BUDAPEST UNIVERSITY OF TECHNOLOGY AND ECONOMY
FACULTY OF MECHANICAL ENGINEERING

DEPARTMENT OF DEPARTMENT OF MECHATRONICS, OPTICS AND MECHANICAL INFORMATICS ENGINEERING

ADVANCE CONTROL AND INFORMATICS

HOMEWORK

(Sate feedback and observer design for inverted pendulum*)*

By: **Bhat Ishfaq Ahmad**
Neptun: **DS1FPA**

Task 1: The nonlinear model.

For imposing our system to Simulink, first we need to convert the given matrix equation into state space form by expanding the matrix equations to simple differential equation.

$$(Js + l^2m)\ddot{\theta} + ml\,cos\theta\,\ddot{x} - mlg\,sin\theta = 0$$

$$ml\,cos\theta + (M+m)\ddot{x} + b\dot{x} - m\,l\,sin\dot{\theta}^2 sin\,\theta = u$$

Solving the system with respect to $\ddot{x}$ and $\ddot{\theta}$:

$$\ddot{x} = \frac{-(J_s + l^2m)\cdot(b\dot{x} - u) + \mathrm{ml}\left(\dot{\theta}^2(Js + l^2m) - g\mathrm{lm}\cdot\cos(\theta)\right)\cdot\sin(\theta)}{(Js + l^2m)\cdot(m+M) - l^2m^2\cos(\theta)^2}$$

$$\ddot{\theta} = \frac{\mathrm{ml}\left(g(m+M)\sin(\theta) - \cos(\theta)\left(-b\dot{x} + u + \dot{\theta}^2\mathrm{lm}\,\sin(\theta)\right)\right)}{(Js + l^2m)\cdot(m+M) - l^2m^2\cos(\theta)^2}$$

After getting the equations for $\ddot{x}$ and $\ddot{\theta}$, we can change the variables to the space states:

$$x = x_1$$
$$\dot{x} = x_2$$
$$\theta = x_3$$
$$\dot{\theta} = x_4$$

After implementing these equations in the Simulink:

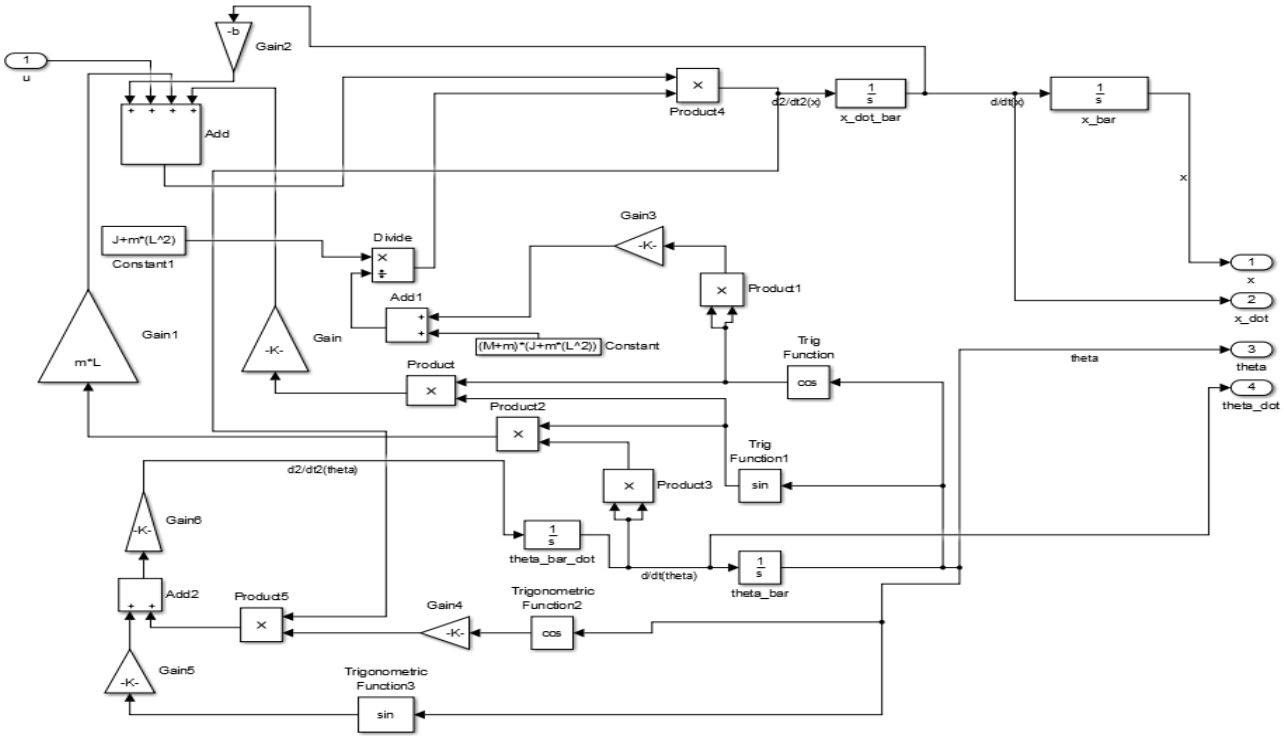


Fig1. Nonlinear Simulink model

Task 2: Linear model and comparing it with analytical linearization.

As shown in figure 2 below, the linearized model has been calculated. Now to derive the analytical linearization, we use the linearized equation of motion:

$$\begin{bmatrix} J_s + ml^2 & ml \\ ml & M+m \end{bmatrix}\begin{bmatrix} \ddot{\theta} \\ \ddot{x} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & b \end{bmatrix}\begin{bmatrix} \dot{\theta} \\ \dot{x} \end{bmatrix} + \begin{bmatrix} -mgl & 0 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} \theta \\ x \end{bmatrix} = \begin{bmatrix} 0 \\ u \end{bmatrix}$$

The state-space representation of multiple inputs linear times invariant systems with nth-order state differential equation can be written as is given by the equations:

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

Where,

$$\boldsymbol{x} = \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix}$$

Therefore,

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{b(J_s+ml^2)}{l^2m^2-(M+m)R} & \frac{gl^2m^2}{l^2m^2-(M+m)R} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{-bml}{l^2m^2-(M+m)R} & -\frac{gml(m+M)}{l^2m^2-(M+m)R} & 0 \end{bmatrix}; \qquad \boldsymbol{B} = \begin{bmatrix} 0 \\ -\frac{J_s+ml^2}{Kl^2m^2-(M+m)R} \\ 0 \\ \frac{\text{ml}}{l^2m^2-(M+m)R} \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \qquad \boldsymbol{D} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Now we have the state space from analytical solution.

The state space matrix of the analytical model:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -0.0208 & -1.2556 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0.0079 & 4.1915 & 0 \end{bmatrix}; \qquad \boldsymbol{B} = \begin{bmatrix} 0 \\ 0.1041 \\ 0 \\ -0.0394 \end{bmatrix}$$

C and D matrix are the same

From the MATLAB, we get the space state matrix of the linearized model as:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -0.0208 & -1.2556 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0.0079 & 4.1915 & 0 \end{bmatrix} \qquad \boldsymbol{B} = \begin{bmatrix} 0 \\ 0.1041 \\ 0 \\ -0.0394 \end{bmatrix}$$

If we compare the matrices of the analytical equation and the linearized model, we see that the numerical value of our analytical equation's matrices and the linearized model matrices are the same.

After implementing both in Simulink and getting the results from the systems, we see that they are approximately the same.

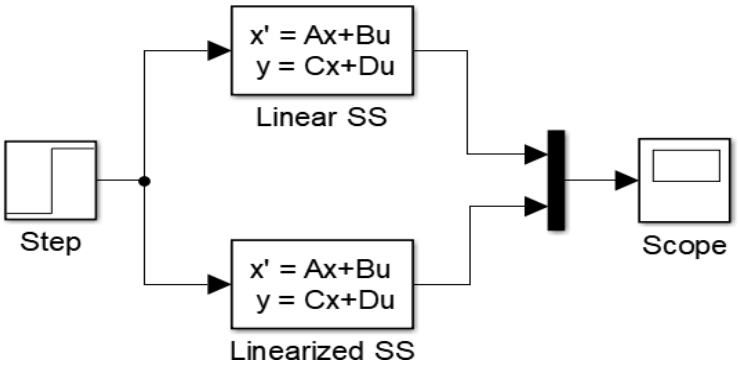


Fig2. Linearized Model

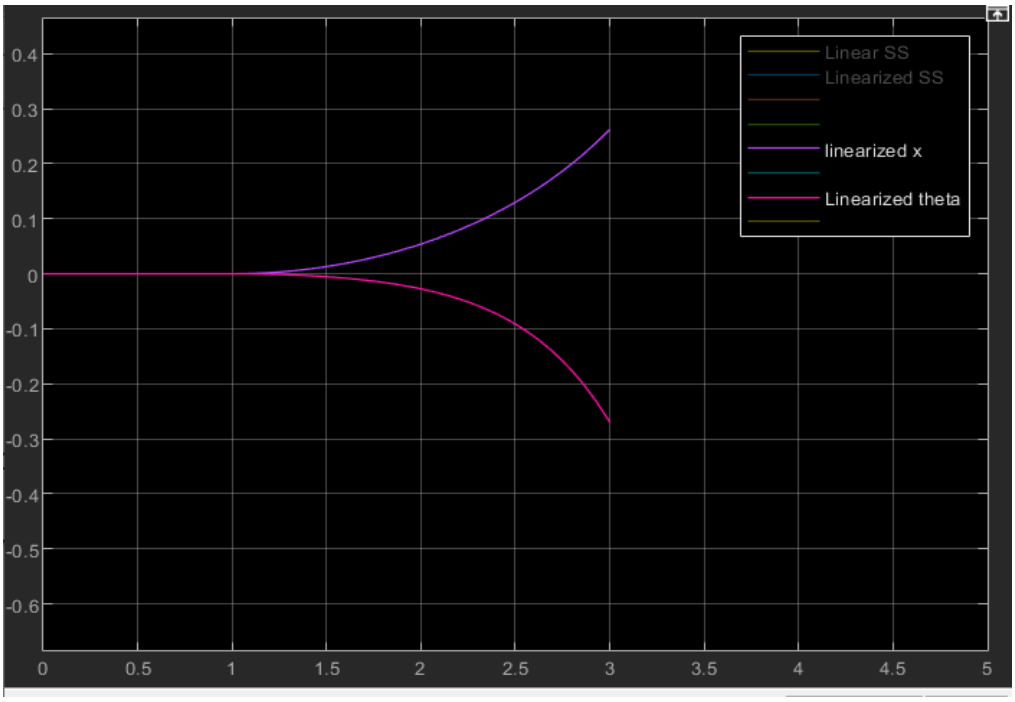


Fig3. Graph showing the two systems

Task 2: State feedback controller with pole placement.

The aim of designing a state feedback controller is to stabilize an unstable system, or to improve its performance in terms of stability so that transients die very fast.
By selecting a proper gain matrix for state feedback, it is easy to assign the closed-loop poles of the system at the desired locations in the complex plane, it is possible when the system is completely state controllable. For the selection of desired closed-loop poles in the complex plane there is a compromise need to be done between response of error vector in terms of rapidity and the responsiveness to perturbations and measurement noises. If there is an increase in speed of error response is shown, then there will be an increase in harmful effects of perturbations and measurement noises will be shown.
For this task, we have to design the close loop system so that the Eigen values can be anywhere. So, we have to use different poles to get the value of K which will be the feedback gain of the system to stabilize the system. We have used different poles to see the effect of the poles on the stability of the system. These poles are the vectors of the desired Eigen values.
The closed loop system, which is formed by feeding back the state variables through a real constant matrix:

$$u = -Kx$$

Simulink Model:

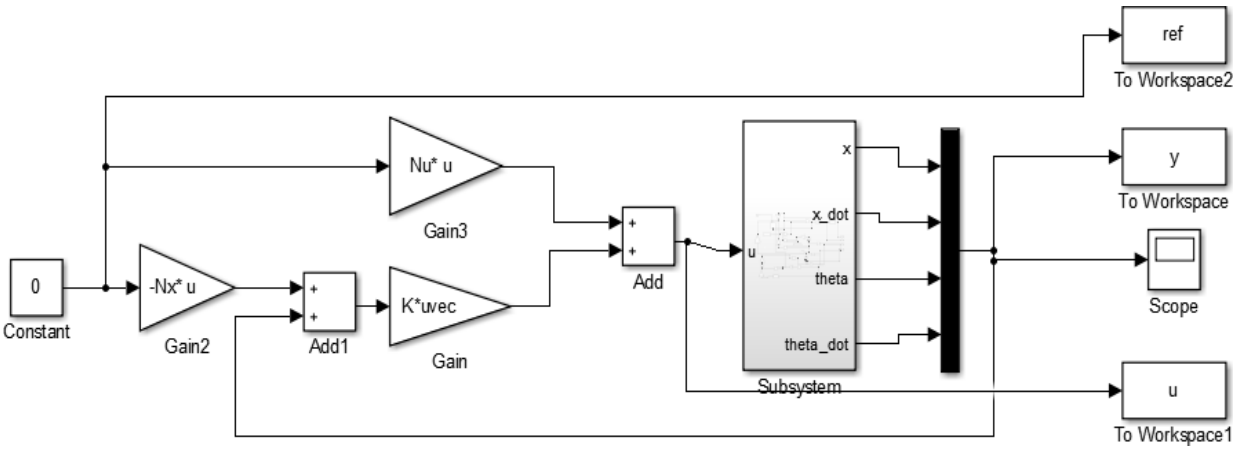


Fig 4. Nonlinear pole placement mode

Also, we need to calculate *N*x and *N*u which are there in our code:

$$\boldsymbol{K} = \begin{bmatrix} -4.4379 \\ -15.7843 \\ -318.7363 \\ -157.8464 \end{bmatrix}$$

Also

$$\boldsymbol{N_x} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}; \qquad \boldsymbol{N_u} = \mathbf{0}$$

After running the simulation, we got the graph for the pole given in the task. As we can see in the below graph, the system goes to steady state after 10 secs approximately. If we will change our poles, then it will affect the stability of the system.

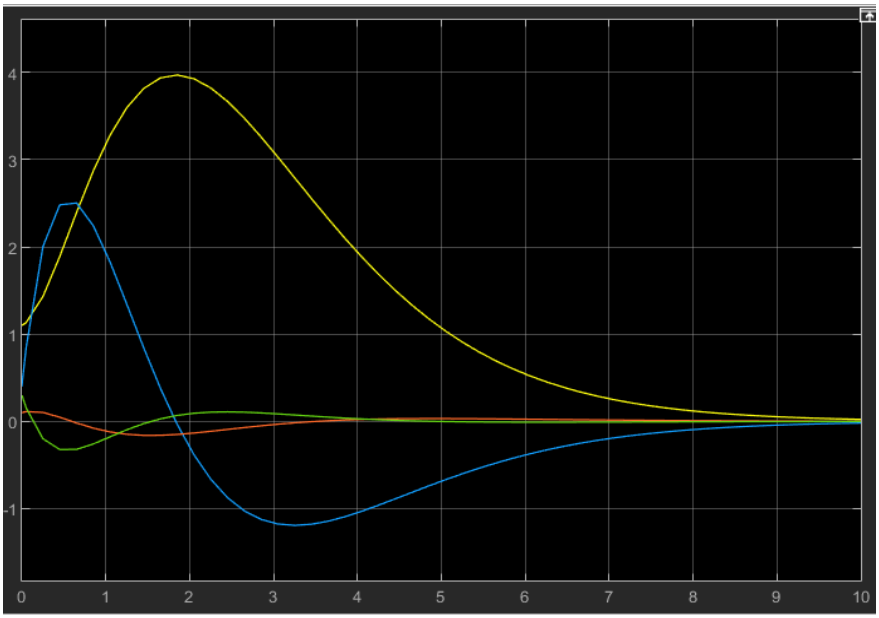


Fig 5. Pole placement graph

Now, if we increase our poles i.e. set them to $p = [-3.7;\ -4.0;\ -4.2;\ -4.6]$, we find that these are the last set of poles for which our system is still stable, as shown in figure 6. We see that the system goes to the steady state, but the variation of oscillations is way too much large than the previous one.

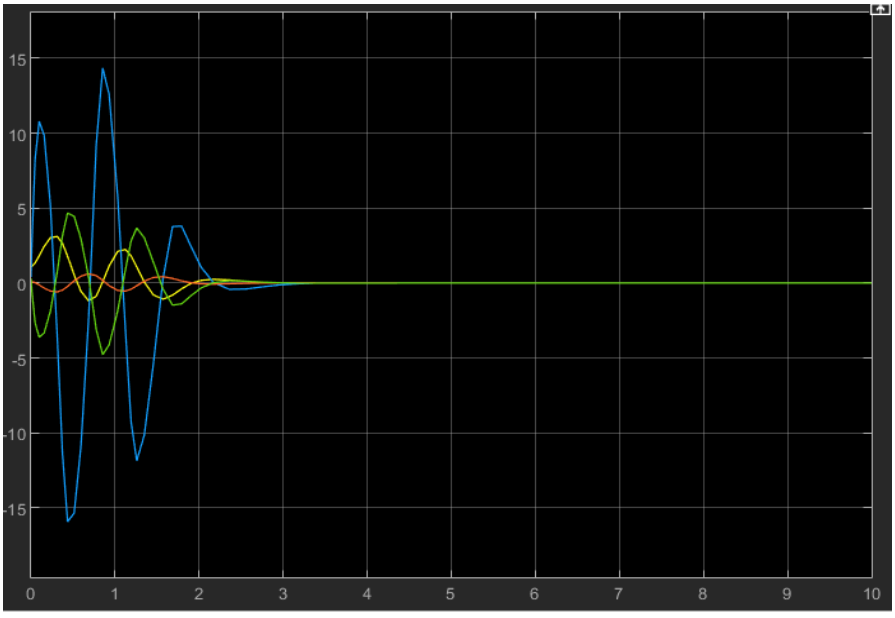


Fig 6. Pole placement graph

If the poles are set as $p = [-3.8;\ -4.0;\ -4.2;\ -4.6]$, we find out that our system breaks down and the controller does not have enough power to get our poles into steady state. The graph below shows the break down point for this pole placement.

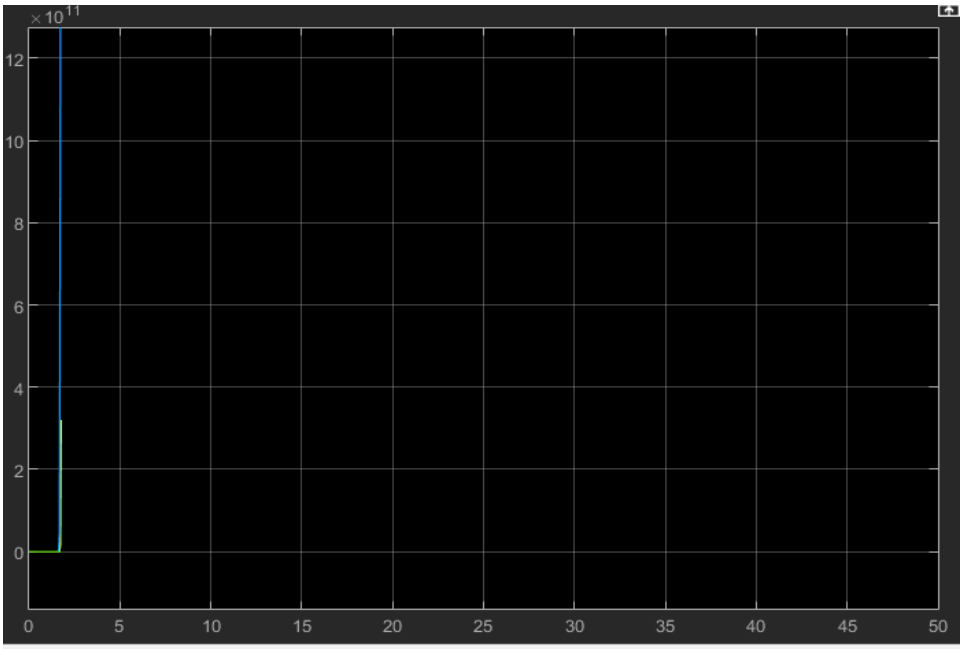


Fig 7. Pole placement graph

Task 4: LQR state feedback.

To get rid of some problems that are faced by conventional PID controller, the control strategy is used, which is Linear-Quadratic Regulator (LQR) optimal control. LQR is a control strategy which operates the system with minimum cost when the system dynamics is expressed by differential equations which are linear.
LQR is just an automatic method to find out an appropriate state. In this task, we had to design a LQR controller.
So first we have to calculate the gain of our LQR controller which for our case was calculated with our Q and R matrix listed as:

$$\boldsymbol{Q} = \begin{bmatrix} 100 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 100 \end{bmatrix};\quad \mathrm{R} = 1$$

Where Q and R are weight matrices, Q must be positive definite or positive semi-definite symmetric matrix. R must be positive definite symmetric matrix. Weighting matrices should be diagonal matrix. The value of the elements of the weighting matrices are linked to the impact on the performance index J. The feedback control law is:

$$u = -Kx$$

Implementing the model into Simulink:

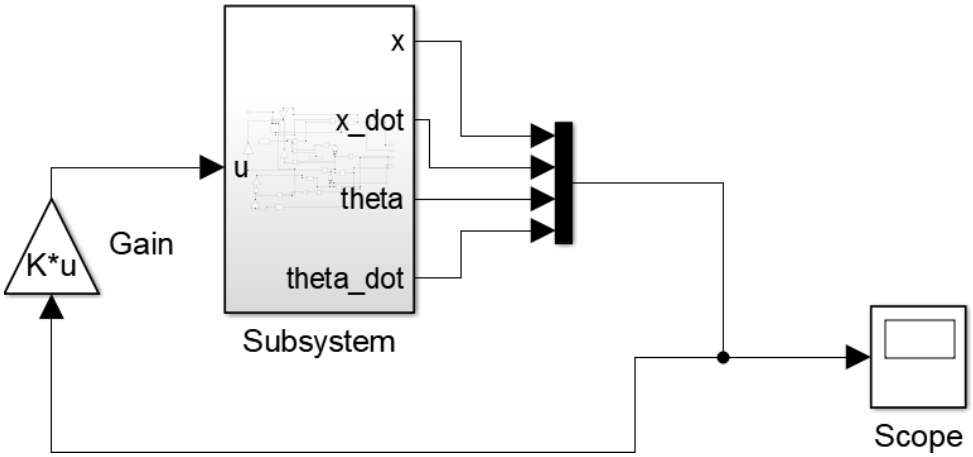


Fig8. LQR model

After running the simulation, we see that system (Figure 9) reaches to steady state after 7 secs approximately.

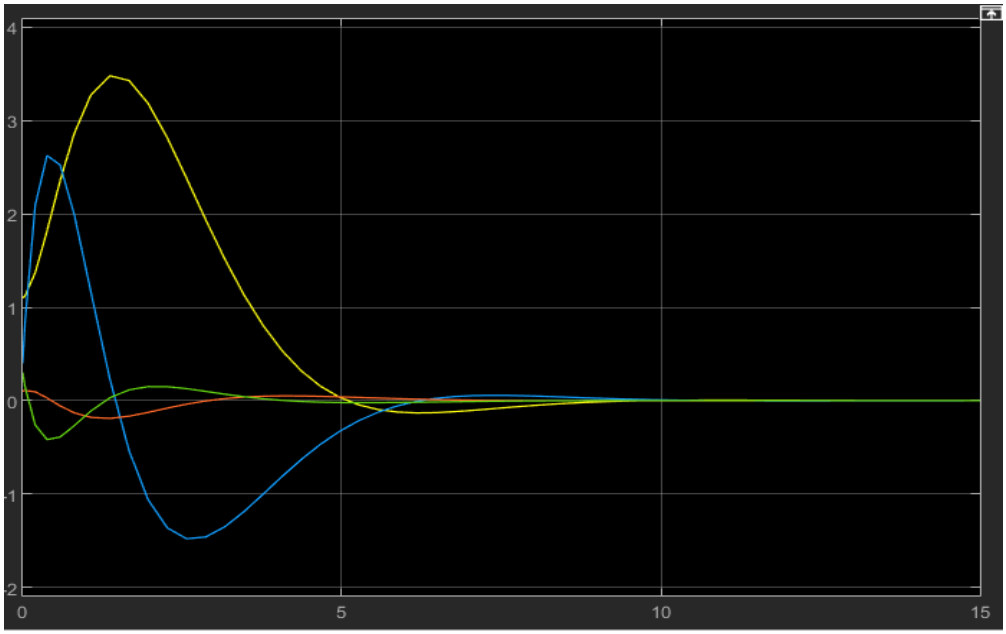
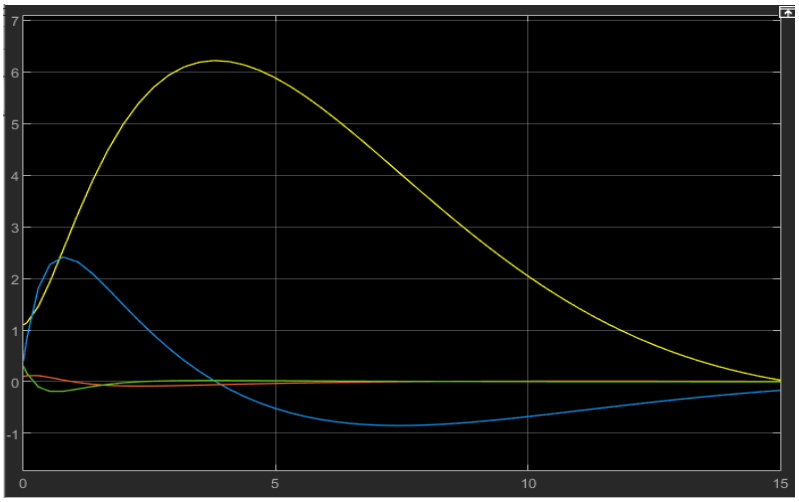


Fig 9. LQR model

Now we will analyze the system state by changing the values of Q and R matrices

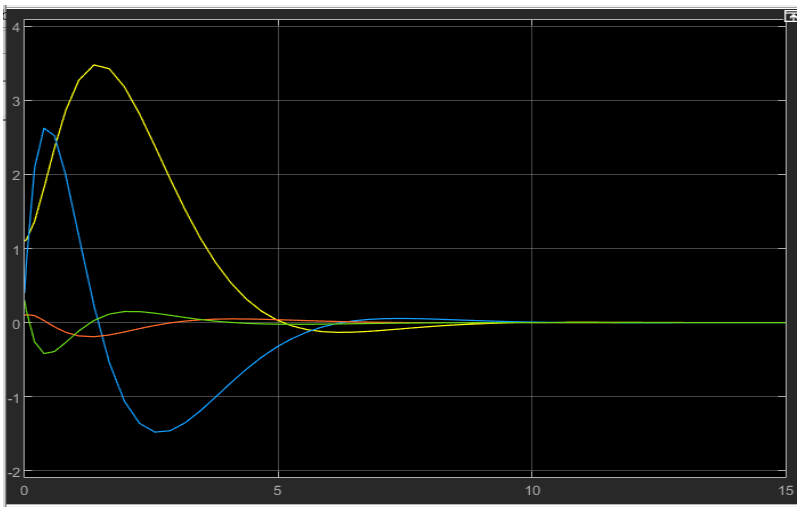
Case 1: Suppressing the Input state 1

$$\boldsymbol{Q} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 100 \end{bmatrix}; \qquad R = 1$$



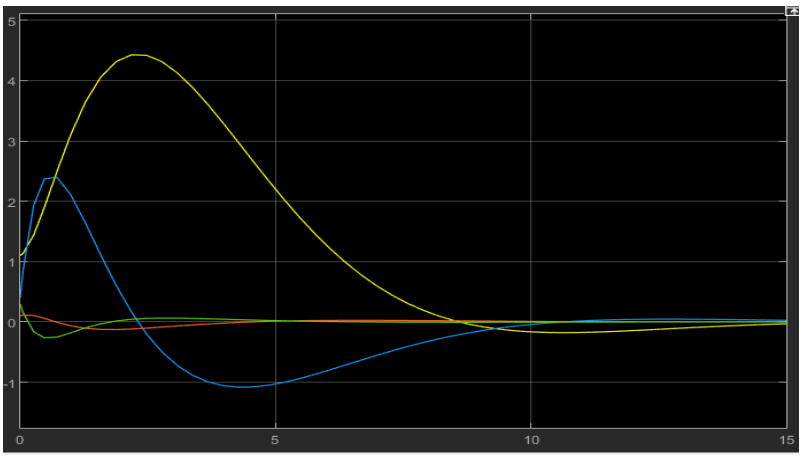
Case 2: Suppressing the Input state 3

$$\boldsymbol{Q} = \begin{bmatrix} 100 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 100 \end{bmatrix}; \qquad R = 1$$



Case 3: Suppressing the control signal

$$\boldsymbol{Q} = \begin{bmatrix} 100 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 100 \end{bmatrix}; \qquad R = 0.1$$



From all the cases, when the Control signal is suppressed, the system reaches to steady state at much faster rate and reaches to steady state in around 3 secs. As we know R represents the strength of the control signal. So, if we suppress the control signal, we can reach the steady state at much faster rate. If we compare the case 1 with the system response at given R and Q values, the system takes much more time to reach the steady state.

Task 5: Full state observer based on the measurement of *x*.

In this task we need to design an observer for the system. We have implemented Kalman filter to the Simulink model. We have used the same LQR controller as our controller so that the Q, R and K matrices are the same for this case.
The gain of Kalman filter:

$$\boldsymbol{K} = \begin{bmatrix} 9.5924 \\ 41.0068 \\ -84.1208 \\ -172.3311 \end{bmatrix}$$

Given:

$$\boldsymbol{V_d} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 10 \end{bmatrix}; \quad \boldsymbol{V_n} = \mathbf{1}$$

Implementing the Kalman filter in the model and simulate it , we get

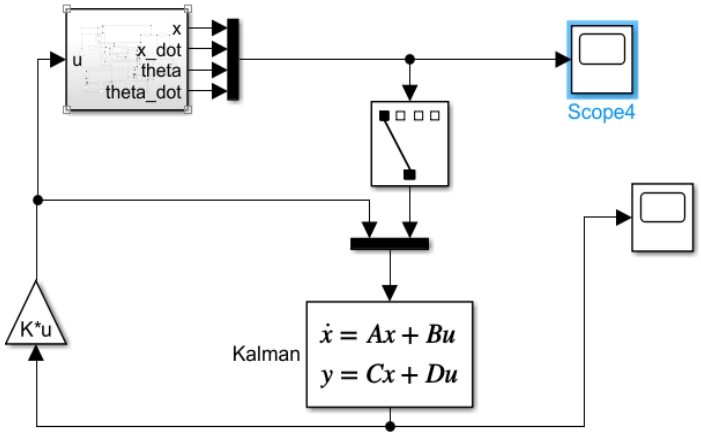


Fig 10. Kalman filter model

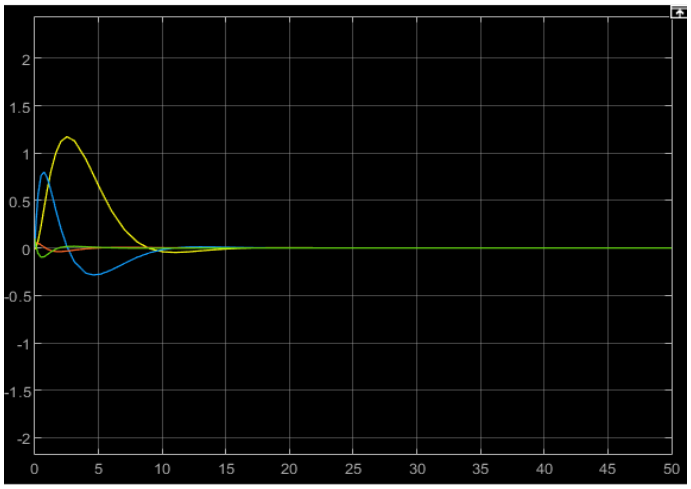


Fig 11. Kalman filter response

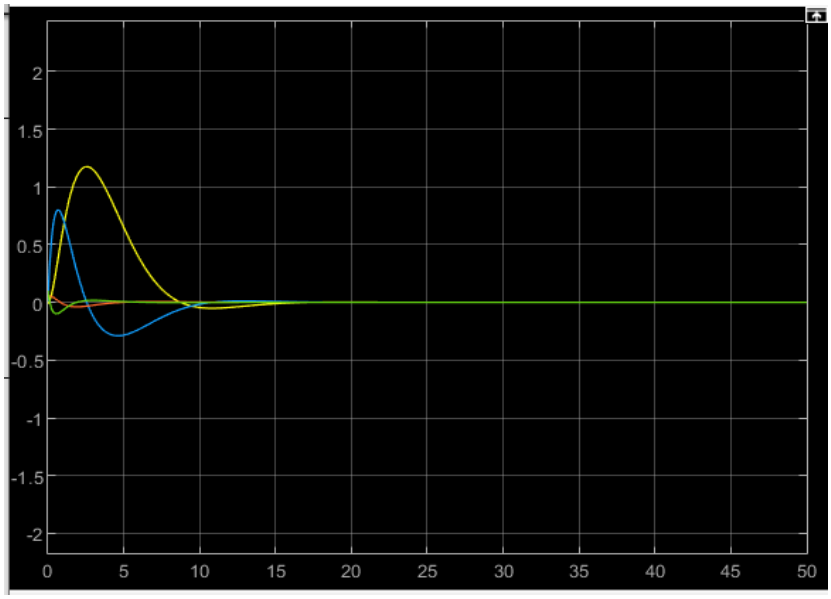
The initial conditions for the system are:

x_0_1 = -0.01; x_0_2 = 0.01; x_0_3 = 0.05; x_0_4 = 0.1;

After running the simulation, we can see that system reaches to steady state after 3 seconds.
We will now analyze the observer ability by changing the values of V_d and V_n.

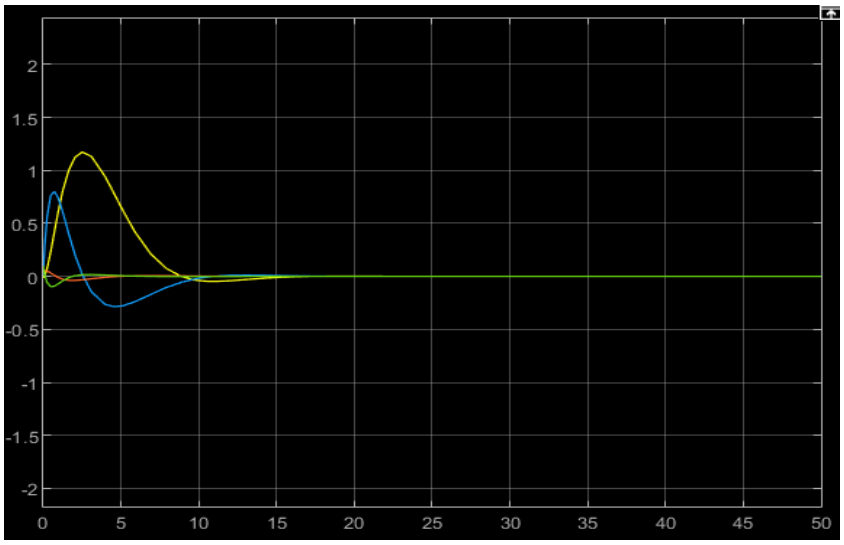
Case 1: Changing V_d matrix

$$\boldsymbol{V_d} = \begin{bmatrix} 100 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 10 \end{bmatrix}; \quad \boldsymbol{V_n = 0.1}$$



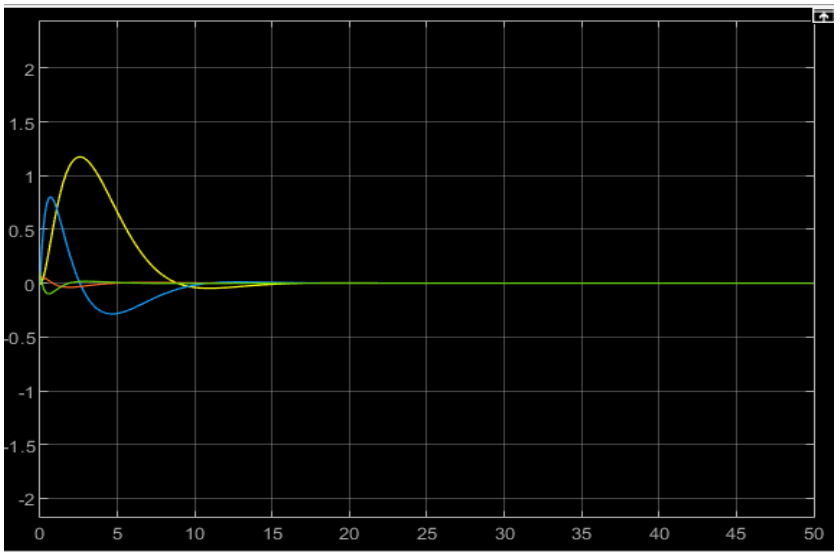
Case 2: Changing V_d matrix

$$\boldsymbol{V_d} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 10 \end{bmatrix}; \quad \boldsymbol{V_n = 0.1}$$



Case 3: Changing V_d matrix

$$\boldsymbol{V_d} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 10 \end{bmatrix}; \quad \boldsymbol{V_n = 0.001}$$



From all the cases, when the value of V_d is changed, the system reaches to steady state at much faster rate and reaches to steady state in around 3.5 secs.
If we change the value of V_n the Kalman filter follows the output more tightly and get more accurate results compared to other values.

Task 6: Trajectory tracking of the controlled system.

In this task we need to draw the trajectory of controlled system compared to input that we gave it. The Simulink model for trajectories is given below.

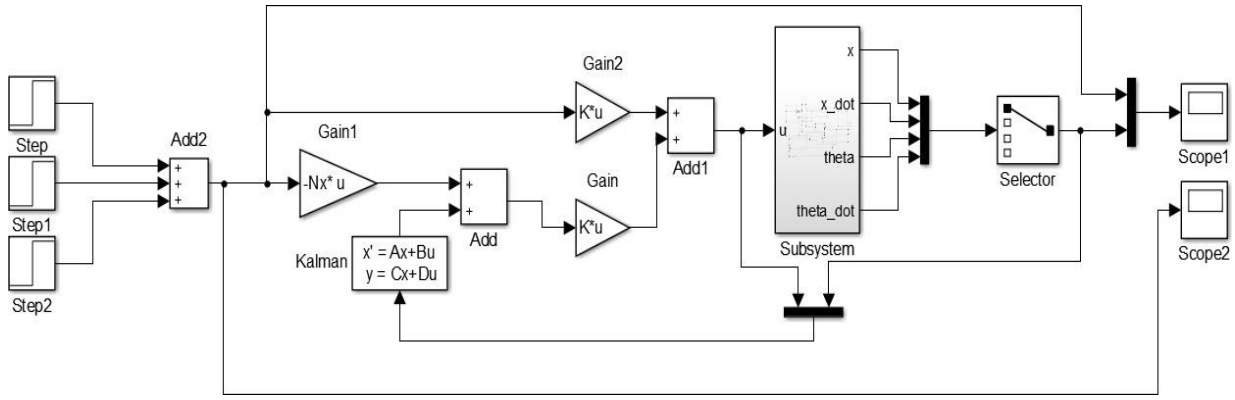
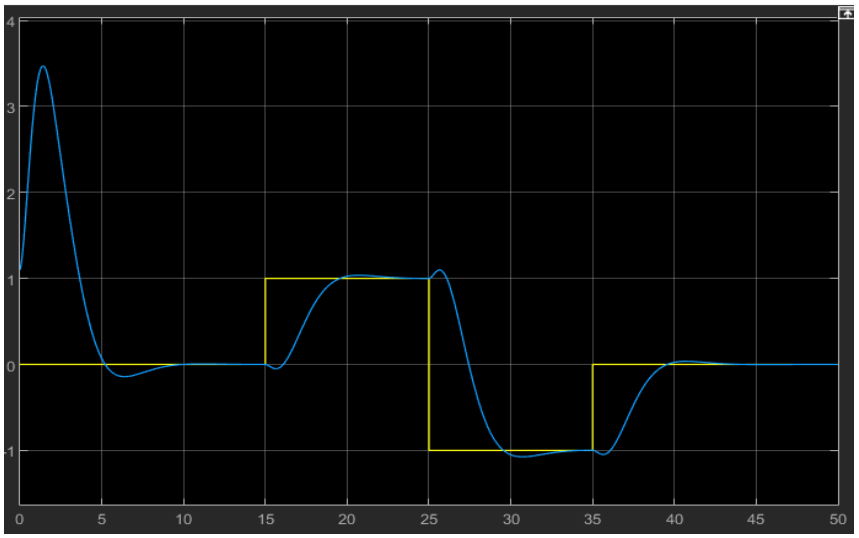


Fig12. Simulink model for trajectories

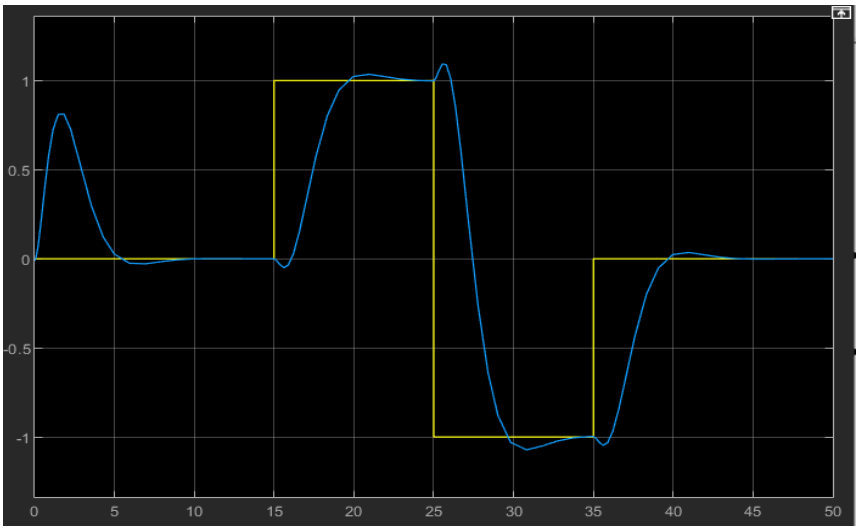
Case 1: For LQR filter

$$\boldsymbol{Q} = \begin{bmatrix} 100 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 100 \end{bmatrix}; \quad R = 1$$



Case 2: For Kalman filter

$$\boldsymbol{V_d} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 10 \end{bmatrix}; \quad \boldsymbol{V_n = 1}$$



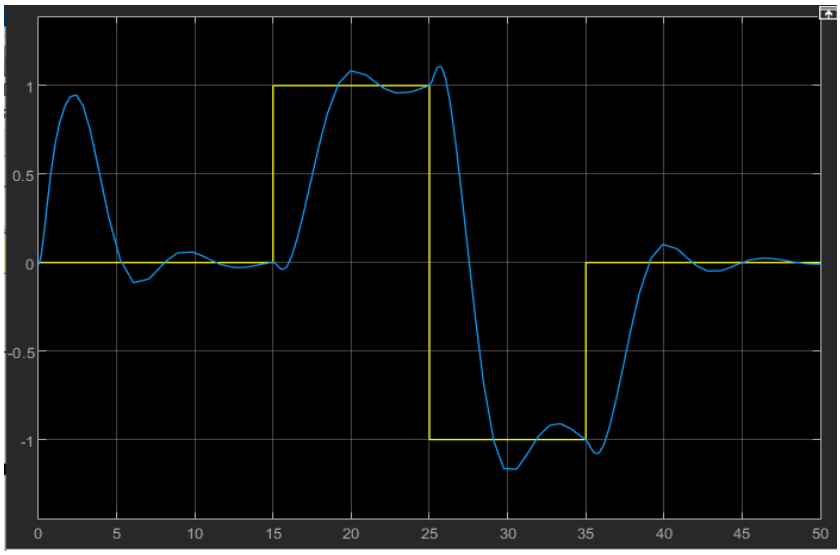
We can see the system follows the input but there are some delays in our system and also in the oscillations if we need to take these into control, we can do that by fine tuning our LQR gain and also Kalman filter gain and matrix.

Task 6: Robustness of final LQG controller.

In this task we need to check the robustness of our system with **M** = ±5% and l = ±5% We have analyzed different cases with different value of M and l. As we can see the control of the system is greatly affected by the mass of the cart that is M.

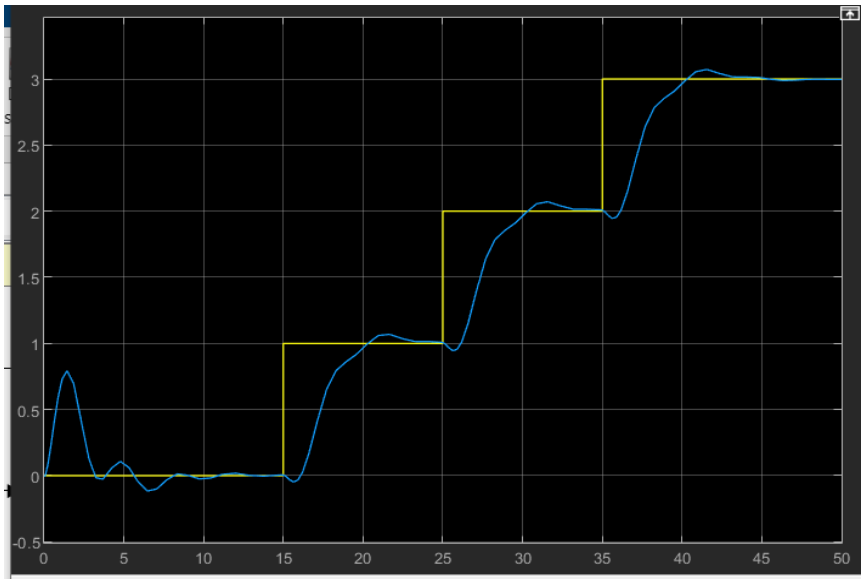
1. L= +5%, M= M (No change).

We can see that the controller is able to control the system. And the system goes to steady state after 50 secs.



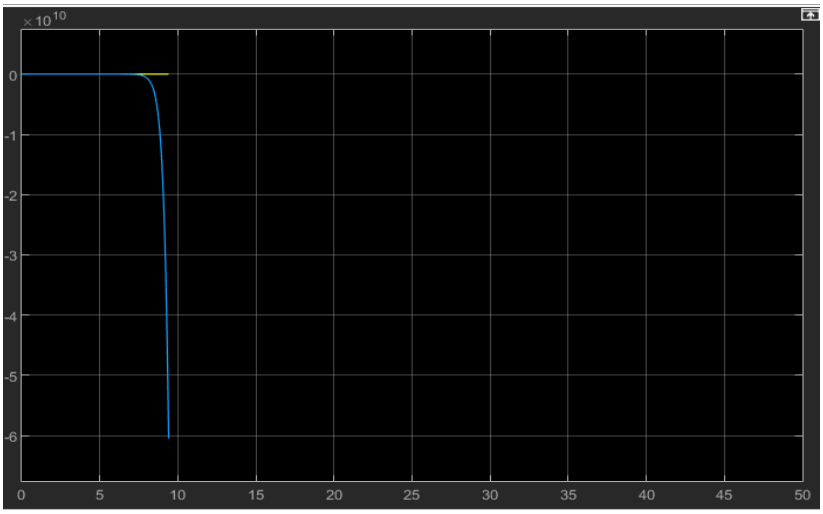
2. L= -5%, M= M (No change)

We can see that the controller is able to control the system. And the system goes to steady state after 35 secs.



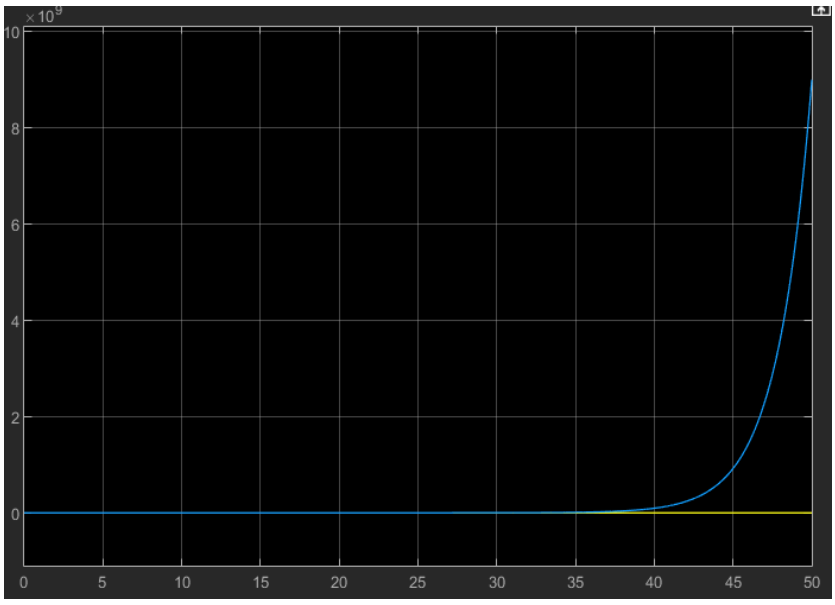
3. M= +5%, L= L (No change)

The controller breaks down and could not control the system.



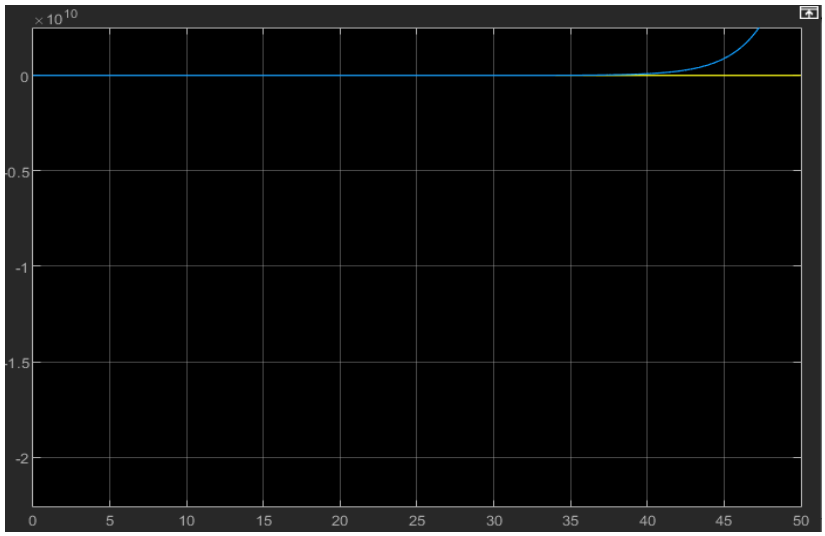
4. M= -5%, L= L (No change)

The controller breaks down and could not control the system.



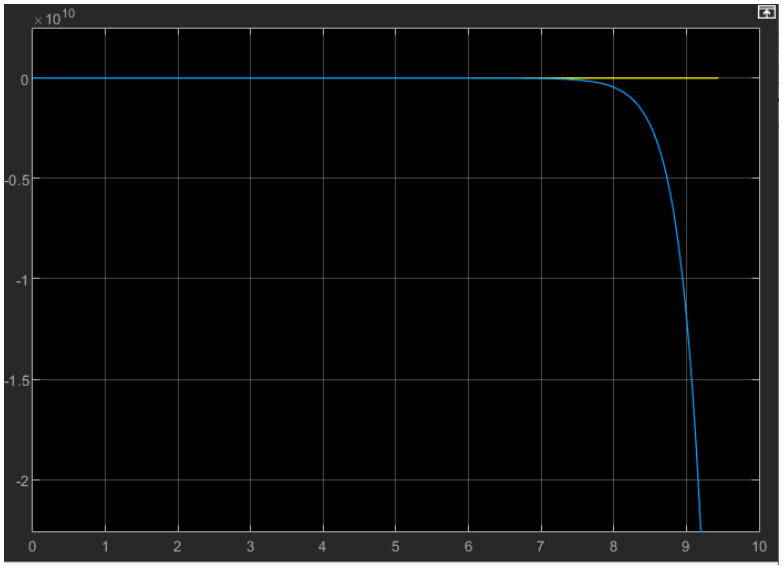
5. M= -5%, L= -5%

The controller breaks down and could not control the system.



6. M= +5%, L= +5%

The controller breaks down and could not control the system.



As it can be seen on the figures above, the output barely changes for 5% changes in M and L. Therefore, the system could be called robust.