Observation Report Modelling and control

Introduction

Here in this task we learnt the basics of mathematical modelling and different control algorithms like-

- Pole placement
- LQR controller

We worked on the models like-

- Simple pendulum
- Spring mass system
- simple and complex pulley
- Inverted cart pendulum

1 Mathematical modeling

We have a model having state variable **X** and input **U** as -

$$X = \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ \cdot \\ X_n \end{bmatrix}, U = \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ \cdot \\ U_m \end{bmatrix}$$

We will deal with dynamical systems that are modelled by a finite number of coupled first order ordinary differential equations

$$\dot{X} = \begin{bmatrix} f_1(t, X_1, X_2, \cdots, X_n, U_1, U_2, \cdots, U_m) \\ f_2(t, X_1, X_2, \cdots, X_n, U_1, U_2, \cdots, U_m) \\ f_3(t, X_1, X_2, \cdots, X_n, U_1, U_2, \cdots, U_m) \\ \cdot \\ f_n(t, X_1, X_2, \cdots, X_n, U_1, U_2, \cdots, U_m) \end{bmatrix}$$

We call above equation as the **State Equation** of the system and refer to **X** as the state and **U** as the input.

1.1 Stability at Equilibrium Points

A typical problem that arises while dealing with non-linear dynamical systems is to check if a system is stable or unstable at a given equilibrium point.

Equilibrium Points of a system is the point at which the state of the system doesn't change. The equilibrium points can be estimated by setting $\dot{X} = 0$ and solving the resulting equations for **X**.

Stable Equilibrium If a system always returns to the equilibrium point after small perturbations.

Unstable Equilibrium If a system moves away from equilibrium point after small perturbations.

To simply check whether an equilibrium is stable or not, we first find the equilibrium point itself. followed by linearization of the model near the equilibrium point using *Jacobian* matrix.

By setting $\dot{X} = 0$ and solving the resulting equations for **X**, we got the **K** no. of equilibrium points as -

$$X := (x1, x2, \cdots xn)_K$$

The Jacobian J for the set of equations can be calculated as-

$$J = \begin{bmatrix} \frac{\partial f_1}{\partial X_1} & \frac{\partial f_1}{\partial X_2} & \cdots & \frac{\partial f_1}{\partial X_n} \\ \frac{\partial f_2}{\partial X_1} & \frac{\partial f_2}{\partial X_2} & \cdots & \frac{\partial f_2}{\partial X_n} \\ \frac{\partial f_3}{\partial X_1} & \frac{\partial f_3}{\partial X_2} & \cdots & \frac{\partial f_3}{\partial X_n} \\ . & . & \cdots & . \\ \frac{\partial f_n}{\partial X_1} & \frac{\partial f_n}{\partial X_2} & \cdots & \frac{\partial f_n}{\partial X_n} \end{bmatrix}$$

Substitute the value of the equilibrium points in the matrix **J** to find **K** different matrices, each corresponding to an equilibrium point

$$J_1, J_2, \cdots, J_k$$

Now, to check the stability of any i^{th} equilibrium point, we find the *Eigen values* of J_i matrix by constructing the characteristic equation of the matrix and equating it to zero -

$$\left| sI - J_i \right| = 0$$

where, I is the identity matrix and roots of **s** gives the *Eigen values.*

For each Equilibrium point

- If all the eigenvalues have **negative real part**, the system is **Stable** at the given Equilibrium point.
- If even one of the eigenvalues has **positive real part**, the system is **Unstable** at the given Equilibrium point.
- If there are multiple repeated eigenvalues the the imaginary axis the system is unstable otherwise said marginally stable.

1.2 State-space modelling

state-space representation is a mathematical model of a physical system as a set of input, output and state variables related by first-order differential equations or difference equations. **State variables** are variables whose values evolve through time in a way that depends on the values they have at any given time and also depends on the externally imposed values of input variables. **Output variables** values depend on the values of the state variables.

The state space equations for a **linear time invariant system (LTI)** system can be given as follows:

$$StateEquation => \dot{X} = A.X + B.U$$

$$OutputEquation => Y = C.X + D.U$$

here -
X - State Vector (n x 1 matrix)
Y - Output Vector (p x 1 matrix)
U - Input Vector (m x 1 matrix)
A - State (or system) matrix (n x n matrix)
B - Input matrix (n x m matrix)
C - Output Matrix (p x n matrix)
D - Feed-forward matrix (p x m matrix)

Here it is mentioned that the state equations are for linear systems. therefore when we are dealing with a non-linear system, we try to linearize the system around the vicinity of equilibrium point.

For our non-linear system, mentioned above, we try to find the state space equation for it's i^{th} equilibrium point. for linearization we will use *Jacobian*.

$$A = \begin{bmatrix} \frac{\partial f_1}{\partial X_1} & \frac{\partial f_1}{\partial X_2} & \cdots & \frac{\partial f_1}{\partial X_n} \\ \frac{\partial f_2}{\partial X_1} & \frac{\partial f_2}{\partial X_2} & \cdots & \frac{\partial f_2}{\partial X_n} \\ \frac{\partial f_3}{\partial X_1} & \frac{\partial f_3}{\partial X_2} & \cdots & \frac{\partial f_3}{\partial X_n} \\ \cdot & \cdot & \cdots & \cdot \\ \frac{\partial f_n}{\partial X_1} & \frac{\partial f_n}{\partial X_2} & \cdots & \frac{\partial f_n}{\partial X_n} \end{bmatrix}, B = \begin{bmatrix} \frac{\partial f_1}{\partial U_1} & \frac{\partial f_1}{\partial U_2} & \cdots & \frac{\partial f_1}{\partial U_m} \\ \frac{\partial f_2}{\partial U_1} & \frac{\partial f_2}{\partial U_2} & \cdots & \frac{\partial f_2}{\partial U_m} \\ \frac{\partial f_3}{\partial U_1} & \frac{\partial f_3}{\partial U_2} & \cdots & \frac{\partial f_3}{\partial U_m} \\ \cdot & \cdot & \cdots & \cdot \\ \frac{\partial f_n}{\partial U_1} & \frac{\partial f_n}{\partial U_2} & \cdots & \frac{\partial f_n}{\partial U_m} \end{bmatrix}$$

For i^{ih} equilibrium point, we calculate the value of A_i and B_i by putting the point in the above equation. and the State equation will be given by -

$$\dot{X} = A_i.X + B_i.U$$

Note that this approximation of the set of non-linear equations given in above equation will only hold true for point close to the i^{th} equilibrium point i.e.,

$$X := (x1, x2, \cdots xn)_i$$

This means that around the vicinity of the i^{th} equilibrium point, the non-linear system will behave like a linear system and the state equation given above will hold true around the vicinity of that point. Likewise, the state equations for equilibrium points.

1.3 Controllability and Observability

In control theory, controllability and observability are two very important properties of the system.

Controllability is the ability to drive a state from any initial value to a final value in finite amount of time by providing a suitable input. A matrix which determines if a system is fully controllable or not is called the controllability matrix.

Observability is the property of the system that for any possible sequence of state and control inputs, the current state can be determined in finite time using only the outputs. A matrix which determines if a system is fully observable or not is called the observability matrix. A fully observable system means that it is possible to know all the state variables from the system outputs.

Controllability matrix (R) of a system is given by the following:

$$B = \begin{bmatrix} B & AB & A^2B & \cdots & A^{n-1}B \end{bmatrix}$$

Observability matrix (O) of a system is given by the following:

$$B = \begin{bmatrix} C \\ CA \\ CA^2 \\ . \\ CA^{n-1} \end{bmatrix}$$

If a system is fully controllable, then

$$rank(R) = n$$

If a system is fully observable, than

$$rank(O) = n$$

Rank of a matrix is defines as the maximum number of linearly independent rows or columns in a matrix.

2 Controls

So far we have discussed the basics of state space analysis. In this section, we will discuss different types of controller design. Consider the State Space Equations of a system:

$$\dot{X} = A.X + B.U$$

$$Y = C.X + D.U$$

We feed the input of the system **U** as:

$$U = rK_r - KX$$

the State Equation for the system can be written as follows:

$$\dot{X} = AX + B(rK_r - KX)$$

$$=> \dot{x} = (A - BK)X + B_r K_r$$

The new state matrix (A-BK) defines the dynamics of the system where $-KX$is feed as input and is known as state feedback system. The system stability can be calculated by finding the eigenvalues of the (A-BK) matrix.

2.1 Pole Placements Method

One method to ensure that the system is stable is to select the gain matrix K in such a way so that the eigenvalues of the (A-BK) matrix has all the eigenvalues with negative real part. Placing all the eigenvalues to desired location is only possible when the system is fully controllable. We can select the desired eigenvalues for the system and calculate the K matrix such that (A-BK) has our desired eigenvalues.

The characteristic equation for this system will be:

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

We put the eigenvalues to calculate the gain matrix K. then we simply feed the input -KX.
Note that, we chose the eigenvalues such that system become stable abound the set-point.

2.2 Linear Quadratic Regulator (LQR)

So far we have seen that if we have a system which is controllable, then we can place its eigenvalues anywhere in the left half plane by choosing appropriate gain matrix K. But the main question is where should we place our eigenvalues? Till now we have only discussed about the stability of the system. But nowhere have we asked that what is our performance measure? In this section, we'll see how to optimise the value of gain matrix K to get the desired performance measure from the system.

Linear Quadratic Regulator is a powerful tool which helps us choose the K matrix according to our desired response.Here we use a cost function

$$J = \int_0^\infty (X^T QX + U^T RU)dt$$

where Q and R are positive semi-definite diagonal matrices (positive semi-definite matrices are those matrices whose all the eigenvalues are greater than or equal to zero). Also to remind you that for a diagonal matrix, the diagonal entries are its eigenvalues.
X and U are the state vector and input vector respectively.

Let us say that you have your system with

$$X = \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ . \\ X_n \end{bmatrix}, U = \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ . \\ U_m \end{bmatrix}$$

Let,

$$Q = \begin{bmatrix} Q_1 & 0 & 0 & \cdots & 0 \\ 0 & Q_2 & 0 & \cdots & 0 \\ 0 & 0 & Q_3 & \cdots & 0 \\ . & . & . & \cdots & . \\ 0 & 0 & 0 & \cdots & Q_n \end{bmatrix}, R = \begin{bmatrix} R_1 & 0 & 0 & \cdots & 0 \\ 0 & R_2 & 0 & \cdots & 0 \\ 0 & 0 & R_3 & \cdots & 0 \\ . & . & . & \cdots & . \\ 0 & 0 & 0 & \cdots & R_n \end{bmatrix}$$

Then,

$$X^T QX + u^T RU = Q_1 X_1^2 + Q_2 X_2^2 + \cdots + R_n X_n^2 + R_1 U_1^2 + R_2 U_2^2 + \cdots + R_n U_n^2$$

Thus, we may see that the system taken here is our usual system represented as

$$\dot{X} = A.X + B.U$$

$$Y = C.X + D.U$$

The controller is of form $U = -KX$ which is a Linear controller and the underlying cost function is Quadratic in nature and hence the name Linear Quadratic Regulator. With a careful look at the integrand of the cost function J, we may observe that each Q_i are the weights for the respective states X_i.

So, the trick is to choose weights Q_i for each state X_i so that the desired performance criteria is achieved. Greater the state objective is, greater will be the value of Q corresponding to the said state variable.We can choose R = 1 for single input system. In case we have multiple inputs, we could use similar arguments for weighing the inputs as well.

LQR minimizes this cost function J based on the chosen matrices Q and R. Its a bit complicated to find out matrix K which minimizes this cost function.This is usually done by solving Algebraic Riccatti Equation (ARE). There is inbuilt **lqr()** command in octave to find K matrix. What is required to be done is to choose the Q and R matrix appropriately to get the desired performance.

3 Systems under test

3.1 Simple Pendulum

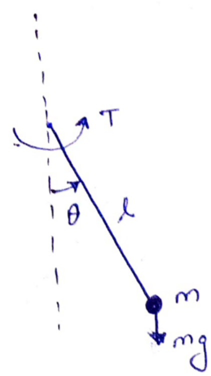

Fig. 1 - Simple pendulum
(with external torque)

State variables -

$$Y = \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix}, \dot{Y} = \begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \end{bmatrix}, U = T$$

State equation of the system-

$$\dot{Y}[1] = \dot{\theta}$$

$$\dot{Y}[2] = \frac{-g.sin(\theta)}{L} + \frac{T}{mL^2}$$

We linearize the system at $Y := (\pi, 0)$, i.e., at the top position.
A, B matrices for State space equation are -

$$A = \begin{bmatrix} 0 & 1 \\ \frac{g}{L} & 0 \end{bmatrix}, B = \begin{bmatrix} 0 \\ \frac{1}{mL^2} \end{bmatrix}$$

3.2 Mass spring system

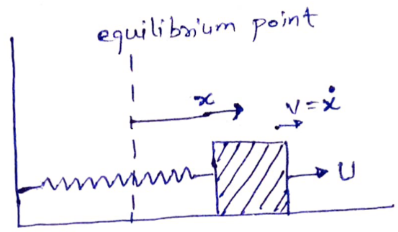

Fig-2 - Mass spring System

State variables -

$$Y = \begin{bmatrix} x \\ \dot{x} \end{bmatrix}, \dot{Y} = \begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix}, U = F$$

State equation of the system-

$$\dot{Y}[1] = \dot{x}$$

$$\dot{Y}[2] = \frac{U}{m} - \frac{kx}{m}$$

We linearize the system at $Y := (0, 0)$
A, B matrices for State space equation are -

$$A = \begin{bmatrix} 0 & 1 \\ \frac{-k}{m} & 0 \end{bmatrix}, B = \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix}$$

3.3 Simple Pulley

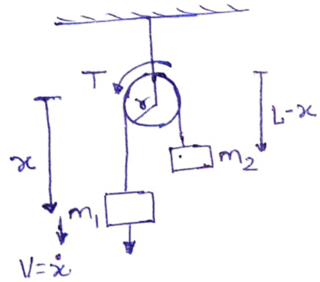

Fig-3- Simple pulley system

State variables -

$$Y = \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix}, \dot{Y} = \begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \end{bmatrix}, U = T$$

State equation of the system-

$$\dot{Y}[1] = \dot{\theta}$$

$$\dot{Y}[2] = \frac{T}{r(m1 + m2)} - \frac{(m2 - m1)g}{m1_m 2}$$

We linearize the system at $Y := (0.5, 0)$

A, B matrices for State space equation are -

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, B = \begin{bmatrix} 0 \\ \frac{1}{r(m1+m2)} \end{bmatrix}$$

3.4 Inverted cart pendulum

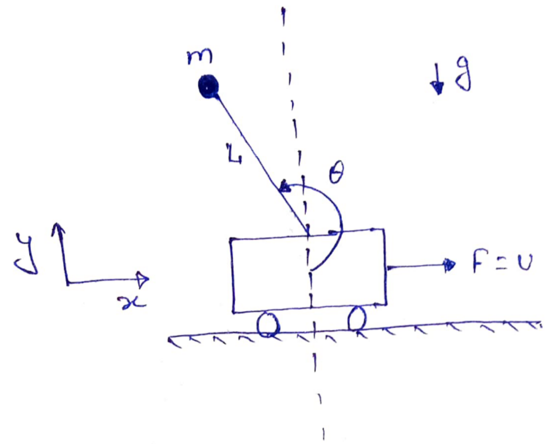

Fig-5 - inverted Cart pendulum

State variables -

$$Y = \begin{bmatrix} X \\ \dot{X}\theta \\ \dot{\theta} \end{bmatrix}, \dot{Y} = \begin{bmatrix} \dot{X} \\ \ddot{X} \\ \dot{\theta} \\ \ddot{\theta} \end{bmatrix}, U = F$$

State equation of the system-

$$\dot{Y}[1] = \dot{X}$$

$$\dot{Y}[2] = \frac{u + m.L.sin(\theta)(\dot{\theta})^2 + m.g.sin(\theta)cos(\theta)}{M + m.sin(\theta)}$$

$$\dot{Y}[3] = \dot{\theta}$$

$$\dot{Y}[4] = \frac{-g.sin(\theta)}{L} - \frac{cos(\theta).\dot{Y}[2]}{L}$$

The derivation is much similar to- click here

We linearize the system at $Y := (0, 0, \pi, 0)$

A, B matrices for State space equation are -

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{g.m}{M} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & \frac{g}{L}\left(\frac{m}{M} + 1\right) & 0 \end{bmatrix}, B = \begin{bmatrix} 0 \\ \frac{1}{M} \\ 0 \\ \frac{1}{ML} \end{bmatrix}$$

To view the simulation- click here, Source code- click here