Abstract

In this, we consider the problem of estimating the states of a system with perspective outputs. We formulate the problem in a deterministic setting by searching for the value of the state that is "most compatible" with the dynamics, in the sense that it requires the least amount of noise to explain the measured output. We show that, under appropriate observability assumptions, the optimal estimate converges globally to the true value of the state and can be used to design output-feedback controllers by using the estimated state to drive a state-feedback controller. We apply these results to the estimation of unobserved states of an Aircraft Model. (majorly Linear Control System)

Contents

Chapter 1

Introduction

1.1 Overview

An observer is a dynamic system that is used to estimate the state of a system or some of the states of a system using the inputs, initial states, and output of that system.

1.2 Motivation of the Research Work

In state-space representation, sometimes all state space variables are not available for measurements, or it is not practical to measure all of them, or it is too expensive to measure all state space variables. In order to be able to apply the state feedback control to a system, all of its state space variables must be available at all times. Also, in some control system applications, one is interested in having information about system state space variables at any time instant. Thus, there is a problem of estimating system state space variables. This can be done by constructing another dynamical system called the observer, connected to the system under consideration, whose role is to produce good estimates of the state space variables of the original system.

Chapter 2

Project Work

2.1 Linear Control Systems

A control system is a system of devices that manages, commands, directs or regulates the behavior of other devices to achieve a desired result. In other words, the definition of a control system can be simplified as a system which controls other systems to achieve a desired state. So, Those control systems which follow the principle of homogenity and additivity are called Linear Control Systems.

Generally, Linear Control Systems are represented as:

$$\frac{dx(t)}{dt} = Ax(t) + Bu(t) \qquad x(t_o) = x_o$$

$$y(t) = Cx(t) + Du(t)$$

Where, x(t) is the state-vector

y(t) is the Output Signal

A, B, C, & D are the matrices of appropriate dimensions

2.2 Observability and Controllability Tests

2.2.1 Observability Test

Observable System: A system is said to be observable if from measurements of output carried over a finite interval of time ($0 \leq t \leq t_f$), state of the system can be determined.

Observability Test: For a system to be completely state observable, the necessary and sufficient condition is that the following $n \times np$ matrix, Q_O has a rank of n.

$$Q_O = [C^T \ A^T C^T \ (A^T)^2 C^T \ \ldots\ldots \ (A^T)^{(n-1)} C^T]$$

2.2.2 Controllability Test

Controllable System: A system is said to be controllable if a control function $u(t)$ can transform the initial state $x(t_o)$ of a system to some desired final state $x(t_f)$ inr a finite interval of time $(t_f - t_o)$; $t_o \geq 0$

Controllability Test: For a system to be completely state controllable, the necessary and sufficient condition is that the following $n \times nm$ matrix, Q_O has a rank of n.

$$Q_O = [B \ AB \ (A)^2 B \ \ldots\ldots \ (A)^{(n-1)} B]$$

2.3 Full Order Observer

Observer which has the same dimension as that of the original system i.e. It gives information about each and every state used in the system. (whether previously known or unknown)

Mathematical Modelling:

Consider a linear time invariant continuous system

$\dot{x} = Ax(t) + Bu(t) \qquad x(t_o) = x_o$

$y(t) = Cx(t)$

where, $x(t) \epsilon \mathbb{R},\ y(t) \epsilon \mathbb{R},\ \&\ u(t) \epsilon (R)$

A is $n \times n$ constant matrix

B is $n \times 1$ constant matrix

C is $1 \times n$ constant matrix

Since the system output variables, $y(t)$, are available at all times, we may construct another artificial dynamic system of order (built, for example, of capacitors and resistors) having the same matrices $A,\ B,\ \&\ C$

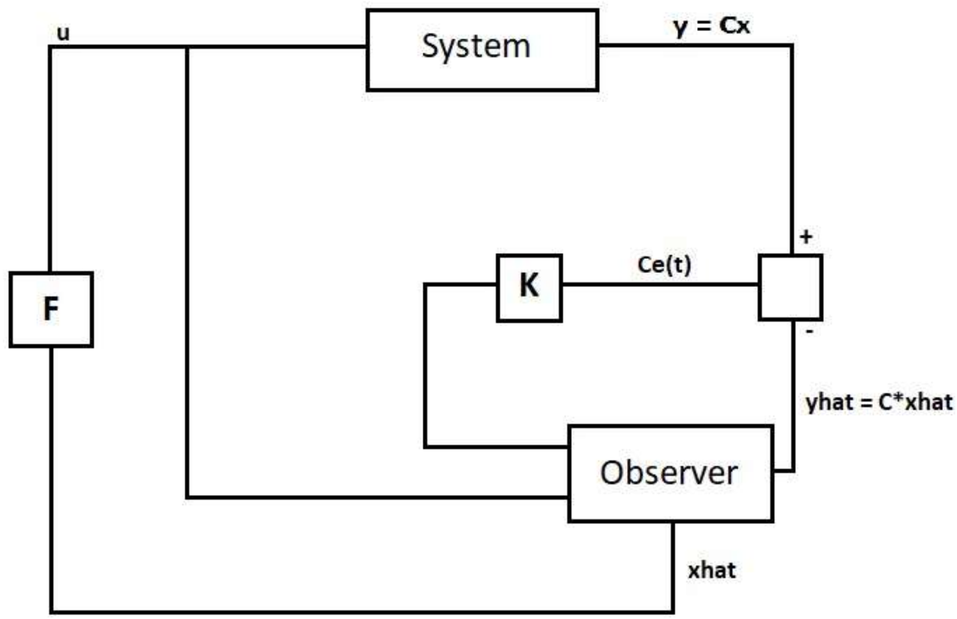
$\dot{\hat{x}} = A\hat{x}(t) + Bu(t) \qquad \hat{x}(t_o) = \hat{x}_o$

$\hat{y}(t) = C\hat{x}(t)$

The two outputs, $y(t)\ and\ \hat{y}(t)$ will be different since in the first case the system initial condition is unknown, and in the second case it has been chosen arbitrarily. The difference between these two outputs will generate an error signal:

$y(t) - \hat{y}(t) \ = \ C[x(t) - \hat{x}(t)] = Ce(t)$

Now this error signal can be used as the feedback signal to the artificial system such that the estimation (observation) error is reduced as much as possible, hopefully to zero (at least at steady state).



Block Diagram of System-Observer Structure

From the above block diagram, we can write observer alone as:

$\dot{\hat{x}} = A\hat{x}(t) + Bu(t) + K(Ce(t))$

As the observer is usually implemented on line as a dynamic system driven by the same input as the original system and the measurements coming from the original systems. So, we can write:

$\dot{\hat{x}} = (A - KC)\hat{x}(t) + Bu(t) + Ky(t)$

Now, It is easy derive the dynamics for the observation error as:

$$\dot{e}(t) = \dot{x}(t) - \dot{\hat{x}}(t) = (A - KC)e(t)$$

If the observer gain K is chosen such that the feedback matrix is $(A - KC)$ asymptotically stable, then the estimation error, $e(t)$, will decay to zero for any initial condition, $e(t_O)$. This can be achieved if the pair (A, C) is observable. More precisely, by taking the transpose of the estimation error feedback matrix, i.e. $A^T - C^T K^T$, we see that if the pair (A^T, C^T) is controllable, then we can locate its poles in arbitrarily asymptotically stable positions. Note that controllability of the pair A^T, C^T is equal to observability of the pair (A, C).

In practice the observer poles should be chosen to be about ten times faster than the system poles. This can be achieved by setting the minimal real part of observer eigenvalues to be ten times bigger than the maximal real part of system eigenvalues, that is

$$|Re[\lambda_{min}]|_{system} \geq x.|Re[\lambda_{max}]|_{observer} \qquad ; 10 \geq x \geq 5$$

The system under perfect state feedback control is given by: $u(t) = -Fx(t)$ and this system will have a closed loop form of:

$$\dot{x}(t) = (A - BF)x(t)$$

So, the eigenvalues of the matrix $(A - BF)$ are the closed-loop system poles under perfect state feedback.

In the case of the system-observer structure, as given in the given block diagram, we see that the actual control applied to both the system and the observer is given by:

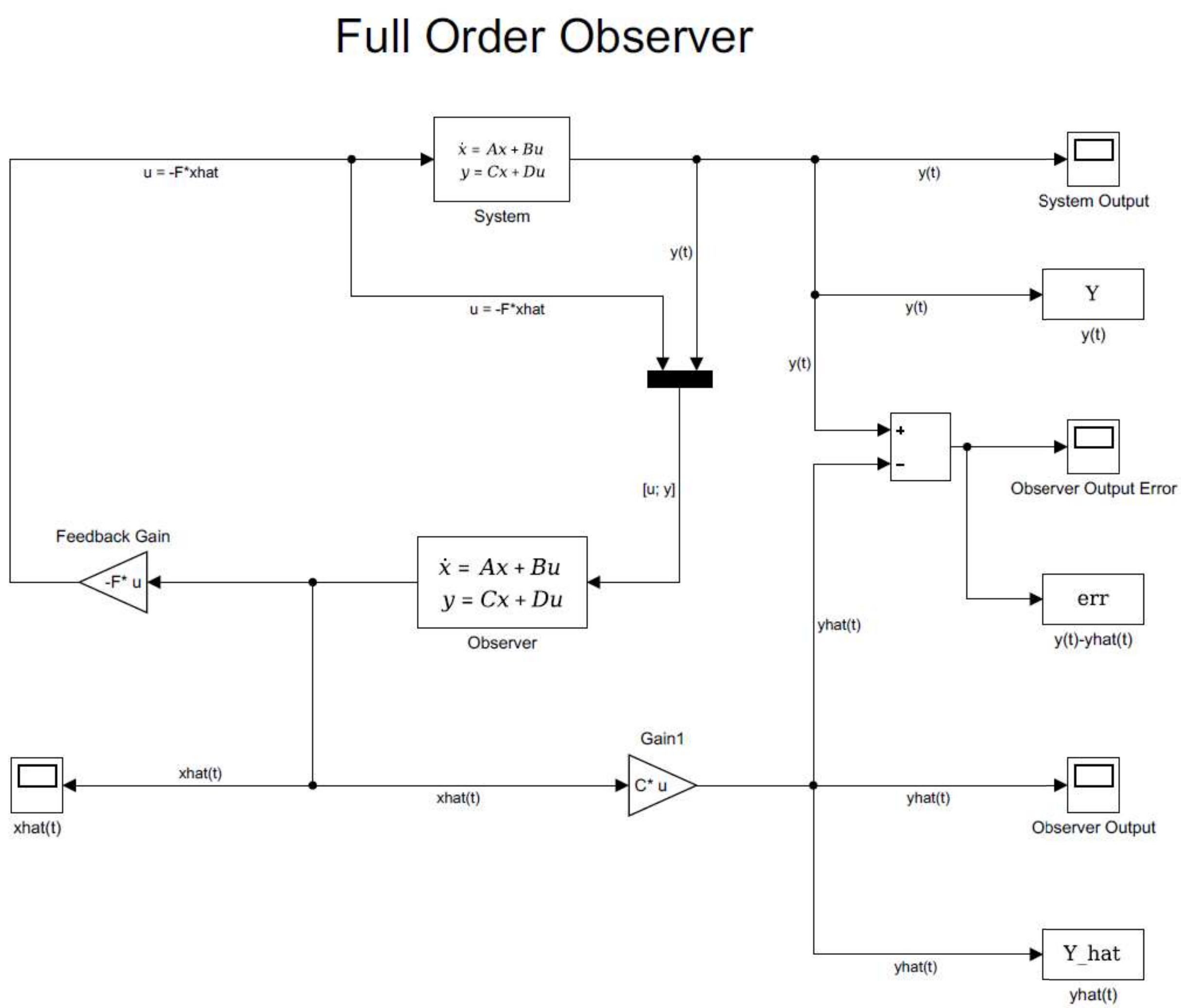
$$u(t) = -F\hat{x}(t) = -Fx(t) + Fe(t)$$

By eliminating, $u(t) = -F\hat{x}(t);\ y(t) = Cx(t);\ \hat{y}(t) = C\hat{x}(t)$, we obtain the closed loop form of the system-observer configuration as:

$$\begin{bmatrix} \dot{x} \\ \dot{\hat{x}} \end{bmatrix} = \begin{bmatrix} A & -BF \\ KC & A - KC - BF \end{bmatrix} \begin{bmatrix} x \\ \hat{x} \end{bmatrix}$$

Separation Principle:

This important observation that the system-observer configuration has closed loop poles separated into the original system closed-loop poles obtained under perfect state feedback, $\lambda(A - BF)$, and the actual observer closed-loop poles, $\lambda(A - KC)$, is known as the separation principle. Hence, we can independently design the system poles using the system feedback gain F and independently design the observer poles using the observer feedback gain K.


Full Order Observer
u = -F*xhat
$\dot{x} = Ax + Bu$
$y = Cx + Du$
System
y(t)
System Output
y(t)
u = -F*xhat
y(t)
Y
y(t)
y(t)
+
–
Observer Output Error
[u; y]
Feedback Gain
-F* u
$\dot{x} = Ax + Bu$
$y = Cx + Du$
Observer
err
y(t)-yhat(t)
yhat(t)
Gain1
xhat(t)
C* u
xhat(t)
yhat(t)
xhat(t)
Observer Output
yhat(t)
Y_hat
yhat(t)

2.4 Reduced Order Observer

Observer which has the dimension less than that of the original system i.e. It gives information about selected states used in the system.

Mathematical Modelling (Without Change of Coordinates):

Consider the linear system with the corresponding measurements:

$\dot{x} = Ax(t) + Bu(t) \qquad x(t_o) = x_o$

$y(t) = Cx(t)$

Assume that the output matrix C has rank p, which means that the output equation represents linearly independent algebraic equations. Thus, the equation $y = Cx(t)$ produces p algebraic equations for n unknowns of $x(t)$. So, we will be constructing an observer of order $n - p$ for estimation of the remaining $n - p$ state space variables.

Let us assume a matrix C_1 such that

$$Rank \begin{bmatrix} C \\ C_1 \end{bmatrix} = n$$

Introducing a vector: $p(t) = C_1 x(t)$

Now, we have: $\begin{bmatrix} y(t) \\ p(t) \end{bmatrix} = \begin{bmatrix} C \\ C_1 \end{bmatrix} x(t)$

$$\implies x(t) = \begin{bmatrix} C \\ C_1 \end{bmatrix}^{-1} \begin{bmatrix} y(t) \\ p(t) \end{bmatrix}$$

Since vector p is unknown, so we will be constructing an observer to estimate it.

Introducing the notation: $\begin{bmatrix} C \\ C_1 \end{bmatrix}^{-1} = \begin{bmatrix} L_1 & L_2 \end{bmatrix}$

$\Longrightarrow$ x(t) = L$_1 y(t) + L_2 p(t)$

Now, an observer for $p(t)$ can be constructed by finding first a differential equation for $p(t)$, that is

$\dot{p} = C_1\dot{x} = C_1Ax(t) + C_1Bu(t) = C_1AL_2p(t) + C_1AL_1y(t) + C_1Bu(t)$

From the above equation, we are not able to construct an observer for $p(t)$ since $y(t)$ does not contain explicit information about the vector $p(t)$

Since, $\begin{bmatrix} C \\ C_1 \end{bmatrix} \begin{bmatrix} C \\ C_1 \end{bmatrix}^{-1} = I$

$$\Longrightarrow \begin{bmatrix} C \\ C_1 \end{bmatrix} \begin{bmatrix} L_1 & L_2 \end{bmatrix} = I$$

$$\Longrightarrow \begin{bmatrix} CL_1 & CL_2 \\ C_1L_1 & C_1L_2 \end{bmatrix} = I$$

$\Longrightarrow CL_1 = I;\ CL_2 = 0;\ C_1L_1 = 0;\ C_1L_2 = I$

The measurements $y(t)$ are given by:

$y(t) = Cx(t) = CL_1y(t) + CL_2p(t) = Iy(t) + 0 = y(t)$

So, from $y(t)$, we can not construct an observer for $p(t)$. But if we differentiate $y(t)$:

$\dot{y}(t) = C\dot{x}(t) = CAx(t) + CBu(t) = CAL_2p(t) + CAL_1y(t) + CBu(t)$

So, we can conclude that $\dot{y}(t)$ carries information about $p(t)$

Now, an observer for $p(t)$ can be constructed using the two equations as:

$\dot{\hat{p}}(t) = C_1AL_2\hat{p}(t) + C_1AL_1y(t) + C_1Bu(t) + K_1(\dot{y}(t) - \dot{\hat{y}}(t))$

where, K_1 is the Observer Gain. And If we replace $p(t)$ by its estimate in the differential equation of $y(t)$, we get:

$\dot{\hat{y}}(t) = CAL_2\hat{p}(t) + CAL_1y(t) + CBu(t)$

$$\implies \dot{\hat{p}}(t) = C_1AL_2p(t) + C_1AL_1y(t) + CBu(t) + K_1[\dot{y}(t) - CAL_2\hat{p}(t) - CAL_1y(t) - CBu(t)]$$

Since it is impractical and undesirable to differentiate $y(t)$ in order to get $\dot{y}(t)$, as this operation produces noice in practical applications. So, we take the change of variables:

$\hat{q} = \hat{p} - K_1y(t)$

So, now we have an observer for $\hat{q}(t)$ of the form:

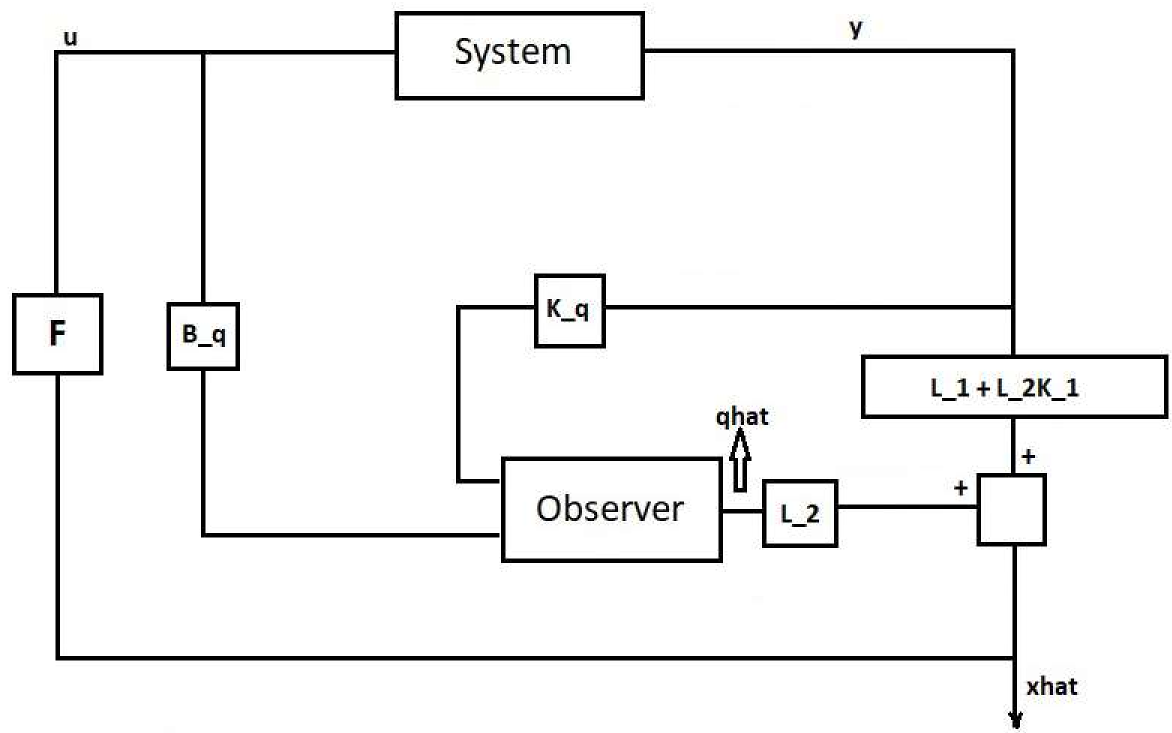
$\dot{\hat{q}}(t) = A_q\hat{q}(t) + B_qu(t) + K_qy(t)$

where, $A_q = C_1AL_2 - K_1CAL_2; \qquad B_q = C_1B - K_1CB$

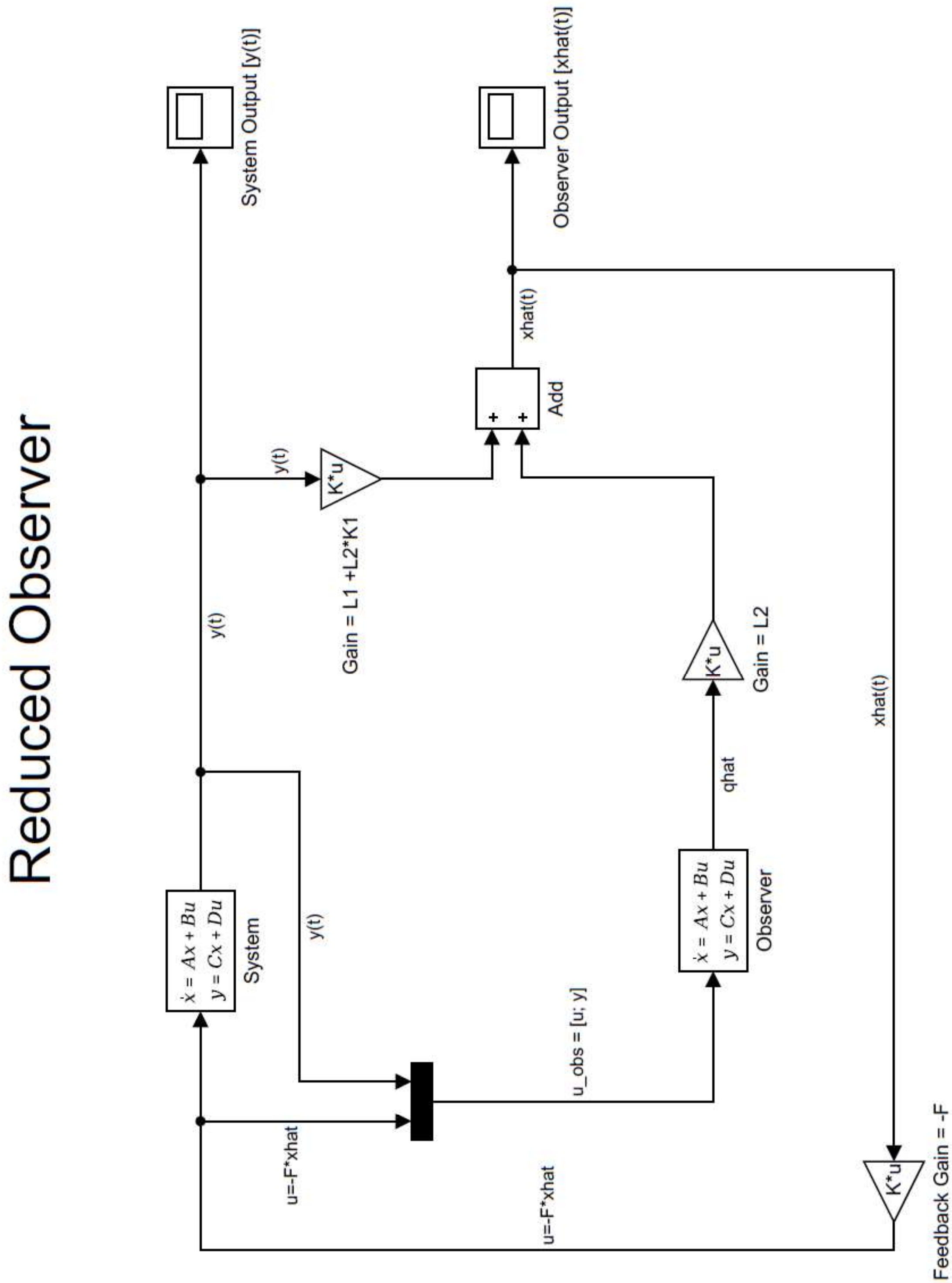
$K_q = C_1AL_2K_1 + C_1AL_1 - K_1CAL_1 - K_1CAL_2K_1$

Now, the estimates of the original system state space variables are now obtained as:

$\hat{x}(t) = L_1\hat{q}(t) + L_2p(t) = L_1\hat{q}(t) + (L_1 + L_2K_1)y(t)$



Block Diagram of System-Observer Structure


Reduced Observer
System Output [y(t)]
Observer Output [xhat(t)]
xhat(t)
Add
y(t)
K*u
Gain = L1 +L2*K1
y(t)
K*u
Gain = L2
xhat(t)
qhat
System
y(t)
Observer
u_obs = [u; y]
u=-F*xhat
u=-F*xhat
K*u
Feedback Gain = -F

2.5 Kalman Filter

Kalman filters are used to estimate states based on linear dynamical systems in state space format. The process model defines the evolution of the state from time $k-1$ to time k as:

$x_k = Fx_{k-1} + Bu_{k-1} + \omega_{k-1}$

where, F is the state transition matrix

B is the control-input matrix applied to the control vector u_{k-1}

ω_{k-1} is the process noise vector that is assumed to be zero-mean Gaussian with the covariance Q , i.e., $\omega_{k-1} \sim N(0, Q)$

The process model is paired with the measurement model that describes the relationship between the state and the measurement at the current time step k as:

$z_k = Hx_k + v_k$

where, z_k is the measurement vector

H is the measurement matrix

v_k is the measurement noise vector that is assumed to be zero-mean Gaussian with the covariance R , i.e., $v_{k-1} \sim N(0, R)$

Kalman Filter Algorithm:

Kalman Filter Algorithm consists of two stages:

1. Prediction
2. Update

Prediction:

$$\textit{Predicted State Estimate} \quad \hat{x}_k^- = F\hat{x}_{k-1}^+ + Bu_{k-1}$$

$$\textit{Predicted Error Covariance} \quad P_k{}^- = FP_{k-1}^+F^T + Q$$

Update:

Measurement Residual	$\widetilde{y}_k = z_k - H\hat{x}_k^-$
Kalman Gain	$K_k = P_k^- H^T (R + P_k{}^- H^T)^{-1}$
Updated State Estimate	$\hat{x}_k^+ = \hat{x}_k^- + K_k \widetilde{y}_k$
Updated Error Covariance	$P_k^+ = (I - K_k H) P_k$

NOTE: In the above equations, the hat operator, $\hat{}$, means an estimate of a variable. That is, $\hat{x}$ is an estimate of x . The superscripts $-$ and $+$ denote predicted (prior) and updated (posterior) estimates, respectively.

The predicted state estimate is evolved from the updated previous updated state estimate.

In the update stage, the measurement residual $\widetilde{y}_k$ is computed first. The measurement residual, also known as innovation, is the difference between the true measurement, z_k, and the estimated measurement, $H\hat{x}_k^-$. The filter estimates the current measurement by multiplying the predicted state by the measurement matrix. The residual, $\widetilde{y}_k$, is later then multiplied by the Kalman Gain, K_k, to provide the correction, $K_k\widetilde{y}_k$, to the predicted estimate $\hat{x}_k^-$. After it obtains the updated state estimate, the Kalman Filter calculates the updated error covariance, P_k^+ , which will be used in the next time step.

Initialisation:

We need an initialization stage to implement the Kalman Filter. As initial values, we need the initial guess of state estimate, $\hat{x}_0^+$, and the initial guess of the error covariance matrix, P_0^+.Q and R , $\hat{x}_0^+$ and P_0^+ plays an important role to obtain desired performance. There is a rule of thumb called "initial ignorance," which means that the user should choose a large P_0^+ for quicker convergence. Finally, one can obtain and implement a Kalman Filter by implementing the prediction and update stages for each time step, $k = 1,\ 2,\ 3,\ .....$, after the initialization of estimates.

2.6 Sliding Mode Control

Sliding Mode Control (SMC) is a technique, suitable for systems with disturbances and uncertainties. This method provides a robust approach for reaching the desired performance. This is a discontinuous method, in which the control input switches between two limits.

Advantage of sliding mode controllers is their insensitivity to parameter variations and disturbances once in the sliding mode, there by eliminating the necessity of exact modeling. The SMC design is composed of two steps:

1. In the first step, a custom-made surface should be designed. While on the sliding surface, the plants dynamics is restricted to the equations of the surface and is robust to match plant uncertainties and external disturbances.

2. In the second step, a feedback control law should be designed to provide convergence of a systems trajectory to the sliding surface; thus, the sliding surface should be obtained in a finite time. The systems motion on the sliding surface is called the sliding mode.

First Step: The first step in SMC is to define the sliding surface, S(t), which represents a desired global behavior, such as stability and tracking performance.

$s(t) = (\frac{d}{dt} + \lambda)^n \int_0^t e(t)dt$

Second Step: Once the sliding surface has been selected, attention must be turned to designing the control law that drives the controlled variable to its reference value and satisfies the equation:

$$U(t) = U_c(t) + U_d(t)$$

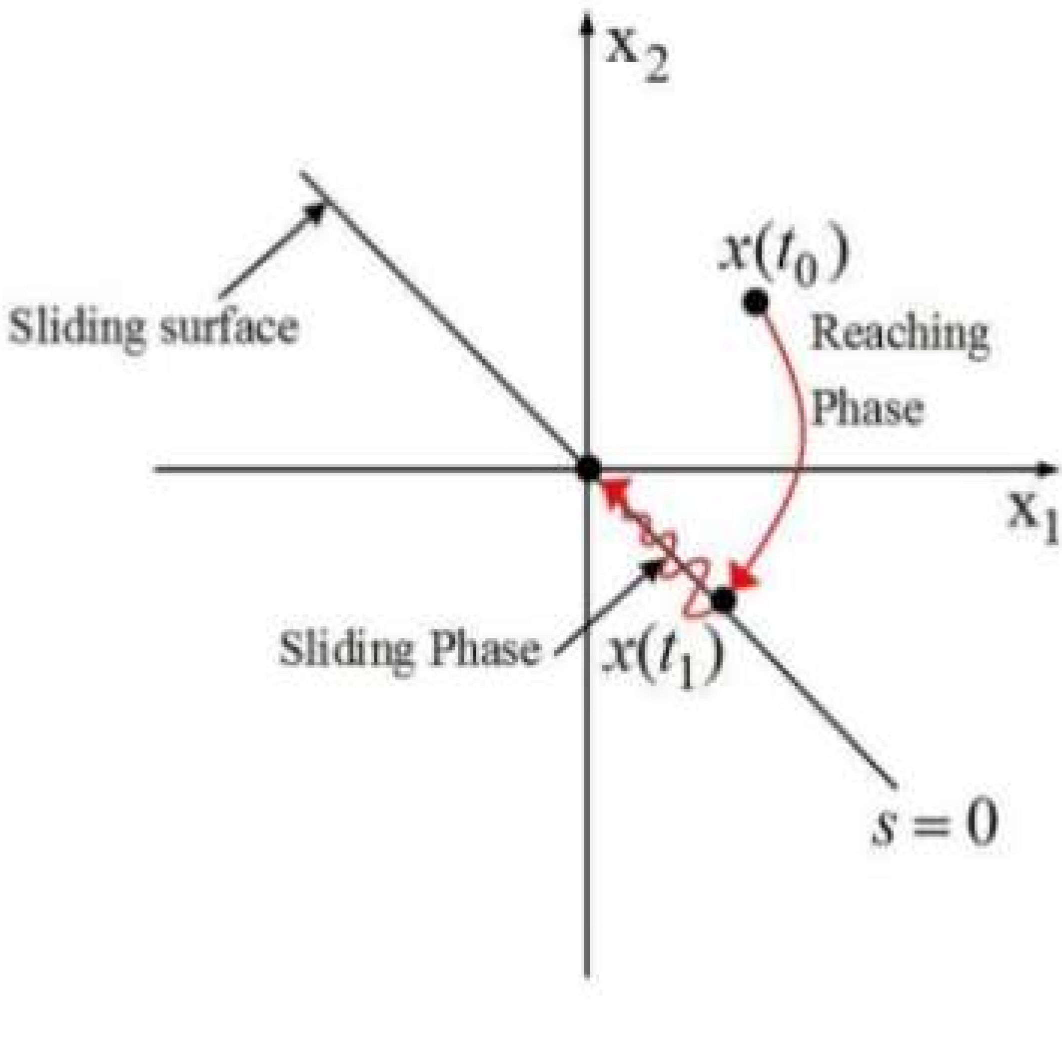
where, $U_d = b^{-1}[a_1x_1 - (c + a_2)x_2 - nsin(\sigma)]$

The continuous part is given by $U_c(t) = f(x(t), r(t))$

The continuous part of the controller is obtained by combining the process model and sliding condition. The discontinuous part is nonlinear and represents the switching element of the control law.

The aggressiveness to reach the sliding surface depends on the control gain but if the controller is too aggressive it can collaborate with the chattering.

$$U_D(t) = K_D \frac{s(t)}{|s(t)| + \delta}$$



Reaching Phase: In this phase a trajectory, starting from a nonzero initial condition and reach to sliding surface.

Sliding Surface: In which the trajectory on reaching the sliding surface, remains there for all further times and thus evolves according to the dynamics specified by the sliding surface.

Chattering (Main drawback of SMC)

1. In theory, the trajectories slide along the switching function.
2. In practice, there is high frequency switching.
3. A high-frequency motion called chattering (the states are repeatedly crossing the surface rather than remaining on it), so that no ideal sliding mode can occur in practice.
4. Yet, solutions have been developed to reduce the chattering and so that the trajectories remain in a small neighborhood (boundary) of the surface. Called as chattering because of the sound made by old mechanical switches.

Chattering can be reduced by:

1. Non-Linear Gains
2. Dynamic Extension
3. Higher Order Sliding Mode Control

2.6.1 Merits and Demerits of SMC

Merits

1. Controller design provides a systematic approach to the problem of maintaining stability consistent performance in the face of modeling imprecision.
2. Possibility of stabilizing some nonlinear systems which are not stabilizable by continuous state feedback laws.
3. Robustness property that is once the system is on sliding surface then it produced bounded parameter variation and bounded disturbances.

Demerits

1. The main obstacle to the success of these techniques in the industrial community is the implementation had an important drawback that is the actuators had to cope with the high frequency control actions that could produce premature wear or even breaking.

Chapter 3

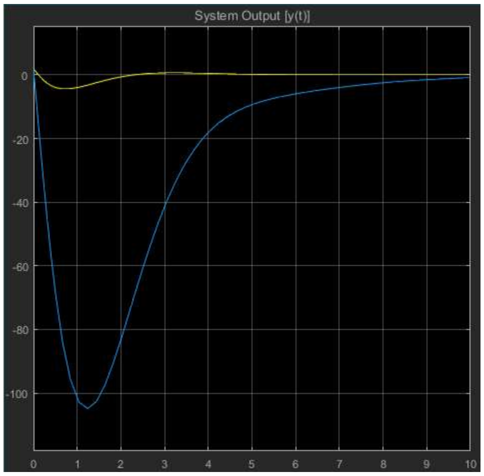
Simulations

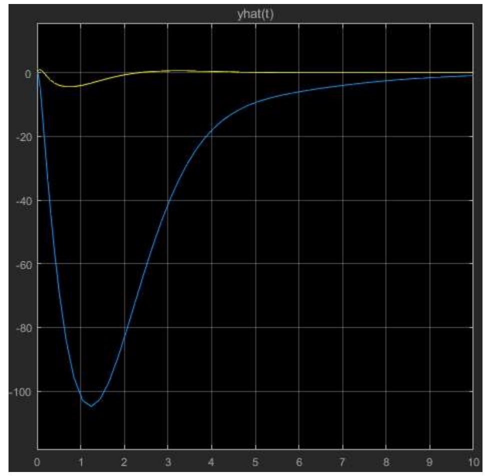
3.1 Simulation of Aircraft Model (Full-Oder)

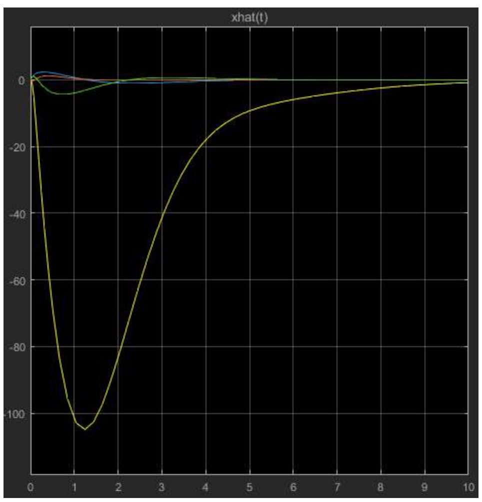
3.1.1 MATLAB Code

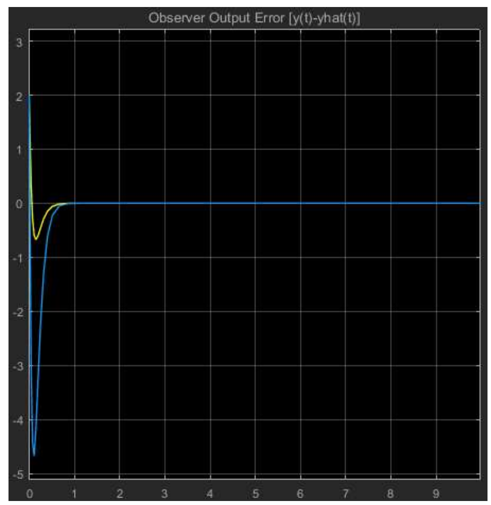
```
% n=system order; m=number inputs; l=number of outputs; c=rank of the output matrix
n=4;
m=1;
l=2;
c=2;

%Defining A Matrix
A=[-0.01357 -32.2 -46.3 0; 0.00012 0 1.214 0; -0.0001212 0 -1.214 1; 0.00057 0 -9.1 -0.6696];

%Defining B Matrix
B=[-0.433;0.1394;-0.1394;-0.1577];

%Defining c Matrix
C=[0 0 0 1; 1 0 0 0];

%Defining D Matrix
D=[0;0];

%Given Initial Conditions
x0=[2;2;2;2];

%Finding the Feedback Gain Matrix using the Desired System Poles
lambdaSystemDesired=[-0.5, -1+1i1, -1-1i1, -2];
F=place(A,B,lambdaSystemDesired);

%Finding the Observer Gain Matrix using the Desired Observer Poles
lambdaObsDesired=[-10 -11 -12 -13];
KT=place(A',C',lambdaObsDesired);
K=KT';
```

3.1.2 Graphs


System Output [y(t)]


yhat(t)


xhat(t)

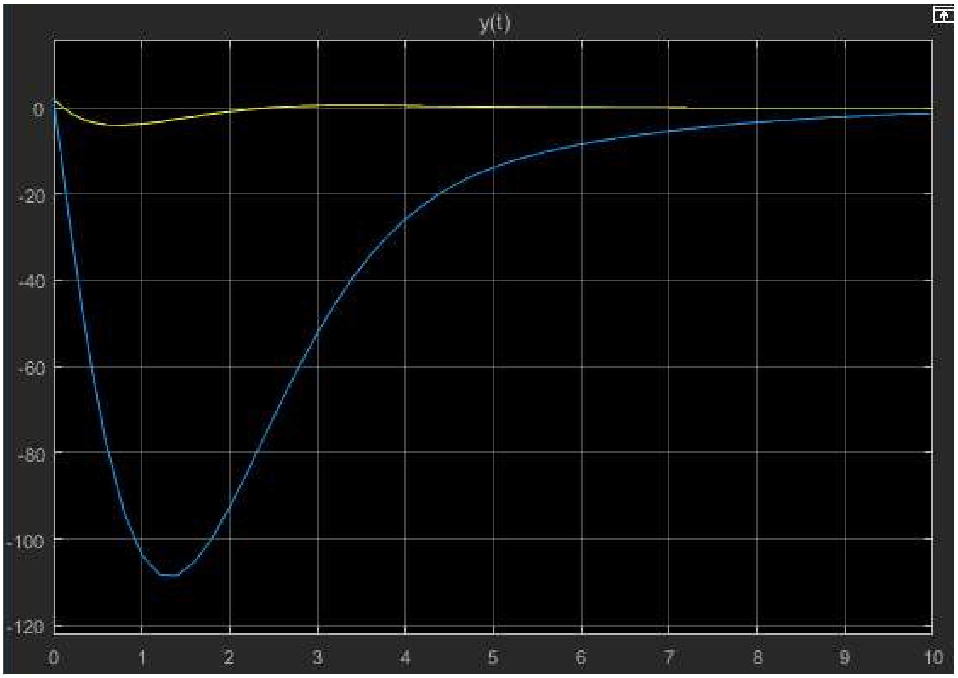

Observer Output Error [y(t)-yhat(t)]

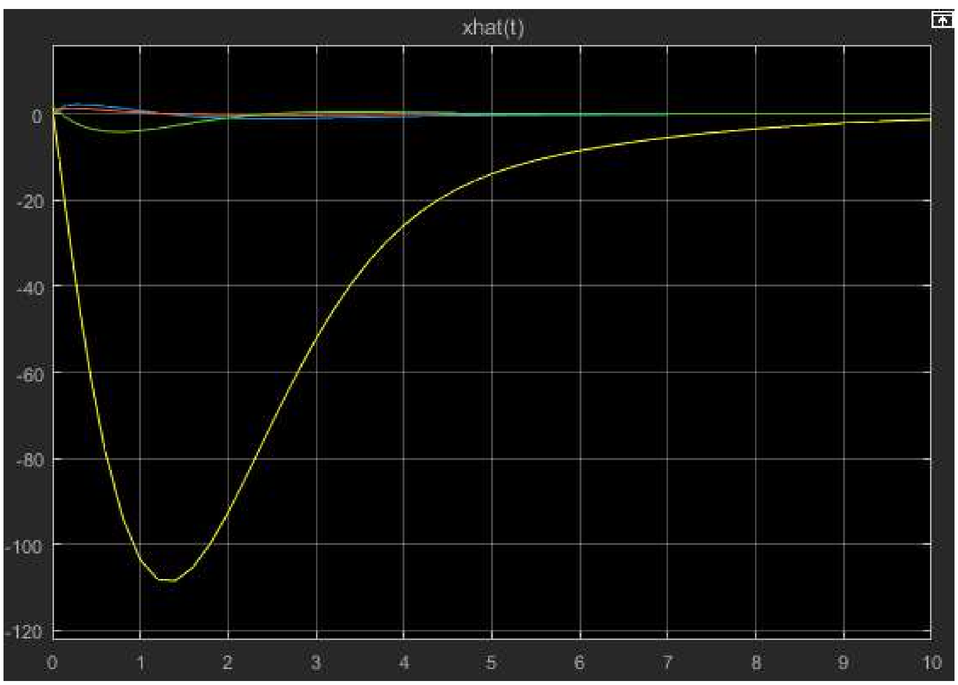
3.2 Simulation of Aircraft Model (Reduced-Observer)

3.2.1 MATLAB Code

```
% n=system order; m=number inputs; % l=number of outputs; c=rank of the output matrix
n=4; m=1; l=2; c=2;

%Defining A Matrix
A=[-0.01357 -32.2 -46.3 0; 0.00012 0 1.214 0; -0.0001212 0 -1.214 1; 0.00057 0 -9.1 -0.6696];

%Defining B Matrix
B=[-0.433;0.1394;-0.1394;-0.1577];

%Defining c Matrix
C=[0 0 0 1; 1 0 0 0];

%Defining D Matrix
D=[0;0];

%Given Initial Conditions
x0=[2;2;2;2];

%Finding the Feedback Gain Matrix using the Desired System Poles
lambdaSystemDesired=[-0.5, -1+1i*1, -1-1i*1, -2];
F=place(A,B,lambdaSystemDesired);

% Proceeding towards the Reduced-order Observer Design
C1=[0 1 0 0;0 0 1 0]; %Assumning the C1 Matrix such that rank([C: C1])=n

Caug=[C;C1];

Laug=inv(Caug); %Lets suppose [C: C1]^- = [L1 L2]

%Finding L1
L1=Laug(1:n,1:c);

%Finding L2
L2=Laug(1:n,c+1:n);

%Finding Observer Gain 'K1'
lambda_red_obs=[-10 -11];
K1T=place((C1*A*L2)',(C*A*L2)',lambda_red_obs);
K1=K1T';

%Finding Values of Aq, Bq and Kq as explained in the modeling of Reduced Observer
Aq=C1*A*L2-K1*C*A*L2;
Bq=C1*B-K1*C*B;
Kq=C1*A*L2*K1+C1*A*L1-K1*C*A*L1-K1*C*A*L2*K1;

% Least-Squares Choices for Initial Conditions
y0=C*x0;
x0hat=pinv(C'*C)*C'*y0;
x0hat_reduced=inv(L2'*L2)*L2'*(pinv(C'*C)*C'-(L1+L2*K1))*y0;
```

3.2.2 Graphs


y(t)
0
-20
-40
-60
-80
-100
-120
0
1
2
3
4
5
6
7
8
9
10

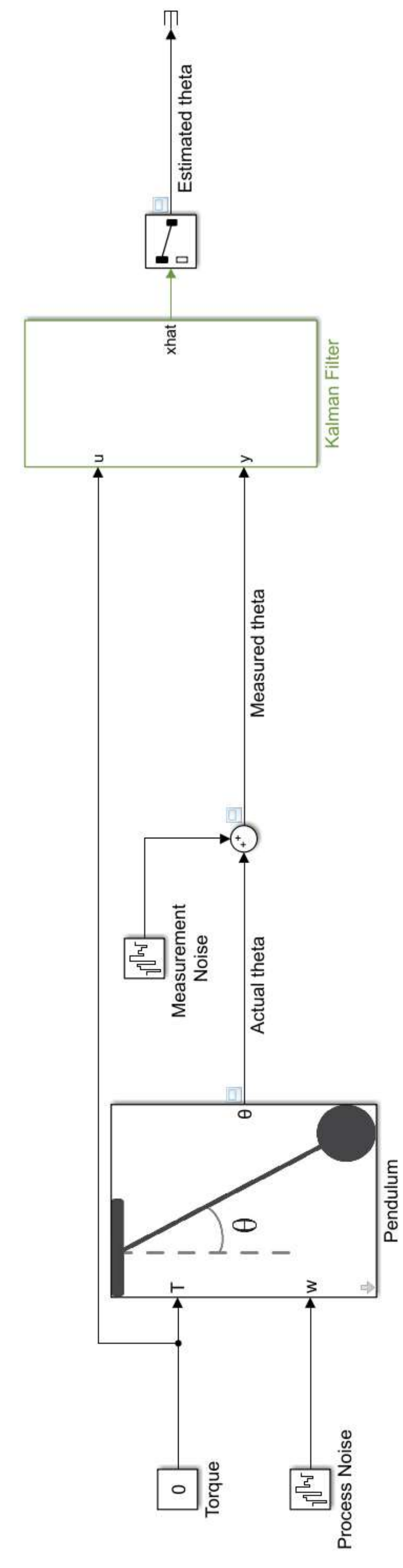

xhat(t)
0
-20
-40
-60
-80
-100
-120
0
1
2
3
4
5
6
7
8
9
10

3.3 Kalman Filter Simulation

3.3.1 MATLAB Code

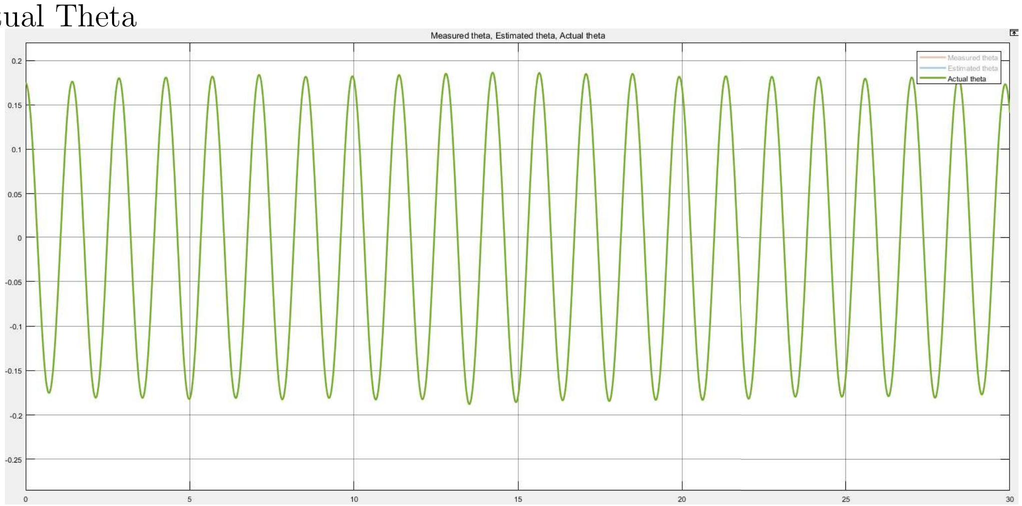
```
% Pendulum model
% Gravity
g = 9.81; % [m/s^2]
% Pendulum mass
m = 1; % [kg]
% Pendulum length
l = 0.5; % [m]

% State space representation
A = [0 1; -g/l 0];
B = [0; 1/(m*l^2)];
C = [1 0];
D = 0;

% Process noise covariance
Q = 1e-3;
% Measurement noise covariance
R = 1e-4;
% Sampling time
Ts = 0.01; % [s]
```

3.3.2 SIMULINK Block

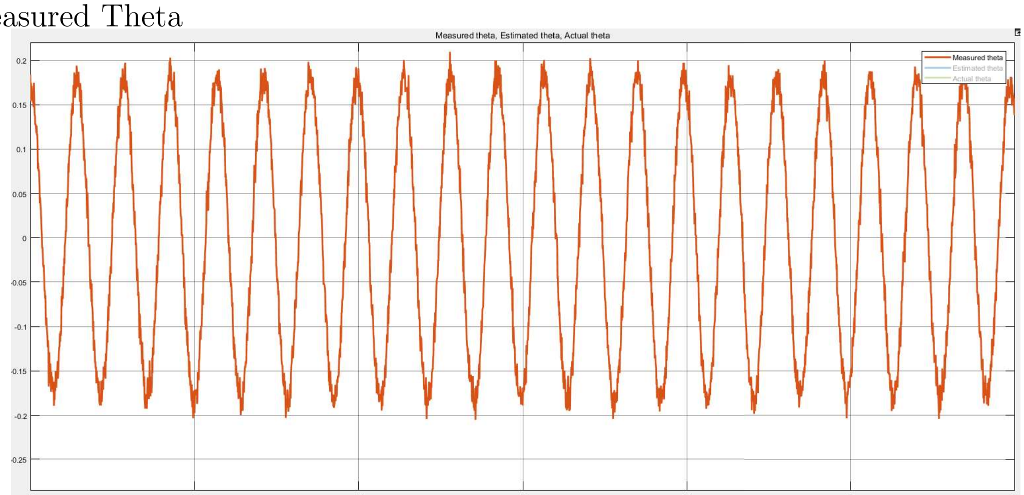


3.3.3 Graphs

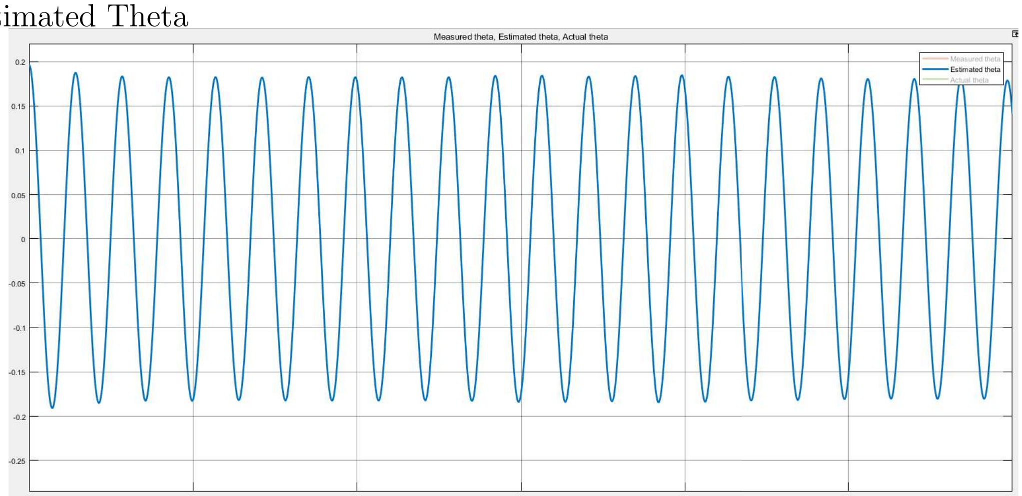
Actual Theta



Measured Theta



Estimated Theta



Bibliography

1. IEEE Xplore "Full and Reduced Order Linear Observer Implementations in Matlab/Simulink" by 'Verica Radisavljevic-Gajic'
2. "Sliding Mode Observers for Fault Detection and Isolation" by 'Christopher Edwards', 'Sarah K. Spurgeon', and 'Ron J. Patton'
3. "Sliding Mode Observers" by 'J. P. BARBOT', 'M. DJEMAF', and 'T. BOUKHOBZA'
4. "Introduction to Linear and Nonlinear Observers" by 'Zoran Gajic'