ROCO218: Control Engineering
Dr Ian Howard

Lecture 6

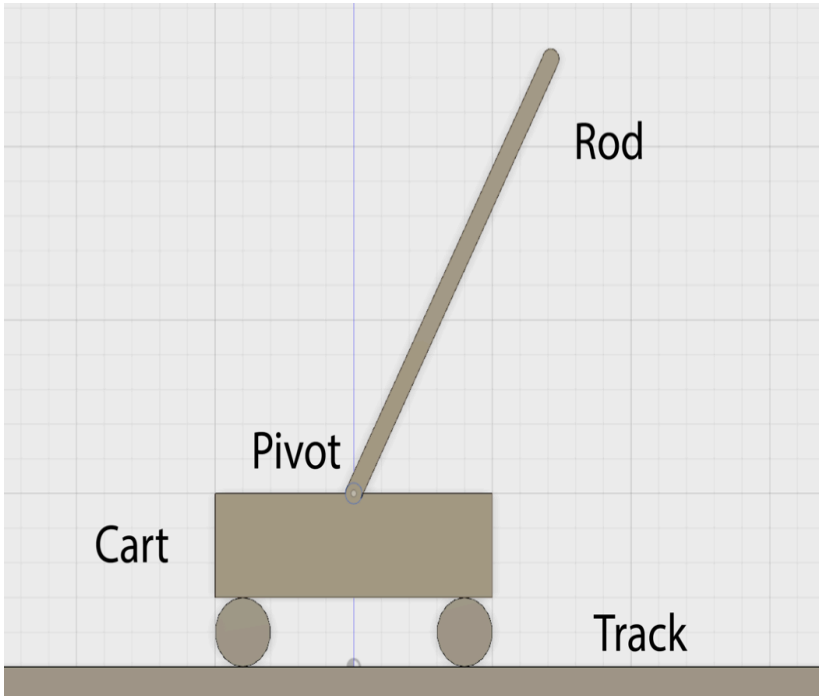
Inverted pendulum acceleration control

Inverted pendulum state space acceleration control

The linearized differential equation describing the inverted pendulum is given by

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

Where:
The angle to the vertical is denoted by θ
The coefficient of viciously is denoted by μ
The mass of the pendulum is denoted by m
The moment of inertial of the rod about the center of mass is denoted by I
The length to the centre of mass is denoted by *l*
The displacement of the pivot is given by x_p



Acceleration control of inverted pendulum

Re-writing the differential equation describing the inverted pendulum

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

To control the pendulum by cart acceleration we can write

$$\frac{d^2x_p}{dt^2} = a_c$$

This leads to the equation

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

Acceleration control of inverted pendulum

Given the equation

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

Let the constant terms be represented by

$$a_1 = \frac{\mu}{\left(I + ml^2\right)} \qquad b_0 = \frac{ml}{\left(I + ml^2\right)}$$

$$a_2 = \frac{-mgl}{\left(I + ml^2\right)}$$

Which lets us write the equation in the previous canonical form

$$\Rightarrow \frac{d^2\theta}{dt^2} = -a_1\frac{d\theta}{dt} - a_2\theta + b_0 a_c$$

Choosing state space representations

$$x_1 = \theta$$

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

Acceleration control of inverted pendulum

From the state space representations

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

$$x_2 = \frac{d\theta}{dt} \qquad \Rightarrow \dot{x}_1 = x_2$$

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

From before

$$\frac{d^2\theta}{dt^2} = -a_1 \frac{d\theta}{dt} - a_2\theta + b_0 a_c$$

$$\Rightarrow \frac{d^2\theta}{dt^2} = -a_1 x_2 - a_2 x_1 + b_0 a_c$$

Substituting into

$$\Rightarrow \dot{x}_2 = -a_1 x_2 - a_2 x_1 + b_0 a_c$$

Acceleration control of inverted pendulum

This we have the state space representations

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

$$\Rightarrow \dot{x}_2 = -a_1 x_2 - a_2 x_1 + b_0 a_c$$

State space notation takes the form

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

Writing in matrix format we therefore have

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

Where output y is the pendulum angle θ

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

$$\Rightarrow \frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \dfrac{mgl}{\left(I + ml^2\right)} & -\dfrac{\mu}{\left(I + ml^2\right)} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \dfrac{ml}{\left(I + ml^2\right)} \end{bmatrix} a_c$$

Revision: State space feedback control

Remember state space equations are

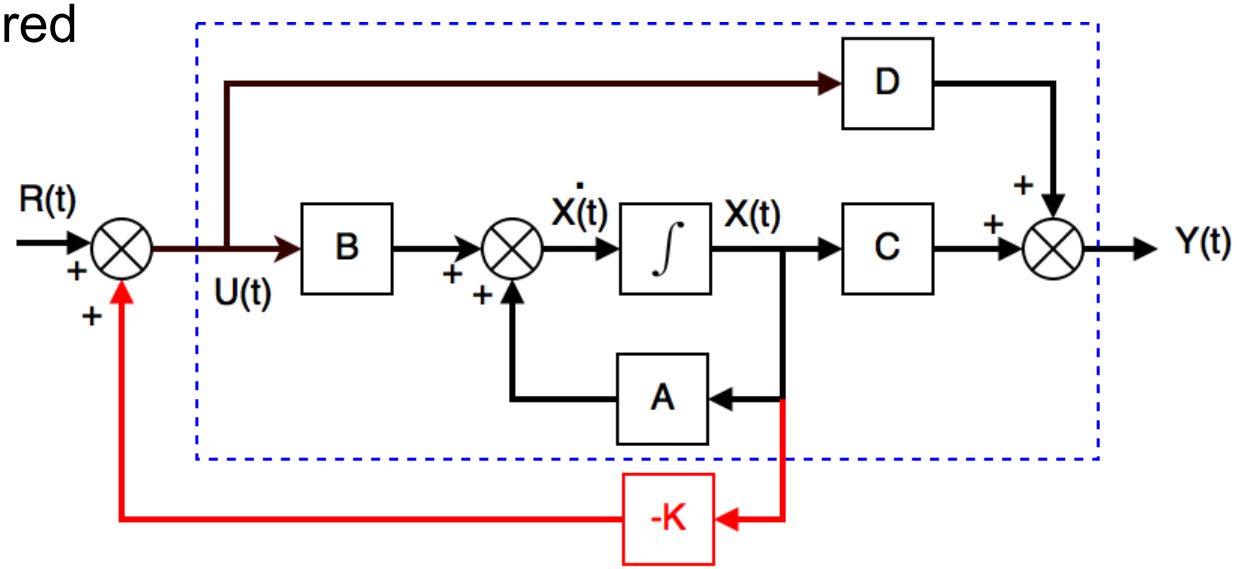
$$\dot{X} = AX + BU \qquad \text{and} \qquad Y = CX + DU$$

To achieve state feedback control we substituting in

$$U = -KX$$ Thus giving

$$\dot{X} = \left(A - BK\right)X \qquad \text{and} \qquad Y = \left(C - DK\right)X$$

Corresponding signal flow graph is shown below with state feedback path depicted in red



Generally we need to choose the gains using computation, e.g. Matlab
This can be done by pole placement or using optimality criteria

Ackermann SFC gain calculation

- Pole placement can be achieved using the Ackermann formula
- We can also use the Matlab place command

```
>> help place
 place  Closed-loop pole assignment using state feedback.

    K = place(A,B,P) computes a state-feedback matrix K such that
    the eigenvalues of A-B*K are those specified in the vector P.
    No eigenvalue should have a multiplicity greater than the
    number of inputs.

    [K,PREC] = place(A,B,P) returns PREC, an estimate of how
    closely the eigenvalues of A-B*K match the specified locations P
    (PREC measures the number of accurate decimal digits in the actual
    closed-loop poles).  A warning is issued if some nonzero closed-loop
    pole is more than 10% off from the desired location.

    See also acker.

    Reference page for place
```

```
% compute SFC gains to set eigenvalues
PX=8 * [-1 -1.1 ];
ssm.K = place(ssm.A,ssm.B,PX);
disp(ssm.K);
```

ROCO218: Control Engineering
Dr Ian Howard

Lecture 6

Analytical solution of the state space equations

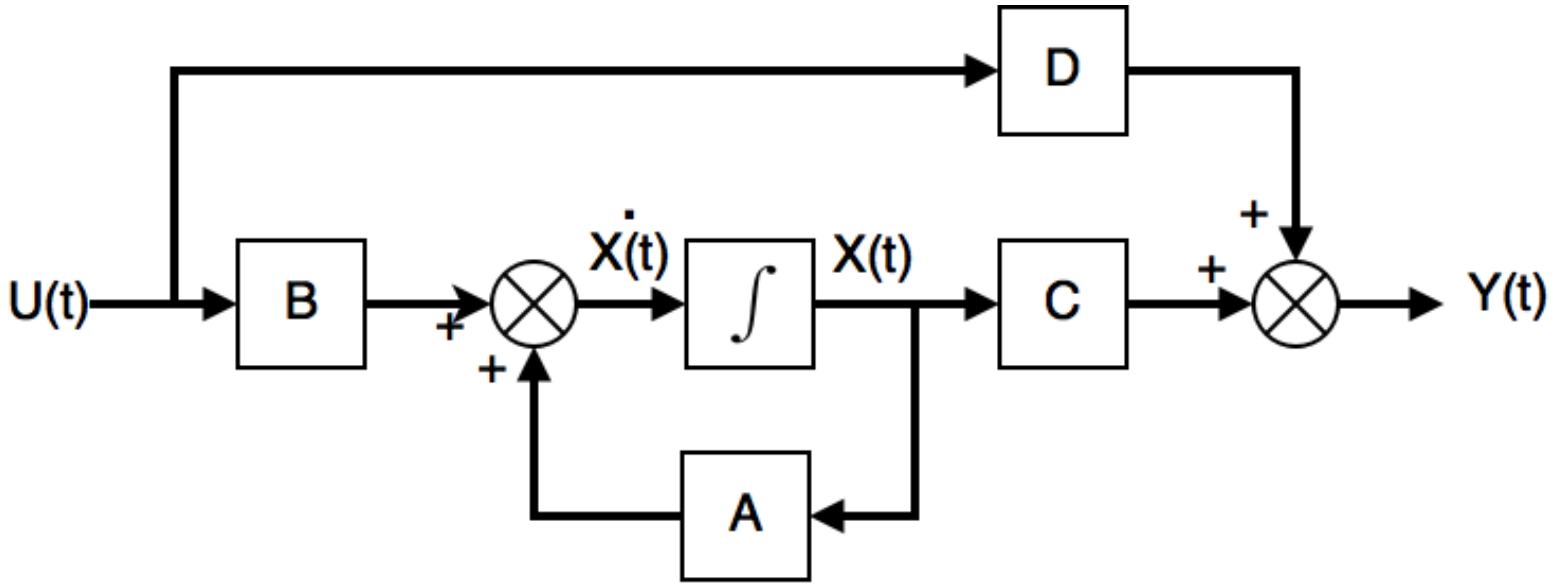
Solution to the state space equation

The matrix state space equations that describe a MIMO system are

$$\dot{x}(t) = Ax(t) + Bu(t)$$

$$y(t) = Cx(t) + Du(t)$$

As we saw previously they cam be described by the the following system structure



Where A,B,C and D are matrices and the integrator block performs vector integration

Solution to the state space equation

- Consider the single 1st order homogeneous differential equation of the form

$\dot{x}(t) = \alpha x(t)$ with initial conditions $x(0) = x_o$

- This equation has the solution

$x(t) = x_o e^{\alpha t}$

- It is easy to show this by differentiating both size of the equation as follows:

So given

$x(t) = x_o e^{\alpha t}$

$\Rightarrow \dot{x}(t) = \frac{d}{dt}\left(x_o e^{\alpha t}\right) = \alpha x_o e^{\alpha t} = \alpha x(t)$

Remember for an exponential

$$\frac{d}{dt}\left(e^{\alpha t}\right) = \alpha e^{\alpha t}$$

- Also when time t = 0 then

$x(0) = x_o e^0 = x_o$

Solution to the state space equation

So the scalar equation

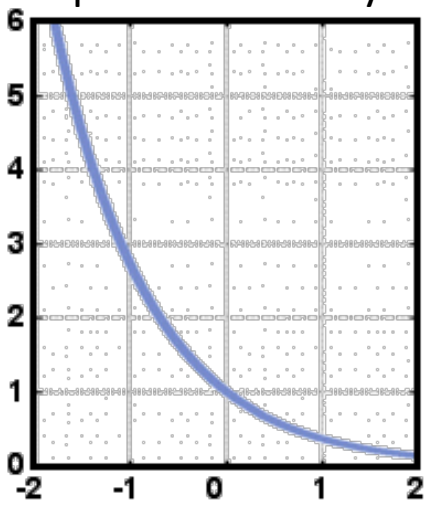
$$\dot{x}(t) = \alpha x(t)$$

Has the solution

$$x(t) = x_o e^{\alpha t} \qquad \text{where} \qquad x(0) = x_o$$

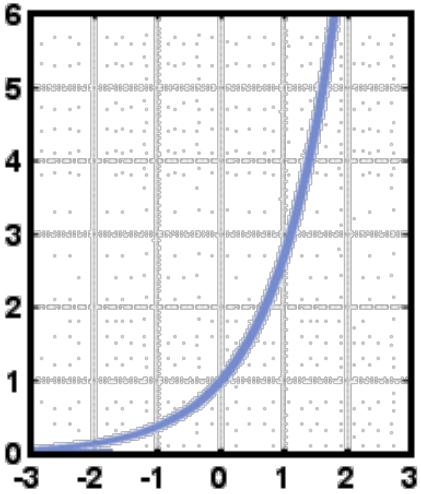
If α < 0 then the solution will have exponential decay and finally go to zero
So the system will be stable

exponential decay



If α = 0 then the solution have a constant value

exponential growth



If α > 0 then the solution have exponential growth
So the system will be unstable

Solution to state space equation without input

- State space models describe dynamical systems using a set of 1st order ODEs in matrix representation.
- That is

$$\dot{X}(t) = AX(t) + BU(t)$$

$$Y(t) = CX(t) + DU(t)$$

- Here X(t), Y(t) and U(t) are vectors and A,B C and D are matrices
- To solve the first equation, we will first ignore the input term u(t)
- We then need to first define the matrix exponential
- That is the meaning of the expression:

e^{At} where A is a matrix

The matrix exponential

The function e^{At} must satisfy the following conditions

$$\frac{d}{dt}\left(e^{At}\right) = Ae^{At} \quad \text{and} \quad e^{0} = I$$

The scalar exponential is defined as

$$e^{a} = 1 + a + \frac{a^2}{2!} + \frac{a^3}{3!} + \ldots + \frac{a^n}{n!} + \ldots$$

The matrix exponential is similarly defined as

$$e^{A} = I + A + \frac{A^2}{2!} + \frac{A^3}{3!} + \ldots + \frac{A^n}{n!} + \ldots$$

By also including the time term we therefore have the expression

$$e^{At} = I + At + \frac{(At)^2}{2!} + \frac{(At)^3}{3!} + \ldots + \frac{(At)^n}{n!} + \ldots$$

Checking properties of the matrix exponential

The matrix exponential has the following properties

$$e^0 = I + 0 + \frac{0^2}{2!} + \frac{0^3}{3!} + \ldots + \frac{0^n}{n!} + \ldots \quad \Rightarrow e^0 = I$$

The differential of the matrix exponential is given by

$$\frac{d}{dt}\left(e^{At}\right) = \frac{d}{dt}\left(I + At + \frac{(At)^2}{2!} + \frac{(At)^3}{3!} + \ldots + \frac{(At)^n}{n!} + \ldots \right)$$

$$= 0 + A + A\frac{2At}{2!} + A\frac{3(At)^2}{3!} + \ldots + A\frac{n(At)^{n-1}}{n!} + \ldots$$

$$= A\left(I + At + \frac{(At)^2}{2!} + \frac{(At)^3}{3!} + \ldots + \frac{(At)^{n-1}}{(n-1)!} + \ldots \right)$$

$$\Rightarrow \frac{d}{dt}\left(e^{At}\right) = Ae^{At}$$

Checking properties of the matrix exponential

Therefore the solution of the equation

$$\dot{X}(t) = AX(t) + BU(t)$$

in the absence of input is given by

$$x_h(t) = x(0)e^{At}$$

Which can be written as

$$x_h(t) = \Phi(t)x(0)$$

Where the state transition matrix is written as

$$\Phi(t) = e^{At}$$

Example 1: system output with no input

Consider the system with the state equations

$$\dot{x}_1 = -2x_1 + u$$

$$\dot{x}_2 = x_1 - x_2$$

With initial conditions

$$\dot{x}_1(0) = 3$$

$$\dot{x}_2(0) = 2$$

First we write down the equations in matrix form

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -2 & 0 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u$$

Therefore the system matrix A is given by

$$A = \begin{bmatrix} -2 & 0 \\ 1 & -1 \end{bmatrix}$$

Example 1: system output with no input

Given the system matrix

$$A = \begin{bmatrix} -2 & 0 \\ 1 & -1 \end{bmatrix}$$

This leads to the state transition matrix expression

$$\Phi(t) = e^{At}$$

Since the matrix exponential is defined as

$$e^{At} = I + At + \frac{A^2t^2}{2!} + \frac{A^3t^3}{3!} + \ldots + \frac{A^nt^n}{n!} + \ldots$$

To evaluate it we first need to calculate the terms so we can form the sum

Example 1: system output with no input

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$ Identity matrix

$$A = \begin{bmatrix} -2 & 0 \\ 1 & -1 \end{bmatrix}$$ System matrix

$$A^2 = \begin{bmatrix} -2 & 0 \\ 1 & -1 \end{bmatrix}\begin{bmatrix} -2 & 0 \\ 1 & -1 \end{bmatrix} = \begin{bmatrix} 4 & 0 \\ -3 & 1 \end{bmatrix}$$ Squared system matrix

$$A^3 = \begin{bmatrix} 4 & 0 \\ -3 & 1 \end{bmatrix}\begin{bmatrix} -2 & 0 \\ 1 & -1 \end{bmatrix} = \begin{bmatrix} -8 & 0 \\ 7 & -1 \end{bmatrix}$$ Cubed system matrix

And so on …

Example 1: system output with no input

Substituting in these terms

$$\Rightarrow e^{At} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} -2 & 0 \\ 1 & -1 \end{bmatrix} t + \begin{bmatrix} 4 & 0 \\ -3 & 1 \end{bmatrix} \frac{t^2}{2!} + \begin{bmatrix} -8 & 0 \\ 7 & -1 \end{bmatrix} \frac{t^3}{3!} + \ldots$$

$$\Rightarrow e^{At} = \begin{bmatrix} \left(1 - 2t + 4\frac{t^2}{2!} - 8\frac{t^3}{3!} + \ldots\right) & 0 \\ \left(0 + t - 3\frac{t^2}{2!} + 7\frac{t^3}{3!} + \ldots\right) & \left(1 - t + \frac{t^2}{2!} - \frac{t^3}{3!} + \ldots\right) \end{bmatrix}$$

Example 1: system output with no input

It can be seen that

$$1 - 2t + 4\frac{t^2}{2!} - 8\frac{t^3}{3!} + \ldots = e^{-2t}$$

$$1 - t + \frac{t^2}{2!} - \frac{t^3}{3!} + \ldots = e^{-t}$$

$$0 + t - 3\frac{t^2}{2!} + 7\frac{t^3}{3!} + \ldots = e^{-t} - e^{-2t}$$

$$\Rightarrow e^{At} = \begin{bmatrix} e^{-2t} & 0 \\ e^{-t} - e^{-2t} & e^{-t} \end{bmatrix}$$

Example 1: system output with no input

So from our expression for the state transition matrix

$$\Phi(t) = e^{At} = \begin{bmatrix} e^{-2t} & 0 \\ e^{-t} - e^{-2t} & e^{-t} \end{bmatrix}$$

The homogeneous response to initial conditions is given by

$$x_h(t) = \Phi(t)x(0)$$

$$\Rightarrow x_h(t) = \begin{bmatrix} e^{-2t} & 0 \\ e^{-t} - e^{-2t} & e^{-t} \end{bmatrix} \begin{bmatrix} x_1(0) \\ x_2(0) \end{bmatrix}$$

With initial conditions

$$\dot{x}_1(0) = 3$$

$$\dot{x}_2(0) = 2$$

$$\Rightarrow x_1(t) = e^{-2t}x_1(0) \; = 3e^{-2t}x_1$$

$$\Rightarrow x_2(t) = \left(e^{-t} - e^{-2t}\right)x_1(0) + e^{-t}x_2(0) \; = 2\left(e^{-t} - e^{-2t}\right) + 3e^{-t}$$

$$= 5e^{-t} - 2e^{-2t}$$

Example 2: system output with no input

Given a system with a diagonal system matrix

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u$$

Where the system matrix is

$$A = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

We calculate the matrix exponential as before

$$e^{At} = I + At + \frac{A^2 t^2}{2!} + \frac{A^3 t^3}{3!} + \ldots + \frac{A^n t^n}{n!} + \ldots$$

Example 2: system output with no input

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$ Identity matrix

$$A = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$ System matrix

$$A^2 = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}\begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} = \begin{bmatrix} \lambda_1^2 & 0 \\ 0 & \lambda_2^2 \end{bmatrix}$$ Squared system matrix

$$A^3 = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}\begin{bmatrix} \lambda_1^2 & 0 \\ 0 & \lambda_2^2 \end{bmatrix} = \begin{bmatrix} \lambda_1^3 & 0 \\ 0 & \lambda_2^3 \end{bmatrix}$$ Cubed system matrix

And so on …

Example 2: system output with no input

Substituting in these terms

$$\Rightarrow e^{At} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} t + \begin{bmatrix} \lambda_1^2 & 0 \\ 0 & \lambda_2^2 \end{bmatrix} \frac{t^2}{2!} + \begin{bmatrix} \lambda_1^3 & 0 \\ 0 & \lambda_2^3 \end{bmatrix} \frac{t^3}{3!} + \ldots$$

$$\Rightarrow e^{At} = \begin{bmatrix} \left(1 + \lambda_1 t + \frac{(\lambda_1 t)^2}{2!} + \frac{(\lambda_1 t)^3}{3!} + \ldots\right) & 0 \\ 0 & \left(1 + \lambda_2 t + \frac{(\lambda_2 t)^2}{2!} + \frac{(\lambda_3 t)^3}{3!} + \ldots\right) \end{bmatrix}$$

Example system output in case of no input

It can be seen that

$$1+\lambda_1 t+\frac{(\lambda_1 t)^2}{2!}+\frac{(\lambda_1 t)^3}{3!}+\ldots=e^{\lambda_1 t}$$

$$1+\lambda_2 t+\frac{(\lambda_2 t)^2}{2!}+\frac{(\lambda_2 t)^3}{3!}+\ldots=e^{\lambda_2 t}$$

$$\Rightarrow e^{At}=\begin{bmatrix} e^{\lambda_1 t} & 0 \\ 0 & e^{\lambda_2 t} \end{bmatrix}=\Phi(t)$$

Homogeneous response to initial conditions is

$$x_h(t)=\Phi(t)x(0) \qquad \Rightarrow x_h(t)=\begin{bmatrix} e^{\lambda_1 t} & 0 \\ 0 & e^{\lambda_2 t} \end{bmatrix}\begin{bmatrix} x_1(0) \\ x_2(0) \end{bmatrix}$$

$$\Rightarrow x_1(t)=e^{\lambda_1 t}x_1(0)$$

$$\Rightarrow x_2(t)=e^{\lambda_2 t}x_2(0)$$

Note that since system matrix A was diagonal outputs are now decoupled!

System output with input term

In general for a state space system driven with an input u(t)

$$\dot{x}(t) = Ax(t) + Bu(t)$$

$$y(t) = Cx(t) + Du(t)$$

Taking Laplace transforms with initial conditions

$$sX(s) - x(0) = AX(s) + BU(s)$$

$$\Rightarrow (sI - A)X(s) = x(0) + BU(s)$$

$$\Rightarrow X(s) = (sI - A)^{-1}x(0) + (sI - A)^{-1}BU(s)$$

Before taking the inverse Laplace transforms we note that

$$e^{At} = L^{-1}\left\{(sI - A)^{-1}\right\}$$

$$\int_0^t f(\tau)g(t-\tau)d\tau = L^{-1}\left\{F(s)G(s)\right\}$$

A product in the s-domain becomes a convolution in the time domain

System output with input term

$$\Rightarrow X(s) = (sI - A)^{-1} x(0) + (sI - A)^{-1} BU(s)$$

Taking inverse Laplace transforms

$$\Rightarrow x(t) = e^{At} x(0) + \int_0^t e^{A(t-\tau)} Bu(\tau) d\tau$$

Substituting into the output equation this gives

$$\Rightarrow y(t) = Ce^{At} x(0) + \int_0^t Ce^{A(t-\tau)} Bu(\tau) d\tau + Du(t)$$

ROCO218: Control Engineering
Dr Ian Howard

Lecture 6

Numeric solutions of state space equations

Revision: Using ode45 to perform integration

help ode45

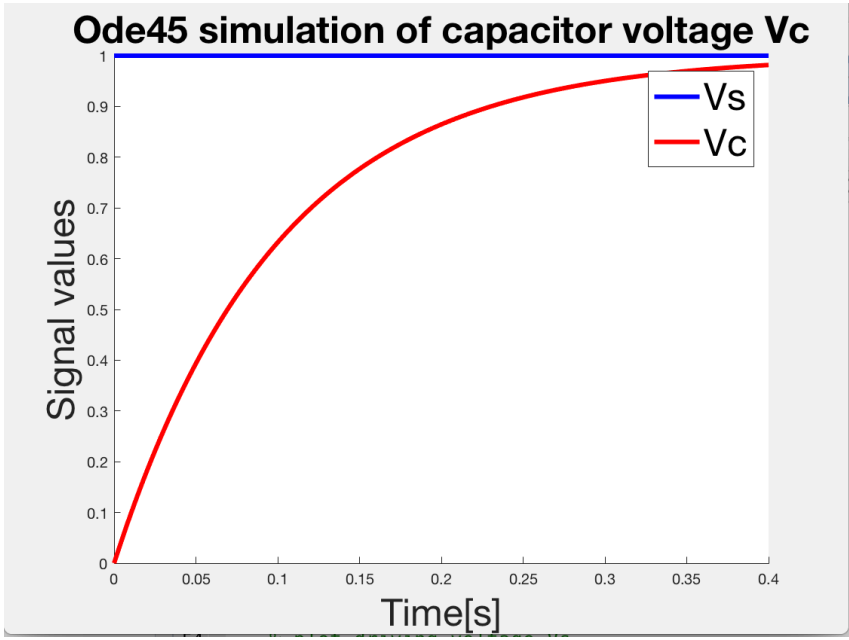
- Solve non-stiff differential equations, medium order method.
- [TOUT,YOUT] = ode45(ODEFUN,TSPAN,Y0)
- with TSPAN = [T0 TFINAL] integrates the system of differential equations y' = f(t,y) from time T0 to TFINAL with initial conditions Y0.
- ODEFUN is a function handle.
- For a scalar T and a vector Y, ODEFUN(T,Y) must return a column vector corresponding to f(t,y).
- Each row in the solution array YOUT corresponds to a time returned in the column vector TOUT.
- To obtain solutions at specific times T0,T1,...,TFINAL (all increasing or all decreasing), use TSPAN = [T0 T1 ... TFINAL].

Revision: Using ode45 to perform integration

```
function VcDot = RCDE(Vc, R, C, Vs)
% Vc is capatotor voltage
% R resistance
% C capacitance
% Vs input control applied voltage
% xdot is the returned 1st time derivative of capacitor voltage

% differential equation for capacitor voltage
VcDot = (Vs - Vc)/(R*C);
```

```
% parameters
timeLen = 0.4; % time duration of seconds
samplingRate = 1000; %sampling frequency in Hz
freq = 5; % signal frequency in  Hz

% resistance 1000 ohms
R = 1000;
% capacitor value 100 microFarads
C = 100e-6;
% step in secoinds
h = 1/samplingRate;
% time sampling points
T = 0:h:timeLen;

% number of data samples
sampleCnt = length(T);

% build target unity driving voltage
Vs = ones(size(T));

% initial conditions - no capacotr votlage
V0 = 0;

% run simulation
[tSS, Vc] = ode45(@(t,x)RCDE(x, R, C,  Vs(1)), T, V0);
```

```
% plot the figure
figure
% want all plots to remain on figure
hold on
% write a title
h = title('Ode45 simulation of capacitor voltage Vc');
set(h, 'FontSize', 25)
% plot driving voltage Vs
h = plot(T, Vs, 'b');
set(h,'LineWidth', 3);
% plot capacitor voltage Vx
h = plot(T, Vc, 'r');
set(h,'LineWidth', 3);
% label plots
h=legend('Vs','Vc');
set(h, 'FontSize', 25)
% label the axes
h= xlabel('Time[s]')
set(h, 'FontSize', 25)
h = ylabel('Signal values');
set(h, 'FontSize', 25)
```



Using ode45 to perform SFC integration

- ode45 solve systems of equations of the form $y'=f(t,y)$
- To use ode45 you need to write a function that returns an xDot vector
- This is what the state space dynamics equation does. That is

$$\dot{X} = AX + BU$$

- Therefore write a function and pass it the parameters: A, B and u and return xDot

```
function xDot = SSSimulate(X, A, B, u)
% state space model
% x is state
% A,B are state space matrices
% u is ther control input
% xdot is the returned 1st time derivative of state

% implement model
xDot = A * X + B * u;
```

- Then ode45 can integrate it for you!

```
% compute output of linearized state space system
% ssmP.A is A matrix
% ssmP.B is B matrix
% x0 are initial conditions
% note input to state space model is full state feedback
[tSS, xDirectK] = ode45(@(t,y)SSSimulate(y, ssmP.A, ssmP.B,  -ssmP.K * y), params.t, x0);
```

Revision: Euler's method of integration

- This is Euler's method to solve a 1st order differential equation!
- It uses a local linear approximation to the gradient
- We can iteratively estimate the value of the function at the next step
- Given the relationship

$$y' = f(t, y),\ y(t_0) = y_0$$

$$y_{n+1} = y_n + h.f\left(t_n, y_n\right)$$

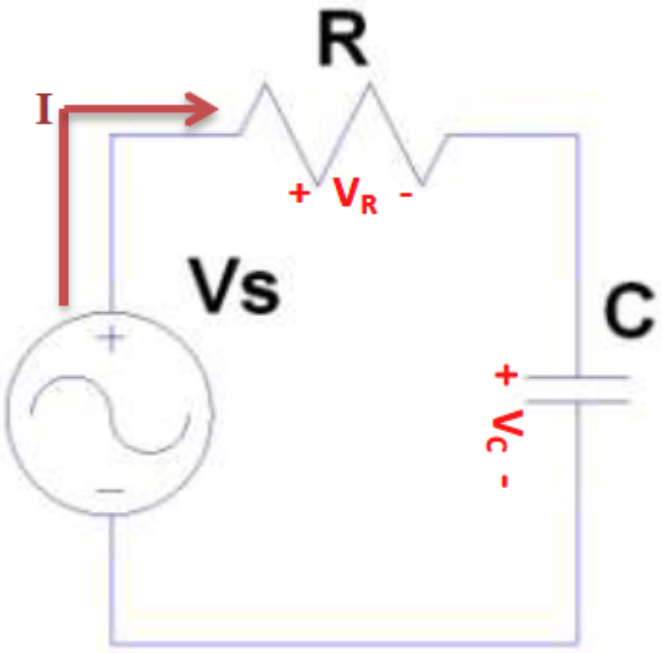
Where
t is time
y is the output value of the function
At initial time t_o the output is y_o
At future time $t_{(n+1)}$ the output is $y_{(n+1)}$
h is the step size

- Notice the simulation error will rise as the step size increases!

Revision: Implementing Euler integration in Matlab

We can write simple script using the update equation to estimate capacitor voltage is given by the equation

$$\Rightarrow V_C(t+1) = V_C(t) + h\frac{1}{RC}\left(V_S - V_C(t)\right)$$




```
function Vc = GetCapVoltage(R, C, Vs, h)
% calling function to get Vc
% given resistance R,capacitance C, driving voltage Vs
% and time step h

% got input sample count
len = length(Vs)-1;

% set initial votage to zero
Vc(1) = 0;

% loop over all samples -1
for n = 1:len

    % Euler update equation
    Vc(n+1) = Vc(n) + h * (1/(R*C)) * (Vs(n) - Vc(n));
end

end
```

Using Euler method to perform integration for SFC

- We need to solve the equations

$$\dot{X} = AX + BU \qquad Y = CX + DU \quad \text{Where} \qquad U = -KX$$

- This time we need to iteratively estimate the state vector X
- Thus starting by setting the initial conditions

$$X = X_0$$

- We then have to calculate the recurrence relationships in a loop

$$U(k) = -KX(k)$$

$$X(k+1) = X(k) + h\left(AX(k) + BU(k)\right)$$

- Where state X and input U are now described by a step index to make the recurrence clear
- Again we will need t define the time step h

```
% randomly perturb intial condition
% [theta(0) thetaDot(0))]
x0 = [0; 2 * randn; ];
[yDirectK, tDirectK, xDirectK] = SimulateSFCC2x2(ssm.A, ssm.B, ssm.C, ssm.D, ssm.K, params.t, x0);
```

Using Euler method to perform integration for SFC

```
function [y, t, xout] = SimulateSFCArduino2x2(A, B, C, D, K, t, x0)
% simulate state space feedback control using C-language compatible formulation
% performs integration by Euler's method
% A,B,C,D are state space matrices and k is state feeeback gain
% t is time at each sample,  x0 is initial state
% retuns outout y, each time t and full state at each update
% Author: Dr. Ian Howard% all rights reserved

% get signal length
len = length(t);

% init outout
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-language compatible formulation
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 states only
    y(idx)= C(1) * x(1) + C(2) * x(2) + D(1) * u(idx);
end
```

Interlude

10 minute break

ROCO218: Control Engineering
Dr Ian Howard

Lecture 6

Linearization of state space models

Linear and non-linear models

We will focus on the analysis of linear models

Linear models

- Time invariant model

$$\dot{x}(t) = Ax(t) + Bu(t)$$

$$y(t) = Cx(t) + Du(t)$$

- Time variant model

$$\dot{x}(t) = A(t)x(t) + B(t)u(t)$$

$$y(t) = C(t)x(t) + D(t)u(t)$$

Non-linear models

- Time invariant model

$$\dot{x}(t) = f\left(x(t), u(t)\right)$$

$$y(t) = f\left(x(t).u(t)\right)$$

- Time variant model

$$\dot{x}(t) = f\left(x(t), u(t), t\right)$$

$$y(t) = f\left(x(t).u(t), t\right)$$

Linear and non-linear models

- We will consider time invariant- model with continuous time

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

- However we note that analysis of discrete time models

$$x(k+1) = Ax(k) + Bu(k)$$
$$y(k) = Cx(k) + Du(k)$$

can be treated in a similar way, but uses difference equations rather than differential equations

Linear and non-linear models

- We can find linear approximations to non-linear models around their equilibrium points and then describe them as we would linear time invariant systems using state space methods

- Given a non-linear model

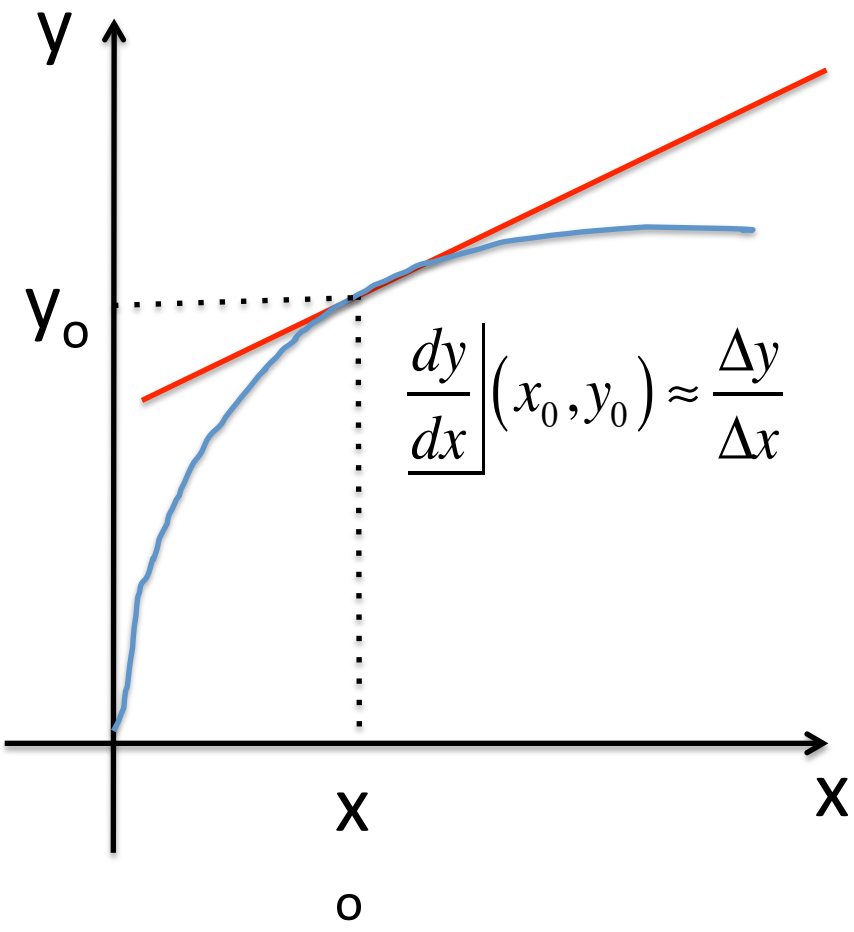
$$\dot{x}(t) = f\left(x(t), u(t)\right)$$

$$y(t) = f\left(x(t).u(t)\right)$$

- We first need to find the equilibrium position.
- This occurs when the system is stationary and therefore when the differential is equal to zero

$$\dot{x}(t) = f\left(x(t), u(t)\right) = 0$$

Local linear approximation of gradient



In the 1-dimensional case, we can calculate the gradient of a curve at any point by calculating the simple derivative at that point

We can then model the system close to that point with a linear approximation

For example we can model local behavior at the point (x_o, y_o) using a line with the same gradient at that point

If we have a multidimensional system we need to compute the partial derivatives

Ordinary derivatives

- With a function of a single variable

$$z = f(x)$$

- The rate of change of z w.r.t. x is given by the full derivative

$$\frac{dz}{dx} = f'(x)$$

- Straight d symbols used to denote total differential where all variables (x) are allowed to vary

Ordinary derivatives

- E.g. given the equation where k is a constant

$$z = (k - x)^2$$

- Let $y = k - x \qquad \Rightarrow z = y^2$

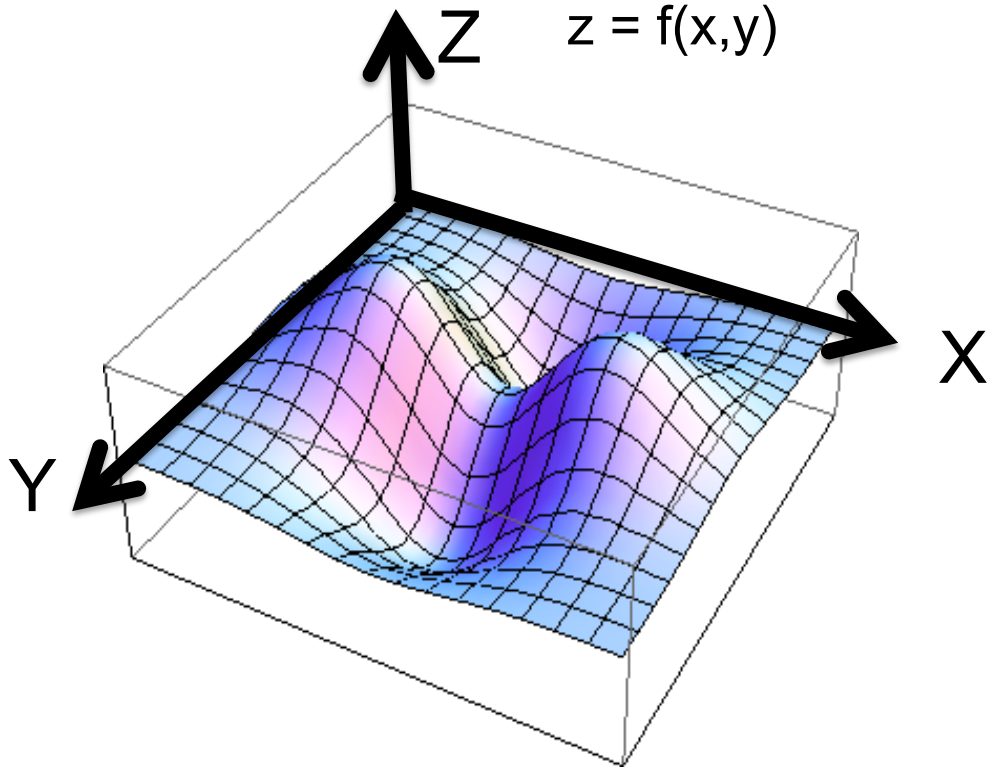
- Use the chain rule to differentiate a function

$$\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx}$$

$$\Rightarrow \frac{dy}{dx} = \frac{d}{dx}(k - x) \quad = -1 \qquad \Rightarrow \frac{dz}{dy} = \frac{d}{dy}\left(y^2\right) \quad = 2y$$

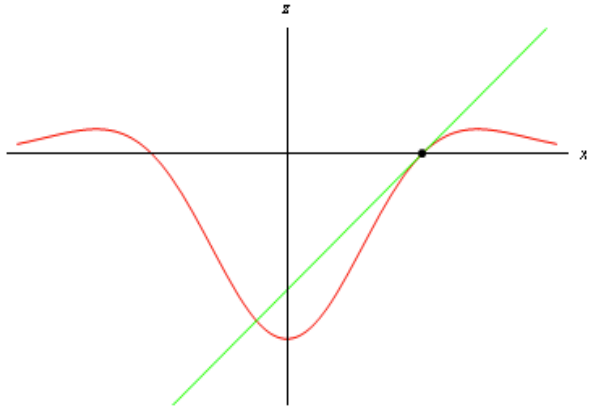
$$\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx} \quad = 2y \cdot -1 \quad = 2(k - x) \cdot -1 \quad = -2(k - x)$$

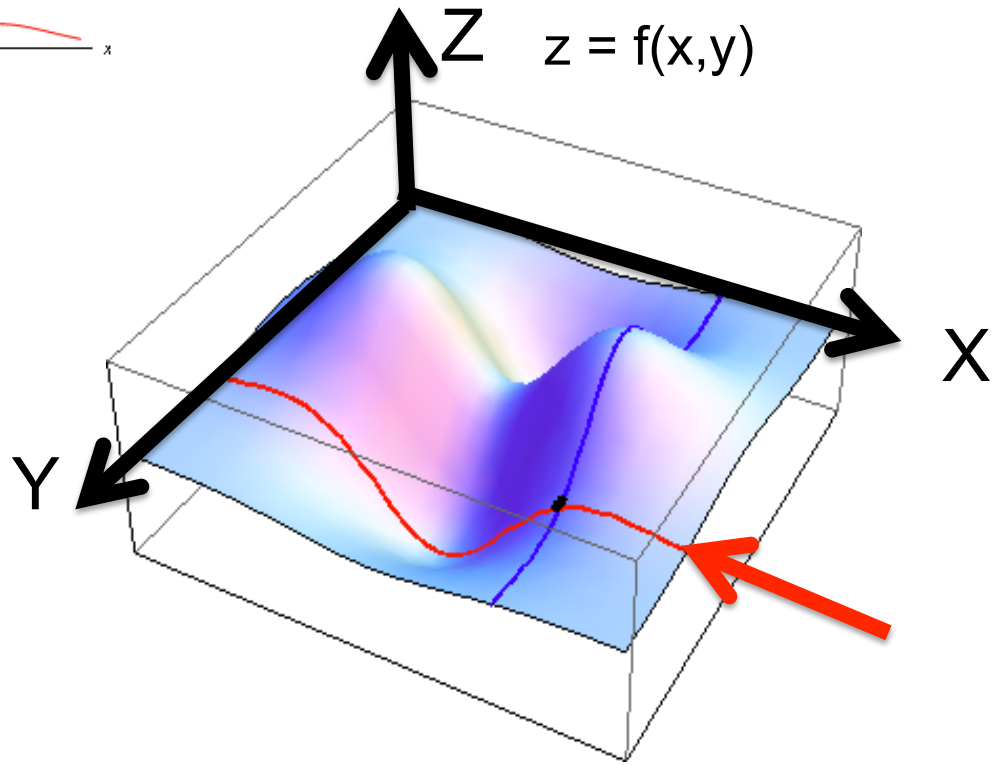
Functions of multiple variables



- Consider function of two valuables
- z = f(x,y)

Gradient along two dimensions



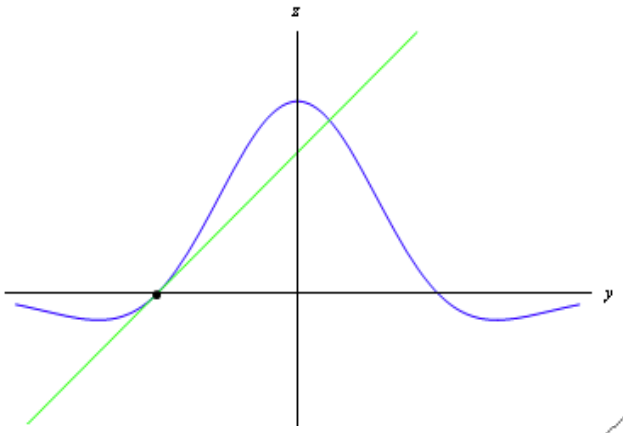


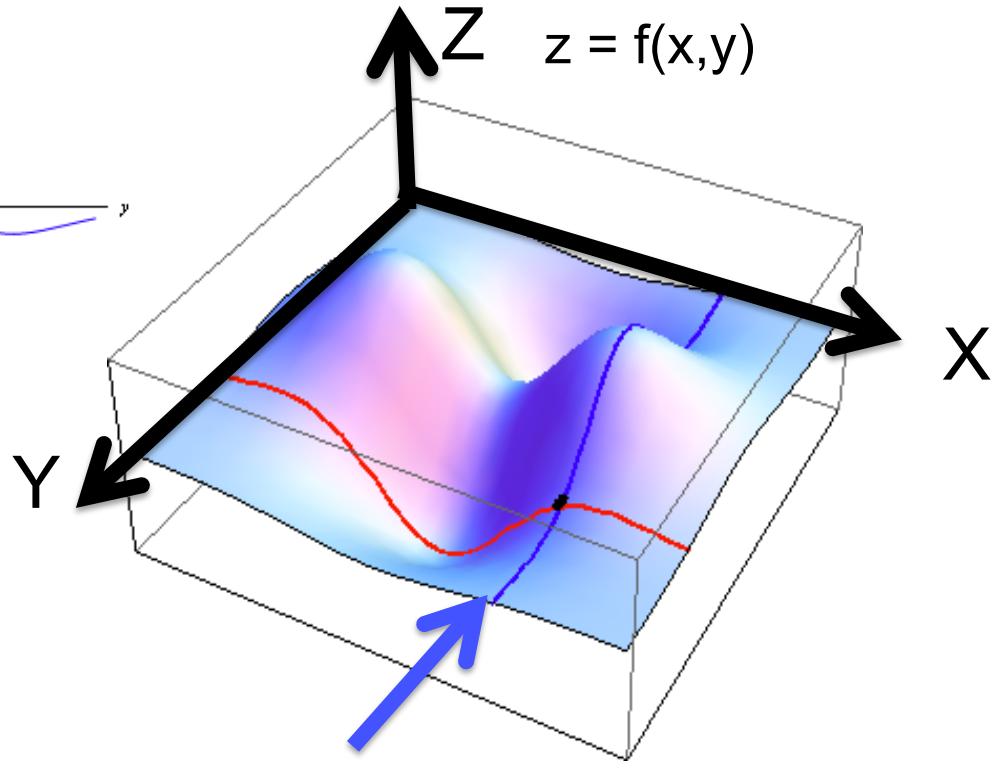
Can find gradient by differentiating curve w.r.t. x and then evaluating at a given point

This partial differential is denoted by $\frac{\partial z}{\partial x}$

Keep y constant and follow path along x-axis

Gradient along two dimensions





Can find gradient by differentiating curve w.r.t. y and then evaluating at a given point

This is denoted by $\frac{\partial z}{\partial y}$

- Keep x constant and follow path along y-axis

Partial derivatives

- In the case when a function depends on 2 variables

$$z = f(x, y)$$

- The partial derivative of z w.r.t. x is the ordinary differential w.r.t. x while y is held constant and written as

$$\frac{\partial z}{\partial x}$$

- Curly ∂ symbols used to denote partial differential wrt valuable x

- The partial derivative of z w.r.t. y is the ordinary differential w.r.t. y while x is held constant and written as

$$\frac{\partial z}{\partial y}$$

- Curly ∂ symbols used to denote partial differential w.r.t. valuable y

Partial derivatives

- E.g. given the equation that is a function of x and y

$$z = 3x^2 + 2xy + y^2$$

$$\Rightarrow \frac{\partial z}{\partial x} = 6x + 2y$$ Since all y terms are treated as constant

Similarly

$$\Rightarrow \frac{\partial z}{\partial y} = 2x + 2y$$ Since all x terms are treated as constant

1-dimensional Taylor series approximation

- State space models are of the form

$$\frac{dx}{dt} = f(x)$$

- i.e., a 1st order differential is equated to some function

- The function f(x) can be approximated using a Taylor series around a steady-state operating point x_s

$$f(x) = f(x_s) + \left(\frac{\partial f}{\partial x} | x_s \right)(x - x_s) + \left(\frac{\partial^2 f}{\partial x^2} | x_s \right)(x - x_s) + \ldots HOTs$$

- If we ignore terms higher than the 1st differential then we have the approximation

$$f(x) = f(x_s) + \left(\frac{\partial f}{\partial x} | x_s \right)(x - x_s)$$

1-dimensional Taylor series approximation

- At steady state equilibrium the system is stationary
- Therefore we have the condition that

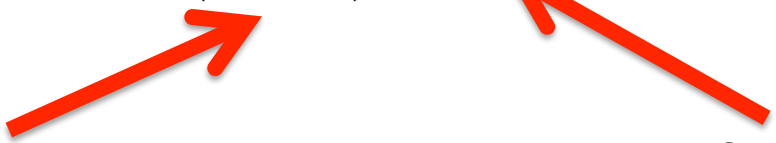
$$\frac{dx}{dt} = f(x_s) = 0$$

- Therefore the 1st order Taylor approximation

$$f(x) = f(x_s) + \left(\frac{\partial f}{\partial x} \middle| x_s \right) \left(x - x_s \right)$$

since the 1st term on the RHS is zero so the expression simplifies to

$$f(x) = \left(\frac{\partial f}{\partial x} \middle| x_s \right) \left(x - x_s \right)$$



Gradient at the equilibrium

Deviation from the equilibrium

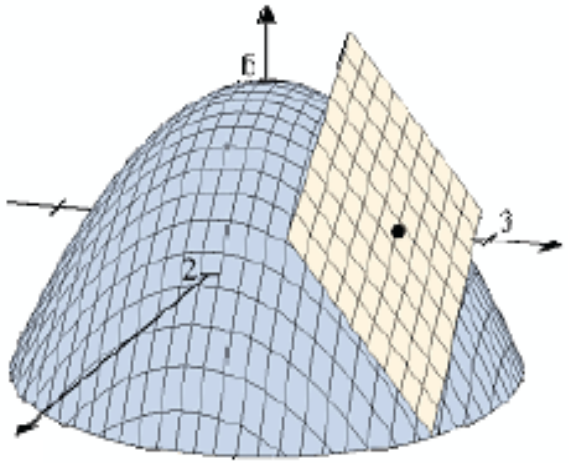
Linearization of high-order non-linear systems

If we have the equation $\dot{x} = F(x)$

Where $F(x) = (f_1(x), f_2(x), \ldots f_n(x))$

- To linearize a high order non-linear system using a Taylor expansion, we need to calculate the Jacobian matrix and evaluate it at its equilibrium position(s)
- This leads to a local approximation of the function

$$\Rightarrow J = \left(\frac{\partial f_i}{\partial x_j} \right) \Big| x = x_{equilibrium} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & & \ddots & \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}$$



Jacobian example

If we have the equation

$$f(x,y)=\begin{bmatrix} \frac{x^2 y}{2} \\ x+\sin y \end{bmatrix} \quad \Rightarrow f_1(x,y)=\frac{x^2 y}{2}$$

$$\Rightarrow f_2(x,y)=x+\sin y$$

$$\Rightarrow J_f(x,y)=\begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} \end{bmatrix}$$

It is easy to see that

$$\frac{\partial f_1}{\partial x}=xy$$

$$\frac{\partial f_1}{\partial y}=\frac{x^2}{2}$$

$$\frac{\partial f_2}{\partial x}=1$$

$$\frac{\partial f_2}{\partial y}=\cos y$$

$$\Rightarrow J_f(x,y)=\begin{bmatrix} xy & \frac{x^2}{2} \\ 1 & \cos y \end{bmatrix}$$

ROCO218: Control Engineering
Dr Ian Howard

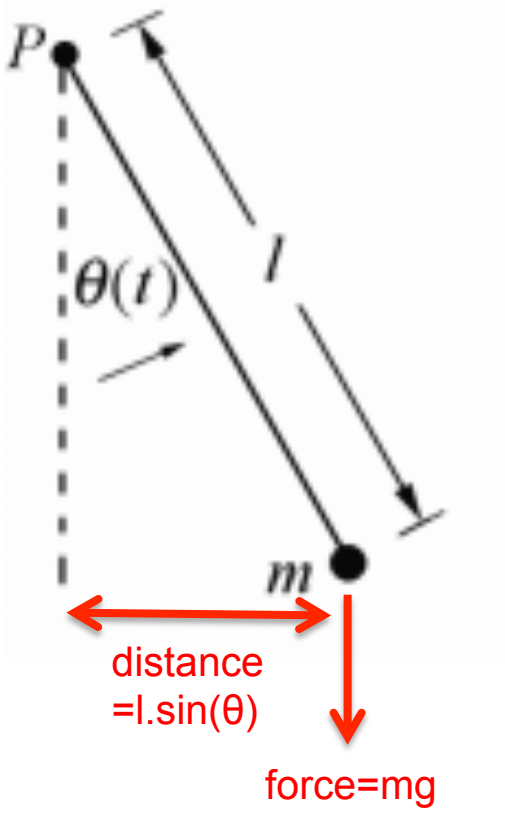
Lecture 6

Example state space model
of a simple pendulum

Fixed point mass pendulum

Gravitation force acting on mass leads to torque about the pivot given by

$$T_g = -mgl\sin(\theta)$$



A resistance torque to movement is due to the moment of inertia of system

$$T_I = I\frac{d^2\theta}{dt^2}$$

$$\Rightarrow I\frac{d^2\theta}{dt^2} = -mgl\sin(\theta)$$

For point mass $I = ml^2$

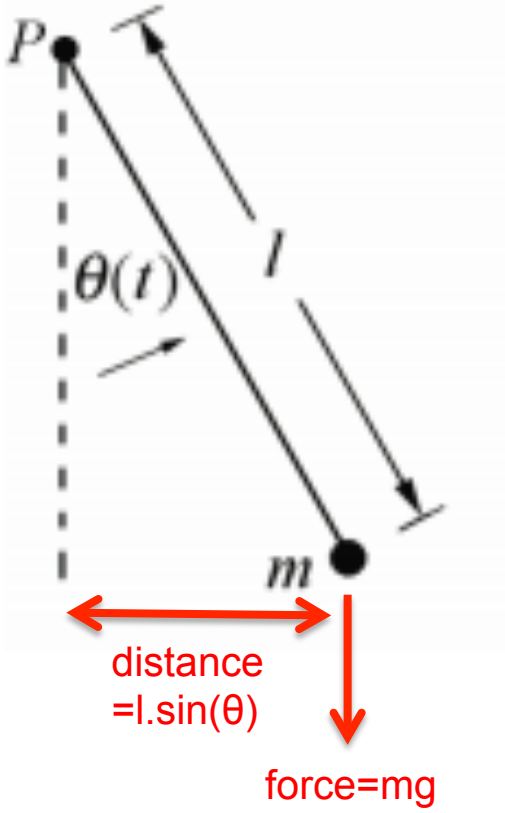
$$\Rightarrow ml^2\frac{d^2\theta}{dt^2} = -mgl\sin(\theta)$$

$$\Rightarrow \frac{d^2\theta}{dt^2} = -\frac{g}{l}\sin(\theta)$$

State space model of simple pendulum

From differential equation for point mass pendulum

$$\frac{d^2\theta}{dt^2} = -\frac{g}{l}\sin(\theta)$$



Choosing state variables of angle and angular velocity

$$\begin{bmatrix} \theta \\ \omega \end{bmatrix}$$

Thus

$$\dot{\theta} = \omega \quad = f_1$$

$$\dot{\omega} = -\frac{g}{l}\sin(\theta) \quad = f_2$$

NB: This is a non-linear equation

State space model of simple pendulum

We now find the equilibrium points

First term is $f_1 = \dot{\theta} = \omega$

Therefore at equilibrium

$$\dot{\theta} = \omega = 0 \qquad \Rightarrow \omega = 0$$

Second term is $f_2 = \dot{\omega} = -\frac{g}{l}\sin(\theta)$

Therefore at equilibrium

$$\dot{\omega} = -\frac{g}{l}\sin(\theta) = 0 \qquad \Rightarrow \theta = 0, .\pi$$

We now need to calculate the Jacobian to linearize the system at the equilibrium point. I.e. we need to compute J where

$$J\big|_{\theta=0} = \begin{bmatrix} \frac{\delta f_1}{\delta\theta} & \frac{\delta f_1}{\delta\omega} \\ \frac{\delta f_2}{\delta\theta} & \frac{\delta f_2}{\delta\omega} \end{bmatrix}$$

State space model of simple pendulum

To calculate the Jacobian we look at the functions:

$$f_1 = \dot{\theta} = \omega \qquad f_2 = \dot{\omega} = -\frac{g}{l}\sin(\theta)$$

$$\Rightarrow \frac{\delta f_1}{\delta \theta} = \frac{\delta}{\delta \theta}(\omega) = 0$$

$$\Rightarrow \frac{\delta f_1}{\delta \omega} = \frac{\delta}{\delta \omega}(\omega) = 1$$

$$\Rightarrow \frac{\delta f_2}{\delta \theta} = \frac{\delta}{\delta \theta}\left(-\frac{g}{l}\sin(\theta)\right) = -\frac{g}{l}\cos(\theta)$$

$$\Rightarrow \frac{\delta f_2}{\delta \omega} = \frac{\delta}{\delta \omega}\left(-\frac{g}{l}\sin(\theta)\right) = 0$$

State space model of simple pendulum

Substituting in the partial derivative terms and evaluating at the zero equilibrium point we get

$$J\big|_{\theta=0} = \begin{bmatrix} \frac{\delta f_1}{\delta\theta} & \frac{\delta f_1}{\delta\omega} \\ \frac{\delta f_2}{\delta\theta} & \frac{\delta f_2}{\delta\omega} \end{bmatrix} \Rightarrow J\big|_{\theta=0} = \begin{bmatrix} 0 & 1 \\ -\frac{g}{l}\cos(\theta) & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{g}{l} & 0 \end{bmatrix}$$

This has the associated linear system

$$\dot{X} = JX \qquad \text{where} \qquad \begin{bmatrix} \dot{\theta} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{g}{l} & 0 \end{bmatrix} \begin{bmatrix} \theta \\ \omega \end{bmatrix}$$

The characteristic polynomial is give by the equation

$$\left|J - \lambda I\right| = 0$$

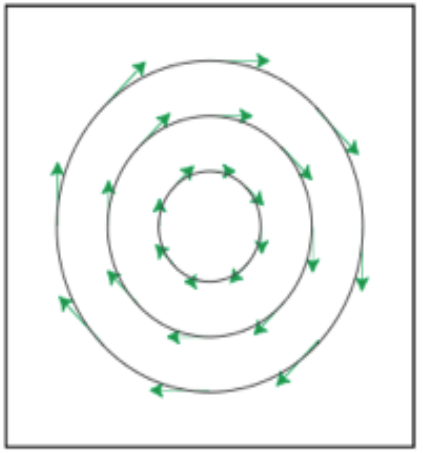
State space model of simple pendulum

We solve the characteristic polynomial for the eigenvalues λ

$$\left\| \begin{bmatrix} -\lambda & 1 \\ -\frac{g}{l} & -\lambda \end{bmatrix} \right\| = 0 \quad \Rightarrow \lambda^2 + \frac{g}{l} = 0 \quad \Rightarrow \lambda^2 = -\frac{g}{l} \quad \Rightarrow \lambda = \pm j\sqrt{\frac{g}{l}}$$

These eigenvalues λ are complex conjugate, this means that there are oscillatory dynamics.

- The dynamics are neutrally stable since the eigenvalues are on the imaginary axis.
- This means that, whatever the initial condition is, the phase plane trajectory motion is a circle centered at the origin and the amplitude determined by the initial condition



State space model of simple pendulum

Substituting in the partial derivative terms and evaluating at the π equilibrium point we get

$$J\Big|_{\theta=\pi} = \begin{bmatrix} \frac{\delta f_1}{\delta\theta} & \frac{\delta f_1}{\delta\omega} \\ \frac{\delta f_2}{\delta\theta} & \frac{\delta f_2}{\delta\omega} \end{bmatrix} \Rightarrow J\Big|_{\theta=\pi} = \begin{bmatrix} 0 & 1 \\ -\frac{g}{l}\cos(\theta) & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \frac{g}{l} & 0 \end{bmatrix}$$

This has the associated linear system

$$\dot{X} = JX \qquad \text{where} \qquad \begin{bmatrix} \dot{\theta} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \frac{g}{l} & 0 \end{bmatrix} \begin{bmatrix} \theta \\ \omega \end{bmatrix}$$

The characteristic polynomial is give by the equation

$$\left|J - \lambda I\right| = 0$$

State space model of simple pendulum

We solve the characteristic polynomial for the eigenvalues λ

$$\left\| \begin{bmatrix} -\lambda & 1 \\ \frac{g}{l} & -\lambda \end{bmatrix} \right\| = 0 \quad \Rightarrow \lambda^2 - \frac{g}{l} = 0 \quad \Rightarrow \lambda^2 = \frac{g}{l} \quad \Rightarrow \lambda = \pm\sqrt{\frac{g}{l}}$$

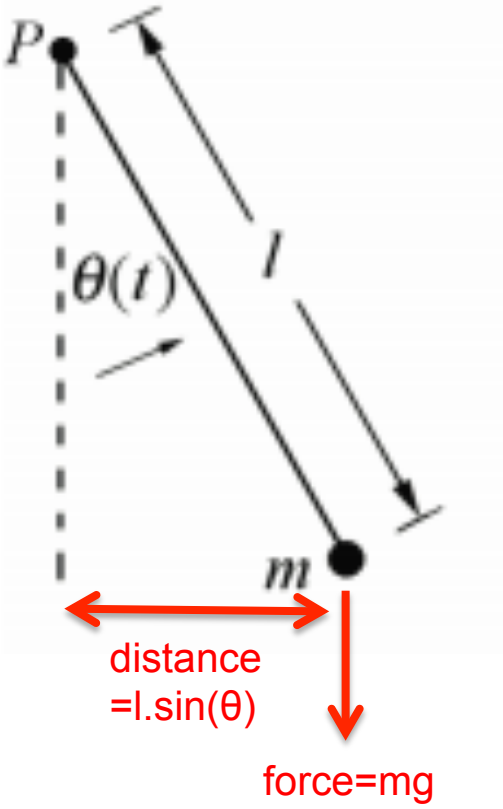
The eigenvalues λ are +ve and −ve real values,

This indicates that the dynamics are unstable.

For fixed point mass pendulum

The pendulum linear system matrix equation

$$\begin{bmatrix} \dot{\theta} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{g}{l} & 0 \end{bmatrix} \begin{bmatrix} \theta \\ \omega \end{bmatrix}$$



Implies that the linearized equation of motion for point mass pendulum is

$$\frac{d^2\theta}{dt^2} = -\frac{g}{l}\theta$$

We can also take Laplace transforms with zero initial condition

$$\Rightarrow s^2\Phi(s) = -\frac{g}{l}\Phi(s) \qquad \Rightarrow s = \pm j\sqrt{\frac{g}{l}}$$

To get angular frequency substitute $s = j\omega$

$$\Rightarrow \omega = \pm\sqrt{\frac{g}{l}} \qquad \Rightarrow f = \frac{1}{2\pi}\sqrt{\frac{g}{l}}$$

Where f is the frequency of oscillation of the pendulum in Hz

ROCO218: Control Engineering
Dr Ian Howard

Lecture 6

2016R exam for ROCO319 Modern Control
Solutions to relevant questions

Q1: State space derivation and analysis

Consider the pendulum system shown in Figure (1) below, which can be used to model a one linked robot arm. The systems dynamics is described by equation (1) given below:

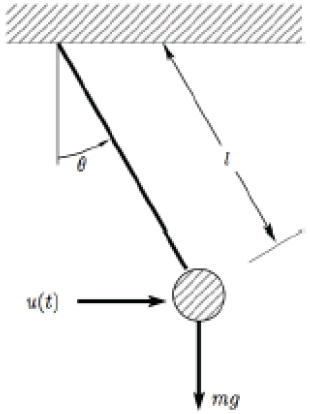


Figure 1: Pendulum.

$$ml\ddot{\theta} = -mg\sin\theta + u\cos\theta, \tag{1}$$

where the main dynamic variable is the angle, θ and the control is $u(t)$. The parameters are the length of the link, l, the mass attached to the link, m and the force of gravity g.

(a) Re-write equation (1) as a first order differential equation.

(5 marks)

(b) Linearise the obtained first order differential equation about small angle, $\theta \approx 0$, and small angular velocity, $\dot{\theta} \approx 0$.

(10 marks)

(c) Evaluate the eigenvalues of the uncontrolled system $(u = 0)$ about $(\theta \approx 0, \dot{\theta} \approx 0)$ and determine the stability of the system.

Q1: State space derivation and analysis

System dynamics are described by

$$ml\ddot{\theta} = -mg\sin\theta + u\cos\theta$$

To re-write as 1st order equations we first let

$$\omega = \dot{\theta}$$

$$\Rightarrow ml\dot{\omega} = -mg\sin\theta + u\cos\theta$$

$$\Rightarrow \dot{\omega} = -\frac{g}{l}\sin\theta + \frac{u}{ml}\cos\theta$$

Choosing state variables X

$$x_1 = \theta$$

$$x_2 = \omega \qquad \Rightarrow \dot{x}_1 = x_2$$

And also

$$\dot{\omega} = -\frac{g}{l}\sin\theta + \frac{u}{ml}\cos\theta \qquad \Rightarrow \dot{x}_2 = -\frac{g}{l}\sin x_1 + \frac{u}{ml}\cos x_1$$

Q1: State space derivation and analysis

We linearize using the Jacobian, we first need to find the equilibrium
Equilibrium in the absence of input u occurs when

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

$$\dot{x}_2 = -\frac{g}{l}\sin x_1 + \frac{u}{ml}\cos x_1 = 0$$

$$\Rightarrow x_1 = 0, \pm\pi, \pm 2\pi, \ldots$$ Here we only consider the first solution $x_1 = x_2 = 0$

Writing the state equations as the functions

$$f_1 = \dot{x}_1 = x_2 \qquad f_2 = \dot{x}_2 = -\frac{g}{l}\sin x_1 + \frac{u}{ml}\cos x_1$$

The Jacobian matrices for system and control are given by

$$J_f(x,y) = \begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} \end{bmatrix}$$

$$J_u(x,y) = \begin{bmatrix} \frac{\partial f_1}{\partial u} \\ \frac{\partial f_2}{\partial u} \end{bmatrix}$$

Q1: State space derivation and analysis

Given the functions

$$f_1 = x_2$$

$$f_2 = -\frac{g}{l}\sin x_1 + \frac{u}{ml}\cos x_1$$

Differentiating w.r.t. to state variables

$$\Rightarrow \frac{\partial f_1}{\partial x_1} = 0 \qquad \Rightarrow \frac{\partial f_1}{\partial x_2} = 1$$

$$\Rightarrow \frac{\partial f_2}{\partial x_1} = -\frac{g}{l}\cos x_1 - \frac{u}{ml}\sin x_1 \qquad \Rightarrow \frac{\partial f_2}{\partial x_2} = 0$$

Substituting in terms, the Jacobian system matrix is given by

$$\Rightarrow J_f(x,y)\big|(0,0) = \begin{bmatrix} 0 & 1 \\ -\frac{g}{l}\cos x_1 - \frac{u}{ml}\sin x_1 & 0 \end{bmatrix}\Big|(0,0) = \begin{bmatrix} 0 & 1 \\ -\frac{g}{l} & 0 \end{bmatrix}$$

Q1: State space derivation and analysis

Given the functions

$$f_1 = x_2$$

$$f_2 = -\frac{g}{l}\sin x_1 + \frac{u}{ml}\cos x_1$$

Differentiating w.r.t. to control input

$$\Rightarrow \frac{\partial f_1}{\partial u} = 0 \qquad \Rightarrow \frac{\partial f_2}{\partial u} = \frac{1}{ml}\cos x_1$$

Substituting in terms, the Jacobian control matrix is given by

$$\Rightarrow J_u(x,y)\big|(0,0) = \begin{bmatrix} 0 \\ \frac{1}{ml}\cos x_1 \end{bmatrix}\Big|(0,0) \qquad = \begin{bmatrix} 0 \\ \frac{1}{ml} \end{bmatrix}$$

Q1: State space derivation and analysis

The state space equation is therefore

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{g}{l} & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{ml} \end{bmatrix} u$$

The eigenvalues of the uncontrolled system are determined by the equation

$$\det(J - \lambda I) = 0$$

$$\Rightarrow \left| \begin{bmatrix} 0 & 1 \\ -\frac{g}{l} & 0 \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right| = 0$$

$$\Rightarrow \left| \begin{bmatrix} -\lambda & 1 \\ -\frac{g}{l} & -\lambda \end{bmatrix} \right| = 0$$

Q1: State space derivation and analysis

The expression

$$\left\| \begin{bmatrix} -\lambda & 1 \\ -\frac{g}{l} & -\lambda \end{bmatrix} \right\| = 0$$

Leads to the characteristic equation

$$\Rightarrow (-\lambda)(-\lambda) - (-\frac{g}{l}) = 0 \qquad \Rightarrow \lambda^2 + \frac{g}{l} = 0$$

$$\Rightarrow \lambda^2 = -\frac{g}{l}$$

$$\Rightarrow \lambda = \pm j\sqrt{\frac{g}{l}}$$

It can be seen that the eigenvalues are complex conjugate
therefore there are oscillatory dynamics and system is neutrally stable