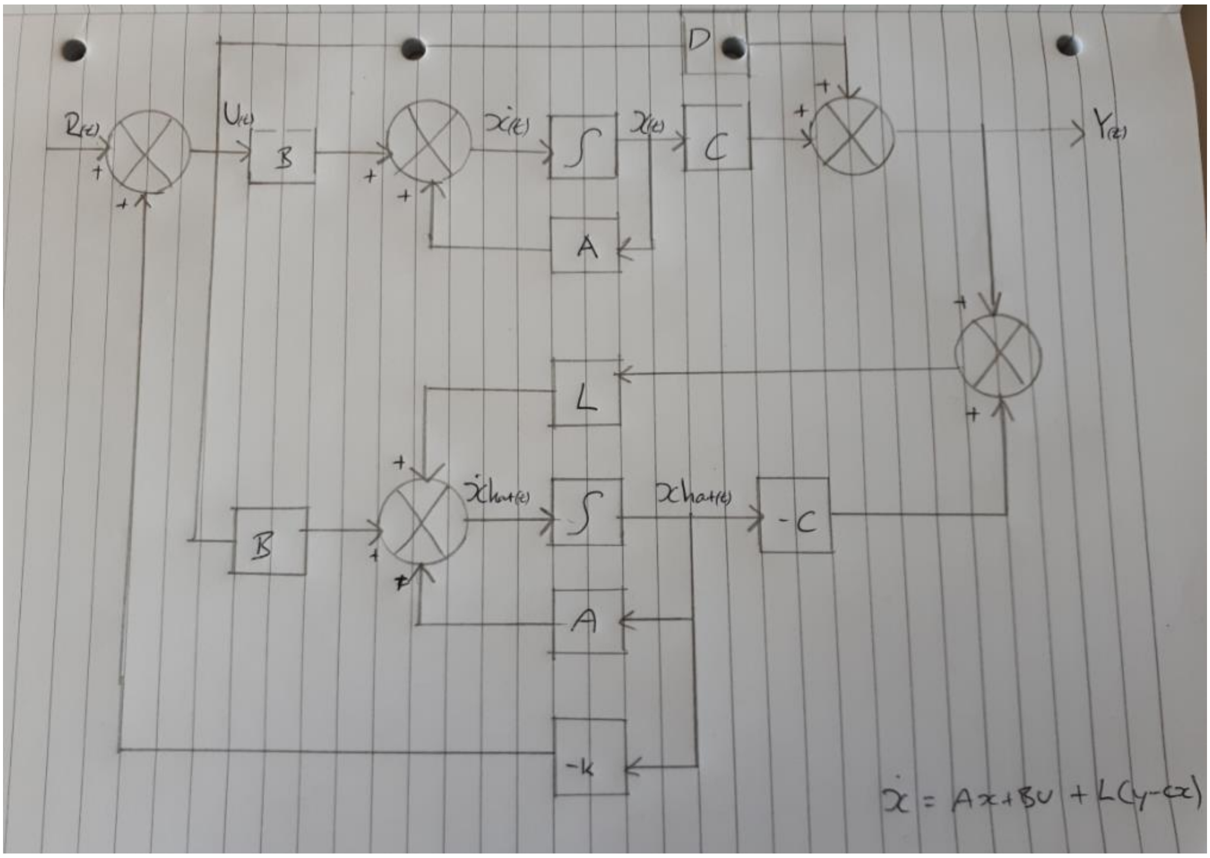
7. Add a Luenberger observer to your state feedback controller

Luenberger gain

$$\dot{X} = AX + BU + L(y - CX)$$

Set up params with Observer gain

```

% clean up matlab before launching script
clear all
close all
clc

u = 0.05;  %Coefficent of viciously
m = 0.314;% pendulum point mass
M = 2;% cart mass
L = 1; % pendulum length
l =0.64;
g = -9.81;% acceleration due to gravity
d = 1;% damping
I = ((1/12)*m*(l^2)); %Moment of Inertial of the pendulum rod at centre of mass

%Coefficients
a0 = 1;
a1 = (m*u)/(I+m*(l/2)^2);
a2 = (-m*g*(l/2))/(I+m*(l/2)^2);
b0 = (m*(l/2))/(I+m*(l/2)^2);
b1 = 0;

%State spaces
A = [0 1; -a2 -a1; ];
B = [b0; (b1-a1*b0);];
C = [1 0;];
D = 0;

% Inverted config
A_inverted = [0 1; -a2 -a1; ];

%Feedback
PX =8 *[-1 -1.1];
Feedback = place(A,B,PX);
Feedback

%observer gain
PX =20 * [-1 -1.2]
L = place(A, C, PX);
LT=L;
```


R(t)
U(t)
B
ẋ(t)
x(t)
C
D
Y(t)
A
L
B
ẋhat(t)
xhat(t)
-C
A
-k
ẋ = Ax+BU + L(y-cx)

8. Augment positional state into your state space model

8. Augment Positional State into your State Space Model

x_3 = Input Velocity Signal

$$\dot{x}_3 = \frac{d\theta}{dt}$$

$$a_1 = \frac{\mu}{(I+ml^2)} \qquad a_2 = \frac{-mgl}{(I+ml^2)} \qquad b_0 = \frac{ml}{(I+ml^2)}$$

Velocity Controlled State Space Pendulum

$$\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -a_2 & -a_1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} b_0 \\ -a_1 b_0 \end{bmatrix} V_C$$

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$ // y = pendulum angle (θ)

With the differential of x_3 being zero x_3 is just the velocity out input

$$\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ -a_2 & -a_1 & 0 \\ 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} b_0 \\ -a_1 b_0 \\ 1 \end{bmatrix} V_C$$

With y being

$$y = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

9. Implement the augmented state feedback controller

10. Implement the augmented state feedback controller on the Arduino Mega