ENPM667 Final Project

Adam Lobo, Brendan Neal, and Hoang Pham

December 18th, 2023

Contents

1 Introduction

2 Equations of Motion

Given the system depicted in Figure 1, define the equations of motion:

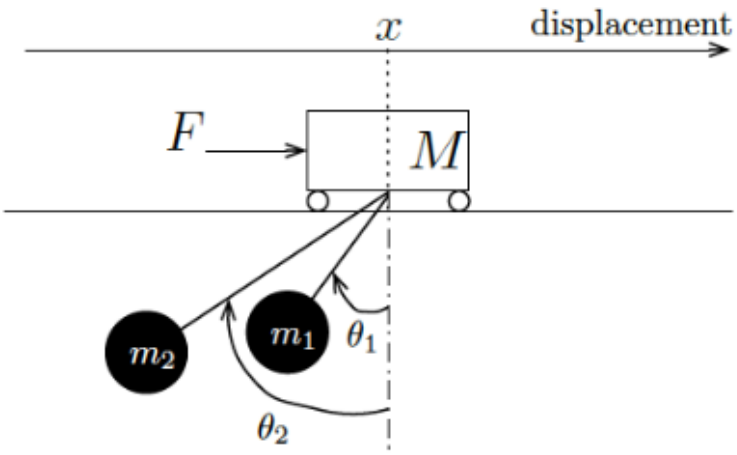

Figure 1: System Description

The equations of motion for the system can be calculated from the Euler-Lagrangian equation:

$$L = K - P \tag{1}$$

where K is the kinetic energy of the system and P is the potential energy of the system.

The kinetic energy, K, is defined as:

$$K = \frac{1}{2}M\dot{x}^2 + \frac{1}{2}m_1{V_1}^2 + \frac{1}{2}m_2{V_2}^2 \tag{2}$$

where V_1 and V_2 are the magnitude of the velocities of the pendulum masses. Since V_1^2 and V_2^2 are functions of x and θ and taking clockwise rotation as positive θ they are written as:

$${V_1}^2 = (\dot{x} - l_1\dot{\theta_1}cos(\theta_1))^2 + (l_1\dot{\theta_1}sin(\theta_1))^2 \tag{3}$$

$${V_2}^2 = (\dot{x} - l_2\dot{\theta_2}cos(\theta_2))^2 + (l_2\dot{\theta_2}sin(\theta_2))^2 \tag{4}$$

where l_1 and l_2 are the lengths of the string connecting $m1$ and $m2$ to the cart. Substituting equations 3 and 4 into equation 2 yields the fully defined kinetic

energy of the system:

$$K = \frac{1}{2}M\dot{x}^2 + \frac{1}{2}m_1((\dot{x} - l_1\dot{\theta_1}cos(\theta_1))^2 + (l_1\dot{\theta_1}sin(\theta_1))^2) + \frac{1}{2}m_2((\dot{x} - l_2\dot{\theta_2}cos(\theta_2))^2 + (l_2\dot{\theta_2}sin(\theta_2))^2) \tag{5}$$

The next element of the Euler-Lagrange equation is the potential energy of the system. The only source of potential energy for the system is gravity acting upon the two pendulum masses. As a result, the potential energy can be defined as:

$$P = -m_1gl_1cos(\theta_1) - m_2gl_2cos(\theta_2) \tag{6}$$

Combining equations 5 and 6 yields the fully defined Euler-Lagrange Equation:

$$L = \frac{1}{2}M\dot{x}^2 + \frac{1}{2}m_1((\dot{x} - l_1\dot{\theta_1}cos(\theta_1))^2 + (l_1\dot{\theta_1}sin(\theta_1))^2) + \frac{1}{2}m_2((\dot{x} - l_2\dot{\theta_2}cos(\theta_2))^2 + (l_2\dot{\theta_2}sin(\theta_2))^2) + m_1gl_1cos(\theta_1) + m_2gl_2cos(\theta_2) \tag{7}$$

The equations of motion can be solved for by solving the Euler-Lagrange equation in terms of x and its derivatives, θ_1 and its derivatives, and θ_2 and its derivatives:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{x}} - \frac{\partial L}{\partial x} = F \tag{8}$$

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta_1}} - \frac{\partial L}{\partial \theta_1} = 0 \tag{9}$$

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta_2}} - \frac{\partial L}{\partial \theta_2} = 0 \tag{10}$$

Solving equation 8:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{x}} = (m_1 + m_2 + M)\ddot{x} - m_1l_1(\ddot{\theta_1}cos(\theta_1) - (\dot{\theta_1})^2sin(\theta_1)) - m_2l_2(\ddot{\theta_2}cos(\theta_2) - (\dot{\theta_2})^2sin(\theta_2)) \tag{11}$$

$$\frac{\partial L}{\partial x} = 0 \tag{12}$$

Thus:

$$F = (m_1 + m_2 + M)\ddot{x} - m_1l_1(\ddot{\theta_1}cos(\theta_1) - (\dot{\theta_1})^2sin(\theta_1)) - m_2l_2(\ddot{\theta_2}cos(\theta_2) - (\dot{\theta_2})^2sin(\theta_2)) \tag{13}$$

Solving equation 9:

$$\frac{\partial L}{\partial \dot{\theta_1}} = m_1{l_1}^2\dot{\theta_1} - m_1l_1\dot{x}cos(\theta_1) \tag{14}$$

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta_1}} = m_1 {l_1}^2 \ddot{\theta_1} - m_1 l_1 \ddot{x} cos(\theta_1) + m_1 l_1 \dot{x}\dot{\theta_1} sin(\theta_1) \tag{15}$$

$$\frac{\partial L}{\partial \theta_1} = m_1 l_1 \dot{\theta_1} \dot{x} sin(\theta_1) - m_1 l_1 g sin(\theta_1) \tag{16}$$

Thus:

$$\begin{aligned} 0 = m_1 {l_1}^2 \ddot{\theta_1} - m_1 l_1 \ddot{x} cos(\theta_1) + m_1 l_1 \dot{x}\dot{\theta_1} sin(\theta_1) - \\ (m_1 l_1 \dot{\theta_1} \dot{x} sin(\theta_1) - m_1 l_1 g sin(\theta_1)) \end{aligned} \tag{17}$$

And simplifying:

$$\begin{aligned} 0 = m_1 l_1^2 \ddot{\theta_1} - m_1 \ddot{x} l_1 cos(\theta_1) + m_1 l_1 g sin(\theta_1) \\ 0 = l_1 \ddot{\theta_1} - \ddot{x} cos(\theta_1) + g sin(\theta_1) \end{aligned} \tag{18}$$

Solving equation 10:

$$\frac{\partial L}{\partial \dot{\theta_2}} = m_2 {l_2}^2 \dot{\theta_2} - m_2 \dot{x} l_1 cos(\theta_2) \tag{19}$$

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta_2}} = m_2 {l_2}^2 \ddot{\theta_2} - m_2 l_2 \ddot{x} cos(\theta_2) + m_2 l_2 \dot{x}\dot{\theta_2} sin(\theta_2) \tag{20}$$

$$\frac{\partial L}{\partial \theta_2} = m_2 l_2 \dot{\theta_2} \dot{x} sin(\theta_2) - m_2 l_2 g sin(\theta_2) \tag{21}$$

Thus:

$$\begin{aligned} 0 = m_2 {l_2}^2 \ddot{\theta_2} - m_2 l_2 \ddot{x} cos(\theta_2) + m_2 l_2 \dot{x}\dot{\theta_2} sin(\theta_2) - \\ (m_2 l_2 \dot{\theta_2} \dot{x} sin(\theta_2) - m_2 l_2 g sin(\theta_2)) \end{aligned} \tag{22}$$

And simplifying:

$$\begin{aligned} 0 = m_2 l_2^2 \ddot{\theta_2} - m_2 \ddot{x} l_2 cos(\theta_2) + m_2 l_2 g sin(\theta_2) \\ 0 = l_2 \ddot{\theta_2} - \ddot{x} cos(\theta_2) + g sin(\theta_2) \end{aligned} \tag{23}$$

Thus, the equations of motion for this system are equations 13, 18, and 23.

3 Nonlinear State Space Representation

In order to derive the nonlinear state space representation of the system, the state variables must be selected. The state variables for this system are:

$$X(t) = \begin{bmatrix} x \\ \dot{x} \\ \theta_1 \\ \dot{\theta_1} \\ \theta_2 \\ \dot{\theta_2} \end{bmatrix} \tag{24}$$

and the input is:

$$U(t) = F \tag{25}$$

As a result, $\dot{X}$ is:

$$\dot{X}(t) = \begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta_1} \\ \ddot{\theta_1} \\ \dot{\theta_2} \\ \ddot{\theta_2} \end{bmatrix} \tag{26}$$

The next step in deriving the state space representation is to solve for $\ddot{x}$, $\ddot{\theta_1}$, and $\ddot{\theta_2}$. Since $\ddot{\theta_1}$ and $\ddot{\theta_2}$ appear in 13, we can solve for them and substitute the resulting $\ddot{x}$ back into equations 18 and 23. Solving for $\ddot{\theta_1}$:

$$\ddot{\theta_1} = \frac{\ddot{x}cos(\theta_1) - gsin(\theta_1)}{l_1} \tag{27}$$

Solving for $\ddot{\theta_2}$:

$$\ddot{\theta_2} = \frac{\ddot{x}cos(\theta_2) - gsin(\theta_2)}{l_2} \tag{28}$$

Substituting equations 27 and 28 into equation 13:

$$\begin{aligned} F = (m_1 + m_2 + M)\ddot{x} - m_1 l_1 (&\frac{\ddot{x}cos(\theta_1) - gsin(\theta_1)}{l_1} cos(\theta_1) - (\dot{\theta_1})^2 sin(\theta_1)) \\ -m_2 l_2 (&\frac{\ddot{x}cos(\theta_2) - gsin(\theta_2)}{l_2} cos(\theta_2) - (\dot{\theta_2})^2 sin(\theta_2)) \end{aligned} \tag{29}$$

Simplifying and solving for $\ddot{x}$ yields:

$$\begin{aligned} \ddot{x} = &\frac{1}{(m_1 + m_s + M) - m_1 cos^2(\theta_1) - m_2 cos^2(\theta_2)} \\ &*(F - m_1 gsin(\theta_1)cos(\theta_1) - m_1 l_1 (\dot{\theta_1})^2 sin(\theta_1) \\ &-m_2 gsin(\theta_2)cos(\theta_2) - m_2 l_2 (\dot{\theta_2})^2 sin(\theta_2)) \end{aligned} \tag{30}$$

Now that $\ddot{x}$ is a function of only the state variables, we substitute equation 30 into both equations 27 and 28 in order to form the proper state space equations. For the sake of visual clarity within the report we will forgo presenting the result.

Because of the non-linearity of this system, we cannot write the traditional state space form of $\dot{X}(t) = AX(t) + BU(t)$. However, because we have equations for $\ddot{x}$, $\ddot{\theta_1}$, and $\ddot{\theta_2}$ in terms of the state variables, we can write the general state space form of nonlinear systems:

$$\dot{X} = f(X, U) \tag{31}$$

3.1 Final Answers for Part A

The equations of motion are:

$$\begin{aligned} F = (m_1 + m_2 + M)\ddot{x} - m_1 l_1(\ddot{\theta_1} cos(\theta_1) - (\dot{\theta_1})^2 sin(\theta_1)) \\ -m_2 l_2(\ddot{\theta_2} cos(\theta_2) - (\dot{\theta_2})^2 sin(\theta_2)) \end{aligned} \tag{32}$$

$$\begin{aligned} 0 = m_1 l_1^2 \ddot{\theta_1} - m_1 \ddot{x} l_1 cos(\theta_1) + m_1 l_1 g sin(\theta_1) \\ 0 = l_1 \ddot{\theta_1} - \ddot{x} cos(\theta_1) + g sin(\theta_1) \end{aligned} \tag{33}$$

$$\begin{aligned} 0 = m_2 l_2^2 \ddot{\theta_2} - m_2 \ddot{x} l_2 cos(\theta_2) + m_2 l_2 g sin(\theta_2) \\ 0 = l_2 \ddot{\theta_2} - \ddot{x} cos(\theta_2) + g sin(\theta_2) \end{aligned} \tag{34}$$

The nonlinear state space system is given by:

$$\begin{aligned} \ddot{x} = \frac{1}{(m_1 + m_s + M) - m_1 cos^2(\theta_1) - m_2 cos^2(\theta_2)} \\ *(F - m_1 g sin(\theta_1) cos(\theta_1) - m_1 l_1 (\dot{\theta_1})^2 sin(\theta_1) \\ -m_2 g sin(\theta_2) cos(\theta_2) - m_2 l_2 (\dot{\theta_2})^2 sin(\theta_2)) \end{aligned} \tag{35}$$

$$\ddot{\theta_1} = \frac{\ddot{x} cos(\theta_1) - g sin(\theta_1)}{l_1} \tag{36}$$

$$\ddot{\theta_2} = \frac{\ddot{x} cos(\theta_2) - g sin(\theta_2)}{l_2} \tag{37}$$

$$\dot{X} = f(X, U) \tag{38}$$

4 Linearization of System

Since there is an equilibrium point around $x = 0$, $\theta_1 = 0$, and $\theta_2 = 0$, we will be using the small-angle approximation theorem in order to linearize this system. Small angle approximation is:

$$\begin{aligned} sin(\theta_1) \approx \theta_1 \\ sin(\theta_2) \approx \theta_2 \\ cos(\theta_1) \approx 1 \\ cos(\theta_2) \approx 1 \end{aligned} \tag{39}$$

Additionally, this equilibrium point selection places another limiting condition on the system:

$$\begin{aligned} (\dot{\theta_1})^2 \approx 0 \\ (\dot{\theta_2})^2 \approx 0 \end{aligned} \tag{40}$$

Applying these limiting conditions to equation 30:

$$\ddot{x} = \frac{1}{(M + m_1 + m_2) - m_1 - m_2} * (F - m_1 g \theta_1 - m_2 g \theta_2) \tag{41}$$

Simplifying:

$$\ddot{x} = \frac{1}{M} * (F - m_1 g\theta_1 - m_2 g\theta_2) \tag{42}$$

Applying these limiting conditions to equation 27:

$$\ddot{\theta_1} = \frac{1}{l_1} * (\ddot{x} - g\theta_1) \tag{43}$$

Substituting equation 42 into 43:

$$\ddot{\theta_1} = \frac{1}{l_1} * (\frac{1}{M} * (F - m_1 g\theta_1 - m_2 g\theta_2) - g\theta_1) \tag{44}$$

Simplifying:

$$\ddot{\theta_1} = \frac{1}{Ml_1} F - \frac{(M + m_1)g}{Ml_1}\theta_1 - \frac{m_2 g}{Ml_1}\theta_2 \tag{45}$$

Applying these limiting conditions to equation 28:

$$\ddot{\theta_2} = \frac{1}{l_2} * (\ddot{x} - g\theta_2) \tag{46}$$

Substituting equation 42 into 46:

$$\ddot{\theta_2} = \frac{1}{l_2} * (\frac{1}{M} * (F - m_1 g\theta_1 - m_2 g\theta_2) - g\theta_2) \tag{47}$$

Simplifying:

$$\ddot{\theta_2} = \frac{1}{Ml_2} F - \frac{m_1 g}{Ml_2}\theta_1 - \frac{(M + m_2)g}{Ml_2}\theta_2 \tag{48}$$

Now, using equations 42, 45, and 48, we can form our linearized state space system. Using the state variables and inputs defined in equation 24, we can write $\dot{X}(t) = AX(t) + BU(t)$, where:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{m_1 g}{M} & 0 & -\frac{m_2 g}{M} & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{(M+m_1)g}{Ml_1} & 0 & -\frac{m_2 g}{Ml_1} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & -\frac{m_1 g}{Ml_2} & 0 & -\frac{(M+m_2)g}{Ml_2} & 0 \end{bmatrix} \tag{49}$$

and:

$$B = \begin{bmatrix} 0 \\ \frac{1}{M} \\ 0 \\ \frac{1}{Ml_1} \\ 0 \\ \frac{1}{Ml_2} \end{bmatrix} \tag{50}$$

4.1 Final Answers for Part B

The full state space form is expressed as:

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta_1} \\ \ddot{\theta_1} \\ \dot{\theta_2} \\ \ddot{\theta_2} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{m_1 g}{M} & 0 & -\frac{m_2 g}{M} & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{(M+m_1)g}{Ml_1} & 0 & -\frac{m_2 g}{Ml_1} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & -\frac{m_1 g}{Ml_2} & 0 & -\frac{(M+m_2)g}{Ml_2} & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta_1 \\ \dot{\theta_1} \\ \theta_2 \\ \dot{\theta_2} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{M} \\ 0 \\ \frac{1}{Ml_1} \\ 0 \\ \frac{1}{Ml_2} \end{bmatrix} F \tag{51}$$

5 Controllability Conditions

In order to determine the controllability of the system, we first had to set up the controllability matrix. Because the A matrix is 6x6, the controllability matrix C is as follows:

$$C = \begin{bmatrix} B & AB & A^2B & A^3B & A^4B & A^5B \end{bmatrix} \tag{52}$$

Using MATLAB to perform symbolic matrix multiplication, we found the controllability matrix for this system. The system is controllable if and only if the $rank(C) = 6$ (full rank). Using a manual systematic grid search method, we would try different values for m_1, m_2, M, l_1, and l_2 and check the rank of C. In our search, the only time the rank condition failed was when:

$$l_1 = l_2 \tag{53}$$

Thus, to maintain controllability of the system, $l_1 \neq l_2$.

Though not related to controllability, in order to keep elements of A and B bounded, the following conditions must also be held:

$$\begin{aligned} l_1 &\neq 0 \\ l_2 &\neq 0 \\ M &\neq 0 \end{aligned} \tag{54}$$

5.1 Final Answers for Part C

$$\begin{aligned} l_1 &\neq l_2 \\ l_1 &\neq 0 \\ l_2 &\neq 0 \\ M &\neq 0 \end{aligned} \tag{55}$$

6 LQR Controller

6.1 Controllability Check

In order to check controllability, we plugged $M = 1000$, $m_1 = m_2 = 100$, $l_1 = 20$, $l_2 = 10$, and $g = 9.81$ into our the script used in the previous section and checked the rank of C. Since the given $l_1 \neq l_2$, we could predict that the system will be controllable, but we verified by substituting the values. This system is controllable using the given parameters.

6.2 Design of the LQR Controller

For our LQR controller, we will be using the cost function given by:

$$J(k, X(0)) = \int_0^\infty X^T(t)QX(t) + U_K^T(t)RU_K(t)\,dt \tag{56}$$

Where Q and R are user-selected values. We also decided on using the following for our initial conditions:

$$X_0 = \begin{bmatrix} 0 & 0 & \frac{\pi}{4} & 0 & \frac{\pi}{4} & 0 \end{bmatrix} \tag{57}$$

We uses MATLAB's in-built LQR function to get the gain values and MATLAB's in-built state space function to simulate the response under initial conditions for the linearized system.

6.3 Simulations of the Linearized System

6.3.1 Initial (Bad) Values of Q and R

To get an idea of how the response would look under an LQR controller, we first simulated the response using Q as an identity matrix (I) and $R = 1$. Depicted in Figure 2 is our system response under the previous Q and R values:

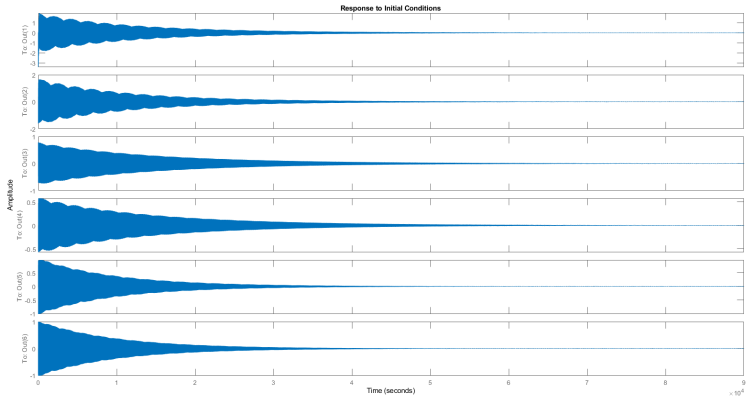

Figure 2: Response Under Bad Q and R Values

6.3.2 Tuned Values of Q and R

Our process for tuning the values of Q and R was to systematically change them, run the response, then decide of the output is suitable or not. Our first trial was to increase the magnitude of Q and decrease the magnitude of R. Our new Q matrix looked like:

$$Q = \begin{bmatrix} 1000 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1000 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1000 \end{bmatrix} \tag{58}$$

and $R = 0.001$. Depicted in Figure 3 is our system response under the previous Q and R values:

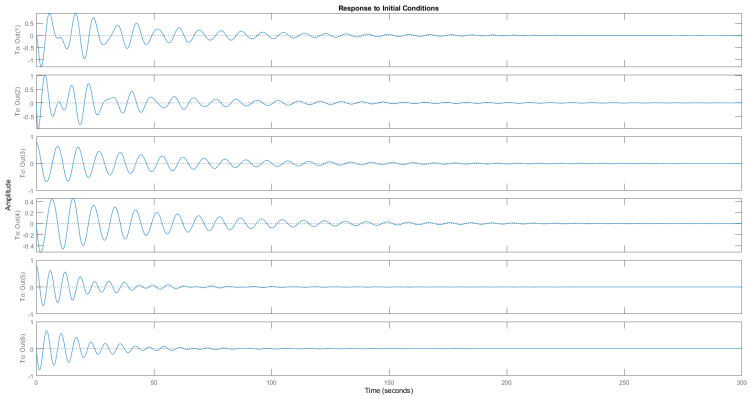

Figure 3: Response Under Good Q and R Values

This response was suitable for the purposes of this assignment.

6.4 Simulations of the Original Nonlinear System

Because the nonlinear system could not be put into the standard form for state space, we had to get more creative. We first created a function that included all of our state variables based off of equations 30, 27, and 28. We then used MATLAB's in-built *ode*45 tool in order to simulate the time state history of the system under the same initial conditions as the linearized model.

6.4.1 Initial (Bad) Values of Q and R

Using the "good" Q and R values for the linearized system just above, we obtained the following response for the nonlinear system depicted in Figure 4:

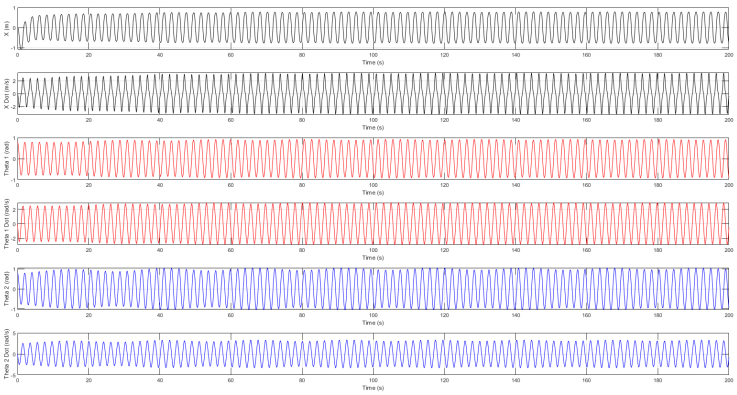

Figure 4: Response Under Bad Q and R Values

This response is not suitable, and thus more tuning for the nonlinear system had to be performed.

6.4.2 Tuned Values of Q and R

Keeping Q the same and setting $R = 0.00001$, we obtained the following response for the nonlinear system depicted in Figure 5:

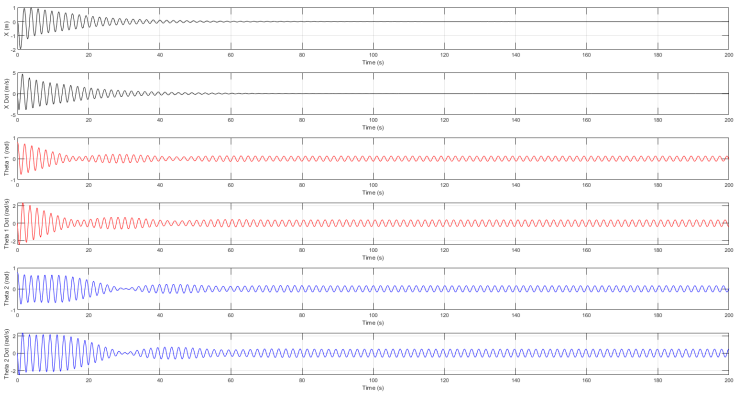

Figure 5: Response Under Good Q and R Values

This response is suitable for the purposes of this assignment as x is now tracking at 0.

6.5 Lyapunov's Indirect Method to Verify Stability

6.5.1 Initial (Bad) Values of Q and R (Linearized)

The eigenvalues for this system are:

$$\begin{bmatrix} -0.0001 + 0.7285i \\ -0.0001 - 0.7285i \\ -0.0001 + 1.0430i \\ -0.0001 - 1.0430i \\ -0.0204 + 0.0204i \\ -0.0204 - 0.0204i \end{bmatrix} \tag{59}$$

Because all of the eigenvalues have negative real part, the system is locally stable.

6.5.2 Tuned Values of Q and R (Linearized)

The eigenvalues for this system are:

$$\begin{bmatrix} -0.0199 + 0.7054i \\ -0.0199 - 0.7054i \\ -0.0506 + 1.0047i \\ -0.0506 - 1.0047i \\ -0.8205 + 0.5306i \\ -0.8205 - 0.5306i \end{bmatrix} \tag{60}$$

Because all of the eigenvalues have negative real part, the system is locally stable.

7 Observability Conditions

In order to determine the observability of the system, we first had to set up the observability matrix. Because the A matrix is 6x6, the observability matrix C is as follows:

$$O = \begin{bmatrix} C^T & A^T C^T & A^{T^2} C^T & A^{T^3} C^T & A^{T^4} C^T & A^{T^5} C^T \end{bmatrix} \tag{61}$$

The C matrices corresponding to the output vectors are:

$$(x) : \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{62}$$

$$(\theta_1, \theta_2) : \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{63}$$

$$(x, \theta_2) : \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{64}$$

$$(x, \theta_1, \theta_2) : \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{65}$$

The system is observable if the $rank(O) = 6 = \text{full}$.

7.1 Final Answer for Part E

Using MATLAB to perform the matrix multiplication and rank tests, the system is observable for output vectors: (x), (x, θ_2), and (x, θ_1, θ_2).

8 Luenberger Observer

8.1 Design

In order to design our Luenberger observer for the system, we first have to define the observer system. The general form for an observer system is:

$$\begin{aligned} \dot{\hat{X}} &= A\hat{X} + BU + L(Y - \hat{Y}) \\ \hat{Y} &= C\hat{X} + DU \end{aligned} \tag{66}$$

where L is the observer gain matrix. We will use the pole placement method in order to find out observer gain matrix. We will define our desired poles to be in the left half plane and then use MATLAB's "place" function. Once we have our gain matrix, we define our closed loop states as follows:

$$A_{cl} = \begin{bmatrix} A - BK & BK \\ 0 & A - LC \end{bmatrix} \tag{67}$$

$$B_{cl} = \begin{bmatrix} B \\ 0 \end{bmatrix} \tag{68}$$

$$C_{cl} = \begin{bmatrix} C \\ 0 \end{bmatrix} \tag{69}$$

$$D_{cl} = D = \begin{bmatrix} 0 \end{bmatrix} \tag{70}$$

With C being the output vector we are testing and K being the gain matrix from the LQR controller. For our desired poles, P, we are selecting:

$$P = \begin{bmatrix} -5 & -4 & -2 & -8 & -9 & -3 \end{bmatrix} \tag{71}$$

8.2 Simulation of the Linearized System

We test the observer under the given initial condition:

$$X_0 = \begin{bmatrix} 3 & 0 & \frac{\pi}{4} & 0 & \frac{\pi}{4} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T \tag{72}$$

8.2.1 Output Vector 1

This output vector is (x). Below in Figure 6 is the system response to the initial condition:

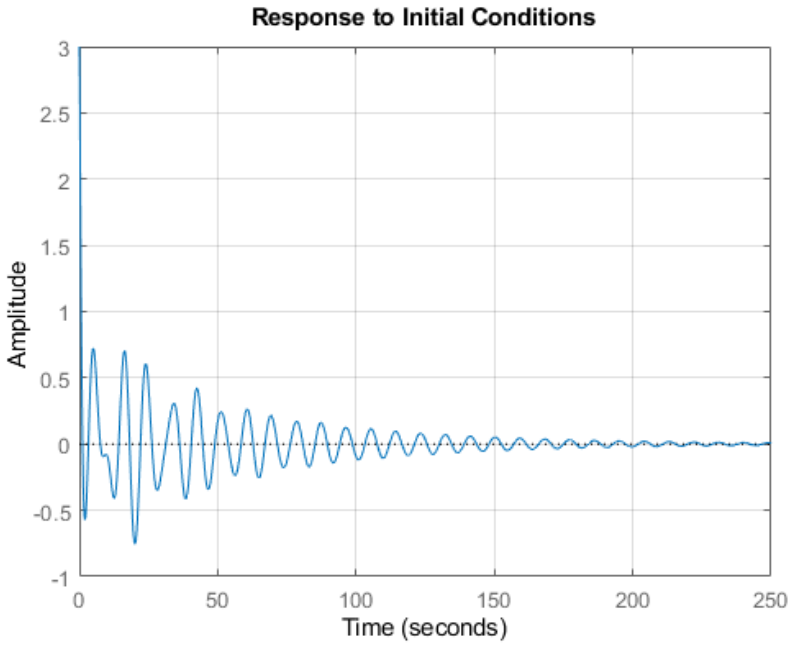

Figure 6: Observer Initial Condition Response

Below in Figure 7 is the system response to a step input:

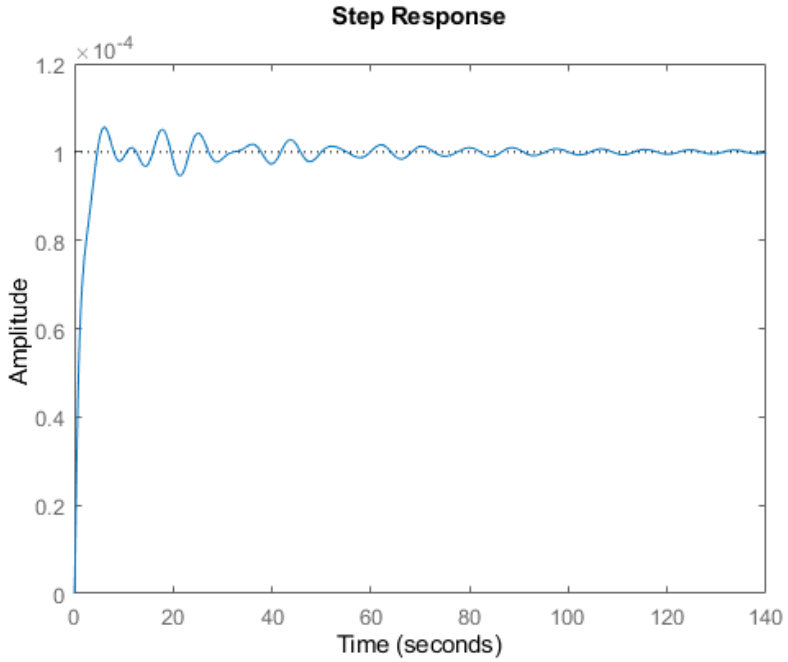

Figure 7: Observer Unit Step Response

8.2.2 Output Vector 2

This output vector is (x, θ_2). Below in Figure 8 is the system response to the initial condition:

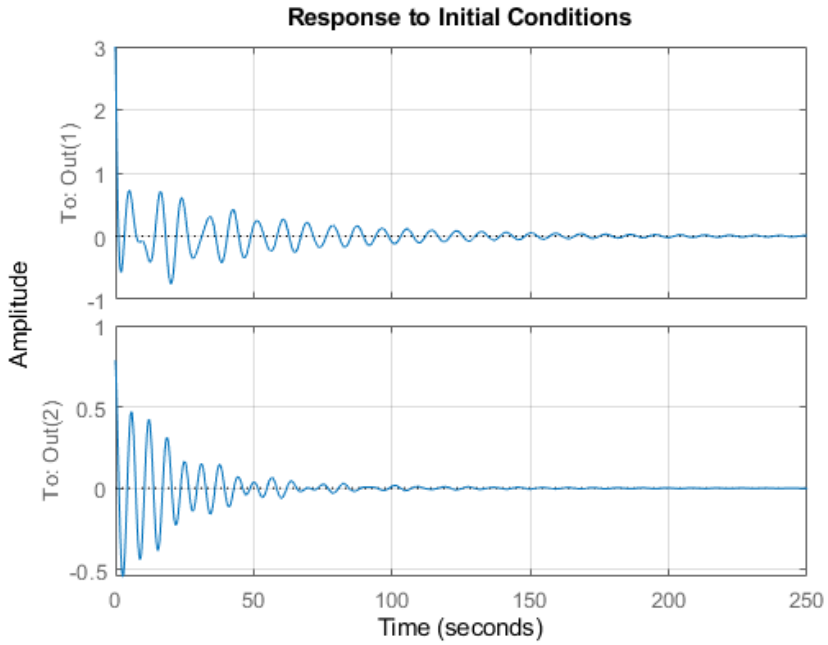

Figure 8: Observer Initial Condition Response

Below in Figure 9 is the system response to a step input:

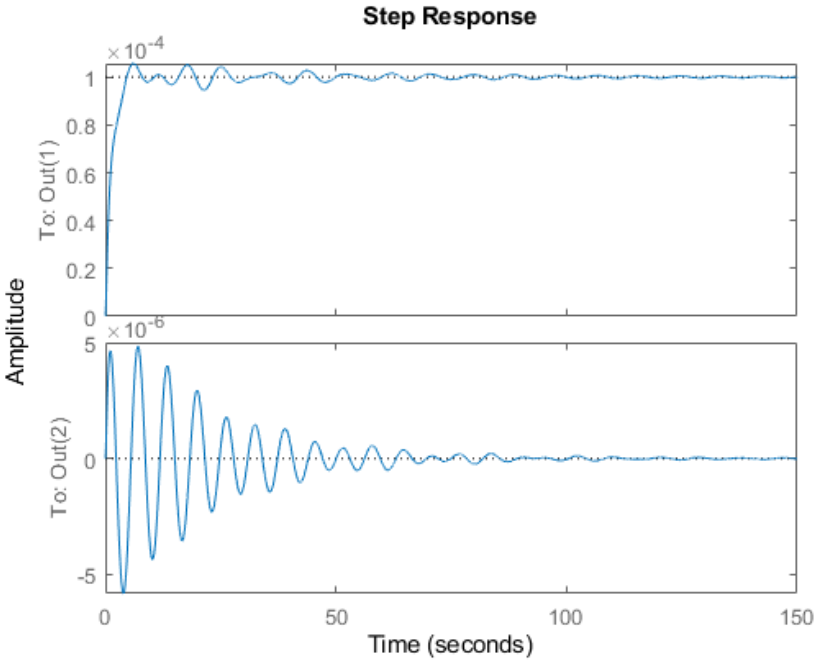

Figure 9: Observer Unit Step Response

8.2.3 Output Vector 3

This output vector is (x, θ_1, θ_2). Below in Figure 10 is the system response to the initial condition:

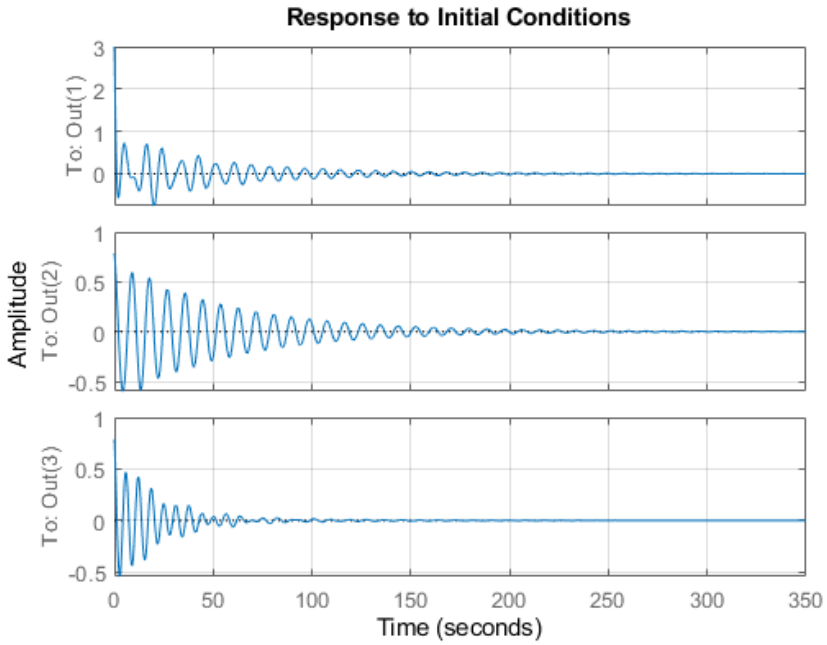

Figure 10: Observer Initial Condition Response

Below in Figure 11 is the system response to a step input:

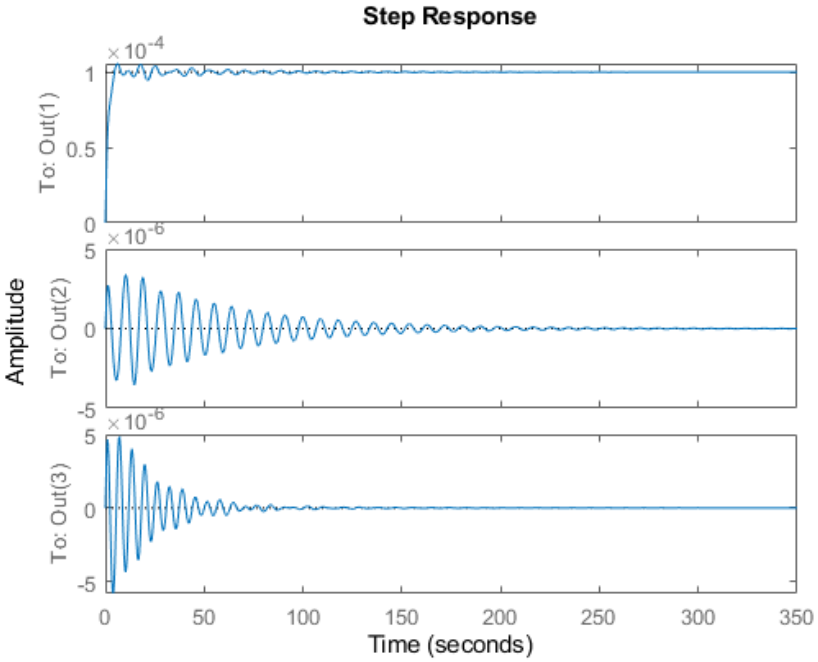

Figure 11: Observer Unit Step Response

8.3 Simulation of the Original Nonlinear System

Unfortunately, we could not figure out how to design and apply the Luenberger observer to the nonlinear system.

9 LQG Controller

9.1 Design of the LQG Controller

In order to design a LQG controller, we have to use the gain matrix (K) obtained from the LQR controller as well as a Kalman filter in order to obtain the optimal L. The smallest observable output vector is $x(t)$. The system used for the LQG controller is as follows:

$$\dot{X}(t) = AX(t) + BU(t) + BW = CX(t) + V \tag{73}$$

where W and V are the disturbance and noise. In MATLAB, we use the LQR system and pass that to the in-built Kalman function along with the covariance of disturbance set to 0.01 and the covariance of noise as 1. From here, we obtain out L. From here, we can form our closed loop state space matrices as follows:

$$A_{cl} = \begin{bmatrix} A - BK & BK \\ 0 & A - LC \end{bmatrix} \tag{74}$$

$$B_{cl} = \begin{bmatrix} B_d \\ B_d \end{bmatrix} \tag{75}$$

$$C_{cl} = \begin{bmatrix} C \\ 0 \end{bmatrix} \tag{76}$$

$$D_{cl} = D = \begin{bmatrix} 0 \end{bmatrix} \tag{77}$$

With all of our closed loop matrices in place, we can now simulate the system.

9.2 Simulations of the Original Nonlinear System

Unfortunately, we could not figure out how to apply the LQG controller to the nonlinear system. As a result, the following was applied to the linearized system. Taking:

$$X_0 = \begin{bmatrix} 3 & 0 & \frac{\pi}{4} & 0 & \frac{\pi}{4} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T \tag{78}$$

And $F = 0$, we used MATLAB's in-built state space functions as well as $lsim$ to simulate the response of the system. Depicted below in Figure 12 are the state variables as a function of time:

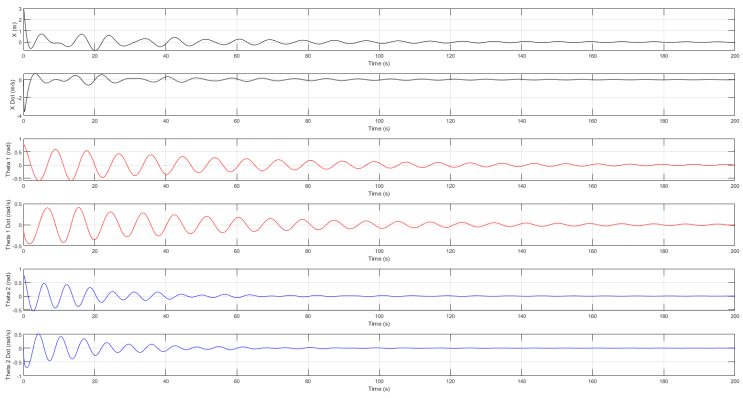

Figure 12: LGQ State Response

Depicted below in Figure 13 is the output vector (x) as a function of time:

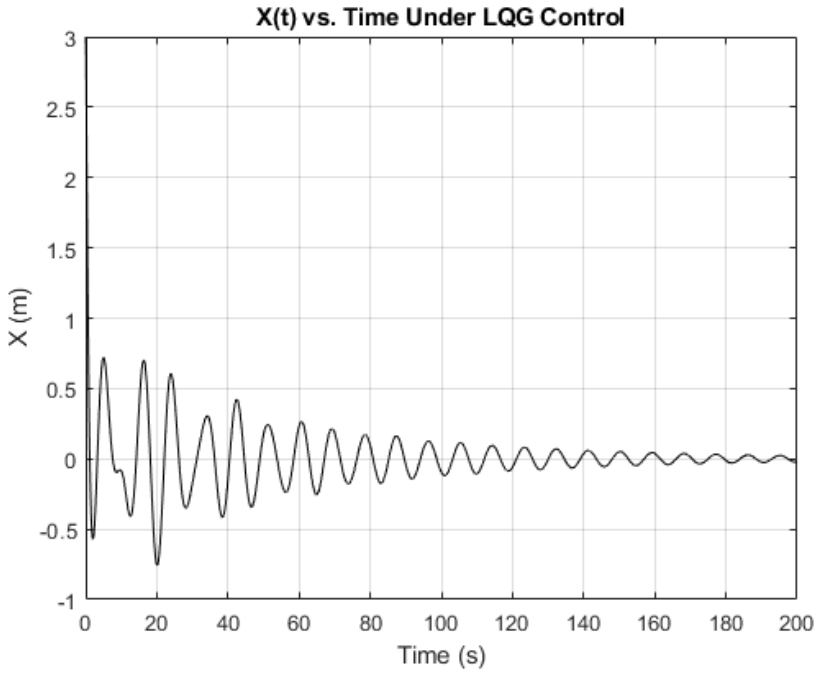

Figure 13: LQG Output Response

9.3 Analysis

9.3.1 Question 1: Optimal Reference Tracking

In order to reconfigure the controller for optimal reference tracking, we would have to change the cost function. To optimally track a reference, the new cost function would have to be minimized:

$$J = \int_0^\infty (X(t) - X_d)^T Q(X(t) - X_d) + (U_K - U_\infty)^T(t)(U_K - U_\infty)\, dt \qquad (79)$$

$U(t)$ would become: $U(t) = K(X(t) - X_d)$.

9.3.2 Question 2: Constant Force Disturbance

As of right now the current design of the controller will not reject constant force disturbances. Our state definition does not contain an integral component, and thus when a constant disturbance is acting on the closed loop system, there will be a constant steady state error in the output.

10 Link to Simulation Videos

Clicking on Clicking on this link will take you to a YouTube video of us performing the simulations for this project.

Appendix 1:

MATLAB Code in PDF Format

```
%% Brendan Neal, Adam Lobo, and Hoang Pham
%% ENPM667 Project 2 (Final Project)

%% Part 1 Part C: Symbolic Controllability Conditions
clc

syms MM m1 m2 l1 l2 g

%Comment these out for symbolic form or in for numerical values and form
% MM = 20
% m1 = 5
% m2 = 4
% l1 = 1
% l2 = 1
% g = -9.8

%A matrix
A = [0, 1, 0, 0, 0, 0;
    0, 0, -m1*g/MM, 0, -m2*g/MM, 0;
    0, 0, 0, 1, 0, 0;
    0, 0, -(MM+m1)*g/(MM*l1), 0, -m2*g/(MM*l1), 0;
    0, 0, 0, 0, 0, 1;
    0, 0, -m1*g/(MM*l2), 0, -(MM+m2)*g/(MM*l2), 0];

%B matrix
B = [0; 1/MM; 0; 1/(MM*l1); 0; 1/(MM*l2)];

%Controllability Matrix (not simplified)
CNTR = [B, A*B, A^2*B, A^3*B, A^4*B, A^5*B];

%Simplification
for r = 1:6

    for c = 1:6

        CNTR(r, c) = simplify(CNTR(r, c));

    end

end

%Output controllability matrix (simplified)
CNTR

%Rank of controllability matrix: < 6 = not controllable, = 6 = controllable
rank(CNTR)
```

```
%% Brendan Neal, Adam Lobo, and Hoang Pham
%% ENPM667 Project 2 (Final Project)
%% Part 1 Part D: LQR Controller
clear
clc

%% Controllability Check

% Parameters
MM = 1000;
m1 = 100;
m2 = 100;
l1 = 20;
l2 = 10;
g = 9.81;

%A matrix
A = [0, 1, 0, 0, 0, 0;
    0, 0, -m1*g/MM, 0, -m2*g/MM, 0;
    0, 0, 0, 1, 0, 0;
    0, 0, -(MM+m1)*g/(MM*l1), 0, -m2*g/(MM*l1), 0;
    0, 0, 0, 0, 0, 1;
    0, 0, -m1*g/(MM*l2), 0, -(MM+m2)*g/(MM*l2), 0];

%B matrix
B = [0; 1/MM; 0; 1/(MM*l1); 0; 1/(MM*l2)];

%Controllability Matrix
CNTR = [B, A*B, A^2*B, A^3*B, A^4*B, A^5*B];

%Output controllability matrix (simplified)
disp('The controllability matrix is:')
CNTR

%Rank of controllability matrix: < 6 = not controllable, = 6 = controllable
disp('The rank of the controllability matrix is:')
rank(CNTR)

%% Design of the LQR Controller -- Linearized System
%% First-Pass Design
%Initial Conditions
X_0 = [0, 0, pi/4, 0, pi/4, 0];
C = eye(6);
D = 0;


% Initial Selection of Q and R:
Q = [1 0 0 0 0 0;
    0 1 0 0 0 0;
    0 0 1 0 0 0;
    0 0 0 1 0 0
```

```
    0 0 0 0 1 0
    0 0 0 0 0 1];
R = 1;

% Designing LQR Controller:
[K, ~, EigenValues] = lqr(A, B, Q, R);

% Simulating LQR Test #1 using built in
LQR_Test_1 = ss(A-(B*K),B, C, D);

% Plotting Response
 figure()
 initial(LQR_Test_1, X_0)

 disp('Eigenvalues for Lyapunovs Indirect Method:')
 disp(EigenValues)
%% Area for Adjusting Q and R
%Initial Conditions
X_0 = [0, 0, pi/4, 0, pi/4, 0];
C = eye(6);
D = 0;


% New Selection of Q and R:
Q = [1000 0 0 0 0 0;
    0 1000 0 0 0 0;
    0 0 1000 0 0 0;
    0 0 0 1000 0 0
    0 0 0 0 1000 0
    0 0 0 0 0 1000];
R = 0.001;

% Designing LQR Controller:
[K, ~, EigenValues] = lqr(A, B, Q, R);

% Setting up LQR system using built in state space function
LQR_Test_1 = ss(A-(B*K),B, C, D);

% Simulate and Plot Response
 figure()
 initial(LQR_Test_1, X_0)

 % Display Eigenvalues to test for stability.
 disp('Eigenvalues for Lyapunovs Indirect Method:')
 disp(EigenValues)

 %% Applying Tuned LQR Controller to Original Nonlinear System
 % Bring Down Tuning Params into This Section
 % Initial Selection of Q and R:
Q = [1000 0 0 0 0 0;
    0 1000 0 0 0 0;
```

```
    0 0 1000 0 0 0;
    0 0 0 1000 0 0
    0 0 0 0 1000 0
    0 0 0 0 0 1000];

% R = 0.001; % BAD R
R = 0.00001; % GOOD R

% Designing LQR Controller:
[K, P, EigenValues] = lqr(A, B, Q, R);

 % Simulating Function
 timevector = 0:0.1:200;
 X_0 = [0, 0, pi/4, 0, pi/4, 0];

 %Use ode45 to solve the state equation and provide the output as a
 %function of time and X_0
[t,state_history] = ode45(@(t,state)NL_system(state,t,K),timevector,X_0);

%Plot x
figure()
subplot(6,1,1);
plot(t,state_history(:,1),'k')
grid on;
ylabel('X (m)')
xlabel('Time (s)')

%Plot X_Dot
subplot(6,1,2);
plot(t,state_history(:,2),'k')
grid on;
ylabel('X Dot (m/s)')
xlabel('Time (s)')

%Plot theta1
subplot(6,1,3);
plot(t,state_history(:,3),'r')
grid on;
ylabel('Theta 1 (rad)')
xlabel('Time (s)')

%Plot theta1_dot
subplot(6,1,4);
plot(t,state_history(:,4),'r')
grid on;
ylabel('Theta 1 Dot (rad/s)')
xlabel('Time (s)')

%Plot theta2
subplot(6,1,5);
plot(t,state_history(:,5),'b')
```

```
grid on;
ylabel('Theta 2 (rad)')
xlabel('Time (s)')

%Plot theta2_dot
subplot(6,1,6);
plot(t,state_history(:,6),'b')
grid on;
ylabel('Theta 2 Dot (rad/s)')
xlabel('Time (s)')




%% Custom Built Functions
%Function that plugs in the state of the system and calculates the
% numeric NL system outputs based on the state.
 function Xdot = NL_system(s,t,K)
 MM = 1000;
 m1 = 100;
 m2 = 100;
 l1 = 20;
 l2 = 10;
 g = 9.81;
 F = -K*s;
 Xdot = [s(2);
     (1/(m1+m2+MM-m1*cos(s(3))^2-m2*cos(s(5))^2))*(F-m1*g*sin(s(3))*cos(s(3))-m1*l1*(s↙
(4))^2*sin(s(3))-m2*g*sin(s(5))*cos(s(5))-m2*l2*(s(6))^2*sin(s(5)));
     s(4);
     (1/l1)*(((1/(m1+m2+MM-m1*cos(s(3))^2-m2*cos(s(5))^2))*(F-m1*g*sin(s(3))*cos(s(3))-↙
m1*l1*(s(4))^2*sin(s(3))-m2*g*sin(s(5))*cos(s(5))-m2*l2*(s(6))^2*sin(s(5))))*cos(s(3)))-↙
(g*sin(s(3)))
     s(6);
     (1/l2)*(((1/(m1+m2+MM-m1*cos(s(3))^2-m2*cos(s(5))^2))*(F-m1*g*sin(s(3))*cos(s(3))-↙
m1*l1*(s(4))^2*sin(s(3))-m2*g*sin(s(5))*cos(s(5))-m2*l2*(s(6))^2*sin(s(5))))*cos(s(5)))-↙
(g*sin(s(5)))];
 end
```

```
%% Brendan Neal, Adam Lobo, and Hoang Pham
%% ENPM667 Project 2 (Final Project)
%% Part 2 Part E: Observability Conditions
clear
clc

syms MM m1 m2 l1 l2 g

%Comment these out for symbolic form or in for numerical values and form
%MM = 1000;
%m1 = 100;
%m2 = 100;
%l1 = 20;
%l2 = 10;
%g = 9.8;

%A matrix
A = [0, 1, 0, 0, 0, 0;
    0, 0, -m1*g/MM, 0, -m2*g/MM, 0;
    0, 0, 0, 1, 0, 0;
    0, 0, -(MM+m1)*g/(MM*l1), 0, -m2*g/(MM*l1), 0;
    0, 0, 0, 0, 0, 1;
    0, 0, -m1*g/(MM*l2), 0, -(MM+m2)*g/(MM*l2), 0];

%B matrix
B = [0; 1/MM; 0; 1/(MM*l1); 0; 1/(MM*l2)];

%C matrices (1 -> x;   2 -> o1 & o2;   3 -> x & o2;   4 -> x & o1 & o2)
C1 = [1, 0, 0, 0, 0, 0];
C2 = [0, 0, 1, 0, 1, 0];
C3 = [1, 0, 0, 0, 1, 0];
C4 = [1, 0, 1, 0, 1, 0];

%Observability matrices
OBSVBLT1 = [transpose(C1), transpose(A)*transpose(C1), transpose(A)^2*transpose(C1),↙
transpose(A)^3*transpose(C1), transpose(A)^4*transpose(C1), transpose(A)^5*transpose↙
(C1)];
OBSVBLT2 = [transpose(C2), transpose(A)*transpose(C2), transpose(A)^2*transpose(C2),↙
transpose(A)^3*transpose(C2), transpose(A)^4*transpose(C2), transpose(A)^5*transpose↙
(C2)];
OBSVBLT3 = [transpose(C3), transpose(A)*transpose(C3), transpose(A)^2*transpose(C3),↙
transpose(A)^3*transpose(C3), transpose(A)^4*transpose(C3), transpose(A)^5*transpose↙
(C3)];
OBSVBLT4 = [transpose(C4), transpose(A)*transpose(C4), transpose(A)^2*transpose(C4),↙
transpose(A)^3*transpose(C4), transpose(A)^4*transpose(C4), transpose(A)^5*transpose↙
(C4)];

%Simplification (comment out if using numerical values)
for r = 1:6

    for c = 1:6
```

```
        OBSVBLT1(r, c) = simplify(OBSVBLT1(r, c));
        OBSVBLT2(r, c) = simplify(OBSVBLT2(r, c));
        OBSVBLT3(r, c) = simplify(OBSVBLT3(r, c));
        OBSVBLT4(r, c) = simplify(OBSVBLT4(r, c));

    end

end

%Output controllability matrix (simplified)
OBSVBLT1
OBSVBLT2
OBSVBLT3
OBSVBLT4

%Rank of observability matrices: < 6 = not observable, = 6 = observable
rank1 = rank(OBSVBLT1)
rank2 = rank(OBSVBLT2)
rank3 = rank(OBSVBLT3)
rank4 = rank(OBSVBLT4)
```

```
%% Brendan Neal, Adam Lobo, and Hoang Pham
%% ENPM667 Project 2 (Final Project)
%% Part 2 Part F: Luenberger Observer
clear
clc

%% Setup
MM = 1000;
m1 = 100;
m2 = 100;
l1 = 20;
l2 = 10;
g = 9.81;

%A matrix
A_open_loop = [0, 1, 0, 0, 0, 0;
    0, 0, -m1*g/MM, 0, -m2*g/MM, 0;
    0, 0, 0, 1, 0, 0;
    0, 0, -(MM+m1)*g/(MM*l1), 0, -m2*g/(MM*l1), 0;
    0, 0, 0, 0, 0, 1;
    0, 0, -m1*g/(MM*l2), 0, -(MM+m2)*g/(MM*l2), 0];

%B matrix
B_open_loop = [0; 1/MM; 0; 1/(MM*l1); 0; 1/(MM*l2)];

%Observable Output Vectors
%Have to change the format to get graphs properly.
C1 = [1, 0, 0, 0, 0, 0];
C2 = [1,0,0,0,0,0; 0,0,0,0,1,0];
C3 = [1, 0, 0, 0, 0, 0 ; 0, 0, 1, 0, 0, 0 ;0, 0, 0, 0, 1, 0];

D = 0;

% LQR
% Selection of Q and R:
Q = [1000 0 0 0 0 0;
    0 1000 0 0 0 0;
    0 0 1000 0 0 0;
    0 0 0 1000 0 0
    0 0 0 0 1000 0
    0 0 0 0 0 1000];
R = 0.00001; % GOOD R

% Designing LQR Controller:
[K, P, EigenValues] = lqr(A_open_loop, B_open_loop, Q, R);

%% Placeing Poles and Getting L
%Placing Poles
des_poles = [-5, -4, -2, -8, -9, -3];
%Output Vector 1
L_C1=place(A_open_loop',C1',des_poles);
```

```
L_C1=L_C1';

%Output Vector 2
L_C2=place(A_open_loop',C2',des_poles);
L_C2=L_C2';

%Output Vector 3
L_C3=place(A_open_loop',C3',des_poles);
L_C3=L_C3';

%% Forming Closed Loop Systems

%First output vector
A_closed_loop_1=[(A_open_loop-B_open_loop*K) (B_open_loop*K); zeros(size(A_open_loop))↙
(A_open_loop-L_C1*C1)];
B_closed_loop_1=[ B_open_loop ;zeros(size(B_open_loop))];
C_closed_loop_1= [C1 zeros(size(C1))];
D_closed_loop_1=D;

%Second output vector
A_closed_loop_2=[(A_open_loop-B_open_loop*K) (B_open_loop*K); zeros(size(A_open_loop))↙
(A_open_loop-L_C2*C2)];
B_closed_loop_2=[ B_open_loop ;zeros(size(B_open_loop))];
C_closed_loop_2= [C2 zeros(size(C2))];
D_closed_loop_2=D;

%Third output vector
A_closed_loop_3=[(A_open_loop-B_open_loop*K) (B_open_loop*K); zeros(size(A_open_loop))↙
(A_open_loop-L_C3*C3)];
B_closed_loop_3=[ B_open_loop ;zeros(size(B_open_loop))];
C_closed_loop_3= [C3 zeros(size(C3))];
D_closed_loop_3=D;


%% Simulating First System
%Initial Conditions
X_0 = [3; 0; pi/4; 0; pi/4; 0; 0; 0; 0; 0; 0; 0];

Vector1_Sys=ss(A_closed_loop_1,B_closed_loop_1,C_closed_loop_1,D_closed_loop_1);
figure()
step(Vector1_Sys);
figure()
initial(Vector1_Sys,X_0);
grid on

%% Simulating Second System
Vector2_Sys=ss(A_closed_loop_2,B_closed_loop_2,C_closed_loop_2,D_closed_loop_2);
figure()
step(Vector2_Sys);
figure()
initial(Vector2_Sys,X_0);
```

```
grid on

%% Simulating Third System
Vector3_Sys=ss(A_closed_loop_3,B_closed_loop_3,C_closed_loop_3,D_closed_loop_3);
figure()
step(Vector3_Sys);
figure()
initial(Vector3_Sys,X_0);
grid on
```

```
%% Brendan Neal, Adam Lobo, and Hoang Pham
%% ENPM667 Project 2 (Final Project)
%% Part 2 Part G: LQG Controller
clear
clc

%% Design
MM = 1000;
m1 = 100;
m2 = 100;
l1 = 20;
l2 = 10;
g = 9.81;

%A matrix
A_open_loop = [0, 1, 0, 0, 0, 0;
    0, 0, -m1*g/MM, 0, -m2*g/MM, 0;
    0, 0, 0, 1, 0, 0;
    0, 0, -(MM+m1)*g/(MM*l1), 0, -m2*g/(MM*l1), 0;
    0, 0, 0, 0, 0, 1;
    0, 0, -m1*g/(MM*l2), 0, -(MM+m2)*g/(MM*l2), 0];

%B matrix
B_open_loop = [0; 1/MM; 0; 1/(MM*l1); 0; 1/(MM*l2)];

% Smallest Output Vector
C_open_loop = [1,0,0,0,0,0];

% LQR
% Selection of Q and R:
Q = [1000 0 0 0 0 0;
    0 1000 0 0 0 0;
    0 0 1000 0 0 0;
    0 0 0 1000 0 0
    0 0 0 0 1000 0
    0 0 0 0 0 1000];
R = 0.00001; % GOOD R

% Designing LQR Controller:
[K, P, EigenValues] = lqr(A_open_loop, B_open_loop, Q, R);

% Setting up LQR System
LQR_System = ss(A_open_loop,[B_open_loop B_open_loop],C_open_loop,[zeros(1,1) zeros↙
(1,1)]);

% Define covariance of disturbance (ensure gaussian)
disturbance = 0.01;
% Define covariance of noise (ensure gaussian)
noise = 1;

% Obtaining L from Kalman Filter
```

```
[~,L,~] = kalman(LQR_System, disturbance, noise, [], 1, 1);

% Forming Closed Loop System
A_closed_loop = [A_open_loop-B_open_loop*K B_open_loop*K;zeros(size(A_open_loop))
A_open_loop-L*C_open_loop];
B_closed_loop = zeros(12,1);
C_closed_loop = [C_open_loop zeros(size(C_open_loop))];
D = 0;

%Use state space function to set up LQG System
LQG_System = ss(A_closed_loop, B_closed_loop, C_closed_loop, D);

% Setting Simulation Conditions
timevector = 0:0.1:200;
X_0 = [3; 0; pi/4; 0; pi/4; 0; 0; 0; 0; 0; 0; 0];
F = zeros(size(timevector));

% Simulating
[Y,~,X] = lsim(LQG_System,F,timevector, X_0);

%% Plotting
% State Variables
figure()
subplot(6,1,1);
plot(timevector,X(:,1),'k')
ylabel('X (m)')
xlabel('Time (s)')
grid on;

subplot(6,1,2);
plot(timevector,X(:,2),'k')
grid on;
ylabel('X Dot (m/s)')
xlabel('Time (s)')

subplot(6,1,3);
plot(timevector,X(:,3),'r')
grid on;
ylabel('Theta 1 (rad)')
xlabel('Time (s)')

subplot(6,1,4);
plot(timevector,X(:,4),'r')
grid on;
ylabel('Theta 1 Dot (rad/s)')
xlabel('Time (s)')

subplot(6,1,5);
plot(timevector,X(:,5),'b')
grid on;
ylabel('Theta 2 (rad)')
```

```
xlabel('Time (s)')

subplot(6,1,6);
plot(timevector,X(:,6),'b')
grid on;
ylabel('Theta 2 Dot (rad/s)')
xlabel('Time (s)')

% Output
figure()
plot(timevector,Y(:,1),'k')
ylabel('X (m)')
xlabel('Time (s)')
title('X(t) vs. Time Under LQG Control')
grid on;
```