Fall 2022

Final Project

Control of Robotic Systems

December 20, 2022

Students:
Shantanu Parab(sparab)

Vineet Singh(vsingh03)

Instructors:
Dr. Waseem Ansar Mallik

Course code:
ENPM667

Contents

1 Introduction **3**

2 Dynamic Equations of Motion **3**

3 Linearization and State Space representation **5**

4 Controllability Analysis **5**

5 LQR Controller **13**

6 Observability Analysis **21**

7 Luenberger Observer Calculation **23**
7.1 Luenberger Linear 23
7.2 Luenberger Non Linear Analysis 30

8 LQG Feedback Controller Analysis **38**
8.1 LQG linear analysis 38
8.2 LQG Non-linear analysis 46

9 Conclusion **52**

1 Introduction

This project focuses on the design and implementation of a non linear system dynamics to linearize the system around a given equilibrium. The linearization is done using the Jacobian linearization. Both the matrices A and B are linearized with respect to the states variables and input respectively. The linearized models are tested for controllability and the condition for keeping the system controllable. This test is done using the property of determinant of the Controllability matrix. The controllability is then alalyzed after substituting the physical parameters into the linearized system dynamics.

An LQR controller is developed and simulated for the initial condition and the step response. Further to this, observability analysis is done on the linear equations. Three different cases were analyzed corresponding to different states being observed on the output. The cases which were observable were then used to design a Luenberger Observer and subsequently an LQG controller. These were also simulated for the cases of initial and step response. The LQG controller was able to correctly trace the system initial state using the Kalman filter (a Linear Quadratic Estimator). Finally the system was tested by adding White Gaussian Noise to the system. The nature of white noise and the final system output error is finally shown at the end of this report.

2 Dynamic Equations of Motion

The system consists of a cart of mass M, to which two masses of m_1 and m_2 are suspended using cables of length l_1 and l_2 respectively. The displacement of the cart is measured from its center and given by x. The angular displacement of the masses m_1 and m_2 is given by θ_1 and θ_2 respectively.

The cart is restricted to linear motion and is subjected to force F affecting its motion.

Equation of motion follows the Euler-Lagrange method the generalized coordinates we have are x, θ_1, θ_2. The Euler-Lagrange equation will be given as

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{x}} - \frac{\partial L}{\partial x} = F$$

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta_1}} - \frac{\partial L}{\partial \theta_1} = 0$$

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta_1}} - \frac{\partial L}{\partial \theta_1} = 0$$

Where L=K-P, K is the Kinetic energy of the system and P is the potential energy of the system.

We first derive the potential energy. The potential energy of the system is gven by

$$P = -gl_1m_1\cos(\theta_1) - gl_2m_2\cos(\theta_2)$$

The Kinetic energy of the system is given as

$$K = \frac{M\dot{x}^2}{2} + \frac{l_1^2 m_1 \sin^2(\theta_1)\dot{\theta}_1^2}{2} + \frac{l_2^2 m_2 \sin^2(\theta_2)\dot{\theta}_2^2}{2} + \frac{m_1\left(-l_1\cos(\theta_1)\dot{\theta}_1 + \dot{x}\right)^2}{2} + \frac{m_2\left(-l_2\cos(\theta_2)\dot{\theta}_2 + \dot{x}\right)^2}{2}$$

Therefore L can be calculated as L=K-P

$$L = \frac{M\dot{x}^2}{2} + gl_1m_1\cos(\theta_1) + gl_2m_2\cos(\theta_2) + \frac{l_1^2m_1\dot{\theta}_1^2}{2} - l_1m_1\cos(\theta_1)\dot{\theta}_1\dot{x} + \frac{l_2^2m_2\dot{\theta}_2^2}{2} - l_2m_2\cos(\theta_2)\dot{\theta}_2\dot{x} + \frac{m_1\dot{x}^2}{2} + \frac{m_2\dot{x}^2}{2}$$

We can further compute,

For x:

$$\frac{\partial L}{\partial \dot{x}} = M\dot{x} - l_1 m_1 \cos(\theta_1)\dot{\theta}_1 - l_2 m_2 \cos(\theta_2)\dot{\theta}_2 + m_1\dot{x} + m_2\dot{x}$$

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{x}} = M\ddot{x} + l_1 m_1 \sin(\theta_1)\dot{\theta}_1^2 - l_1 m_1 \cos(\theta_1)\ddot{\theta}_1 + l_2 m_2 \sin(\theta_2)\dot{\theta}_2^2 - l_2 m_2 \cos(\theta_2)\ddot{\theta}_2 + m_1\ddot{x} + m_2\ddot{x}$$

$$\frac{\partial L}{\partial x} = 0$$

---

for θ_1:

$$\frac{\partial L}{\partial \dot{\theta_1}} = l_1 m_1 \left(l_1\dot{\theta}_1 - \cos(\theta_1)\dot{x}\right)$$

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta_1}} = l_1 m_1 \left(l_1\ddot{\theta}_1 + \sin(\theta_1)\dot{\theta}_1\dot{x} - \cos(\theta_1)\ddot{x}\right)$$

$$\frac{\partial L}{\partial \theta_1} = l_1 m_1 \left(-g + \dot{\theta}_1\dot{x}\right)\sin(\theta_1)$$

---

for θ_2:

$$\frac{\partial L}{\partial \dot{\theta_2}} = l_2 m_2 \left(l_2\dot{\theta}_2 - \cos(\theta_2)\dot{x}\right)$$

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta_2}} = l_2 m_2 \left(l_2\ddot{\theta}_2 + \sin(\theta_2)\dot{\theta}_2\dot{x} - \cos(\theta_2)\ddot{x}\right)$$

$$\frac{\partial L}{\partial \theta_2} = l_2 m_2 \left(-g + \dot{\theta}_2\dot{x}\right)\sin(\theta_2)$$

Substituting the values we have got in the Lagrange equations we get

$$M\ddot{x} + l_1 m_1 \sin(\theta_1)\dot{\theta}_1^2 - l_1 m_1 \cos(\theta_1)\ddot{\theta}_1 + l_2 m_2 \sin(\theta_2)\dot{\theta}_2^2 - l_2 m_2 \cos(\theta_2)\ddot{\theta}_2 + m_1\ddot{x} + m_2\ddot{x} = F$$

$$l_1 m_1 \left(g\sin(\theta_1) + l_1\ddot{\theta}_1 - \cos(\theta_1)\ddot{x}\right) = 0$$

$$l_2 m_2 \left(g\sin(\theta_2) + l_2\ddot{\theta}_2 - \cos(\theta_2)\ddot{x}\right) = 0$$

Substituting and rearranging the equations into one another to get rid of the energy terms we get the following final equations.

$$\ddot{x} = \frac{F - \frac{g m_1 \sin(2\theta_1)}{2} - \frac{g m_2 \sin(2\theta_2)}{2} - l_1 m_1 \sin(\theta_1)\dot{\theta}_1^2 - l_2 m_2 \sin(\theta_2)\dot{\theta}_2^2}{M + m_1 \sin^2(\theta_1) + m_2 \sin^2(\theta_2)}$$

$$\ddot{\theta_1} = \frac{-g\left(M + m_1\sin^2(\theta_1) + m_2\sin^2(\theta_2)\right)\sin(\theta_1) - \frac{\left(-2F + gm_1\sin(2\theta_1) + gm_2\sin(2\theta_2) + 2l_1m_1\sin(\theta_1)\dot{\theta}_1^2 + 2l_2m_2\sin(\theta_2)\dot{\theta}_2^2\right)\cos(\theta_1)}{2}}{l_1\left(M + m_1\sin^2(\theta_1) + m_2\sin^2(\theta_2)\right)}$$

$$\ddot{\theta_2} = \frac{-g\left(M + m_1\sin^2(\theta_1) + m_2\sin^2(\theta_2)\right)\sin(\theta_2) - \frac{\left(-2F + gm_1\sin(2\theta_1) + gm_2\sin(2\theta_2) + 2l_1m_1\sin(\theta_1)\dot{\theta}_1^2 + 2l_2m_2\sin(\theta_2)\dot{\theta}_2^2\right)\cos(\theta_2)}{2}}{l_2\left(M + m_1\sin^2(\theta_1) + m_2\sin^2(\theta_2)\right)}$$

3 Linearization and State Space representation

We consider the states to be 0 in equilibrium conditions. The linearization around the equilibrium can be obtained as follows.

$$A_F = \nabla_{\bar{X}}|F(X,U) = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{gm_1}{M} & 0 & -\frac{gm_2}{M} & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{-Mg-gm_1}{Ml_1} & 0 & -\frac{gm_2}{Ml_1} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & -\frac{gm_1}{Ml_2} & 0 & \frac{-Mg-gm_2}{Ml_2} & 0 \end{bmatrix}$$

$$B_F = \nabla_{\bar{U}}|F(X,U) = \begin{bmatrix} 0 \\ \frac{1}{M} \\ 0 \\ \frac{1}{Ml_1} \\ 0 \\ \frac{1}{Ml_2} \end{bmatrix}$$

The linearized system is given as follows

$$\dot{X(t)} = A_F X(t) + B_F U(t)$$

4 Controllability Analysis

A system is said to be controllable if the system can be driven to any state within its subspace i.e. the eigenvalues of the system can be placed at any position within the subspace. To develop an intuition, let us consider a system given as:

$$\dot{x} = Ax + Bu$$

, where u is the input to the system. If $u = -Kx$ is considered as feedback then, the new state equation can be written as :

$$\dot{X} = (A - BK)x$$

This is the new dynamics of the system, and here the values of K can be adjusted to drive the system to any reachable state.

To test whether a system is controllable or not, a special matrix called Controllability Matrix (ζ) is used and is given as:

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

A system is controllable if the Controllability matrix has n linearly independent columns then the system is controllable, where n is the number of linearized states of the system. In other words, if the rank of $\zeta = n$, then the system is controllable.

For simulation purposes in MATLAB, the function

$$>> ctrb(A, B)$$

takes in the A and B matrix and returns the ζ matrix. All the controllability analysis in this project is done using this function (where the values of the variables are available).

In summary, if a system is controllable, then poles of the system can be placed arbitrarily anywhere as needed, by choosing a feedback $(-Kx)$ and the system can be driven anywhere needed with an appropriate value of u.

Equivalences for Controllability:

1. System is controllable.
2. Arbitrary eigenvalue (pole) placement.
3. Full Reachability in $\Re^n$. i.e. Reachable set $R_t = \xi \in \Re^n|$ there is an input u(t) such that x(t) $= \xi$; $R_t = \Re^n$

For a system to be controllable the Gramian of controllability should be invertible. The Gramian of Controllability is invertible if and only if the $n \times nm$ controllability matrix satisfies

$$rank[B_k AB_k A^2 B_K A^3 B_K A^4 B_K A^5 B_K] = n$$

The Controllability matrix is given as:

$$Q = \begin{bmatrix} 0 & \frac{1}{M} & 0 & -\frac{gm_2}{M^2 l_2} - \frac{gm_1}{M^2 l_1} & 0 \\ \frac{1}{M} & 0 & -\frac{gm_2}{M^2 l_2} - \frac{gm_1}{M^2 l_1} & 0 & \frac{\frac{g^2 m_1 m_2}{M^2 l_1} - \frac{gm_2(-Mg-gm_2)}{M^2 l_2}}{Ml_2} + \frac{\frac{g^2 m_1 m_2}{M^2 l_2} - \frac{gm_1(-Mg-gm_1)}{M^2 l_1}}{Ml_1} \\ 0 & \frac{1}{Ml_1} & 0 & -\frac{gm_2}{M^2 l_1 l_2} + \frac{-Mg-gm_1}{M^2 l_1^2} & 0 \\ \frac{1}{Ml_1} & 0 & -\frac{gm_2}{M^2 l_1 l_2} + \frac{-Mg-gm_1}{M^2 l_1^2} & 0 & \frac{-\frac{gm_2(-Mg-gm_2)}{M^2 l_1 l_2} - \frac{gm_2(-Mg-gm_1)}{M^2 l_1^2}}{Ml_2} + \frac{\frac{g^2 m_1 m_2}{M^2 l_1 l_2} + \frac{(-Mg-gm_1)^2}{M^2 l_1^2}}{Ml_1} \\ 0 & \frac{1}{Ml_2} & 0 & -\frac{gm_1}{M^2 l_1 l_2} + \frac{-Mg-gm_2}{M^2 l_2^2} & 0 \\ \frac{1}{Ml_2} & 0 & -\frac{gm_1}{M^2 l_1 l_2} + \frac{-Mg-gm_2}{M^2 l_2^2} & 0 & \frac{\frac{g^2 m_1 m_2}{M^2 l_1 l_2} + \frac{(-Mg-gm_2)^2}{M^2 l_2^2}}{Ml_2} + \frac{-\frac{gm_1(-Mg-gm_2)}{M^2 l_2^2} - \frac{gm_1(-Mg-gm_1)}{M^2 l_1 l_2}}{Ml_1} \end{bmatrix}$$

The determinant of the above matrix is given as

$$det|Q| = -\frac{g^6 l_1^2 - 2g^6 l_1 l_2 + g^6 l_2^2}{M^6 l_1^6 l_2^6}$$

If the determinant of the matrix is zero that means the matrix is not full rank. There exist a column of zeros that makes the determinant zero. From the above determinant, it is visible that if $l_1 = l_2$ the determinant will be zero. The rank of the matrix Q reduces to 4 when $l_1 = l_2$. Therefore, we have the condition $l_1 = l_2$ for which the system is uncontrollable.

Linearization Script

December 20, 2022

[1]:
```
from sympy import Matrix, symbols, cos, sin, simplify, pi, pprint, diff
from numpy import linspace, meshgrid, ones_like
import matplotlib.pyplot as plt
import time
import math
import sympy as sp
import pprint
pp = pprint.PrettyPrinter(indent=4)
from sympy.physics.mechanics import dynamicsymbols, init_vprinting
x, y = dynamicsymbols('x y')
t = sp.Symbol('t')
init_vprinting()
```

[2]:
```
P, L, K = symbols('P, L, K')
# m1, m2, m1, m2, v1x, v2x, v1y, v2y,    = symbols('x_{m1},x_{m2},y_{m1}, y_{m2},
 ↪v_{1x}, v_{2x}, v_{1y}, v_{2y} l_1, l_2')
g, M, m1, m2, l1, l2 = symbols('g, M, m_1, m_2 l_1 l_2')
x, theta1, theta2=dynamicsymbols(' x, theta_1, theta_2')
```

[3]:
```
# Potential energy:
P = -(m1*g*l1*cos(theta1) + m2*g*l2*cos(theta2))
P
```

[3]:
$$-gl_1m_1\cos\left(\theta_1\right) - gl_2m_2\cos\left(\theta_2\right)$$

[4]:
```
# Calculation of position and velocities for m1 and m2
xm1 = x -l1*sin(theta1)
ym1 = -l1*cos(theta1)
v1x = x.diff(t) - l1*theta1.diff(t)*cos(theta1)
v1y = l1*theta1.diff(t)*sin(theta1)
xm2 = x - l2*sin(theta2)
ym2 = -l2*cos(theta2)
v2x = x.diff(t) - l2*theta2.diff(t)*cos(theta2)
v2y = l2*theta2.diff(t)*sin(theta2)

# Kinetic Energy
K = (M*x.diff(t)**2 + m1*v1x**2 + m1*v1y**2 + m2*v2x**2 + m2*v2y**2)/2
K
```

[4]:
$$\frac{M\dot{x}^2}{2} + \frac{l_1^2 m_1 \sin^2(\theta_1)\dot{\theta}_1^2}{2} + \frac{l_2^2 m_2 \sin^2(\theta_2)\dot{\theta}_2^2}{2} + \frac{m_1\left(-l_1\cos(\theta_1)\dot{\theta}_1 + \dot{x}\right)^2}{2} + \frac{m_2\left(-l_2\cos(\theta_2)\dot{\theta}_2 + \dot{x}\right)^2}{2}$$

[5]:
```
L = K - P
L = L.expand().simplify()
L
```

[5]:
$$\frac{M\dot{x}^2}{2} + gl_1m_1\cos(\theta_1) + gl_2m_2\cos(\theta_2) + \frac{l_1^2m_1\dot{\theta}_1^2}{2} - l_1m_1\cos(\theta_1)\dot{\theta}_1\dot{x} + \frac{l_2^2m_2\dot{\theta}_2^2}{2} - l_2m_2\cos(\theta_2)\dot{\theta}_2\dot{x} + \frac{m_1\dot{x}^2}{2} + \frac{m_2\dot{x}^2}{2}$$

[6]:
```
# Now calculating the Euler-Lagrange equation terms for each system as separate
Lt1dot = L.diff(theta1.diff(t))
Lt1 = diff(L,theta1)
Lt2dot = diff(L,theta2.diff(t))
Lt2 = diff(L,theta2)

Lx = diff(L,x)
```

[7]:
```
Lx.simplify()
```

[7]:
$$0$$

[8]:
```
Lxdot = diff(L,x.diff(t))
Lxdot
```

[8]:
$$M\dot{x} - l_1m_1\cos(\theta_1)\dot{\theta}_1 - l_2m_2\cos(\theta_2)\dot{\theta}_2 + m_1\dot{x} + m_2\dot{x}$$

[9]:
```
Lt1dot.simplify()
```

[9]:
$$l_1m_1\left(l_1\dot{\theta}_1 - \cos(\theta_1)\dot{x}\right)$$

[10]:
```
Lt1.simplify()
```

[10]:
$$l_1m_1\left(-g + \dot{\theta}_1\dot{x}\right)\sin(\theta_1)$$

[11]:
```
Lt2dot.simplify()
```

[11]:
$$l_2m_2\left(l_2\dot{\theta}_2 - \cos(\theta_2)\dot{x}\right)$$

[12]:
```
Lt2.simplify()
```

[12]:
$$l_2m_2\left(-g + \dot{\theta}_2\dot{x}\right)\sin(\theta_2)$$

[13]:
```
Lxdot.simplify()
```

[13]:
$$M\dot{x} - l_1m_1\cos(\theta_1)\dot{\theta}_1 - l_2m_2\cos(\theta_2)\dot{\theta}_2 + m_1\dot{x} + m_2\dot{x}$$

[14]:
```
Lt1dotdot = Lt1dot.diff(t)
Lt2dotdot = Lt2dot.diff(t)
Lxdotdot = Lxdot.diff(t)
Lxdotdot.simplify()
```

[14]: $$M\ddot{x} + l_1 m_1 \sin(\theta_1)\dot{\theta}_1^2 - l_1 m_1 \cos(\theta_1)\ddot{\theta}_1 + l_2 m_2 \sin(\theta_2)\dot{\theta}_2^2 - l_2 m_2 \cos(\theta_2)\ddot{\theta}_2 + m_1\ddot{x} + m_2\ddot{x}$$

[15]:
```
Lt1dotdot.simplify()
```

[15]: $$l_1 m_1 \left(l_1\ddot{\theta}_1 + \sin(\theta_1)\dot{\theta}_1\dot{x} - \cos(\theta_1)\ddot{x}\right)$$

[16]:
```
Lt2dotdot.simplify()
```

[16]: $$l_2 m_2 \left(l_2\ddot{\theta}_2 + \sin(\theta_2)\dot{\theta}_2\dot{x} - \cos(\theta_2)\ddot{x}\right)$$

[17]:
```
Lxdotdot-Lx
```

[17]: $$M\ddot{x} + l_1 m_1 \sin(\theta_1)\dot{\theta}_1^2 - l_1 m_1 \cos(\theta_1)\ddot{\theta}_1 + l_2 m_2 \sin(\theta_2)\dot{\theta}_2^2 - l_2 m_2 \cos(\theta_2)\ddot{\theta}_2 + m_1\ddot{x} + m_2\ddot{x}$$

[18]:
```
F=symbols('F')
```

[19]:
```
# Calculating FInal Euler Lagrange equations.

eqn1 = Lxdotdot-Lx
eqn1=sp.Eq(eqn1.simplify(),F)
# eqn1=sp.solve(eqn1,x.diff(t).diff(t))
```

[20]:
```
eqn1
```

[20]: $$M\ddot{x} + l_1 m_1 \sin(\theta_1)\dot{\theta}_1^2 - l_1 m_1 \cos(\theta_1)\ddot{\theta}_1 + l_2 m_2 \sin(\theta_2)\dot{\theta}_2^2 - l_2 m_2 \cos(\theta_2)\ddot{\theta}_2 + m_1\ddot{x} + m_2\ddot{x} = F$$

[21]:
```
eqn2 = Lt1dotdot-Lt1
# L2=sp.Eq(eqn2.simplify())
eqn2 = sp.solve(sp.Eq(eqn2.simplify(),0), theta1.diff(t).diff(t))
eqn2[0]
```

[21]: $$\frac{-g\sin(\theta_1) + \cos(\theta_1)\ddot{x}}{l_1}$$

[22]:
```
eqn3 = Lt2dotdot-Lt2
# L3=sp.Eq(eqn3.simplify())
eqn3 = sp.solve(sp.Eq(eqn3.simplify(),0), theta2.diff(t).diff(t))
eqn3[0]
# L3
```

[22]: $$\frac{-g\sin(\theta_2) + \cos(\theta_2)\ddot{x}}{l_2}$$

[23]:
```
eqn2[0]
```

[23]: $$\frac{-g\sin(\theta_1) + \cos(\theta_1)\ddot{x}}{l_1}$$

[24]:
```
eqn3[0]
```

[24]:
$$\frac{-g\sin\left(\theta_2\right)+\cos\left(\theta_2\right)\ddot{x}}{l_2}$$

[25]:
```
a=eqn1.subs([(theta1.diff(t).diff(t),eqn2[0]), (theta2.diff(t).diff(t),eqn3[0])])
a
```

[25]:
$$M\ddot{x} + l_1m_1\sin\left(\theta_1\right)\dot{\theta}_1^2 + l_2m_2\sin\left(\theta_2\right)\dot{\theta}_2^2 - m_1\left(-g\sin\left(\theta_1\right)+\cos\left(\theta_1\right)\ddot{x}\right)\cos\left(\theta_1\right) + m_1\ddot{x} - m_2\left(-g\sin\left(\theta_2\right)+\cos\left(\theta_2\right)\ddot{x}\right)\cos\left(\theta_2\right)+m_2\ddot{x} = F$$

[26]:
```
a=sp.solve(a,x.diff(t).diff(t))
a=simplify(a[0])
a
```

[26]:
$$\frac{F-\frac{gm_1\sin\left(2\theta_1\right)}{2}-\frac{gm_2\sin\left(2\theta_2\right)}{2}-l_1m_1\sin\left(\theta_1\right)\dot{\theta}_1^2-l_2m_2\sin\left(\theta_2\right)\dot{\theta}_2^2}{M+m_1\sin^2\left(\theta_1\right)+m_2\sin^2\left(\theta_2\right)}$$

[27]:
```
b=eqn2[0].subs(x.diff(t).diff(t),a)
b=simplify(b)
b
```

[27]:
$$\frac{-g\left(M+m_1\sin^2\left(\theta_1\right)+m_2\sin^2\left(\theta_2\right)\right)\sin\left(\theta_1\right)-\frac{\left(-2F+gm_1\sin\left(2\theta_1\right)+gm_2\sin\left(2\theta_2\right)+2l_1m_1\sin\left(\theta_1\right)\dot{\theta}_1^2+2l_2m_2\sin\left(\theta_2\right)\dot{\theta}_2^2\right)\cos\left(\theta_1\right)}{2}}{l_1\left(M+m_1\sin^2\left(\theta_1\right)+m_2\sin^2\left(\theta_2\right)\right)}$$

[28]:
```
c=eqn3[0].subs(x.diff(t).diff(t),a)
c=simplify(c)
c
```

[28]:
$$\frac{-g\left(M+m_1\sin^2\left(\theta_1\right)+m_2\sin^2\left(\theta_2\right)\right)\sin\left(\theta_2\right)-\frac{\left(-2F+gm_1\sin\left(2\theta_1\right)+gm_2\sin\left(2\theta_2\right)+2l_1m_1\sin\left(\theta_1\right)\dot{\theta}_1^2+2l_2m_2\sin\left(\theta_2\right)\dot{\theta}_2^2\right)\cos\left(\theta_2\right)}{2}}{l_2\left(M+m_1\sin^2\left(\theta_1\right)+m_2\sin^2\left(\theta_2\right)\right)}$$

[29]:
```
Non_linear_state_eqn = sp.Matrix([[x.diff(t)],[a],[theta1.diff(t)], [b], [theta2.
↪diff(t)], [c]])
Non_linear_state_eqn
```

[29]:
$$\begin{bmatrix} \dot{x} \\ \frac{F-\frac{gm_1\sin\left(2\theta_1\right)}{2}-\frac{gm_2\sin\left(2\theta_2\right)}{2}-l_1m_1\sin\left(\theta_1\right)\dot{\theta}_1^2-l_2m_2\sin\left(\theta_2\right)\dot{\theta}_2^2}{M+m_1\sin^2\left(\theta_1\right)+m_2\sin^2\left(\theta_2\right)} \\ \dot{\theta}_1 \\ \frac{-g\left(M+m_1\sin^2\left(\theta_1\right)+m_2\sin^2\left(\theta_2\right)\right)\sin\left(\theta_1\right)-\frac{\left(-2F+gm_1\sin\left(2\theta_1\right)+gm_2\sin\left(2\theta_2\right)+2l_1m_1\sin\left(\theta_1\right)\dot{\theta}_1^2+2l_2m_2\sin\left(\theta_2\right)\dot{\theta}_2^2\right)\cos\left(\theta_1\right)}{2}}{l_1\left(M+m_1\sin^2\left(\theta_1\right)+m_2\sin^2\left(\theta_2\right)\right)} \\ \dot{\theta}_2 \\ \frac{-g\left(M+m_1\sin^2\left(\theta_1\right)+m_2\sin^2\left(\theta_2\right)\right)\sin\left(\theta_2\right)-\frac{\left(-2F+gm_1\sin\left(2\theta_1\right)+gm_2\sin\left(2\theta_2\right)+2l_1m_1\sin\left(\theta_1\right)\dot{\theta}_1^2+2l_2m_2\sin\left(\theta_2\right)\dot{\theta}_2^2\right)\cos\left(\theta_2\right)}{2}}{l_2\left(M+m_1\sin^2\left(\theta_1\right)+m_2\sin^2\left(\theta_2\right)\right)} \end{bmatrix}$$

[30]:
```
Linear_state_eqn = sp.Matrix([])

states = [x,x.diff(t),theta1,theta1.diff(t),theta2,theta2.diff(t)]
```

```
for i in range(6):
        a = Non_linear_state_eqn.diff(states[i])
        Linear_state_eqn=Linear_state_eqn.col_insert(i,a)
```

[31]:
```
Linear_state_eqn = Linear_state_eqn.subs([(x, 0), (theta1, 0), (theta2, 0)])
Linear_state_eqn=simplify(Linear_state_eqn)
A=Linear_state_eqn #A
A
```

[31]:
$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{gm_1}{M} & 0 & -\frac{gm_2}{M} & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{-Mg-gm_1}{Ml_1} & 0 & -\frac{gm_2}{Ml_1} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & -\frac{gm_1}{Ml_2} & 0 & \frac{-Mg-gm_2}{Ml_2} & 0 \end{bmatrix}$$

[32]:
```
Linear_state_eqnB= sp.Matrix([])

# states = [x,x.diff(t),theta1,theta1.diff(t),theta2,theta2.diff(t)]


a = Non_linear_state_eqn.diff(F)
Linear_state_eqnB=Linear_state_eqnB.col_insert(1,a)
Linear_state_eqnB = Linear_state_eqnB.subs([(x, 0), (theta1, 0), (theta2, 0)])
Linear_state_eqnB=simplify(Linear_state_eqnB)
B=Linear_state_eqnB #B
B
```

[32]:
$$\begin{bmatrix} 0 \\ \frac{1}{M} \\ 0 \\ \frac{1}{Ml_1} \\ 0 \\ \frac{1}{Ml_2} \end{bmatrix}$$

[33]:
```
C_rank=sp.Matrix([]).col_insert(0,B).col_insert(1,A*B).col_insert(2,A*A*B).
↪col_insert(3,A*A*A*B).col_insert(4,A*A*A*A*B).col_insert(5,A*A*A*A*A*B)
```

[34]:
```
C_rank
```

[34]:

$$\begin{bmatrix} 0 & \frac{1}{M} & 0 & -\frac{gm_2}{M^2l_2} - \frac{gm_1}{M^2l_1} & 0 \\ \frac{1}{M} & 0 & -\frac{gm_2}{M^2l_2} - \frac{gm_1}{M^2l_1} & 0 & \frac{\frac{g^2m_1m_2}{M^2l_1} - \frac{gm_2(-Mg-gm_2)}{M^2l_2}}{Ml_2} + \frac{\frac{g^2m_1m_2}{M^2l_2} - \frac{gm_1(-Mg-gm_1)}{M^2l_1}}{Ml_1} \\ 0 & \frac{1}{Ml_1} & 0 & -\frac{gm_2}{M^2l_1l_2} + \frac{-Mg-gm_1}{M^2l_1^2} & 0 \\ \frac{1}{Ml_1} & 0 & -\frac{gm_2}{M^2l_1l_2} + \frac{-Mg-gm_1}{M^2l_1^2} & 0 & \frac{-\frac{gm_2(-Mg-gm_2)}{M^2l_1l_2} - \frac{gm_2(-Mg-gm_1)}{M^2l_1^2}}{Ml_2} + \frac{\frac{g^2m_1m_2}{M^2l_1l_2} + \frac{(-Mg-gm_1)^2}{M^2l_1^2}}{Ml_1} \\ 0 & \frac{1}{Ml_2} & 0 & -\frac{gm_1}{M^2l_1l_2} + \frac{-Mg-gm_2}{M^2l_2^2} & 0 \\ \frac{1}{Ml_2} & 0 & -\frac{gm_1}{M^2l_1l_2} + \frac{-Mg-gm_2}{M^2l_2^2} & 0 & \frac{\frac{g^2m_1m_2}{M^2l_1l_2} + \frac{(-Mg-gm_2)^2}{M^2l_2^2}}{Ml_2} + \frac{-\frac{gm_1(-Mg-gm_2)}{M^2l_2^2} - \frac{gm_1(-Mg-gm_1)}{M^2l_1l_2}}{Ml_1} \end{bmatrix}$$

[35]:
```
C_rank.det()
```

[35]:
$$-\frac{g^6l_1^2 - 2g^6l_1l_2 + g^6l_2^2}{M^6l_1^6l_2^6}$$

If the determinant of the matrix is zero that means te matrix is not full rank. There exist a coulumn of zero's that make the determinant zero. From the above determinant it is visible that if l1=l2 the determinant will be zero.

[36]:
```
C_rank=C_rank.subs(l1,l2)
```

[37]:
```
C_rank
```

[37]:
$$\begin{bmatrix} 0 & \frac{1}{M} & 0 & -\frac{gm_1}{M^2l_2} - \frac{gm_2}{M^2l_2} & 0 \\ \frac{1}{M} & 0 & -\frac{gm_1}{M^2l_2} - \frac{gm_2}{M^2l_2} & 0 & \frac{\frac{g^2m_1m_2}{M^2l_2} - \frac{gm_1(-Mg-gm_1)}{M^2l_2}}{Ml_2} + \frac{\frac{g^2m_1m_2}{M^2l_2} - \frac{gm_2(-Mg-gm_2)}{M^2l_2}}{Ml_2} \\ 0 & \frac{1}{Ml_2} & 0 & -\frac{gm_2}{M^2l_2^2} + \frac{-Mg-gm_1}{M^2l_2^2} & 0 \\ \frac{1}{Ml_2} & 0 & -\frac{gm_2}{M^2l_2^2} + \frac{-Mg-gm_1}{M^2l_2^2} & 0 & \frac{-\frac{gm_2(-Mg-gm_1)}{M^2l_2^2} - \frac{gm_2(-Mg-gm_2)}{M^2l_2^2}}{Ml_2} + \frac{\frac{g^2m_1m_2}{M^2l_2^2} + \frac{(-Mg-gm_1)^2}{M^2l_2^2}}{Ml_2} \\ 0 & \frac{1}{Ml_2} & 0 & -\frac{gm_1}{M^2l_2^2} + \frac{-Mg-gm_2}{M^2l_2^2} & 0 \\ \frac{1}{Ml_2} & 0 & -\frac{gm_1}{M^2l_2^2} + \frac{-Mg-gm_2}{M^2l_2^2} & 0 & \frac{-\frac{gm_1(-Mg-gm_1)}{M^2l_2^2} - \frac{gm_1(-Mg-gm_2)}{M^2l_2^2}}{Ml_2} + \frac{\frac{g^2m_1m_2}{M^2l_2^2} + \frac{(-Mg-gm_2)^2}{M^2l_2^2}}{Ml_2} \end{bmatrix}$$

[39]:
```
C_rank.det()
```

[39]:
$$0$$

[ ]:
```
C_rank.rank()
```

Therefore we have the condition l1-l2 where the system is uncontrollable

5 LQR Controller

Controllability gives a binary result, i.e. whether a system is controllable or not. It does not give analysis about how controllable the system is in $\Re^n$? To analyze this, Singular Value Decomposition (also called Controllability Grammian) is used. The essence of a Controllability Grammian is to output a matrix of eigen vectors and each eigen vectors denotes a direction in the state space. In this matrix, the eigen vectors corresponding to the biggest value is the most controllable direction.

Physically it denotes that, the system can driven farthest in that direction with the same input than in any other direction, and is most controllable in that direction.

Using the above statement, it can be said that higher the values of eigen values in the negative plane, higher the system controllability and the desired state can be reached faster. To obtain the corresponding values of K gain matrix for any desired pole position, MATLAB provides the function which takes the matrices A, B, eigs (a vector of desired eigen values or poles).

$$>> K = place(A, B, eigs);$$

However, it is not necessary that very high negative values of poles are realizable in physical system, and that they have stable dynamics. To optimize the pole placement, an LQR controller is used, which guarantees the most optimal solution for this optimization problem.

For implementation of LQR, a cost function is built which optimizes result between the output error and the input activation energy used. The cost function is given as:

$$J = \int_0^\infty (x^T Q x + u^T R u)\, dt$$

For the given cost function there exists a linear controller given by

$$u = -Kx$$

which gives the optimal solution for the cost function. K can be solved either by Lagrange Multiplier or Ricatti equation.

For the given system, the controllability is analyzed after substituting the values in A and B matrix. The MATLAB code doing the analysis of Controllability is given below.

```
%%
%Declaring variables;
M= 1000; %Crane mass
m1= 100; % Load 1 mass
m2= 100; % Load 2 mass
l1= 20; % Cable length of Load 1
l2= 10; % Cable length of Load 2
g= 9.81;

A=[0  1       0                    0          0                    0;
   0  0  -(m1*g)/M                  0  -(m2*g)/M                    0;
   0  0       0                    1          0                    0;
   0  0  -((M+m1)*g)/(M*l1)        0  -(m2*g)/(M*l1)               0;
   0  0       0                    0          0                    1;
   0  0  -(m1*g)/(M*l2)            0  -(g*(M+m2))/(M*l2)           0];
disp("A matrix after values substitution: "); disp(A);

% Matrix B
 B=[0; 1/M; 0; 1/(M*(l1)); 0; 1/(M*l2)];
 disp("B matrix after value substitution : ");disp(B)
```

```
21
22 disp("Rank of controllability matrix: ");
23 disp(rank(ctrb(A,B)));
24
25 disp("Controllability Matrix is = ")
26 disp(ctrb(A,B));
27 disp(det(ctrb(A,B)));
28
29 disp("Eigen Values of A");
30 disp(eig(A));
31
32
33 %% Output as below:
34
35 A matrix after values substitution:
36          0    1.0000         0         0         0         0
37          0         0   -0.9810         0   -0.9810         0
38          0         0         0    1.0000         0         0
39          0         0   -0.5395         0   -0.0491         0
40          0         0         0         0         0    1.0000
41          0         0   -0.0981         0   -1.0791         0
42
43 B matrix after value substitution :
44    1.0e-03 *
45
46          0
47     1.0000
48          0
49     0.0500
50          0
51     0.1000
52
53 Rank of controllability matrix:
54      6
55
56 Controllability Matrix is =
57    1.0e-03 *
58
59          0    1.0000         0   -0.1472         0    0.1419
60     1.0000         0   -0.1472         0    0.1419         0
61          0    0.0500         0   -0.0319         0    0.0227
62     0.0500         0   -0.0319         0    0.0227         0
63          0    0.1000         0   -0.1128         0    0.1249
64     0.1000         0   -0.1128         0    0.1249         0
65
66   -1.3926e-24
67
68 Eigen Values of A
69    0.0000 + 0.0000i
70    0.0000 + 0.0000i
71   -0.0000 + 0.7285i
72   -0.0000 - 0.7285i
```

```
   0.0000 + 1.0430i
   0.0000 - 1.0430i
```

From the result it is seen that the system is controllable as the rank of the matrix is 6 = n (states of the system).

Hence designing the controller to take the system to any desired state is possible. The LQR implementation in the MATLAB code is as shown below:

```
%%
% Section to check the initial state of the system without LQR.
% The initial conditions are as follows.
X_0 = [0;0;10;0;20;0];
% We assume the values of Q and R.
Q=[ 1 0    0   0   0   0;
    0 10   0   0   0   0;
    0 0 1000 0   0   0;
    0 0    0 10 0    0;
    0 0    0   0 1000 0;
    0 0    0   0   0 10];
R=0.001;
% Q and R are a part of the cost function of LQR controller
% It is an trade-off between Q and R so we use them both to
% develop a system as per our priorities.
% Assumption: C matrix is a direct representation of the output
% matrix, which makes D=0
disp("Using MATLAB inbuild function ss to check the initial response of
    the system.");
disp("Using Q and R matrix of our assumption:");
C = eye(6); D = 0;
sys1 = ss(A,B,C,D);
% ss is the MATLAB function for
% calculating the state space representation of the system
disp("Starting grid");
figure
initial(sys1,X_0);

grid on

% The output of the code is as shown in the below graph.
```

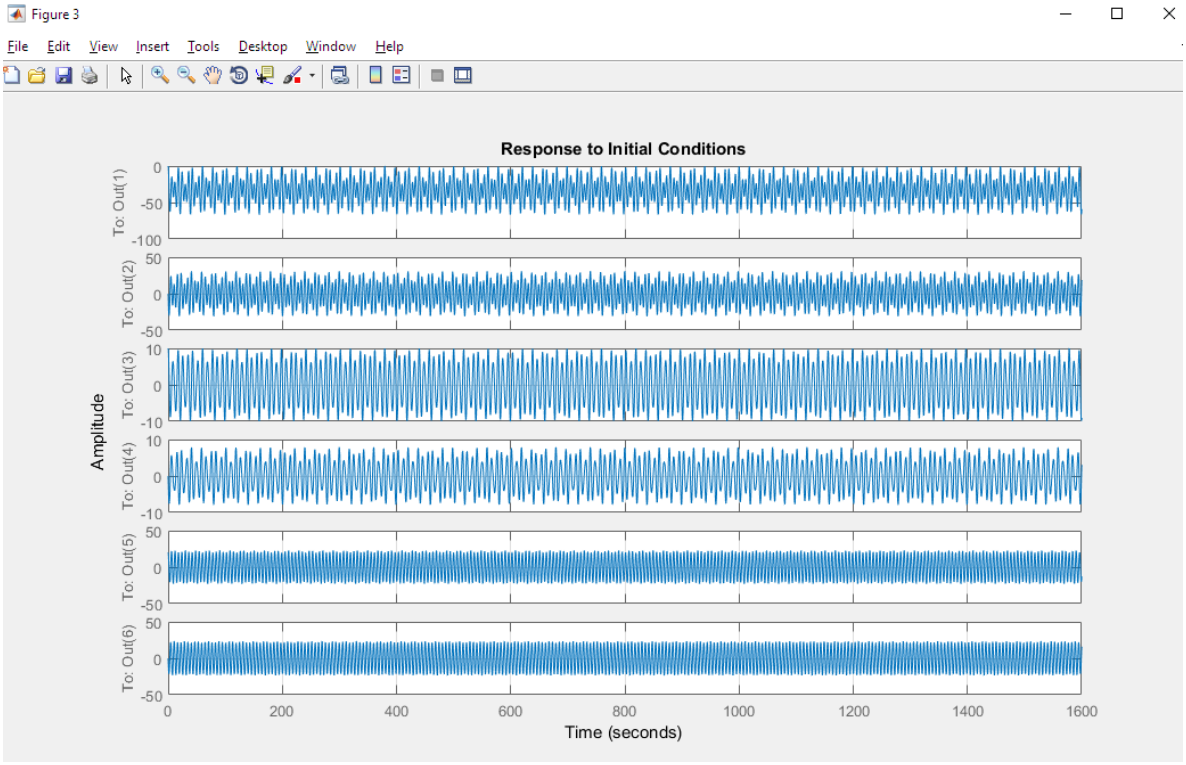

Figure 1: Response to inital condition for Linearized equation without LQR controller

The value of the Q matrix is chosen as per the needs of the states penalization. Considering that the error in states of θ_1 and θ_2 is not desirable. Hence the penalty for these errors are kept maximum. The errors in other states can be tolerated, and hence are selected accordingly.

The below codes and graph shows the LQR implementation on the developed Linear model of the system.

```
%%
% Section D : Checking that the system is controllable after value
    substitution

% Declaring variables;
M= 1000; %Crane mass
m1= 100; % Load 1 mass
m2= 100; % Load 2 mass
l1= 20; % Cable length of Load 1
l2= 10; % Cable length of Load 2
g= 9.81;

A=[0  1       0                    0       0                    0;
   0  0  -(m1*g)/M                  0  -(m2*g)/M                 0;
   0  0       0                    1       0                    0;
   0  0  -((M+m1)*g)/(M*l1)         0  -(m2*g)/(M*l1)            0;
   0  0       0                    0       0                    1;
   0  0  -(m1*g)/(M*l2)             0  -(g*(M+m2))/(M*l2)         0];
disp("A matrix after values substitution: "); disp(A);
```

```
19
20 % Matrix B
21 B=[0; 1/M; 0; 1/(M*(l1)); 0; 1/(M*l2)];
22 disp("B matrix after value substitution : ");disp(B)
23
24 disp("Rank of controllability matrix: ");
25 disp(rank(ctrb(A,B)));
26
27 disp("Controllability Matrix is = ")
28 disp(ctrb(A,B));
29 disp(det(ctrb(A,B)));
30
31 disp("Eigen Values of A");
32 disp(eig(A));
33
34 %%
35 % Section to check the initial state of the system without LQR.
36 % The initial conditions are as follows.
37 X_0 = [0;0;10;0;20;0];
38 % We assume the values of Q and R.
39 Q=[ 1 0    0   0   0   0;
40     0 10   0   0   0   0;
41     0 0 1000 0   0   0;
42     0 0    0 10 0    0;
43     0 0    0   0 1000 0;
44     0 0    0   0   0 10];
45 R=0.001;
46 % Q and R are a part of the cost function of LQR controller
47 % It is an trade-off between Q and R so we use them both to
48 % develop a system as per our priorities.
49 % Assumption: C matrix is a direct representation of the output
50 % matrix, which makes D=0
51 disp("Using MATLAB inbuild function ss to check the initial response of
       the system.");
52 disp("Using Q and R matrix of our assumption:");
53 C = eye(6); D = 0;
54 sys1 = ss(A,B,C,D);
55 % ss is the MATLAB function for
56 % calculating the state space representation of the system
57 disp("Starting grid");
58 figure
59 initial(sys1,X_0);
60
61 grid on
62
63
64 %%
65 % Design of the LQR controller is done in this section.
66 % Q and R are a part of the cost function of LQR controller
67 % It is an trade-off between Q and R so we use them both to
68 % develop a system as per our priorities.
69 % Assumption: C matrix is a direct representation of the output
```

```
% matrix, which makes D=0
disp("Using MATLAB inbuild function ss to check the initial response of the system.");
disp("Using Q and R matrix of our assumption:");
C = eye(6); D = 0;
sys1 = ss(A,B,C,D);
% ss is the MATLAB function for
% calculating the state space representation of the system
disp("Starting grid");
figure
initial(sys1,X_0);

grid on

%In-built MATLAB code for LQR Controllers
[K_Gain_mat, Po_def_mat, Poles] = lqr(A,B,Q,R);
disp("K gain matrix using lqr function of MATLAB: ");
disp(K_Gain_mat);

disp("Solution of the associated Ricatti equation using lqr function of MATLAB: ");
disp(Po_def_mat);

disp("Pole placement using lqr function of MATLAB: ");
disp(Poles);

disp("Using MATLAB inbuild function ss to check the initial response of the system.");
disp("Using Q and R matrix calculated by lqr matrix.");

sys2 = ss(A-(B*K_Gain_mat),B,C,D);
%Using the K matrix to define ss
figure
initial(sys2,X_0)
grid on
```

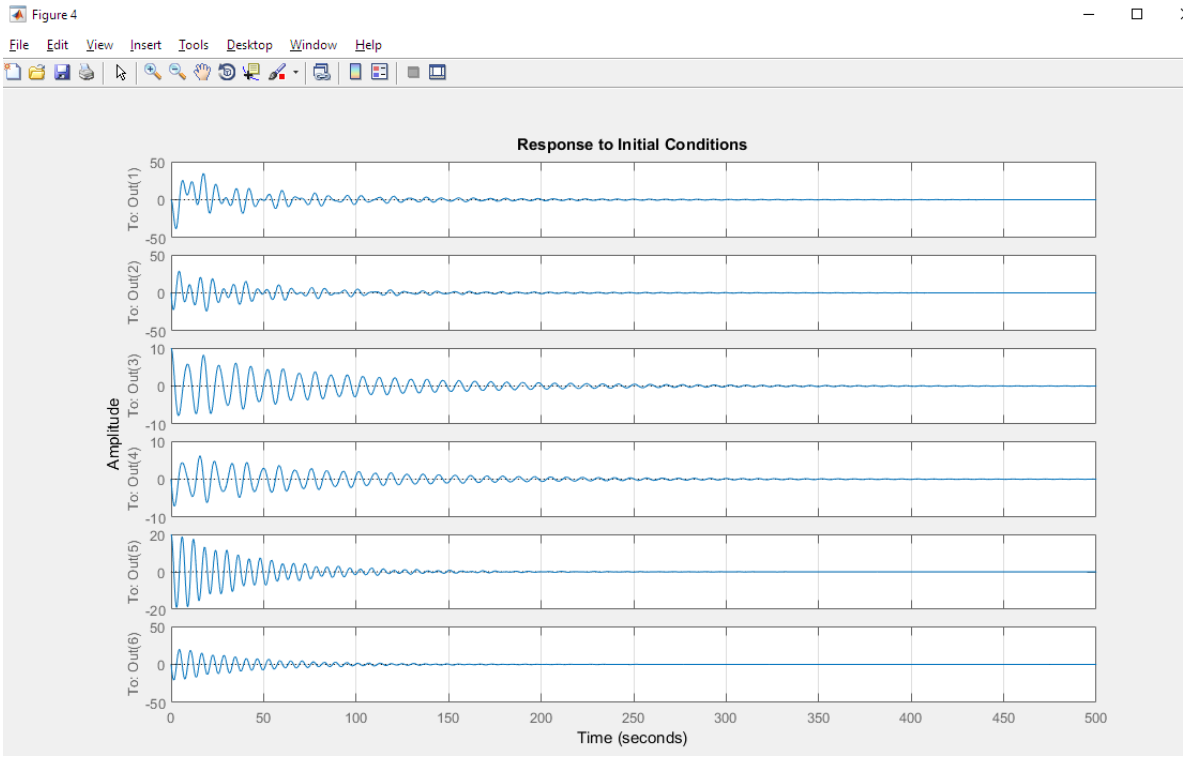

Figure 2: Response to inital condition for Linearized equation using LQR controller

For the non linear analysis, the initial condition is put in the original non linear equation. The code is done using the ode45() function to return the non-linear equation in a form which can be used to analyze the inital condition.

The below code shows the implementation in MATLAB and the generated graphs.

```
% Declaring the new output variables
% An x, theta_1 and theta_2 values are defined
% The system has the single derivatives of the values
% All of them are state variables and contribute to y0
y0 = [0; 0; 10; 0; 20; 0]
%defining the timespan
tspan = 0:0.01:8000;
%using ode45 function for defininig a differential equation
[t1,y1] = ode45(@cart2pend,tspan,y0);
% tspan = [t0 tf], integrates the system of differential equations y =f
    (t,y) from t0 to tf with initial conditions y0.

%plotting the function output on a 2D graph
plot(t1,y1)
grid on
function dydt = cart2pend(t,y)
M=1000;
m1=100;
m2=100;
l1=20;
```

```
l2=10;
g=9.81;
A=[0  1  0  0  0  0;
     0  0  -(m1*g)/M  0  -(m2*g)/M  0;
     0  0  0  1  0  0;
     0  0  -((M+m1)*g)/(M*l1)  0  -(m2*g)/(M*l1)  0;
     0  0  0  0  0  1;
     0  0  -(m1*g)/(M*l2)  0  -(g*(M+m2))/(M*l2)  0];
B=[0;  1/M;  0;  1/(M*l1);  0;  1/(M*l2)];
Q=[1  0  0  0  0  0;
    0  10  0  0  0  0;
    0  0  1000  0  0  0;
    0  0  0  10  0  0;
    0  0  0  0  1000  0;
    0  0  0  0  0  10];
R=0.1;
[K, S, P] = lqr(A,B,Q,R);
% calculates the optimal gain matrix K, the solution S of the associated
    algebraic Riccati
% equation and the closed-loop poles P using the continuous-time state-
    space matrices A and B.
F=-K*y;
dydt=zeros(6,1);
% y(1)=x;
dydt(1) = y(2);
%y(2)=xdot;
dydt(2)=(F-(g/2)*(m1*sind(2*y(3))+m2*sind(2*y(5)))-(m1*l1*(y(4)^2)*sind(y
    (3)))-(m2*l2*(y(6)^2)*sind(y(5))))/(M+m1*((sind(y(3)))^2)+m2*((sind(y
    (5)))^2));%X_DD
%y(3)=theta1;
dydt(3)= y(4);
%y(4)=theta1dot;
dydt(4)= (dydt(2)*cosd(y(3))-g*(sind(y(3))))/l1';
%y(5)=theta2;
dydt(5)= y(6);
%y(6)=theta2dot;
dydt(6)= (dydt(2)*cosd(y(5))-g*(sind(y(5))))/l2;
end
```

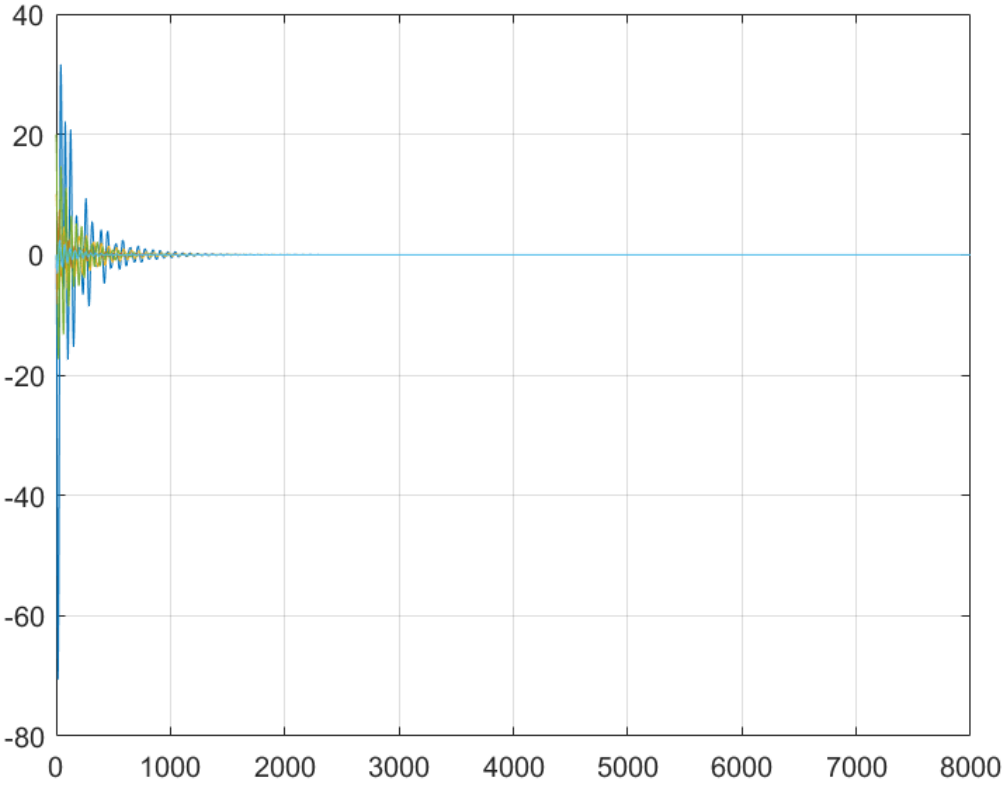

Figure 3: Response to inital condition for original non linear equation using LQR

6 Observability Analysis

The concept of state observation involves using current and past values of the plant input and output signals to generate the estimate of the assumed unknown current state. This is helpful as in practical cases, the entire output of the system is not accessible/cannot be measured at the controller.

The Observability of a system is given by the Observability Matrix. If the Observability matrix has a rank = n; where n is the number of states of the system then the system is said to be Observable. In case if the system is time variant then the Observability is given by the Observability Grammian which is the Singular Value Decomposition of the Observability Matrix. Observability matrix is given as: $\begin{bmatrix} C \\ AC \\ A^2C \\ . \\ . \\ . \\ A^{n-1}C \end{bmatrix}$

Below is the MATLAB script showing the simulation and final output. This is done using the Linear model developed in section 2 of the report.

```
%%
% Section E Code: Study of Observability:
syms M m1 m2 l1 l2 g;
%Declaring variables;
M= 1000; %Crane mass
m1= 100; % Load 1 mass
m2= 100; % Load 2 mass
l1= 20; % Cable length of Load 1
l2= 10; % Cable length of Load 2
g= 9.81;
% Creating a linearised state space equation using A and B matrices
A=[0 1      0                  0       0              0;
   0 0 -(m1*g)/M               0 -(m2*g)/M            0;
   0 0      0                  1       0              0;
   0 0 -((M+m1)*g)/(M*l1)      0 -(m2*g)/(M*l1)       0;
   0 0      0                  0       0              1;
   0 0 -(m1*g)/(M*l2)          0 -(g*(M+m2))/(M*l2)   0];
B=[0; 1/M; 0; 1/(M*l1); 0; 1/(M*l2)]; %Initializing the Linearized B
    matrix
C1 = [1 0 0 0 0 0]; %Case 1: Only x is observed
C2 = [0 0 1 0 0 0; 0 0 0 0 1 0]; %Case 2: theta1 and theta2 are observed.
C3 = [1 0 0 0 0 0; 0 0 0 0 1 0]; % Case 3: x and theta2 are observed.
C4 = [1 0 0 0 0 0; 0 0 1 0 0 0; 0 0 0 0 1 0]; %Case 4: x,theta1 and
    theta2 are observed.
%Individual matrix to check each Observability Condition
Observability1 = obsv(A,C1);
Observability2 = obsv(A,C2);
Observability3 = obsv(A,C3);
Observability4 = obsv(A,C4);
rankArray = [rank(Observability1),rank(Observability2),rank(
    Observability3),rank(Observability4)];
disp("Matrix A of Linearized model is: "); disp(A);
disp("Matrix B of Linearized model is: "); disp(B);
disp("Rank of each Observability matrix");
disp(rankArray);
```

```
% Output of the above script is as below:

Matrix A of Linearized model is:
         0    1.0000         0         0         0         0
         0         0   -0.9810         0   -0.9810         0
         0         0         0    1.0000         0         0
         0         0   -0.5395         0   -0.0491         0
         0         0         0         0         0    1.0000
         0         0   -0.0981         0   -1.0791         0

Matrix B of Linearized model is:
   1.0e-03 *

         0
    1.0000
         0
```

```
    0.0500
         0
    0.1000

Rank of each Observability matrix
     6     4     6     6
```

Here it is can be seen that the rank of the Observability Matrix is full rank for the 1st, 3rd and 4th conditions as per question. In case where only the states of θ_1 and θ_2 is observed, the system is not observable.

7 Luenberger Observer Calculation

Luenberger observer is one of the observers and we will use in this project to analyze the system. The equations is given as:

$$\dot{\hat{X}} = A\hat{X}(t) + B_k U_k(t) + L(Y(t) - C\hat{X}(t)$$

;

$$\hat{X}(0) = 0$$

where L : Observer Gain matrix; Correction term is : $Y(t) - C\hat{X}(t)$

The estimation error $X_e(t) = X(t) - \hat{X}(t)$

has the

$\dot{\hat{X}}_e(t) = \dot{X}(t) - \dot{\hat{X}}(t) = AX_e(t) - L(Y(t) - C\hat{t}) + B_D U_D$

The matrix $A - LC$ is stable iff $(A - LC)^T = A^T - C^T L^T$ is stable.

Using this above concepts to observe the states of our developed model.

Final variation of $\dot{X}_e = (A - LC)X_e(t) + B_D U_D(t)$

Below MATLAB Script shows the final output of the implementation of the developed Linear Model. First we analyze the linear model.

7.1 Luenberger Linear

```
M=1000;%Mass of the Crane
m_1=100;%mass of Load 1
m_2=100;%mass of Load 2
l_1=20;%length of the string of Load 1
l_2=10;%length of the string of Load 2
g=9.81; %declaring the value of the accelertaion due to gravity in m/s^2
%Defining our matrices as follows
A=[0 1 0 0 0 0;
    0 0 -(m_1*g)/M 0 -(m_2*g)/M 0;
    0 0 0 1 0 0;
    0 0 -((M+m_1)*g)/(M*l_1) 0 -(m_2*g)/(M*l_1) 0;
    0 0 0 0 0 1;
    0 0 -(m_1*g)/(M*l_2) 0 -(g*(M+m_2))/(M*l_2) 0];
B=[0; 1/M; 0; 1/(M*l_1); 0; 1/(M*l_2)];
% From previous case, we have determined that only C1, C3 and C4 were
% observable. Hence, we are going to consider only those 3 cases.
C_1 = [1 0 0 0 0 0]; %Corresponding to x component
C_3 = [1 0 0 0 0 0; 0 0 0 0 0 1 0]; %cooresponding to x and theat2
C_4 = [1 0 0 0 0 0; 0 0 1 0 0 0; 0 0 0 0 0 1 0]; %cooresponding to x,
    theta1 and theat2
```

```
D = 0;%declarring the D matrix to be zero
% Cosidering the same Q and R matrices chosen before in our code
Q=[1 0 0 0 0 0;
    0 10 0 0 0 0;
    0 0 1000 0 0 0;
    0 0 0 10 0 0;
    0 0 0 0 1000 0;
    0 0 0 0 0 10];
R=0.01; %%these are the cost variables from LQR
% Initial Conditions for Leunberger observer - 12 state variables,
% 6 actual + 6 estimates
x0=[0,0,30,0,60,0,0,0,0,0,0,0];
% state variables order = [x,dx,theta_1,dthetsa_1,theta_2,dtheta_2,
% estimates taken in the same order]
% For pole placement, lets choose eigen values with negative real part
poles=[-1;-2;-3;-4;-5;-6];
% Calling LQR function to obtain K matrix
K=lqr(A,B,Q,R);
% Framing L for all three cases where output is observable
% Using the pole placement funciton built into MATLAB
L_1 = place(A',C_1',poles)' %L_1 should be a 6x1 matrix
L_3 = place(A',C_3',poles)' %L_3 should be a 6x2 matrix
L_4 = place(A',C_4',poles)' %L_4 should be a 6x3 matrix

A_c1 = [(A-B*K) B*K; % Luenberger A matrix
        zeros(size(A)) (A-L_1*C_1)];
B_c = [B;zeros(size(B))];% Luenberger B matrix
C_c1 = [C_1 zeros(size(C_1))];% Luenberger C matrix

A_c3 = [(A-B*K) B*K;% Luenberger A matrix
        zeros(size(A)) (A-L_3*C_3)];
C_c3 = [C_3 zeros(size(C_3))];% Luenberger C matrix

A_c4 = [(A-B*K) B*K;% Luenberger A matrix
        zeros(size(A)) (A-L_4*C_4)];
C_c4 = [C_4 zeros(size(C_4))];% Luenberger C matrix

sys_1 = ss(A_c1, B_c, C_c1,D);%%MATLAB function to output statespace
    equations
figure % to launch a new figure WINDOW
initial(sys_1,x0)%MATLAB function to check the initial response of the
    system
figure
step(sys_1)%Gives the step response output

sys_3 = ss(A_c3, B_c, C_c3,D);%MATLAB function to output statespace
    equations
figure
initial(sys_3,x0)%MATLAB inbuilt function to check the initial response
    of the system
figure
step(sys_3)%Gives the step response output
```

```
68
69 sys_4 = ss(A_c4, B_c, C_c4, D);%%MATLAB function to output statespace
       equations
70 figure
71 initial(sys_4,x0)%MATLAB inbuilt function to check the initial response
       of the system
72 figure
73 step(sys_4)%Gives the step response output
74
75 %grid on
```

The Out Put Responses for the linear implementation of Luenberger is

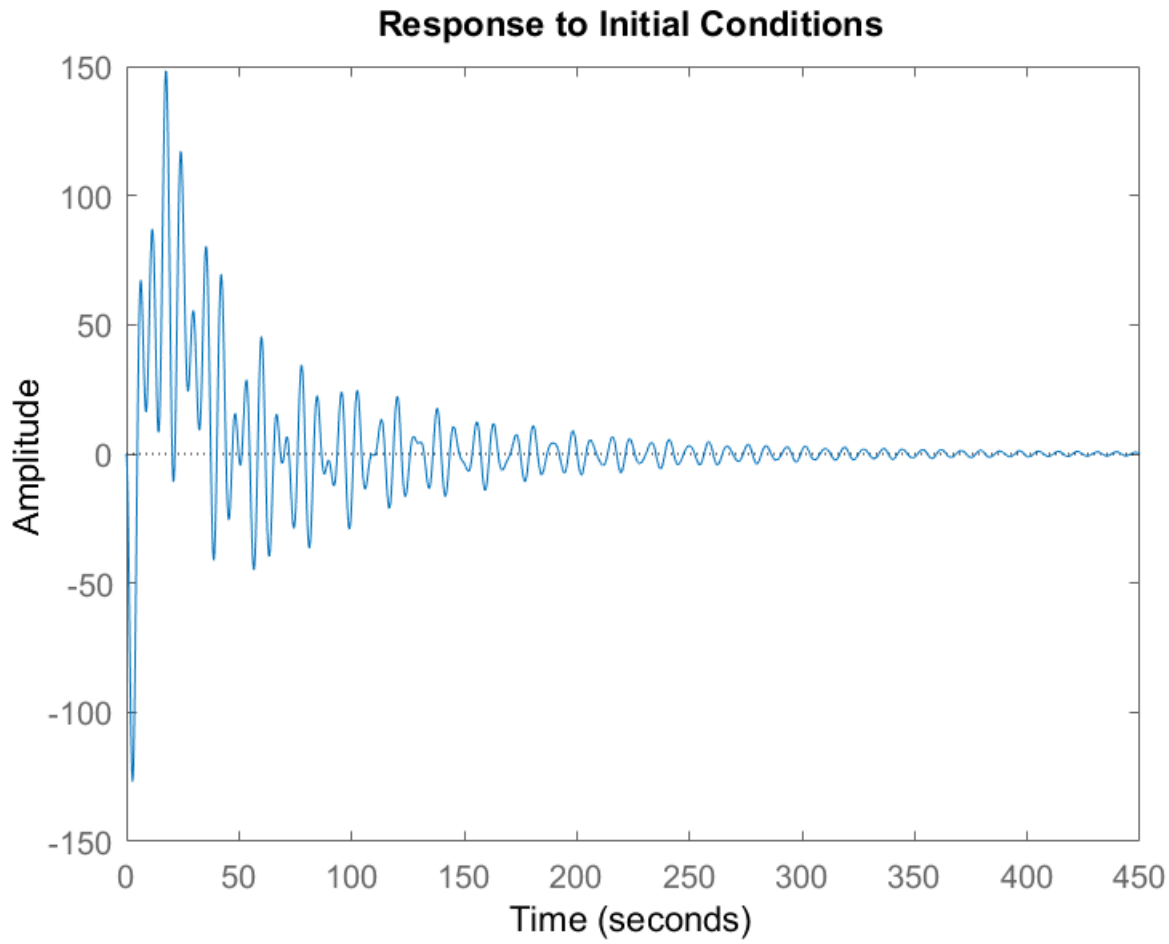

Figure 4: Luenberger linear case 1

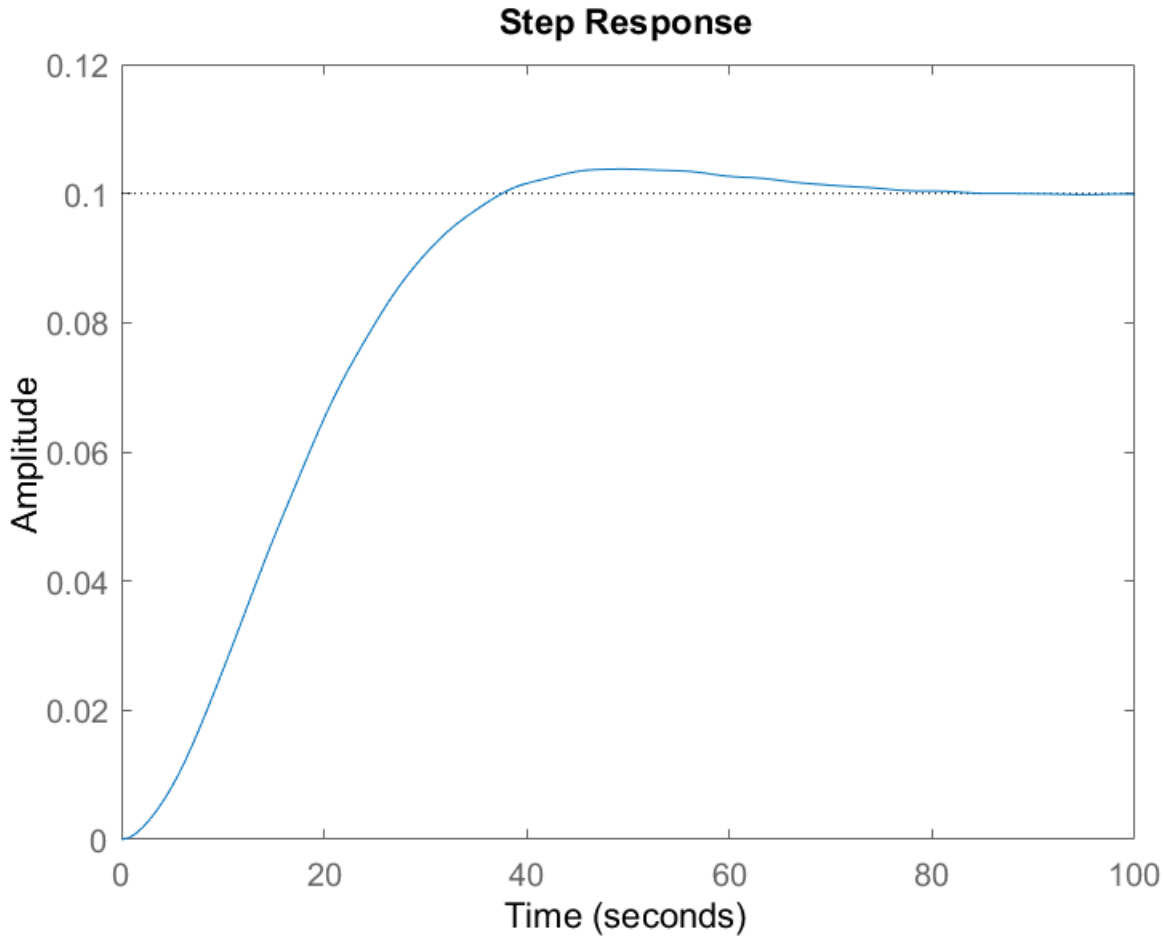

Figure 5: Luenberger linear step case 1

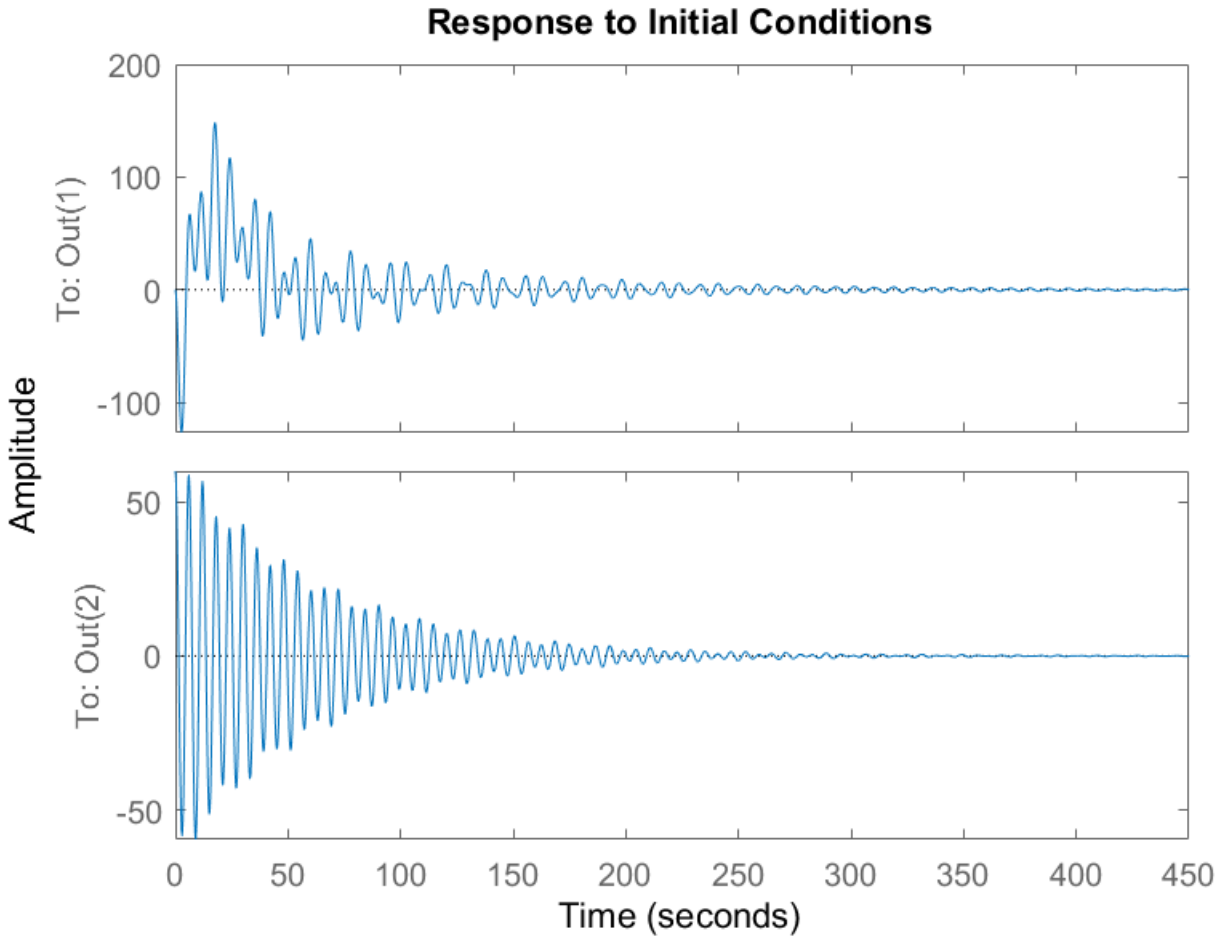

Figure 6: Luenberger linear case 2

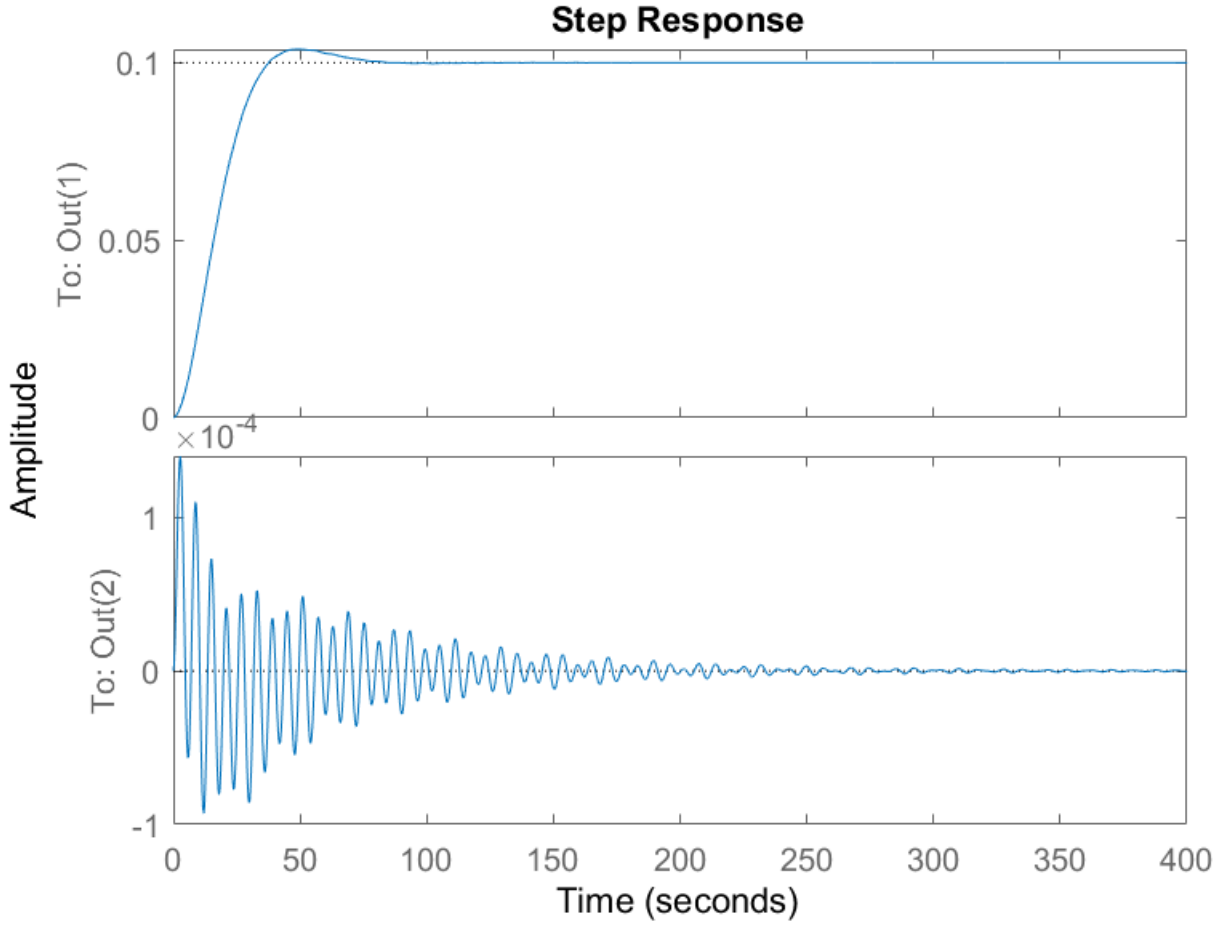

Figure 7: Luenberger linear step case 2

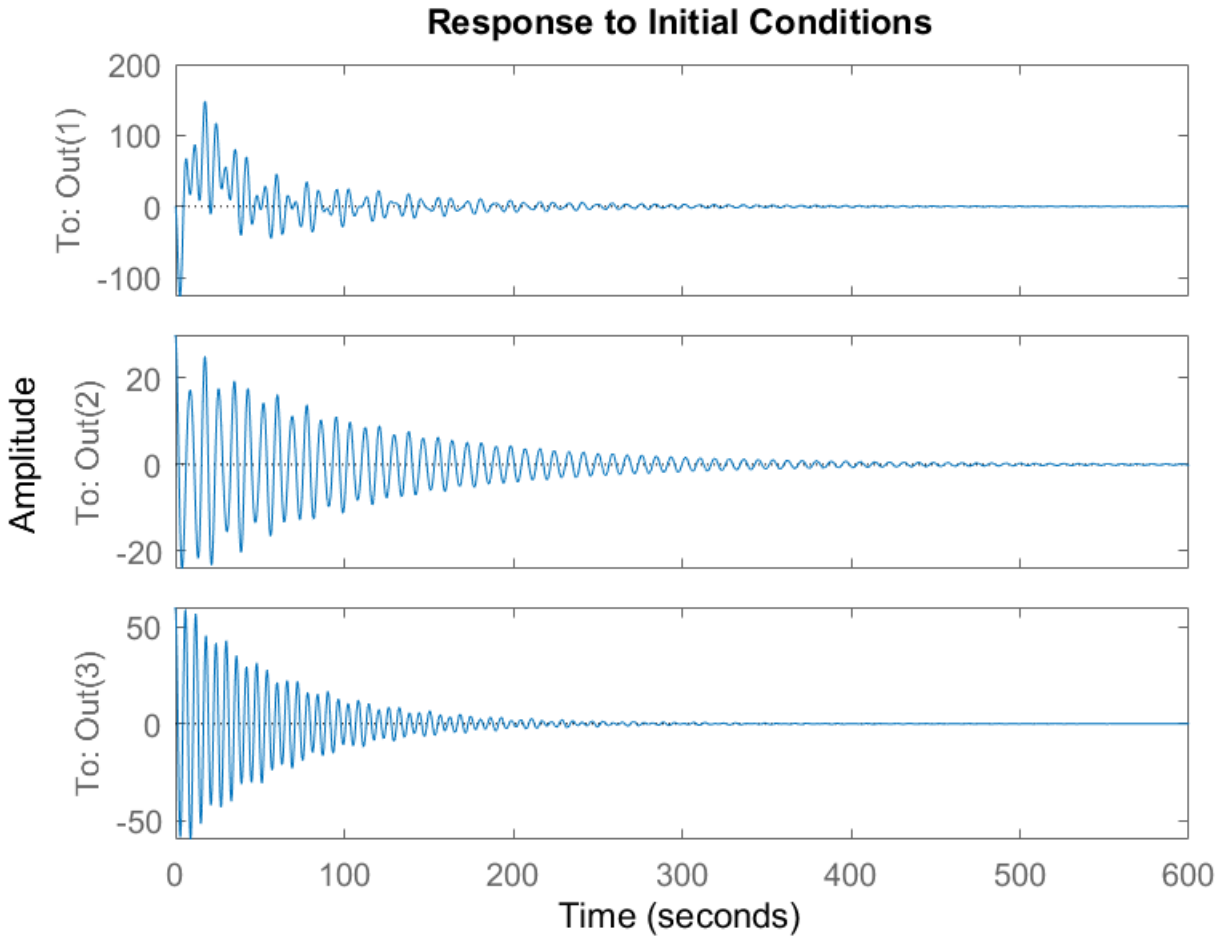

Figure 8: Luenberger linear case 3

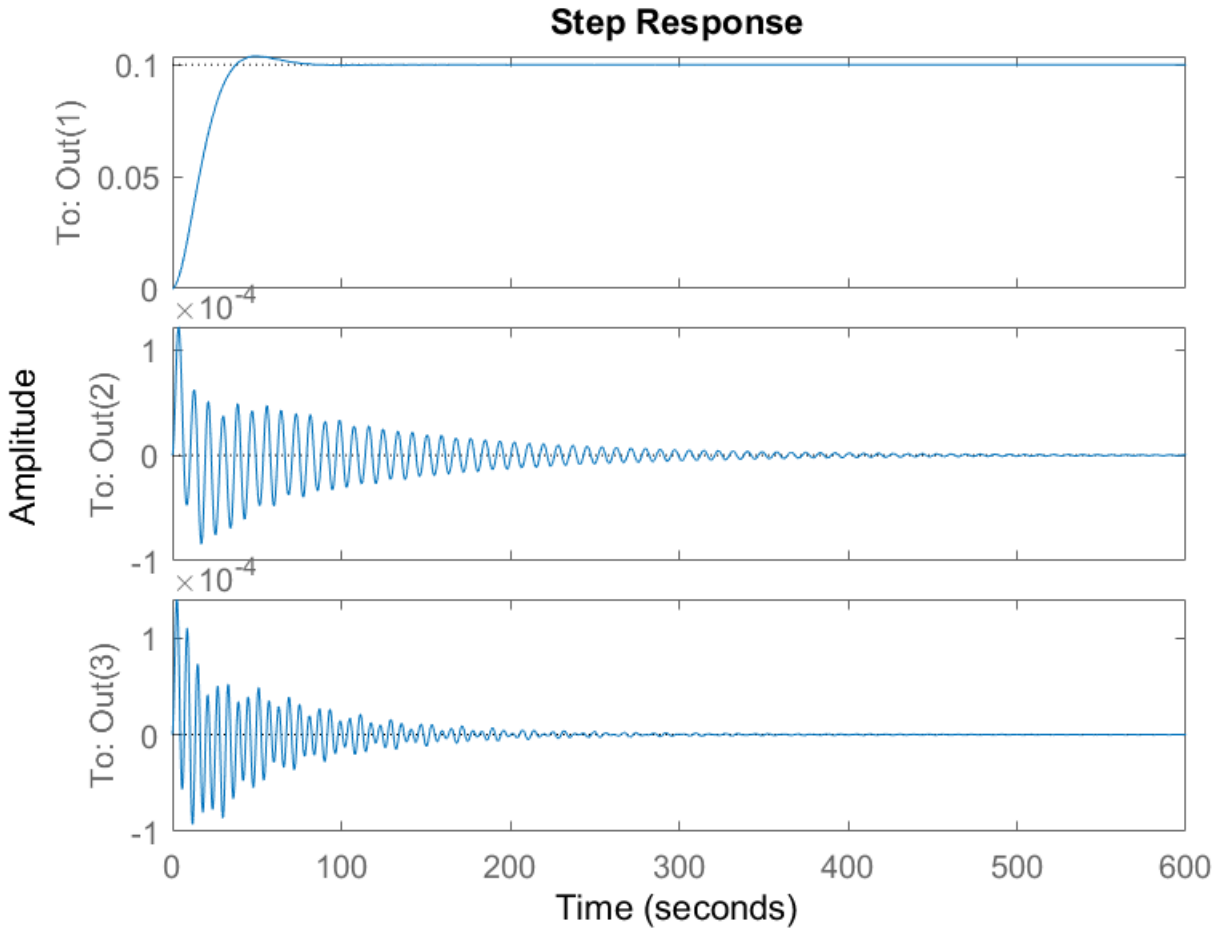

Figure 9: Luenberger linear step case 3

7.2 Luenberger Non Linear Analysis

Analysis of the Luenberger observer applied to the non linear model of the system is simulated using the code as shown below.

```
%clearing all the previous outputs
clc
clear
%Substituting values for our M, m1, m2, l1 and l2
M=1000;%Mass of the cart
m_1=100;%mass of Pendulum 1
m_2=100;%mass of Pendulum 2
l_1=20;%length of the string of Pendulum 1
l_2=10;%length of the string of Pendulum 2
g=9.81; %declaring the value of the accelertaion due to gravity in m/s^2
%declaring the state space matrices
A=[0 1 0 0 0 0;
```

```
13   0  0  -(m_1*g)/M  0  -(m_2*g)/M  0;
14   0  0  0  1  0  0;
15   0  0  -((M+m_1)*g)/(M*l_1)  0  -(m_2*g)/(M*l_1)  0;
16   0  0  0  0  0  1;
17   0  0  -(m_1*g)/(M*l_2)  0  -(g*(M+m_2))/(M*l_2)  0];
18  B=[0;  1/M;  0;  1/(M*l_1);  0;  1/(M*l_2)];
19  % We have previously found out only C1, C3 and C4 are observable.
20  % Hence we will only use C1, C3 and C4.
21
22  C_1 = [1  0  0  0  0  0]; %the output measurment for x component
23  C_3 = [1  0  0  0  0  0;  0  0  0  0  1  0]; %the output measurment for x and theat2
24  C_4 = [1  0  0  0  0  0;  0  0  1  0  0  0;  0  0  0  0  1  0]; %the output measurment for
        x, theta1 and theat2
25  D = 0;%declarring the D matrix to be zero
26
27
28  % declaring the same Q and R matrices
29  Q=[1  0  0  0  0  0;
30    0  10  0  0  0  0;
31    0  0  1000  0  0  0;
32    0  0  0  10  0  0;
33    0  0  0  0  1000  0;
34    0  0  0  0  0  10];
35  R=0.01; %%these are the cost variables from LQR
36
37  % Initial Conditions for Leunberger observer - 12 state variables,
38  % considering 6 actual + 6 estimates
39  x0=[0,0,30,0,60,0,0,0,0,0,0,0];
40  % state variables order = [x,dx,theta_1,dtheta_1,theta_2,dtheta_2,
        estimates taken in the same order]
41  % For pole placement, lets choose eigen values with negative real part
42  poles=[-1;-2;-3;-4;-5;-6];
43  % Calling LQR function to obtain K matrix
44  K=lqr(A,B,Q,R);
45
46  % Framing L for all three cases where output is observable
47  % Using the pole placement funciton built into MATLAB
48  L_1 = place(A',C_1',poles)'; %L1 should be a 6x1 matrix
49  L_3 = place(A',C_3',poles)'; %L3 should be a 6x2 matrix
50  L_4 = place(A',C_4',poles)'; %L4 should be a 6x3 matrix
51  A_c1 = [(A-B*K)  B*K;  zeros(size(A))  (A-L_1*C_1)];% Luenberger A matrix
52  B_c = [B;zeros(size(B))];% Luenberger B matrix
53  C_c1 = [C_1  zeros(size(C_1))];% Luenberger C matrix
54  A_c3 = [(A-B*K)  B*K;  zeros(size(A))  (A-L_3*C_3)];% Luenberger A matrix
55
56  C_c3 = [C_3  zeros(size(C_3))];% Luenberger C matrix
57  A_c4 = [(A-B*K)  B*K;  zeros(size(A))  (A-L_4*C_4)];% Luenberger A matrix
58  C_c4 = [C_4  zeros(size(C_4))];% Luenberger C matrix
59
60  sys_1 = ss(A_c1,  B_c,  C_c1,D);%%MATLAB function to output statespace
        equations
61  figure % to launch a new figure WINDOW
```

```
62 initial(sys_1,x0)%MATLAB inbuilt function to check the initial response
       of the system
63
64 figure
65 step(sys_1)%Gives the step response output
66
67 sys_3 = ss(A_c3, B_c, C_c3,D);%MATLAB function to output statespace
       equations
68 figure
69 initial(sys_3,x0)%MATLAB inbuilt function to check the initial response
       of the system
70
71 figure
72 step(sys_3)%Gives the step response output
73
74 sys_4 = ss(A_c4, B_c, C_c4, D);%%MATLAB function to output statespace
       equations
75 figure
76 initial(sys_4,x0)%MATLAB inbuilt function to check the initial response
       of the system
77
78 figure
79 step(sys_4)%Gives the step response output
80 grid on
```

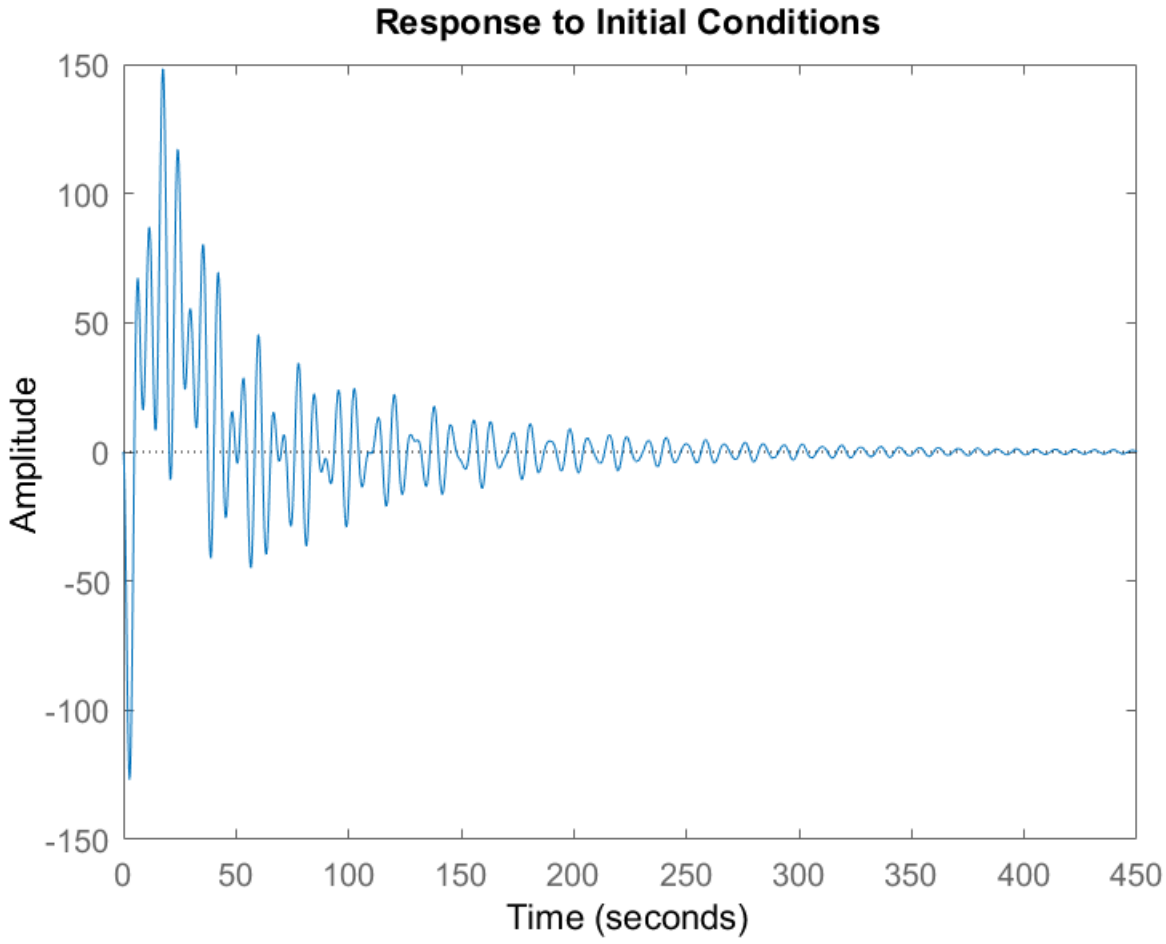

Figure 10: Luenberger non linear Initial response for case 1

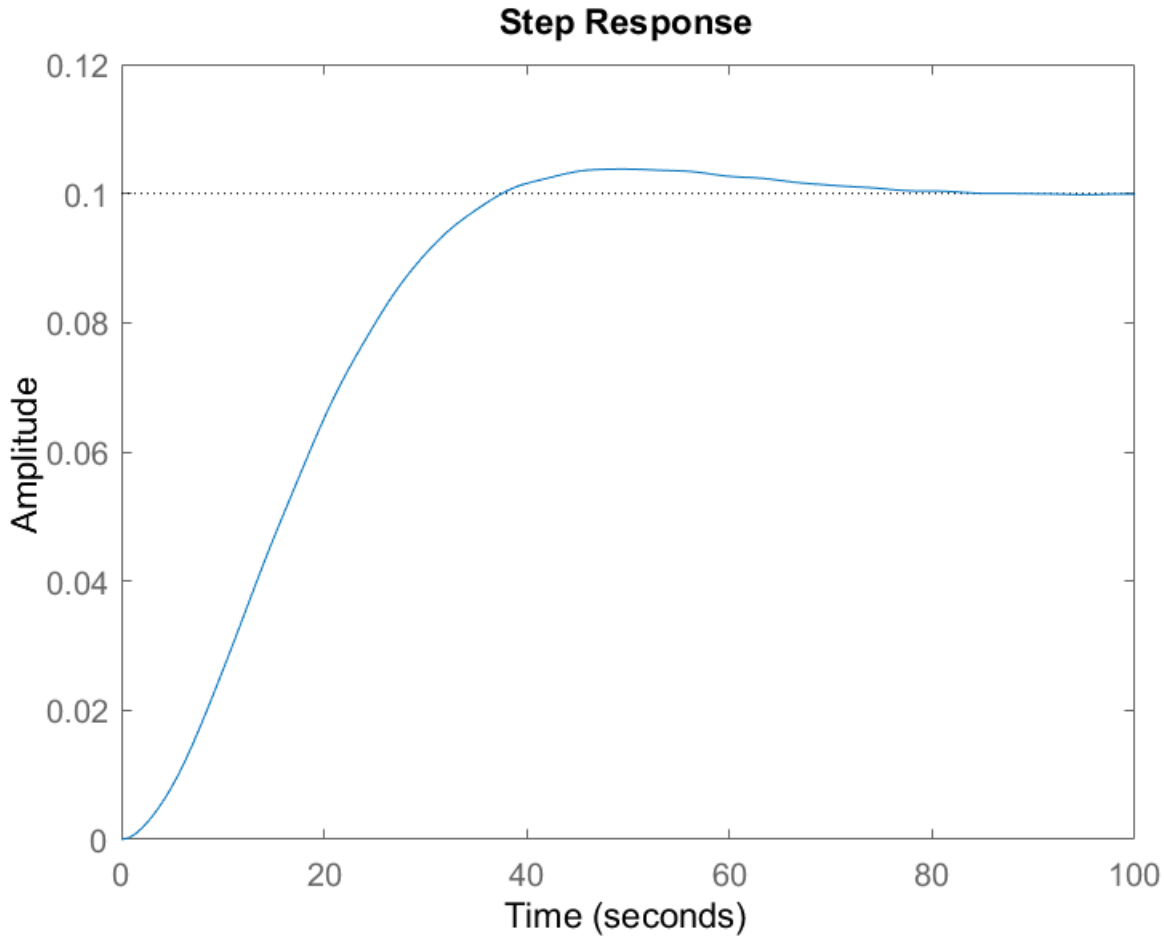

Figure 11: Luenberger non linear Step response for case 1

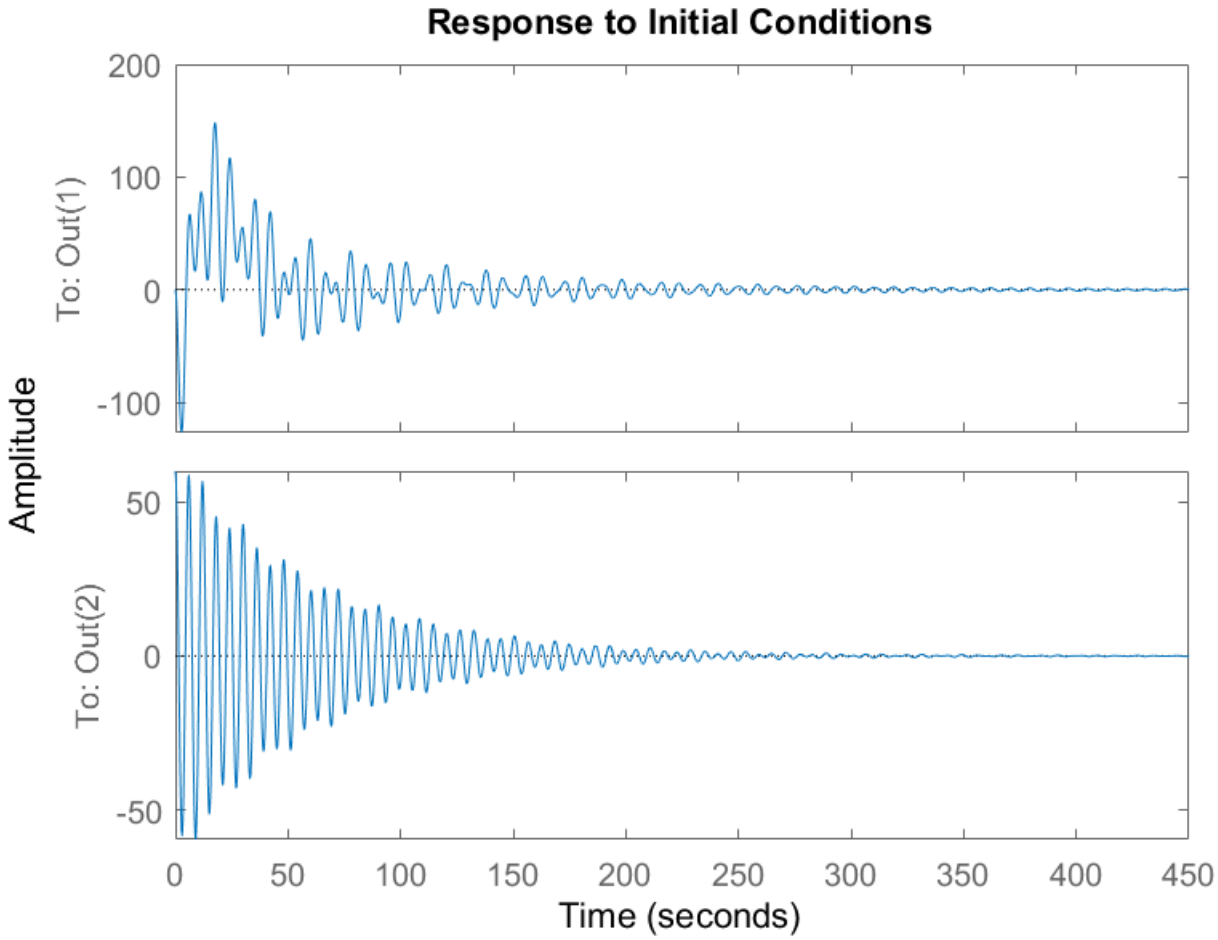

Figure 12: Luenberger non linear Initial response for case 2

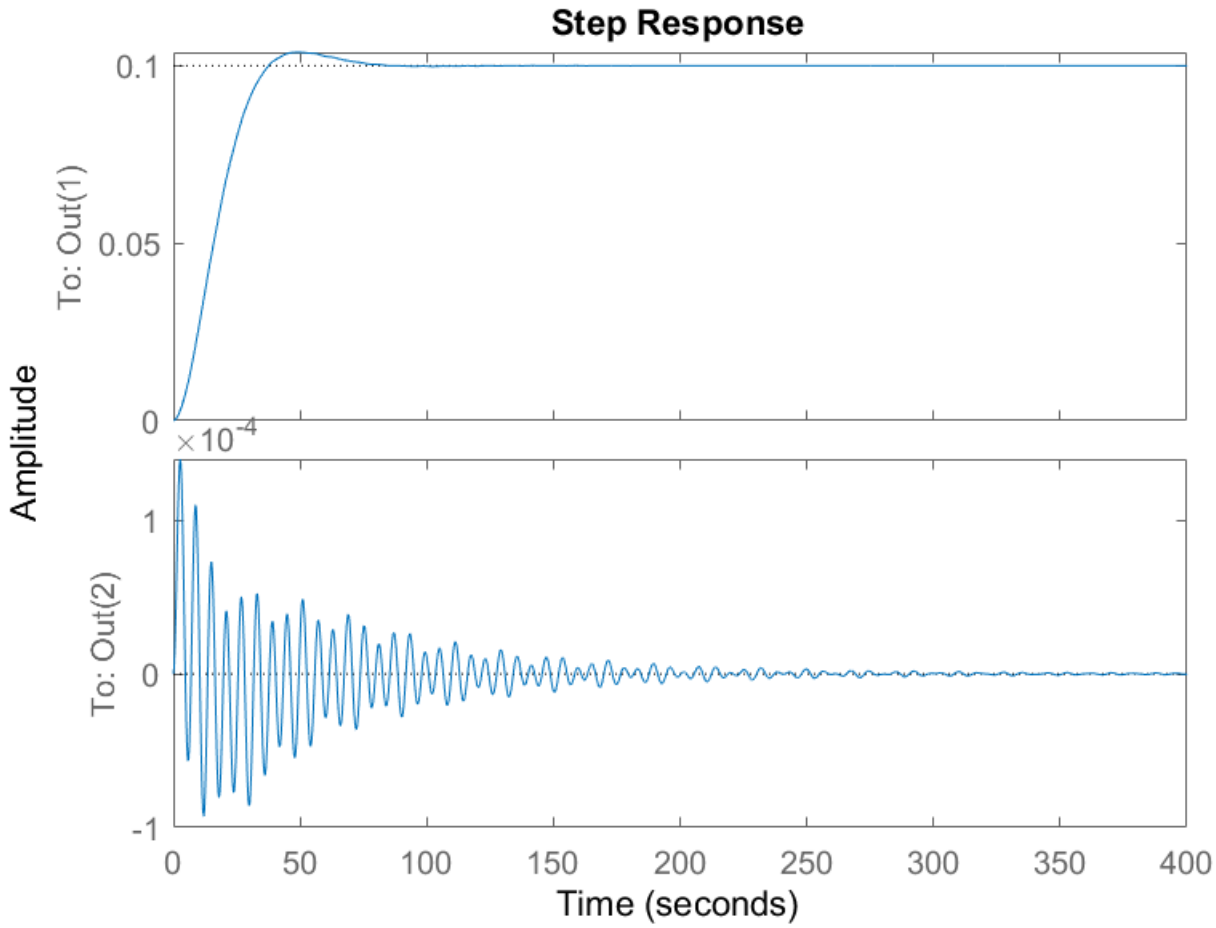

Figure 13: Luenberger non linear Step response for case 2

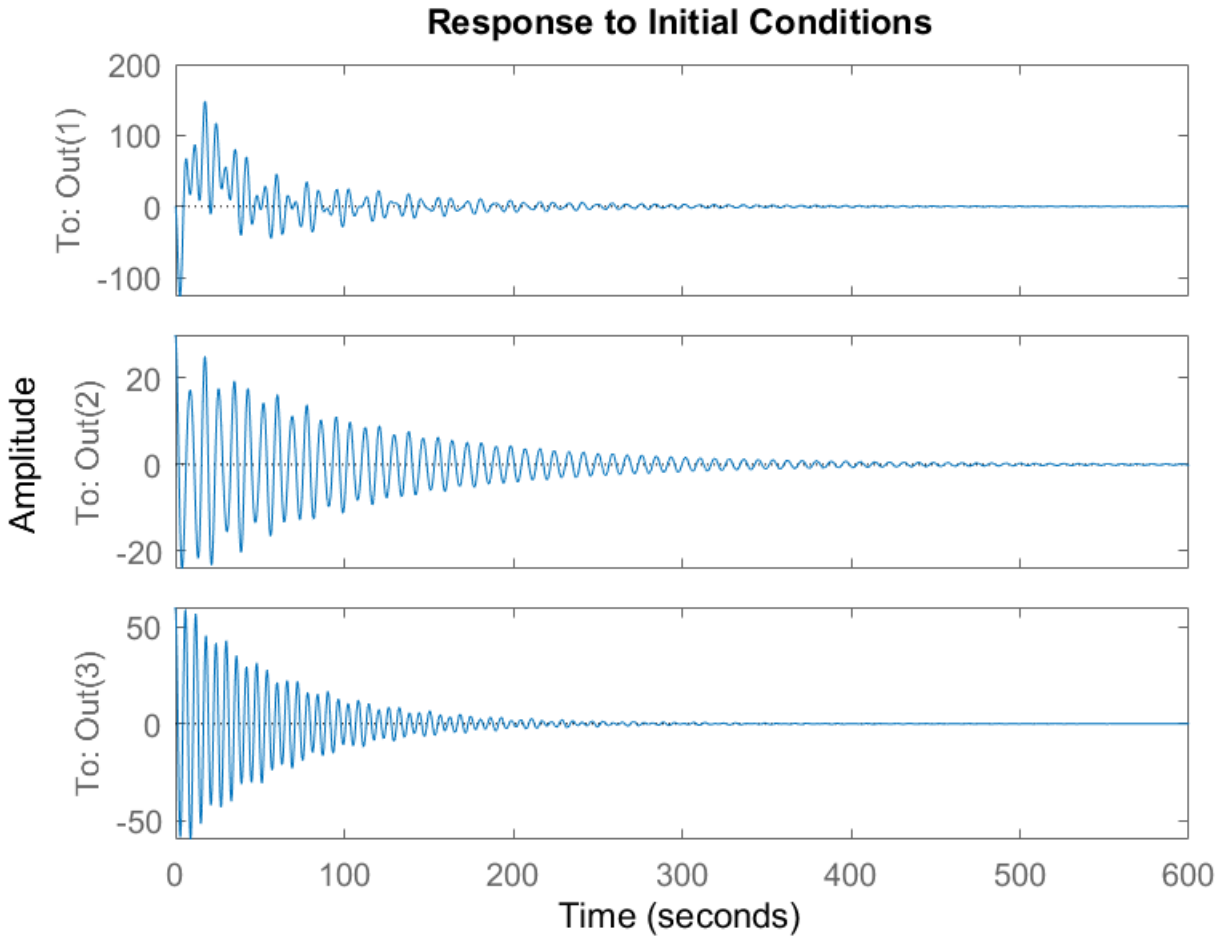

Figure 14: Luenberger non linear Initial response for case 3

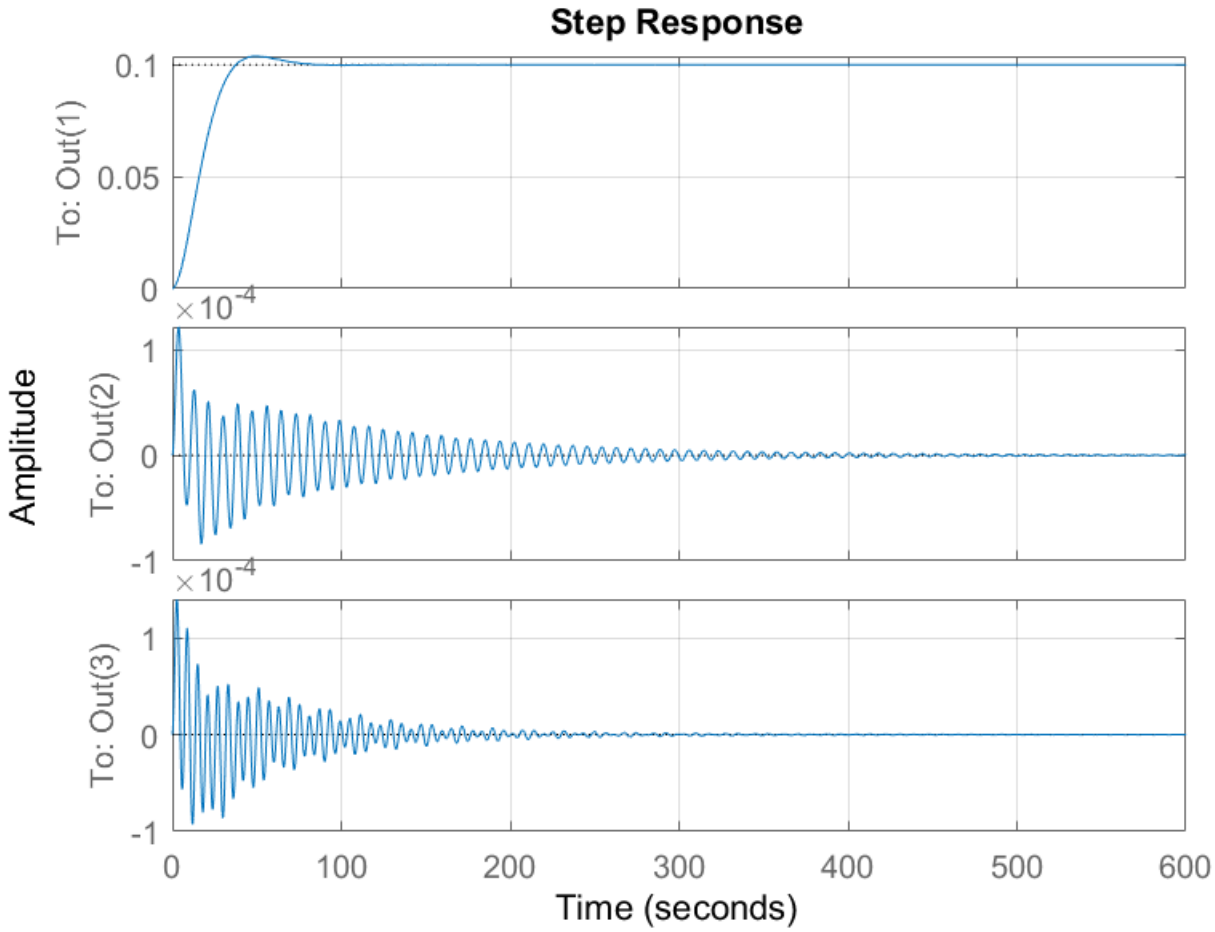

Figure 15: Luenberger non linear Step response for case 3

8 LQG Feedback Controller Analysis

LQG feedback controller works because of the existence of separation principle. It simply states that the states of the optimal control of states of the system can be done using and LQR, and the optimal control of the states of the estimator can be done using Kalman filter, and when combined, the final results is also the optimal solution.

To explain this we can say that the LQR can be used to place the eigen values of the closed loop controller, Kalman filter can be used to place the eigen values of the estimator dynamics and these can be modeled separately. But even when these two systems are combined, the effective result is still the optimal result.

8.1 LQG linear analysis

```
%Substituting values for our M, m1, m2, l1 and l2
M=1000;%Mass of the cart
```

```
m1=100;%mass of Pendulum 1
m2=100;%mass of Pendulum 2
l1=20;%length of the string of Pendulum 1
l2=10;%length of the string of Pendulum 2
g=9.81; %declaring the value of the accelertaion due to gravity in m/s^2
%Defining our matrices as follows
A=[0 1 0 0 0 0;
    0 0 -(m1*g)/M 0 -(m2*g)/M 0;
    0 0 0 1 0 0;
    0 0 -((M+m1)*g)/(M*l1) 0 -(m2*g)/(M*l1) 0;
    0 0 0 0 0 1;
    0 0 -(m1*g)/(M*l2) 0 -(g*(M+m2))/(M*l2) 0];
B=[0; 1/M; 0; 1/(M*l1); 0; 1/(M*l2)];
% Cosidering the same Q and R matrices chosen before in our code
Q=[1 0 0 0 0 0;
   0 10 0 0 0 0;
   0 0 1000 0 0 0;
   0 0 0 10 0 0;
   0 0 0 0 1000 0;
   0 0 0 0 0 10];
R=0.001; %these are the cost variables from LQR
% From previous case, we have determined that only C1, C3 and C4 were
% observable. Hence, we are going to consider only those 3 cases.
C1 = [1 0 0 0 0 0];  %Corresponding to x component
C3 = [1 0 0 0 0 0; 0 0 0 0 1 0]; %corresponding to x and theat2
C4 = [1 0 0 0 0 0; 0 0 1 0 0 0; 0 0 0 0 1 0]; %corresponding to x, theta1
    and theat2
D = 0;
% Initial Conditions for Leunberger observer - 12 state variables,
% 6 actual + 6 estimates
x0 = [4;0;30;0;60;0;0;0;0;0;0;0];
% Calling LQR function to obtain K matrix
K =lqr(A,B,Q,R);
vd=0.3*eye(6); %process noise
vn=1; %measurement noise

K_pop1=lqr(A',C1',vd,vn)'; %gain matrix of kalman filter for C1
K_pop3=lqr(A',C3',vd,vn)'; %gain matrix of kalman filter for C3
K_pop4=lqr(A',C4',vd,vn)'; %gain matrix of kalman filter for C4

% Observing state space corresponding C1 observable system
sys1 = ss([(A-B*K) B*K; zeros(size(A)) (A-K_pop1*C1)], [B;zeros(size(B))
    ],[C1 zeros(size(C1))], D);
figure
initial(sys1,x0)

figure
step(sys1)

t = linspace(0, 10, 1500);                                  % Time Vector
u = sum(sin((1:29)'*2*pi*t/2.5));                           % Arbitrary
    Input
```

```
52
53 figure
54 lsim(sys1,u,t)
55
56 figure
57 plot(t,u)
58
59
60
61
62 % Observing state space corresponding C3 observable system
63 sys3 = ss([(A-B*K) B*K; zeros(size(A)) (A-K_pop3*C3)], [B;zeros(size(B))
       ],[C3 zeros(size(C3))], D);
64 figure
65 initial(sys3,x0)
66
67 figure
68 step(sys3)
69
70
71 % Observing state space corresponding C4 observable system
72 sys4 = ss([(A-B*K) B*K; zeros(size(A)) (A-K_pop4*C4)], [B;zeros(size(B))
       ],[C4 zeros(size(C4))], D);
73 figure
74 initial(sys4,x0)
75 figure
76 step(sys4)
77
78 grid on
```

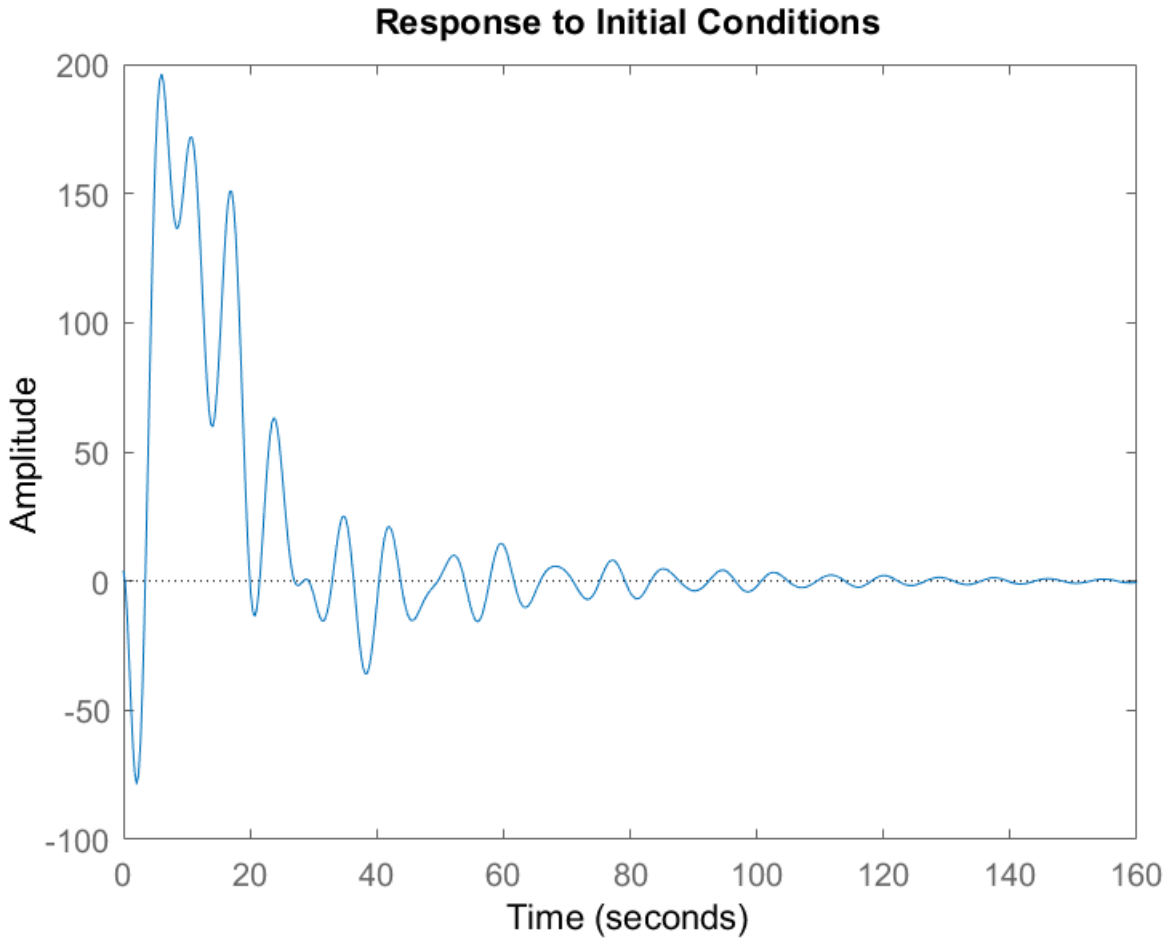

Figure 16: LQG Initial response for Linear State Case 1

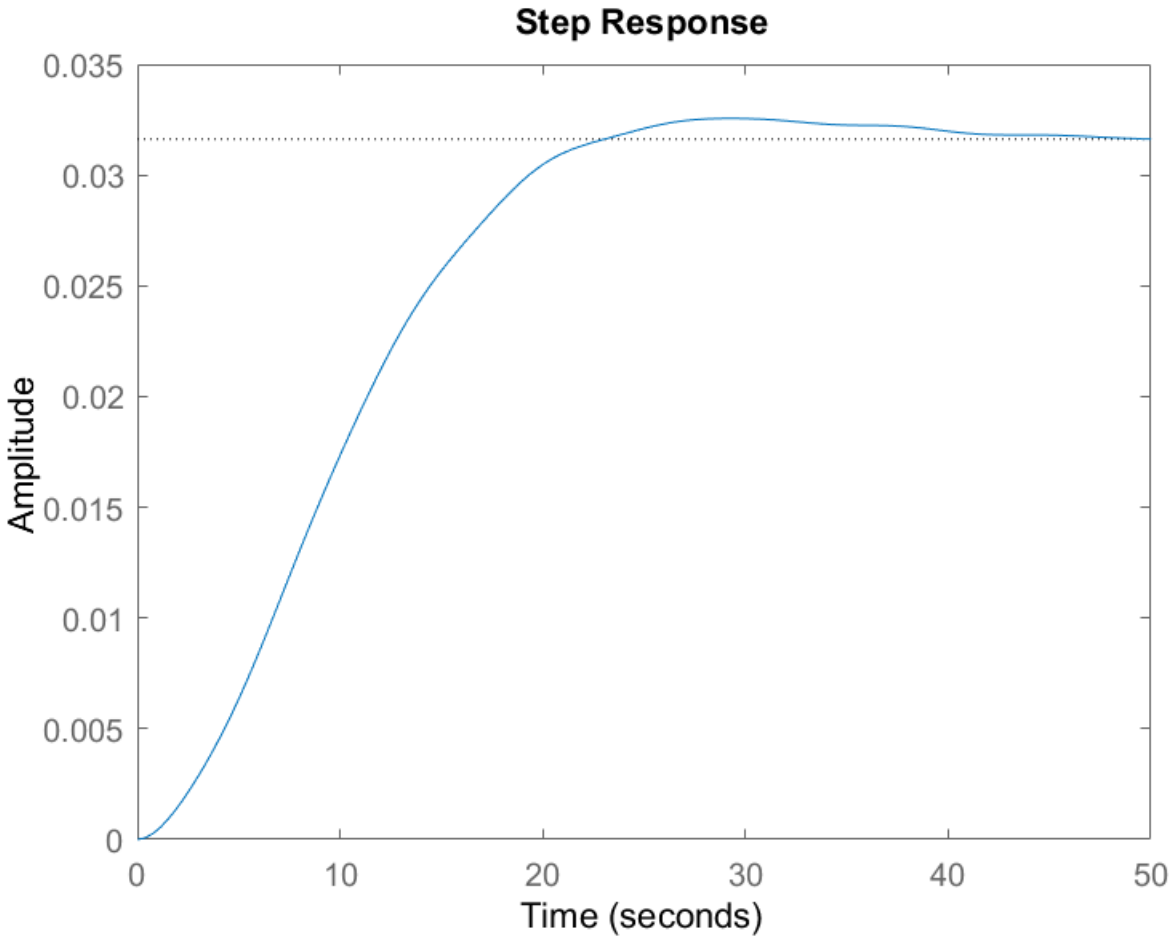

Figure 17: LQG Initial Step response for Linear State Case 1

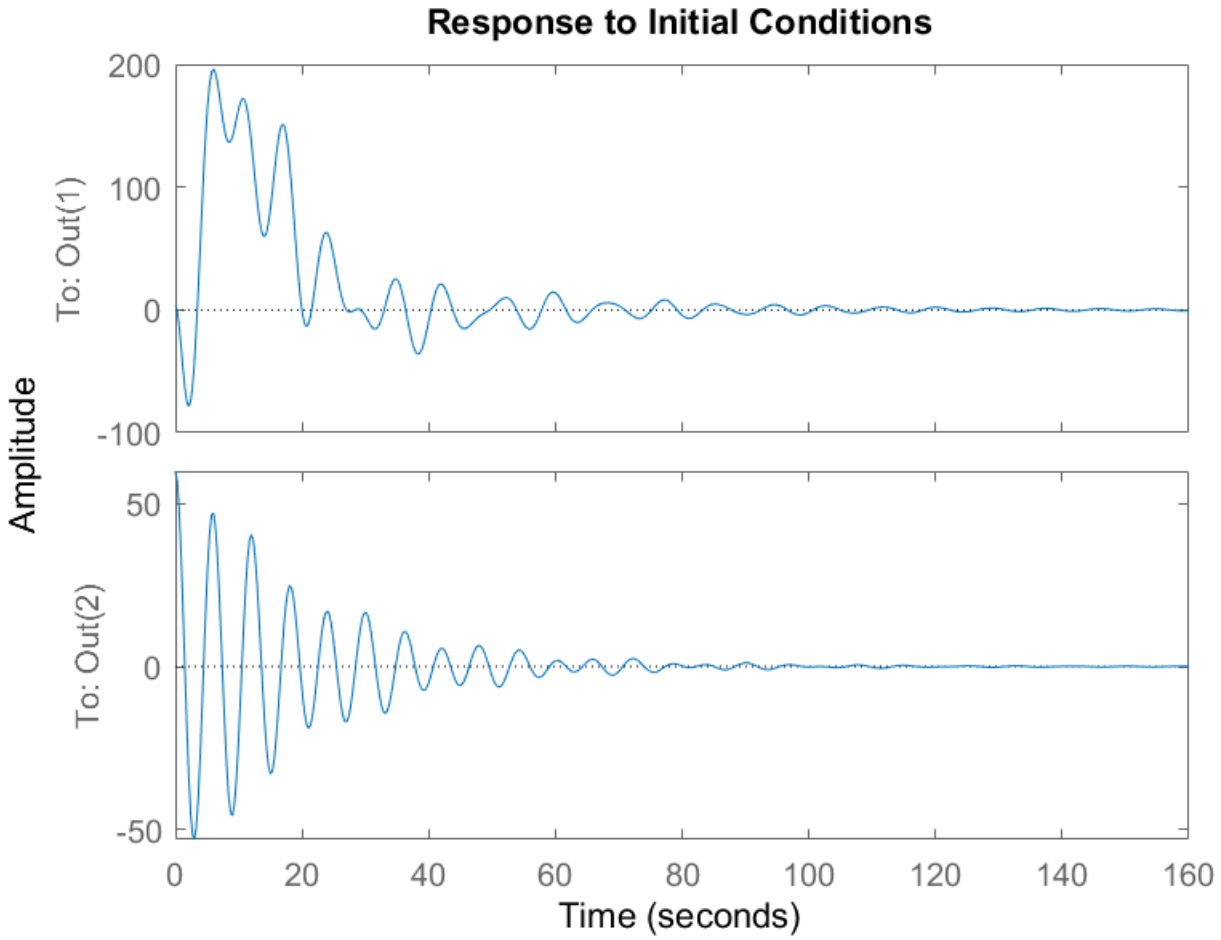

Figure 18: LQG Initial response for Linear State Case 2

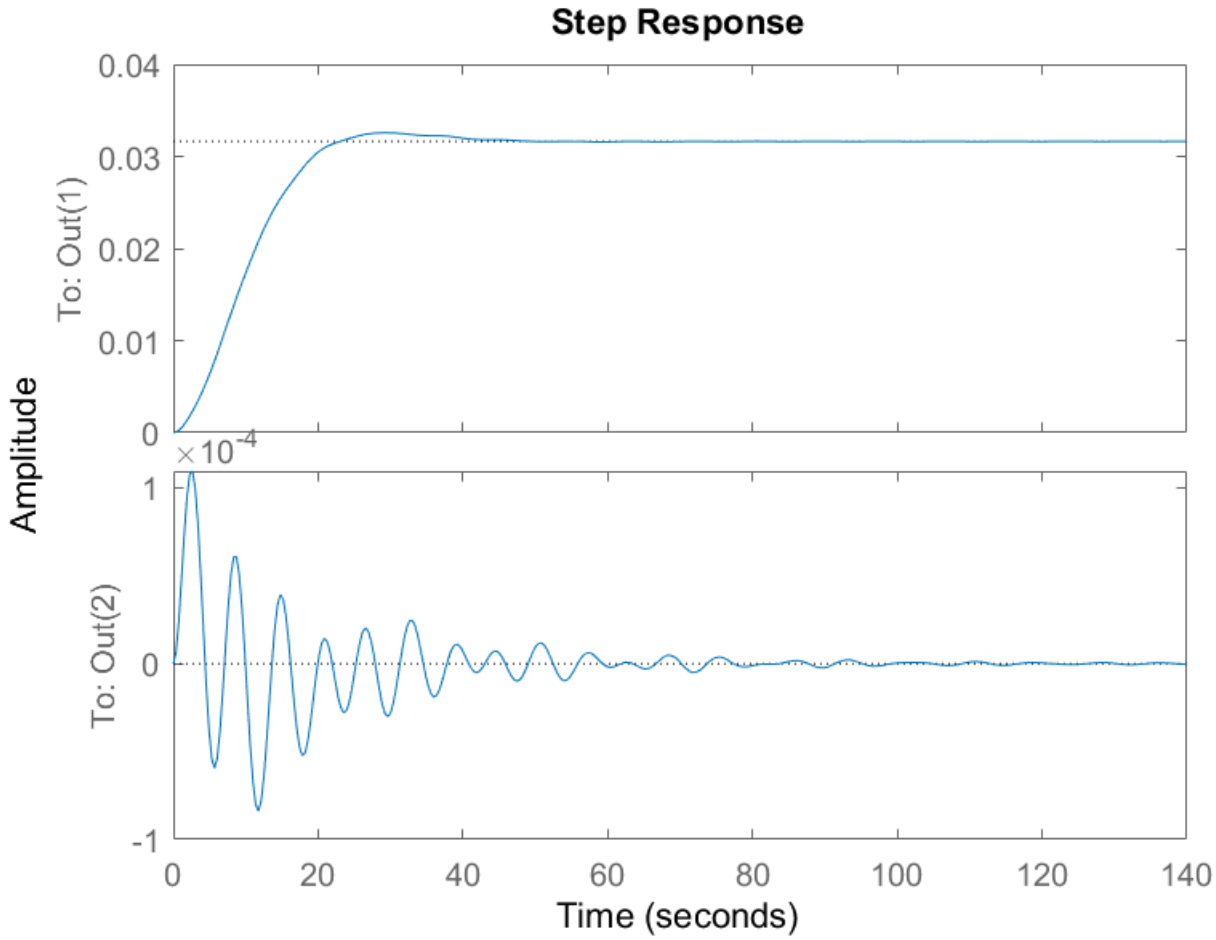

Figure 19: LQG Initial Step response for Linear State Case 2

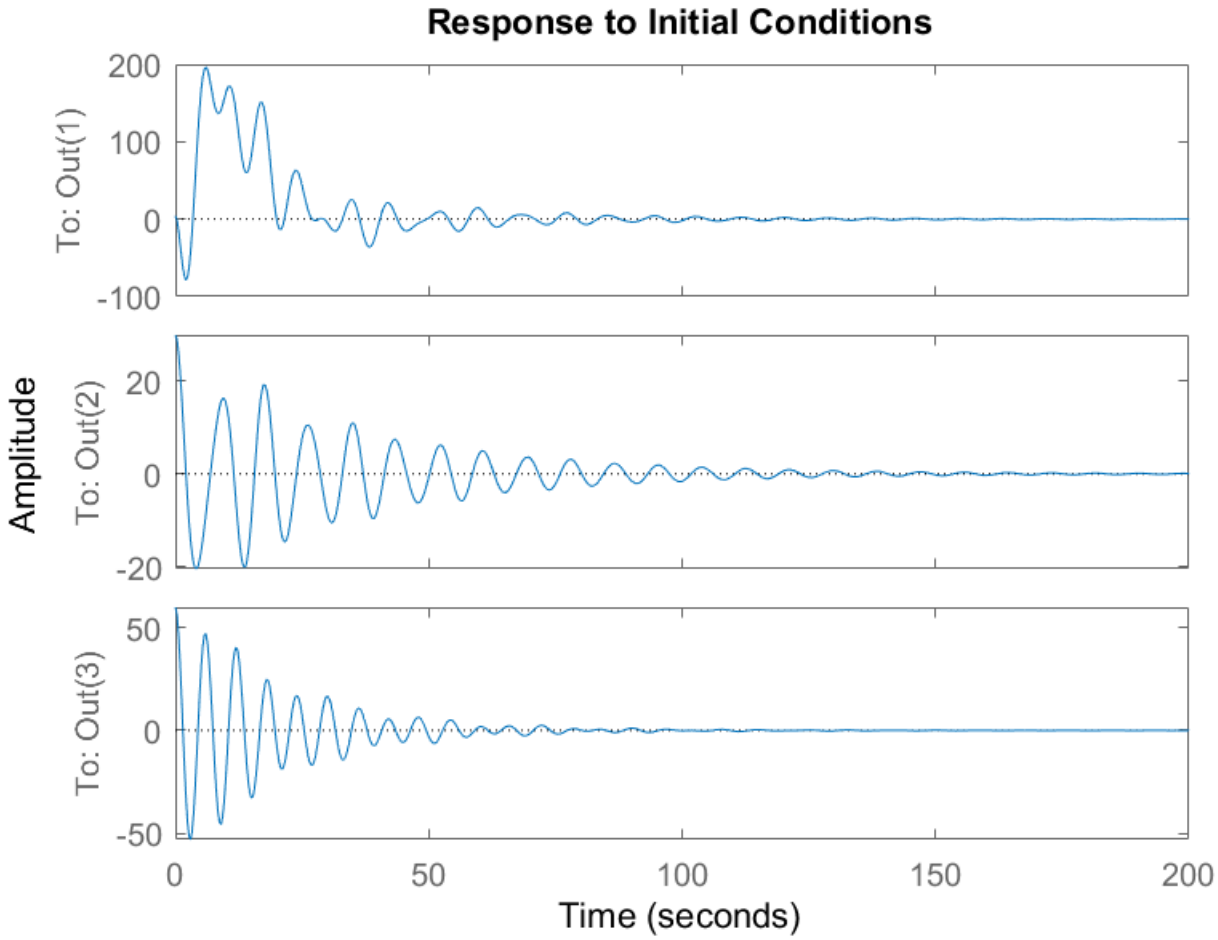

Figure 20: LQG Initial response for Linear State Case 3

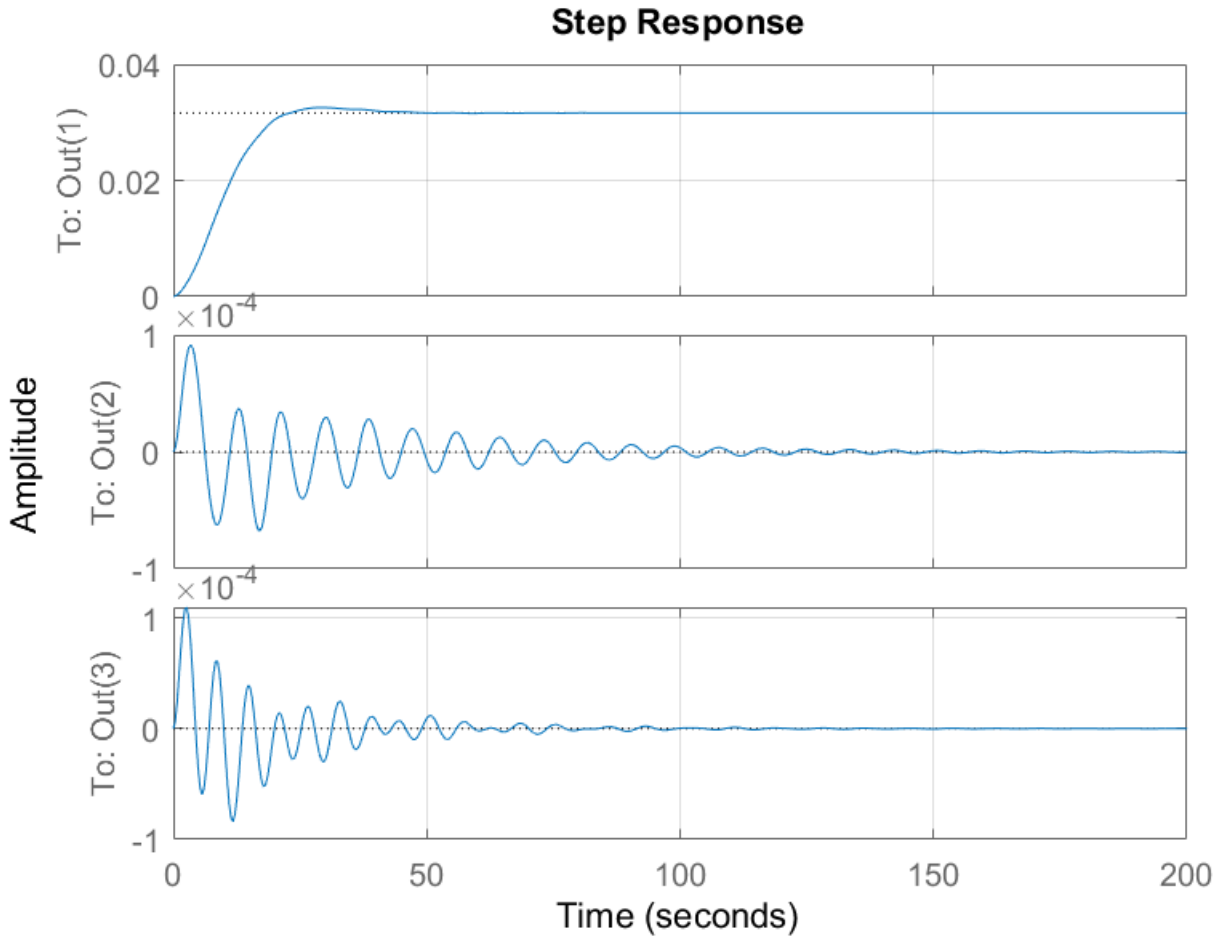

Figure 21: LQG Initial Step response for Linear State Case 3

8.2 LQG Non-linear analysis

```
x_0 = [0;0;30;0;60;0;0;0;0;0;0;0]
% First Observable State
tspan=0:0.1:100;
n_case=1;
[t,x] = ode45(@(tspan,x_0) cart2pend(tspan,x_0,n_case),tspan,x_0);
plot(t,x)
figure
grid on
% Second Observable State
tspan=0:0.1:100;
n_case=2;
[t,x] = ode45(@(tspan,x_0) cart2pend(tspan,x_0,n_case),tspan,x_0);
figure
plot(t,x)
```

```
grid on
% Third Observable State
tspan=0:0.1:100;
n_case=3;
[t,x] = ode45(@(tspan,x_0) cart2pend(tspan,x_0,n_case),tspan,x_0);
figure
plot(t,x)
grid on
function dydt = cart2pend(t,y,n_case)
M=1000;
m1=100;
m2=100;
l1=20;
l2=10;
g=9.81;
% Linearized State Space Dynamic Matrices
%A
A=[0 1 0 0 0 0;
     0 0 -(m1*g)/M 0 -(m2*g)/M 0;
     0 0 0 1 0 0;
     0 0 -((M+m1)*g)/(M*l1) 0 -(m2*g)/(M*l1) 0;
     0 0 0 0 0 1;
     0 0 -(m1*g)/(M*l2) 0 -(g*(M+m2))/(M*l2) 0];
%B
B=[0; 1/M; 0; 1/(M*l1); 0; 1/(M*l2)];
% Q  Gain Matrix for LQG
Q=[1 0 0 0 0 0;
    0 10 0 0 0 0;
    0 0 1000 0 0 0;
    0 0 0 10 0 0;
    0 0 0 0 1000 0;
    0 0 0 0 0 10];
% R Gain Matrix for LQG
R=0.01;

%D
D = 0;

K_val = lqr(A,B,Q,R);
% calculates the optimal gain matrix K, the solution S of the associated
    algebraic Riccati
% equation and the closed-loop poles P using the continuous-time state-
    space matrices A and B.
F=-K_val*y(1:6);

vd=0.3*eye(6);
vn=1;
% From previous case, we have determined that only C1, C3 and C4 were
% observable. Hence, we are going to consider only those 3 cases.
if n_case==1
    C = [1 0 0 0 0 0]; %Corresponding to x component
elseif n_case==2
```

```
65      C = [1 0 0 0 0 0; 0 0 0 0 0 1 0]; %corresponding to x and theat2
66  elseif n_case==3
67      C = [1 0 0 0 0 0; 0 0 1 0 0 0; 0 0 0 0 0 1 0]; %corresponding to x,
            theta1 and theat2
68  end
69  K=lqr(A',C',vd,vn)';
70  sd =(A-K*C)*y(7:12);
71
72  dydt=zeros(12,1);
73  % y(1)=x; y(2)=xdot; y(3)=theta1;    y(4)=theta1dot;
74  % y(5)=theta2;     y(6)=theta2dot;
75  dydt(1) = y(2);%XD;
76  dydt(2)=(F-(g/2)*(m1*sind(2*y(3))+m2*sind(2*y(5)))-(m1*l1*(y(4)^2)*sind(y
        (3)))-(m2*l2*(y(6)^2)*sind(y(5))))/(M+m1*((sind(y(3)))^2)+m2*((sind(y
        (5)))^2));%xDD
77  dydt(3)= y(4);%theta 1D;
78  dydt(4)= (dydt(2)*cosd(y(3))-g*(sind(y(3))))/l1';%theta 1 Ddot;
79  dydt(5)= y(6);%theta 2D
80  dydt(6)= (dydt(2)*cosd(y(5))-g*(sind(y(5))))/l2;%theta 2Ddot;
81  dydt(7)= y(2)-y(10);
82  dydt(8)= dydt(2)-sd(2);
83  dydt(9)= y(4)-y(11);
84  dydt(10)= dydt(4)-sd(4);
85  dydt(11)= y(6)-y(12);
86  dydt(12)= dydt(6)-sd(6);
87  end
```

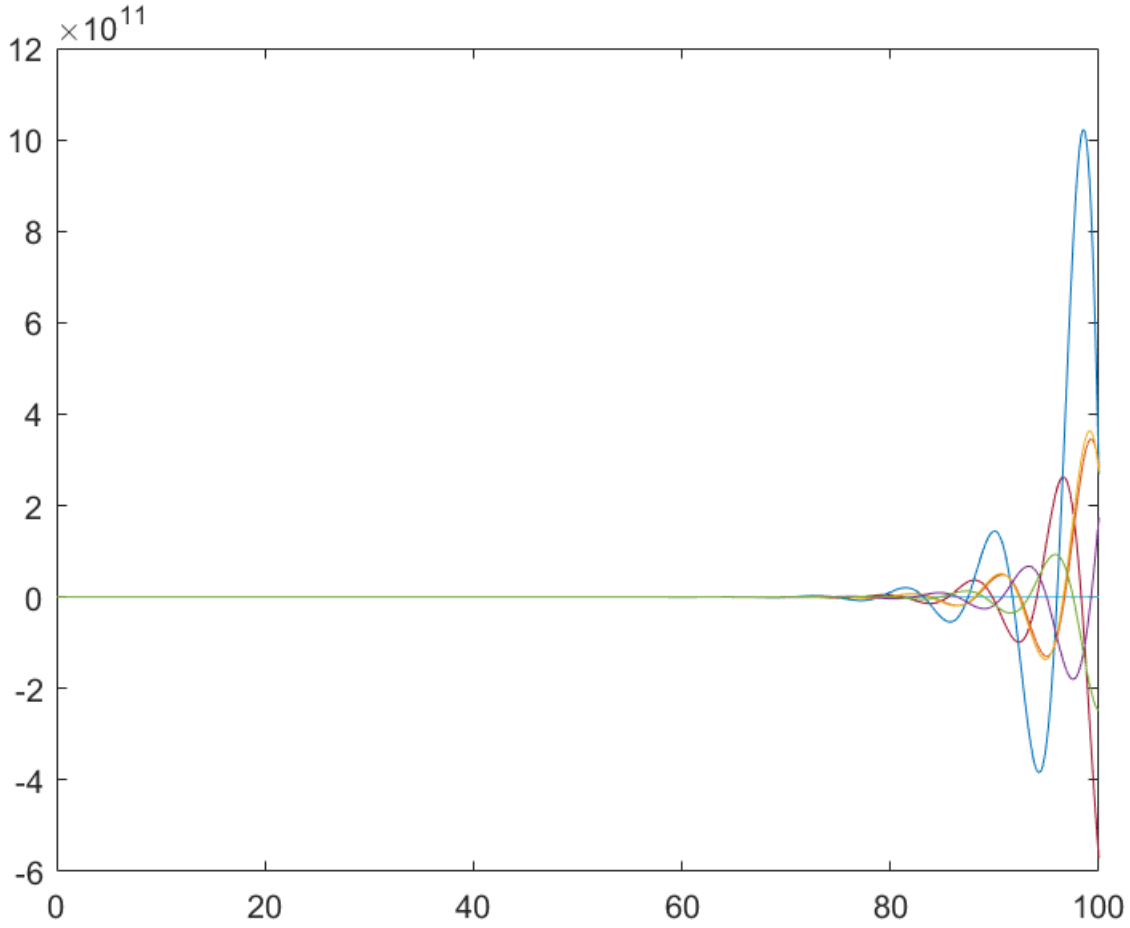

Figure 22: LQG Initial response for Linear State Case 1

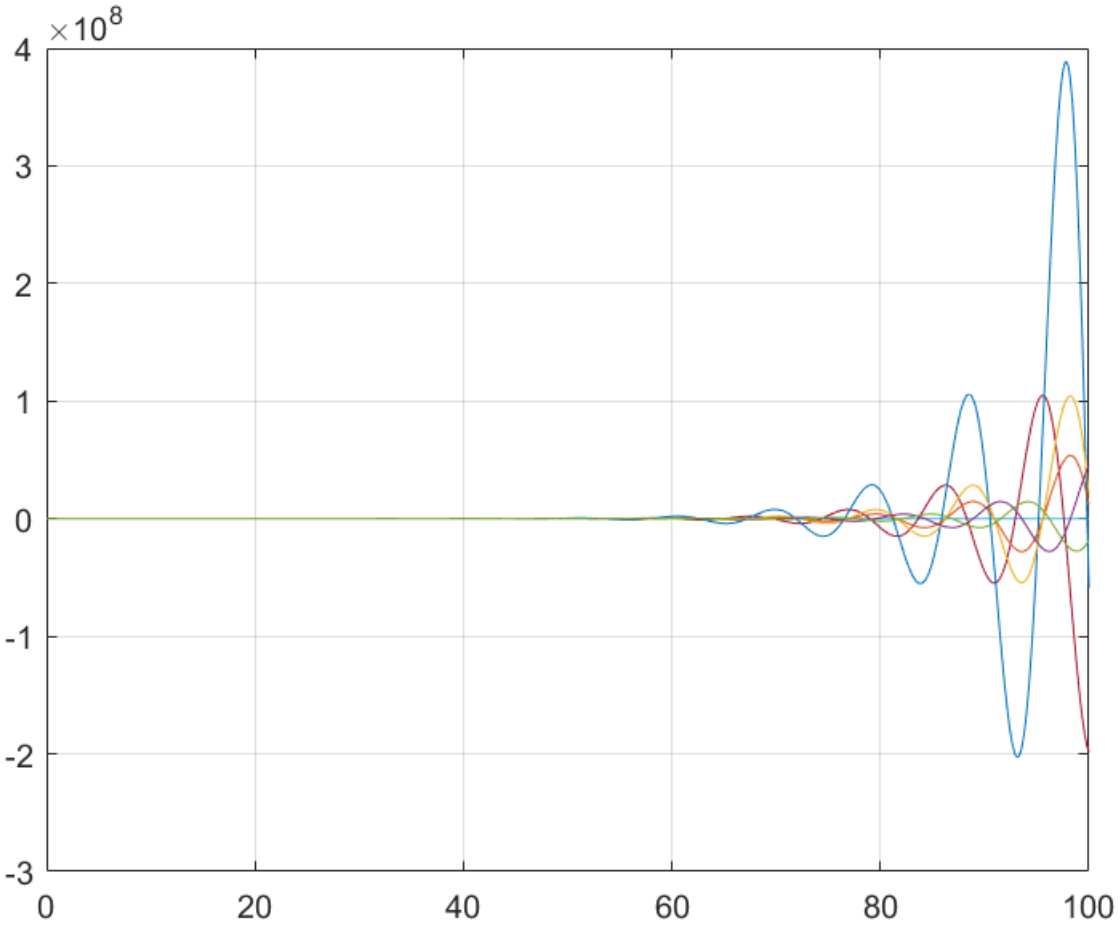

Figure 23: LQG Initial response for Linear State Case 2

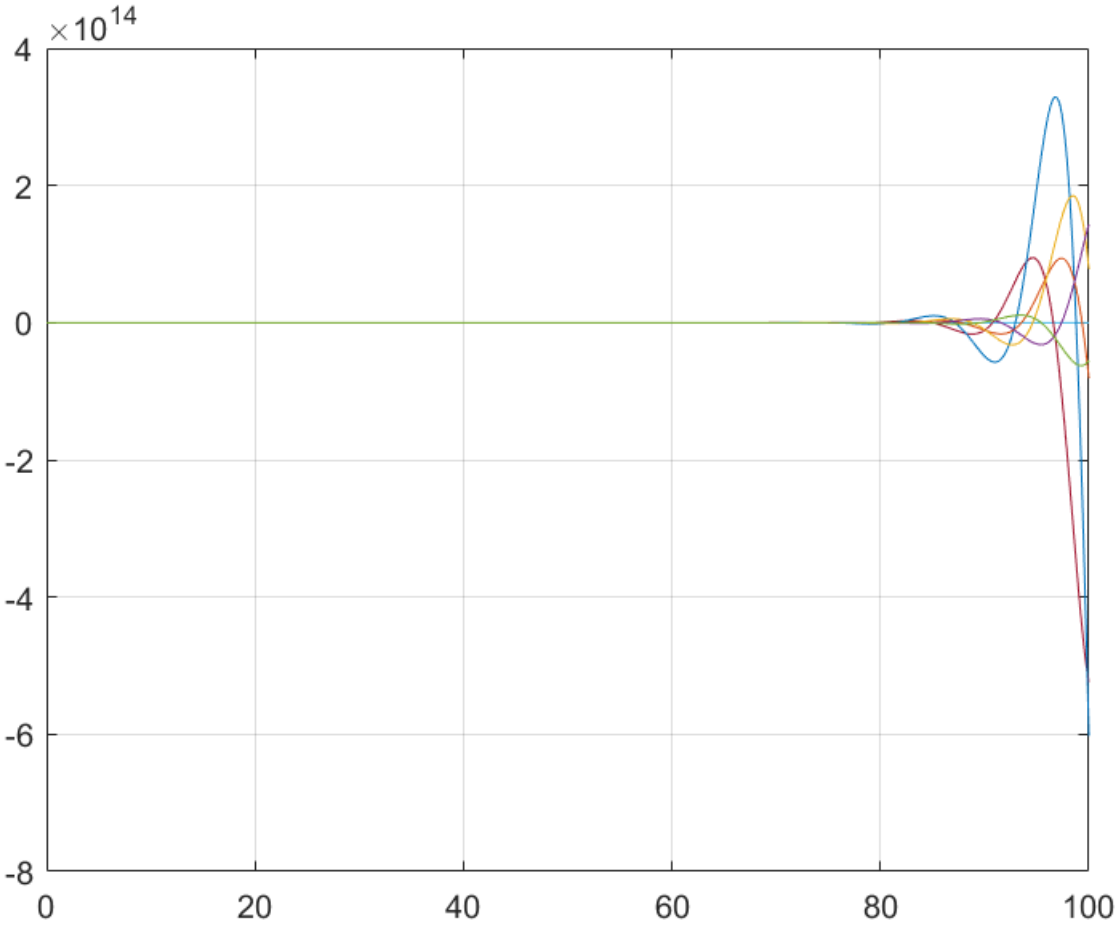

Figure 24: LQG Initial response for Linear State Case 3

We reconfigure our controller to asymptotically track a constant reference on the x axis: Our goal is to reduce the following cost function for the best Reference Tracking:

$$\int_0^\infty (X(t) - X(d))^T Q(X(t) - X(d)) + (U_k - U_\infty)^T R(U_k - U_\infty dt$$

The controller's LQG and LQR sections are changed to reduce the impact of the aforementioned cost function. This provides Optimal Reference Tracking for the system.

Yes, this design can accommodate the Cart's continual force disturbances. The controller will take these disturbances into account assuming that the force disturbances are of a Gaussian type as shown in Figure 25 and Figure 26.

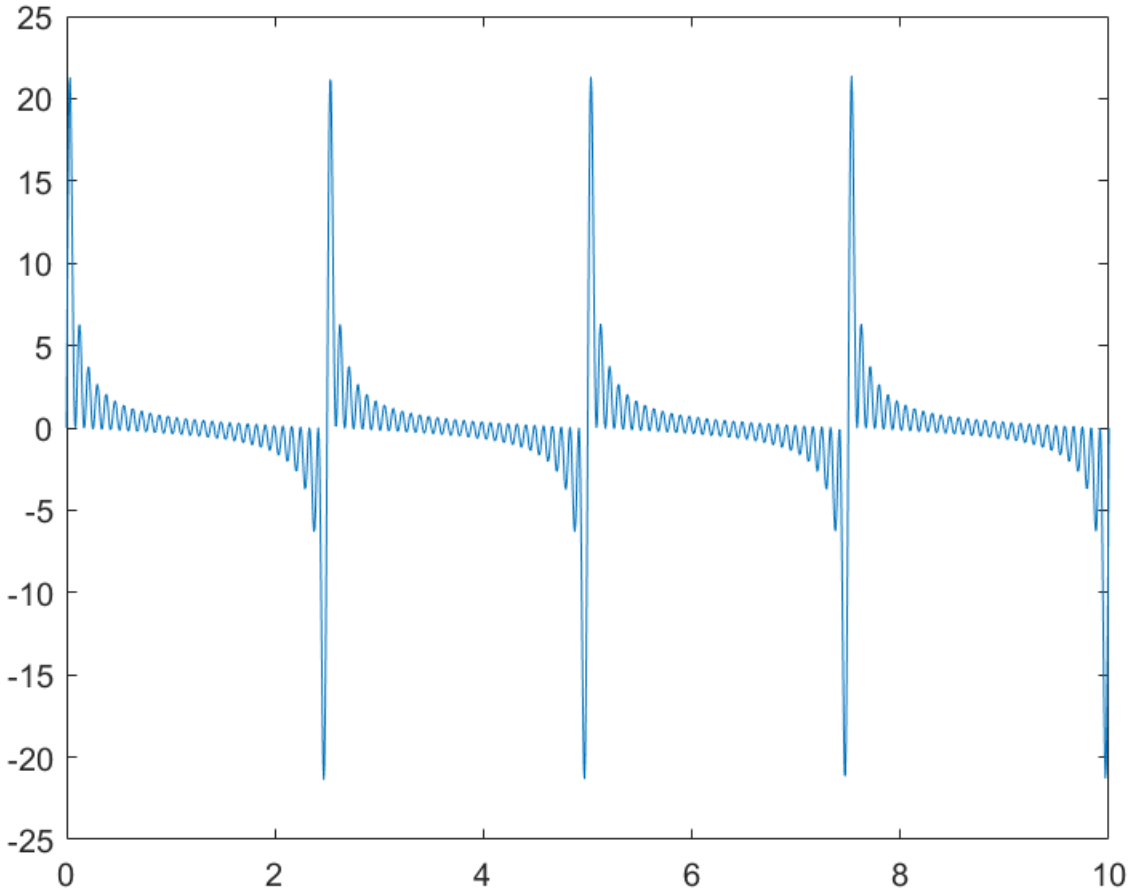

Figure 25: Caption

[H]

9 Conclusion

In this project our team worked on the Linearization of the the given cart and two pendulum model. The system dynamics was then analyzed for controllability and and LQR controller was developed to stabilize the linear dynamics. The developed LQR was also tested on the original non-linear dynamics. The system was tested for observability and a Luenberger Observer was designed to track the Linear system error and the original non-linear system error. Finally an LQG controller was designed which integrate the Linear State Estimator (Kalman Filter) with the LQR controller and the output of the controller is analyzed for the Initial state response and the Step response. Finally the LQG was analyzed in presense of disturbance applied on the system. The controller was able to still perform and produce the desired control output properly.

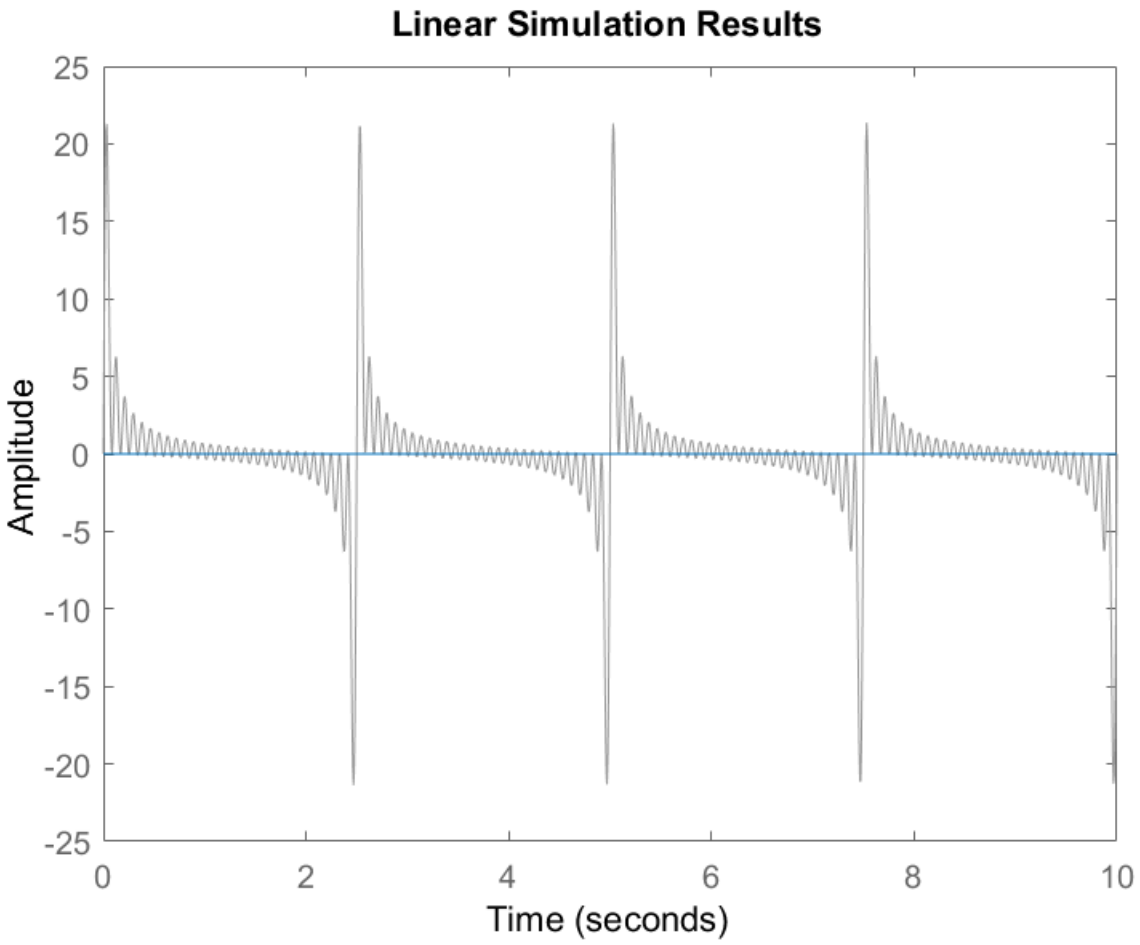

Figure 26: Caption