Lab 2: Phase Plane Analysis of an Inverted Pendulum with LQR Control

Maxwell Cobar, Clifton Huett, and Carlos Montalvo

ME 494 – Nonlinear Systems Dynamics and Controls

February 17, 2020

1 Introduction:

A phase portrait gives a graphical representation of a dynamic system in the phase plane. The graph is made up of several different curves that represent the various initial conditions that may be present in the system. These curves show the trajectories of the dynamical system as the system tends towards an equilibrium point. An equilibrium point is a state at which the system will tend towards and remain forever [2]. In this lab, the motion of an inverted pendulum was simulated and represented with a phase portrait. The phase portrait depicts the dynamics of the system at different initial conditions as the system fall into equilibrium points. A Linear Quadratic Regulator (LQR) controller is then used to stabilize the system. An LQR controller is a controller that minimizes the cost function with weighting factors. The cost function is the sum of the deviation of key measurements. Hence, an LQR controller finds the best setting for the controller to minimize the undesired deviations of a system [1].

2 Math Model:

Before the inverted pendulum could be simulated, the equation of motion (EOM) for an inverted pendulum needed to be found. The EOM for an inverted pendulum was found online and is as followed:

$$\ddot{\theta} - \frac{g}{L} sin\theta = \frac{T}{mL^2} \qquad (1)$$

With the EOM found, the equation was then put into affine form and is shown below:

$$\begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} \dot{\theta} \\ \frac{g}{L} sin\theta \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{mL^2} \end{bmatrix} T \qquad (2)$$

The system is then linearized, and the equilibrium points are found. The inverted pendulum has equilibrium points of $0, \pi, 2\pi, 3\pi, \ldots. n\pi$. The system is linearized at each equilibrium point to find if the system is stable at these points. The eigenvalues of the linear system found at the equilibrium points determine if the system is stable. The linearization with Taylor Series is shown below:

$$\ddot{\theta} = y$$

$$y \approx y(\theta_0) + \frac{dy}{d\theta}|_{\theta_0} \frac{\theta - \theta_0}{1!}$$

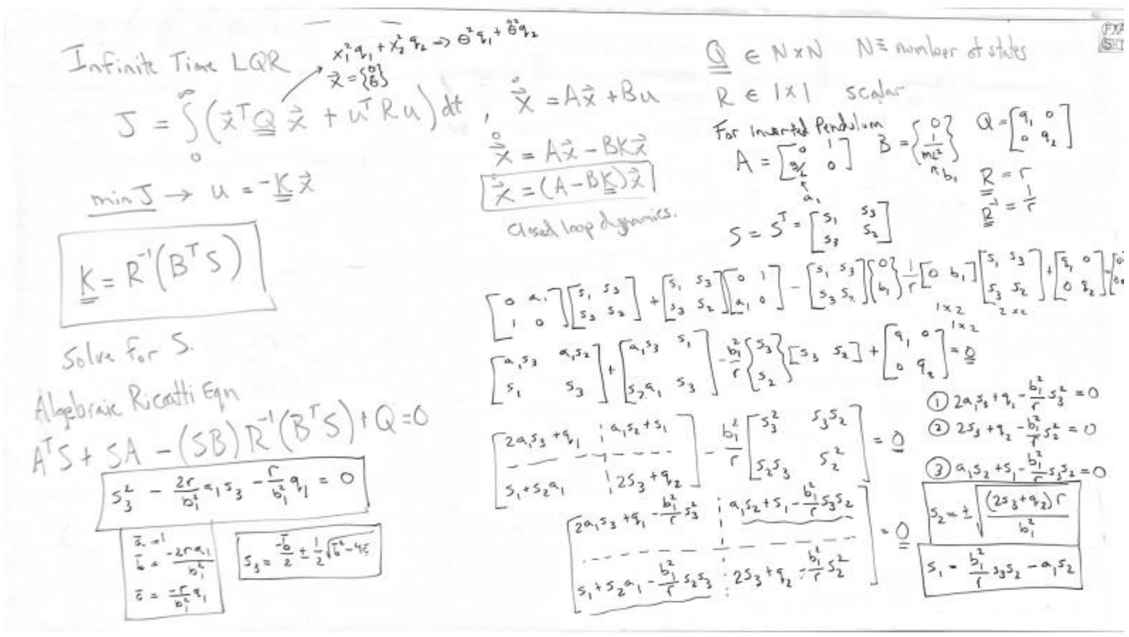
$$\ddot{\theta} \approx \frac{T}{mL^2} + \frac{g}{L}(\theta - \theta_0)$$

$$\ddot{\theta} \approx \frac{T}{mL^2} + \frac{g}{L}(\theta) - \frac{g}{L}(\theta_0)$$

$$@\ \theta_0 = 0: \ddot{\theta} \approx \frac{g}{L}(\theta) \qquad (3)$$

$$@\ \theta_0 = \pi: \ddot{\theta} \approx \frac{g}{L}(\theta) - \frac{g}{L}(\pi) \qquad (4)$$

With the system linearized and the EOM in affine form, the LQR controller needed to be derived in order to code the controller. However, an LQR controller is a function already in MATLAB and so the derivation was not needed to complete the lab. The derivation was still added because it was done during class. The LQR controller derivation is as shown in Fig 1.



Infinite Time LQR

$x_1^2 q_1 + x_2^2 q_2 \Rightarrow \theta^2 q_1 + \dot{\theta}^2 q_2$

$\vec{x} = \begin{Bmatrix} \theta \\ \dot{\theta} \end{Bmatrix}$

$$J = \int_0^\infty \left(\vec{x}^T \underline{\underline{Q}} \vec{x} + u^T R u \right) dt \quad , \quad \dot{\vec{x}} = A\vec{x} + Bu$$

$\underline{\min J} \rightarrow u = -\underline{\underline{K}} \vec{x}$

$$\boxed{\underline{\underline{K}} = R^{-1}(B^T S)}$$

Solve for S.

Algebraic Ricatti Eqn

$$A^T S + SA - (SB)R^{-1}(B^T S) + Q = 0$$

$$\boxed{s_3^2 - \frac{2r}{b_1^2} a_1 s_3 - \frac{r}{b_1^2} q_1 = 0}$$

$$\boxed{\begin{aligned} \bar{a} &= 1 \\ \bar{b} &= \frac{-2 r a_1}{b_1^2} \\ \bar{c} &= \frac{-r}{b_1^2} q_1 \end{aligned}} \quad \boxed{s_3 = \frac{-\bar{b}}{2} \pm \frac{1}{2}\sqrt{\bar{b}^2 - 4\bar{c}}}$$

$\dot{\vec{x}} = A\vec{x} - BK\vec{x}$

$\boxed{\dot{\vec{x}} = (A - B\underline{\underline{K}})\vec{x}}$

Closed loop dynamics.

$\underline{\underline{Q}} \in N \times N \quad N \equiv$ number of states

$R \in 1 \times 1$ scalar

For Inverted Pendulum

$$A = \begin{bmatrix} 0 & 1 \\ \underset{a_1}{g/L} & 0 \end{bmatrix} \quad B = \begin{Bmatrix} 0 \\ \underset{b_1}{\frac{1}{mL^2}} \end{Bmatrix} \quad Q = \begin{bmatrix} q_1 & 0 \\ 0 & q_2 \end{bmatrix}$$

$\underline{\underline{R}} = r$

$\underline{\underline{R}}^{-1} = \frac{1}{r}$

$$S = S^T = \begin{bmatrix} s_1 & s_3 \\ s_3 & s_2 \end{bmatrix}$$

$$\begin{bmatrix} 0 & a_1 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} s_1 & s_3 \\ s_3 & s_2 \end{bmatrix} + \begin{bmatrix} s_1 & s_3 \\ s_3 & s_2 \end{bmatrix}\begin{bmatrix} 0 & 1 \\ a_1 & 0 \end{bmatrix} - \begin{bmatrix} s_1 & s_3 \\ s_3 & s_2 \end{bmatrix}\begin{Bmatrix} 0 \\ b_1 \end{Bmatrix}\frac{1}{r}\underset{1\times 2}{\begin{bmatrix} 0 & b_1 \end{bmatrix}}\underset{2\times 2}{\begin{bmatrix} s_1 & s_3 \\ s_3 & s_2 \end{bmatrix}} + \begin{bmatrix} q_1 & 0 \\ 0 & q_2 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} a_1 s_3 & a_1 s_2 \\ s_1 & s_3 \end{bmatrix} + \begin{bmatrix} a_1 s_3 & s_1 \\ s_2 a_1 & s_3 \end{bmatrix} - \frac{b_1^2}{r}\begin{Bmatrix} s_3 \\ s_2 \end{Bmatrix}\overset{1\times 2}{\begin{bmatrix} s_3 & s_2 \end{bmatrix}} + \begin{bmatrix} q_1 & 0 \\ 0 & q_2 \end{bmatrix} = \underline{\underline{0}}$$

$$\begin{bmatrix} 2a_1 s_3 + q_1 & a_1 s_2 + s_1 \\ s_1 + s_2 a_1 & 2s_3 + q_2 \end{bmatrix} - \frac{b_1^2}{r}\begin{bmatrix} s_3^2 & s_3 s_2 \\ s_2 s_3 & s_2^2 \end{bmatrix} = \underline{\underline{0}}$$

$$\begin{bmatrix} 2a_1 s_3 + q_1 - \frac{b_1^2}{r}s_3^2 & \underbrace{a_1 s_2 + s_1 - \frac{b_1^2}{r}s_3 s_2} \\ \underbrace{s_1 + s_2 a_1 - \frac{b_1^2}{r}s_2 s_3} & 2s_3 + q_2 - \frac{b_1^2}{r}s_2^2 \end{bmatrix} = \underline{\underline{0}}$$

① $2a_1 s_3 + q_1 - \frac{b_1^2}{r}s_3^2 = 0$

② $2s_3 + q_2 - \frac{b_1^2}{r}s_2^2 = 0$

③ $a_1 s_2 + s_1 - \frac{b_1^2}{r}s_3 s_2 = 0$

$$\boxed{s_2 = \pm\sqrt{\frac{(2s_3 + q_2)r}{b_1^2}}}$$

$$\boxed{s_1 = \frac{b_1^2}{r}s_3 s_2 - a_1 s_2}$$

Figure 1: LQR derivation

3 Simulation Results:

A phase portrait was created for the inverted pendulum using the EOM of the inverted pendulum. From Fig 2, two equilibrium points at (-3π,0) and (3π,0) were shown. The trajectory of multiple initial conditions was plotted to show how the system tends towards its equilibrium points. An asymptote can also be seen running diagonally through (0,0). If the initial condition started on the left side, the trajectory tended towards the equilibrium point at (-3π,0), and if the initial condition started on the right side, the trajectory tended towards the equilibrium point at (3π,0).

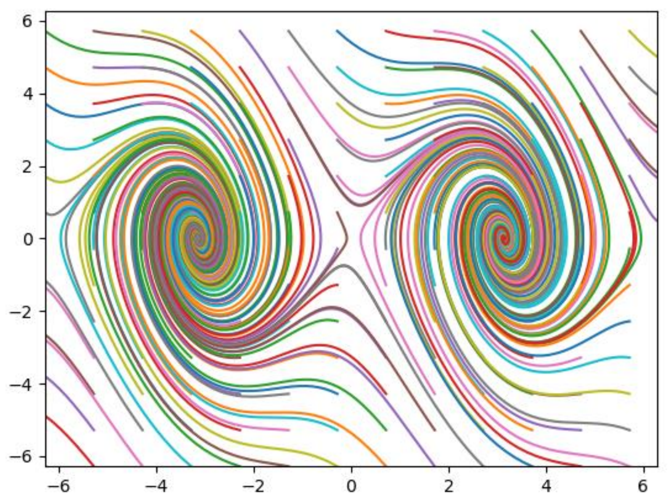


Figure 2: Nonlinear Phase Portrait of Inverted Pendulum

After the phase portrait was plotted, the system response was plotted to show the dynamics of the system. From Fig 3, the system dynamics of the inverted pendulum are shown. The system is shown to stabilize at π steadying out at the final value of 3.14. The system is shown to have some overshoot and is underdamped, however the system reaches its final value with no steady state error.

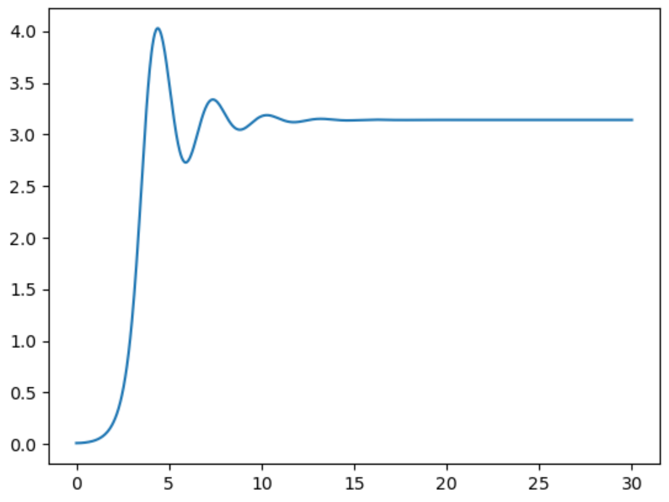


Figure 3: Inverted Pendulum Stabilizing at π

The inverted pendulum is then controlled with an LQR controller. The system response with the LQR controller is shown in Fig 4. The system decays to zero as the inverted pendulum stabilizes. The LQR controller allows for control of the system and allows the inverted pendulum to be critically damped and reach a value determined by the engineer in this case 0.0.

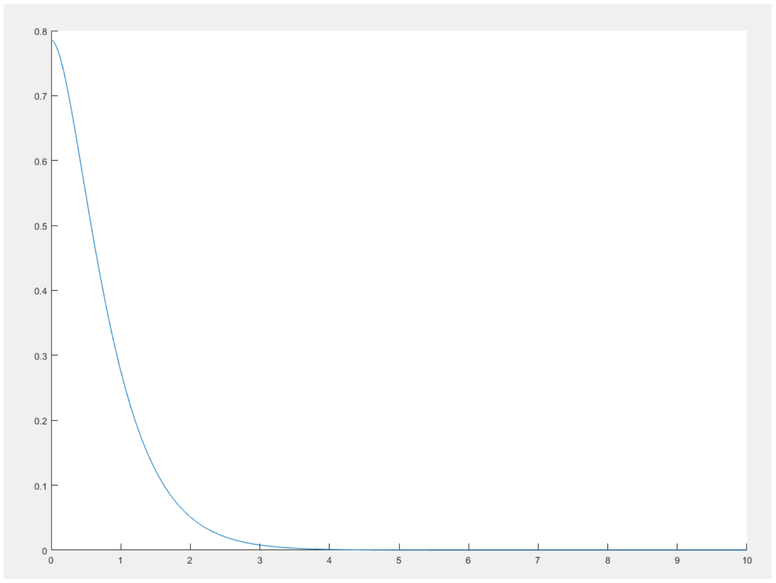


Figure 4: Inverted Pendulum through LQR Controller

4 Conclusion:

In this lab, the motion of an inverted pendulum was simulated and represented with a phase portrait. The equilibrium points were determined to be at nπ where n is an integer. From the phase portrait seen in Fig 2, the inverted pendulum was shown to decay into its equilibrium points despite its initial conditions, and the system response shown in Fig 3 was shown to be underdamped and stabilize as it reached its equilibrium points. The inverted pendulum was then controlled using an LQR controller. From Fig 4, the system is shown to be critically damped and stabilize at the determined value of 0.0.

5 Acknowledgements:

We would like to acknowledge Dr. Carlos Montalvo, James Helton, and Slater Dozier. Dr. Carlos Montalvo helped to derive the LQR controller used to stabilize the inverted pendulum. James Helton and Slater Dozier helped to develop the code used to find the phase portrait and equilibrium points as well as the code to use the LQR controller to stabilize the system.

6 Appendix:

6.1 Code for Phase Portrait with 2 Nonlinear Equilibrium Points:

```
###Integrate an ordinary differential equation
#in MATLAB that's using the function ode45.
#in Python we're going to use the Scipy toolbox and odeint
import numpy as np
import matplotlib.pyplot as plt
import scipy.integrate as I
import control as ctl
m = 1.0
g = 9.81
L = 2
c = 1
def Controller(xstate,t):
    thetac = 0.0
    thetadotc = 0.0
    theta = xstate[0]
    thetadot = xstate[1]
    error = thetac-theta
    errordot = thetadotc-thetadot
    kp = 100.0
    kd = 40.0
    T = kp*error + kd*errordot
    T = 0
    return T
def Derivatives(xstate,t):
    global m,g,L,c
    theta = xstate[0]
    thetadot = xstate[1]
    T = Controller(xstate,t)
    thetadbldot = g/L*np.sin(theta) + T/(m*L**2) - c*thetadot
    xstatedot = np.asarray([thetadot,thetadbldot])
    return xstatedot
tout = np.linspace(0,30,1000)
xstateinitial = np.asarray([0.01,0])
stateout = I.odeint(Derivatives,xstateinitial,tout)
thetaout = stateout[:,0]
thetadotout = stateout[:,1]
plt.plot(tout,thetaout)
##There are 2 types of eq pts
#0,2pi,4pi
A0 = np.asarray([[0,1],[g/L*np.cos(0),-c]])
[s0,v0] = np.linalg.eig(A0)
print(s0)
print(v0)
#pi,3pi,5pi
Api = np.asarray([[0,1],[g/L*np.cos(np.pi),-c]])
[spi,vpi] = np.linalg.eig(Api)
print(spi)
print(vpi)
plt.figure()
for theta in np.arange(-2*np.pi,2*np.pi,1.0):
    for thetadot in np.arange(-2*np.pi,2*np.pi,1.0):
        xstateinitial = np.asarray([theta,thetadot])
        stateout = I.odeint(Derivatives,xstateinitial,tout)
        thetaout = stateout[:,0]
        thetadotout = stateout[:,1]
        plt.plot(thetaout,thetadotout)
        #plt.pause(0.0001)
        plt.axis([-2*np.pi,2*np.pi,-2*np.pi,2*np.pi])
plt.show()
```

6.2 LQR Controller code:

```
function lqr_inverted_pendulum()
global K A B Q R N
clc
close all
%%%Run the LQR controller
g = 9.81;
L = 2;
m = 1.0;
a1 = g/L;
b1 = 1/(m*L^2);
A = [0 1;a1 0];
B = [0;b1];
for q = 1
    q1 = q;
    q2 = q;
    Q = [q1 0;0 q2]
    r = 1;
    R = r;
    N = 0;
    [K,S,e] = lqr(A,B,Q,R,N);
    K
    S
    e
    [myK,myS,mye] = mylqr(r,b1,a1,q1,q2);
    myK
    myS
    mye
    tspan = [0 10];
    xinitial = [45*pi/180;0];
    [tout,xout] = ode45(@Derivatives,tspan,xinitial);
    hold on
    plot(tout,xout(:,1))
end
function [K,S,e] = mylqr(r,b1,a1,q1,q2)
%%%Solve for s3
abar = 1;
bbar = -2*r*a1/(b1^2);
cbar = -r*q1/(b1^2);
s3 = -bbar/2 + 0.5*sqrt(bbar^2 - 4*abar*cbar);
%s3_1 = -bbar/2 - 0.5*sqrt(bbar^2 - 4*abar*cbar);
%s3_sols = [s3_0 s3_1]
%%%Solve for s2
s2 = sqrt((2*s3*r + q2*r)/b1^2);
%s2_1 = sqrt((2*s3_1*r + q2*r)/b1^2);
%s2_2 = -sqrt((2*s3_0*r + q2*r)/b1^2);
%s2_3 = -sqrt((2*s3_1*r + q2*r)/b1^2);
%s2_sols = [s2_0 s2_1 s2_2 s2_3]
%%%Solve for s1
s1 = b1^2/r*s3*s2-a1*s2;
```

```
%s1_1 = b1^2/r*s3_0*s2_1-a1*s2_1;
%s1_2 = b1^2/r*s3_0*s2_2-a1*s2_2;
%s1_3 = b1^2/r*s3_0*s2_3-a1*s2_3;
%s1_4 = b1^2/r*s3_1*s2_0-a1*s2_0 - these are repeated
%s1_5 = b1^2/r*s3_1*s2_1-a1*s2_1 -- and not needed
%s1_6 = b1^2/r*s3_1*s2_2-a1*s2_2
%s1_7 = b1^2/r*s3_1*s2_3-a1*s2_3
%s1_sols = [s1_0 s1_1 s1_2 s1_3];
%%%%Plug in for S
B = [0;b1];
%for s3 = s3_sols
%    for s2 = s2_sols
%        for s1 = s1_sols
            S = [s1 s3;s3 s2];
            K = (1/r)*(B'*S);
%        end
%    end
%end
A = [0 1;a1 0];
e = eig(A-B*K);
function dxdt = Derivatives(t,x)
global K A B
%%%LQR
u = -K*x;
%%%PID
theta = x(1);
thetadot = x(2);
thetac = 0;
thetacdot = 0;
kp = 100;
kd = 40;
%u = kp*(thetac - theta) + kd*(thetacdot - thetadot);
dxdt = A*x + B*u;
```

6.3 References:

[1] Franklin, G. F., Powell, J. D., and Emami-Naeini, A., *Feedback control of dynamic systems*, Upper Saddle River, NJ: Pearson, 2020.

[2] Slotine, J.-J. E., and Li, W., *Applied nonlinear control*, Taipei: Prentice Education Taiwan Ltd., 2005.

[3] "Inverted pendulum," *Wikipedia* Available: https://en.wikipedia.org/wiki/Inverted_pendulum.