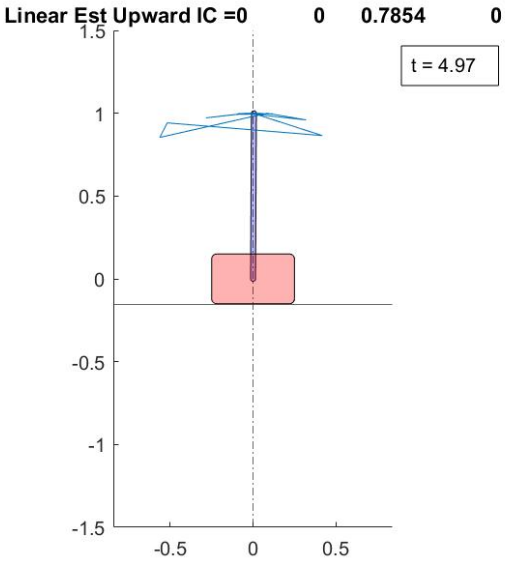
Inverted Pendulum ECE 147B Lab 4



Students: Andrew Yung
Perm Number: 5163191
Email: andrew_yung@ucsb.edu

ECE Dept., UC Santa Barbara

Abstract

This lab focuses on the control of a nonlinear system through state feedback and state estimators. The nonlinear system is an inverted pendulum balanced on a cart moving on a track. With state feedback we simply choose the closed loop poles of our system to directly fit the available voltages the cart motor can take while achieving an optimal settling time. This report reaffirms that linearization about an equilibirum point is an efficient design choice to deal with nonlinearities as long as the states do not deviate form that point too much. The observer in this report demonstrates how all the states in our system may not be available for measurement so we take advantage of the knowing our input, linearized plant, and measured states to create an observer which converges on the actual states of the plant. Overall, linearization, statefeedback and observer estimation are efficient methods of design for a complex nonlinear system, for which all of its states cannot be measured.

1 Introduction

In practice, nonlinear systems are ubiquitous. The physics of the world do not cater to the small detail that linear control is easier than nonlinear control. In addition, with a complex system like an orbiting satellite it would be difficult to measure and control the 30 different states. With linear control we are in a comfortable domain where computation is easily done, so we can approximate a linear model to fit around where our system should operate. Once our system is accessible through linearization, we can use state feedback easily with picking our poles. We can create an observer to model the system's states to save money on 15 extra sensors.

2 Methods

The inverted pendulum on the motor cart is inherently a nonlinear system. So to apply linear state-space methods on this model we must first derive its linearization through equations from classical physics and select an operating point to expand a taylor series to operate around.

2.1 Modeling

Balancing the inverted pendulum via the motor cart is represented by the figure 1

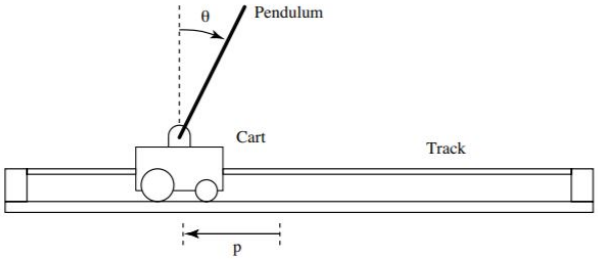


Figure 1: Inverted Pendulum Model

To analyze this model we will create its free body diagram in figure 2

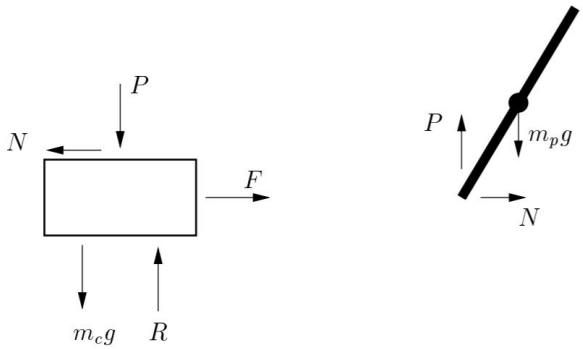


Figure 2: Free Body Diagram for Cart/Pendulum System

From the derivation given in the hand out we yield the following equations 1 and 2.

$$\ddot{p}(M - \frac{m_p l cos^2(\theta)}{L}) = \frac{K_m K_g}{R_r} V - \frac{K_m^2 K_g^2}{Rr^2}\dot{p} - \frac{m_p l g}{L} cos(\theta) sin(\theta) + m_p l sin(\theta)(\dot{\theta}^2) \tag{1}$$

$$\ddot{\theta}(L - \frac{m_p l cos^2(\theta)}{M}) = g sin(\theta) - \frac{m_p l (\dot{\theta})^2}{M} cos(\theta) sin(\theta) - \frac{cos(\theta)}{M}(\frac{K_m K_g}{R_r} V - \frac{K_m^2 K_g^2}{Rr^2}\dot{p}) \tag{2}$$

With $\ddot{p}$ and $\ddot{\theta}$ dependent on their respective velocities we define this system with the states

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} p \\ \dot{p} \\ \theta \\ \dot{\theta} \end{bmatrix} \tag{3}$$

This allows us to rewrite the previous two 2nd-order differential equations as a set of four 1st-order ones (each a function of the state variables and an input variable u equal to the voltage V):

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} x_2 \\ \frac{1}{M - \frac{m_p l cos^2(x_3)}{L}} * (\frac{K_m K_g}{R_r} u - \frac{K_m^2 K_g^2}{Rr^2} x_2 - \frac{m_p l g}{L} cos(x_3) sin(x_3) + m_p l {x_4}^2 sin(x_3)) \\ x_4 \\ \frac{1}{L - \frac{m_p l cos^2(x_3)}{M}} * (g sin(x_3) - \frac{m_p l {x_4}^2}{M} cos(x_3) sin(x_3) - \frac{cos(x_3)}{M}(\frac{K_m K_g}{R_r} u - \frac{K_m^2 K_g^2}{Rr^2} x_2)) \end{bmatrix} \tag{4}$$

2.2 Linearization

All of the control system methods we employ require a system that is linear; otherwise, we will not be able to apply Laplace/z-transforms and obtain a transfer function or state-space representation that we can work with. Looking at the state equations developed in equation 4, we see a variety of nonlinear terms such as $\cos(x_3)\sin(x_3)$ and ${x_4}^2 \sin(x_3)$. The way we get around this is by finding an equilibrium point for all the states (one that yields the zero vector when plugged into equation 4), and approximating the system around that point using a first-order Taylor expansion. For a single-input system, the Taylor approximation has the following form (which represents the line tangent to the curve at the equilibrium point):

$$f(x) = f(x_{eq}) + f'(x_{eq}) * (x - x_{eq}); \tag{5}$$

To extend to multiple independent variables (four in our case), we simply have four more linear terms for each variable. Thus, by choosing the equilibrium point so that $f(x_{eq})$ is 0, we have a purely linear system if we take $x - x_{eq}$, or δx, as the input.

For our model, we would like to linearize around two points; pendulum up (all states equal to 0) and pendulum down ($x_3 = \pi$ and rest of the states 0). After applying the linearization to each state equation, we can rewrite the four equations as a matrix equation of the following form:

$$\begin{bmatrix} \delta\dot{x_1} \\ \delta\dot{x_2} \\ \delta\dot{x_3} \\ \delta\dot{x_4} \end{bmatrix} = A \begin{bmatrix} \delta x_1 \\ \delta x_2 \\ \delta x_3 \\ \delta x_4 \end{bmatrix} + BV \tag{6}$$

The work for calculating all partial derivatives and coming up with the 4x4 matrix A and 4x1 matrix B for pendulum up and down is shown in the appendix. Now that we have linearized the plant, we are ready to start applying control methods.

2.3 Week 1: State Feedback

For the first part of the experiment, we will use a state feedback model that assumes that all the states are available for measurement and can be fed back to the input. The Simulink model we use is shown in figure 3. Note that this model does not linearize the plant as we did in the previous section; however, the linearized plant we came up with will still be useful to generate the feedback matrix K shown in the model

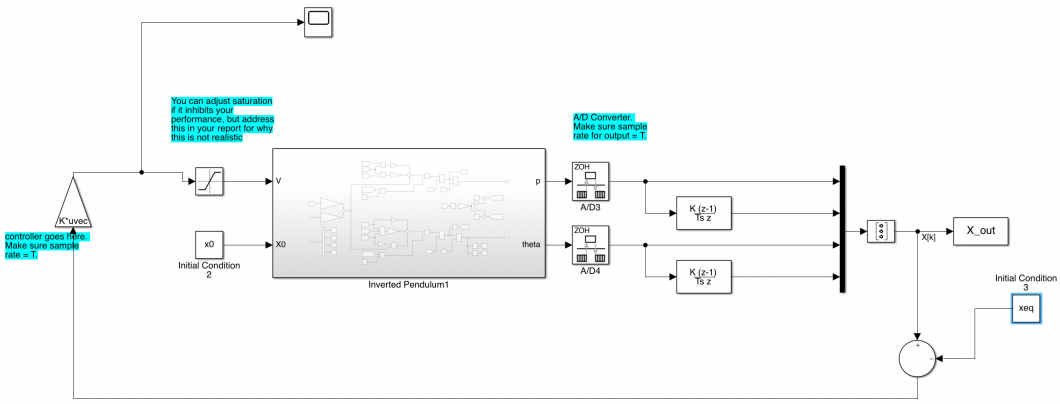


Figure 3: State Feedback Model

The idea with state feedback is that by using a feedback matrix K (of dimension 1x4), we can use MATLAB to help us do direct design. This works because if we feedback x such that u = -Kx, we have the following matrix equations:

$$\dot{x} = Ax + B(-Kx) = (A - BK)x \tag{7}$$

With this model, the closed-loop poles of the system are the eigenvalues of the matrix A-BK. Because solving for eigenvalues and doing the required matrix algebra with any matrices higher that 2x2 is very tedious, we can use MATLAB's "place" command to assist us. By entering four desired poles, A, and B as inputs, MATLAB will output the exact matrix K to achieve the desired poles. There is a slight catch; not every pole combination can be achieved due to the issue of

controllability (not covered in this report). Because of this, in some cases MATLAB will output a K that produces poles as close as possible to the desired ones. An example of how this would be implemented is shown below (note that we first use a zero-order hold conversion for A and B since this best models the actual plant and A-D Converter):

```
T = 0.0005;
theta_0 = pi/4;
%Pendulum Down Case:
x0 = [0 0 theta_0 0];
xeq = [0 0 pi 0];
%Continuous Linearized Matrices
A = [0 1 0 0; 0 -15.14 -3.04 0; 0 0 0 1; 0 -37.23 -31.61 0];
B = [0 3.39 0 8.33].';
C = [1 0 0 0; 0 0 1 0];
D = 0;
sys_c_down = ss(A,B,C,D);

sys_d_down = c2d(sys_c_down,T,'zoh');
[A_d, B_d, C_d, D_d] = ssdata(sys_d_down);

z = [0.99 0.995 0.997 0.998];
K = place(A_d,B_d,z)
pac = eig(A_d-B_d*K) %Display the achieved poles
```

This method of design seems relatively straightforward. The catch is that poles cannot arbitrarily be placed as low or as close to desired as needed because there is a very good chance that the required K matrix to support extremely stable poles (especially closer to the origin on the z-plane) is quite large. This would require the input to also blow up to very high levels, which is not always possible. As a result, a saturation block is included and we measure the inputs before saturation with a scope during simulations to confirm that the magnitude of the input is not going too high above an arbitrary saturation value (chosen as 20 V for now).

To choose poles, we mostly used trial-and-error and found that unless all the poles were quite close to the unit circle (all having magnitude at least 0.99), it was quite difficult to get the input to not blow up. The poles we chose are shown in the next section.

There was one other important addition to the Simulink Model. The plant model outputs position and angle, but does not output the other two states (their velocities). Because we need all the states in order to apply the feedback matrix K, we use a discrete approximation of the derivative which is shown as two blocks in figure 3. This model essentially calculates the difference between the current and previous value of the state and divides by sampling time, effectively approximating instantaneous slope. This model can have problems when noise is introduced, which is why the next section will go into a design that may help deal with this.

2.4 Week 2: Estimator Design

The full state estimator confronts the fact that we may not have all the states such as cart velocity and angle velocity available for measurement. However, knowing the approximate plant from the linearized model, the voltage it is being fed and the position and angle states, we can estimate the missing states unavailable for measurement. From our pre-lab we yielded the following estimator in 4. The *X_obs_0* input connection was substituted with an impulse input at the $\dot{\hat{x}}$ to feed the initial condition.

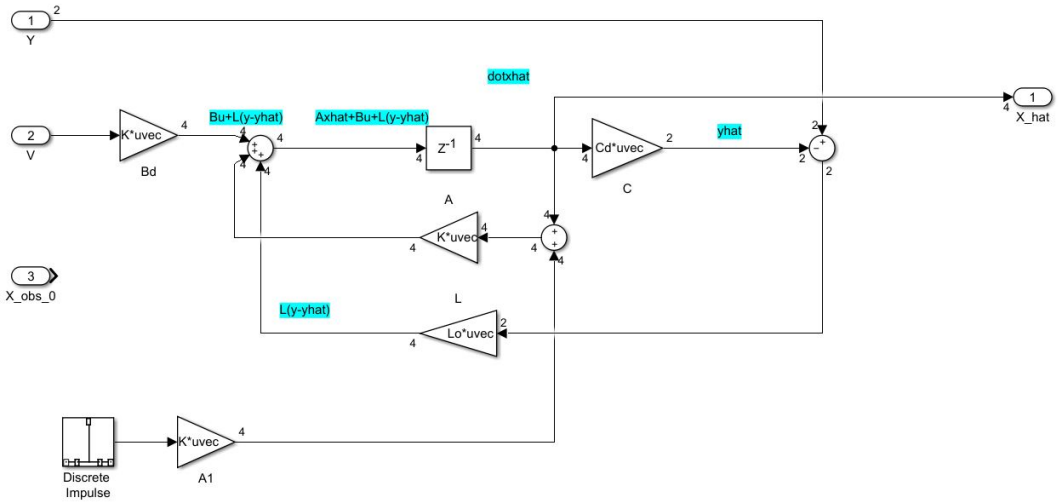


Figure 4: Full State Estimator

2.4.1 Estimating Linearized Model

To ensure that our linear model is valid we will work with the model shown in figure 5

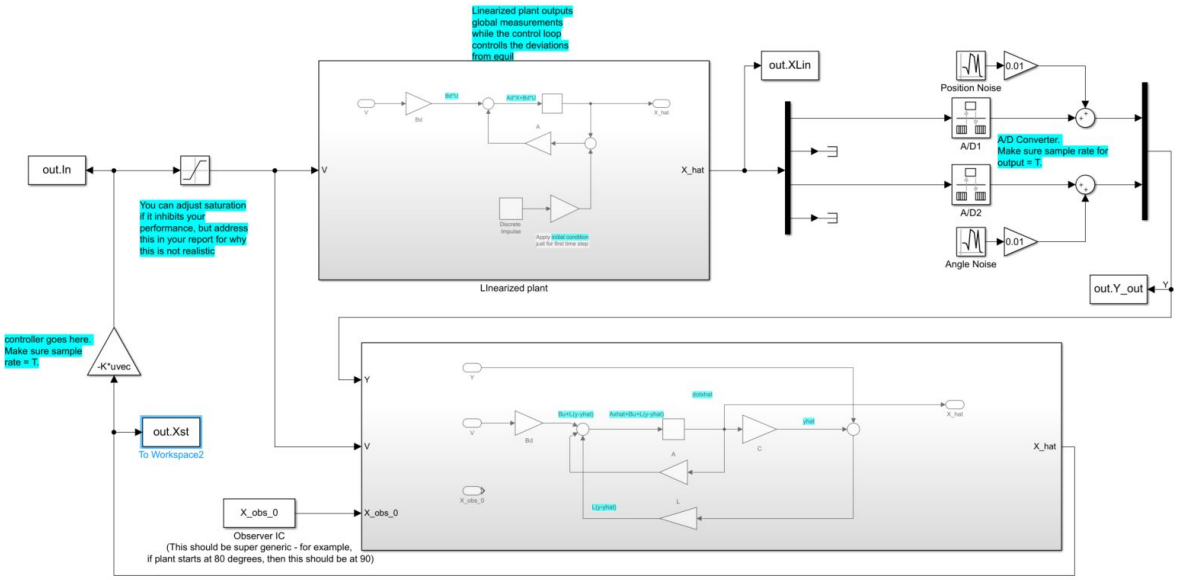


Figure 5: Simulink Linearized Model with Estimator

From this simulation we want to see that our estimator appropriately balances between matching the known states and rejecting noise for the pendulum up and down case.

2.4.2 Estimator and Real Plant

Finally, we apply our estimator to the real plant as seen in figure 6.

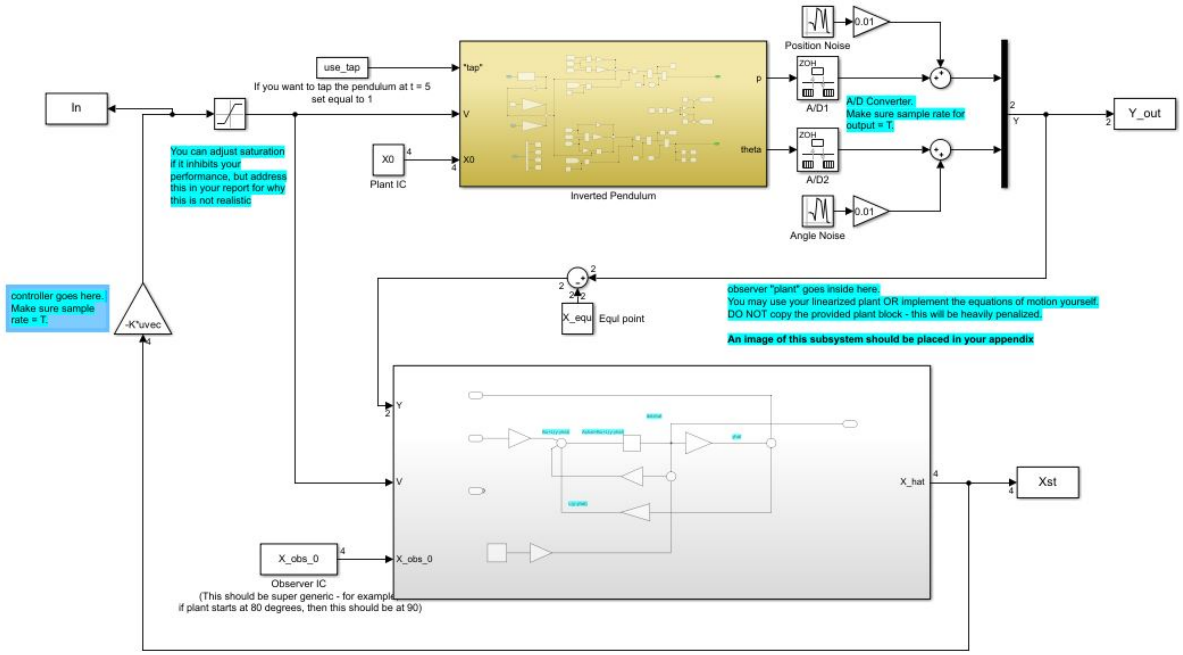


Figure 6: Real Model with Estimator

Through many iterations of trial and error we came up with the following close loop poles and observer poles to best fit the systems in table 2.4.2.

Model	Closed-Loop Poles	Observer Poles
Down Position Linear Est	[0.991 +0.099i, 0.991 - 0.099i, 0.989, 0.979]	.87*Closed Loop Poles
Down Position Real Plant	[0.991 +0.099i, 0.991 - 0.099i, 0.989, 0.979]	.87*Closed Loop Poles
Up Position Linear Est	[0.98+0.099i 0.98-0.099i 0.996 0.99]	.87*Closed Loop Poles
Up Position Real Plant	[0.985+0.1i 0.985-0.1i 0.995 0.996]	0.95*Closed Loop Poles

3 Results

3.1 Week 1: State Feedback

When we run the section of code shown in the methods section (to verify the K matrix we chose gets the poles we want), we obtain the following output:

```
K =

   58.8772   28.8946   35.1148   -8.7732


pac =

    0.9900
    0.9950
    0.9970
    0.9980
```

Since these poles match the poles we entered and the K matrix contains reasonable values, we proceed to test with these values. We started with a small deviation of 0.5 radians from equilibriuma nd obtained the following results for the state outputs shown in figure 7 (primarily noting the angle, or state x3):

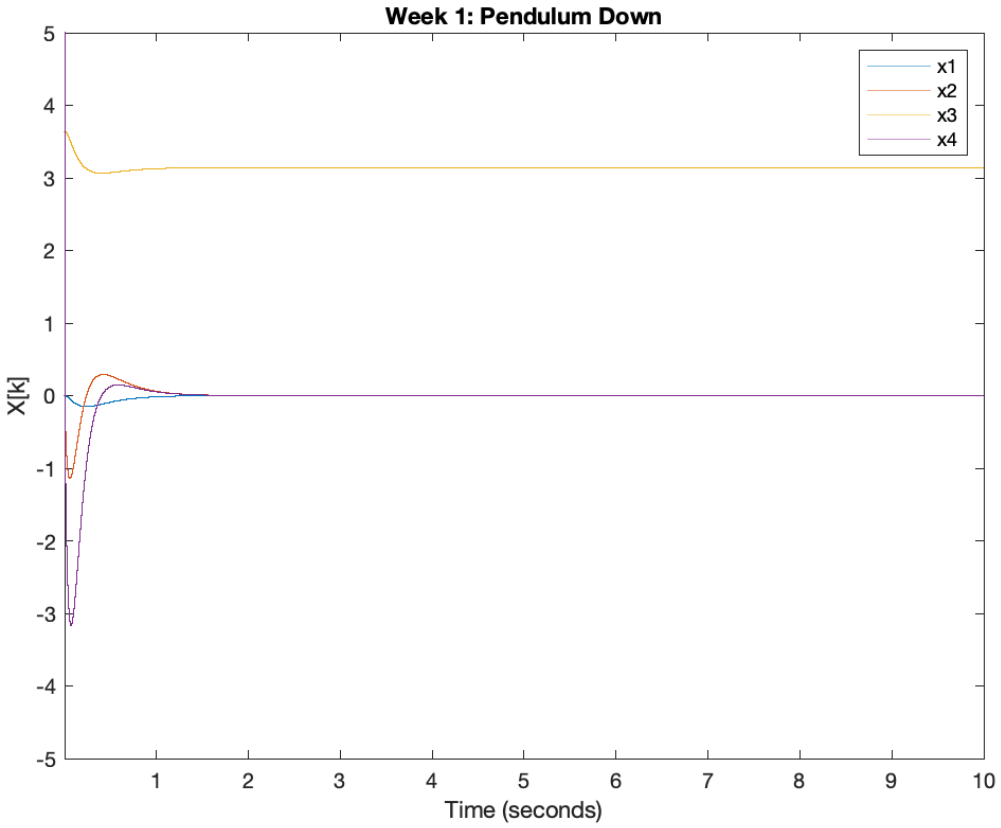


Figure 7: Week 1 Pendulum Down Results: Initial Angle 0.5 rad from Equilibrium

Using the same parameters but setting the initial position at 45°, we have figure 8:

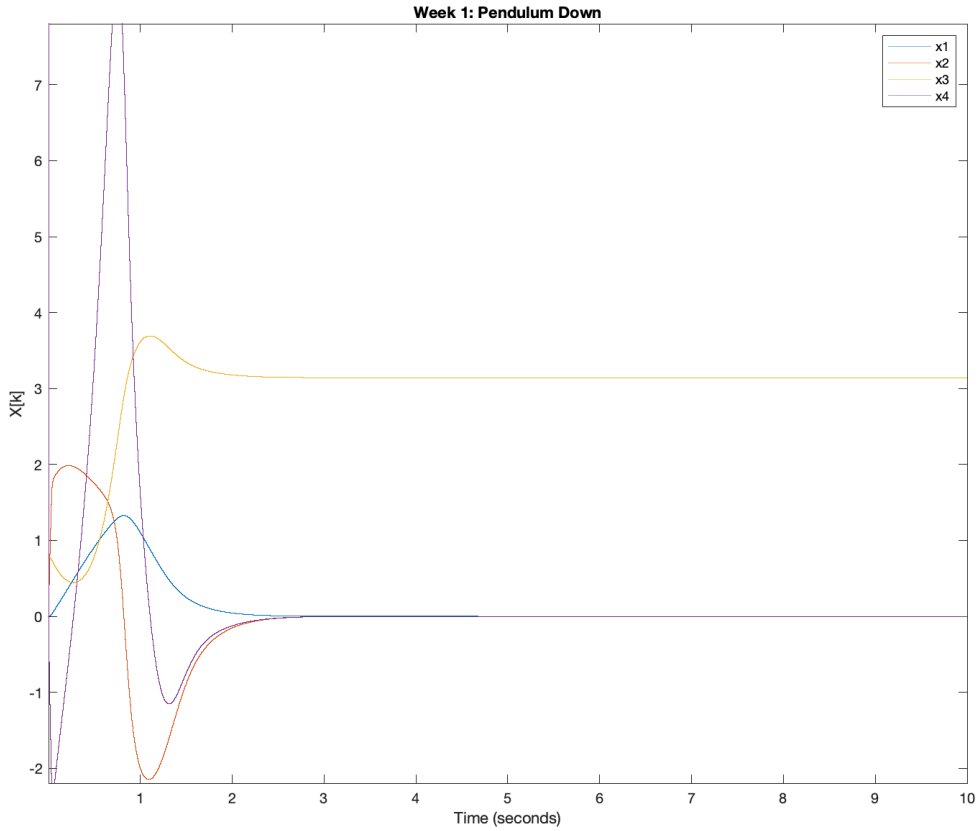


Figure 8: Week 1 Pendulum Down Results: Initial Angle 45 Degrees

Now we want to compare these results to what we would get if there is no feedback. This can be done by adding a line to set K equal to a row vector of 4 0's, getting rid of the feedback factor. Doing so results in the following response in figure 9

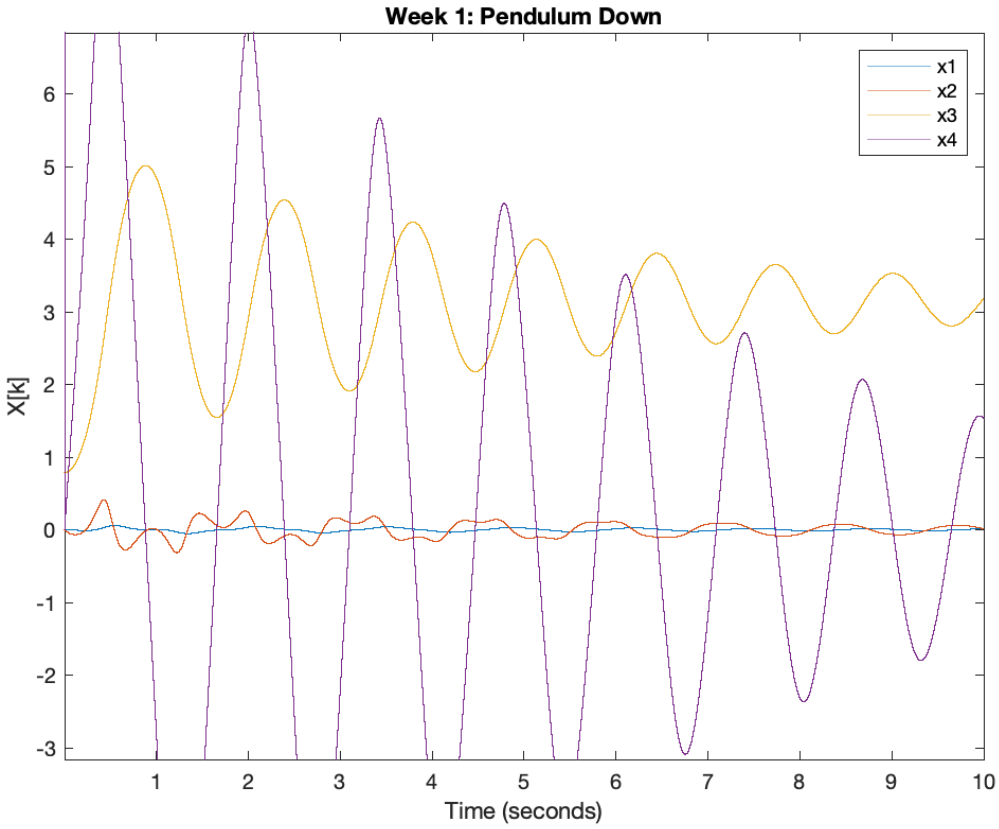


Figure 9: Week 1 Pendulum Down Results (No Feedback): Initial Angle 45 Degrees

We see that there are much more oscillations when no feedback is applied - this is quite easy to imagine intuitively if one releases a pendulum at 45°. With feedback, however, there is barely any oscillatory behavior besides one small overshoot near the beginning.

We will reproduce a similar set of plots, but using the pendulum up case (changing the A and B matrices accordingly from the linearization section). First, we consider the system with the original deviation of 0.5 radians, except now we also add a position offset of 30 cm. This results in the following plot in figure 10:

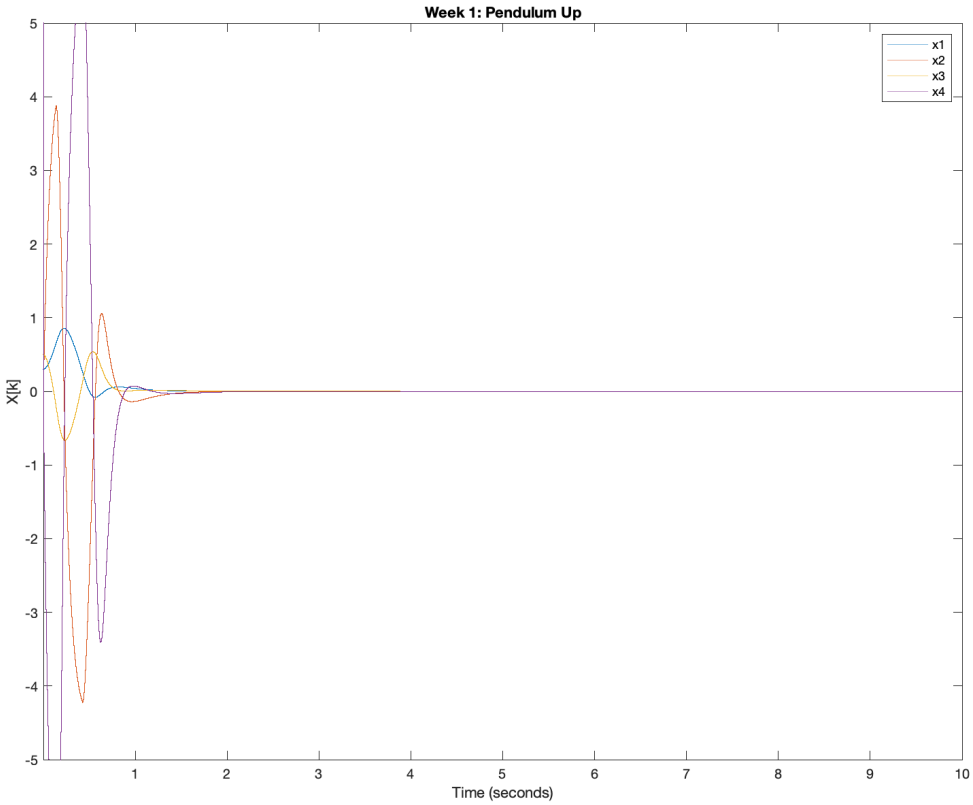


Figure 10: Week 1 Pendulum Up Results: Initial Angle 0.5 rad from Equilibrium

We compare this to no feedback, shown in figure 11:

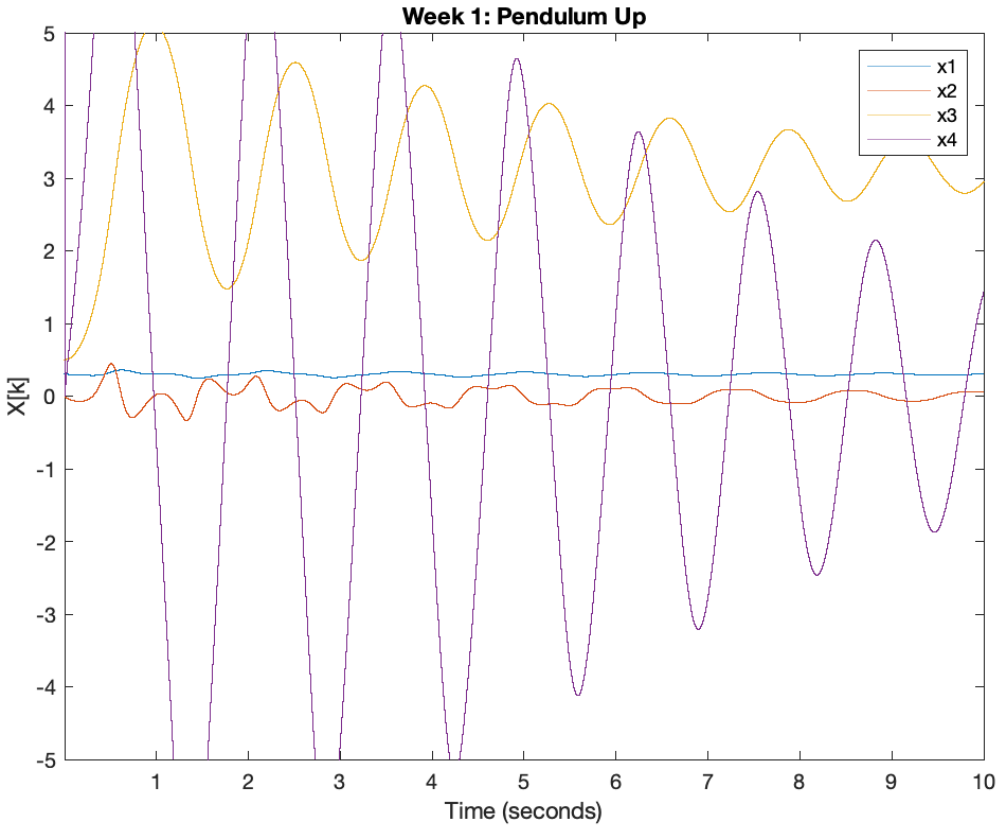


Figure 11: Week 1 Pendulum Up Results (No Feedback): Initial Angle 0.5 Rad

Again, we note that there are many more oscillations as we might expect, but even more noteworthy now is the fact that we don't even oscillate about our original equilibrium point at the top. This seems like a trivial observation since a pendulum obviously will not rise up to the top, but it distinguishes the difference between an unstable equilibrium here versus the stable equilibrium point with the pendulum down. We can see how the input voltage looks when using feedback, shown in figure 12

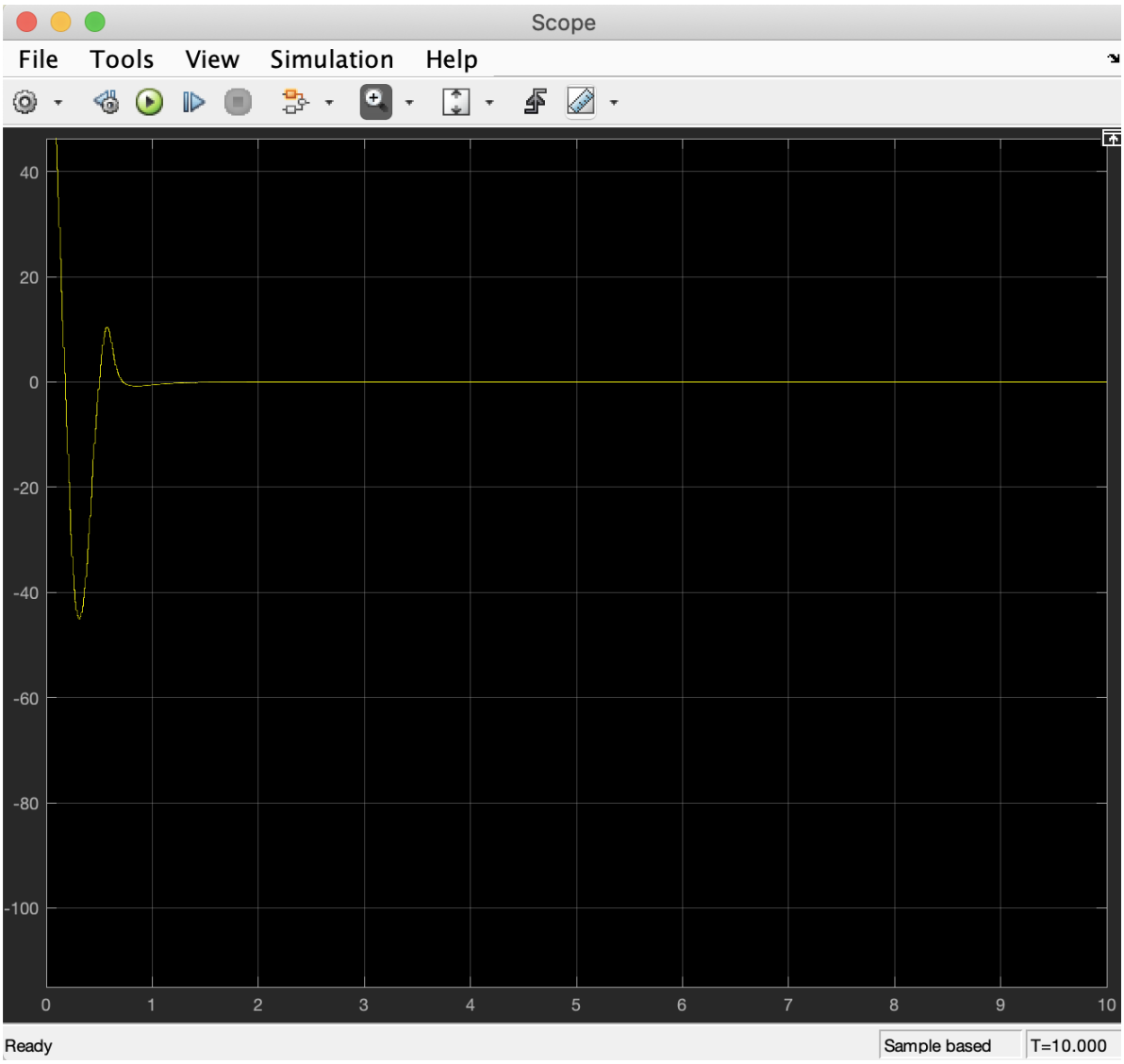


Figure 12: Week 1 Pendulum Up Input Voltage: Initial Angle 0.5 Rad

As we can see, we required a pretty quick change in the input voltage now, which makes sense since the voltage has to help the pendulum to fight gravity more. Although the input voltage had to go above 20, it did not go terribly far above 20 and the saturation block in our Simulink model proved sufficient to mitigate this.

3.2 Week 2: Estimator Design

3.2.1 Estimating Linearized Model

From linearizing our plant with our designed estimator we yield the following figures.

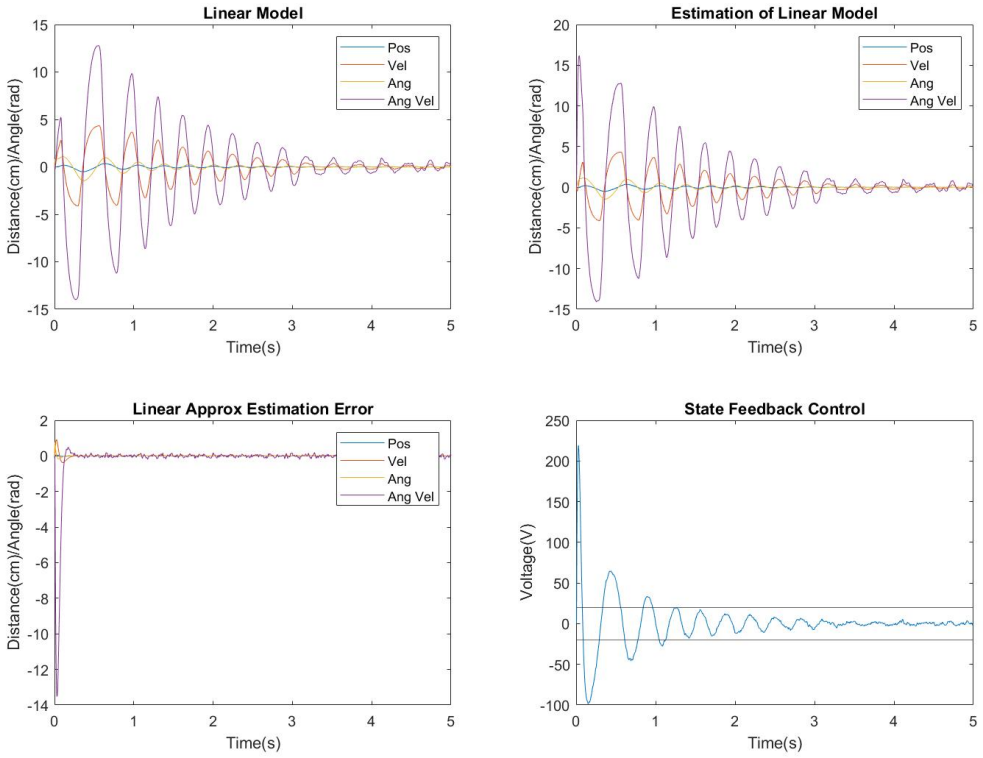


Figure 13: Pendulum Down Linearized Model with Estimator Error and Input

We can see that the estimator quickly converges on the actual states that the linear model outputs. The state feedback only saturates for less than a second before the motor is given feasible voltages. These plots yield the following animated response.

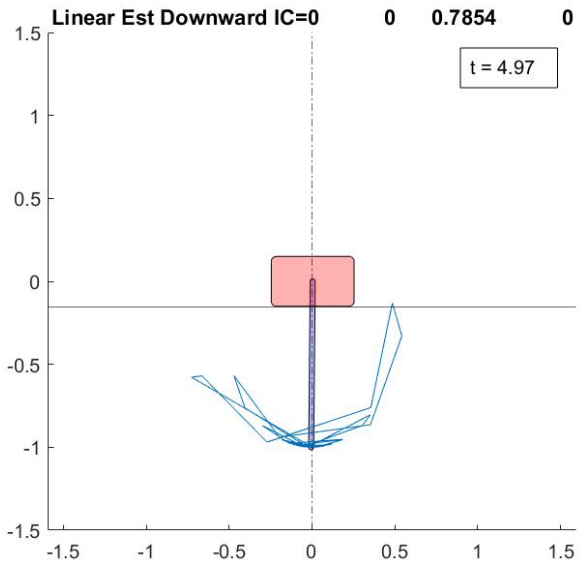


Figure 14: Pendulum DownLinear Model Response

As for the pendulum upright case we yield the following responses.

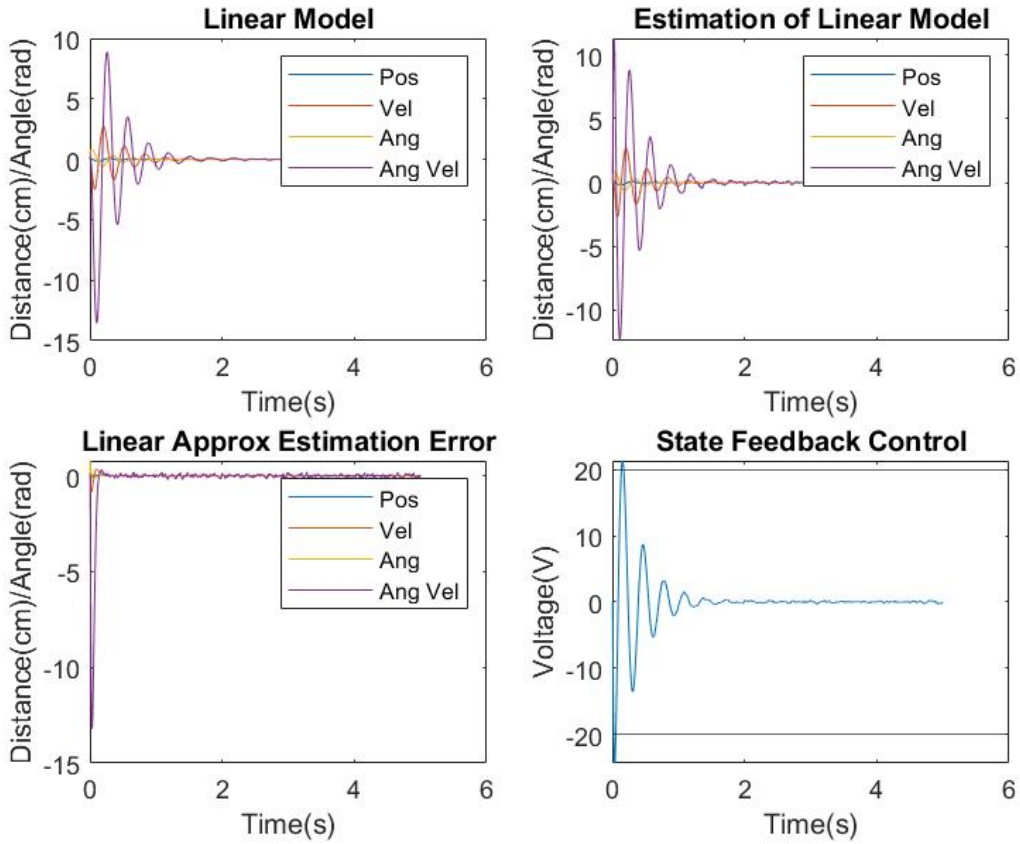


Figure 15: Pendulum Up Linearized Model with Estimator Error and Input

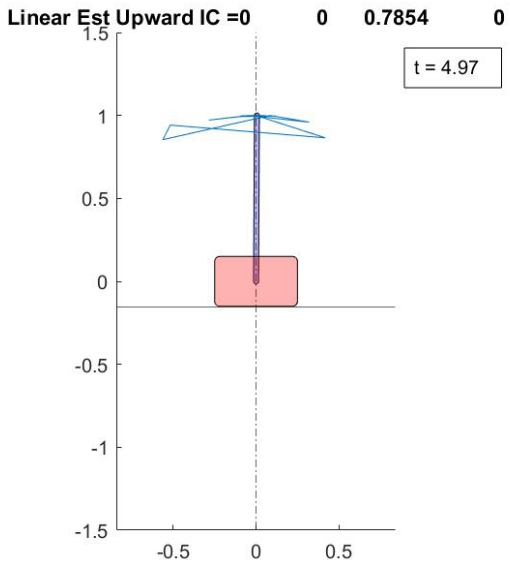


Figure 16: Pendulum Up Linearized Model Response

From figures 15 and 16 we get an ideal estimator. The estimation's error is only seen through the noise injected after the linear model and the initial error decays in less than half a second. The saturation is extremely minimal with only reaching a maximum of 24 volts in the beginning.

3.2.2 Real Model & Estimator

Finally, with the estimator applied to the real cart and pendulum system for the up and down case with a tap we yield the following figures 17 and 19.

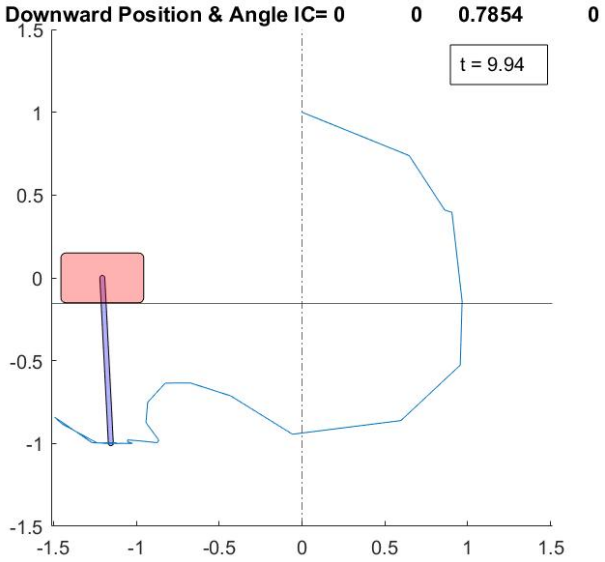


Figure 17: Pendulum Down Tap Real Model Response

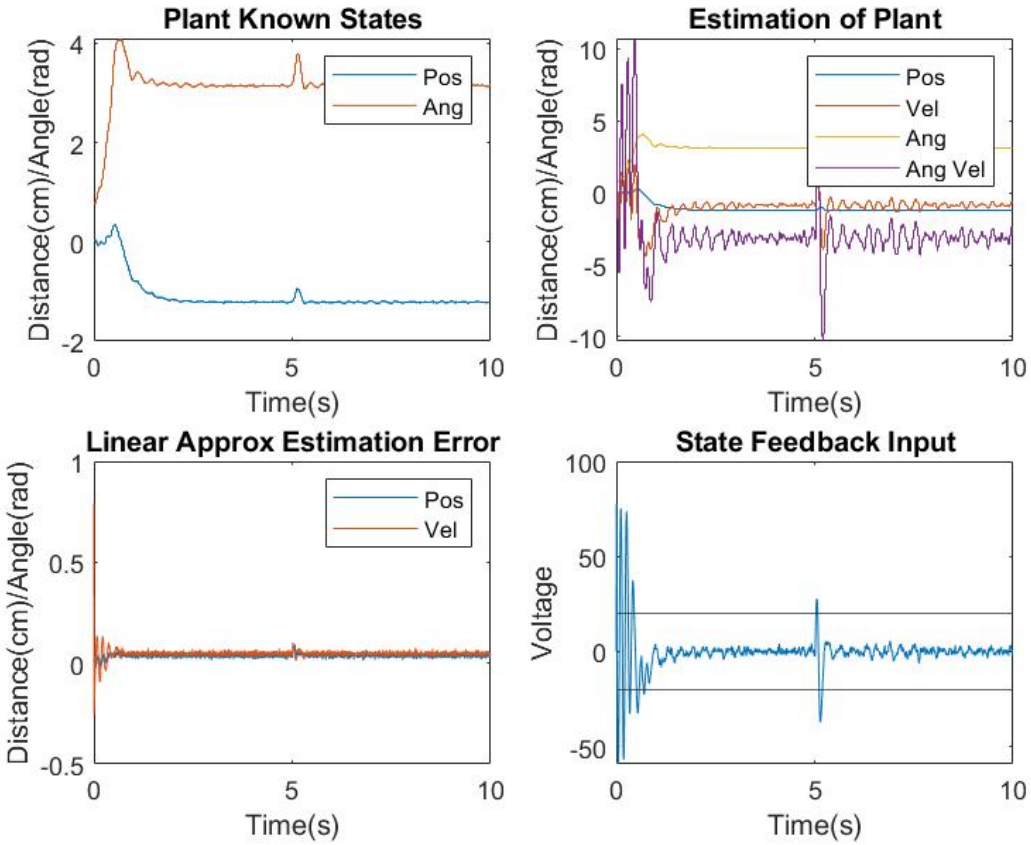


Figure 18: Pendulum Down Tap Real Model Response Error and Input

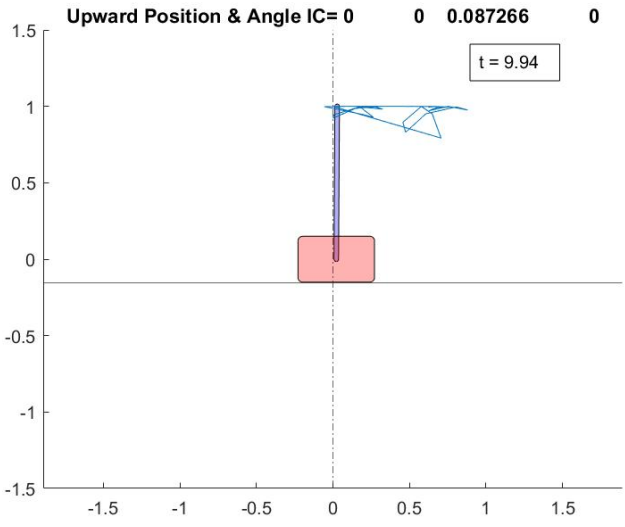


Figure 19: Pendulum Up Real Model Response

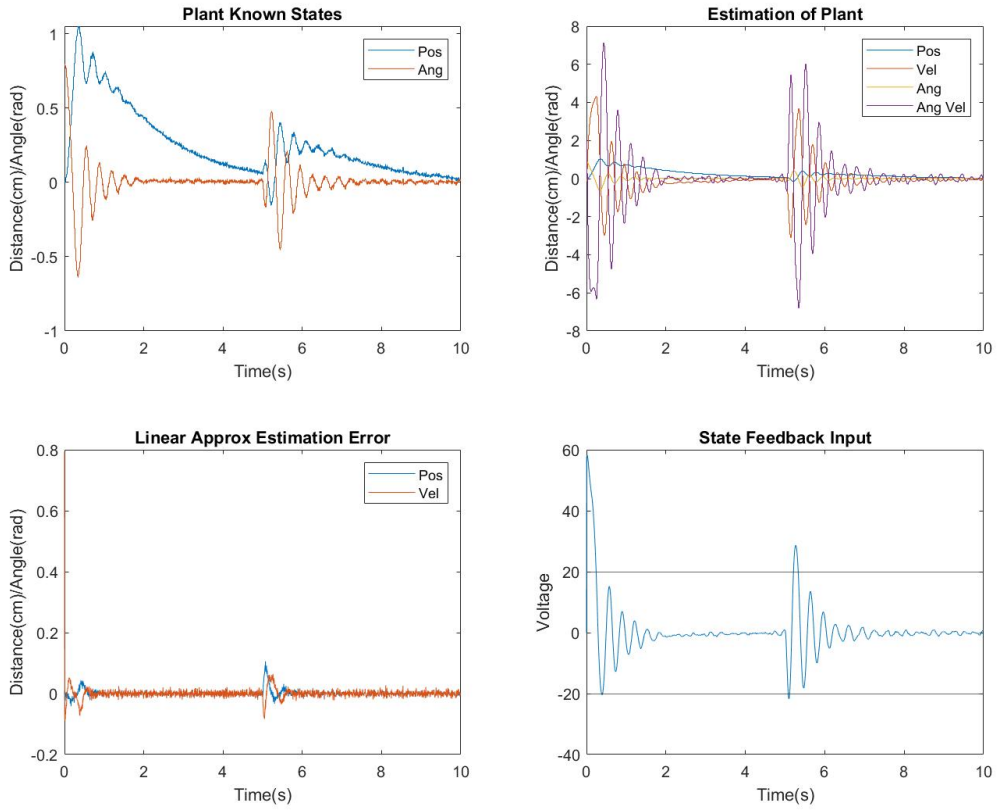


Figure 20: Pendulum Up Tap Real Model Error and Input

As seen from figures 18 and 20 the error in estimate shows that the estimator does not take in too much noise and hovers around and error of zero. The input for the down case in figure 18 does saturate for a second or two but it decays soon after. Both cases are resistant to the tap impulse at T=5seconds.

4 Discussion

4.1 Week 1: Linearization and State Feedback

Based on the results obtained by using state feedback versus no feedback at all, it is clear that state feedback is an effective option for controlling a system - especially if that system has multiple inputs or outputs. As we demonstrated in the previous Lab 2, using transfer function based methods such as root locus/Bode plots and such tend to bring up a lot of complexity for these types of systems, especially when you consider how the variables you're interested in affect each other. In that lab we had attempted to control two separate masses that inherently affected each other's motion, but by only controlling one transfer function at a time. While this can be an effective strategy, especially if you are only concerned with one state or output more than any others, this created a lot of unpredictability if we cared about multiple outputs at once. By using a state feedback model, we can take care of it all with one simple matrix equation and tailor our feedback matrix to account for all states at once. The main drawback is that we seemed to be limited in our

choice of closed-loop poles, and therefore could not obtain as big an array of responses as one could probably conjure more effectively using transfer function methods. It also can be trickier to control more specific aspects of a system, such as wanting a specific final value for one state. However, for systems that generally want to increase their stability and robustness and don't care as much about micromanaging specific parameters, state space proves to be quite efficient.

Another big positive of state feedback is that it allowed us to deal with nonlinear systems. In general, transfer function analysis relies heavily on Laplace/Fourier Transforms and other techniques that are designed specifically for linear systems. While these systems certainly exist, there are many more types of phenomena that exhibit nonlinear behavior that would be much more difficult, if not impossible to control with transfer function methods. As we saw in this lab, we were able to incorporate linearization because of how a state space representation formats state derivatives as functions of all the existing states, allowing us to use the first-order Taylor expansion. Furthermore, we can linearize about any equilibrium point because it is easy to simply add an offset before entering a linearized plant and then remove that offset after. Although a linearized system is meant to approximate the actual system behavior very close to the equilibrium point, we saw in this lab that it can still work quite well for conditions outside of the equilibrium, especially when it's a stable equilibrium. For example, we noticed in the pendulum-down cases that even when we started the pendulum off very high ($\pi/4$ radians from vertical), we still saw pretty good transient behavior that stabilized relatively quickly to the pendulum down point because the natural dynamics of the system want to approach stable equilibria. On the other hand, this tended to be a trickier problem for the pendulum up case since that was an unstable equilibria, so it is much more difficult to attempt to bring the pendulum to that equilibrium point from a farther position (say, with the pendulum initially starting down). Nevertheless, we were still able to get pretty good response behavior if we started the pendulum somewhat close to the up position.

The final point to mention that's desirable about using state feedback is how much easier it can be to implement than transfer function methods because of computer software and programs such as MATLAB and Simulink. Transfer function models require a lot of algebraic analysis or mathematical tricks such as transforms, which are more suited for pencil and paper. While those methods might have been more preferable in the past due to computers not being as computationally efficient or capable, we now live in an age where computers are dominant in technology over calculators or pencil and paper. It is especially convenient to be able to form systems into matrix equations since many computer programs can deal with these very well. Although MATLAB has extra features such as "syms" to deal with certain operations such as continuous derivatives or simplifications, it is not reliable in general especially with how quickly complexity of problems can increase.

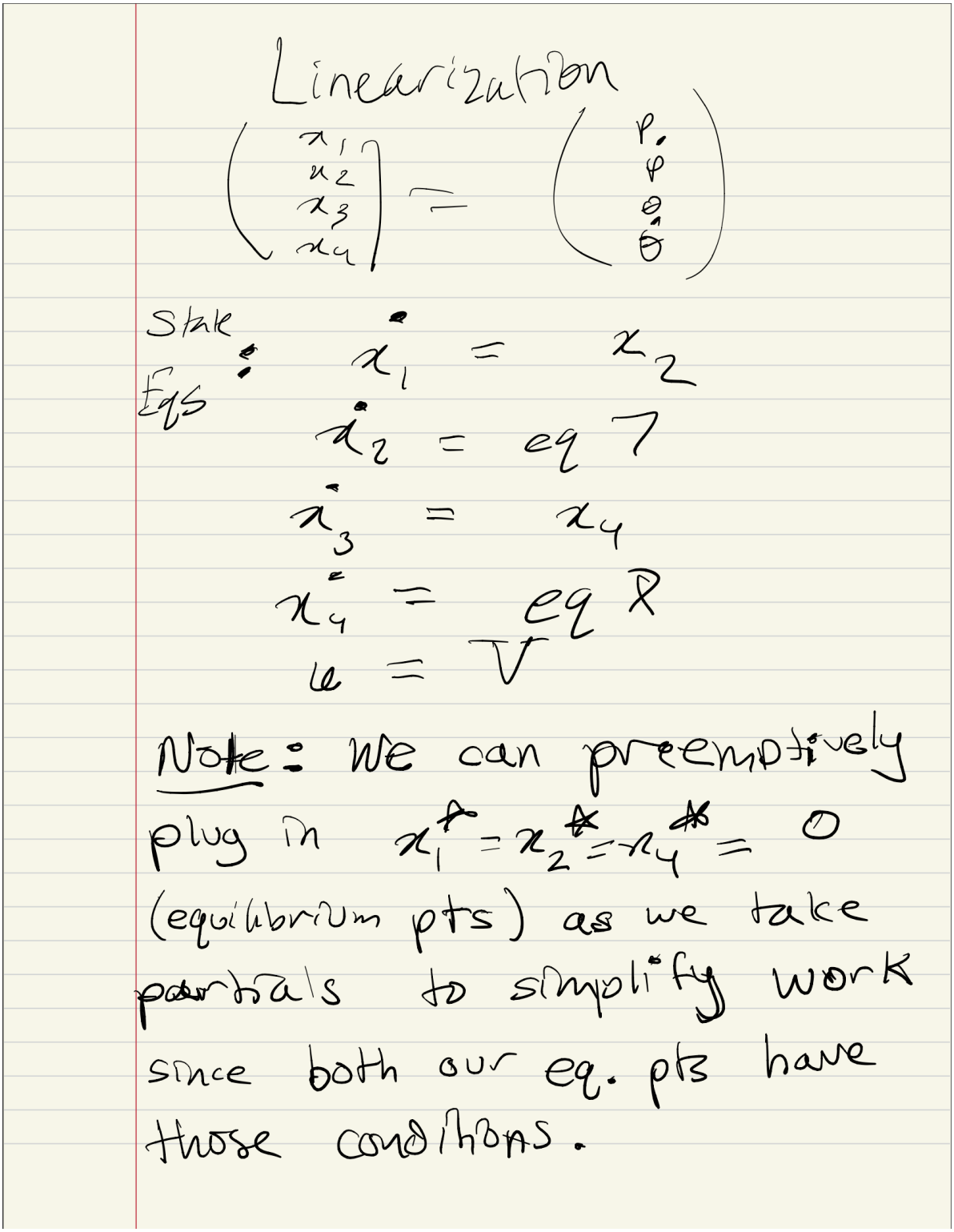
4.2 Week 2: Estimation

Estimation of system states works. As seen in the estimation and error plots the estimator clings to the actual states of the system. Our only job is to ensure that they do not cling too much with the potentially noisy system by adjusting the observer poles to be 2-4 times faster than the designed closed loop poles.

In conclusion, linearization about an operating point gives us the ability to use our linear control techniques. State feedback gives us our entire system's closed loop poles so design is streamlined. The observer reduces the amount of sensors we need to place which in turn reduces the cost of implementing a system with multiple states.

5 Appendix

5.1 Derivation of Linearized System



By plugging in $x_1^* = x_2^* = x_4^* = 0$, we have

$$\dot{\delta x}_1 = 0 + (1)\delta x_2 + 0 + 0$$

$$\dot{\delta x}_3 = 0 + 0 + 0 + (1)\delta x_4$$

$$\dot{\delta x}_2 = 0 \times \delta x_1$$

$$+ \left(\frac{-K_m^2 K_g^2 / R r^2}{M - \frac{m_p l}{L}\cos^2(x_3^*)} \right) \times \delta x_2$$

$$+ \left(\frac{1}{M - \frac{m_p l}{L}\cos^2 x_3^*} \right) \left(-\frac{m_p l g}{L}\cos(2x_3^*) \right) \times \delta x_3$$

$$+ 0 \times \delta x_4$$

$$+ \left(\frac{K_m K_g / R r}{M - \frac{m_p l}{L}\cos^2 x_3^*} \right) \times u$$

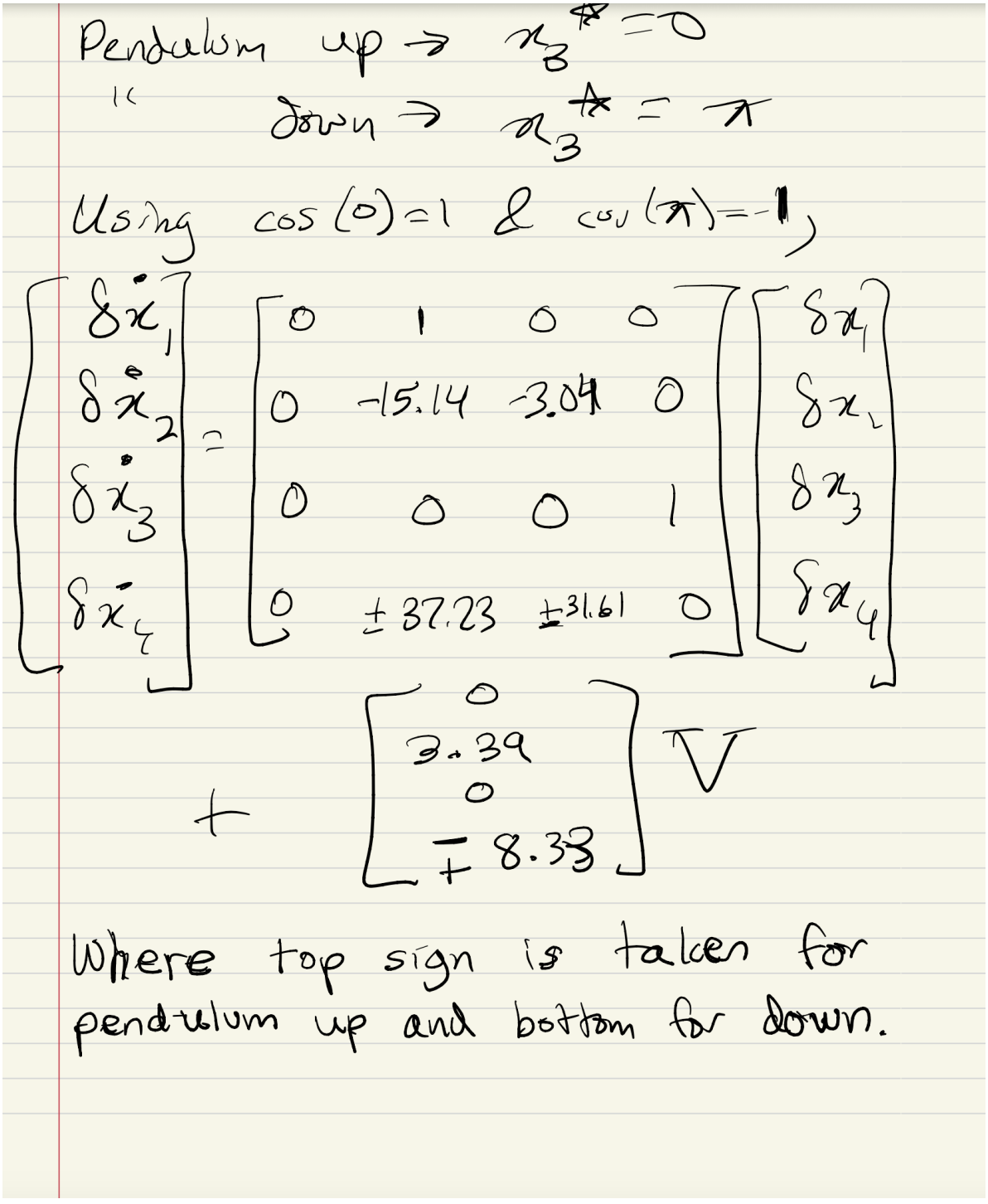
$$\dot{\delta x_4} = 0 \times \delta x_1$$

$$+ \left(\frac{K_m^2 K_g^2 \cos(x_3^*) / MR r^2}{L - \frac{m_p l}{M} \cos^2 x_3^*} \right) \times \delta x_2$$

$$+ \left(\frac{g \cos(x_3^*)}{L - \frac{m_p l}{M} \cos^2 x_3^*} \right) \times \delta x_3$$

$$+ 0 \times \delta x_4$$

$$+ \left(\frac{-K_m K_g \cos x_3^* / MR r}{L - \frac{m_p l}{M} \cos^2 x_3^*} \right) \times u$$



5.2 Week 2: Estimator Code

```
%% Params (TABLE 1)
Km = 0.00767;
Kg = 3.7;
R = 2.6;
r = 0.00635;
```

```
mc = 0.455;
mp = 0.21;
I = 0.00651;
l = 0.305;
g = 9.81;

M = mc + mp;
L = (I + mp*l^2) / (mp * l);
rt=10;
T = 0.005;
%% Week 2 Downward Linear Est
% State Equations
A=[0 1 0 0
    0 -Km^2*Kg^2/(R*r^2*(M-mp*l/L)) -g*mp*l/(L*(M-mp*l/L)) 0
    0 0 0 1
    0 -Km^2*Kg^2/(M*R*r^2*(L-mp*l/M)) -g/(L-mp*l/M) 0];
B=[0; Km*Kg/(R*r*(M-mp*l/L)) ; 0 ;Km*Kg/(M*R*r*(L-mp*l/M))];
C=[1 0 0 0;0 0 1 0];
D=zeros(1,1);
sys=ss(A,B,C,D);
%C2D
dsys=c2d(sys,T,'zoh');
[Ad,Bd,Cd,Dd]=ssdata(dsys);

pz=[0.99+0.099i 0.99-0.099i 0.988 0.97]*1.0007; %Same poles
K=place(Ad,Bd,pz);

pL=[0.99+0.099i 0.99-0.099i 0.988 0.97]*0.87; % Observer poles 2-4 times faster
Lo=place(Ad',Cd',pL)';
X_obs_0=[0;0;0;0];
X0=[0;0;pi/4;0];
X_equ=[0;0;pi;0];
% SIMULATE
out=sim('Lab3BLine.slx',rt);
t=out.tout;
XLin=out.XLin;
Xst=out.Xst;

err=XLin-Xst;
figure(1)
subplot(221);plot(t,XLin);
title('Linear
    Model');xlabel('Time(s)');ylabel('Distance(cm)/Angle(rad)');legend('Pos','Vel','Ang','Ang
    Vel')
subplot(222);plot(t,Xst);
title('Estimation of Linear
    Model');xlabel('Time(s)');ylabel('Distance(cm)/Angle(rad)');legend('Pos','Vel','Ang','Ang
    Vel')
subplot(223);plot(t,err);
title('Linear Approx Estimation Error');
xlabel('Time(s)');ylabel('Distance(cm)/Angle(rad)');legend('Pos','Vel','Ang','Ang Vel')
subplot(224);plot(t,out.In);hold on;yline(20);yline(-20); hold off
title('State Feedback Control');
xlabel('Time(s)');ylabel('Voltage(V)')
```

```
XLin(:,3)=XLin(:,3)+pi;
%cartpole_animate(t,XLin,4,['Linear Est Downward IC=',num2str(X0')]);

%% Week 2 Downward
% SIMULATE
use_tap=1;
X_equ=[0;0];
X0=[0;0;pi/4;0];
sim('Lab3B_sim_starter.slx',rt);
t=Y_out.Time;

figure(4)
subplot(221);plot(Y_out);
title('Plant Known
    States');xlabel('Time(s)');ylabel('Distance(cm)/Angle(rad)');legend('Pos','Ang')
subplot(222);plot(Xst);
title('Estimation of
    Plant');xlabel('Time(s)');ylabel('Distance(cm)/Angle(rad)');legend('Pos','Vel','Ang','Ang
    Vel')
est=[Xst.Data(:,1),Xst.Data(:,3)];
err=Y_out.Data-est;
subplot(223);plot(t,err);
title('Linear Approx Estimation Error');
xlabel('Time(s)');ylabel('Distance(cm)/Angle(rad)');legend('Pos','Vel','Ang','Ang Vel')
subplot(224);plot(t,In);hold on;yline(20);yline(-20); hold off
title('State Feedback Input');xlabel('Time(s)');ylabel('Voltage');
lbe=['Real Plant Tap Down IC=', num2str(X0')];
%cartpole_animate(t,Xst.Data,5,lbe)
%% Week 2 Upward Linear
% State Equations
A=[0 1 0 0
    0 -Km^2*Kg^2/(R*r^2*(M-mp*l/L)) -g*mp*l/(L*(M-mp*l/L)) 0
    0 0 0 1
    0 Km^2*Kg^2/(M*R*r^2*(L-mp*l/M)) g/(L-mp*l/M) 0];
B=[0; Km*Kg/(R*r*(M-mp*l/L)) ; 0 ;-Km*Kg/(M*R*r*(L-mp*l/M))];
C=[1 0 0 0 ; 0 0 1 0];
D=zeros(1,1);
sys=ss(A,B,C,D);
%C2D
dsys=c2d(sys,T,'zoh');
[Ad,Bd,Cd,Dd]=ssdata(dsys);

pz=[0.98+0.099i 0.98-0.099i 0.996 0.99]; %Same poles
K=place(Ad,Bd,pz);

pL=pz*0.87; % Observer poles 2-4 times faster
Lo=place(Ad',Cd',pL)';
X_obs_0=[0;0;0;0];
X0=[0;0;pi/4;0];
X_equ=[0;0];
% SIMULATE
out=sim('Lab3BLine.slx',rt);
```

```
t=out.tout;
XLin=out.XLin;
Xst=out.Xst;

err=XLin-Xst;
figure(1)
subplot(221);plot(t,XLin);
title('Linear
    Model');xlabel('Time(s)');ylabel('Distance(cm)/Angle(rad)');legend('Pos','Vel','Ang','Ang
    Vel')
subplot(222);plot(t,Xst);
title('Estimation of Linear
    Model');xlabel('Time(s)');ylabel('Distance(cm)/Angle(rad)');legend('Pos','Vel','Ang','Ang
    Vel')
subplot(223);plot(t,err);
title('Linear Approx Estimation Error');
xlabel('Time(s)');ylabel('Distance(cm)/Angle(rad)');legend('Pos','Vel','Ang','Ang Vel')
subplot(224);plot(t,out.In);hold on;yline(20);yline(-20); hold off
title('State Feedback Control');
xlabel('Time(s)');ylabel('Voltage(V)')


XLin(:,3)=XLin(:,3);
%cartpole_animate(t,XLin,4,['Linear Est Upward IC =', num2str(X0')]);

%% Week 2 Upward Test
% SIMULATE
use_tap=0;
X_equ=[0;0];
X0=[0;0;5*pi/180;0];
pz=[0.999 0.991 0.992 0.993]; %Same poles
K=place(Ad,Bd,pz);

sim('Lab3B_sim_starter.slx',rt);
t=Y_out.Time;
figure(4)
subplot(221); plot(t,Y_out.Data)
title('Upward Position &
    Angle');xlabel('Time(s)');ylabel('Position(cm)/Angle(rad)');legend('Pos','Ang')

subplot(222);plot(t,err);hold on; yline(0);hold off
title('Est ERROR Upward Position &
    Angle');xlabel('Time(s)');ylabel('Position(cm)/Angle(rad)');legend('Pos','Ang')

subplot(223);plot(t,In);hold on; yline(-20); yline(20);hold off
title('State Feedback Control');xlabel('Time(s)');ylabel('Voltage (V)')


lbe=['Upward Position & Angle IC= ',num2str(X0')];
%cartpole_animate(t,Xst.Data,5,lbe)
```