State Estimation for Various Systems with Partially Unknown Dynamics

Abdulelah Rajeh Alshareef
Marquette University

State Estimation for Various Systems with Partially Unknown Dynamics

By

Abdulelah Alshareef

**A Thesis submitted to the Faculty of the Graduate School,
Marquette University,
in Partial Fulfillment of the Requirements for
the Degree of Masters of Science (Electrical and Computer Engineering)**

**Milwaukee, Wisconsin
May 2019**

ABSTRACT

STATE ESTIMATION FOR VARIOUS SYSTEMS WITH PARTIALLY UNKNOWN DYNAMICS

ABDULELAH ALSHAREEF

MARQUETTE UNIVERSITY, 2018

This thesis is on state estimation for various system and measurement models with uncertain dynamics. The uncertain dynamics may be due to imprecise system modeling or change in parameters due to varying environmental conditions. Uncertainty may be a result of malicious acts such as hacking of sensors or actuators in the system. Uncertainty may also be a result of external disturbances whose waveforms, magnitudes and arrival times may not be known. These types of model uncertainties will be considered and different estimators will be implemented to deal with such uncertainties in state estimation.

In this thesis, for linear stochastic systems with additive noise, the measurements and input are available and noises statistics are known, Kalman filter is used to estimate the state. However, for nonlinear systems, Extended Kalman filter is used under the same conditions. When noise statistics are unknown, H-infinity filter is used to estimate the state of the system if the noises are assumed to be of finite energy.

For identification of parameters, coefficients in transfer functions are identified by using Kalman and H-infinity filters. By using Extended Kalman and H-infinity filters, unknown parameters of the state-space model can be estimated. For parameters whose range of values is available, a bank of Kalman filters is used to find the actual values.

For detection of an intrusion signal that attacks a sensor or actuator of a system, there are several methods considered in this thesis, including the sample mean method, Kalman filter method and stochastic parameter estimation method.

The simulation results of the various applications of these filters will be presented and the performance of these filters will be discussed.

ACKNOWLEDGEMENTS

Abdulelah Alshareef

I would like to thank my parents, Rajeh Alshareef and Salha Alshareef, and my wife, Ashwag Alsharif for all of their support. Furthermore, I would like to thank my research advisors, Dr. Edwin Yaz and Dr. Susan Schneider, for their help, guidance, and support throughout this work. In addition, I would also like to thank my committee member, Dr. Chung Seop Jeong for reading the thesis and giving constructive suggestions to improve my thesis.

TABLE OF CONTENTS

LIST OF FIGURES

1 Introduction

1.1 Introduction

In industry, there are many complex systems that are exposed to several adverse factors. Such factors can make the dynamics of control systems become uncertain. The external and internal disturbances that effect both the system's state and output may be a reason for these uncertainties. An intrusion signal that attacks sensors or actuators of a system, may be another reason for these uncertainties as well. Moreover, when parameters of a system are varying in time due to such factors as change in the ambient temperature, uncertainty may also be experienced. These uncertainties may affect the operation of the system adversely. Therefore, there is a need to investigate techniques that could be implemented in order to detect these uncertainties and compensate for them.

A possible approach is by using estimation theory and various estimators can be used to deal with such uncertainties in state estimation. Kalman, Extended Kalman, H-infinity filters have been used to estimate the states of a system. In addition, there are many techniques to identify parameters using the same type of estimation methods, as well as a bank of Kalman filter to find values of parameters when their values are known to be in a certain range. The same bank of Kalman filters can also be used to detect attacks that may occur through a sensor or actuator.

1.2 Literature Study

To estimate the states of a system, there are several approaches. When the statistics of the noise acting on the system and measurements are known (e.g. Gaussian distribution, zero mean with a certain covariance, and white), Kalman filter can be used to estimate the state of a linear system with minimum estimation error covariance [1, 2]. In addition, when the noise statistics are also known and the system is nonlinear, the extended Kalman filter can be used to estimate the state by linearizing the nonlinear system around the current estimate [1, 2]. On the other hand, when the noise statistics are uncertain but if the noise acting on the system has finite energy, the state of the linear system is estimated by an H-infinity filter by minimizing the maximum estimation error [1, 8, 9, 10].

To identify parameters of a system, recursive least squares algorithm can be used to identify coefficients of a transfer function [11]. In addition, Extended Kalman filter could also be used to identify parameters of a linear state-space model by assuming that unknown parameters as states [12]. On the other hand, when the values of the parameters of the system are uncertain but constant or slowly time-varying with a knowledge of their range of values, a bank of Kalman filters can be used to estimate the uncertain parameters [5, 13]. Moreover, a bank of Kalman filters can be used to detect intrusion signals during sensor or actuator attacks [14, 15, 16].

1.3 Contributions of the Thesis

This thesis proposes to apply a range of methods from estimation theory to estimate the states, identify parameters, and detect intrusion signals for various systems with uncertain dynamics and/or disturbances. In this thesis, systems are considered to be either linear or nonlinear but time- invariant and discrete-time. Disturbances affect the system's state and output additively and can have certain or uncertain characteristics. When disturbances have known characteristics (zero mean, known covariance, white, etc), Kalman and Extended Kalman filters are used to estimate the states of linear and nonlinear systems, respectively. Moreover, they are also used to identify parameters of linear system. On the other hand, when disturbances are uncertain (but have finite energy), H-infinity filter is used to estimate the states of linear and nonlinear systems. In addition, H-infinity techniques are also used to identify parameters of linear systems. When disturbances have known statistics, a bank of Kalman filters is used to detect a certain type of intrusion that attacks the system through sensors and/or actuators of the system.

1.4 Thesis organization

Chapter 1 gives a brief introduction to the work performed and the notation used in this thesis. In chapter 2, we will discuss linear and nonlinear filters. Linear filters are Kalman filter (when noise statistics are known) and H-infinity filter (when noise statistics are unknown) that are used to estimate a state of linear systems. In addition, we introduce a bank of Kalman filters that has many applications such as to estimate the value of a parameter whose range is known. In chapter 3, we will show how to use Kalman, Extended Kalman, and H-infinity filters to estimate parameters of a system. In addition, we will also

discuss how to find the actual value of a parameter when its value is in a given range of values. In chapter 4, we will discuss how to detect an intrusion signal when attack happens to a system through a sensor or actuator by sample mean method, bank of Kalman filters method, and stochastic parameter estimation method. Finally, in chapter 5, we will draw conclusions from the study, and we will discuss some future work.

1.5 Notation

A: Constant system matrix
B: Input matrix
F: Process noise coefficient matrix
C: Output matrix
D: Direct transmission matrix
G: Measurement noise matrix
A_k: Time-variant system matrix
C_k: Time-variant output matrix
F_k: Time-variant process noise matrix
G_k: Time-variant measurement noise matrix
C_z: Performance matrix coefficient
$\mathbb{C}_k$: A random measurement matrix
$\bar{\mathbb{C}}_k$: Expected value of the random measurement matrix
$\tilde{\mathbb{C}}_k$: Zero mean value of the random measurement matrix
u_k: Input of a system
x_k: State of a system
y_k: Measurement of a system
$\hat{x}_k$: Estimated state of a system
$\hat{y}_k$: Estimated measurement of a system
e_k: Estimation error
$\tilde{y}_k$: The difference between an actual and estimated measurement

z_k: Performance output

$x \sim N(\bar{x}, \mathrm{X})$: x is distributed normally with $\bar{x}$ mean and X covariance

v_k: Process noise

w_k: Measurement noise

V : Process noise covariance

W: Measurement noise covariance

K_k: Kalman filter gain

K_k^{∞}: H-infinity filter gain

KF : Kalman Filter

EKF : Extended Kalman filter

P_k: Estimation error covariance

γ: Bound on the ratio between energy of the performance output to noise energy

p: Probability of event

Ω: Covariance of $\tilde{y}_k$

h_k: Intrusion signal

α : Parameter

β : Probability for the absence of attack

2 Optimal Filtering

An optimal filter gives us the best solution according to a performance criterion to estimate the states of a system. In this chapter, we will show how an optimal (or near-optimal) filter is used to estimate states in several situations when the system is either linear or nonlinear; also, when statistics of the system and measurement noises are either certain or uncertain.

2.1 Linear Filters

In this section, we will estimate the states of the system with known noise statistics using a Kalman filter. In addition, we will use an H-infinity filter to estimate the states of a system when statistics of noises are uncertain. Finally, we will explain how a bank of Kalman filters works in order to estimate the parameters of the system.

2.1.1 Kalman Filter

First, we will look at the derivation of Kalman Filter (K.F.) that can be used to estimate the states of linear discrete-time system with additive white noise having known statistics. Kalman filter is a set of equations in one-step-ahead a predictor corrector form that estimates the states with minimum error covariance [2].

In this section, first, the derivation of Kalman filter equation is discussed. After that, we will use the Kalman filter equations to estimate states of two systems (first and second order system). Then, we will discuss the effects of covariance of process and measurement

noises the Kalman filter estimation error when the noises of process and measurement were increased or decreased. Finally, the properties and limitations of the Kalman filter are discussed.

Consider a linear discrete time-invariant system as represented in equations (2.1) and (2.2).

$$x_{k+1} = A\,x_k + B\,u_k + F v_k \tag{2.1}$$

$$y_k = C x_k + D\,u_k + G\,w_k \tag{2.2}$$

where $A, B, C, D, F, and\ G$ are matrices. By assuming that the input u_k is known, the output measurement y_k is available, and the states of the system x_k are unmeasurable, we will attempt to estimate x_k by minimizing the estimation error covariance matrix. In addition, v_k presents process noise, its covariance is *V*, w_k represents measurement noise, its covariance is *W*, and the cross-covariance of process and measurement noises is *S*. Both noises are of zero mean and white.

To determine the expected value of the state, the observer $\hat{x}_k$ can be represented using the following equation [3, 7].

$$\hat{x}_{k+1} = A\,\hat{x}_k + B\,u_k + K_k\,(y_k - (C\hat{x}_k + D\,u_k))$$

(2.3)

$$E\{x_k\} = E\{\hat{x}_k\}$$

(2.4)

All coefficients of the estimated state equation (i.e. equation (2.3)) are known except the gain, K_k. The way to find the gain is by computing estimation error covariance (P_k) first and then by minimizing the estimation error covariance. The estimation error is given by equation (2.5).

$$e_k = x_k - \hat{x}_k$$

and it evolves in time as

$$e_{k+1} = x_{k+1} - \hat{x}_{k+1} = A\,x_k + A\,u_k + F_k v_k - A\,\hat{x}_k - B\,u_k - K_k\,(Cx_k + D\,u_k + G\,w_k - C\hat{x}_k - D\,u_k)$$

$$e_{k+1} = (A - K_k C)e_k + Fv_k - K_k G w_k$$

(2.5)

The covariance of the estimation error is

$$P_{k+1} = E\,\{(e_{k+1})(e_{k+1})^T\}$$

and it satisfies the following equation

$$P_{k+1} = AP_kA^T - K_kCP_kA^T - AP_kC^T{K_k}^T + K_kCP_kC^T{K_k}^T + FVF^T - K_kGS^TF^T - FSG^T{K_k}^T + K_kGWG^T{K_k}^T$$

or

$$P_{k+1} = (A - K_k C)P_k(A - K_k C)^T + [F \quad -K_k G]\begin{bmatrix} V & S \\ S & W \end{bmatrix}\begin{bmatrix} F^T \\ G^T K_k{}^T \end{bmatrix}$$

(2.7)

One very important property of the error covariance matrix, P_k, is that it is positive semi-definite and symmetric. By using these properties, error covariance matrix can be minimized. Matrix minimization will be done by minimizing the trace of this matrix. First, we will find the trace of the error covariance matrix, then we minimize the trace by taking the derivative of the trace of the matrix with respect to gain and set it equal to zero as shown below. Finally, gain of Kalman filter, K_k can be found by solving this in terms of gain [3, 7].

$$\frac{\partial T(P_{k+1})}{\partial K_k} = 0$$

$$\frac{\partial P_{k+1}}{\partial K_k} = -2A_k P_k C_k{}^T - 2F_k S_k G_k + 2K_k\left(C_k P_k C_k{}^T + G_k W_k G_k{}^T\right) = 0$$

(2.8)

From equation (2.8) the gain of Kalman filter K_k can be simplified to

$$K_k = (AP_k C^T + FSG)(CP_k C^T + GWG^T)^{-1}$$

(2.9)

Table 2.1 shows the summary of Kalman filter equations. When process and measurement noises are uncorrelated, the covariance of process and measurement (S) is zero. Equations (2.10) and (2.11) are presented for the estimation error covariance P_k and the gain of the Kalman filter K_k when process and measurement noises are uncorrelated.

Table 2.1: Summary of Kalman Filter Equations in General Case

State Estimate	$\hat{x}_{k+1} = A\,\hat{x}_k + B\,u_k + K_k\,(y_k - (C\hat{x}_k + D\,u_k))$
Gain of Kalman Filter	$K_k = (AP_kC^T + FSG)(CP_kC^T + GWG^T)^{-1}$
Error Covariance	$P_{k+1} = (A - K_kC)P_k(A - K_kC)^T + [F \quad -K_kG]\begin{bmatrix} V & S \\ S & W \end{bmatrix}\begin{bmatrix} F^T \\ G^T K_k{}^T \end{bmatrix}$

$$K_k = (A_kP_kC_k{}^T)\left(C_kP_kC_k{}^T + G_kW_kG_k{}^T\right)^{-1}$$

(2.10)

$$P_{k+1} = A_kP_kA_k{}^T - A_kP_kC_k{}^T\left(C_kP_kC_k{}^T + G_kW_kG_k{}^T\right)^{-1}C_kP_kA_k{}^T + +F_kV_kF_k{}^T$$

(2.11)

Now, we will use the Kalman filter equations to estimate states for first and second order systems. Consider a first order system that is presented by the following equations:

$$x_{k+1} = A\,x_k + Bu_k + Fv_k$$

(2.12)

$$y_k = Cx_k + Gw_k \tag{2.13}$$

where $A = 0.9, B = 0.09, and\ C = F = G = 1$. u_k is a unit step function, $x_{k=0} = 10$. $v_k \sim N(0,0.01)$ $w_k \sim N(0,0.01)$.

The initial values were set as

$$\hat{x}_{k=0} = 12, \qquad P_{k=0} = 100$$

By assuming that x_k is unmeasurable and y_k is available, Kalman filter can be used to estimate x_k. By applying equations (2.3), (2.10), and (2.11), the result as shown in Figure 2.1 can be obtained. Figure 2.1 shows the actual and estimated value of the state.

Because the actual value of the state is unknown in real life, we define $\hat{y}_k$ that is called estimated measurement, by equation (2.14)

$$\hat{y}_k = C\,\hat{x}_k \tag{2.14}$$

Figure 2.2 shows the actual and estimated value of measurements. The difference between the actual and estimated value of measurements is called the output estimation error that can be computed using equation (2.15) below and Figure 2.3 shows the output estimation error. In addition, the percentage of output estimation error is calculated using equation

(2.16) below that is the ratio between the square of the norm of the output estimation error vector to the squared norm of the measurement vector.

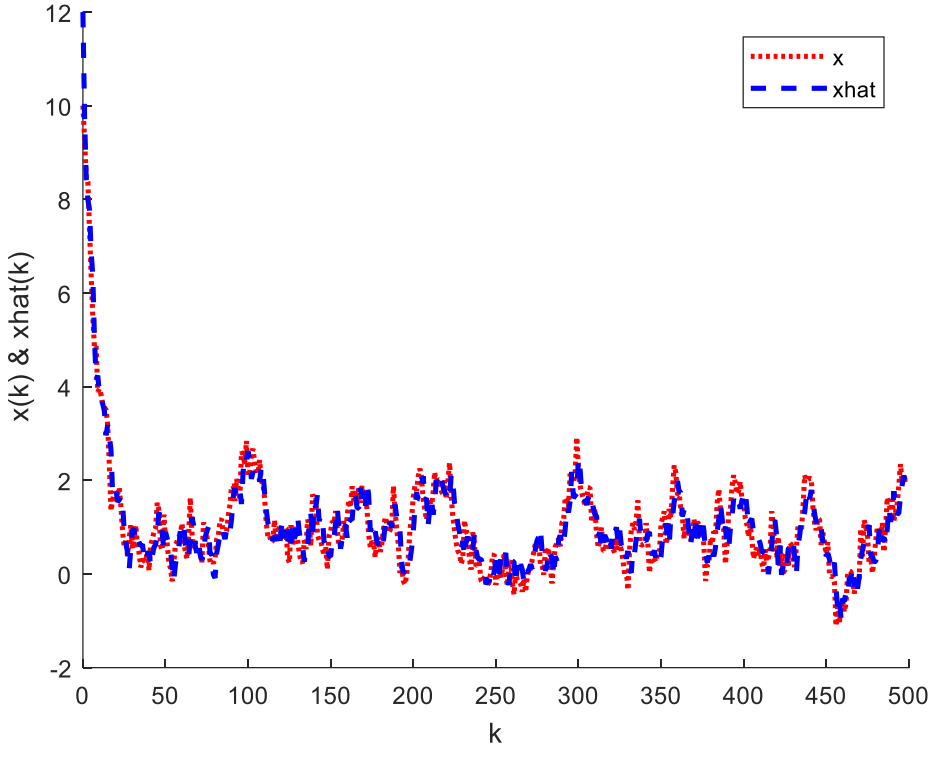

Figure 2.1: The estimation of state for first order system by Kalman Filter

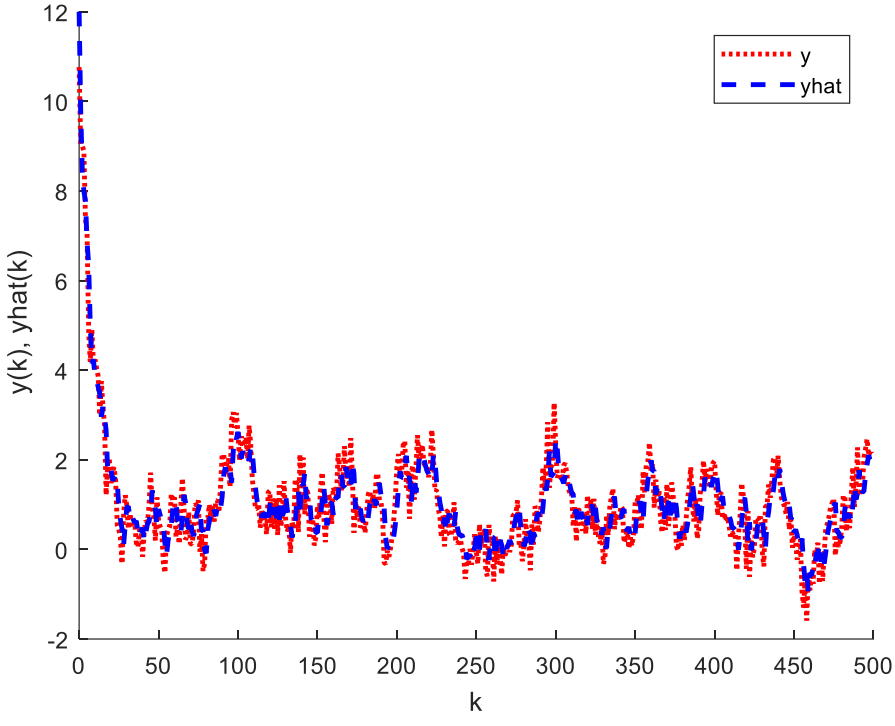

Figure 2.2: The actual and estimated measurement of the system by Kalman Filter to estimate the state for first order system

$$error_y = y_k - \hat{y}_k \tag{2.15}$$

$$Error_{percentage} = \frac{error_y^T error_y}{{y_k}^T y_k} \times 100\% \tag{2.16}$$

By repeating the simulation 25 times, a sample mean for all results is calculated via equation (2.17) below. The percentage of output estimation error for this system after repeated 25 times is 1.58 %.

$$Sample\ Meas_{error\ percentage} = \frac{1}{N}\sum_{i=1}^{N} Error_{percentage_i}$$

(2.17)

where N is the number of simulation runs.

Now, we will use Kalman filter equations to estimate states for a second order system that are presented by equations (2.18) and (2.19) below:

$$\begin{bmatrix} {x_{k+1}}^1 \\ {x_{k+1}}^2 \end{bmatrix} = \begin{bmatrix} 1 & 0.1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} {x_k}^1 \\ {x_k}^2 \end{bmatrix} + \begin{bmatrix} 0.1 \\ 1 \end{bmatrix} u_k + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v1_k \\ v2_k \end{bmatrix}$$

(2.18)

$$y_k = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} {x_k}^1 \\ {x_k}^2 \end{bmatrix} + w_k$$

(2.19)

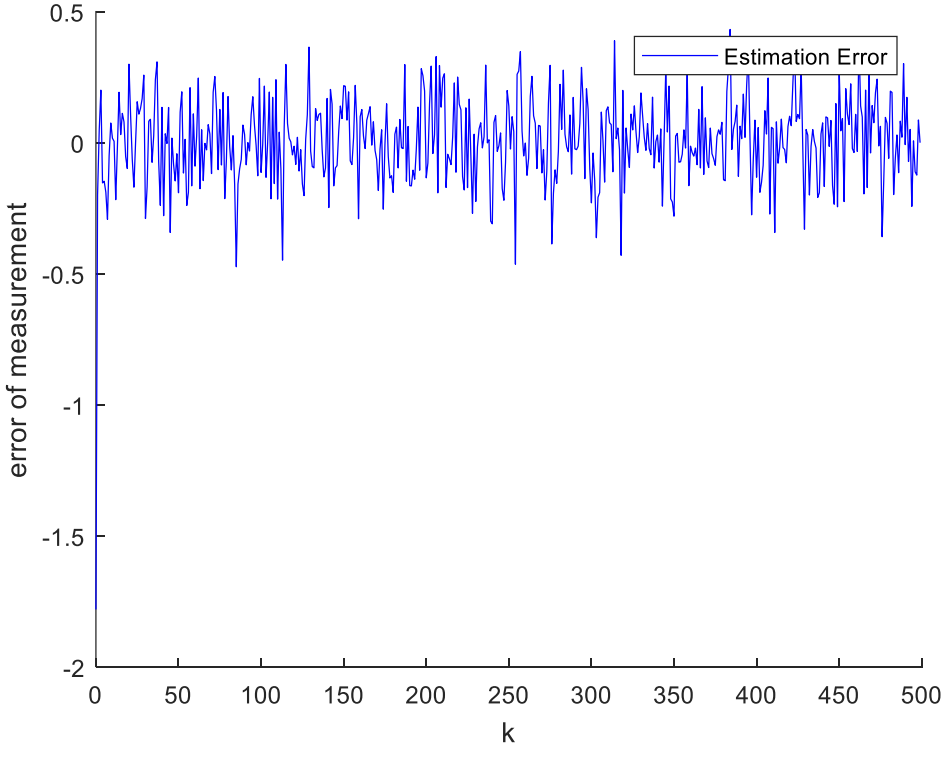

Figure 2.3: The output estimation error for first order system by Kalman Filter

where $A = \begin{bmatrix} 1 & 0.1 \\ -1 & 0 \end{bmatrix}, B = \begin{bmatrix} 0.1 \\ 1 \end{bmatrix}, C = [1 \quad 0],\ F = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, G = 1,\ v_k = \begin{bmatrix} v1_k \\ v2_k \end{bmatrix}$,

and the process and measurement noise are white and mutually, with statistics

$$v1_k \sim N\ (0,0.01)\ ,\ v2_k \sim N\ (0,0.01)\ ,\ \text{and}\ \ w_k \sim N\ (0,0.01)$$

$$V = \begin{bmatrix} V1 & 0 \\ 0 & V2 \end{bmatrix}, \qquad W = 0.01$$

For simulation purposes, the initial values were set as

$$\begin{bmatrix} \hat{x}_0{}^1 \\ \hat{x}_0{}^2 \end{bmatrix} = \begin{bmatrix} 12 \\ 1.2 \end{bmatrix}, \qquad P_0 = 100 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

By assuming, the states are unknown and measurements are available. We apply equations (2.3), (2.10), and (2.11) to estimate states of the systems.

Figure 2.4 shows us the actual and estimated values of x_k^1 whereas, Figure 2.5 shows the actual and estimated values of x_k^2. The actual and estimated measurements are shown in Figure 2.6. By using equations (2.16) and (2.17), the percentage output estimation error is 1.4 %.

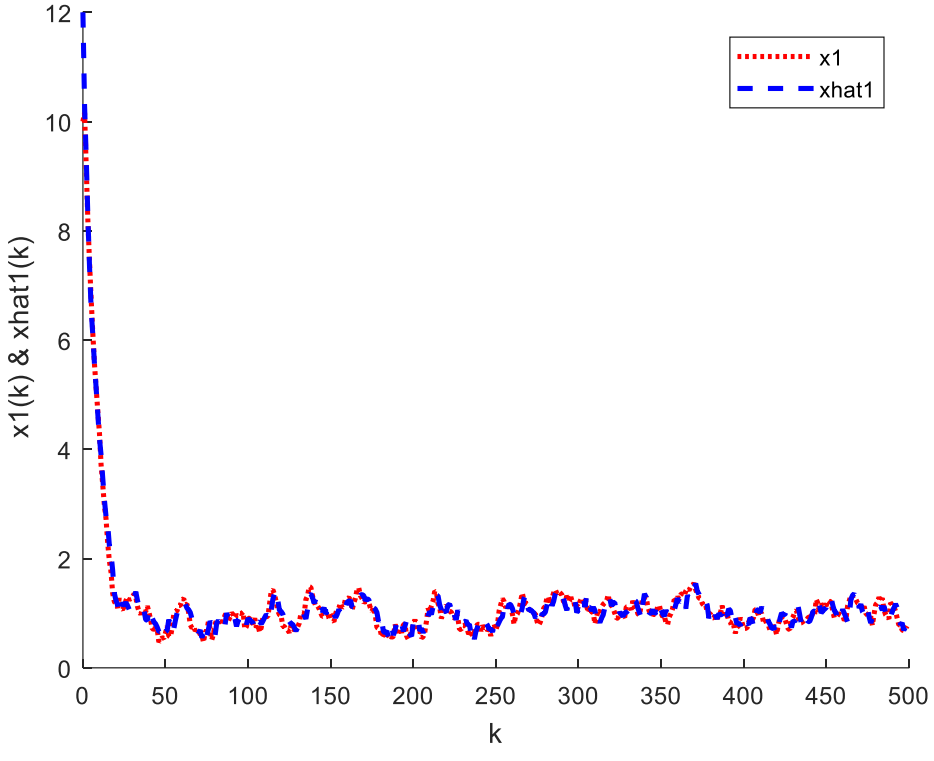

Figure 2.4 The actual and estimated values of x1 by Kalman Filter

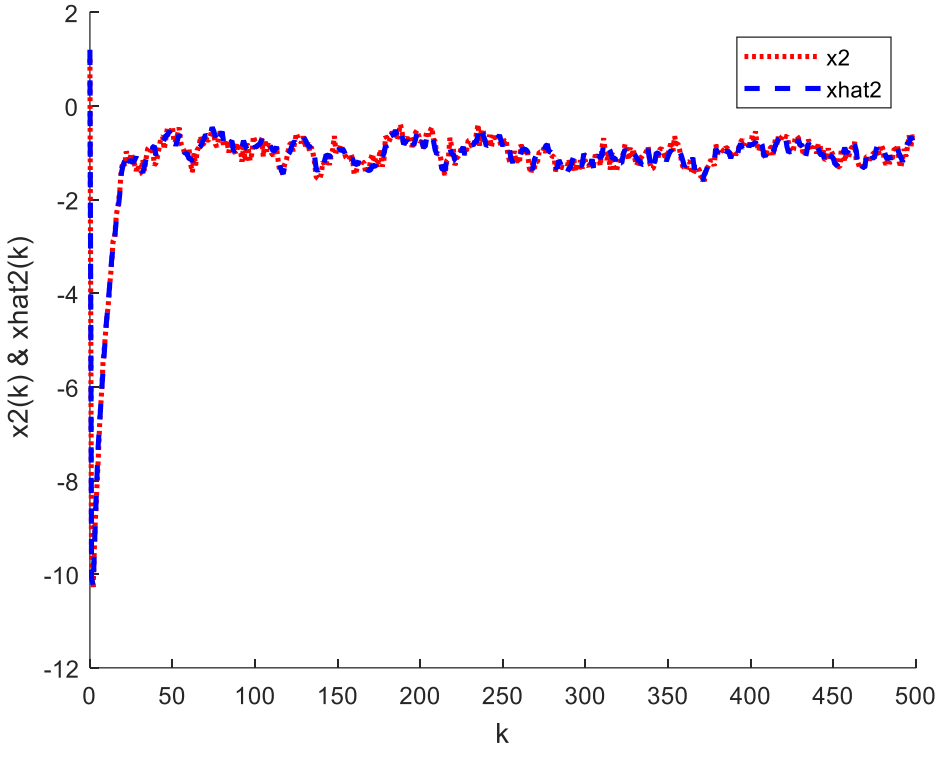

Figure 2.5 The actual and estimated values of x2 by Kalman Filter

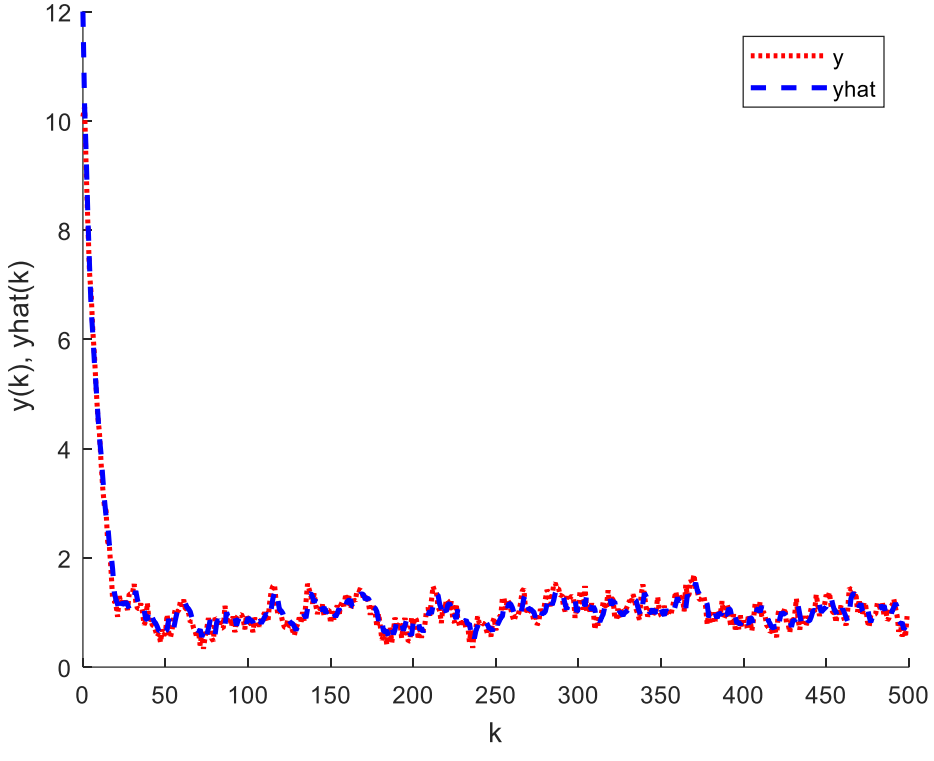

Figure 2.6 The actual and estimated measurement of the system by Kalman Filter for the second order system

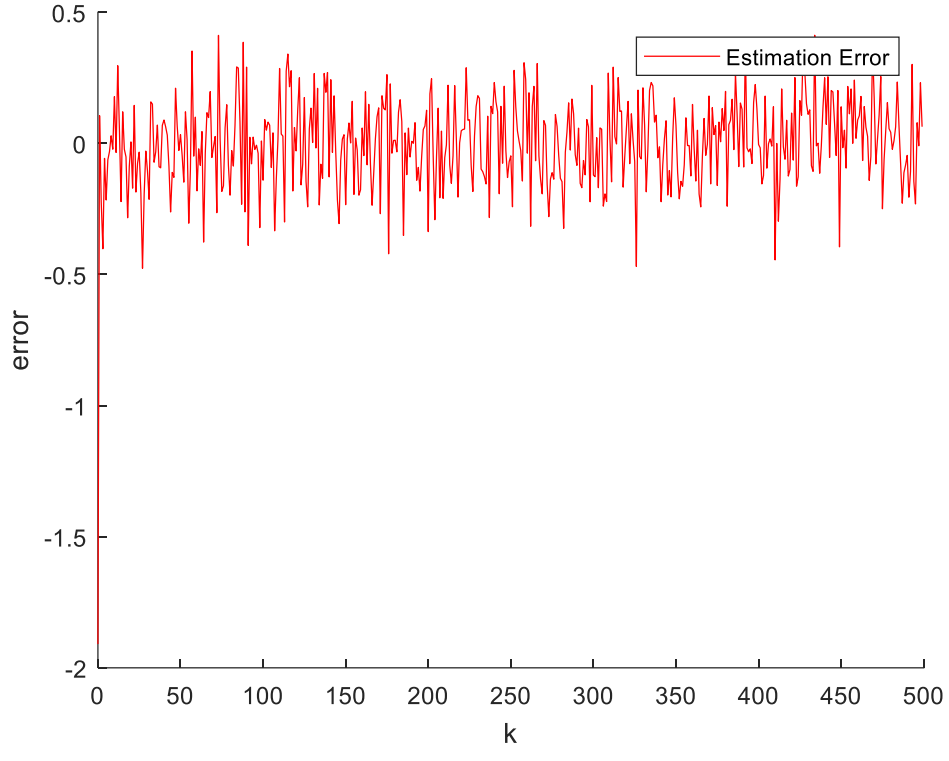

Figure 2.7 The output estimation error for second order system by using Kalman Filter

Now we will shortly discuss the effect of the value of noise covariances on the output estimation error. To show the effect, we use a first order system that is presented by equations (2.13) and (2.13) to estimate the state of the system by changing the covariance of process and measurement noise. Table 2.2 shows the result on the effect of noise on the error estimation. Based on the result, it can be concluded that when one or both of process and measurement noise increases, the estimation error also increases.

Table 2.2: The effect of values of covariance of the noise of the system on the error estimation

V	0.01	0.1	0.01	0.1
W	0.01	0.01	0.1	0.1
Error %	1.41	4.965	6.387	10.671

Now, we will discuss the properties of Kalman filter. There are several important properties of Kalman filter. If a system is linear, noises are additive, noises and initial state are Gaussian, the Kalman filter is the best filter in minimizing the estimation error covariance. However, Kalman filter is the best linear filter if the noises are not Gaussian. If the system is nonlinear, Extended Kalman filter can be used, which we will be discussed in the next section in this chapter.

In addition, some conditions limit the performance of the Kalman filter. These are: system and measurement equation coefficient matrices, the statistics of the initial sate and, process and measurement noises need to be known [1]. On the other hand, when noise statistics are uncertain but if the noises are of finite energy, H-infinity filter, which will be discussed in the next section, can be used.

2.1.2 H-infinity Filter

H-infinity filter (H∞) is an optimal filter which minimizes the maximum estimated error assuming that the noise has finite energy [1]. In this work, we will use a filter that is sub-optimal but can be easily extended to the nonlinear case. Consider a linear discrete linear system described by equations (2.20) and (2.21).

$$x_{k+1} = A\,x_k + B\,u_k + Fw_k \tag{2.20}$$

$$y_k = Cx_k + D\,u_k + G\,w_k \tag{2.21}$$

The state estimate can be computed using equation (2.22).

$$\hat{x}_{k+1} = A\,\hat{x}_k + B\,u_k + K_k^{\infty}\,(y_k - (C\hat{x}_k + D\,u_k)) \tag{2.22}$$

as in the Kalman filter.

The estimation error is defined the same way:

$$e_k = x_k - \hat{x}_k \tag{2.23}$$

Estimation error evolves in time as follows:

$$e_{k+1} = A\,x_k + B\,u_k + Fw_k - A\,\hat{x}_k - B\,u_k - K_k^{\infty}\,(Cx_k + D\,u_k + G\,w_k - C\hat{x}_k - D\,u_k)$$

which is simplified to:

$$e_{k+1} = (A - K_k^{\infty}\,C)e_k + (F - K_k^{\infty}G)w_k \tag{2.24}$$

Performance output (z_k) is defined by

$$z_k = C_z e_k \tag{2.25}$$

The positive definite Lyapunov candidate function is defined by equation (2.26) [4].

$$V_k = e_k^T P_k^{-1} e_k > 0 \tag{2.26}$$

where P_k will be called estimation error covariance because it will be found as the solution to a Riccati equation as in a Kalman filter, although physically w_k is not stochastic and therefore this is a misnomer. The H-infinity filter is based on the inequality

$$V_{k+1} - V_k + z_k^T z_k - \gamma w_k^T w_k \leq 0 \tag{2.27a}$$

where γ is called the attenuation constant. This inequality results in

$$\lim_{N\to\infty} \frac{1}{N} \sum_{k=0}^{N-1} z_k^T z_k - \gamma \lim_{N\to\infty} \frac{1}{N} \sum_{k=0}^{N-1} w_k^T w_k \leq 0 \tag{2.27b}$$

Since

$$\lim_{N\to\infty}\frac{1}{N}\sum_{k=0}^{N-1}V_{k+1}-\ V_k=\ V_0$$

because for a asymptotically stable filter $V_\infty = 0$ and assuming $e_0 = 0$ and the filter task is to reduce the effect of w_k on z_k.

(2.27b) leads to

$$\frac{\lim_{N\to\infty}\frac{1}{N}\sum_{k=0}^{N-1}z_k^T z_k}{\lim_{N\to\infty}\frac{1}{N}\sum_{k=0}^{N-1}w_k^T w_k}\leq\ \gamma$$

(2.28)

The ratio between energy of the performance output and energy of the noise must be equal or less than γ i.e. the required condition of H-infinity filter to work.

Inequality (2.27a) is used to derive the H-infinity filter. The gain of H-infinity filter (K_k^∞) is computed by

$$K_k^\infty = (A\left(P_k^{-1}-{C_z}^T C_z\right)^{-1}C^T+\gamma^{-1}FG)\left(C_k\left(P_k^{-1}-{C_z}^T C_z\right)^{-1}C^T+\gamma^{-1}GG^T\right)^{-1}$$

(2.29)

Estimation error covariance (P_k) is computed by

$$P_{k+1} = A\left(P_k^{-1} - {C_z}^T C_z\right)^{-1} A^T + \gamma^{-1} F F^T - \left(A\left(P_k^{-1} - {C_z}^T C_z\right)^{-1} C^T + \gamma^{-1} F G\right)\left(C\left(P_k^{-1} - {C_z}^T C_z\right)^{-1} C^T + \gamma^{-1} F G^T\right)^{-1}\left(C\left(P_k^{-1} - {C_z}^T C_z\right)^{-1} A^T + \gamma^{-1} G C^T\right) \tag{2.30}$$

Now, we will use the H-infinity filter to estimate states of first and second order systems when their noise energy is finite.

Consider a first order system that is presented by the following equations

$$x_{k+1} = A\, x_k + \mathrm{B}\, u_k + w_{1k} \tag{2.31}$$

$$y_k = C\, x_k + w_{2k} \tag{2.32}$$

where $A = 0.9, B = 0.09, C = 1,\ F = G = 1,\ w_{1k} = 5e^{-k}, w_{2k} = 3e^{-k}$, u_k is unit step function.

Note that it is assumed that the noise statistics are unknown, but their energies are finite, and the control input is known and measurements are available. To estimate the state of the system x_k, filter can be by applying equation (2.22), (2.29), and (2.30).

The initial values were set as

$$\hat{x}_0 = 12, \qquad P_0 = 100$$

The values for γ and C_z were chosen as

$$\gamma = 6 \quad , \quad C_z = 1$$

Figure 2.8 shows the actual and estimated value of x_k. The estimation error is presented in Figure 2.9. In addition, the percentage of the output estimation error is 0.499% that is computed using equations (2.16) and (2.17). The ratio of the energy of output and energy of noise is 1.272, which means the condition of H-infinity filter which is represented via equation (2.27) is verified, $1.272 \leq \gamma = 6$.

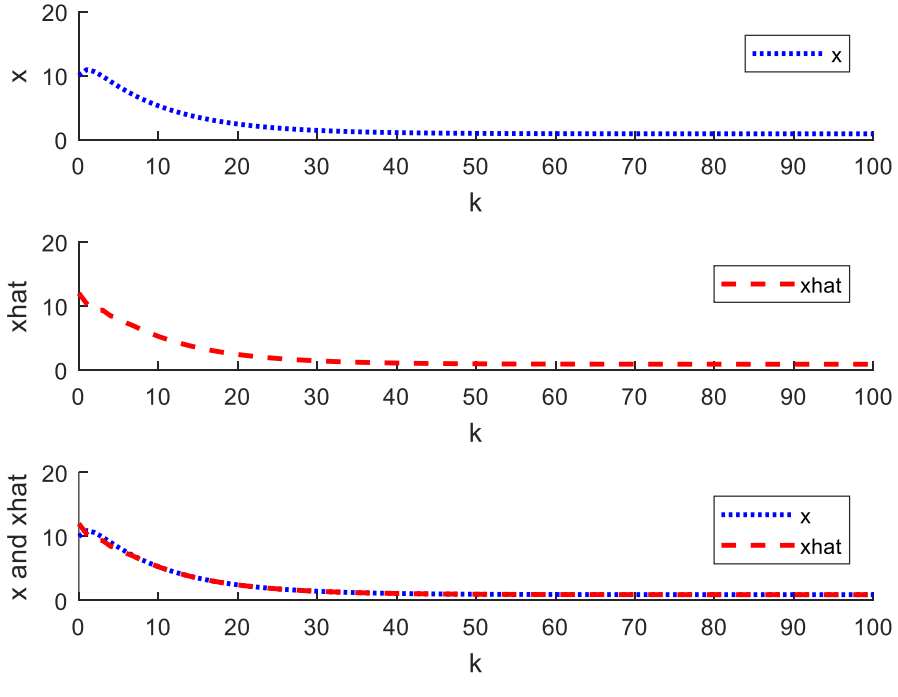

Figure 2.8: The actual and estimated value of the state of first order system by H-infinity filter

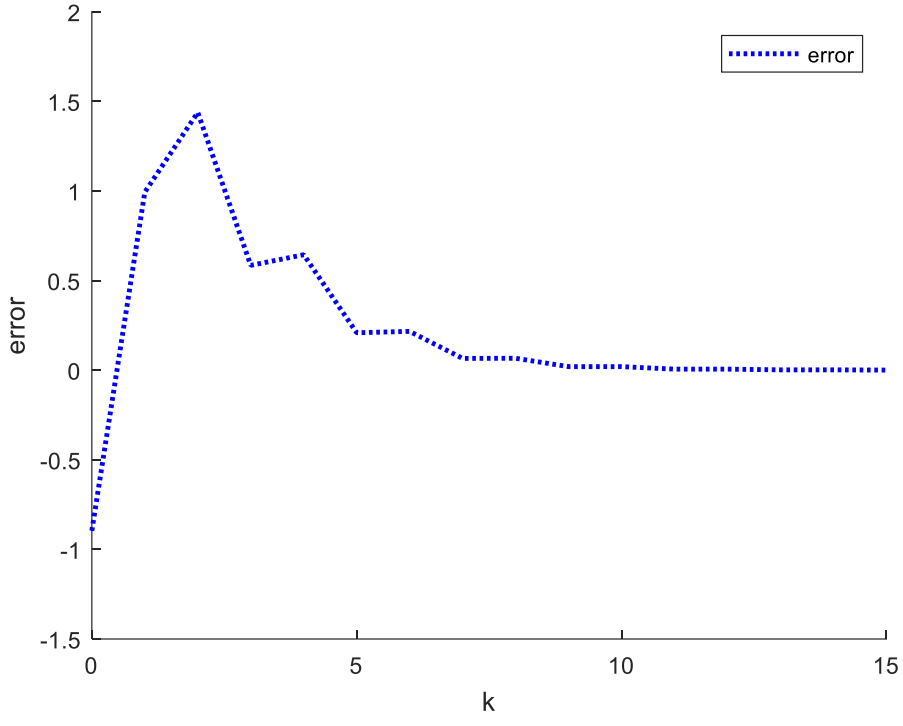

Figure 2.9: The output estimation error of state for first order system by H-infinity filter

Now, we want to estimate states for a second-order system that is presented by the following equations.

$$\begin{bmatrix} {x_{k+1}}^1 \\ {x_{k+1}}^2 \end{bmatrix} = A \begin{bmatrix} {x_k}^1 \\ {x_k}^2 \end{bmatrix} + B\, u_k + F\, w_{1,k} \tag{2.33}$$

$$y_k = C \begin{bmatrix} {x_k}^1 \\ {x_k}^2 \end{bmatrix} + G\, w_{2,k} \tag{2.34}$$

where $A = \begin{bmatrix} 1 & 0.1 \\ -1 & 0 \end{bmatrix}, B = \begin{bmatrix} 0.1 \\ 1 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 \end{bmatrix}, F = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, G = [1,0]$, u_k is unit step function ,$w_{1,k} = \begin{bmatrix} 10e^{-k} \\ 10e^{-k} \end{bmatrix}$, $w_{2,k} = [\, 3e^{-k}, 0]$.

By assuming that the control input is known and measurements are available; also, the states of the system are unmeasurable and the noise statistics are unknown but they are finite energy, we can use H-infinity filter to estimate the states of the system x_k by applying equation (2.22), (2.29), and (2.30).

The initial values were set as

$$\begin{bmatrix} \hat{x}_0{}^1 \\ \hat{x}_0{}^2 \end{bmatrix} = \begin{bmatrix} 12 \\ 1.2 \end{bmatrix}, \qquad P_0 = 100 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

The values of γ and C_z that were chosen

$$\gamma = 10 \text{ and } C_z = [1,1]$$

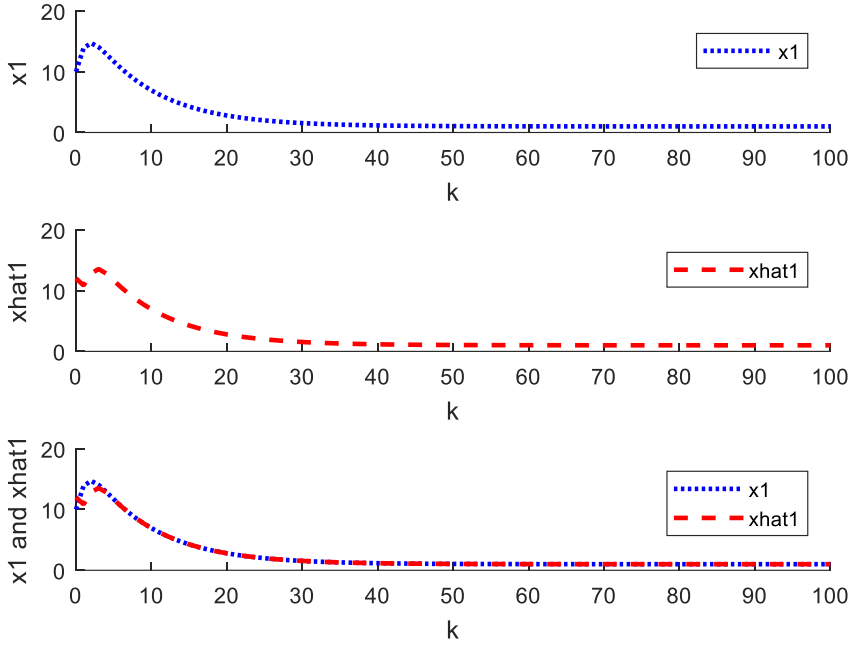

Figure 2.10: The actual and estimated value of the x1 by H-infinity filter

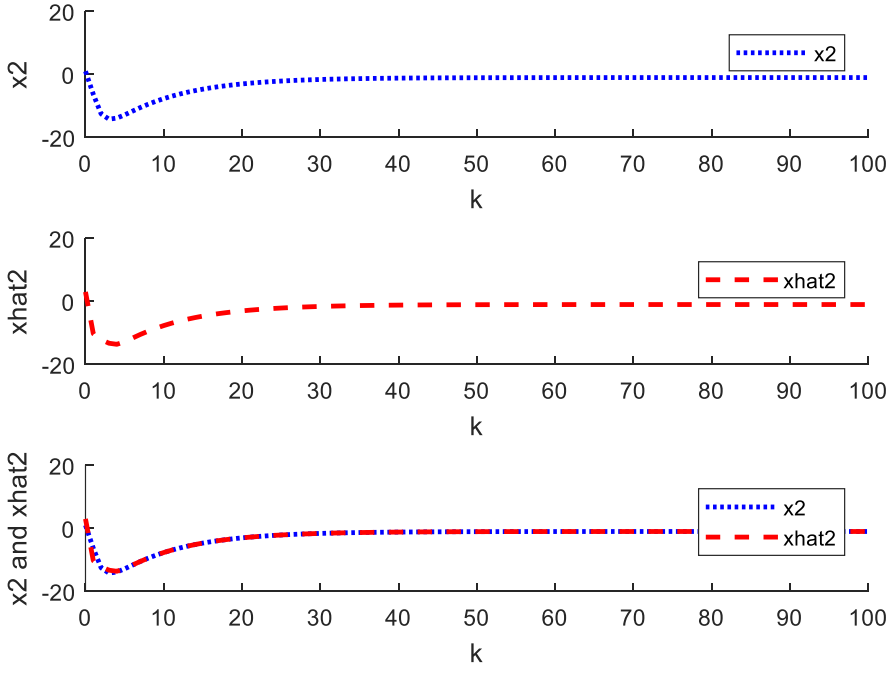

Figure 2.11: The actual and estimated value of the x2 by H-infinity filter

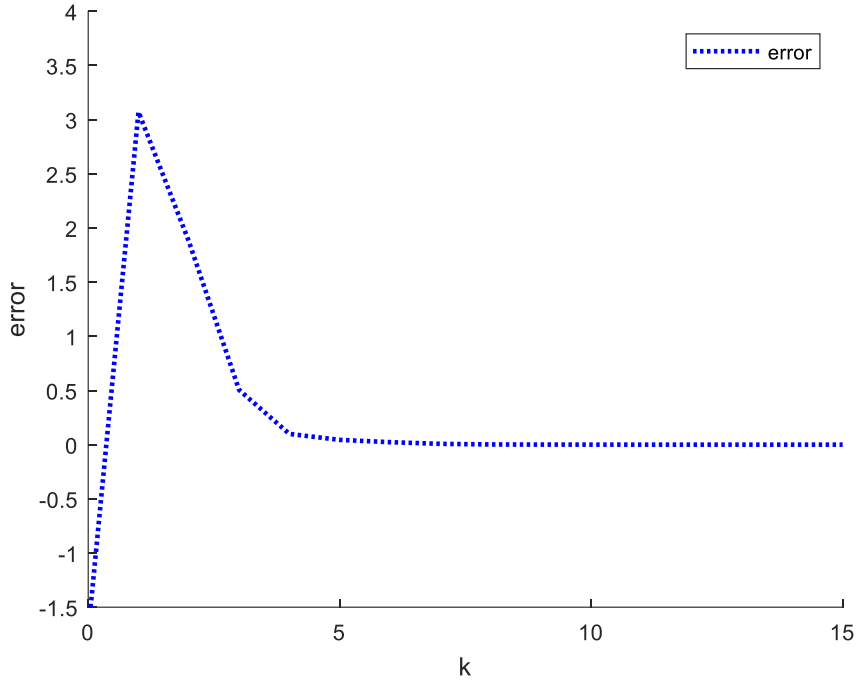

Figure 2.12: The output estimation error for second order system by H-infinity filter

Figure 2.10 and Figure 2.11 show the actual and estimated value of states; also, the estimation error that is computed by equation (2.15) is shown in Figure 2.12. In addition, the percentage of the estimation by this filter is 0.95%. Also, by equation (2.27), we found the ratio of the energy of output and the energy of noise is $2.14 \leq \gamma = 10$.

After we show how to estimate states of a system via H-infinity filter when noise statistics are unknown but energy of finite, we wanted to try to estimate a state of first order system when its noise is Gaussian distributed via H-infinity filter.

Consider a first order system that is presented by equation (2.12) and (2.13). By assuming the noise statistics are unknown, H-infinity filter can be used to estimate the state of the system.

The initial values were set as

$$\hat{x}_0 = 12, \qquad P_0 = 100$$

The values of γ and C_z that were chosen

$$\gamma = 15 , \quad Cz = 1$$

Figure 2.13 shows the actual and estimated value of x and Figure 2.8 shows the output estimation error. The percentage of the output estimation error is 1.01 % that was computed by equation (2.16) and (2.17).

From these results, it can be concluded that H-infinity filter can be used to estimate states of a linear system when noise statistics are unknown but energy of finite. In this case, H-infinity may work as an optimal filter. However, when the noise energy is not finite, H-infinity filter still is a good estimator, but it works as observer not as an optimal filter.

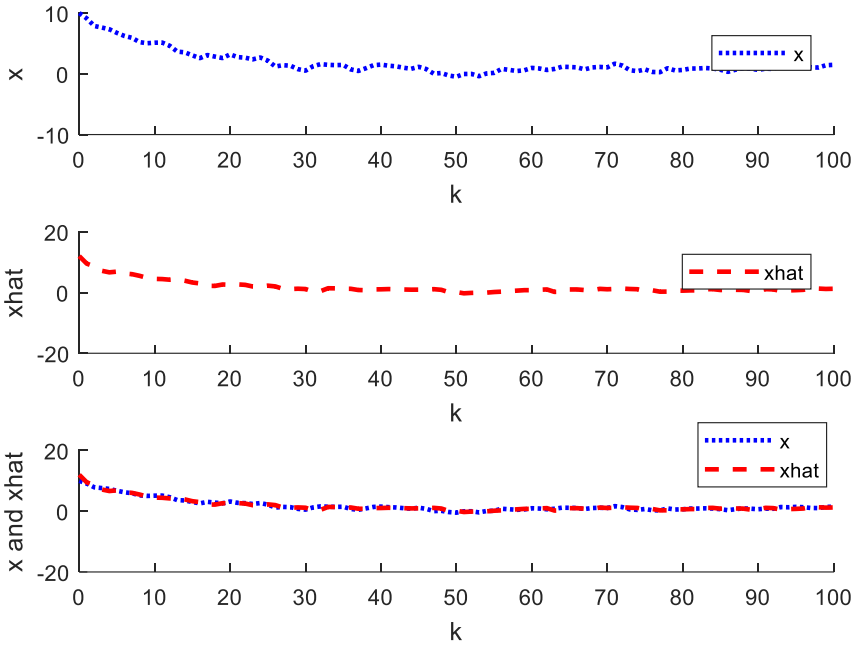

Figure 2.13: The actual and estimated value of the state of first order system by H-infinity filter

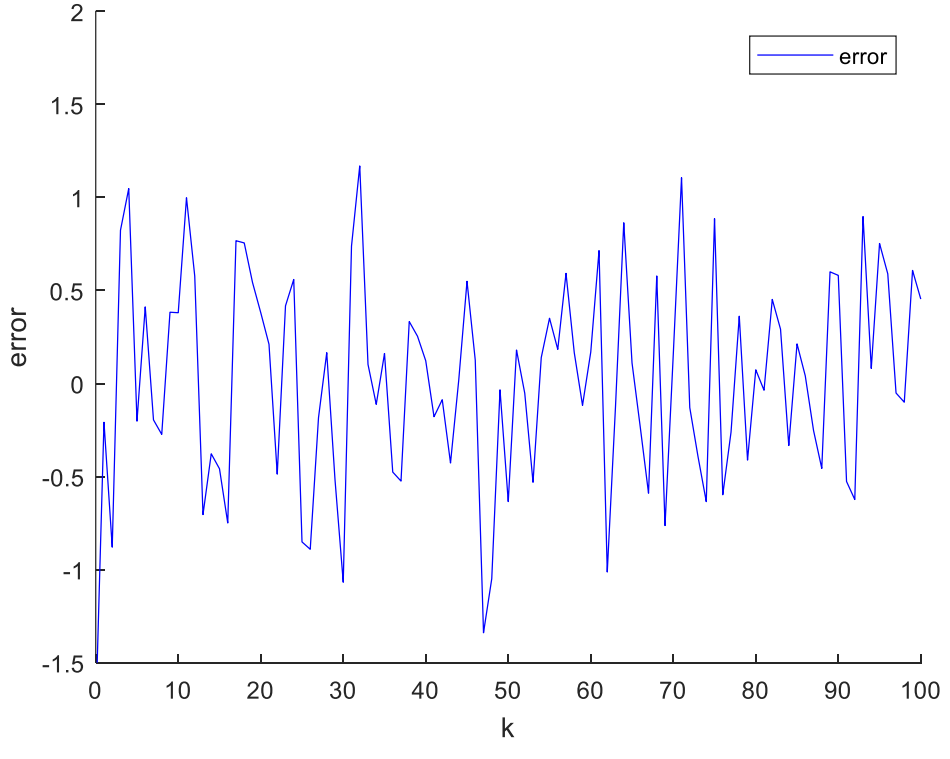

Figure 2.14: The output estimation error for first order system by H-infinity filter

Up to this point, we have presented two techniques to estimate the state of a linear discrete-time system when statistics of process and measurement noises are known (Gaussian distribution, zero mean, white) via the Kalman filter. In addition, when noise statistics are unknown, we have used the H-infinity filter. In the next section, we will show a bank of Kalman filters, that is a parallel set of Kalman filters which is used to estimate the parameters of a system but it has several other applications as well.

2.1.3 Bank of Kalman Filters

A bank of Kalman filters is a parallel set of Kalman filters fed by the same measurement sequence. There are many applications of a bank of Kalman filters, for example, it is used in [5] to estimate winding resistance of a motor. Another application of a bank of Kalman filter is used in [6] to detect and isolate sensor and actuator faults for aircraft engine. In this section, we will show how a bank of Kalman filters works. Moreover, in section 3.3, we will use a bank of Kalman filters to estimate uncertain parameters for a first order system. In addition, we will use a bank of Kalman filters to detect an uncertain intrusion signal for a first order system in section 4.2.

Consider linear time-invariant discrete-time below

$$x_{k+1} = A\,x_k + \mathrm{B}u_k + Fv_k \tag{2.35}$$

$$y_k = C\,x_k + \mathrm{G}w_k \tag{2.36}$$

and

$$v_k \sim N\ (0, V)\ \ ,\ \ w_k \sim N\ (0, W)$$

where A matrix includes many parameters. By assuming one of them, that is called α, is unknown but its value is in a set of hypothesis values that are $\alpha = \{\alpha_1, \alpha_2 \ldots \alpha_n\}$. Our objective is to find the actual value of α by a bank of Kalman filters. Since the parameter α has n possible values, a bank of n Kalman filters will be used. Each Kalman filter is designed for each possible value of α. The purpose of a bank of Kalman filters is to compute the conditional probability for each possible value by a Kalman filter. Equation shows how to compute the conditional probability of a Kalman filter by Bayes' theorem.

$$\mathcal{P}(\alpha_i|Y_k) = \frac{\mathcal{P}(\, y_k|Y_{k-1}, \alpha_i)\mathcal{P}(\alpha_i|Y_{k-1})}{\sum_{j=1}^{n=1} \mathcal{P}(\, y_k|Y_{k-1}, \alpha_j)\, \mathcal{P}\big(\alpha_j\big|Y_{k-1}\big)}$$

(2.37)

Since system's noise is a Gaussian distribution, the Gaussian probability density function give below can be used to compute $\mathcal{P}(\, y_k|Y_{k-1}, \alpha_i)$ [5].

$$\mathcal{P}(\, y_k|Y_{k-1}, \alpha_i) = \frac{1}{\sqrt{2\pi^{-r}}}\ \left|\Omega_{k|\alpha_i}^{-1}\right|^{\frac{1}{2}} exp(-\frac{1}{2}\ \tilde{y}_{k|\alpha_i}^{T}\ \Omega_{k|\alpha_i}^{-1}\tilde{y}_{k|\alpha_i})$$

(2.38)

where r is the order of the system, and Ω is the covariance of $\tilde{y}_k$.

Ω can be computed [5] as follows

$$\Omega = \ \mathrm{C_k}\ \mathrm{P_{k|\alpha_i}}\mathrm{C_k^T} + \mathrm{G_k}\mathrm{W}\mathrm{G_k^T}$$

(2.39)

$\tilde{y}_k$ is computed as

$$\tilde{y}_k = y_k - \hat{y}_{k|k-1,\alpha_i}$$

(2.40)

$\hat{y}_{k|k-1,\alpha_i}$ is computed by

$$\hat{y}_{k|k-1,\alpha_i} = C_k \hat{x}_{k|k-1,\alpha_i}$$

(2.41)

where $\hat{x}_{k|k-1,\alpha_i}$ is an estimated state that is computed by a Kalman filter based $\alpha = \alpha_i$. Now, we know the way to compute the conditional probability for each possible value. When a conditional probability of a possible value is close to one, it means that the algorithm has converged to the actual value of the parameter. Moreover, if we want to estimate more than one parameter, more Kalman filters can be added to the bank. The equation below can be used to compute how many Kalman filters are needed in the bank.

$$Number\ of\ KF = (Number\ of\ hypotheses)^{Number\ of\ parameter}$$

(2.42)

For example, if we want to find the actual values for three parameters, each with five hypotheses, a bank of Kalman filters needs 125 Kalman filters to find the actual values of them. Finally, a bank of Kalman filters is a linear filter which has many applications, two of which we will show in chapters 3 and 4.

2.2 Nonlinear Filters

After we show how to estimate the state of a linear system via Kalman and H-infinity filters in section 2.1, we will discuss how to estimate the state of a nonlinear system in this section by Extended Kalman and Nonlinear H-infinity filters.

2.2.1 Extended Kalman Filter

The extended Kalman filter is used to estimate the state of a nonlinear system when noise statistics are known (Gaussian distribution, zero mean with a certain covariance, white). The extended Kalman filter is estimated by linearizing a system around current estimated state. In this section, we will show how Extended Kalman filter works and will show some examples to estimate a state of nonlinear systems.

Consider a nonlinear system that is presented by (2.43) and (2.44).

$$x_{k+1} = f\ (x_k, u_k, v_k) \tag{2.43}$$

$$y_k = g\ (x_k, u_k, w_k) \tag{2.44}$$

By linearizing a nonlinear system around the current state estimate via Taylor series (with ignoring the high order terms of a Taylor series), we can find A_k, B_k, C_k and G_k that are presented below [3, 8]:

$$A_k = \left.\frac{\delta f}{\delta x}\right|_{x=\hat{x}_k,u,v=0} \tag{2.45}$$

$$F_k = \left.\frac{\delta f}{\delta v}\right|_{x=\hat{x}_k,u,v=0} \tag{2.46}$$

$$C_k = \left.\frac{\delta g}{\delta x}\right|_{x=\hat{x}_k,u,w=0} \tag{2.47}$$

$$G_k = \left.\frac{\delta g}{\delta w}\right|_{x=\hat{x}_k,u,w=0} \tag{2.48}$$

The gain of Extended Kalman filter is found as

$$K_k = (A_k P_k {C_k}^T)\left(C_k P_k {C_k}^T + G_k W_k {G_k}^T\right)^{-1} \tag{2.49}$$

The estimated state is computed as

$$\hat{x}_{k+1} = f(\hat{x}_k, u_k, 0) + K(y_k - g(\hat{x}_k, u_k, 0)) \tag{2.50}$$

the approximate estimation error covariance is found from the Riccati equation

$$P_{k+1} = A_k P_k A_k^T - A_k P_k C_k^T \left(C_k P_k C_k^T + G_k W_k G_k^T\right)^{-1} C_k P_k A_k^T \tag{2.51}$$

Now, we use Extended Kalman filter to estimate a state of a nonlinear systems. Consider the nonlinear system that is presented below

$$x_{k+1} = -x_k^2 + v_k \tag{2.52}$$

$$y_k = x_k + w_k \tag{2.53}$$

with

$$v_k \sim N(0,0.01) \quad w_k \sim N(0,0.01)$$

By assuming the measurements are available and the states of the system are unknown, Extended Kalman filter is used to estimate the state of the nonlinear system. By equations (2.45) – (2.48), Jacobian matrices can be calculated as

$$A_k = -2\,\hat{x}_k\,,\ F_k = 1\,, C_k = 1, G_k = 1$$

By substituting into Extended Kalman filter equations; (2.49), (2.51), and (2.50). Setting the initial values to

$$\hat{x}_0 = 1.1, \qquad P_0 = 1000$$

Figure 2.15 shows us the actual and estimated value of the state. In addition, we show the output estimation error in Figure 2.16. Moreover, the percentage of the output estimation error that is computed by equation (2.16) and (2.17) is 1.726 %.

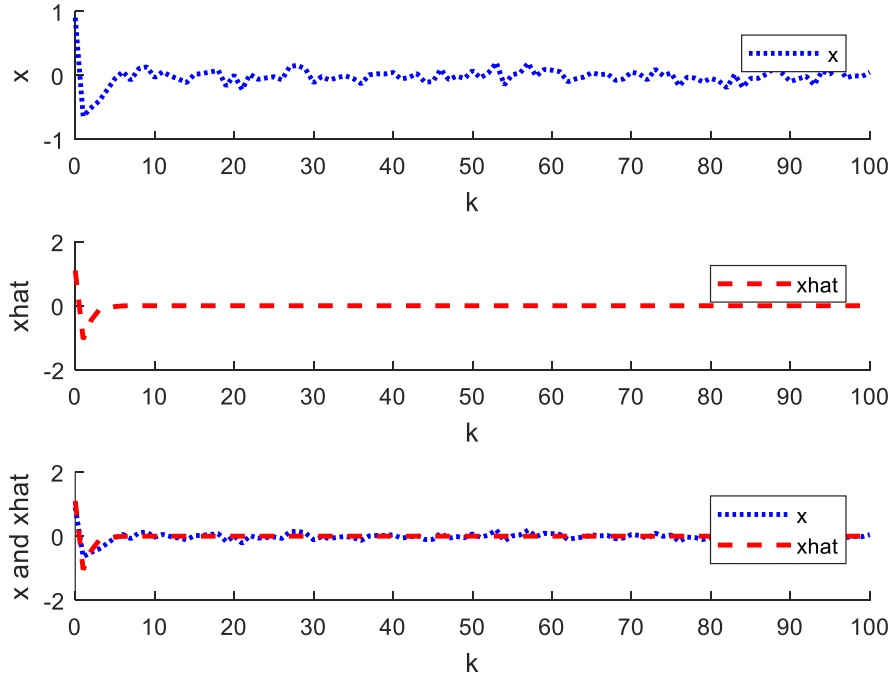

Figure 2.15: The actual and estimated value of the state of nonlinear system by Extended Kalman filter

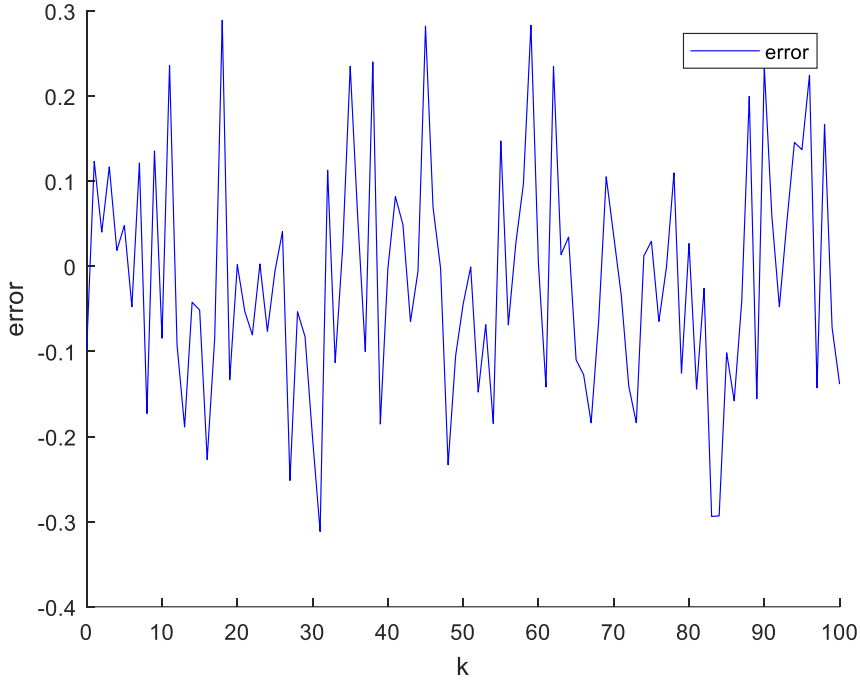

Figure 2.16: The output estimation error of state for nonlinear system by Extended Kalman filter

The next nonlinear system whose state is estimated is below:

$$x_{k+1} = -x_k^3 + v_k \tag{2.54}$$

$$y_k = x_k + w_k \tag{2.55}$$

with

$$v_k \sim N\,(0,0.01) \qquad w_k \sim N\,(0,0.01)$$

We assume that the measurements are available, and the states are unmeasurable. By equations (2.45) – (2.48), the Jacobian matrices are

$$A_k = -3\,x_k^2\,,\ F_k = 1\,, C_k = 1, G_k = 1$$

Using the same initial condition and by substituting into filter equations, we show the results of estimation in Figure 2.17 that presents the actual and estimated values of the state, and Figure 2.18 shows the output estimation error and the percentage of the output estimation error is found to be 2.447 % and it is computed by equations (2.16) and (2.17).

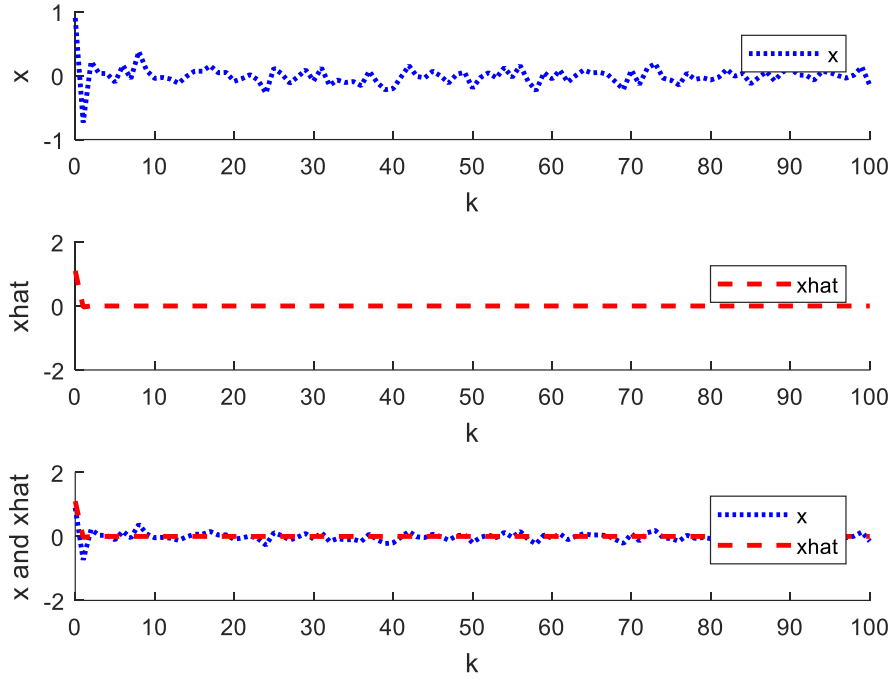

Figure 2.17: The actual and estimated value of the state of nonlinear system by Extended Kalman filter

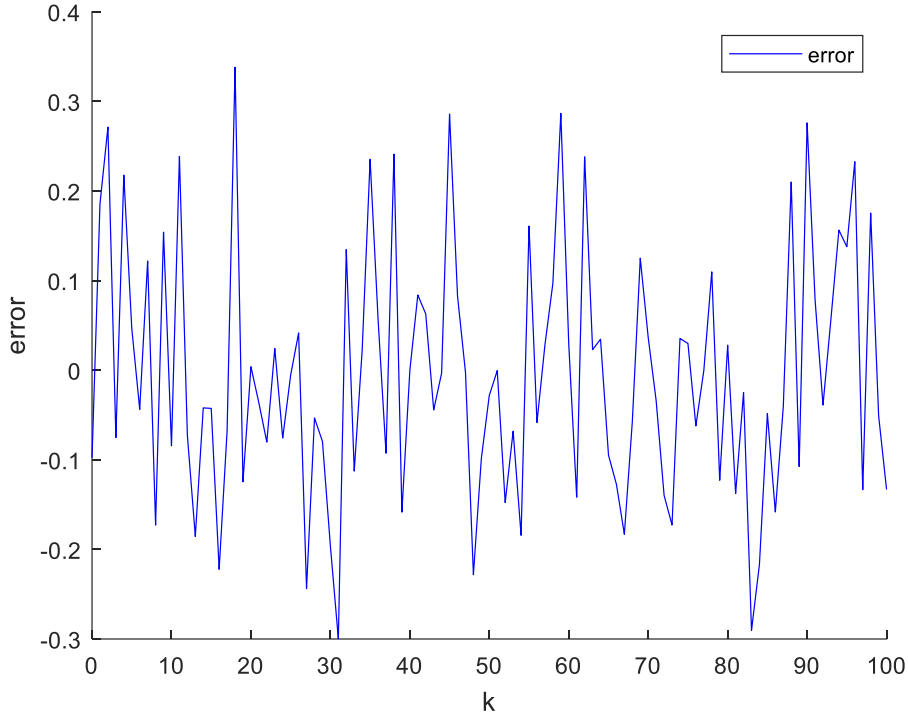

Figure 2.18: The output estimation error of state for nonlinear system by Extended Kalman filter

The last system whose state is estimated is presented by below:

$$x_{k+1} = -\sin(x_k) + v_k \tag{2.56}$$

$$y_k = x_k + w_k \tag{2.57}$$

with

$$v_k \sim N\,(0,0.01) \qquad w_k \sim N\,(0,0.01)$$

From equations (2.45) – (2.48) the Jacobian matrices are

$$A_k = -\cos(x_k),\ F_k = 1, C_k = 1, G_k = 1$$

By setting up the Extended Kalman filter with the same initial conditions, Figure 2.19 shows the actual and estimated values of the state of the nonlinear system. In addition, the output estimation error is shown in Figure 2.20. The percentage of the output estimation error is 0.81 %.

In the examples considered in this section, the Extended Kalman is shown to effectively estimate the state of nonlinear systems when noise statistics are known.

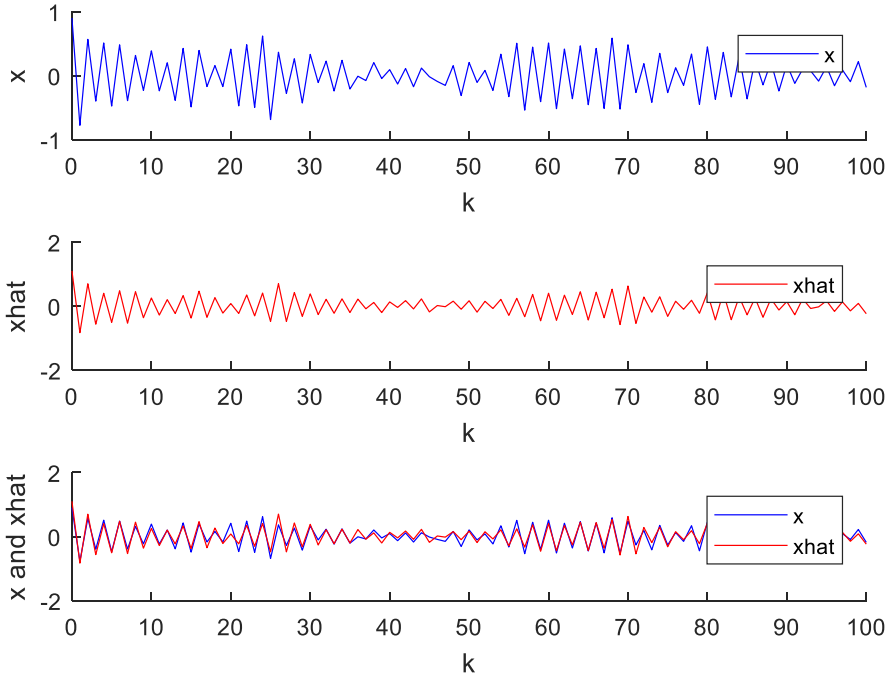

Figure 2.19: The actual and estimated values of the state of nonlinear system by Extended Kalman filter

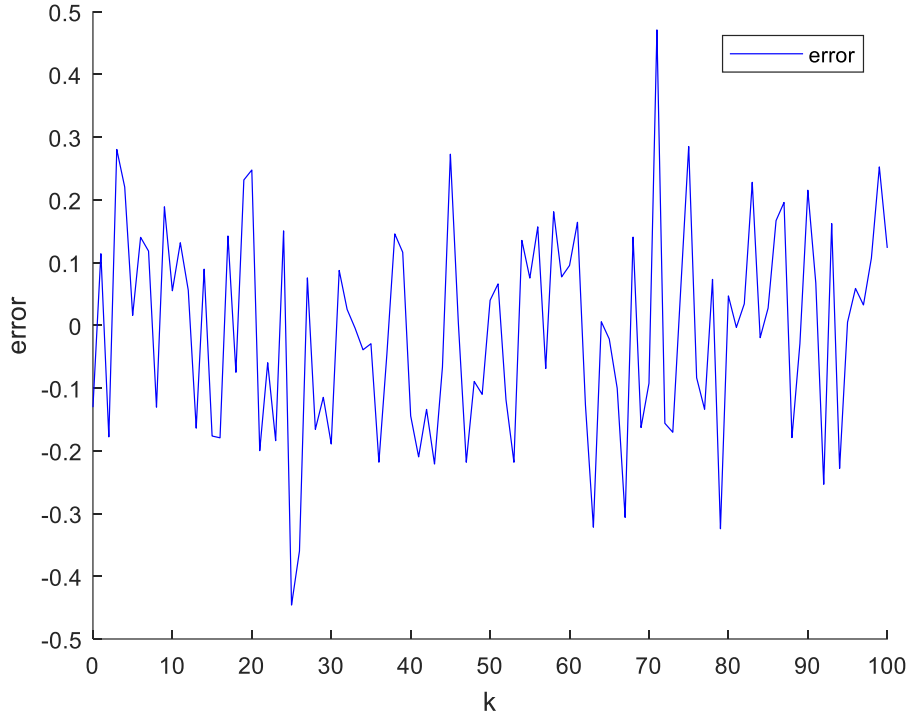

Figure 2.20: The estimation error of state for nonlinear system by Extended Kalman filter

2.2.2 Nonlinear H-infinity Filter

In section 2.1.2, we showed how to estimate the state for linear systems when statistics of the noises are uncertain. In this section, we will show a novel method to estimate the state of a nonlinear system when noise statistics are unknown but the noise is of finite energy type. The method is a combination of the Extended Kalman and linear H-infinity filters to estimate the state of a nonlinear system by linearizing the system around the current estimated state. The derivation of this filter given in the paper that is being prepared [17]. Consider a nonlinear system given by in equations

$$x_{k+1} = f(x_k, w_k) \tag{2.58}$$

$$y_k = g(x_k, w_k)$$

(2.59)

where w_k is a finite energy type noise with otherwise unknown properties.

By assuming the form of the state estimate update equation as

$$\hat{x}_{k+1} = f(\hat{x}_k) + K_k^{\infty}(y_k - g(\hat{x}_k))$$

(2.60)

The estimation error is defined as below:

$$e_k = x_k - \hat{x}_k$$

and it evolves in time as

$$e_{k+1} = f(x_k, w_k) - f(\hat{x}_k) + K_k^{\infty}(y_k - g(\hat{x}_k))$$

(2.61)

An approximate estimation error (e_k) equation that is computed via truncating of Taylor series is given by:

$$e_{k+1} \approx (A_k - K_k^{\infty} C_k)e_k + (F_k - K_k^{\infty} C_k)w_k$$

Jacobian matrices are found as used:

$$A_k = \left.\frac{\delta f}{\delta x}\right|_{x=\hat{x}_k, w=0} \qquad F_k = \left.\frac{\delta f}{\delta w}\right|_{x=\hat{x}_k, w=0}$$

$$C_k = \left.\frac{\delta g}{\delta x}\right|_{x=\hat{x}_k, w=0} \qquad G_k = \left.\frac{\delta g}{\delta w}\right|_{x=\hat{x}_k, w=0}$$

(2.62)

The performance output is defined as

$$z_k = C_z e_k$$

(2.63)

The gain of the H-infinity filter is given by

$$K_k^{\infty} = (A_k\left(P_k^{-1} - {C_z}^T C_z\right)^{-1} {C_k}^T + \gamma^{-1} F_k G_k)\left(C_k\left(P_k^{-1} - {C_z}^T C_z\right)^{-1} {C_k}^T + \gamma^{-1} G_k {G_k}^T\right)^{-1}$$

(2.64)

where the Riccati equation is:

$$P_{k+1} = A_k\left(P_k^{-1} - {C_z}^T C_z\right)^{-1} {A_k}^T + \gamma^{-1} F_k {F_k}^T - K_k^{\infty}\left(C_k\left(P_k^{-1} - {C_z}^T C_z\right)^{-1} A_k^T + \gamma^{-1} G_k {F_k}^T\right)$$

(2.65)

Now, we will estimate states of some nonlinear systems by this H-infinity filter. Consider the nonlinear system that is presented by the following:

$$x_{k+1} = -\sin(x_k) + \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} 2e^{-k} \\ 2e^{-2k} \end{bmatrix}$$

(2.66)

$$y_k = x_k + \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} 2e^{-k} \\ 2e^{-2k} \end{bmatrix}$$

(2.67)

By assuming the measurements are available and the noise statistics are unknown (noise energy is finite), we can be use the nonlinear H-infinity filter to estimate the state. Jacobian matrices are found by equation (2.62)

$$A_k = -\cos(\hat{x}_k)\ , F_k = [1,0]\ , C_k = 1, \quad G_k = [0,1]$$

The initial values were set as

$$\hat{x}_0 = 1.1, \qquad P_0 = 1000$$

The values for γ and C_z were chosen as

$$\gamma = 2 \quad , \quad C_z = 0.5$$

By setting up into the nonlinear H-infinity filter, Figure 2.21 show the actual and estimated values of the state. The estimation error is presented in Figure 2.22. The percentage of estimation error that is computed via equations (2.16) is 0.809 %.

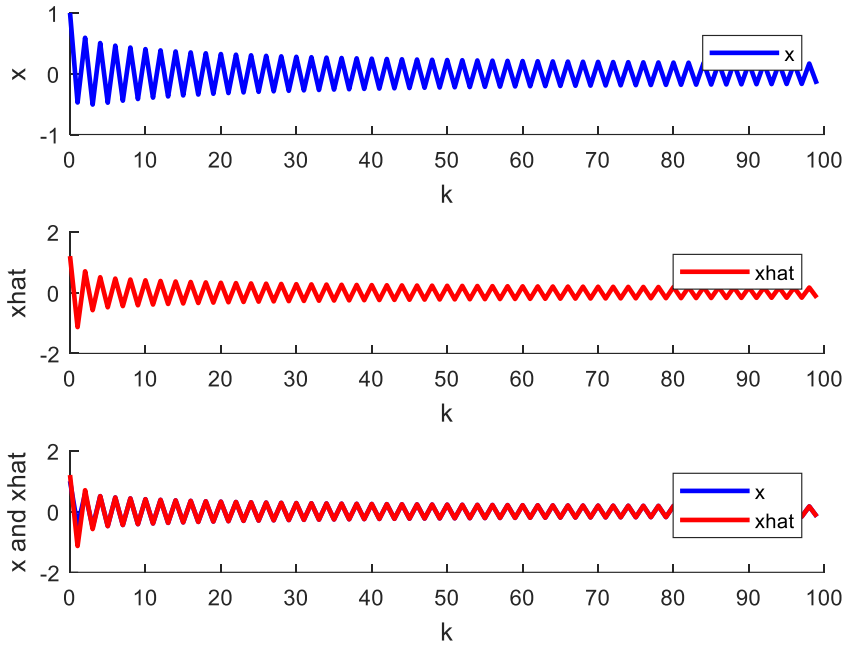

Figure 2.21: The actual and estimated values of the state of the nonlinear system by H-infinity Filter

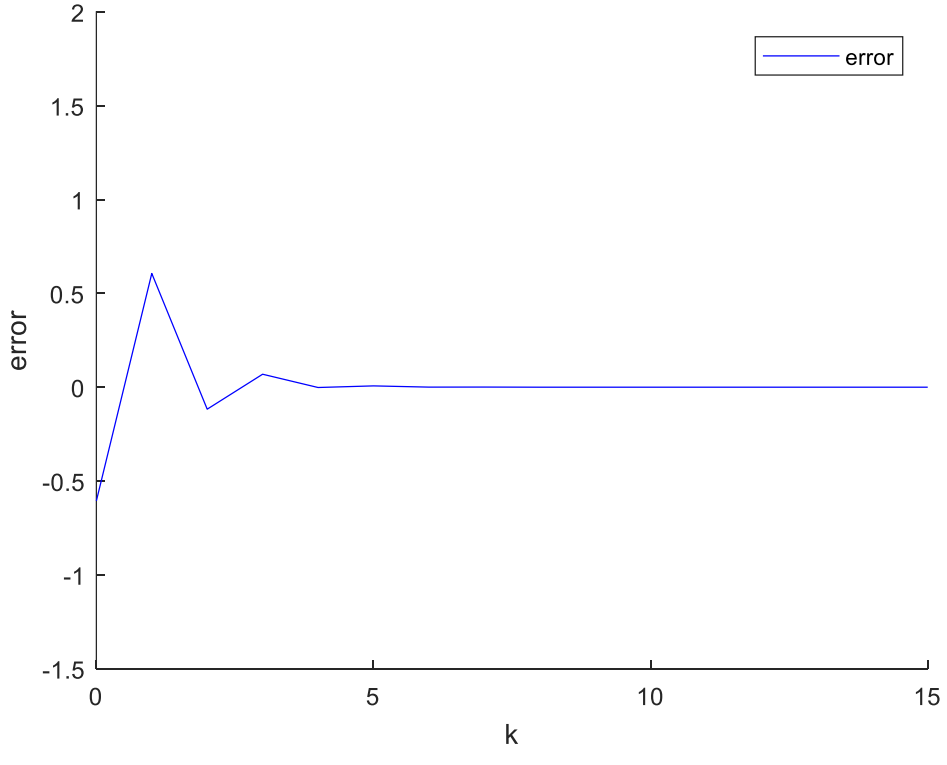

Figure 2.22 : The output estimation error of the state of the nonlinear System by H-infinity Filter

The next system that we want to estimate is the state of a system presented by equations:

$$x_{k+1} = -x_k^2 + [1 \quad 0]\begin{bmatrix} 0.3e^{-k} \\ -2e^{-2k} \end{bmatrix} \tag{2.68}$$

$$y_k = x_k + [0 \quad 1]\begin{bmatrix} 0.3e^{-k} \\ -2e^{-2k} \end{bmatrix} \tag{2.69}$$

Jacobian matrices are found by equation (2.62).

$$A_k = -2\hat{x}_k \ , \quad F_k = 1 \ , \ C_k = 1 \ , \ G_k = 1$$

The initial values were set as

$$\hat{x}_0 = 0.4 \ , \qquad P_0 = 100$$

The values for γ and C_z were chosen as

$$\gamma = 2 \quad , \quad C_z = 0.5$$

By setting up into H-infinity filter to estimate the state, we obtained the estimation result that includes actual and estimated values of the state as shown in Figure 2.23. Figure 2.22 shows us the estimation error. The percentage of estimation error is 0.3447 %.

From these results, it can be seen that this nonlinear H-infinity filter can be used to effectively estimate the state of the nonlinear system when the noise statistics are unknown by linearizing the system around the current estimated state.

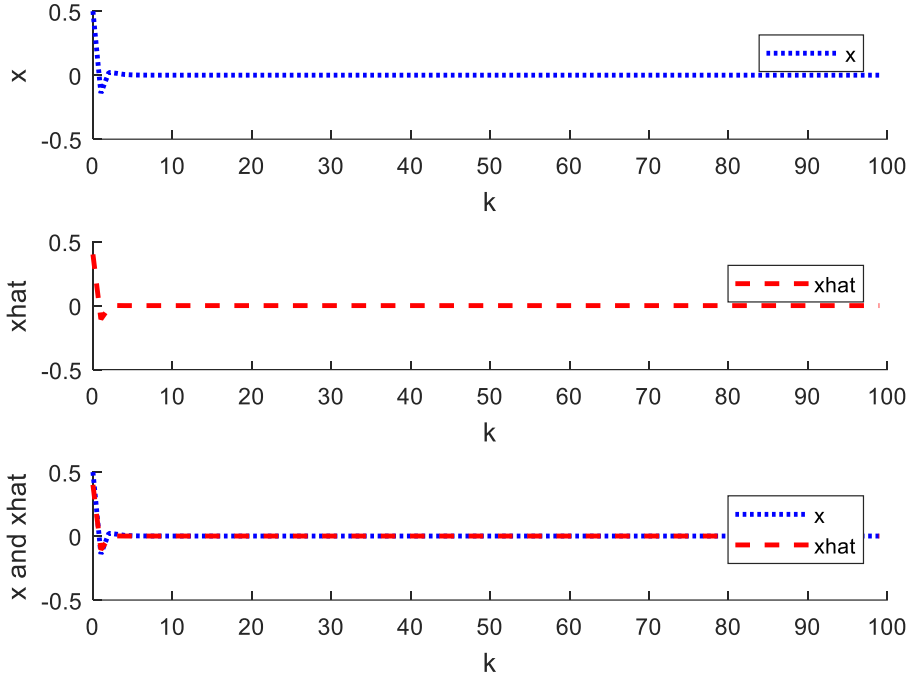

Figure 2.23: The actual and estimated values of the state of the nonlinear filter by H-infinity

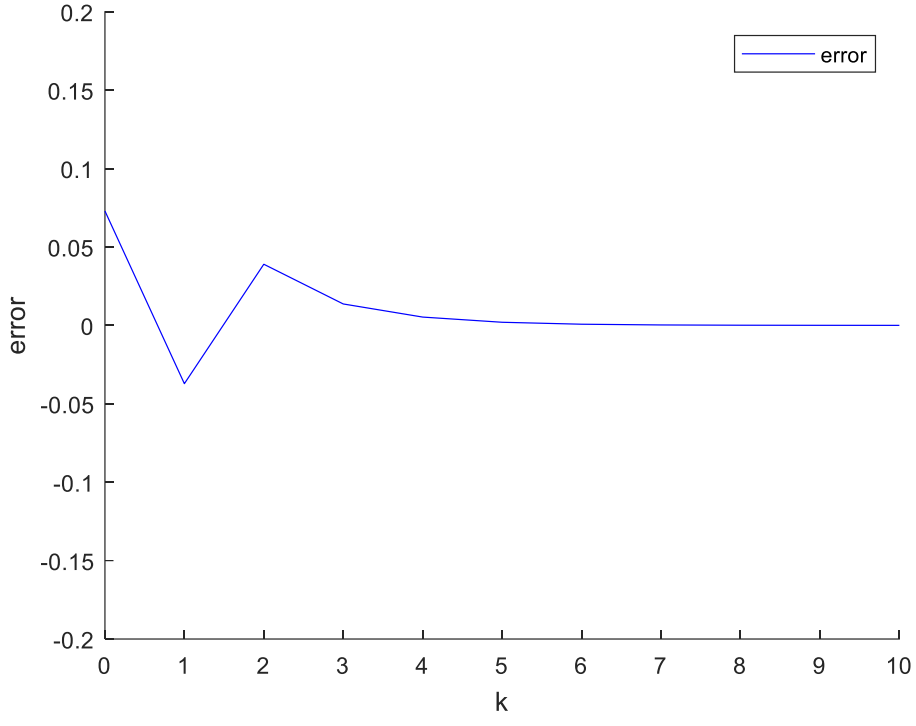

Figure 2.24: The output estimation error of the state of the nonlinear System by H-infinity Filter

In this section, we have presented two techniques to estimate the state of a nonlinear discrete-time system by linearizing the system around the current estimated state. The first technique, Extended Kalman filter can be used to estimate the state of nonlinear systems when statistics of process and measurement are known. However, when noise statistics are unknown but the noise is of finite energy, nonlinear H-infinity can be used to estimate the state.

3 Case Studies of Parameter Estimation by Various Methods

In this chapter, several methods that can be used to estimate the parameters of a stochastic system are presented. First, parameter estimation technique using linear optimal filters, specifically, using Kalman filter and H-infinity filters are presented. These linear optimal filters will be used to estimate the coefficients of a transfer function. Next, parameter estimation technique using nonlinear optimal filter, Extended Kalman filter and H-infinity filter are presented and will be demonstrated using a state-space model of a system. Finally, parameter estimation technique using a bank of Kalman filters are presented. The bank of Kalman filters will be used to estimate an actual value of a parameter when it has many possible hypotheses.

3.1 Estimation of Coefficients in Transfer Functions

In this section, we will discuss how to estimate coefficients of a transfer function using linear optimal filters (which was discussed in Chapter 2) [7]. Then, the estimated parameters will be validated by comparing simulated measurement data and estimated measurement data.

Consider a discrete time transfer function which as shown in equation (3.1), where m $\leq$ n,

$$Y_z = \frac{b_0 + b_1 z^{-1} + \cdots + b_m z^{-m}\, U_z + w_z}{1 + a_1 z^{-1} + \cdots + a_n z^{-n}} \tag{3.1}$$

Equation (3.1) can be rewritten as,

$$Y_z(\,1 + a_1 z^{-1} + \cdots + a_n z^{-n}) = (b_0 + b_1 z^{-1} + \cdots + b_m z^{-m}) U_z + w_z \tag{3.2}$$

By taking inverse Z- transform the following equation can be obtained,

$$y_k + \, a_1 y_{k-1} + \cdots + a_n y_{k-n} = \, b_0 u_k + b_k u_{k-1} + \cdots + \, b_m u_{k-m} + w_k \tag{3.3}$$

Equation (3.3) can be rewritten as,

$$y_k = \, - a_1 y_{k-1} - \cdots - a_n y_{k-n} + b_0 u_k + b_k u_{k-1} + \cdots + \, b_m u_{k-m} + w_k$$

$$y_k = \, C_k x_k \, + w_k \tag{3.4}$$

where,

$$C_k = \, [- y_{k-1}, -y_{k-2}\,, \ldots, -y_{k-n}\,, u_k, u_{k-1}, u_{k-2}, \ldots, u_{k-m}]$$

$$x_k = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \\ b_0 \\ b_1 \\ \vdots \\ b_m \end{bmatrix}$$

Note that x_k represents the constant coefficients of a transfer function. Therefore, x_k does not change in time and can be represented as

$$x_{k+1} = x_k \tag{3.7}$$

The state-space model of the transfer function can be represented as,

$$x_{k+1} = Ax_k \tag{3.8a}$$

$$y_k = C_k x_k + Gw_k \tag{3.8b}$$

where,

$$A = \begin{bmatrix} 1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & 1 \end{bmatrix}$$

$$C_k = [-y_{k-1}, -y_{k-2}, \dots, -y_{k-n}, u_k, u_{k-1}, u_{k-2}, \dots, u_{k-m}]$$

$$G = 1$$

Using the state-space model of equation (3.8), linear optimal filters can be used to estimate the states of the system (i.e. the coefficients of the transfer function of equation (3.1). Estimation techniques using optimal filters will be demonstrated using the following cases:

- Case 1:

In the first case, it is assumed that the statistics of noise is known. In this case, Kalman filter will be used to estimate the coefficients of transfer function. Consider a first order system which is presented in equation (3.9).

$$Y_z = \frac{b_0\, U_z + w_z}{1 + a_1 z^{-1}} \tag{3.9}$$

where $a_1 = 0.9$, $b_0 = 1$, assuming that the coefficients are unknown and measurement and the control input are known. After taking the inverse Z-transform of equation (3.9) and simplifing it, the following equation can be obtained,

$$x_{k+1} = Ax_k \tag{3.10a}$$

$$y_k = C_k x_k + Gw_k \tag{3.10b}$$

where $A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $C_k = [-y_{k-1}, u_k]$, $x_k = \begin{bmatrix} a_{1,k} \\ b_{0,k} \end{bmatrix}$

$$w_k \sim N\,(0,0.1)$$

Based on state-space model of equation (3.10), equations (2.3), (2.10), and (2.11), can be modified into

$$\hat{x}_{k+1} = \hat{x}_k + K_k\,(y_k - C_k\hat{x}_k) \tag{3.11}$$

$$K_k = (P_k {C_k}^T)\left(C_k P_k {C_k}^T + W_k\right)^{-1} \tag{3.12}$$

$$P_{k+1} = P_k - P_k {C_k}^T\left(C_k P_k {C_k}^T + W_k\right)^{-1} C_k P_k \tag{3.13}$$

The system of equation (3.9) was simulated by assuming $a_1 = 0.9$, $b_0 = 1$ using MATLAB. The simulated measurement data was then used to test the estimation technique using state-space model of equation (3.10) and modified Kalman filter equations (3.11) – (3.13). The initial values were set as

$$\begin{bmatrix}\hat{a}_{1,k=0}\\ \hat{b}_{0,k=0}\end{bmatrix} = \begin{bmatrix}0.85\\ 1.5\end{bmatrix}\ ,\quad P_{k=0} = 10^3\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$$

Figure 3.1 shows us the actual and estimated values of coefficients obtained using Kalman filter. The estimated values of the coefficients, were found to be

$$\hat{a}_1 = 0.891 \quad , \quad \hat{b}_0 = 0.996$$

$$Percentage\ of\ Estimation\ Error_a = |a - \hat{a}| \times a^{-1} \times 100 = 1\,\%$$

$$Percentage\ of\ Estimation\ Error_b = \left|b - \hat{b}\right| \times b^{-1} \times 100 = 0.4\ \%$$

It should be noted that in real life, one will not know the actual value of the coefficients. In such cases, one has to come up with a method to check the performance of the estimator.

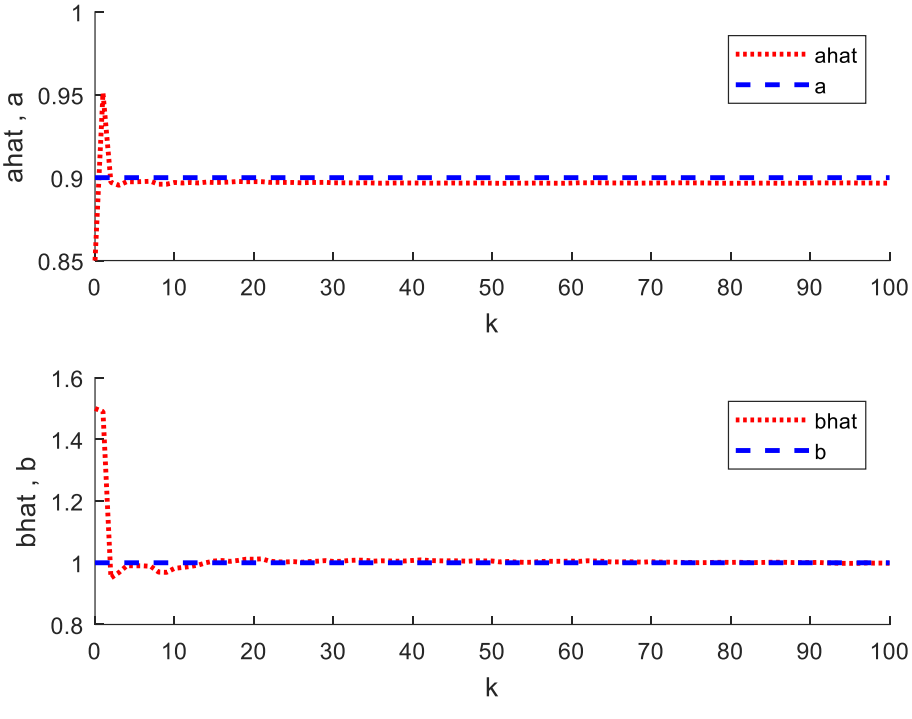

Figure 3.1 : The actual and estimated values of coefficients via Kalman filter

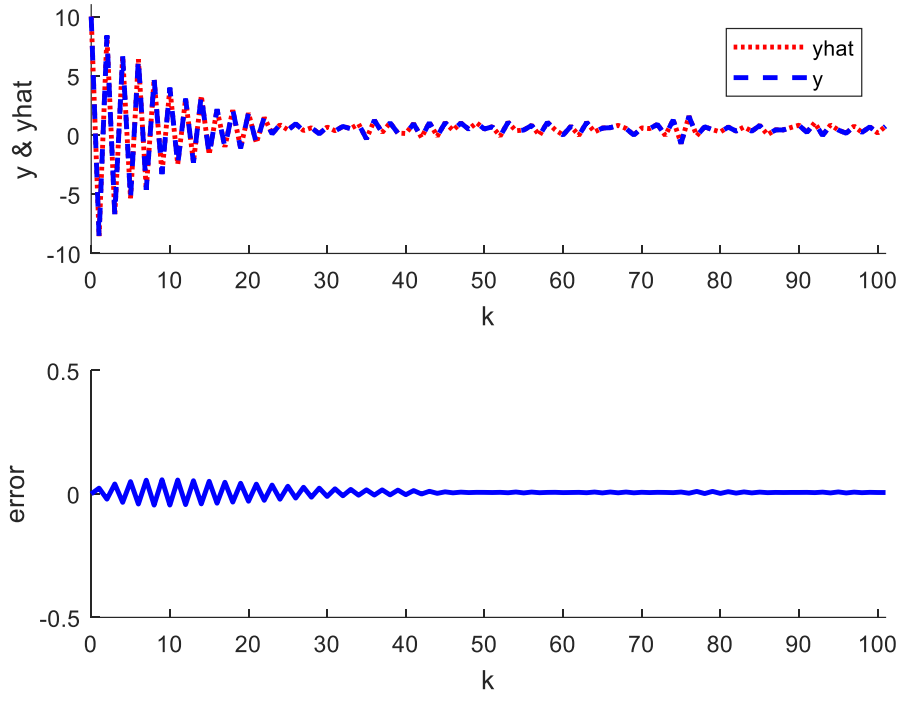

Figure 3.2: The actual and estimated values of measurement and the estimation error

One possible method to evaluate the performance of the estimator is by comparing estimated measurement with the actual measurement.

The estimated measurement is computed by simulating a system with estimated coefficients. Figure 3.2 shows us the actual and estimated measurement and the error between them. From Figure 3.2, it can be concluded that Kalman filter effectively estimates the coefficients of transfer function when noise statistics is known.

- Case 2:

In this case, the same system as discussed in Case 1 will be studied. However, here it is assumed that the noise has finite energy and the statistics of the noise unknown, The noise

$$w_k = 5\, e^{-k}$$

will be used in simulation. In this case, H-infinity filter will be used to estimate the coefficients. Based on this assumption and state-space model of equation (3.10), equations (2.22), (2.29), and (2.30) are modified,

$$\hat{x}_{k+1} = \hat{x}_k + K_\infty\,(y_k - C_k\hat{x}_k) \tag{3.14}$$

$$K_k^\infty = \left(P_k^{-1} - C_z{}^T C_z\right)^{-1} C_k{}^T \left(C_k\left(P_k^{-1} - C_z{}^T C_z\right)^{-1} C_k{}^T + \gamma^{-1}\right)^{-1} \tag{3.15}$$

$$P_{k+1} = \left(P_k^{-1} - C_z{}^T C_z\right)^{-1} - \left(\left(P_k^{-1} - C_z{}^T C_z\right)^{-1} C_k^T\right)\left(C\left(P_k^{-1} - C_z{}^T C_z\right)^{-1} C_k^T + \gamma^{-1}\right)^{-1}\left(C_k\left(P_k^{-1} - C_z{}^T C_z\right)^{-1}\right) \tag{3.16}$$

The initial values were set as

$$\begin{bmatrix}\hat{a}_{1,k=0}\\ \hat{b}_{0,k=0}\end{bmatrix} = \begin{bmatrix}0.75\\ 1.2\end{bmatrix}\ ,\ P_{k=0} = 10^3\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$$

The values for γ and C_z were chosen as

$$\gamma = 10\ \ ,\ \ C_z = [0.1\,, 0.1\,]$$

After getting the simulated measurements of the system using MATLAB, the estimated values of the coefficients were obtained via equations (3.14), (3.15), and (3.16). The results obtained are

$$\hat{a}_1 = 0.888 \ , \quad \hat{b}_0 = 1.004$$

$$Percentage\ Estimated\ Errror_a = (a - \hat{a}) \times a^{-1} \times 100 = 1.33\ \%$$

$$Percentage\ Estimated\ Errror_b = \left(b - \hat{b}\right) \times b^{-1} \times 100 = 0.4\ \%$$

The estimation results are shown in Figure 3.3 and Figure 3.4. Figure 3.4 shows the actual and estimated values of measurements and the estimation error. From the values of the error, it can be concluded that the estimated values of coefficients are very close to the actual values of coefficients. Hence, this method under these conditions is a good way to estimate coefficients of transfer function.

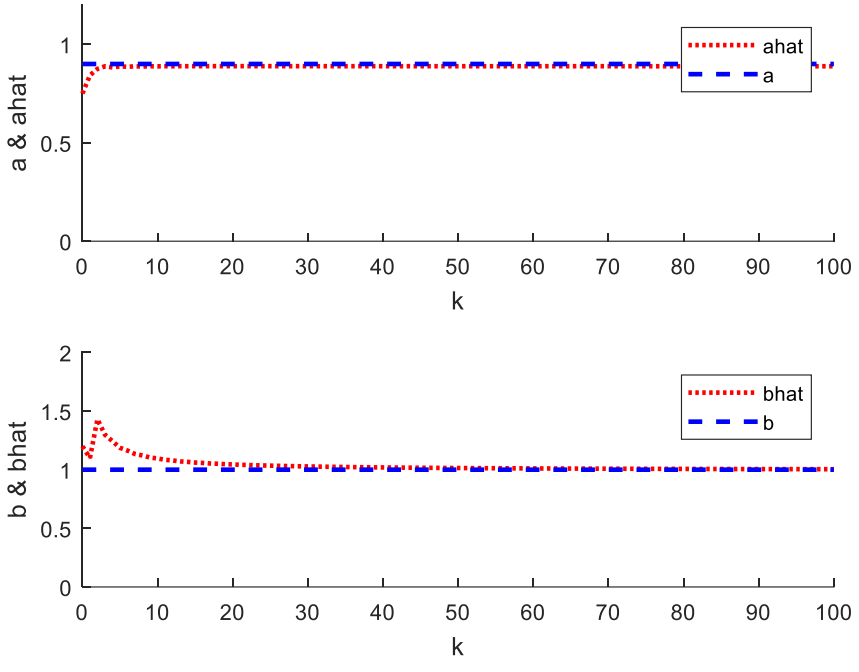

Figure 3.3: The actual and estimated values of coefficients via H-infinity filter

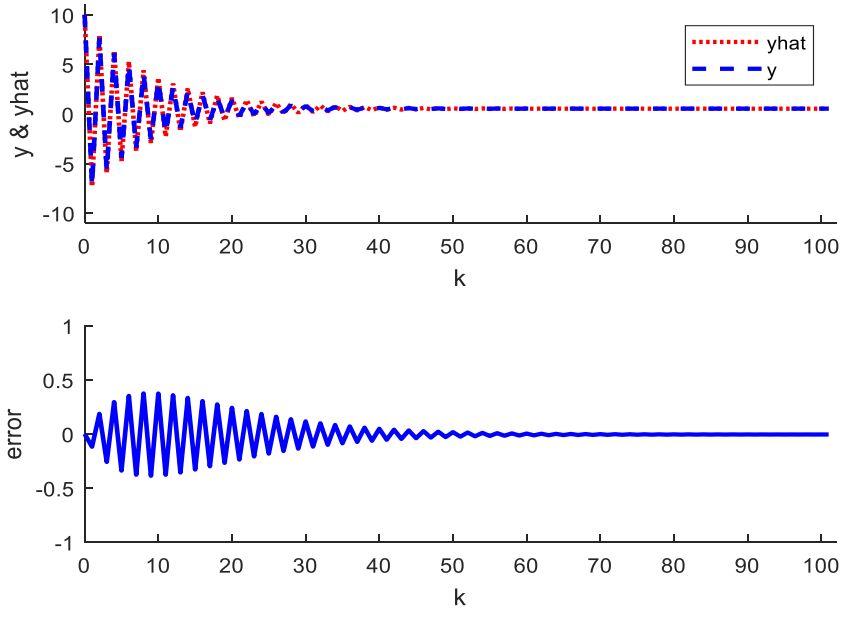

Figure 3.4: The actual and estimated values of measurement and the estimation error

Case 3:

In this case, the coefficients of the transfer function will be estimated the using H-infinity filter when the noise is Gaussian distributed. H-infinity filter in this case will function as an observer and not as an optimal filter. The noise statistic used in this case is

$$w_k \sim N\ (0,0.1)$$

Here the values of γ and C_z are chosen as

$$\gamma = 10 \quad , \quad C_z = [0.1\ , 0.1\ ]$$

The initial values were set as

$$\begin{bmatrix} \hat{a}_{1,k=0} \\ \hat{b}_{0,k=0} \end{bmatrix} = \begin{bmatrix} 0.75 \\ 1.5 \end{bmatrix} \ , \ P_{k=0} = \ 10^3 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

By using the H-infinity filter to estimate the coefficients, the results of the estimation were obtained as follows

$$\hat{a}_1 = \ 0.898 \quad , \quad \hat{b}_0 = 0.997$$

$$Percentage\ Estimated\ Errror_a = \ |a - \hat{a}| \times a^{-1} \times 100 \ = \ 0.22\ \%$$

$$Percentage\ Estimated\ Errror_b = \ \left|b - \hat{b}\right| \times b^{-1} \times 100 \ = \ 0.3\ \%$$

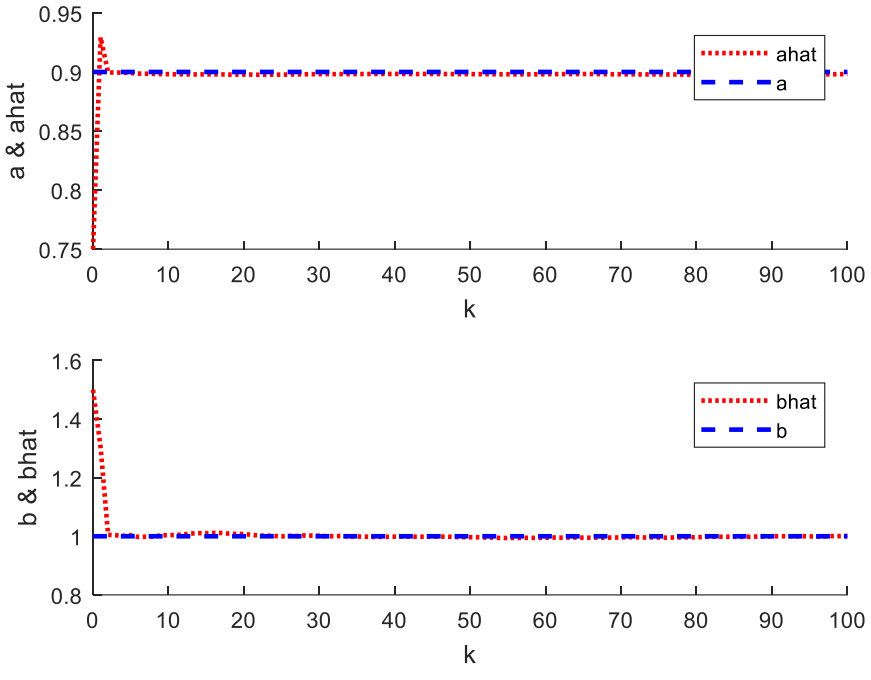

Figure 3.5: The actual and estimated values of coefficients via H-infinity filter

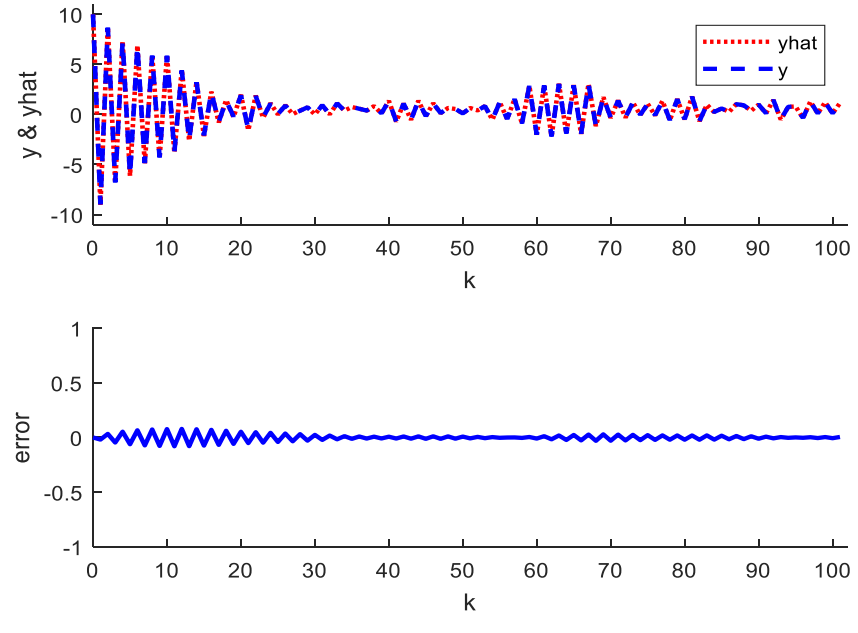

Figure 3.6: The actual and estimated values of measurement and the error via H-infinity filter

Figure 3.5 shows the estimated values of coefficients, which were estimated via the H-infinity filter when the noise follows a Gaussian distribution. In addition, the error between the actual and estimated values of measurement are shown in Figure 3.6. From Figure 3.5, it is clear that H-infinity filter has the ability to estimate coefficients of the transfer function when the noise of a model is a Gaussian distributed. However, note that the H-infinity filter functions as an observer in this case and not as an optimal filter because the noise energy is not finite.

In conclusion, in this section, two parameter estimation techniques have been discussed included the Kalman filter and the H-infinity filter. Based on the results obtained, it can be concluded that both of these methods are effective in estimating coefficients of transfer functions. In the next section, we will show how to estimate parameters based on state-space representation.

3.2 Simultaneous Parameter / State Estimation

After we showed how to use a linear optimal filter (Kalman and H-infinity filters) to estimate coefficients of a transfer function, in this section, we will show how to use nonlinear filters (i.e. Extended Kalman and nonlinear H-infinity filters) to estimate parameters of a state space model. Estimation techniques using Extended Kalman filter will be demonstrated using the following cases

- Case 1:

Consider a first order system, which is represented in equation (3.17).

$$x_{k+1} = ax_k + v_k \tag{3.17a}$$

$$y_k = cx_k + w_k \tag{3.17b}$$

where $a = 0.9$, $c = 1$, $x_{k=0} = 5$, $v_k \sim N\ (0,0.1)$, and $w_k \sim N\ (0,0.1)$, assume that a is unknown and the measurement is available. The constant unknown parameter of state-space model can be represented as

$$a_{k+1} = a_k \tag{3.18}$$

From equations (3.17) and (3.18), the nonlinear system can be modeled as shown in equation (3.19).

$$\begin{bmatrix} x_{k+1} \\ a_{k+1} \end{bmatrix} = \begin{bmatrix} f_1(x_k, a_k, v_k) \\ f_2(x_k, a_k, v_k) \end{bmatrix} \tag{3.19a}$$

$$y_k = g(x_k, a_k, w_k) \tag{3.19b}$$

where

$$f_1(x_k, a_k, v_k) = a_k x_k + v_k$$

$$f_2(x_k, a_k, v_k) = a_k$$

$$g(x_k, a_k, w_k) = cx_k + w_k.$$

By using equations (2.45),(2.46),(2.47), and (2.48), we found

$$A_k = \left.\frac{\delta f}{\delta x}\right|_{x=\hat{x}_k, v=0} = \begin{bmatrix} \hat{a}_k & \hat{x}_k \\ 0 & 1 \end{bmatrix} \qquad C_k = \left.\frac{\delta g}{\delta x}\right|_{x=\hat{x}_k, w=0} = [1 \quad 0]$$

$$F_k = \left.\frac{\delta f}{\delta v}\right|_{x=\hat{x}_k, v=0} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \qquad G_k = \left.\frac{\delta g}{\delta w}\right|_{x=\hat{x}_k, w=0} = 1$$

By using, the equation (2.50), the state estimate equations for the model can be represented as

$$\hat{x}_{k+1} = \hat{a}_k \hat{x}_k + K_k(y_k - \hat{x}_k) \tag{3.20a}$$

$$\hat{a}_{k+1} = \hat{a}_k + K_k(y_k - \hat{x}_k) \tag{3.20b}$$

The initial values were set as

$$\begin{bmatrix}\hat{x}_0\\ \hat{a}_0\end{bmatrix} = \begin{bmatrix}7\\ 0.8\end{bmatrix}\ ,\ \ P_{k=0} = \ 10^3 \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$$

The system that is presented in equation (3.17) was simulated using MATLAB in order to obtain the measurement data which will be used to test the estimation technique. By applying the Extended Kalman filter algorithm, the unknown parameter was found to be

$$\hat{a} = \ 0.892$$

$$Percentage\ Estimated\ Errror_a = \ |a - \hat{a}| \times a^{-1} \times 100 \ = \ 0.88\ \%$$

Figure 3.7 shows the actual and estimated value of the parameter obtained using Extended Kalman filter. The actual and estimated measurement and the error between them are shown in Figure 3.8. From Figure 3.8, it can be concluded that Extended Kalman filter can be used to estimate the parameters of state-space model when noise statistics are known.

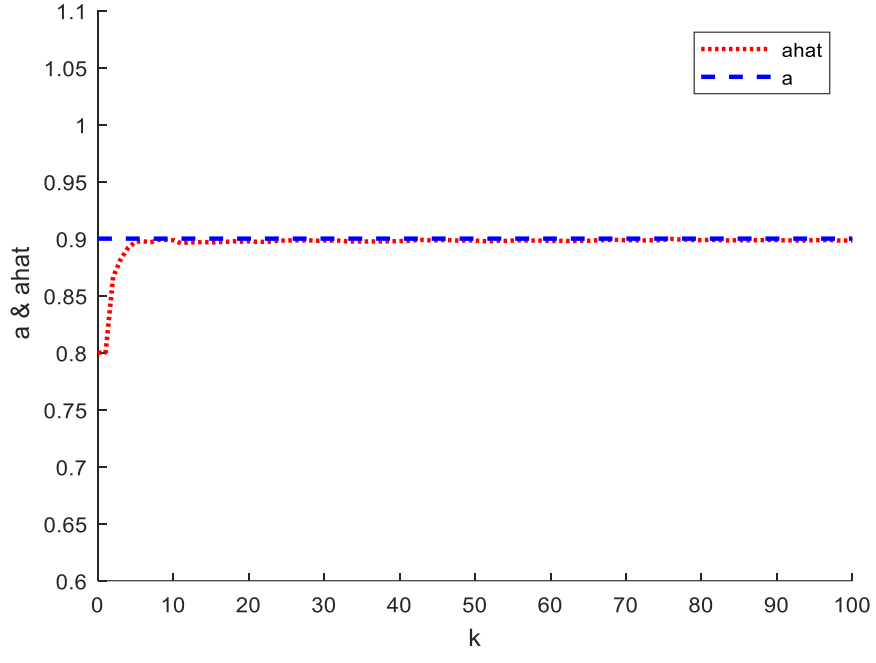

Figure 3.7: The actual and estimated value of parameters via Extended Kalman filter

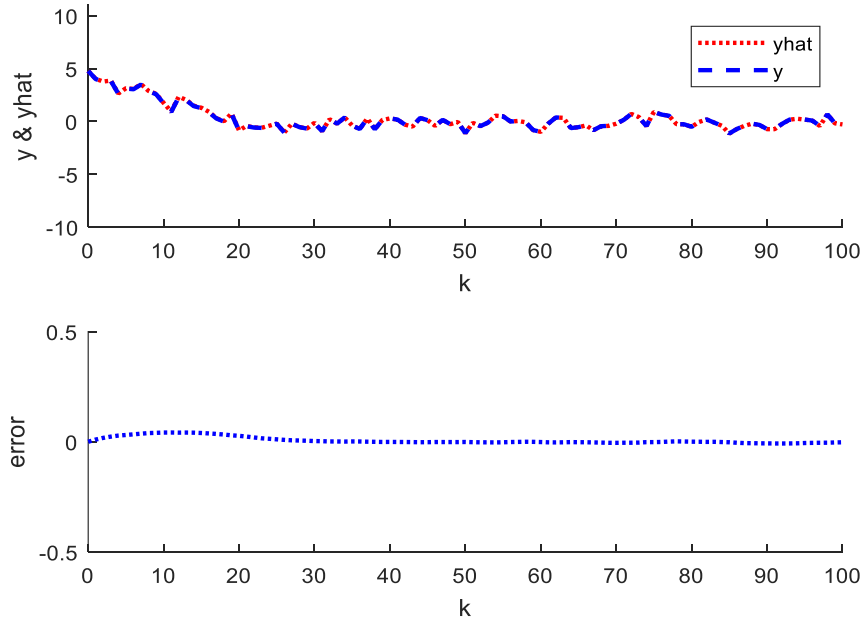

Figure 3.8: The actual and estimated values of measurement and the error via Extended Kalman filter

- Case 2:

Consider a first order system that as presented in equations (3.21).

$$x_{k+1} = ax_k + w_k \tag{3.21a}$$

$$y_k = c\, x_k + w_k \tag{3.21b}$$

where $a = 0.9\,, c = 1, x_{k=0} = 10\,,\ w_k = \begin{bmatrix} 3e^{-k} \\ 5e^{-2k} \end{bmatrix}$, assuming the measurement is available and that a is unknown. Assuming the unknown parameter is constant, it can be represented as shown in equation (3.18). A nonlinear system that is presented in equation (3.22) is modeled by using equations (3.21) and (3.18).

$$\begin{bmatrix} x_{k+1} \\ a_{k+1} \end{bmatrix} = \begin{bmatrix} a_k x_k \\ a_k \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 3e^{-k} \\ 5e^{-2k} \end{bmatrix} \tag{3.22a}$$

$$y_k = c\, x_k + \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} 3e^{-k} \\ 5e^{-2k} \end{bmatrix} \tag{3.22b}$$

By equation (2.62),

$$A_k = \left.\frac{\delta f}{\delta x}\right|_{x=\hat{x}_k, w=0} = \begin{bmatrix} \hat{a}_k & \hat{x}_k \\ 0 & 1 \end{bmatrix}\ ,\quad C_k = \left.\frac{\delta g}{\delta x}\right|_{x=\hat{x}_k, w=0}$$

$$F_k = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \quad , \quad G_k = \begin{bmatrix} 0 & 1 \end{bmatrix}$$

By equation (2.60), the estimated state equations for the model is presented by equation (3.23)

$$\hat{x}_{k+1} = \hat{a}_k \hat{x}_k + K_k^{\infty}(y_k - \hat{x}_k)$$

(3.23a)

$$\hat{a}_{k+1} = \hat{a}_k + K_k^{\infty}(y_k - \hat{x}_k)$$

(3.23b)

After getting the measurement by simulation from MATLAB, it can be used to validate the H-infinity filter's ability to estimate the unknown parameter, by using equation (2.64), (2.65), and (3.23).

The initial values were set as

$$\begin{bmatrix} \hat{x}_0 \\ \hat{a}_0 \end{bmatrix} = \begin{bmatrix} 9 \\ 0.8 \end{bmatrix} \ , \ P_{k=0} = 10^3 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

The values of γ and C_z are chosen as

$$\gamma = 25 \quad , \quad C_z = [1 \ , 1 \ ]$$

The result obtained is

$$\hat{a} = 0.902$$

$$Percentage\ Estimated\ Errror_a = |a - \hat{a}| \times a^{-1} \times 100 = 0.22\ \%$$

The actual and estimated values of parameter is shown in Figure 3.9, also, Figure 3.10 shows us the actual and estimated values of measurement and their error. Based on this result, it can be concluded that H-infinity filter can be used to estimate the unknown parameter in the state-space model when the noise statistics are unknown, but with finite energy.

In conclusion, the estimation technique based on nonlinear filters (such as Extended Kalman and H-infinity filters) can be used to estimate parameters of state-space model.

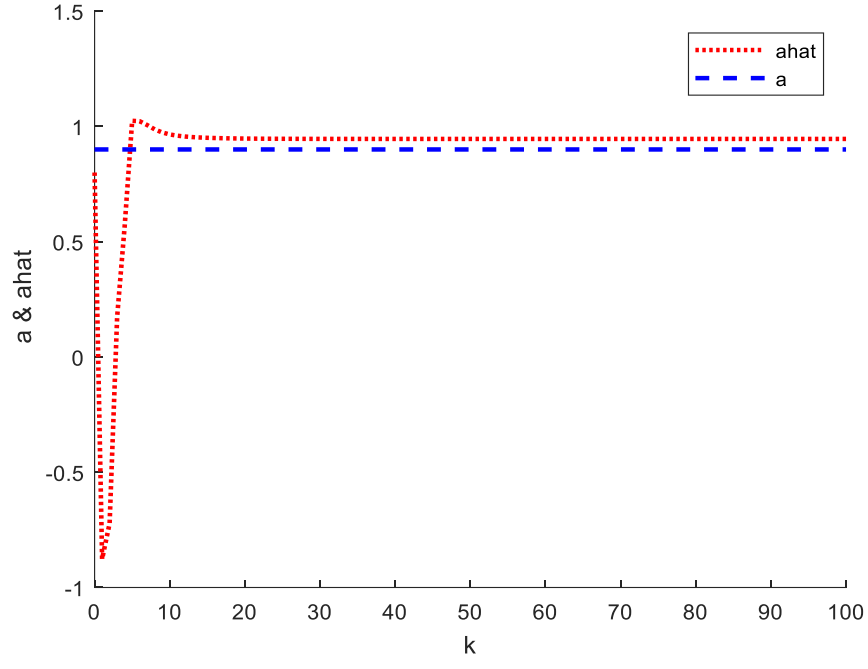

Figure 3.9 : The actual and estimated value of parameters via H-infinity filter

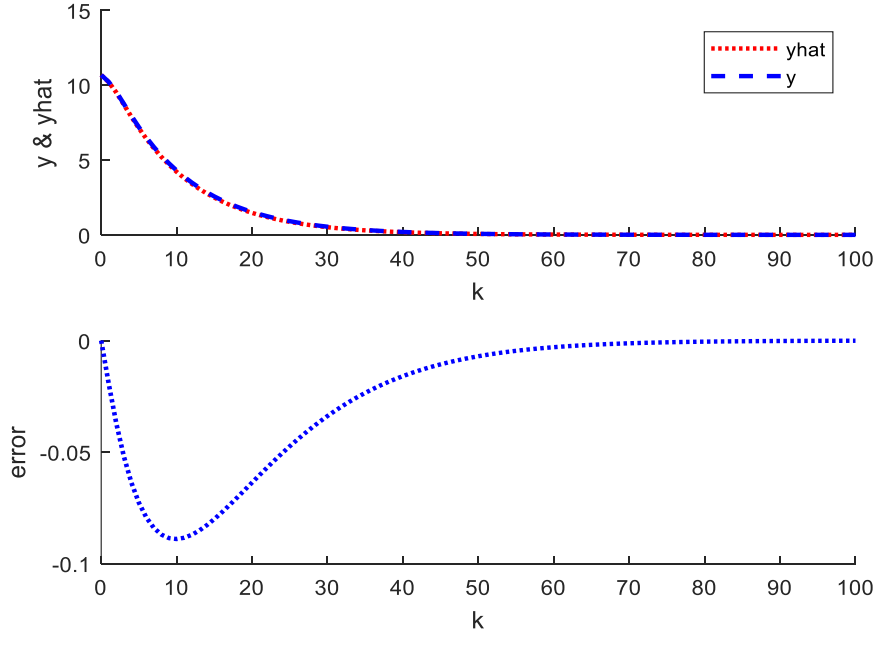

Figure 3.10: The actual and estimated values of measurement and the error via H-infinity filter

3.3 Estimating Parameters using a Bank of Kalman Filters

The objective of this section is to find an actual value of a parameter when its value is not known but its range of values is known. A bank of Kalman filters (which was discussed in Chapter 2) can be used to find an actual value of the unknown parameter. In this section, a bank of Kalman filters is used to find actual value of a parameter for two different cases. The first case is when there is only one parameter with five possibilities for its actual values. The second case is when there are two parameters; the actual value for each one have three possible values.

- Case 1:

Consider a first order system that is presented in equations (3.24).

$$x_{k=1} = a\, x_k + bu_k + v_k \tag{3.24a}$$

$$y_k = cx_k + w_k \tag{3.24b}$$

where $a = 0.9$, $b = 1$, $c = 1$, $v_k \sim N\ (0,0.1)$, $w_k \sim N\ (0,0.1)$

It is assumed that the possible values for a is one of these values

$$a_i = [0.7 \quad 0.8 \quad 0.9 \quad 1 \quad 1.1]$$

In addition, we assume that the input and output are available, and noise statistics are known. The number of Kalman filter in the bank can be determined using equation (2.42).

$$Number\ of\ KF = (5)^{1} = 5$$

Because there are five possible values for the parameter, we assume that the initial value of probabilities for all possible values are 20%. In addition, the initial value of the state and the error covariance for each Kalman filter are

$$\hat{x}_{i,k=0} = 6, P_{i,k=0} = 100$$

By setting up the Kalman filters, we can update the conditional probability for each possible value by equation (2.37); which is based on (2.38), (2.39), (2.40), and (2.41). From Figure 3.11, we conclude the actual value of the parameter is 0.9 because the conditional probability for 0.9 converges one and others converge to zero.

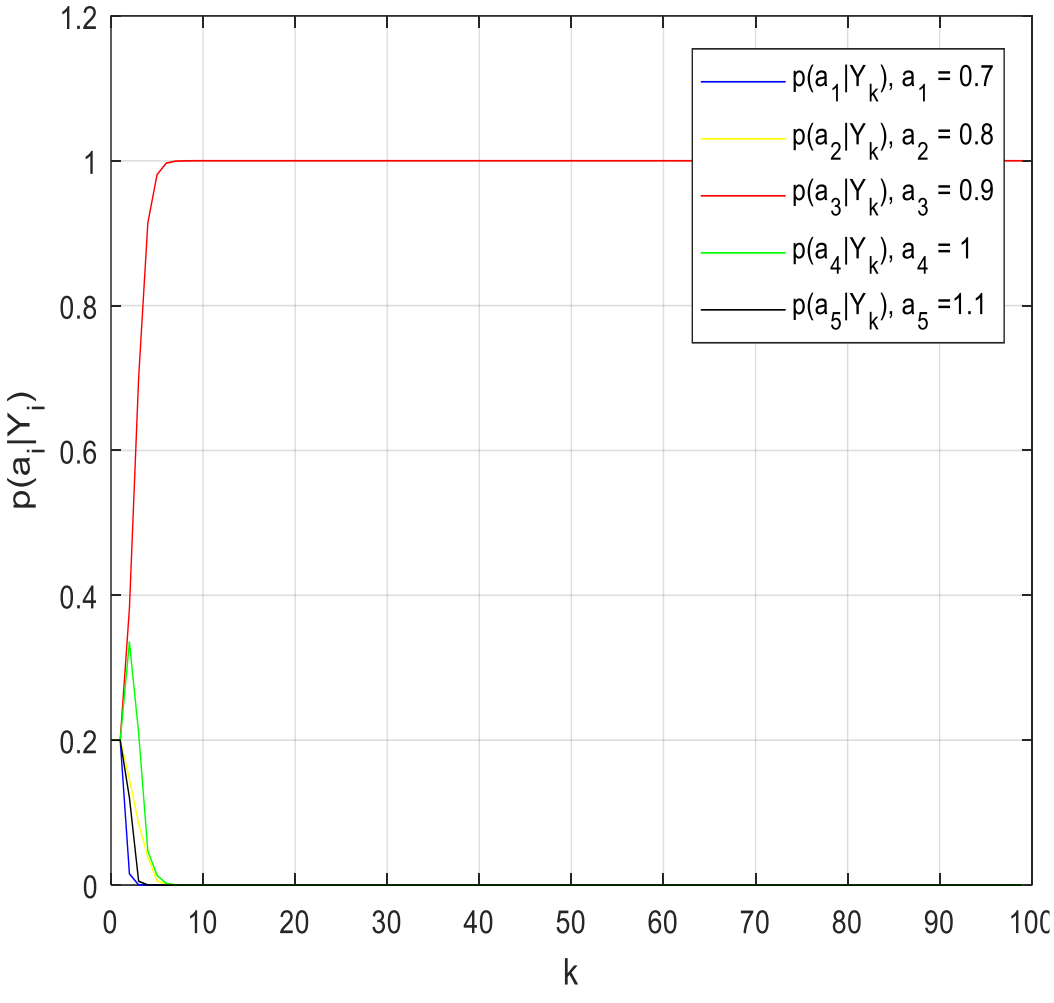

Figure 3.11: Probabilities for each Kalman filter for case 1

- Case 2:

Consider the same system that is shown in equations (3.38) and (3.39). In this case, we assume that both a and b each have three possible values:

$$a_i = [0.8 \quad 0.9 \quad 1] \ , b_i = [0.9 \quad 1 \quad 1.1]$$

From equation (2.42), we need nine Kalman filters. Table 3.1 shows us hypothesis value of parameters for each Kalman filter:

$$Number\ of\ K = (3)^2 = 9$$

Table 3.1: Hypotheses values for each Kalman filter

Hypotheses of a	Hypotheses of b	Number of K.F.
0.8	0.9	1
0.8	1	2
0.8	1.1	3
0.9	0.9	4
0.9	1	5
0.9	1.1	6
1	0.9	7
1	1	8
1	1.1	9

After setting up the bank of Kalman filters, we can update the conditional probability for hypothesis using equation (2.37) based on measurement data. By running the bank of Kalman filters algorithm, we have found that the conditional probability of fifth hypothesis ($a = 0.9, b = 1$) converges to one and others converge to zero. Figure 3.12 shows the conditional probability for each hypothesis.

In conclusion, based on the results obtained, a bank of Kalman filter can be used to find the actual value of parameter(s) that is unknown but with the range of possible values are known.

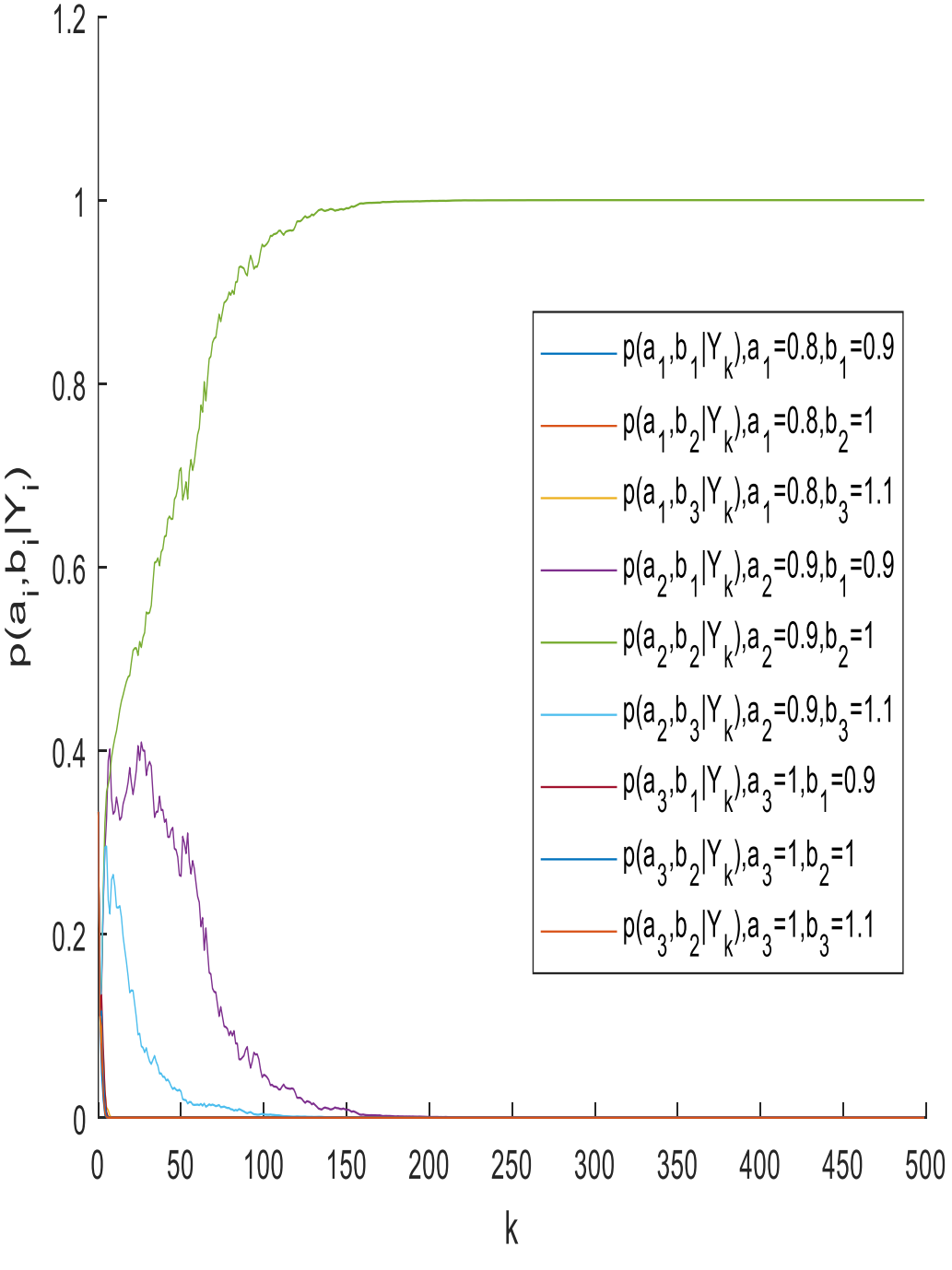

Figure 3.12: Probabilities for each Kalman filter for case 2

4 Case Studies in Intrusion Detection

The objective of this chapter is to detect an intrusion signal when attacks occur to an actuator or a sensor of a system. The techniques that will be used include sample mean, Kalman filter (bank of Kalman filters), and stochastic parameter estimation. We assume that the intrusion signal, which is used to attack a system is a step function that is given as

$$h = \begin{cases} 10\,, & k \geq 0 \\ 0\,, & otherwise \end{cases} \tag{4.1.a}$$

which is modelled in state space as

$$h_{k+1} = h_k \tag{4.1.b}$$

In the first section, the sample mean method is presented to detect the intrusion signal when attack happens to the sensor and actuator of the system. In the section section, Kalman filter method will be used to detect the intrusion signal when the attacks happen, specifically, by using a bank of Kalman filters. In the last section, we will present the stochastic parameter estimation method to detect the intrusion signal when the attack happens to the system. In every case, we assume that the state information in the measurement in the sensor or actuator are either partially or totalley replaced by a constant intrusion signal.

4.1 Sample Mean Method

In this section, we will show how to detect the intrusion signal using the sample mean method. The sample mean method is based on comparison between an actual and a theoretical sample mean values of a state or a measurement of a system to detect the intrusion signal. We will discuss two cases in this section. The first case is when the intrusion signal attacks the system through the sensor. Second case is when the intrusion signal attacks the system through the actuator.

- Case 1: Sensor is attacked

Consider a linear discrete time system equation

$$x_{k+1} = A\,x_k + Bu_k + v_k \tag{4.2a}$$

and the measurement

$$y_k = C\,x_k + w_k \tag{4.2b}$$

where $A = 0.85$, $B = 2$, $C = 3$, $x_0 = 3$, $u_k = \begin{cases} 1, & k \geq 0 \\ 0, & otherwise \end{cases}$

$$v_k \sim N(0,0.1)\ w_k \sim N(0,0.1)$$

However, the measurement will fluctuate around a constant value when the sensor is attacked and the measurement will be described by

$$y_k = C\,h + w_k \tag{4.3}$$

The theoretical sample mean of the system state ($\bar{x}_k$) that is presented in equation (4.2a) is calculated by [1].

$$\bar{x}_{k+1} = A\,\bar{x}_k + + Bu_k \tag{4.4}$$

By solving equation (4.4), we obtain [1].

$$\bar{x}_k = A^k\,\bar{x}_0 + \sum_{i=0}^{k-1} A^{k-i-1}\,Bu_k \tag{4.5}$$

From equation (4.5), the sample mean of the measurement is computed as

$$\bar{y}_k = C\left(A^k\,\bar{x}_0 + \sum_{i=0}^{k-1} A^{k-i-1}\,Bu_k\right) \tag{4.6}$$

After computing the sample mean of the measurement, we define a new variable that calculates the difference between the actual measurement and the theoretical sample mean, $\tilde{y}_k$ as

$$\tilde{y}_k = y_k - \bar{y}_k \tag{4.7}$$

When the sensor is not attacked, the difference between the actual and sample mean of measurement is around zero for a first order system if $|A| < 1$ and the input is a unit step function. The sample mean method algorithm when a sensor is attacked is presented in Figure 4.1.

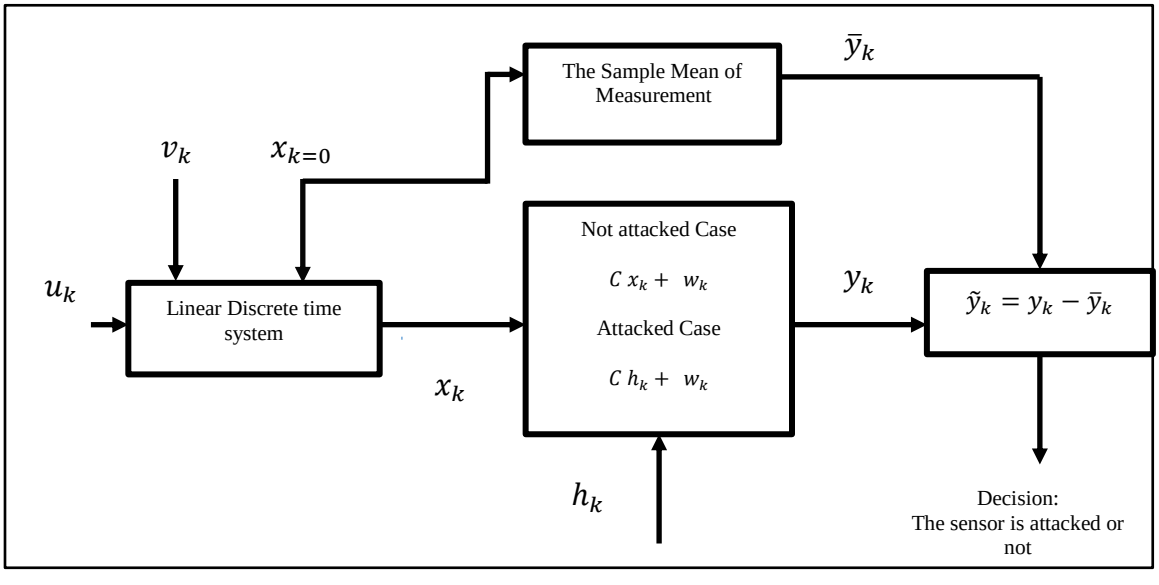

Figure 4.1: The sample mean method algorithm when the sensor is attacked

By setting up the system, which is presented in equation (4.2), and the sensor is attacked by the intrusion signal when $k = 100$. Figure 4.2 shows us the state and

measurement of the system. In addition, we show the difference between the theoretical sample mean and actual measurement in Figure 4.3.

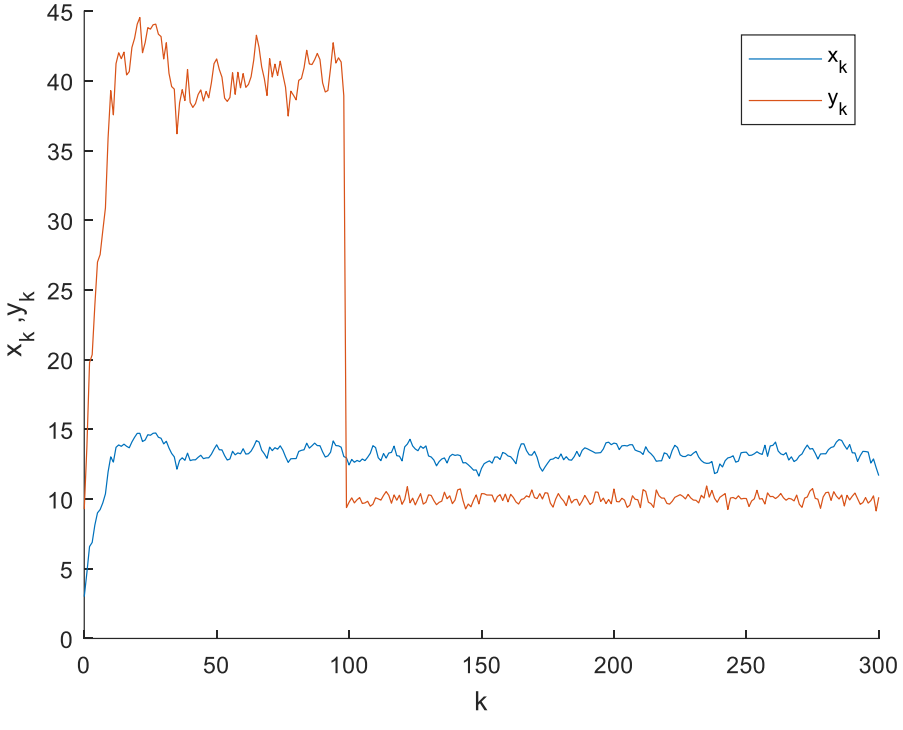

Figure 4.2: The state and measurement of the system when the sensor is attacked at k = 100

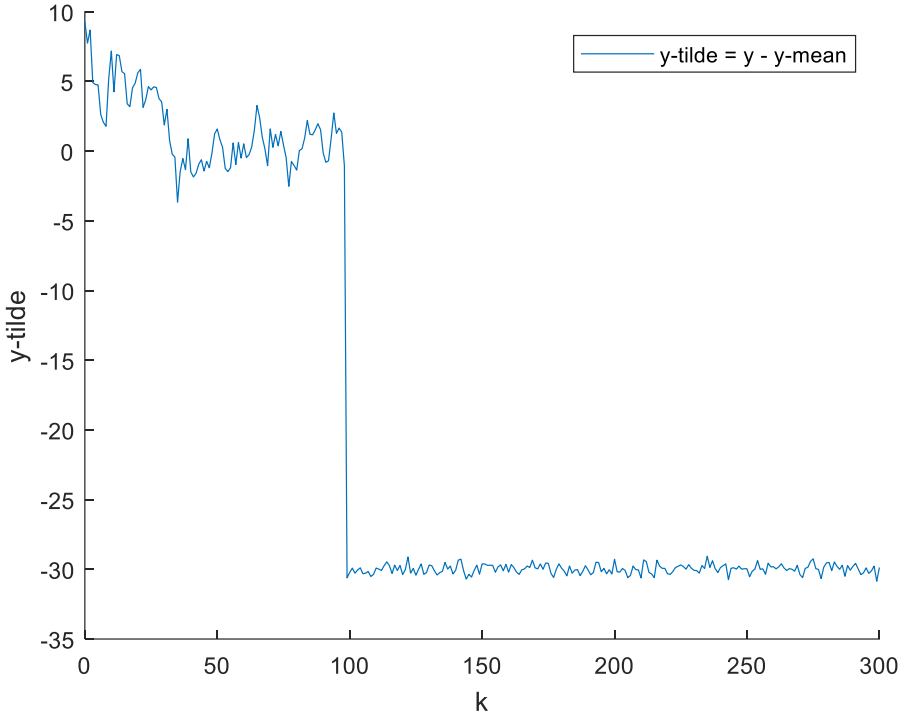

Figure 4.3: The difference between the actual and the theoretical sample mean measurement

From Figure 4.3, it is noticed that the difference between the actual and theoretical sample mean values of measurement is around zero before the attack happens. However, when the sensor is attacked at $k = 100$, difference between the actual and theoretical sample mean values goes to -30. Of course, the state value is not changed before and after $k = 100$. This gives us indication that the sensor is attacked via the intrusion signal.

- Case 2: Actuator is attacked

Consider a linear discrete time system below:

$$x_{k+1} = A\,x_k + B\,u_k + v_k \tag{4.8a}$$

$$y_k = C\,x_k + w_k \tag{4.8b}$$

where $A = 0.9$, $B = 1$, $C = 2$, $K = 1.1$, $x_0 = 10$, $u_k = -Kx_k$.

$$v_k \sim N\ (0,0.1)\ \ , w_k \sim N\ (0,0.1).$$

When the actuator is not attacked, the state of the system is

$$x_{k+1} = (A - BK)x_k + v_k \tag{4.9}$$

assuming that the state is measurable and is fed back to control the system.

The state equation of the system when the actuator is attacked is

$$x_{k+1} = Ax_k + Bh + v_k \tag{4.10}$$

The theoretical sample mean of the state when the actuator is not attacked is

$$\bar{x}_{k+1} = (A - BK)\bar{x}_k \tag{4.11}$$

By solving equation (4.11) the following equation can be obtained,

$$\bar{x}_k = (A - BK)^k\ \bar{x}_0$$

(4.12)

so, $\lim_{k\to\infty} \bar{x}_k = 0$ if the closed loop system is asymptotically stable.

On the other hand, when the actuator is attacked, the sample mean of the state is given by:

$$\bar{x}_{k+1} = A\bar{x}_k + Bh$$

(4.13)

By solving equation (4.13), the following equation can be obtained,

$$\bar{x}_k = A^k\, \bar{x}_{k=0} + \sum_{i=0}^{k-1} A^{k-i-1}\, Bh$$

(4.14)

By simplification of equation (4.14), the following equation can be obtained,

$$\sum_{i=0}^{k-1} A^{k-i-1} = A^{k-1} + A^{k-2} + A^{k-3} + \cdots + A^2 + A + I$$

$$\sum_{i=0}^{k-1} A^{k-i-1} = \left(A^{k-1} + A^{k-2} + A^{k-3} + \cdots + A^2 + A + I\right)(A - I)(A - I)^{-1}$$

$$\sum_{i=0}^{k-1} A^{k-i-1} = (A^k + A^{k-1} + A^{k-2} + \cdots + A^3 + A^2 + A - A^{k-1} + A^{k-2}$$

$$+ A^{k-3} + \cdots + A^2 + A + I)(A - I)^{-1}$$

$$\sum_{i=0}^{k-1} A^{k-i-1} = \left(A^k - I\right)(A - I)^{-1} \tag{4.15}$$

By substituting equation (4.15) into equation (4.14), we get

$$\bar{x}_k = A^k\, \bar{x}_0 + \left(A^k - I\right)(A - I)^{-1} Bh \tag{4.16}$$

and $\lim_{k\to\infty} \bar{x}_k = (I - A)^{-1} Bh$ for $|A| < 1$

From equation (4.12), when the actuator is not attacked, it can be conclude that the theoretical sample mean of the state goes to zero when k approaches to infinity which is shown in Figure 4.4. On the other hand, from equation (4.16), if the original system is asymptotically stable, the theoretical sample mean of the state goes to a constant value when the actuator is attacked as shown Figure 4.5. The theoretical sample mean of the state goes to infinity, when magnitude value of A is greater than one which is shown in Figure 4.6. Figure 4.7 shows the sample mean algorithm.

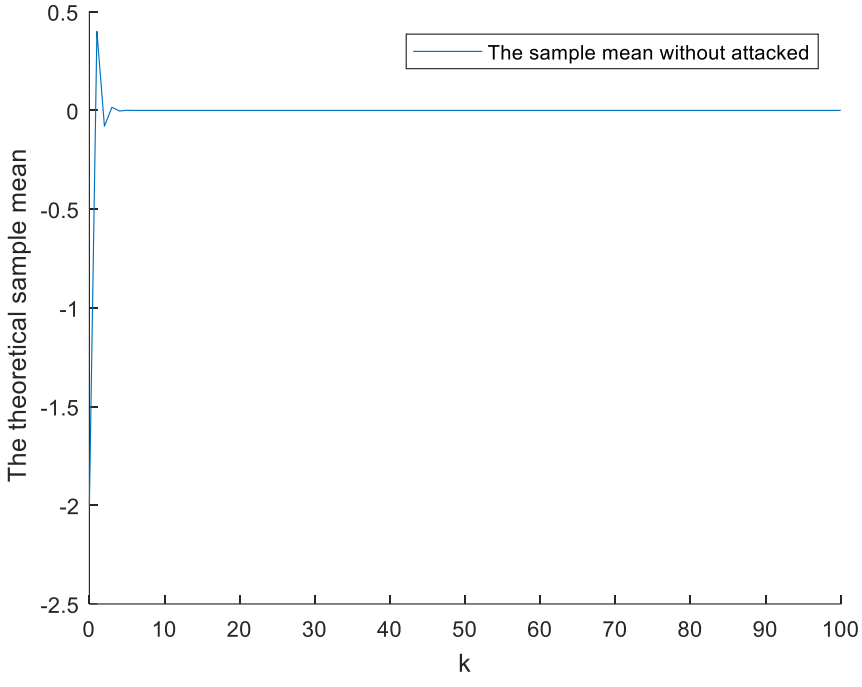

Figure 4.4: The theoretical sample mean of the state when the actuator is not attacked

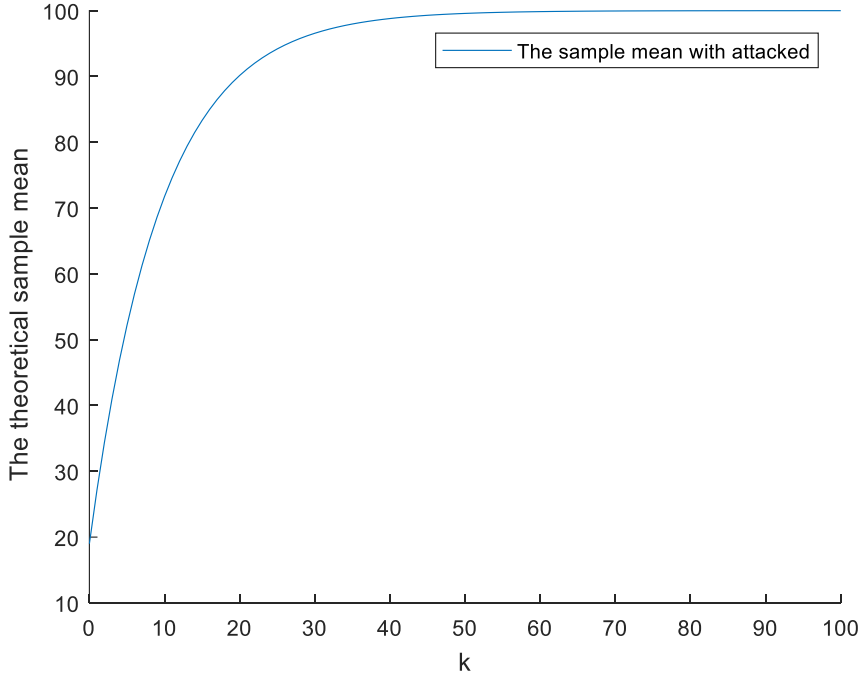

Figure 4.5: The theoretical sample mean of the state when the actuator is attacked, $|A_k| < 1$

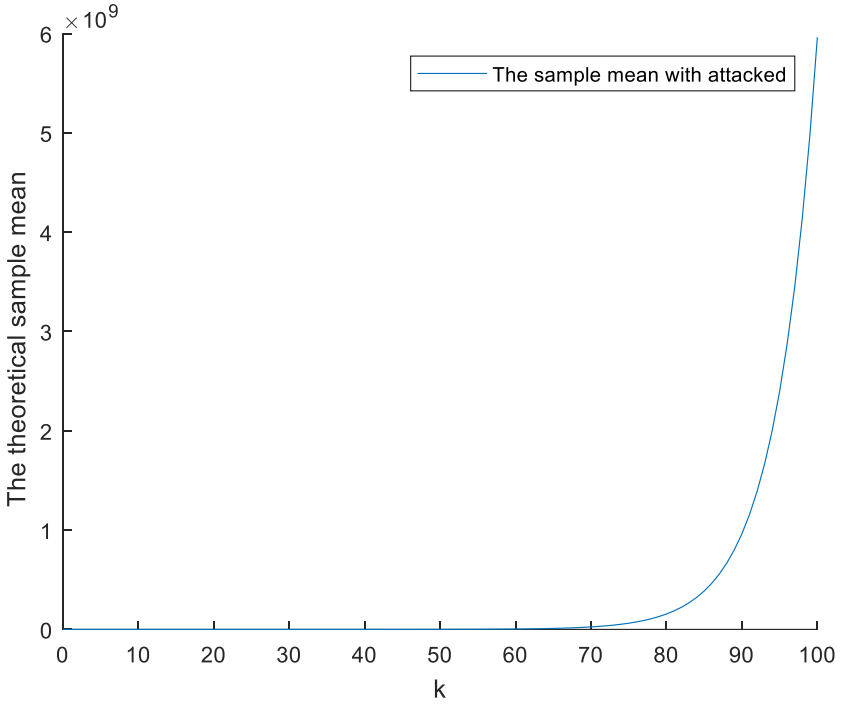

Figure 4.6: The theoretical sample mean of the state when the actuator is attacked, $|A_k| > 1$

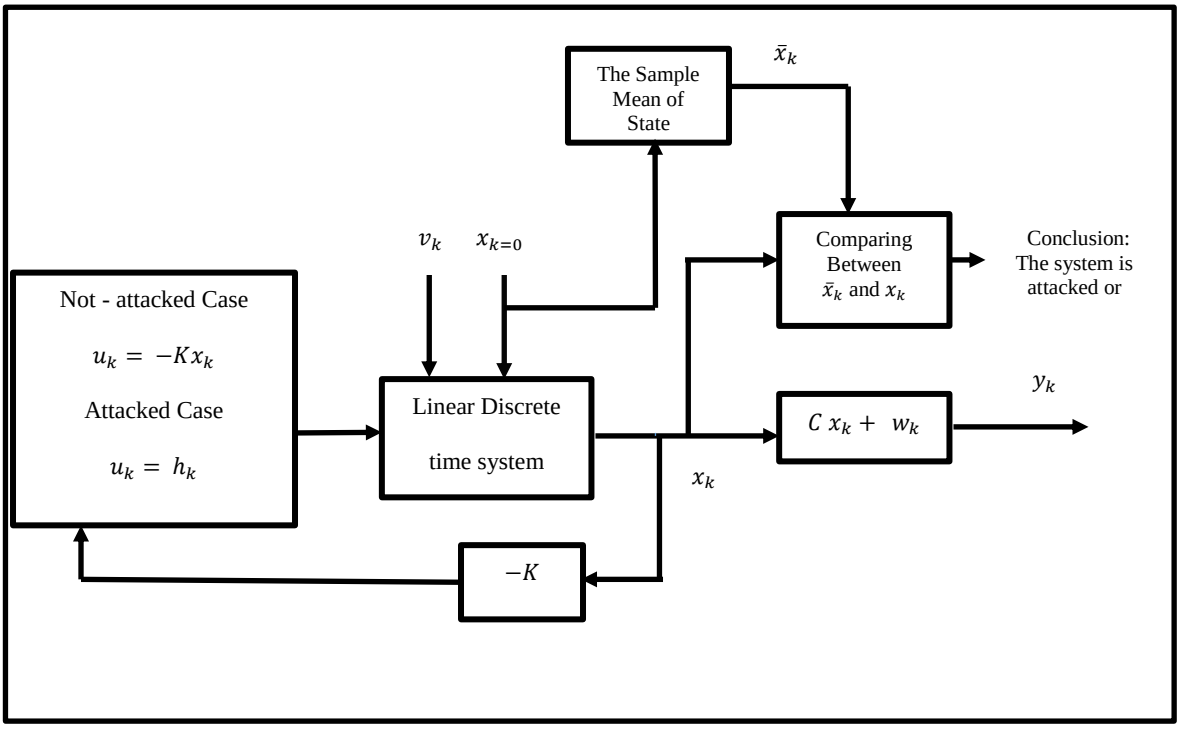

Figure 4.7: The sample mean algorithm when the actuator is attacked

By setting up the system which is presented in equations (4.9) and (4.10), when A= 0.9 , and the actuator is attacked via the intrusion signal when $k = 30$. Figure 4.8 shows us that before $k = 100$, the actual value of the state goes to zero. Then, it starts to increase until it reaches a constant value. It is noticed that the actuator is attacked via the intrusion signal.

In addition, we reset the system coefficient to $A = 1.2$, and the actuator is attacked via the intrusion signal when $k = 30$. From Figure 4.9, the value of the state approaches to infinity which indicates that the actuator is attacked.

In conclusion, based on the results obtained, the sample mean method is seen to be effective in detecting the intrusion signal when attacks occur to the system through either the sensor or the actuator.

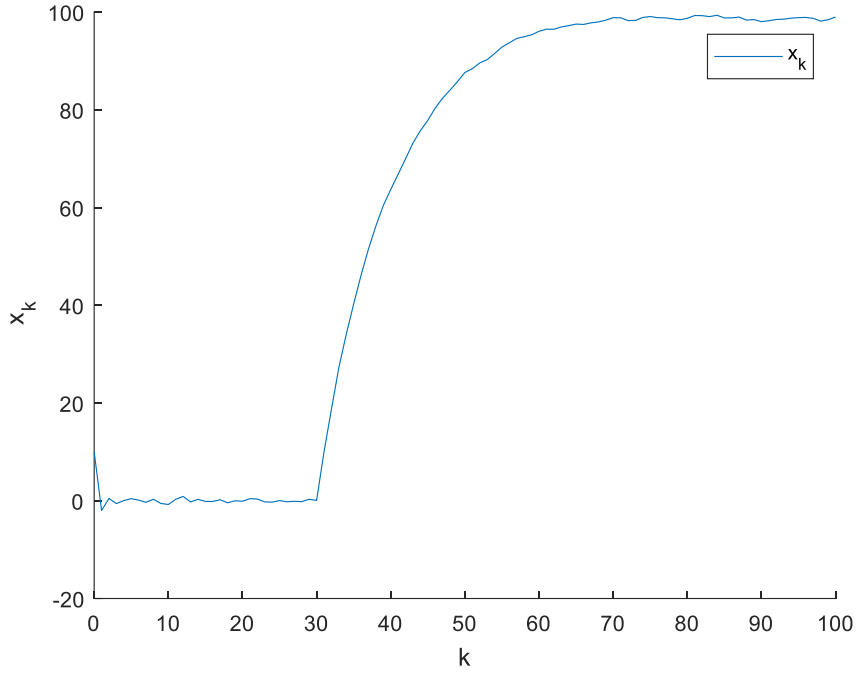

Figure 4.8: The state of the system when the actuator is attacked at k = 100, $|A_k| < 1$

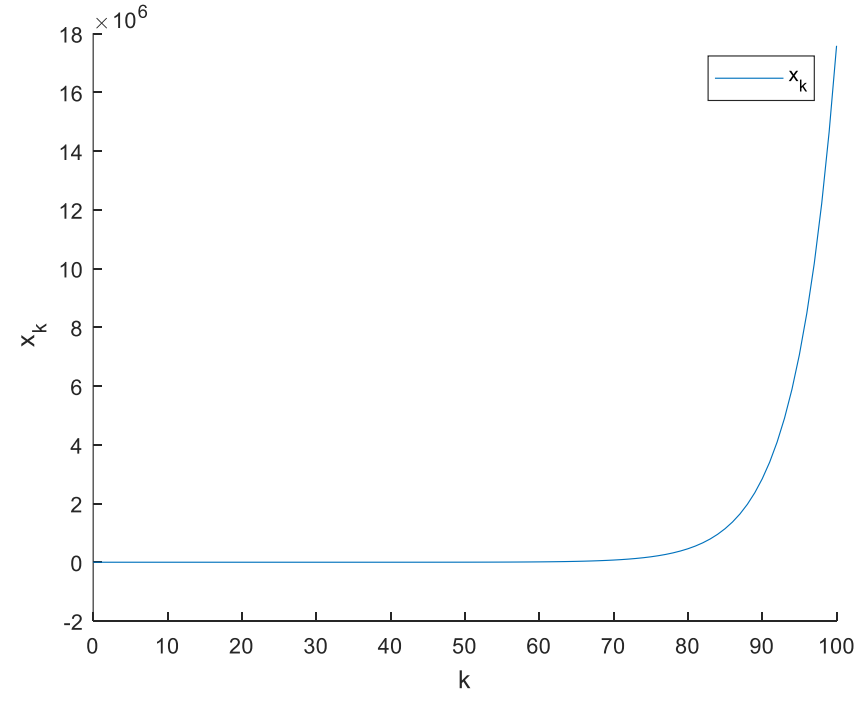

Figure 4.9 : The state of the system when the actuator is attacked at k = 100, $|A_k| > 1$

4.2 Kalman Filter Method

After we showed how to detect the intrusion signal via the sample mean method, we will now show how to detect the intrusion signal using a Kalman filter method in this section. In the Kalman filter method, a bank of Kalman filters is used to detect the intrusion signal. In this section, we will again present two cases when the sensor is attacked and when the actuator is attacked.

- Case 1: Sensor is attacked

Consider a linear discrete-time system that is presented in equation (4.17).

$$x_{k+1} = A\,x_k + F\,v_k \tag{4.17a}$$

$$y_k = C\,x_k + G\,w_k \tag{4.17b}$$

where $A = 0.9, B = 1, C = F = G = 1$, $v_k \sim N(0,0.1)$, $w_k \sim N(0,0.1)$. The equation (4.17b) is used to present the sensor when is not attacked; however, when the sensor is attacked, equation (4.18) is presented the measurement of system.

$$y_k = C\,h_k + Gw_k \tag{4.18}$$

In the re-modeling of the system including the attack, the state is presented by following:

$$\begin{bmatrix} x_{k+1} \\ h_{k+1} \end{bmatrix} = \begin{bmatrix} 0.9 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_k \\ h_k \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} v_k \tag{4.19a}$$

$$y_k = \begin{bmatrix} \varepsilon & 1 \end{bmatrix} \begin{bmatrix} x_k \\ h_k \end{bmatrix} + w_k \tag{4.19b}$$

where $A = \begin{bmatrix} 0.9 & 0 \\ 0 & 1 \end{bmatrix}, F = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$, $C = \begin{bmatrix} 1 & 0 \end{bmatrix}, G = 1$, $v_k \sim N\ (0,0.01)$, $w_k \sim N\ (0,0.01)$. $\varepsilon = 0.001$

By using a bank of Kalman filters, we compute the conditional probability for each state of the model when the measurement is obtained. Two Kalman filters are used in the bank. The first one ($K.F_1$) is used to estimate the state of model that is presented in equation (4.17). The second Kalman filter ($K.F_2$) is used to estimate the states of model that is presented in equation (4.19). After estimation of the states for each model, we compute the conditional probabilities by using equations (2.37)-(2.40) (Note: The equations were discussed in the section 2.1.3).

The initial values for $K.F_1$ were set as

$$\hat{x}_0 = 4,\ P_0 = 100$$

For K.F$_2$, initial values were set as

$$\begin{bmatrix}\hat{x}_0\\ \hat{h}_0\end{bmatrix} = \begin{bmatrix}4\\7\end{bmatrix},\ P_0 = \begin{bmatrix}100 & 0\\ 0 & 100\end{bmatrix}$$

Figure 4.10 shows the Kalman filter algorithm. When the sensor is not attacked, the conditional probability that is computed using estimated state value for the first model aproaches to 1 and the other the conditional probability aproaches to zero. On the other hand, the conditional probability that is computed by using estimated state value for second model aproaches to 1 when the sensor is attacked and the probability for the first filter aproaches to zero.

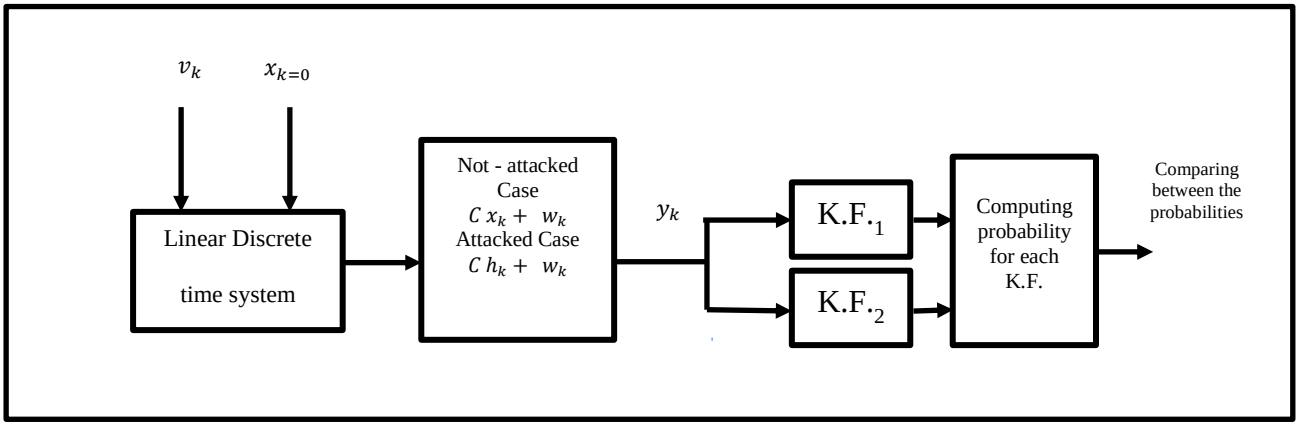

Figure 4.10: Kalman filter method algorithm when the sensor is attacked

Figure 4.11 shows us the conditional probability for a possible situation when the sensor is attacked via the intrusion signal at $k = 30$.

Based on these results, it can be concluded that the Kalman filter method has the ability to detect the intrusion signal when noise statistics are known.

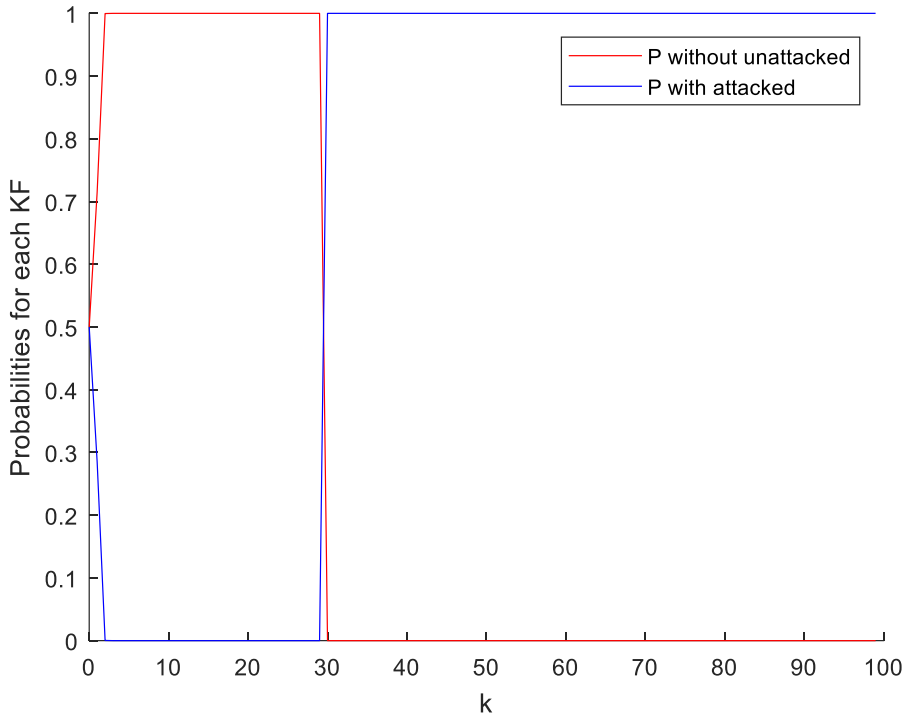

Figure 4.11: The probability of each Kalman filter when sensor attacked at k=30

- Case 2: Actuator is attacked

Consider a linear discrete time system that is presented below:

$$x_{k+1} = Ax_k + Bu_k + v_k \tag{4.20a}$$

$$y_k = Cx_k + w_k \tag{4.20b}$$

where the state is available for feedback $u_k = -Kx_k$, $K = 0.1, A = 0.85, B = C = F = G = 1$, $x_{k=0} = 1$, $v_k \sim N(0,0.1)$, $w_k \sim N(0,0.1)$.

By rewriting equation (4.20c),

$$x_{k+1}^1 = (A - KB)x_k^1 + v_k \tag{4.20c}$$

$$y_k = Cx_k^1 + w_k \tag{4.20d}$$

Equation (4.20) represents the actuator when it is not attacked. However, when the actuator is attacked, the state of the system is presented as below:

$$x_{k+1}^2 = Ax_k^2 + Bh + v_k \tag{4.21}$$

The equation below represents a model of system when the actuator is attacked:

$$\begin{bmatrix} x_{k+1}^2 \\ h_{k+1} \end{bmatrix} = \begin{bmatrix} 0.85 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_k^2 \\ h_k \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} v_k \tag{4.22a}$$

$$y_k = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_k^2 \\ h_k \end{bmatrix} + w_k \tag{4.22b}$$

where $A = \begin{bmatrix} 0.85 & 1 \\ 0 & 1 \end{bmatrix}, F = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$, $C = \begin{bmatrix} 1 & 0 \end{bmatrix}, G = 1$, $v_k \sim N(0,0.1), w_k \sim N(0,0.1)$.

The bank of Kalman filters in this case also needs two Kalman filters. The first one (K.F.$_1$) is used to estimate the state for the model that is presented in equation (4.19). Then, by using the estimated value of the state and error covariance, the conditional probability for not attacking situation is computed by (2.37)-(2.40).

However, the second Kalman filter is used to estimate the states for the model that is presented by equation (4.22). The conditional probability for the attack situation is computed using the estimated values of the states and error covariance, by equation (2.37)-(2.40).

The initial estimates of K.F$_1$ was set to

$$\hat{x}_0 = 1\ ,\ P_0 = 100$$

For K.F$_2$, the initial values were set to

$$\begin{bmatrix}\hat{x}_0\\ \hat{h}_0\end{bmatrix} = \begin{bmatrix}1\\1\end{bmatrix},\ P_0 = \begin{bmatrix}100 & 0\\ 0 & 100\end{bmatrix}$$

On the other hand, the control input is presented by equation when the actuator is either not attacked or attacked:

$$u_k = -K(x_k^1 p_1 + x_k^2 p_2) \tag{4.23a}$$

$$u_k = h \tag{4.23b}$$

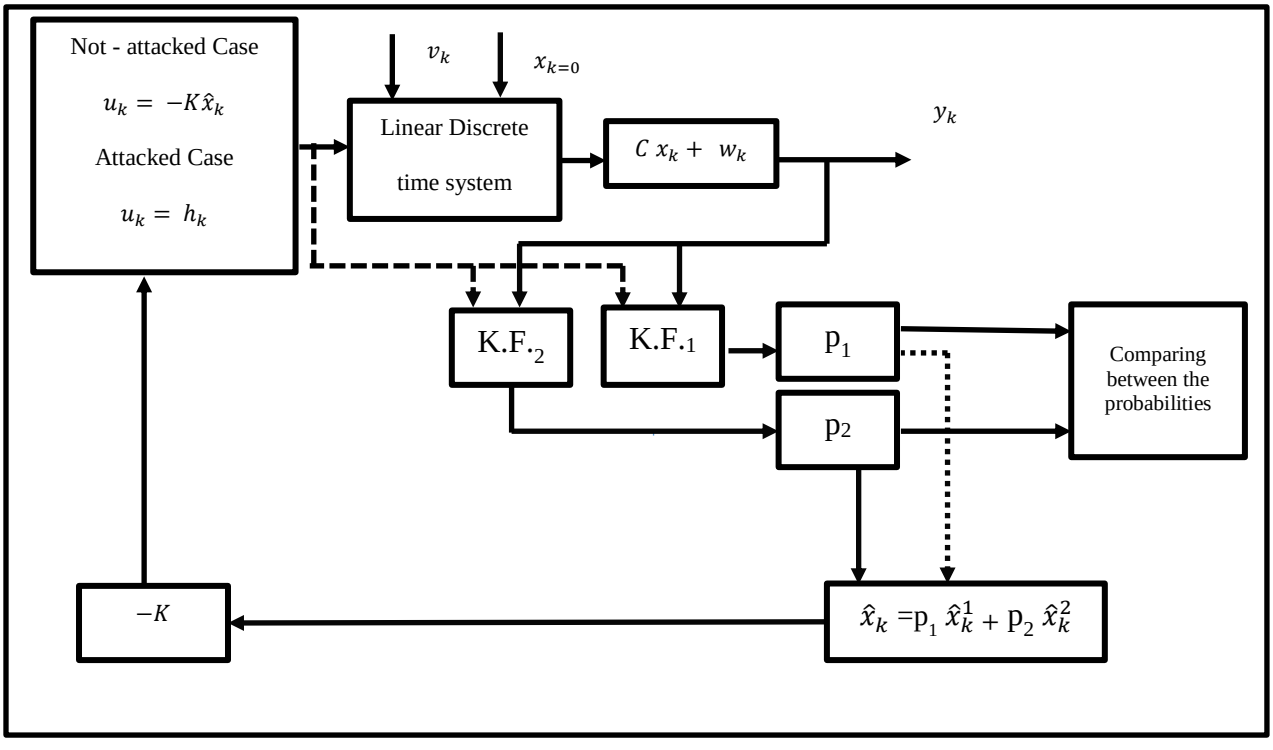

Figure 4.12: Kalman filter algorithm when the actuator is attacked

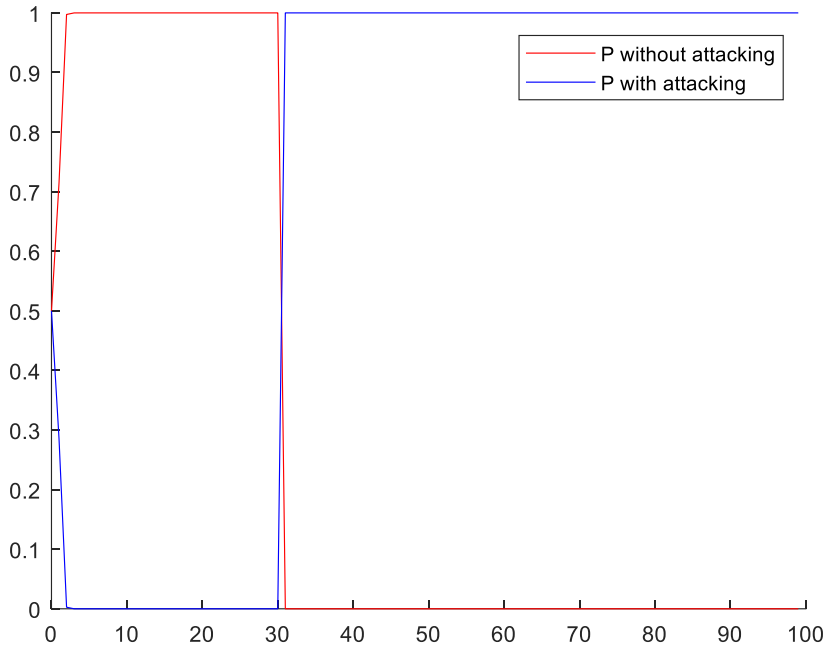

Figure 4.13: The probability of each Kalman filter when the actuator is attacked k=30

Figure 4.11 shows us Kalman filter algorithm when the actuator is attacked. We assume that the initial value of conditional probability for each situation is 50%. Figure 4.12 shows us the probability for each Kalman filter when the sensor is attacked via the intrusion signal at $k = 30$.

Therefore, based on the attainable result, Kalman filter method is viewed to be capable of detecting the attack on the actuator.

4.3 Stochastic Parameter Estimation Method

In this section, we will present stochastic parameter estimation method to detect the intrusion signal, which is presented in equation (4.1). The assumption of this method is that

there is a probability of attack on the sensor of a system, which is $(1-\beta)$. Consider the system and measurement equations below:

$$x_{k+1}^{1} = A\, x_k^1 + F v_k \tag{4.24a}$$

The measurement of the system when it is not attacked is:

$$y_k = C\, x_k^1 + G\, w_k \tag{4.24b}$$

Equation (4.25) represent the measurement of the system when it is attacked is:

$$y_k = Ch + G\, w_k \tag{4.25}$$

where $A = 0.9$, $B = C = F = G = 1, v_k \sim N\,(0,0.1)$, $w_k \sim N\,(0,0.1)$.

By remodeling the system, we have

$$\begin{bmatrix} x_{k+1}^1 \\ h_{k+1} \end{bmatrix} = \begin{bmatrix} A & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} x_k^1 \\ h_k \end{bmatrix} + \begin{bmatrix} v_k \\ 0 \end{bmatrix} \tag{4.26}$$

Equation (4.26) can be rewritten as,

$$x_{k+1} = \mathcal{A}\, x_k + + \begin{bmatrix} v_k \\ 0 \end{bmatrix}$$

The measurement of the system is

$$y_k = \mathbb{C}_k x_k + \ G_k w_k \tag{4.27}$$

where $\mathcal{A} = \begin{bmatrix} A & 0 \\ 0 & I \end{bmatrix}$, $x_k = \begin{bmatrix} x_k^1 \\ h_k \end{bmatrix}$. $\mathbb{C}_k$ is a random matrix and it is based on the probability for the sensor not attacked, β.

$$\mathbb{C}_k = [\, C_k \,, 0]$$

However, the probability for sensor attacked is $(1 - \beta)$.

$$\mathbb{C}_k = [\, 0 \,, I]$$

By computing the expected value and the second moment of $\mathbb{C}_k$, we can estimate the state of the system. The expected value of $\mathbb{C}_k$ is computed by equation

$$\bar{\mathbb{C}}_k = E\{\, \mathbb{C}_k\} = [\, \beta\ C_k \,, (1 - \beta)] \tag{4.28}$$

When the sensor is not attacked, $\tilde{\mathbb{C}}_k^1$ is

$$\tilde{\mathbb{C}}_k^1 = [\, C_k \,, 0\,] - \, E\{\, \mathbb{C}_k\} = [\, (1-\beta)C_k, (\beta-1)I]$$

(4.29)

However, when the sensor is attacked, $\tilde{\mathbb{C}}_k^2$ is

$$\tilde{\mathbb{C}}_k^2 = [\, 0 \,, I\,] - \, E\{\, \mathbb{C}_k\} = [\, -\beta\, C_k \,, \beta\, I\,]$$

(4.30)

The covariance of $\mathbb{C}_k$ is

$$E\,\{\mathbb{C}_k \mathrm{X}_k\, \mathbb{C}_k^T\} \;=\; \beta(\tilde{\mathbb{C}}_k^1 \mathrm{X}_k \tilde{\mathbb{C}}_k^{1\,T}) + \,(1-\beta)\,(\tilde{\mathbb{C}}_k^2 \mathrm{X}_k \tilde{\mathbb{C}}_k^{2\,T})$$

$$= \;(\beta\,[\,(1-\beta)C_k, (\beta-1)I]\mathrm{X}_k[\,(1-\beta)C_k, (\beta-1)I]^T\,)$$

$$+\,\big((1-\beta)[\,-\beta\, C_k \,, \beta\, I\,]\,\mathrm{X}_k\,[\,-\beta\, C_k \,, \beta\, I\,]^T\,\big)$$

(4.31)

where the covariance of the state in (4.26) is given by

$$\mathrm{X}_{k+1} = \;\mathcal{A}\,\mathrm{X}_k \mathcal{A}^T + \begin{bmatrix} V & 0 \\ 0 & 0 \end{bmatrix}$$

(4.32)

For this method, the equations for the Kalman filter are modified as follows:

The estimated state is:

$$\hat{x}_{k+1} = \;\mathcal{A}\,\hat{x}_k + K_k\,(y_k - \;\bar{\mathbb{C}}_k \hat{x}_k\,)$$

(4.33)

The gain of Kalman filter is:

$$K_k = \mathcal{A}\, P_k \bar{\mathbb{C}}_k^T \left(\bar{\mathbb{C}}_k P_k \bar{\mathbb{C}}_k^T + W + E\{\tilde{\mathbb{C}}_k \mathrm{X}_k \tilde{\mathbb{C}}_k^T\}\right)^{-1} \tag{4.34}$$

The error covariance is:

$$P_{k+1} = \mathcal{A}\, P_k \mathcal{A}^T - \mathcal{A}\, P_k \bar{\mathbb{C}}_k^T \left(\bar{\mathbb{C}}_k P_k \bar{\mathbb{C}}_k^T + W + E\{\tilde{\mathbb{C}}_k \mathrm{X}_k \tilde{\mathbb{C}}_k^T\}\right)^{-1} \bar{\mathbb{C}}_k^T P_k \mathcal{A} + \begin{bmatrix} V & 0 \\ 0 & 0 \end{bmatrix} \tag{4.35}$$

The initial values for K.F was set as follows: $\begin{bmatrix} \hat{x}_0 \\ \hat{h}_0 \end{bmatrix} = \begin{bmatrix} 3 \\ 6 \end{bmatrix}$, the initial error covariance, $P_0 = \begin{bmatrix} 100 & 0 \\ 0 & 100 \end{bmatrix}$, and state covariance, $\mathrm{X}_0 = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$. The probability for sensor not attacked is 90%, $\beta = 90\%$. Figure 4.13 shows the estimated values for the state of the system and the intrusion signal before and after the sensor is attacked at $k = 100$.

From Figure 4.14 shows that although the estimation quality of this technique is poor, it still indicates very quickly that is there is a change in the sensor signal, there by detection the intrusion.

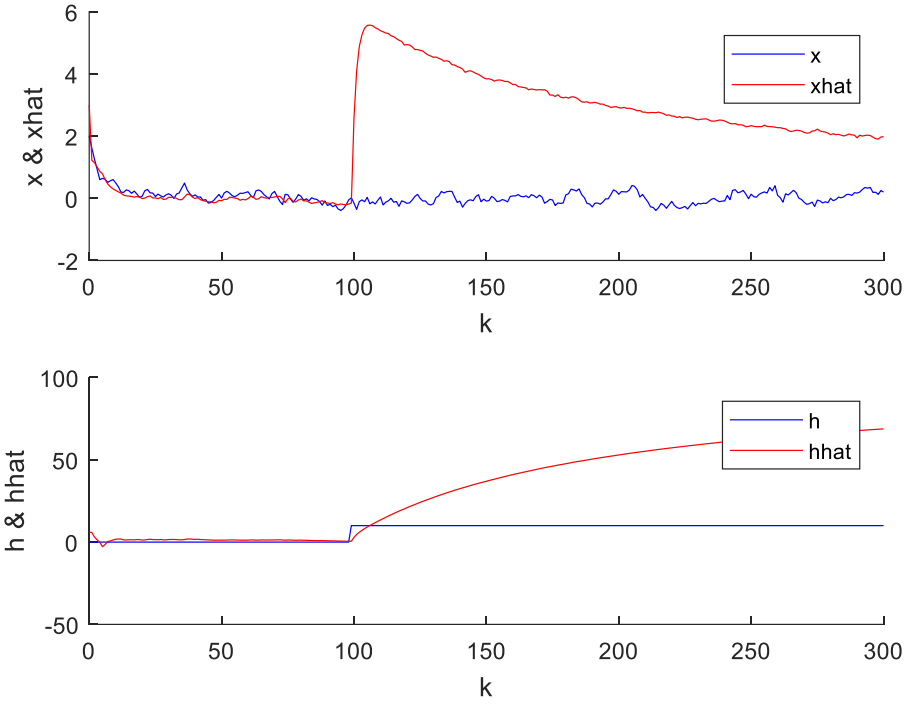

Figure 4.14: The actual and estimate values for the state and the intrusion signal when the sensor is attacked at $k = 100$

5 Conclusion and Future Work

For estimation of a state of the system, Kalman filter is the optimal filter to use if the model is linear, noises are additive and Gaussian with known first two moment. When the system is nonlinear, the system can be linearized about the current estimate and the Extended Kalman filter can be used to estimate the states. However, Extended Kalman filter is not an optimal filter.

For the case where the noise statistics are uncertain but the energy of noise is finite, H-infinity filters can be used. In this case, H-infinity may serve as the optimal filter to estimate the state of the linear system with minimization of the maximum estimation error. In addition, nonlinear H-infinity filter, which is discussed in section 2.2.2, can be used to estimate the state of nonlinear systems when the noise statistics are uncertain and the energy of noise is finite.

For parameter identification, Kalman and H-infinity filters are used in this thesis to estimate coefficients in the transfer functions after some manipulations. In addition, Extended Kalman and nonlinear H-infinity filters gave us good results when they were used to estimate the parameters of the state-space models. Moreover, when actual value of a parameter is unknown but its range is known, a bank of Kalman filters is used to estimate the parameter of the systems.

For detection of an intrusion signal that is of constant type acting on a sensor or actuator, both sample mean and Kalman filter methods are shown to be able to detect the

intrusion signals. Stochastic parameter estimation method could also detect the intrusion signal for the sensor intrusion case, although it resulted in large estimation error.

Future work would focus on generalizing these estimation applications to systems of higher order, with time-varying parameters, with certain types of nonlinear terms and when intrusion signals are of higher order polynomial or sinusoidal types.

BIBLIOGRAPHY

[1] D. Simon, Optimal state estimation: Kalman, H infinity, and nonlinear approaches. John Wiley & Sons, 2006.

[2] G. Bishop and G. J. P. o. S. Welch, Course, "An introduction to the Kalman filter," vol. 8, no. 27599-3175, p. 59, 2001.

[3] K. Sothivelr, "Analysis of Sensor Signals and Quantification of Analytes Based on Estimation Theory," Master Dissertation, Dept., Electrical and Computer Engineering, Marquette Univ., Milwaukee, WI, 2014

[4] D. Simon. (2001). From Here to Infinity. Embedded Systems Programming, 14(11), 20-32.

[5] A. R. Strandt, A. P. Strandt, S. C. Schneider and E. E. Yaz, "Stator Resistance Estimation Using Adaptive Estimation via a Bank of Kalman Filters," *2018 Annual American Control Conference (ACC)*, Milwaukee, WI, 2018, pp. 1078-1083.

[6] W. Xue, Y. Guo and X. Zhang, "A Bank of Kalman Filters and a Robust Kalman Filter Applied in Fault Diagnosis of Aircraft Engine Sensor/Actuator," *Second International Conference on Innovative Computing, Informatio and Control (ICICIC 2007)*, Kumamoto, 2007, pp. 10-10.

[7] E. Yaz, EECE 6340, Class Notes, "*Stochastic Systems Estimation and Control*," Electrical and Computer Engineering Department, Marquette University, Milwaukee, WI, Spring 2018.

[8] K. Kiriakidis, "H-infinity optimal filters for a class of nonlinear models," *Proceedings of the 2002 American Control Conference (IEEE Cat. No.CH37301)*, Anchorage, AK, USA, 2002, pp. 2336-2339 vol.3.doi: 10.1109/ACC.2002.1023989

[9] J. Zhao, "Dynamic State Estimation With Model Uncertainties Using H_∞ Extended Kalman Filter," in *IEEE Transactions on Power Systems*, vol. 33, no. 1, pp. 1099-1100, Jan. 2018.doi:10.1109/TPWRS.2017.2688131

[10] W. Bai, W. Xue, Y. Huang and H. Fang, "Extended state based H∞filter," *2016 35th Chinese Control Conference (CCC)*, Chengdu, 2016, pp. 292-297.doi: 10.1109/ChiCC.2016.7553098

[11] G. Gu, Discrete-Time Linear Systems: Theory and Design with Applications. Springer Science & Business *Media, 2012.*

[12] Chui, Charles K., and Guanrong Chen. Kalman filtering. Springer International Publishing, 2017.

[13] B. D. Anderson and J. B. J. E. C. Moore, "Optimal filtering," vol. 21, pp. 22-95, 1979.

[14] L. Li, Z. Wang and Y. Shen, "Fault diagnosis for attitude sensors via a bank of extended Kalman filters," *2016 35th Chinese Control Conference (CCC)*, Chengdu, 2016, pp. 6634-6638.doi: 10.1109/ChiCC.2016.7554400

[15] G. Rigatos, D. Serpanos and N. Zervos, "Detection of Attacks Against Power Grid Sensors Using Kalman Filter and Statistical Decision Making," in *IEEE Sensors Journal*, vol. 17, no. 23, pp. 7641-7648, 1 Dec.1, 2017.

[16] G. Rigatos, D. Serpanos, N. Zervos and P. Siano, "Kalman filtering and statistical decision making for detection of attacks against power grid sensors," *2017 IEEE 26th International Symposium on Industrial Electronics (ISIE)*, Edinburgh, 2017, pp. 1921-1926.

[17] J. Bonniwell, A. Alshareef, S. Schneider and E. Yaz " A Novel Extended H-infinity Filter" under preparation

Computer Code

```
% Chapter 2
% 2.1 Linear filter
% Kalman filter
% First order system, To estimate a state of a system
%.....................................................................
.....

% This code for estimation the state for first order system
% the system is x_k+1 = 0.9 x_k + u_k + v_k
%               y_k   = x_k + w_k
% v_k ~ N (0,V)  ,  w_k ~ N (0,W)

clear  ;
close all;
clc ;
kmax = 500; % the maximum value of time
N =25  ;   % number for iterations of program
error_n = NaN (1,N); %  %  error percent for each iteration

for n = 1 : 1 : N

    % To identify the parameters
    A = 0.9 ;
    B = 0.09;
    F = 1 ;
    C = 1 ;
    G = 1 ;
    V = 0.1 ;
    W = 0.1;
    v = 0+sqrt(V) * randn (1,100000);
    w = 0+sqrt(W) * randn (1,100000);
    x = NaN (1,kmax);
    y = NaN (1,kmax);
    x(1) = 10 ;
    u = 1 + zeros(1,kmax);
        xhat = NaN(1,kmax); % The estimation state
    Kk   = NaN(1,kmax); % The gain of Kalman Filter
    P    = NaN(1,kmax); % Covariance of estimation error
    error_y = NaN(1,kmax);
    xhat(1) = 12 ;
    P(1) = 100 ;

   for k = 1 : 1 : kmax

    % Simulation of the system

        if k < kmax
            x(k+1) = A*x(k) + B*u(k)+v(k) ;
        end
        y(k) = C*x(k)+G*w(k);


%.....................................................................
...%
```

```
    % Kalman Filter to estimat the state of the system

       Kk(k)= (A*P(k)*C')/(C*P(k)*C'+ G*W*G');
       if k < kmax
           xhat(k+1)= A*xhat(k)+B*u(k)+Kk(k)*(y(k)-C*xhat(k));
           P(k+1)   = A*P(k)*A'-(Kk(k)*C*P(k)*A')+F*V*F';
       end
       % Calculate the error
       yhat(k) = C*xhat(k);
       error_y(k) = (y(k) - yhat(k));
   end

   error_n(n) = ((error_y*error_y') /(y*y')) *100 ;
end

Net_error = (1/N) * (sum(error_n));

% Plot
k = 0 : 1 : kmax -1 ;
figure (1)
hold on
p1 = plot(k,x,'r:') ;
set(p1,'Linewidth',2);
p2 = plot(k,xhat,'b--');
set(p2,'Linewidth',2);
%title('The estimation the stat of first order system by Kalman
filter');
legend('x ','xhat ');
xlabel('k');
ylabel(' x(k) & xhat(k)');
hold off

figure (2)
hold on
p1 = plot(k,y,'r:') ;
set(p1,'Linewidth',2);
p2 = plot(k,yhat,'b--');
set(p2,'Linewidth',2);
%title('The estimation the stat of first order system by Kalman
filter');
legend('y  ','yhat ');
xlabel('k');
ylabel(' y(k), yhat(k)');
hold off

figure (3)
hold on
plot(k,error_y,'b') ;
%title('The estimation error for first order system by Kalman
filter');
legend('Estimation Error ') ;
xlabel('k');
ylabel('error of measurement');
hold off
```

```
% Chapter 2
% 2.1 Linear filter
% Kalman filter
% Second order system, To estimate  states of a system
%.....................................................................
.....

% This code for estimation the state for Second order system
% the system is x_k+1 = A x_k + B u_k + F v_k
%               y_k   = C x_k + G w_k
% v_k ~ N (0,V)  ,  w_k ~ N (0,W)

clear  ;
close all;
clc ;
kmax= 500; % the maximum value of time
N = 100 ;   % number for iterations of program
error_n = NaN(1,N); %   error percent for each iteration



for n = 1 : 1 : N

    % To identify the parameters

    summ = 0 ; %  the summation of ratio between squer of error by
squer of state x(K)
    A = [1,0.1;-1,0];
    B = [0.1;0];
    F = [1,0;0,1] ;
    C = [1,0] ;
    G = 1 ;
    V1 = 0.01 ;
    V2 = 0.01;
    V  = [V1,0;0,V2];
    W = 0.01;
    v1 = sqrt(V1) * randn (1,100000);
    v2 = sqrt(V2) * randn (1,100000);
    v = [v1 ; v2];
    w = sqrt(W) * randn (1,100000);
    x = NaN (2,kmax);
    y = NaN (1,kmax);
    x(:,1) = [10;1] ;
    u = 1 + zeros(1,kmax);

    % Simulation of the system

    for k = 1 : 1 : kmax
        if k < kmax
            x(:,k+1) = A*x(:,k) + B*u(k)+F*v(:,k) ;
        end
        y(k) = C*x(:,k)+G*w(k);
    end

    % Kalman Filter to estimat the state of the system

    xhat = NaN(2,kmax); % The estimation state
```

```
    Kk    = NaN(2,kmax); % The gain of Kalman Filter
    P     = NaN(2,2,kmax); % Covariance of estimation error
   error_y = NaN(1,kmax); % the error = y(k) -(C xhat(k))
   summ   = NaN(1,kmax);

   xhat(:,1) = [12,1.2] ;
   P(:,:,1)        = [100,0;0,100] ;

   for k = 1 : kmax

       Kk(:,k)= (A*P(:,:,k)*C')/(C*P(:,:,k)*C'+ G*W*G');
       if k < kmax
           xhat(:,k+1)= A*xhat(:,k)+B*u(k)+Kk(:,k)*(y(k)-C*xhat(:,k));
           P(:,:,k+1)= A*P(:,:,k)*A'-(Kk(:,k)*C*P(:,:,k)*A')+F*V*F';
       end
       % Calculate the error
       yhat(k) = C*xhat(:,k);
       error_y(k) = y(k)- yhat(k);

   end
  error_n(n) = ((error_y*error_y')/((y*y')))*100 ;
end

Net_error = (1/N) * (sum(error_n));


% Plot part
k = 0 : 1 : kmax -1 ;

figure (1)
hold on
p1 = plot(k,x(1,:),'r:') ;
set(p1,'Linewidth',2);
p2 = plot(k,xhat(1,:),'b--');
set(p2,'Linewidth',2);
%title('The estimation the state of x1 by Kalman filter');
legend('x1 ','xhat1');
xlabel('k');
ylabel(' x1(k) & xhat1(k)');
hold off

figure(2)
hold on
p1 = plot(k,x(2,:),'r:') ;
set(p1,'Linewidth',2);
p2 = plot(k,xhat(2,:),'b--');
set(p2,'Linewidth',2);
%title('The estimation the state of x2 by Kalman filter');
legend('x2 ','xhat2');
xlabel('k');
ylabel(' x2(k) & xhat2(k)');
hold off

figure (3)
hold on
p1 = plot(k,y,'r:') ;
set(p1,'Linewidth',2);
```

```
p2 =plot(k,yhat,'b--');
set(p2,'Linewidth',2);
%title('The estimation the stat of first order system by Kalman
filter');
legend('y  ','yhat ');
xlabel('k');
ylabel(' y(k), yhat(k)');
hold off

figure (4)
hold on
plot(k,error_y,'r') ;
%title('The estimation error for second order system by Kalman
filter');
legend('Estimation Error') ;
xlabel('k');
ylabel('error');
hold off
```

```
% Chapter 2
% 2.1 Linear filter
% H-infinity filter
% First order system, To estimate a state of a system
%.....................................................................
.....

% This code for estimation the state for first order system by using
% H-infinty filter
% the system is x_k+1 = 0.9 x_k + u_k +(5*exp(-k))
%               y_k   = x_k + (3 * exp(-2*k));
%
clear  ;
close all;
clc ;
kmax = 101; % the maximum value of time
N = 25 ;   % number for iterations of program
error_n = NaN ( 1, N) ;


for n = 1 : 1 : N

    % To identify the parameters

    A = 0.9 ;
    B = 0.09;
    F = 1 ;
    C = 1 ;
    G = 1 ;
    Cz = 1;
    gamma = 6;
    w1 = NaN (1,kmax); % the process noise
    w2 = NaN (1,kmax); % the measurement noise
    x = NaN (1,kmax); % the state of the system
    y = NaN (1,kmax);% the output of the system
    x(1) = 10 ; % initial value of state of the system
```

```
    u = 1 + zeros(1,kmax); % the input of the system
    xhat = NaN(1,kmax); % The estimation state
    K   = NaN(1,kmax); % The gain of of H-nfinty
    P    = NaN(1,kmax); %
    error_y = NaN(1,kmax);% Difference between y - ( C * xhat)
    squer_d= NaN(1,kmax) ;% Divide error square to y square
    xhat(1) = 12 ; % initial value of x estimation
    P(1) = 100 ; % initial value of P


    for k = 1 : 1 : kmax

        % Simulation of the system
%.....................................................................
....%
        w1(k)= 5*exp(-k);
        w2(k) = 3* exp(-k) ;
        if k < kmax
            x(k+1) = A*x(k) + B*u(k)+ F* w1(k);

        end
            y(k) = C*x(k)+G*w2(k);


%.....................................................................
...%

    % Kalman Filter to estimat the state of the system
%.....................................................................
....%



       K(k)= ((A*(P(k)^-1 - Cz'*Cz)^-1*C')+ gamma^-1*F*G')*(C*(P(k)^-1
- ...
           Cz'*Cz)*C'+gamma^-1*G*G');
       if k < kmax
           xhat(k+1)= A*xhat(k)+B*u(k)+K(k)*(y(k)-C*xhat(k));

           P(k+1)    = A*(P(k)^-1 - Cz'*Cz)*A'+gamma^-1*F*F'-(A*(P(k)^-
1-...
           Cz'*Cz)*C'+gamma^-1*F*G')*(C*(P(k)^-1 - Cz'*Cz)*A'+...
               gamma^-1*G*F')*(C*(P(k)^-1 - Cz'*Cz)*C'+gamma^-1*G*G');
       end

       yhat(k) = C*xhat(k);
       error_x(k) = x(k) - xhat(k) ;
       z(k) = Cz * error_x(k);
       e_p(k)= z(k) * z(k)';% energy of the performance output
       e_n(k) = (w1(k)*w1(k)')+(w2(k)*w2(k)') ; % energy of the noise
       error_y(k) = y(k) - ( C*xhat(k));


    end

    error_n(n) = ((error_y*error_y')) / ((y*y')) ;
```

```
    ration = sum(e_p)/sum(e_n);
end

Net_error = (1/N) * (sum(error_n))*100;


%.....
....%
% Plot
k = 0 : 1 : kmax -1 ;
figure (1)
hold on
p1 = plot(k,x,'r:') ;
set(p1,'Linewidth',2);
p2 = plot(k,xhat,'b--');
set(p2,'Linewidth',2);
%title('The estimation the stat of first order system by Kalman
filter');
legend('x ','xhat ');
xlabel('k');
ylabel(' x(k) & xhat(k)');
hold off

figure (2)
hold on
p1 = plot(k,y,'r:') ;
set(p1,'Linewidth',2);
p2 = plot(k,yhat,'b--');
set(p2,'Linewidth',2);
%title('The estimation the stat of first order system by Kalman
filter');
legend('y  ','yhat ');
xlabel('k');
ylabel(' y(k), yhat(k)');
hold off

figure (3)
hold on
plot(k,error_y,'b') ;
%title('The estimation error for first order system by Kalman
filter');
legend('Estimation Error ') ;
xlabel('k');
ylabel('error of measurement');
hold off
```

```
% Chapter 2
% 2.2 Linear filter
% Exteneded Kalman filter
% First order system, To estimate a state of a system
%.....
.....
% Extended Kalman Filter to estimate the state of x(k+1) = - sin(k) +
v(k)
clear ;
close all ;
clc ;
N = 25;
```

```
for n = 1 : 1 : N
%.................................................%
kmax = 101 ;
V = 0.01 ;
W = 0.01;
v = 0+sqrt(V) * randn (1,100000);
w = 0+sqrt(W) * randn (1,100000);
x = NaN(1,kmax);
x(1) = 0.9 ;
y = NaN(1,kmax);
xhat = NaN(1,kmax);
xhat(1) = 1.1 ;
P = NaN(1,kmax);
P(1) = 1000 ;
K = NaN(1,kmax);
%.................................................%

for k = 1 : 1 :kmax

    % Simulation of the system
    if k < kmax

        x(k+1) = - sin(x(k)) + v(k) ;
    end
     y(k) =  x(k) + w(k);

    % Extended Kalman Filter to estimat the state of the system
    A = -cos(xhat(k));
    C = 1 ;
    F = 1 ;
    G = 1 ;

    K(k)= (A*P(k)*C')/(C*P(k)*C'+ G*W*G');
    if k < kmax
    xhat(k+1) = - sin(xhat(k)) + K(k) * ( y(k) - xhat(k));
    P(k+1)   = A*P(k)*A'-(K(k)*C*P(k)*A')+F*V*F';
    end
    error_y(k)  = y(k)  -  xhat(k);

end
     norm_error(k) = ((error_y*error_y') /(y*y') ) * 100;
end

net_norm_error = (sum(norm_error))/N ;

t = 0 : 1 : kmax-1   ;

figure (1)
subplot(3,1,1)
hold on
plot(t,x,'b');
legend('x');
xlabel('k');
ylabel('x');
subplot(3,1,2)
hold on
```

```
plot(t,xhat,'r');
legend('xhat');
xlabel('k');
ylabel('xhat');
subplot(3,1,3)
hold on
plot(t,x,'b');
plot(t,xhat,'r');
legend('x','xhat');
xlabel('k');
ylabel('x and xhat');
hold off

figure (2)
hold on
plot(t,error_y,'b');
legend('error');
xlabel('k');
ylabel('error');
hold off
```

```
% Chapter 2
% 2.2 Linear filter
% Exteneded Kalman filter
% First order system, To estimate a state of a system
%.....................................................................
.....
% Extended Kalman Filter to estimate the state of x(k+1) = - x^2 +
v(k)
clear ;
close all ;
clc ;
N = 25;

for n = 1 : 1 : N
%.................................................%
kmax = 101 ;
V = 0.01 ;
W = 0.01;
v = 0+sqrt(V) * randn (1,100000);
w = 0+sqrt(W) * randn (1,100000);
x = NaN(1,kmax);
x(1) = 0.9 ;
y = NaN(1,kmax);
xhat = NaN(1,kmax);
xhat(1) = 1.1 ;
P = NaN(1,kmax);
P(1) = 1000 ;
K = NaN(1,kmax);
%.................................................%

for k = 1 : 1 :kmax

    % Simulation of the system
    if k < kmax

        x(k+1) = - x(k)^2 + v(k) ;
    end
     y(k) =  x(k) + w(k);
```

```
    % Extended Kalman Filter to estimat the state of the system
    A = -2*xhat(k);
    C = 1 ;
    F = 1 ;
    G = 1 ;

    K(k)= (A*P(k)*C')/(C*P(k)*C'+ G*W*G');
    if k < kmax
    xhat(k+1) =  - xhat(k)^2  + K(k) * ( y(k) - xhat(k));
    P(k+1)   = A*P(k)*A'-(K(k)*C*P(k)*A')+F*V*F';
    end
    error_y(k)  = y(k)  -  xhat(k);

end
     norm_error(k) = ((error_y*error_y') /(y*y')) * 100;
end

net_norm_error = (sum(norm_error))/N ;

t = 0 : 1 : kmax-1   ;

figure (1)
subplot(3,1,1)
hold on
p1 = plot(t,x,'b:');
set(p1,'Linewidth',2);
legend('x');
xlabel('k');
ylabel('x');
subplot(3,1,2)
hold on
p1 =plot(t,xhat,'r--');
set(p1,'Linewidth',2);
legend('xhat');
xlabel('k');
ylabel('xhat');
subplot(3,1,3)
hold on
p1 = plot(t,x,'b:');
set(p1,'Linewidth',2);
p2= plot(t,xhat,'r--');
set(p2,'Linewidth',2);
legend('x','xhat');
xlabel('k');
ylabel('x and xhat');
hold off

figure (2)
hold on
plot(t,error_y,'b');
legend('error');
xlabel('k');
ylabel('error');
hold off
```

```
% Chapter 2
```

```
% 2.2 Linear filter
% Exteneded Kalman filter
% First order system, To estimate a state of a system
%.....................................................................
.....

% Extended Kalman Filter to estimate the state of x(k+1) = - x^2 +
v(k)
clear ;
close all ;
clc ;
N = 25;

for n = 1 : 1 : N
%.................................................%
kmax = 101 ;
V = 0.01 ;
W = 0.01;
v = 0+sqrt(V) * randn (1,100000);
w = 0+sqrt(W) * randn (1,100000);
x = NaN(1,kmax);
x(1) = 0.9 ;
y = NaN(1,kmax);
xhat = NaN(1,kmax);
xhat(1) = 1.1 ;
P = NaN(1,kmax);
P(1) = 1000 ;
K = NaN(1,kmax);
%.................................................%

for k = 1 : 1 :kmax

    % Simulation of the system
    if k < kmax

        x(k+1) = - x(k)^3 + v(k) ;
    end
     y(k) =  x(k) + w(k);

    % Extended Kalman Filter to estimat the state of the system
    A = -3*xhat(k)^2;
    C = 1 ;
    F = 1 ;
    G = 1 ;

    K(k)= (A*P(k)*C')/(C*P(k)*C'+ G*W*G');
    if k < kmax
    xhat(k+1) =  - xhat(k)^3  + K(k) * ( y(k) - xhat(k));
    P(k+1)   = A*P(k)*A'-(K(k)*C*P(k)*A')+F*V*F';
    end
    error_y(k)  = y(k)  -  xhat(k);

end
     norm_error(k) = ((error_y*error_y') /(y*y') ) * 100;
end

net_norm_error = (sum(norm_error))/N ;
```

```
t = 0 : 1 : kmax-1   ;

figure (1)
subplot(3,1,1)
hold on
p1 = plot(t,x,'b:');
set(p1,'Linewidth',2);
legend('x');
xlabel('k');
ylabel('x');
subplot(3,1,2)
hold on
p1 =plot(t,xhat,'r--');
set(p1,'Linewidth',2);
legend('xhat');
xlabel('k');
ylabel('xhat');
subplot(3,1,3)
hold on
p1 = plot(t,x,'b:');
set(p1,'Linewidth',2);
p2= plot(t,xhat,'r--');
set(p2,'Linewidth',2);
legend('x','xhat');
xlabel('k');
ylabel('x and xhat');
hold off

figure (2)
hold on
plot(t,error_y,'b');
legend('error');
xlabel('k');
ylabel('error');
hold off
```

```
% Chapter 2
% 2.2 Linear filter
% H-infinity filter
% First order system, To estimate a state of a nonlier system
% Code for thesis of a master degree of Electrical and Computer
Engineering
%.....
.....

% This code for estimation the state for nonlinear system by using
% H-infinty filter
% the system is x(k+1) = - sin(x(k)) + [1,0] * [exp(-k); -3 * exp(-
2*k)]
%               y(k)   = x(k) +[0,1]*[exp(-k); -3 * exp(-2*k)]
%
clear ;
close all ;
clc ;

%................................................%
N = 25;
```

```
error_n = NaN (1,N);

for n=1:1:N

    kmax = 100 ;
    F = [1,0];
    G = [0,1];
    Cz = 0.5 ;
    C  = 1 ;
    g = 2 ;
    x = NaN(1,kmax);
    x(1) = 1 ;
    y = NaN(1,kmax);
    w = NaN(1,kmax);
    z = NaN(1,kmax);
    xhat = NaN(1,kmax);
    xhat(1) = 1.2 ;
    P = NaN(1,kmax);
    P(1) = 100 ;
    K = NaN(1,kmax);
    error_y = NaN(1,kmax);
    squer_d = NaN(1,kmax) ;
    for k = 1 : 1 :kmax
         % Simulation of the system
%.....
....%
        if k < kmax
        x(k+1) = - sin(x(k)) + F * [exp(-k); -3 * exp(-2*k)] ;
        end
        y(k) = C * x(k) + G * [exp(-k); -3 * exp(-2*k)];
%.....
....%

       % H-infinity Filter to estimat the state of the system
%.....
....%
        A = - cos(xhat(k));
        C = 1 ;
        M = ( P(k)^-1 - Cz'*Cz) ;
        K(k) = (A*M*C'+ g^-1*F*G')/(C*M*C'+g^-1*G*G');
        if k < kmax
            xhat(k+1) = -sin(xhat(k)) + K(k) * ( y(k) - C*xhat(k));
            P(k+1) = (A*M*A'+g^-1*F*F')-(A*M*C'+g^-1*F*G')*(C*M*C'+g^-
1*...
                G*G')^-1*(C*M*A'+g^-1*G*F');
        end
        z(k) = (Cz * ( x(k)-xhat(k)));
        w(k) = exp(-2*k)+0.25*exp(-4*k);
        error_y(k) = y(k) - (C*xhat(k));
        squer_d(k) = (error_y(k))^2 / y(k)^2 ;
    end
     error_n(n) = (1/kmax) * ( sum(squer_d)) ;
end

Net_error = (1/N) * ( sum(error_n)) * 100 ;

Z = z.*z ;
a = sum (Z) /sum( w );
```

```
t = 0 : 1 : 100-1   ;



t = 0 : 1 : kmax-1   ;

figure (1)
subplot(3,1,1)
hold on
p1 = plot(t,x,'b');
set(p1,'Linewidth',2);
legend('x');
xlabel('k');
ylabel('x');
subplot(3,1,2)
hold on
p1 =plot(t,xhat,'r');
set(p1,'Linewidth',2);
legend('xhat');
xlabel('k');
ylabel('xhat');
subplot(3,1,3)
hold on
p1 = plot(t,x,'b');
set(p1,'Linewidth',2);
p2= plot(t,xhat,'r');
set(p2,'Linewidth',2);
legend('x','xhat');
xlabel('k');
ylabel('x and xhat');
hold off

figure (2)
hold on
plot(t,error_y,'b');
legend('error');
xlabel('k');
ylabel('error');
axis([0,15,-1.5 , 2])
hold off
```

```
% Chapter 2
% 2.2 Linear filter
% H-infinity filter
% First order system, To estimate a state of a nonlier system
% Code for thesis of a master degree of Electrical and Computer
Engineering
%.....
.....

% This code for estimation the state for nonlinear system by using
% H-infinty filter
% the system is x(k+1) = - x(k)^2 + [1,0] * [0.3 exp(-k); -0.2 * exp(-
2*k)]
%               y(k)   = x(k) +[0,1]*[exp(-k); -3 * exp(-2*k)]
%
clear ;
```

```
close all ;
clc ;

%.....................................................%
kmax = 100 ;

F = [1,0];

G = [0,1];

Cz = 0.5 ;
c  = 1 ;
g = 2 ;
x = NaN(1,kmax);
x(1) = 0.5 ;
y = NaN(1,kmax);
w = NaN(1,kmax);
z = NaN(1,kmax);

%.....................................................%

for k = 1 : 1 :kmax
    if k < kmax
        x(k+1) = - x(k)^2 + F * [0.3*exp(-k); -0.2 * exp(-2*k)] ;
    end
     y(k) = C * x(k) + G * [0.3*exp(-k); -0.2 * exp(-2*k)];
end
%.....................................................%

xhat = NaN(1,kmax);

xhat(1) = 0.4 ;

P = NaN(1,kmax);

P(1) = 100 ;

K = NaN(1,kmax);

for k =1 : 1 : kmax
    A = -2*xhat(k);
    C = 1 ;
    M = ( P(k)^-1 - Cz'*Cz) ;
    K(k) = (A*M*C'+ g^-1*F*G')/(C*M*C'+g^-1*G*G');
    if k < kmax

        xhat(k+1) = -xhat(k)^2 + K(k) * ( y(k) - C*xhat(k));
        P(k+1) = (A*M*A'+g^-1*F*F')-(A*M*C'+g^-1*F*G')*(C*M*C'+g^-
1*G*G')^-1*(C*M*A'+g^-1*G*F');
    end

    error_y(k)  = y(k) - xhat(k) ;
    z(k) = (Cz * ( x(k)-xhat(k)));

    w(k) = exp(-2*k)+0.25*exp(-4*k);
end
```

```
Net_error = ((error_y*error_y')/(y*y'))* 100 ;
Z = z.*z ;
W = w.*w;
r = sum (Z) /sum(w );

t = 0 : 1 : kmax-1   ;

figure (1)
subplot(3,1,1)
hold on
p1 = plot(t,x,'b:');
set(p1,'Linewidth',2);
legend('x');
xlabel('k');
ylabel('x');
subplot(3,1,2)
hold on
p1 =plot(t,xhat,'r--');
set(p1,'Linewidth',2);
legend('xhat');
xlabel('k');
ylabel('xhat');
subplot(3,1,3)
hold on
p1 = plot(t,x,'b:');
set(p1,'Linewidth',2);
p2= plot(t,xhat,'r--');
set(p2,'Linewidth',2);
legend('x','xhat');
xlabel('k');
ylabel('x and xhat');
hold off

figure (2)
hold on
plot(t,error_y,'b');
legend('error');
xlabel('k');
ylabel('error');
axis([0,10,-0.2 , 0.2]);
hold off
```

Chapter 3

Section 3.1

```
% Chapter 3
% Estimation of Coefficients in Transfer Functions
% Case 1

%.....................................................................
.....

clear ;
close all;
clc ;
%.....................................................................
....%
```

```
% This is  to estimat parameters of first order system by using Kalamn
% filter
%.....................................................................
....%
N = 25 ;
x_s = zeros(2,101);

for n = 1 : 1 : N
% simulation of the system

kmax  = 102 ; %The number of iteration
t = 0 : 1 : kmax-1;
W = 0.1; % the covariance of noise
w = 0 + sqrt(W) * randn(1,10000);
y=zeros(1,kmax);
u=1+zeros(1,kmax);
y(1)=10;
a = 0.9 ;
b = 1;


for k = 2 : 1 : kmax

     y(k) = [-y(k-1),u(k)]*[a,b]'+ w(k);

end
%.....................................................................
....%

%  Estimated Parameters by Kalman Filter

xhat = NaN(2,kmax);
xhat(:,2) = [0.85,1.5];
P    = NaN(2,2,kmax);
P(:,:,2) = 1000*[1,0;0,1];
K    = NaN(2,kmax);

  for k = 2 : 1 : kmax

    C = [ -y(k-1),u(k) ] ;

    K(:,k) = P(:,:,k)*C' * (C*P(:,:,k)*C'+W)^-1 ;
    if k < kmax
        xhat(:,k+1)= xhat(:,k)+ K(:,k)*(y(k)-C*xhat(:,k));
        P(:,:,k+1) =P(:,:,k)-
(P(:,:,k)*C'*C*P(:,:,k))*(C*P(:,:,k)*C'+W)^-1;
    end
  end
  % To delete a first column
  xhat(:,1) = [];
  x_s = x_s + xhat ;

end


xhat = (1/N) * x_s ;
```

```
% simulation of the system with Estimared Parameters
ahat = xhat(1,kmax-1); % Estimated value of a
bhat = xhat(2,kmax-1); % Estimated value of b
yhat(1)=10;
for k = 2 : 1 : kmax

     yhat(k) = [-yhat(k-1),u(k)]*[ahat,bhat]'+ w(k);

end

error = y - yhat;
% Plot
k = 0 : 1  : kmax -2 ;
a = 0.9+zeros(1,kmax-1);
b = 1 + zeros(1,kmax-1);
%h1 = stairs(k,Theta(1,:),'r');
%set(h1,'linewidth',2);
figure (1)
subplot(2,1,1)
hold on
p1 = plot(k,xhat(1,:),'r:');
p2 = plot(k,a,'b--');
set(p1,'Linewidth',2);
set(p2,'Linewidth',2);
legend ('ahat','a');
xlabel('k');
ylabel('ahat , a')
hold off
subplot(2,1,2)
hold on
p3 = plot(k,xhat(2,:),'r:');
p4 = plot(k,b,'b--');
set(p3,'Linewidth',2);
set(p4,'Linewidth',2);
legend ('bhat','b');
xlabel('k');
ylabel('bhat , b')
hold off
k = 0 : 1  : kmax -1 ;
figure (2)
subplot(2,1,1)
hold on
p1 = plot(k,y,'r:');
p2 = plot(k,yhat,'b--');
set(p1,'Linewidth',2);
set(p2,'Linewidth',2);
xlabel('k');
ylabel('y & yhat');
legend('yhat' , 'y')
axis([0,kmax-1,-10,11])
hold off
subplot(2,1,2)
hold on
p1 =plot(k,error,'b');
set(p1,'Linewidth',2);
xlabel('k');
ylabel('error');
```

```
axis([0,kmax-1,-0.5,0.5])
hold off
% Chapter 3
% Estimation of Coefficients in Transfer Functions
% Case 2

%.....................................................................
.....

clear ;
close all;
clc ;
%.....................................................................
....%
% This is  to estimat parameters of first order system by using H-
infinity
% filter
%.....................................................................
....%
N = 25 ;
x_s = zeros(2,101);

for n = 1 : 1 : N
% simulation of the system

kmax  = 102 ; %The number of iteration
t = 0 : 1 : kmax-1;
y=zeros(1,kmax);
u=1+zeros(1,kmax);
w=zeros(1,kmax);
y(1)=10;
a = 0.9 ;
b = 1;


for k = 2 : 1 : kmax
     w(k) = 5*exp(-k);
     y(k) = [-y(k-1),u(k)]*[a,b]'+ w(k);

end
%.....................................................................
....%

%  Estimated Parameters by H-infinity

A = [1,0;0,1];
Cz = [0.1 , 0.1 ] ;
G = 1 ;
gamma = 10;
xhat=NaN(2,kmax);
xhat(:,1)=[0;0];
xhat(:,2)=[0.75;1.2];

K = NaN(2,kmax);

P = NaN(2,2,kmax);
P(:,:,2)=[1000,0;0,1000];
```

```
  for k=2:1:kmax
    C = [-y(k-1),u(k)];
    M = (P(:,:,k)^-1-Cz'*Cz)^-1;
    K(:,k)=(A*M*C')*(C*M*C'+gamma^-1*G*G')^-1;
    if k < kmax
        xhat(:,k+1) = A*xhat(:,k)+K(:,k)*(y(k)-C*xhat(:,k));
        P(:,:,k+1)=A*M*A'-(A*M*C')*(C*M*C'+gamma^-1*G*G')^-1*(C*M*A');
    end
   end
  % To delete a first column
  xhat(:,1) = [];
  x_s = x_s + xhat ;

end

xhat = (1/N) * x_s ;

% simulation of the system with Estimared Parameters
ahat = xhat(1,kmax-1); % Estimated value of a
bhat = xhat(2,kmax-1); % Estimated value of b
yhat(1)=10;
for k = 2 : 1 : kmax

     yhat(k) = [-yhat(k-1),u(k)]*[ahat,bhat]'+ w(k);

end



error = y - yhat;
% Plot
k = 0 : 1  : kmax -2 ;
a = 0.9+zeros(1,kmax-1);
b = 1 + zeros(1,kmax-1);

figure (1)
subplot(2,1,1)
hold on
p1 = plot(k,xhat(1,:),'r:');
p2 = plot(k,a,'b--');
set(p1,'Linewidth',2);
set(p2,'Linewidth',2);
axis([0 100 0 1.2])
xlabel('k');
ylabel('a & ahat');
legend('ahat' ,'a')
hold off
subplot(2,1,2)
hold on
p1 =plot(k,xhat(2,:),'r:');
p2 =plot(k,b,'b--');
p1 = set(p1,'Linewidth',2);
p2 = set(p2,'Linewidth',2);
axis([0 100 0 2])
xlabel('k');
ylabel('b & bhat');
legend('bhat' , 'b')
```

```
hold off
k = 0 : 1  : kmax -1 ;
figure (2)
subplot(2,1,1)
hold on
p1 = plot(k,y,'r:');
p2 = plot(k,yhat,'b--');
set(p1,'Linewidth',2);
set(p2,'Linewidth',2);
xlabel('k');
ylabel('y & yhat');
legend('yhat' , 'y')
axis([0 102 -11 11])
hold off
subplot(2,1,2)
hold on
p1 =plot(k,error,'b');
set(p1,'Linewidth',2);
xlabel('k');
ylabel('error');
axis([0 kmax -1 1])
hold off
```

```
% Chapter 3
% Estimation of Coefficients in Transfer Functions
% Case 3

%.....

clear ;
close all;
clc ;
%.....
....%
% This is  to estimat parameters of first order system by using H-
infinity
% filter
%.....
....%
N = 100 ;
x_s = zeros(2,101);

for n = 1 : 1 : N
% simulation of the system

kmax  = 102 ; %The number of iteration
t = 0 : 1 : kmax-1;
y=zeros(1,kmax);
u=1+zeros(1,kmax);
W = 0.1; % the covariance of noise
w = 0 + sqrt(W) * randn(1,10000);
y(1)=10;
a = 0.9 ;
b = 1;
```

```
for k = 2 : 1 : kmax

     y(k) = [-y(k-1),u(k)]*[a,b]'+ w(k);

end
%.....
....%

%  Estimated Parameters by H-infinity

A = [1,0;0,1];
Cz = [0.1 , 0.1 ] ;
G = 1 ;
gamma =10;
xhat=NaN(2,kmax);
xhat(:,1)=[0;0];
xhat(:,2)=[0.75;1.5];

K = NaN(2,kmax);

P = NaN(2,2,kmax);
P(:,:,2)=[1000,0;0,1000];

  for k=2:1:kmax
    C = [-y(k-1),u(k)];
    M = (P(:,:,k)^-1-Cz'*Cz)^-1;
    K(:,k)=(A*M*C')*(C*M*C'+gamma^-1*G*G')^-1;
    if k < kmax
        xhat(:,k+1) = A*xhat(:,k)+K(:,k)*(y(k)-C*xhat(:,k));
        P(:,:,k+1)=A*M*A'-(A*M*C')*(C*M*C'+gamma^-1*G*G')^-1*(C*M*A');
    end
   end
  % To delete a first column
  xhat(:,1) = [];
  x_s = x_s + xhat ;

end


xhat = (1/N) * x_s ;

% simulation of the system with Estimared Parameters
ahat = xhat(1,kmax-1); % Estimated value of a
bhat = xhat(2,kmax-1); % Estimated value of b
yhat(1)=10;
for k = 2 : 1 : kmax

     yhat(k) = [-yhat(k-1),u(k)]*[ahat,bhat]'+ w(k);

end


error = y - yhat;
% Plot
k = 0 : 1  : kmax -2 ;
a = 0.9+zeros(1,kmax-1);
```

```
b = 1 + zeros(1,kmax-1);

figure (1)
subplot(2,1,1)
hold on
p1 = plot(k,xhat(1,:),'r:');
p2 = plot(k,a,'b--');
set(p1,'Linewidth',2);
set(p2,'Linewidth',2);
%axis([0 100 0 1.2])
xlabel('k');
ylabel('a & ahat');
legend('ahat' ,'a')
hold off
subplot(2,1,2)
hold on
p1 =plot(k,xhat(2,:),'r:');
p2 =plot(k,b,'b--');
p1 = set(p1,'Linewidth',2);
p2 = set(p2,'Linewidth',2);
%axis([0 100 0 2])
xlabel('k');
ylabel('b & bhat');
legend('bhat' , 'b')
hold off

k = 0 : 1  : kmax -1 ;
figure (2)
subplot(2,1,1)
hold on
p1 = plot(k,y,'r:');
p2 = plot(k,yhat,'b--');
set(p1,'Linewidth',2);
set(p2,'Linewidth',2);
xlabel('k');
ylabel('y & yhat');
legend('yhat' , 'y')
axis([0 102 -11 11])
hold off
subplot(2,1,2)
hold on
p1 =plot(k,error,'b');
set(p1,'Linewidth',2);
xlabel('k');
ylabel('error');
axis([0 kmax -1 1])
hold off
```

Section 3.2

```
% Chapter 3
% 3.2   Simultaneous Parameter / State Estimation
% Case 1

%.....................................................................
.....

clear ;
close all ;
clc ;
```

```
clear ;
m = 25;
kmax = 101 ; % number of samples
s = zeros(2,kmax);

for i = 1 : 1 : m

kmax = 101 ; % number of samples

% Constant_Parametor_Identification

% x(k+1) = a x(k)+ f v(k)
% y(k) = c x(k)+ g w(k)

a1 = 0.9 ;
f = 1 ;
c = 1 ;
g = 1 ;


% Generation the noise which is normal distribution with zero mean

v = 0 + sqrt(0.1)*randn(1,10000); % mean + squeroot(covariance of
noise)* randn(1,k)
w = 0 + sqrt(0.1)*randn(1,10000); % mean + squeroot(covariance of
noise)* randn(1,k)
V = 0.1 ; % covariance of the process noise
W = 0.1 ; % covariance of the measurement noise

%.....................................................................
.....
% Simulation the system

x = NaN(1,kmax);
x(1) = 5 ; % initial value of x
y = NaN(1,kmax);

for k = 1 : 1 : kmax

    if k<kmax
        x(k+1) = a1*x(k)+f*v(k);
    end
    y(k) = c*x(k)+g*w(k);
end

%  Model system whihc will use in extended Kalman filter to estimate a
%  constant prameter that is unkwon. we consider linear system
% x(k+1) = a x(k)+ f v(k)
% y(k) = c x(k)+ g w(k)
% 'a' is unkown. we want to estimat its
% a(k+1) = a(k) . we rewrite the state of the system:
% [ x(k+1); a(k+1) ] = [a(k)*x(k) ; a(k)] + [1;0] v(k)
% y(k+1) = [c,0]*[x(k);a(k)]+w(k)

xhat = NaN(2,kmax) ; % It is estimation value of [x(k);a(k)]
xhat(:,1) = [7;0.8];
```

```
P = NaN(2,2,kmax) ; % Process convariance matrix
P(:,:,1) = [1000,0;0,1000] ;
K_k = NaN (2,kmax); % Gain of Kalman Filter

%erorr = NaN(1,N) ;
A = NaN(2,2,kmax);
C = [c , 0 ];
G = 1 ;
F =[1,0;0,0];


% Extended Kalman Filter
for k=1 : 1 : kmax
    x_hat = xhat(1,k);
    a_hat = xhat(2,k);
    A(:,:,k)= [ a_hat , x_hat ; 0 , 1 ] ;

    K_k(:,k) = A(:,:,k)*P(:,:,k)*C'*(C*P(:,:,k)*C'+G*W*G')^-1 ;
    if k < kmax
        %xhat(:,k+1) = A(:,:,k)*xhat(:,k)+K_k(:,k)*(y(k)-C*xhat(:,k))
;
        xhat(1,k+1) = a_hat * x_hat +K_k(1,k)*(y(k)-x_hat);
        xhat(2,k+1) = a_hat+K_k(2,k)*(y(k)-x_hat);

        P(:,:,k+1) = A(:,:,k)*P(:,:,k)*A(:,:,k)'-
K_k(:,k)*C*P(:,:,k)*A(:,:,k)'+F*V*F';
    end
end

s = s + xhat;

end

xhat = s / m ;

xhat_mean =  sum(xhat,2)/kmax;
ahat = 0.898 ;

% The estimated measurement
for k = 1 : 1 : kmax

    if k<kmax
        x(k+1) = ahat*x(k)+f*v(k);
    end
    yhat(k) = c*x(k)+g*w(k);
end
error = y - yhat;

t = 0 : 1 : kmax-1 ;
a = 0.9 + zeros(1,kmax);
figure (1)
hold on
p1 = plot(t,xhat(2,:),'r:');
p2 = plot(t,a,'b--');
set(p1,'Linewidth',2);
set(p2,'Linewidth',2);
```

```
xlabel('k');
ylabel(' a & ahat');
legend(' ahat ' , ' a');
axis([0,100,-1.2,1.2]);
hold off
axis([0 100, 0.6 1.1]);
hold off

k = 0 : 1  : kmax -1 ;
figure (2)
subplot(2,1,1)
hold on
p1 = plot(k,y,'r:');
p2 = plot(k,yhat,'b--');
set(p1,'Linewidth',2);
set(p2,'Linewidth',2);
xlabel('k');
ylabel('y & yhat');
legend('yhat' , 'y')
axis([0,kmax-1,-10,11])
hold off
subplot(2,1,2)
hold on
p1 =plot(k,error,'b:');
set(p1,'Linewidth',2);
xlabel('k');
ylabel('error');
axis([0,kmax-1,-0.5,0.5])
hold off
```

```
% Chapter 3
% 3.2   Simultaneous Parameter / State Estimation
% Case 2
%.....................................................................
.....

clear ;
close all ;
clc;

kmax =101 ;
x = NaN (1,kmax);
x(:,1) = 10 ;
y = NaN (1,kmax);

a = 0.9 ;
c = 1 ;
Cz = [1,1];
gamma = 25;

for k=1:1:kmax
    w1(k) = 3*exp(-k);
    w2(k) = 5*exp(-2*k);
    if k < kmax
        x(:,k+1) = a*x(k)+w1(k) ;
    end
    y(k) = c*x(k) +w2(k) ;
end
```

```
K = NaN (2,kmax);
P = NaN (2,2,kmax);
A = NaN (2,2,kmax);
xhat=NaN (2,kmax);
xhat(:,1) = [9,0.8] ;
P(:,:,1) =10000*[1,0;0,1];
g = gamma^-1;

for k=1:1:kmax

    x_hat = xhat(1,k);
    a_hat = xhat(2,k);

    A = [a_hat , x_hat ; 0 ,1 ];
    C = [c,0];
    F = [1,0;0,0];
    G = [0,1];
    M = (P(:,:,k)^-1 - Cz'*Cz)^-1 ;
    K(:,k)= M*C'*(C*M*C'+g*G*G')^-1;

    if k< kmax
        xhat(1,k+1) = a_hat*x_hat+K(1,k)*(y(k)-x_hat);
        xhat(2,k+1) = a_hat+K(2,k)*(y(k)-x_hat);

       P(:,:,k+1)   = (A*M*A'+g*F*F')-(A*M*C')*(C*M*A')*...
           (C*M*C'+g*G*G')^-1;
    end
end

ahat = sum(xhat(2,:))/kmax ;
% The estimated measurement
for k = 1 : 1 : kmax

    if k<kmax
        x(k+1) = ahat*x(k)+ w1(k);
    end

    yhat(k) = c*x(k)+w2(k);
end
error = y - yhat;

t = 0 : 1 : kmax-1 ;
a = 0.9 + zeros(1,kmax);

figure (1)
hold on
p1 = plot(t,xhat(2,:),'r:');
p2 = plot(t,a,'b--');
set(p1,'Linewidth',2);
set(p2,'Linewidth',2);
xlabel('k');
ylabel(' a & ahat');
legend(' ahat ' , ' a');
%axis([0,100,-1.2,1.2]);
hold off
%axis([0 100, 0.6 1.1]);
```

```
hold off
figure (2)
subplot(2,1,1)
hold on
p1 = plot(t,y,'r:');
p2 = plot(t,yhat,'b--');
set(p1,'Linewidth',2);
set(p2,'Linewidth',2);
xlabel('k');
ylabel('y & yhat');
legend('yhat' , 'y')
%axis([0,kmax-1,-10,11])
hold off
subplot(2,1,2)
hold on
p1 =plot(t,error,'b:');
set(p1,'Linewidth',2);
xlabel('k');
ylabel('error');
%axis([0,kmax-1,-0.5,0.5])
hold off
```

Section 3.3

```
% Chapter 3
% 3.3   Estimating Parameters by a Bank of Kalman Filters
% Case 1
%.....................................................................
.....

clear ;
clc ;
close all ;
% %  Estimation a parameter of first order system by using Bank of
Kalman Filter which
% its characteristics of noise is known wk ~ N(0, W^2)

%simulation of the system
N = 100 ;
t = 0 : 1 : N-1;
A = 0.9 ;
B = 1 ;
C = 1 ;
F = 1  ;
G = 1 ;

u = 1+zeros(1,N);
x = NaN(1,N);
x(:,1) = 5 ;
y = NaN(1,N);

% Generation the noise which is normal distribution with zero mean

v = 0 + sqrt(0.01)*randn(1,N); % mean + squeroot(covariance of noise)*
randn(1,k)
w = 0 + sqrt(0.01)*randn(1,N); % mean + squeroot(covariance of noise)*
randn(1,k)
V = 0.1 ; % covariance of the process noise
W = 0.1 ; % covariance of the measurement noise
```

```
for k = 1 : 1 : N

    if k<N
        x(k+1) = A*x(k)+B*u(k)+F*v(k);
    end
    y(k) = C*x(k)+G*w(k);
end

% Bank of Kalman Filter
a = [ 0.7, 0.8 ,0.9 , 1 ,1.1];

K = NaN(1,N,5); % Gain of Kalman FIlter
P = NaN(1,N,5); % Process Convariance Matrix
xhat=NaN(1,N,5);% Estimation of state
yhat=NaN(5,N);% yhat = C xhat(k)
y_t =NaN(5,N);% y_t = y(k) - yhat

% initial value  of P and xhat
for i = 1 : 5
P(:,:,i)=100;
xhat(:,:,i)= 6 ;
end
p_hat_a1 = [1/length(a) NaN(1,length(N)-1)];
p_hat_a2 = [1/length(a) NaN(1,length(N)-1)];
p_hat_a3 = [1/length(a) NaN(1,length(N)-1)];
p_hat_a4 = [1/length(a) NaN(1,length(N)-1)];
p_hat_a5 = [1/length(a) NaN(1,length(N)-1)];

for k = 2 : 1 : N
    for i = 1 : 1 : 5
        A = a(i);
        K(1,k,i) = A*P(1,k,i)*C'*(C*P(1,k,i)*C'+G*W*G')^-1 ;
         if k < N
             xhat(1,k+1,i)= A*xhat(1,k,i)+B*u(k)+K(1,k,i)*(y(k)-
C*xhat(1,k,i));
             P(1,k+1,i)= A*P(1,k,i)*A'-K(1,k,i)*C*P(k)*A'+F*V*F';
         end
         yhat(i,k) = C * xhat(1,k,i) ;
    end
    y_t(:,k)= [y(k);y(k);y(k);y(k);y(k)] - yhat(:,k);

    % Design Covariance for Kalman Filter

    dc1=C*P(1,k,1)*C+G*W*G';
    dc2=C*P(1,k,2)*C+G*W*G';
    dc3=C*P(1,k,3)*C+G*W*G';
    dc4=C*P(1,k,4)*C+G*W*G';
    dc5=C*P(1,k,5)*C+G*W*G';

    % Gaussian Conditional Probabilities
    p_y_Y1= (2*pi)^(-1/2)*sqrt(1/det(dc1))*exp(-
0.5*y_t(1,k)'*eye/dc1*y_t(1,k));
    p_y_Y2= (2*pi)^(-1/2)*sqrt(1/det(dc2))*exp(-
0.5*y_t(2,k)'*eye/dc2*y_t(2,k));
    p_y_Y3= (2*pi)^(-1/2)*sqrt(1/det(dc3))*exp(-
0.5*y_t(3,k)'*eye/dc3*y_t(3,k));
```

```
    p_y_Y4= (2*pi)^(-1/2)*sqrt(1/det(dc4))*exp(-
0.5*y_t(4,k)'*eye/dc4*y_t(4,k));
    p_y_Y5= (2*pi)^(-1/2)*sqrt(1/det(dc5))*exp(-
0.5*y_t(5,k)'*eye/dc5*y_t(5,k));

    S =(p_y_Y1*p_hat_a1(k-1))+(p_y_Y2*p_hat_a2(k-
1))+(p_y_Y3*p_hat_a3(k-1))...
        +(p_y_Y4*p_hat_a4(k-1))+(p_y_Y5*p_hat_a5(k-1));

    p_hat_a1(k) = p_y_Y1*p_hat_a1(k-1)/S;
    p_hat_a2(k) = p_y_Y2*p_hat_a2(k-1)/S;
    p_hat_a3(k) = p_y_Y3*p_hat_a3(k-1)/S;
    p_hat_a4(k) = p_y_Y4*p_hat_a4(k-1)/S;
    p_hat_a5(k) = p_y_Y5*p_hat_a5(k-1)/S;
end



figure()
plot(t,p_hat_a1,'b',t,p_hat_a2,'y',t,p_hat_a3,'r',t,p_hat_a4,'g',t,p_h
at_a5,'k');
axis([0 100 0 1.2]);
legend(['p(a_1|Y_k), a_1 = ' num2str(a(1)) ],['p(a_2|Y_k), a_2 = '
num2str(a(2)) ],['p(a_3|Y_k), a_3 = ' num2str(a(3)) ],['p(a_4|Y_k),
a_4 = ' num2str(a(4)) ],['p(a_5|Y_k), a_5 =' num2str(a(5))]);
%title('A Posteriori Probabilities for each KF ');
xlabel('k');
ylabel('p(a_i|Y_i)');
grid;
```

```
% Chapter 3
% 3.3   Estimating Parameters by a Bank of Kalman Filters
% Case 2
%.....................................................................
.....

clear ;
clc ;
close all ;
% %  Estimation a parameter of first order system by using Bank of
Kalman Filter which
% its characteristics of noise is known wk ~ N(0, W^2)

%simulation of the system
N = 500 ;
t = 0 : 1 : N-1;
A = 0.9 ;
B = 1 ;
C = 1 ;
F = 1  ;
G = 1 ;

u = 1+zeros(1,N);
x = NaN(1,N);
x(:,1) = 5 ;
y = NaN(1,N);
```

```
% Generation the noise which is normal distribution with zero mean

v = 0 + sqrt(0.01)*randn(1,N); % mean + squeroot(covariance of noise)*
randn(1,k)
w = 0 + sqrt(0.01)*randn(1,N); % mean + squeroot(covariance of noise)*
randn(1,k)
V = 0.1 ; % covariance of the process noise
W = 0.1 ; % covariance of the measurement noise

for k = 1 : 1 : N

    if k<N
        x(k+1) = A*x(k)+B*u(k)+F*v(k);
    end
    y(k) = C*x(k)+G*w(k);
end

% Bank of Kalman Filter
a = [0.8,0.9,1];
b = [0.9,1,1.1];

K = NaN(1,N,9); % Gain of Kalman FIlter
P = NaN(1,N,9); % Process Convariance Matrix
xhat=NaN(1,N,9);% Estimation of state
yhat=NaN(9,N);% yhat = C xhat(k)
y_t =NaN(9,N);% y_t = y(k) - yhat

% initial value  of P and xhat
for i = 1 : 9
P(:,:,i)=100;
xhat(:,:,i)= 10 ;
end
p_hat_a1 = [1/length(a) NaN(1,length(N)-1)];
p_hat_a2 = [1/length(a) NaN(1,length(N)-1)];
p_hat_a3 = [1/length(a) NaN(1,length(N)-1)];
p_hat_a4 = [1/length(a) NaN(1,length(N)-1)];
p_hat_a5 = [1/length(a) NaN(1,length(N)-1)];
p_hat_a6 = [1/length(a) NaN(1,length(N)-1)];
p_hat_a7 = [1/length(a) NaN(1,length(N)-1)];
p_hat_a8 = [1/length(a) NaN(1,length(N)-1)];
p_hat_a9 = [1/length(a) NaN(1,length(N)-1)];

for k = 2 : 1 : N
    i = 0 ;
    for j = 1 : 1 : 3
        A = a(j);
        for l=1 : 1 : 3
        B = b(l);
        i = i+1  ;
        K(1,k,i) = A*P(1,k,i)*C'*(C*P(1,k,i)*C'+G*W*G')^-1 ;
         if k < N
             xhat(1,k+1,i)= A*xhat(1,k,i)+B*u(k)+K(1,k,i)*(y(k)-
C*xhat(1,k,i));
             P(1,k+1,i)= A*P(1,k,i)*A'-K(1,k,i)*C*P(k)*A'+F*V*F';
         end
         yhat(i,k) = C * xhat(1,k,i) ;
        end
```

```
    end
    y_t(:,k)= [y(k);y(k);y(k);y(k);y(k);y(k);y(k);y(k);y(k)] -
yhat(:,k);

    % Design Covariance for Kalman Filter

    dc1=C*P(1,k,1)*C+G*W*G';
    dc2=C*P(1,k,2)*C+G*W*G';
    dc3=C*P(1,k,3)*C+G*W*G';
    dc4=C*P(1,k,4)*C+G*W*G';
    dc5=C*P(1,k,5)*C+G*W*G';
    dc6=C*P(1,k,6)*C+G*W*G';
    dc7=C*P(1,k,5)*C+G*W*G';
    dc8=C*P(1,k,8)*C+G*W*G';
    dc9=C*P(1,k,9)*C+G*W*G';
    % Gaussian Conditional Probabilities
    p_y_Y1= (2*pi)^(-1/2)*sqrt(1/det(dc1))*exp(-
0.5*y_t(1,k)'*eye/dc1*y_t(1,k));
    p_y_Y2= (2*pi)^(-1/2)*sqrt(1/det(dc2))*exp(-
0.5*y_t(2,k)'*eye/dc2*y_t(2,k));
    p_y_Y3= (2*pi)^(-1/2)*sqrt(1/det(dc3))*exp(-
0.5*y_t(3,k)'*eye/dc3*y_t(3,k));
    p_y_Y4= (2*pi)^(-1/2)*sqrt(1/det(dc4))*exp(-
0.5*y_t(4,k)'*eye/dc4*y_t(4,k));
    p_y_Y5= (2*pi)^(-1/2)*sqrt(1/det(dc5))*exp(-
0.5*y_t(5,k)'*eye/dc5*y_t(5,k));
    p_y_Y6= (2*pi)^(-1/2)*sqrt(1/det(dc6))*exp(-
0.5*y_t(6,k)'*eye/dc6*y_t(6,k));
    p_y_Y7= (2*pi)^(-1/2)*sqrt(1/det(dc7))*exp(-
0.5*y_t(7,k)'*eye/dc7*y_t(7,k));
    p_y_Y8= (2*pi)^(-1/2)*sqrt(1/det(dc8))*exp(-
0.5*y_t(8,k)'*eye/dc8*y_t(8,k));
    p_y_Y9= (2*pi)^(-1/2)*sqrt(1/det(dc9))*exp(-
0.5*y_t(9,k)'*eye/dc9*y_t(9,k));

    S =(p_y_Y1*p_hat_a1(k-1))+(p_y_Y2*p_hat_a2(k-
1))+(p_y_Y3*p_hat_a3(k-1))...
        +(p_y_Y4*p_hat_a4(k-1))+(p_y_Y5*p_hat_a5(k-1))+ ...
        (p_y_Y6*p_hat_a6(k-1))+(p_y_Y7*p_hat_a7(k-1))+...
        (p_y_Y8*p_hat_a8(k-1))+(p_y_Y9*p_hat_a9(k-1));

    p_hat_a1(k) = p_y_Y1*p_hat_a1(k-1)/S;
    p_hat_a2(k) = p_y_Y2*p_hat_a2(k-1)/S;
    p_hat_a3(k) = p_y_Y3*p_hat_a3(k-1)/S;
    p_hat_a4(k) = p_y_Y4*p_hat_a4(k-1)/S;
    p_hat_a5(k) = p_y_Y5*p_hat_a5(k-1)/S;
    p_hat_a6(k) = p_y_Y6*p_hat_a6(k-1)/S;
    p_hat_a7(k) = p_y_Y7*p_hat_a7(k-1)/S;
    p_hat_a8(k) = p_y_Y8*p_hat_a8(k-1)/S;
    p_hat_a9(k) = p_y_Y9*p_hat_a9(k-1)/S;
end



figure()
hold on
plot(t,p_hat_a1);
```

```
plot(t,p_hat_a2);
plot(t,p_hat_a3);
plot(t,p_hat_a4);
plot(t,p_hat_a5);
plot(t,p_hat_a6);
plot(t,p_hat_a7);
plot(t,p_hat_a8);
plot(t,p_hat_a9);

legend(['p(a_1,b_1|Y_k),a_1=',num2str(a(1)),',b_1=',num2str(b(1))],...
    ['p(a_1,b_2|Y_k),a_1=',num2str(a(1)),',b_2=',num2str(b(2))],...
    ['p(a_1,b_3|Y_k),a_1=',num2str(a(1)),',b_3=',num2str(b(3))],...
    ['p(a_2,b_1|Y_k),a_2=',num2str(a(2)),',b_1=',num2str(b(1))],...
    ['p(a_2,b_2|Y_k),a_2=',num2str(a(2)),',b_2=',num2str(b(2))],...
    ['p(a_2,b_3|Y_k),a_2=',num2str(a(2)),',b_3=',num2str(b(3))],...
    ['p(a_3,b_1|Y_k),a_3=',num2str(a(3)),',b_1=',num2str(b(1))],...
    ['p(a_3,b_2|Y_k),a_3=',num2str(a(3)),',b_2=',num2str(b(2))],...
    ['p(a_3,b_2|Y_k),a_3=',num2str(a(3)),',b_3=',num2str(b(3))]);
xlabel('k');
ylabel('p(a_i,b_i|Y_i)');
axis([0 500 0 1.2]);

hold off
```

Chapter 4

Section 4.1

```
% Chapter 4
% 4.1 The sample mean method
% Case 1
%.....................................................................
.....

clear ;
clc ;
close all ;
% A sample mean method to detect a fault signal
% Sensor Hacking



a = 0.85;
b = 2 ;
c = 3;
h  = 10 ;
s_t = 100 ;
kmax =301 ; % number of simple

v = 0 + sqrt(0.1) * randn(1,100000);
w = 0 + sqrt(0.1) * randn(1,100000);
u = 1 + zeros(1,kmax);
x = NaN (1,kmax);
y = NaN (1,kmax);
y_dash = NaN (1,kmax);

x(1) = 3;

% x(k+1) = a x(k) + v(k)
%
for k = 1 : 1 : kmax
```

```
    % Simolation the system

    if k < kmax
        x(k+1) = a*x(k)+b*u(k)+v(k);
    end

    if k < s_t
           y(k) = c * x(k) + w(k) ;
    else
        y(k) =  h + w(k) ;
    end

    %-------------------------------------------

    % TO Compute a sample mean

       %x_mean(k)  = (x(k) + s_m ) / k  ;
       %y_dash(k)  = (y(k) + s_m ) / k  ;
       %  s_m  = y_dash(k) ;


    % ------------------------------------------------------


end

% A sample mean of the measurement

[n,d]=ss2tf(a,b,c,0);
y_mean = filter(n,d,u);
y_tilde = y - y_mean ;
t  =  0 : kmax-1 ;

figure (1)
hold on
plot(t,x);
plot( t,y );
legend('x_k','y_k ');
xlabel('k');
ylabel('x_k ,y_k');
hold off
figure (2)
hold on
plot ( t, y_mean);
legend('Sample mean of y');
xlabel('k');
ylabel('Sample Mean of y')
axis([0 300 -1 43]);
hold off

figure (3)
hold on
plot(t,y_tilde);
legend('y-tilde = y - y-mean');
xlabel('k');
```

```
ylabel('y-tilde')
hold off
% Chapter 4
% 4.1 The sample mean method
% Case 2
%.....................................................................
.....


clear ;
clc ;
close all ;

% The Sample mean method to detect a hacked when it happen at actuator
%  x(k+1) = (a+Kb) x(k) + v(k)
%  y(k)   = c x(k) + w(k)
a = 1.2;
b = 1 ;
c = 2 ;
K = 1.1 ;  % the feedbake
h  = 10 ;
s_t = 100 ;
kmax =101 ; % number of simple

v = 0 + sqrt(0.1) * randn(1,kmax);
w = 0 + sqrt(0.1) * randn(1,kmax);

x = NaN (1,kmax);
y = NaN (1,kmax);
x_dash = NaN (1,kmax);
s_m = 0 ;
x(1) = 10  ;



for k = 1 : 1 : kmax


    % sample mean when the actuator is not attacked

    x_m(k) = (a-b*K)^k * x(1) ;

    % sample mean when the actuator is attacked

    x_mh(k) = a^k * x(1) + (a^k - 1)*(a-1)^-1 * b *h;



end

for k = 1 : 1 : kmax

    % Simolation the system

    if k < kmax
```

```
        x(k+1) = (a-b*K)*x(k)+ v(k);

        if k > s_t
            x(k+1) = a*x(k)+ b *h + v(k);
        end
    end
    y(k) = c * x(k) + w(k) ;
end
t  =  0 : kmax-1 ;
figure (1)
hold on
plot(t,x);
legend('The sample mean with attacked');
xlabel('k');
ylabel('The theoretical sample mean  ');
%axis ([0 100 10 55])
hold off
```

Section 4.2

```
% Chapter 4
% 4.2 Kalman filter method
% Case 1
%.....
% Sensor Hacking
% x(k+1) = a x(k) + F v(k)
% y(k) = c x(k) + G w(k)
% y(k) = h(k) + Gw(k)
clear ;
close all ;
clc ;

kmax = 100 ;
t = 0 : 1 : kmax-1 ;

a = 0.9;
F = 1 ;
c = 1 ;
G = 1 ;
V = 0.01;
W = 0.01;
v= sqrt(V)*randn(1,10000); % the noise of  the state
w = sqrt(W)*randn(1,10000); % the noise of the measurement

% simulation for system
x = zeros(1,kmax);
h = 10+zeros(1,kmax);
y = zeros(1,kmax);
x(1,1) = 1;
t_s = 30; % The time which the system changes from normal situation to
hacking situation

for k = 1 : 1 : kmax

        if k < kmax
            x(k+1) = a * x(k) + F * v(k) ;
        end
        if k < t_s
```

```
        y(k) = c * x(k) + G * w(k) ;
        else

        y(k) = c * h(k)+ G * w(k)  ;
        end
end

% Kalman Filter for Model one and two
xhat1 = NaN(1,kmax);
xhat1(1,1) = 0.8 ;
xhat2 = NaN(2,kmax);
xhat2(:,1) = [0.8 ; 4 ] ;
Kk1 = NaN(1,kmax);
Kk2 = NaN(2,kmax);
P1 = NaN(1,kmax);
P1(1,1) = 10000 ;
P2 = NaN(2,2,kmax);
P2(:,:,1) = [10000,0;0,10000];

ydilt1 = NaN(1,kmax);
ydilt2 = NaN(1,kmax);
Omega1  = NaN(1,kmax);
Omega2  = NaN(1,kmax);
pThetaZk1 = NaN(1,kmax);
pThetaZk2 = NaN(1,kmax);
pThetaZk1(1,1) = 0.5;
pThetaZk2(1,1) = 0.5;
A = [ a , 0 ; 0 ,1] ;
C = [0.001,1] ;
V1 = [V,0;0,0];
Ff = [ F,0 ; 0,0 ] ;

for k = 1 : 1 : kmax


     Kk1(k) = a * P1(k) * c' * (c*P1(k)*c'+ G*W*G')^(-1) ;  % The Gain
for first model

     Kk2(:,k) = A * P2(:,:,k) * C' * ( C*P2(:,:,k)*C'+G*W*G')^-1 ;  %
The Gain for Second model

    if k < kmax
      xhat1(k+1) = a * xhat1(k) + Kk1(1,k) * (y(k) - c * xhat1(k));
      xhat2(:,k+1) = A * xhat2(:,k) + Kk2(:,k) * (y(k) - C *
xhat2(:,k));

      P1(k+1) = a * P1(k) * a' - Kk1(k) * c*P1(k)*a' + F*V*F' ;
      P2(:,:,k+1) = A * P2(:,:,k) * A' -
Kk2(k)*C*P2(:,:,k)*A'+Ff*V1*Ff';
    end

    ydilt1(k) = y(k) - c * xhat1(k);
    ydilt2(k) = y(k) - C * xhat2(:,k);

    Omega1(k) = c * P1(k) * c' + W;
```

```
    Omega2(k) = C* P2(:,:,k) * C' + W;

    pzkTheta1(k)= ((2*pi)^(-1/2))*sqrt(1/abs( Omega1(k)))*exp(-
0.5*ydilt1(k)'*(Omega1(k)^-1)*ydilt1(k));

    pzkTheta2(k)= ((2*pi)^(-2/2))*sqrt(1/abs( Omega2(k)))*exp(-
0.5*ydilt2(k)'*(Omega2(k)^-1)*ydilt2(k));

    den(k) = pzkTheta1(k)*pThetaZk1(k) + pzkTheta2(k)*pThetaZk2(k);

    if k < kmax
        pThetaZk1(k+1) = (pzkTheta1(k)*pThetaZk1(k))/den(k);
        pThetaZk2(k+1) = (pzkTheta2(k)*pThetaZk2(k))/den(k);
    end

end

figure (1)
hold on
plot(t,pThetaZk1,'r') ;
plot(t,pThetaZk2,'b');
%title('A Posteriori Probabilities for each KF ');
legend('P without unattacked','P with attacked');
xlabel('k');
ylabel('Probabilities for each KF');
hold off
```

```
% Chapter 4
% 4.2 Kalman filter method
% Case 2
%.....................................................................
% Actuator Hacking
clear;
clc;
close all;

% The system which we will simulate
%[x(k+1);h(k+1)]=[A,0;0,I]*[x(k);h(k)]+[Kc,0;0,0]*[xhat(k);hhat(k)]+[F
;0]*v(k)
% y(k) =  [C,I]*[x(k);h(k)]+ G w(k)

A = 0.85;
B = 1 ;
F = 1;
C = 1;
G = 1 ;
I = 1 ;
Kc = 0.1;
kmax = 100; % it is the high value of  time
t = 0 : 1 : kmax-1;
s_t = 30; % The time when switching to hacking situation


x = NaN(1,kmax);
x (1,1) = 1;
```

```
h = NaN(1,kmax);
y= NaN(1,kmax);
v= sqrt(0.1)*randn(1,10000); % the noise of  the state
w = sqrt(0.1)*randn(1,10000); % the noise of the measurement
V = 0.1;
W = 0.1;

A2 = [A , B ; 0 , 1];
F2 = [F;0];
C2 = [C , 0];
Xhat1 = NaN (1,kmax);
Xhat2 = NaN (2,kmax);
Kk1   = NaN (1,kmax);
Kk2   = NaN (2,kmax);
P1    = NaN (1,kmax);
P2    = NaN (2,2,kmax);
ydilt1= NaN (1 , kmax);
ydilt2= NaN (1 , kmax);
Omega1= NaN (1,kmax);
Omega2= NaN (1,kmax);
pThetaZk1 = [0.5 NaN(1,length(t)-1)]; %weight
pThetaZk2 = [0.5 NaN(1,length(t)-1)]; %weight

x(:,1) = 1;
Xhat1(1,1)= 1;
Xhat2(:,1)= [1,1];
u = NaN ( 1 , kmax);
u(1,1) = 1;
P1(1,1) = 100 ;
P2(:,:,1)   = [100,0;0,100];



for k = 1 : 1 : kmax
     % System
   if k < s_t

       if k < kmax
       x(1,k+1) = A * x(k) - B*Kc*u(k)+ F*v(1,k);
       end
       y(1,k) = C * x(1,k) + G*w(1,k);
       h(1,k) = 0 ;

    else

        if k < kmax
        h(k) = 10;
        x(1,k+1) = A * x(1,k)+ B * h(1,k) + F*v(1,k);
        u(k) = h(1,k);
        end
        y(1,k) = C * x(1,k) + G*w(1,k);

   end


    % Kalman Filter
```

```
    Kk1(k) = A* P1(k) * C' * (C*P1(k)*C' + G*W*G')^-1;
    Kk2(:,k) = A2* P2(:,:,k) * C2' * (C2*P2(:,:,k)*C2' + G*W*G')^-1;

    if k < kmax
        Xhat1(k+1) = A * Xhat1(k) - Kc * B * u(k) + Kk1(k)*(y(k)-
C*Xhat1(k));

        Xhat2(:,k+1) = A2 * Xhat2(:,k)+ Kk2(:,k)*(y(k)-C2*Xhat2(:,k));

        P1(k+1)  = A * P1(k) * A - C * P1(k) *C' * ( C ...
            *  P1(k) *C' + G*W*G')^-1 * C* P1(k)*A' + F*V*F';


        P2(:,:,k+1)  = A2 * P2(:,:,k) * A2 - C2* P2(:,:,k) *C2' * ( C2
...
            * P2(:,:,k) *C2' + G*W*G')^-1 * C2*P2(:,:,k)*A2' +
F2*V*F2';


    end


    ydilt1(k) = y(k) - C * Xhat1(1,k);
    ydilt2(k) = y(k) - C2 * Xhat2 (:,k);

     Omega1(k) = C* P1(k) * C' + W;
     Omega2(k) = C2* P2(:,:,k) * C2' + W;



    pzkTheta1= ((2*pi)^(-1/2))*sqrt(1/abs( Omega1(k)))*exp(-
0.5*ydilt1(k)'*(Omega1(k)^-1)*ydilt1(k));
    pzkTheta2= ((2*pi)^(-1/2))*sqrt(1/abs( Omega2(k)))*exp(-
0.5*ydilt2(k)'*(Omega2(k)^-1)*ydilt2(k));


    den = pzkTheta1*pThetaZk1(k) + pzkTheta2*pThetaZk2(k);

    if k < kmax
        pThetaZk1(k+1) = (pzkTheta1*pThetaZk1(k))/den;
        pThetaZk2(k+1) = (pzkTheta2*pThetaZk2(k))/den;
    end
   u(k+1) = pThetaZk1(k)*Xhat1(k) + pThetaZk2(k)*Xhat2(1,k);
end



figure (1)
hold on
plot(t, pThetaZk1,'r');
plot(t, pThetaZk2,'b');
%title('A Posteriori Probabilities for each KF ');
legend('P without attacking','P with attacking ');
hold off
```

Section 4.3

```
% Chapter 4
% 4.3 Stochastic Parameter Estimation Method
%
%.....................................................................
.....

clear ;
close all ;
clc ;
% State Estimate of System with Stochastic

% The system in Unhacking case
% x(k+1) = a x(k) + v(k) , v ~ N(0,0.1)
% y(k) = Ck x(k) + w(k)    , w ~ N(0,0.1)
% The system in Hacking case
% x(k+1) = [ A , 0 ; 0 , I ] * x(k) + [ v(k) ; 0 ]
% y(k) = Ck x(k) + w(k)
clear ;
close all ;
clc  ;

kmax = 301 ;
t = 0 : 1 : kmax-1;
%s_t = randi ([20,kmax-25]); % The time when switching to hacking
situation
s_t  = 100 ;
a = 0.8 ;
c = 1 ;
pu = 0.9 ; % Probability for Unhacking case
ph = 1- pu ; % Probability for Hacking case
v  = 0 + 0.1 * randn(1,kmax);
V  = 0.1;
w  = 0 + 0.1 * randn(1,kmax);
W = 0.1;

A = [a,0;0,1] ;

% Ck = [ c , 0 ] when probability is pu
% Ck = [ 0 , 1 ] when probability is ph
Ckm = [ pu*c , ph * 1] ; % Ckm is E{Ck} (mean of Ck)

% Ck_dilta = [ c , 0 ] - [ pu*c , ph * 1] when probability is pu
% Ck_dilta = [ 0 , 1 ] - [ pu*c , ph * 1] when probability is ph
X = NaN (2,2,kmax);
X(:,:,1) = [ 2 , 1 ; 1, 2 ];



x = NaN  (1,kmax);
x(:,1) =2;
h = 10 + zeros(1,kmax);
y = NaN  (1,kmax);

Kk = NaN(2,kmax);
```

```
P  = NaN(2,2,kmax);
P(:,:,1) = 1000 * [1,0;0,1];

xhat=NaN(2,kmax);
xhat(:,1) = [3;6];




for k=1:1:kmax

  % Simulation the system
  if k < kmax

      if k < s_t
      x(k+1) = a*x(k)+ v(k);
      h(k) = 0;
      y(k) = x(k) + w(k);
      else
      x(k+1) = a*x(k)+ v(k);
      h(k+1)= h(k);
      y(k) = h(k) + w(k);
      end
  end



  X(:,:,k+1) = A * X(:,:,k)* A' + [V,0;0,0];

  Ck_diltam = pu*([c,0]-[pu*c,ph*1])*X(:,:,k)*([c,0]-[ pu*c , ph *
1])'...
+ ph*([0,1] - [ pu*c,ph*1]) * X(:,:,k) *([0,1]-[ pu*c,ph*1])'; % This
is E{Ck_dilta * X * Ck_dilta'}


  % Kalman Filter



    Kk(:,k) = A*P(:,:,k)*Ckm'*(Ckm*P(:,:,k)*Ckm'+W+Ck_diltam)^-1;

        if k < kmax

            xhat(:,k+1) = A*xhat(:,k)+Kk(:,k)*(y(k)-Ckm*xhat(:,k));

            P(:,:,k+1)= A*P(:,:,k)*A'-
A*P(:,:,k)*Ckm'*(Ckm*P(:,:,k)*Ckm'...
                +W+Ck_diltam)^-1*Ckm*P(:,:,k)*A'+[V,0;0,0];
        end



end
```

```
figure (1)
subplot(2,1,1)
hold on
plot(t,x,'b');
plot(t, xhat(1,:),'r');
title('');
legend('x','xhat');
xlabel('k');
ylabel('x & xhat')
hold off
subplot(2,1,2)
hold on
plot(t,h,'b');
plot(t, xhat(2,:),'r');
title('');
legend('h','hhat');
xlabel('k');
ylabel('h & hhat')
hold off
```