2013

Moving Horizon Estimation with Dynamic Programming

Mohan Kumar Ramalingam
Cleveland State University

MOVING HORIZON ESTIMATION WITH DYNAMIC PROGRAMMING

MOHAN KUMAR RAMALINGAM

Bachelor of Technology in Chemical Engineering

Anna University, Chennai, India

submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE IN CHEMICAL ENGINEERING

at the

CLEVELAND STATE UNIVERSITY

December 2013

We hereby approve this thesis of

MOHAN KUMAR RAMALINGAM

Candidate for the Master of Science in Chemical Engineering degree for the

Department of Chemical and Biomedical Engineering

and the **CLEVELAND STATE UNIVERSITY**

College of Graduate Studies

_____

Thesis Committee Chairperson, Dr. Sridhar Ungarala

_____

Chemical and Biomedical Engineering & Date

_____

Dr. Rolf Lustig

_____

Chemical and Biomedical Engineering & Date

_____

Dr. Jorge Gatica

_____

Chemical and Biomedical Engineering & Date

Student's Date of Defense: December 3, 2013

Dedicated to my dear Mom and Dad

ACKNOWLEDGMENTS

I would like to thank a number of people who have encouraged me in the completion of my thesis.

First and foremost, I would like to give my sincere thanks and regards to my advisor Dr. Sridhar Ungarala, who has given his heart and soul in encouraging and motivating me right throughout the completion of this thesis. He has showed a lot of patience and I sincerely owe him a all of my gratitude for everything he has done.

I would like to thank my thesis committee members Dr. Jorge Gatica who had laid the basis of my programming skills and Dr. Rolf Lustig for his guidance throughout this work.

I would also love to give my thanks to the whole of the Department of Chemical and Biomedical Engineering and its staffs, including Ms. Becky Laird and Ms. Darlene Montgomery.

My hearty thanks to all my friends at the university along with my thesis group and my friends back home who have provided me with as much support as possible.

All thanks are undone if my parents and God are not thanked whole heartedly. They have helped me so much that my thanks wont do much good.

MOVING HORIZON ESTIMATION WITH DYNAMIC PROGRAMMING

MOHAN KUMAR RAMALINGAM

ABSTRACT

Moving Horizon Estimation(MHE) is a optimization based strategy to state estimation. It involves computation of arrival cost, a penalty term, based on the MHE cost function. Minimization of this arrival cost is done through various methods. All these methods use nonlinear programming optimization technique which gives the estimate. The main idea of MHE revolves around minimizing the estimation cost function. The cost function is dependent on prediction error computation from data and arrival cost summarization. The major issue that hampers the MHE is choosing the arrival cost for ensuring stability of the overall estimation and computational time. In order to attain this stability, this thesis incorporates dynamic programming algorithm to estimate MHE cost function. Dynamic programming is an algorithm for solving complex problems. The MHE cost function algorithm has been modified based on dynamic programming algorithm in order to ensure stability of the overall estimation. In order to apply this algorithm, a specific non-linear filter, particle filter is used for the initialization of MHE. The reason of using particle filter for initialization of MHE is due to fact that dynamic programming algorithm works on principle of samples and particle filter provides the samples. A comparison of mean squared

error(MSE) using the nonlinear programming optimization and dynamic programming optimization is verified for the proposed theory of using dynamic programming algorithm in estimation of cost function.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

At present the competitive nature of this ever growing market trend seizes to amaze everyone. The increase in importance of quality of every product and the other environmental issues have given rise to the need of improving the performance of the existing chemical processes.

Therefore in order to improve the performance, knowledge on the actual state of the system is required. The heart of any chemical engineering, or generally engineering and sciences deals with observation or measurement and process or state. This information is obtained from processes by collecting a set of data or by an already existing model. The model is given to estimate, on the basis of given initial knowledge of the system. But finding an accurate model may be a difficult problem in any application.

The essential need of improving performance on any system requires attaining reliable and complete information about the process. The main idea of estimation is because of the fact that in almost all realistic situations, the observations under study are contaminated with disturbances or errors. Observations always contain some type

of error, it is necessary to correct the values. So there is a need for certain methods to filter out the disturbances in order to arrive at the result. The errors are of two types, random and systematic errors. Small errors that are due to the normal fluctuation of the process or random variant inherent in instrument or sensor operations are called random errors. In this thesis, all supporting examples correspond to random errors. Systematic errors are large errors due to wrong calibration or malfunction of the instruments which occur occasionally.

Interest in more detailed knowledge on the state of the system leads to state estimation. In simple terms, state of the process is a variable which determines the behavior of the system. If the system is without any errors, then knowledge on the state of the system at a particular time is enough to predict the state at the future time instant.

All physical systems are modeled so as to perform certain functions. In order to determine whether a system is performing properly, the engineer must know what the system is doing at any time instant. In navigation, the state consists of position and velocity of the craft. In a batch reactor, the state consists of concentrations, partial pressures, temperature, mole fractions, etc. In an AC electric power system, the state consists of voltages and phase angles at network nodes. Therefore, in order to determine these states the engineer builds an observation or measurement device. The observation or measurement device can be sensors or other distributed control systems. Here in this thesis, the measurements are considered to be obtained from sensor devices and are generally contaminated by random errors called noise.

The physical system is modeled by a finite dimensional Markov process, the output of a stochastic differential or difference equation. The state estimation in a batch reactor system is an example for dynamic model representation in this thesis. The Bayesian or probabilistic view of filtering is used. That is explained later in

Bayesian state estimation.

In general, at a given time point or step, an estimate can be arrived from the measurement and the model of the system using any filtering methods existing, given, its initial conditions. Often in practical estimation problems a reliance of any one, that is, either the measurement or the state model can provide estimate with insufficient accuracy. Therefore, it is of considerable practical interest to have knowledge on optimal strategies that can be used to combine both the measurement and state model in wide range of estimation problems. In order to determine the optimal estimation strategy, satisfying a minimum variance estimate for a wide range of problems is the topic of interest today.

The main goal of state estimation is to redefine the state of the system in derivative form, from process measurements and model. The role of state estimators is to understand the complexity of the state of the system and thereby, use different filtering and smoothing techniques available in hand to estimate the system. For instance, estimation done to predict the future is called filtering and estimation done to retrace the past is called smoothing. This thesis mainly works on estimation through filtering techniques. One of best known examples to give a hint on filtering is weather predictions. Prediction of weather for a future time is done through one of the filtering techniques.

When the description of system is known, either linear or nonlinear, the state estimator needs to estimate the system that will minimize the error between the true state and the estimated state. This leads to the optimal estimation problem which is solved by the Kalman filter. The Kalman filter(KF) has been implemented in literally thousands of applications since its inception in the early 1960s. It was found and named after R.E. Kalman. This was the first step in filtering applicable for linear systems, which further led to many other filtering techniques applicable for nonlinear

systems.

This thesis covers some basic information on some of the existing nonlinear filters and focuses on Moving Horizon Estimation(MHE).

CHAPTER II

Literature Review

State estimation is an active research field having a wide range of application. There are many state estimation techniques and algorithms one of which is Bayesian state estimation which is based on probabilistic approach.

Bayesian state estimation is an important method, of all the estimation techniques, because of the following reasons [1]:

- They preserve information as they are based on the probability axioms
- They give the probability density function (pdf) of the state conditioned on the available observations or measurements

With the available pdf, the state of the system can be estimated along with the uncertainties. One of the earliest Bayesian state estimation algorithm was for linear systems, which is known as the Kalman filter. It was developed by Kalman and Bucy in 1960 [2]. Since then, the KF has had a wide range of applications and was always a subject for research and analysis. The KF is a set of mathematical equations through which the state of the process can be estimated recursively. The estimation of the

past, present and future states of the process are obtained using the filter. However, one limitation is that, KF is applicable only for linear systems.

Later, several modifications were made on the KF technique in order to make it applicable for nonlinear systems. One of the modified estimation techniques is called the extended Kalman filter(EKF) [3]. The EKF is a widely used Bayesian state estimation algorithm for nonlinear systems. However, it has its limitations. It is only reliable for systems that are almost linear [5], [7]. Chemical engineering systems are always highly nonlinear, hence other novel methods have been used in place of the EKF. The unscented Kalman filter(UKF) and particle filter(PF) are examples of state estimation techniques that are applicable for nonlinear systems of higher order [7], [8], [10]. State estimation for nonlinear dynamic systems is still an active research area.

Moving horizon estimation is an efficient method for state estimation for constrained, linear and nonlinear systems. MHE has gained a lot of interest because it is proved to be performing superior to traditional state estimation filters such as EKF [13]. The advantage of MHE is that it handles complex nonlinear dynamic models. The disadvantage is that it requires on-line solutions of dynamic optimization problems which results in computational delays [15].

MHE minimizes the estimation cost function defined on a moving window, which involves a finite number of time steps. The cost function comprises of two parts: a stage cost and an arrival cost. Initially, an approach was proposed which involves the numerical solution of the measurement observation problem based on Newton's method [16]. Similar optimization based techniques were developed [17] and [18] for continuous time dynamic systems. MHE estimation for nonlinear systems under discrete time intervals was developed by Michalska and Mayne [19]. Recently, advancements have been made for MHE in linear, nonlinear and hybrid systems.

The constraints on the system are taken into account and the solution for nonlinear programming are obtained at each time step [20]. This approach requires exact on-line minimization of a nonlinear cost function. The possibility of practical applications are less. This is the main drawback of MHE.

The unscented Kalman filter, particle filter and cell filter(CF) are the three different sampling methods that were proposed for obtaining the arrival cost for MHE [21]. Instead of approximating the arrival cost using a Gaussian assumption, another method using the numerical approximation of the state probability density function provided by the PF and CF are considered. It is shown that, the arrival cost parameters can be accurately computed and updated by sampling based methods without using linearization. The Gaussian assumption is replaced by kernel density estimation [21].

In this research thesis, the arrival cost of the MHE is computed using the dynamic programming . Dynamic programming is a recursive method for solving sequential decision problems, which is used to find optimal decisions. It is also known as backward induction. A number of researchers have worked on this topic, especially in the field of economics. R. Bellman [22] is one of the most credited researcher who identified the common structure underlying the sequential decision problems and proved the use of backward induction in solving the sequential decision problems with uncertainty. He is the person who defined backward induction in a new term called dynamic programming. The use of dynamic programming in computing the arrival cost of MHE is explained in the following chapters.

CHAPTER III

Scope of Thesis

3.1 Aim

The scope of this thesis is to improve the already existing method of optimization strategy in moving horizon estimation. The main idea behind this strategy is to improve the stability of arrival cost estimation, computational time and Mean Squared Error(MSE) value which will be later discussed.

3.2 Hypothesis

One of the optimization strategies that have been used often in moving horizon estimation is an inbuilt matlab nonlinear programming (NLP) algorithm. Using this algorithm, the optimization of the cost function takes place by which we evaluate the estimate. In order to enhance or improve the optimization strategy, it is hypothesized that dynamic programming algorithm can be used in place of the already existing optimization strategy.

3.3 Specific Aims

Aim 1 : To explain moving horizon estimation through NLP

A clear understanding of MHE will be achieved in the following chapters. This involves initialization of MHE through different nonlinear filters, arrival cost estimation and optimization. Focus will be made on the arrival cost estimation and optimization. An algorithm will be exhibited which will further be implemented in several mathematical examples.

Aim 2 : To implement dynamic programming in moving horizon estimation

The concept of dynamic programming will be clearly explained in regards to certain mathematical examples. An algorithm for dynamic programming will be explained. The concept of dynamic programming will be implemented in the optimization of cost function for MHE. The new MHE algorithm will be explained.

Aim 3 : To provide an example and make a detailed analysis

A mathematical example will be presented. The modified MHE algorithm which implements the concept of dynamic programming will be explained through the example. A comparison between the original MHE and the modified MHE will be made and a detailed analysis will be presented. The advantages and disadvantages of one algorithm over the other will be critically analyzed.

CHAPTER IV

Layout

The different chapters in this thesis are laid out as follows:

In chapter 5 the concept of state estimation is introduced and a brief description about the state and measurement of the system are explained.

Chapter 6 gives an introduction to different linear and nonlinear filters. The underlying phenomenon behind the development of these filters is explained. A general problem statement from which the state and measurement are derived is explained with respect to both linear and nonlinear system. Kalman filter, extended Kalman filter, unscented Kalman filter and particle filter are some of the filters explained in detail.

Chapter 7 deals with the topic of interest in the thesis, moving horizon estimation, an optimization filtering technique used in nonlinear systems. It also explains sub topics like initialization strategies, arrival Cost and optimization. The method of least squares is introduced.

Chapter 8 explains dynamic programming an optimization technique. This chapter also explains principle of optimality and its algorithm. Also, certain basic

examples in order to understand the optimization strategy are provided. Finally its advantages and disadvantages are explained.

In chapter 9 a mathematical nonlinear system is considered as an example. The performance of the nonlinear filters with respect to moving horizon estimation are illustrated with plots and performance curves.

In chapter 10 the concept of dynamic programming is implemented in MHE and the performance of existing and modified algorithm is explained through the mathematical example. A comparison is made between those two algorithms extensively.

Chapter 11 is the conclusion of the thesis in which, the extent to which each aim in the scope of the thesis is achieved, is explained in detail.

CHAPTER V

State Estimation

State estimation is a branch of systems engineering, that deals with estimating the values, based on measured/empirical data that has a random component. Observations are not always predictable but they are distributed at random. In estimation theory, one aims to guess the underlying distribution of random observations from the data. In particular, the known measurements model of the system is used to obtain the estimate. A point estimate gives a good approximation for the true value.

State estimation determines the current state of a complex system such as location of a spacecraft, temperature, batch reactors, robotics, given, the observations from the system sensors. In the past, state estimation has always been used in diagnosis, detecting and identifying faults when they occur, but, safe and effective autonomous control of system requires estimating all aspects of system state. In addition, estimating continuous system parameters has also become increasingly important.

State estimation is critical for a number of reasons: accurate state estimates make way for much easier control of states of the system, and allow selection of better

control aided actions. Finally, state estimation can provide prognostic information, identifying components or systems that are likely to fail soon and should be repaired, replaced, changing density, volume etc.

A key aspect of state estimation is that it is rarely certain. There is inevitably some ambiguity in the sensor data received from a system, and it is of great use to have a state estimate that represents this uncertainty explicitly [4]. This is for several reasons: firstly, a probability distribution representing the uncertainty can summarize all the measurements or observations received by the state estimator so far, making it easier for the state estimate to update. Secondly, this probabilistic representation is of use in decision making by allowing the effects of planned future actions to be evaluated in states that have low probability rather than only in the most likely state. Finally, probabilistic information is of use for prediction and maintenance, providing information about state of the system involved at the necessary point.

Before going in detail to model formulation a brief introduction on probability: Probability is the estimation of occurrence of an event based on its likelihood/chance. The value of probability of any event is between 0 and 1. The higher the degree of probability, the higher is the chance of event happening. The important probability principle which is used a lot in filtering techniques is conditional probability. Conditional probability is occurrence of event A given the occurrence of other event B. It is mathematically shown as,

$$p(A|B) = \frac{p(B|A)p(A)}{p(B)} \tag{5.1}$$

The problem statement of state estimation is given as a time variant of measurement and dynamic model of the system. What is the most likely state of the system given these measurements and system model? The problem here is usually

formulated probabilistically, that is, calculated as,

$$\hat{x}_{k|k} = arg\max_{x_k} p(x_k|y_0, ..., y_k) \tag{5.2}$$

in which x_k and y_k are the state and measurement respectively, at time t_k, and, $\hat{x}_{k|k}$ is the *a posteriori* state estimate of x at time t_k, given all measurements till time t_k.

There are many systems. One is linear, unconstrained, with additive Gaussian noise that can be solved using the Kalman filter which provides a closed-form solution to Eq. 5.2. The other one is constrained, nonlinear for which the solution can be arrived through nonlinear filters. For addressing the nonlinear system there are many filtration techniques like extended Kalman filter, unscented Kalman filter, particle filter, moving horizon estimation, etc.,

CHAPTER VI

Linear and Nonlinear Filters

6.1 Introduction

Estimation of the state of the system from noisy measurements is a necessary element in model-based applications. Linear filters process time-varying measurements to evaluate the state of the system, subject to the constraint of linearity. This results from system composed models or algorithms classified as having a linear models which is expressed in the form of ordinary differential equations (ODE). Most filters implemented in analog electronics, in digital signal processing, or in chemical mathematical systems are classified as causal, time invariant, and linear.

The state of a system, for example, a sample material is defined by specifying the values of all the variables describing the system. If the system is a sample of a pure substance this would mean specifying the values of the temperature, pressure, volume, and the number of moles of the substance. Consider a batch reactor, where the state of the system is the mole fraction of components involved in the reactor. The measurements or observations are the function of state of the system that are

measured through control devices. The state of the system is evaluated from the measurements. For example, in the batch reactor process, the temperature is the measurement variable in the system. The system is described as a mathematical model. The state and measurement of the system are described in the model as an ordinary differential equation, with mass and energy balances of the system taken into account. The error that is added to the system model is called the noise variance.

The general concept of linear filtering is also used in statistics, data analysis, and chemical engineering among other fields and technologies. This includes non causal filters and filters in more than one dimension such as used in image processing; those filters are subject to different constraints leading to different design methods. Linear systems with Gaussian noise can be evaluated through Kalman filter an optimal estimate filter. Given, the knowledge about distributions of the initial state, disturbance, and measurement noise, the Kalman filter provides a recursive solution to the real-time, minimum-variance estimation problem. But, in general, not all systems that exist in reality are linear. Majority of the systems are nonlinear. Kalman filter, as such, can only be used for linear constrained and unconstrained systems. Therefore, a necessity for solving nonlinear system arises and the Kalman filter was improvised, in order to estimate the nonlinear systems.

Nonlinear filters have many applications, especially in the removal of certain types of noise that are not additive. Indeed, all radio receivers use nonlinear filters to convert kilo to giga-hertz signals to the audio frequency range; and all digital signal processing depends on nonlinear filters. However, nonlinear filters are considerably harder to use and design than linear ones, because the most powerful mathematical tools cannot be used on them. Considering the processes today, majority of them are nonlinear systems. The need for understanding the nonlinear systems are essential in order to estimate them using nonlinear filtering theory.The performance of these

nonlinear filters are entirely dependent on approximations made through development of Kalman filter. The approximations made on the linear filtering theory is always accompanied by some degree of uncertainty. Some examples of the nonlinear filters are extended Kalman filter and unscented Kalman filter which are a result of the approximations made on Kalman filter. In case of the extended Kalman filter the nonlinear functions are linearized using Jacobian [6]. In unscented Kalman filter the choice of sigma points are made over the pdf are used [9]. These nonlinear filters use mean as their estimate and it is called minimum *a posteriori estimate*.

In case of a particle filter the estimate can either be the mean median or mode of the probability density function. Here a set of particles are sampled out of the pdf, whose initial mean and covariance are known [11]. This filter works on the principle of importance density. There are special cases of particle filter one of which is bootstrap filtering where, the transition prior is assumed to be the importance density [10]. All these filters work on the principle of Bayesian estimation.

6.2 Bayesian State Estimation

The roots of Bayesian state estimation lie in the Bayes theorem. The Bayesian estimation is widely used and powerful among state estimation because they are rigorously based on the probability axioms and therefore preserve information.

The Bayes theorem:

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)} \tag{6.1}$$

Consider x to be a model variable - for example the state of the dynamic system like concentration, partial pressure and mole fraction, y to be observed variable - for example the output from sensor also called measurement like pressure, temperature, pH [1].

- $p(x)$ is the probability distribution of the state of the system which is independent of the measurement. This is called the *prior* of x.

- $p(y)$ is the probability distribution of the measurement of the system which is called the marginal probability or the evidence. It is generally known as a normalization factor.

- $p(x|y)$ is the probability distribution of the state of the system given the measurements in hand. This is termed as the *posterior* of the system. This is the estimate of the system which is under consideration.

- $p(y|x)$ is the likelihood of the system based on the condition that the given model is true.

6.3 Problem Statement

Consider the state of a dynamic system model, [21]

$$x_{k+1} = f(x_k) + w_k \tag{6.2}$$

where f is a linear or nonlinear function given the system model, x_k is the state of the system for any time k and w_k is the state noise vector distributed according to Gaussian probability density function $N(0, Q)$. The measurement/observation is given by

$$y_{k+1} = h(x_{k+1}) + v_{k+1} \tag{6.3}$$

where h is a linear or nonlinear function given the observation model of the state of the system, y is the measurement/observation of the system model and v_k is the measurement noise vector distributed according to the Gaussian probability density function $N(0, R)$. The initial conditions or the initial probability density function

of the state vector is a Gaussian probability density function $N(\hat{x}_0, \hat{P}_0)$ given the fact there are no measurements yet. The $\hat{x}_0$ is the mean at time $k = 0$,$\hat{P}_0$ is the initial covariance at time $k = 0$. The problem statement mentioned here is general, and is applicable for all above known filters. The process noise covariance Q and measurement noise covariance R change with time but here it is considered to be constant.

6.4 Linear Filter - Kalman Filter

When the description of system is linear, the state estimator needs to estimate the system that will minimize the error between the true state and the estimated state. This leads to the optimal estimation problem which is solved by the Kalman filter. Kalman filter was found it 1960 and named after R.E.Kalman. The Kalman filter has wide range of applications. The optimal estimate filter existing until date for all linear systems is Kalman filter. The optimal estimate infers parameters of interest from indirect, inaccurate and uncertain observations. It is recursive so that new measurements can be processed as they arrive [2]. The Kalman Filter addresses the general problem of estimating the state x, for a discrete time varying linear model, which are expressed in the form of difference equations

$$x_{k+1} = Ax_k + w_k \tag{6.4}$$

$$y_{k+1} = Hx_{k+1} + v_{k+1} \tag{6.5}$$

From this it is understood that these equations resemble the problem statement. The two functions f and h are linear and hence represented as constants, multiplied to the the state vector. The noise variables w and v are additive. Assuming noise is Gaussian, the Kalman filter minimizes the mean square error of the estimated parameters.

The Kalman filter works in two steps:

- The current state and error noise covariances are used to project forward through the state model in order to estimate the predicted mean and covariances. This is called as the *a priori* estimate.
- When the measurement comes in, incorporating them back in the *a priori* estimate results in the *a posteriori* estimate.

Thus, the estimation algorithm resembles that of *predictor - corrector* algorithm. The predict phase is defined by the time update equations where the state and error noise covariance are projected forward.

$$\hat{x}_{k+1}^{-} = A\hat{x}_k \tag{6.6}$$

$$P_{k+1}^{-} = AP_kA^T + Q \tag{6.7}$$

After projecting forward, the *a priori* estimate is obtained and using this measurement update equations are updated.

$$\hat{x}_{k+1} = \hat{x}_{k+1}^{-} + K_{k+1}(y_{k+1} - H\hat{x}_{k+1}^{-}) \tag{6.8}$$

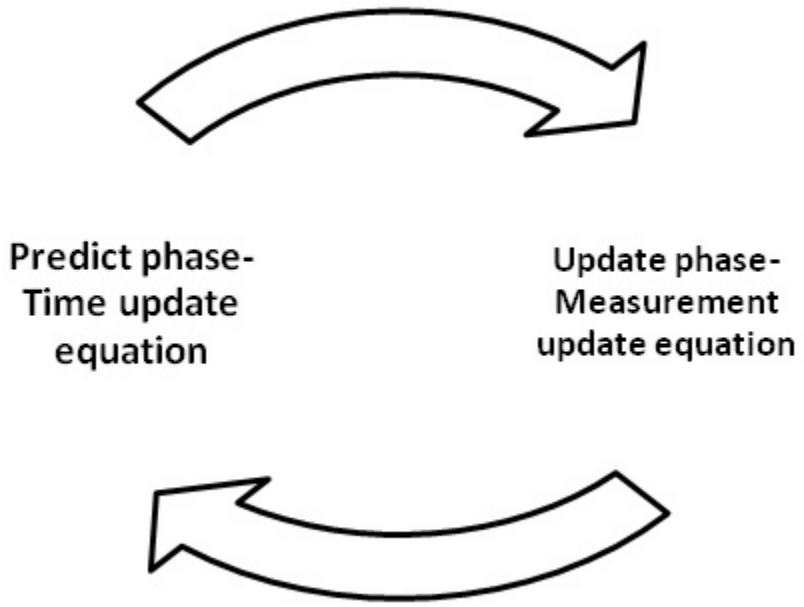
$$P_{k+1} = (I - K_{k+1}H)P_{k+1}^{-} \tag{6.9}$$

$$K_{k+1} = P_{k+1|k}H^T(HP_{k+1|k}H^T + R)^{-1} \tag{6.10}$$

where K is the Kalman gain which minimizes the *a posteriori* error covariance, I is the identity matrix and P_{k+1} is the error covariance. Fig. 1 shows the recursive solution that follows for given time steps.

The importance and reliability of Kalman filter is based on the good results in practice due to optimality and structure. It is one of the convenient form for online real time processing.

Figure 1: Kalman Filter Cycle



6.5 Nonlinear Filter - Extended Kalman Filter

When the system model is nonlinear, linear filters fails to perform therefore several modifications were made on the linear filter algorithm to adapt to nonlinear models. One of the first basic nonlinear filter which works on Bayesian state estimation is extended Kalman filter. It is similar to the Taylor series expansion, linearizing the nonlinear system through partial derivatives or Jacobian. Thus, the problem statement, in Eq. 6.2 and Eq. 6.3, containing the function f and h are linearized using the Jacobian [6]. The same algorithm as in the Kalman filter is followed after the approximations made on the nonlinear equations.

The state and measurement model are the same as mentioned in the problem statement, in Eq. 6.2 and Eq. 6.3, where, the function f and h are nonlinear functions on the the state vector. Since it is similar to Kalman filter, there are two phases,

prediction phase and correction phase. Prediction is done through process model,

$$\bar{x}_{k+1} = f(\hat{x}_k) \tag{6.11}$$

$$\bar{P}_{k+1} = F_k P_k F_k^T + Q \tag{6.12}$$

Here F_k is the Jacobian of the function of the state equation. This is also called as the linearization of the process model at $\hat{x}_k$. After projecting forward the *a priori* estimate is obtained, using which the update is done through measurement update equations. The update phase is as shown below:

$$\hat{x}_{k+1} = \hat{x}_{k+1}^- + K_{k+1}(y_{k+1} - H(\hat{x}_{k+1}^-)) \tag{6.13}$$

$$P_{k+1} = (I - K_{k+1}H)P_{k+1}^- \tag{6.14}$$

The Kalman gain is given as,

$$K_{k+1} = P_{k+1|k}H^T(HP_{k+1|k}H^T + R)^{-1} \tag{6.15}$$

The linearization of the measurement model through Jacobian of function H takes place in the update stage.

6.5.1 Advantages and Disadvantages

Unlike Kalman filter, extended Kalman filter is not the optimal estimator. It is optimal only when the measurement and state transition model are linear. The extended Kalman filter is one of the standard techniques in nonlinear estimation with wide range of applications. It estimates the state of a nonlinear dynamic system, also, it estimates parameters for nonlinear system identification like learning the weights of a neural network, and dual estimation like the expectation maximization algorithm where estimation of both state and parameter takes place simultaneously [8]. Here, in this thesis, only state estimation is considered.

When propagated through the first-order linearization of the nonlinear system, the state function is approximated. This can cause large errors depending on the nonlinearity of the system model. The large errors are on the posterior mean and covariance in the transformed function, which results in sub-optimal performance and divergence of the filter. Thus leading to one of the major flaws of linearizing the nonlinear system. The next nonlinear filter, unscented Kalman filter, addresses this problem.

6.6 Nonlinear Filter - Unscented Kalman Filter

When nonlinearity of state and measurement models are higher, at some instances, extended Kalman filter fails. The unscented Kalman filter(UKF) addresses these problems. The UKF linearizes a nonlinear function of a random variable through a linear regression between $2n + 1$ points drawn from the prior distribution of the random variable. This technique tends to be more accurate than Taylor series linearization [8].

The state distribution of extended Kalman filter is propagated analytically through the first-order linearization of the nonlinear system, due to which, the posterior mean and covariance could be corrupted. UKF overcomes this problem by using a deterministic sampling approach.

Unscented Kalman filter works on unscented transformation. When nonlinear transformation takes place, unscented transformation is the method to calculate the statistics of a random variable in which transformation occurs [7]. The mean and covariance of the system are propagated through the state and measurement model. Eq. 6.2 and Eq. 6.3 are the problem statements which are mentioned earlier. From the given data, $2n + 1$ sigma points (n is the size of the state vector), called $\mathcal{X}_i$, are deterministically selected along with their associated weights W_i. The sigma points

are obtained from,

$$\mathcal{X}_0 = \hat{x}_0 \tag{6.16}$$

$$\mathcal{X}_i = \hat{x}_0 + (\sqrt{(n+\lambda)P_x})_i \qquad i = 1, ..., n \tag{6.17}$$

$$\mathcal{X}_i = \hat{x}_0 - (\sqrt{(n+\lambda)P_x})_{i-n} \qquad i = n+1, ..., 2n \tag{6.18}$$

The associated weights are evaluated as,

$$W_0 = \lambda/(n+\lambda) \tag{6.19}$$

$$W_0^{(m)} = \lambda/(n+\lambda) + (1 - \alpha^2 + \beta) \tag{6.20}$$

$$W_i^{(c)} = W_i^{(m)} = 1/\{2(n+\lambda)\} \qquad i = 1, ..., 2n \tag{6.21}$$

where, λ is a scaling parameter, given as, $\alpha^2(n+\kappa) - n$. α is the spread of the sigma points which is a small positive number in general. κ is another scaling parameter set to 0. β is equal to 2 in case of Gaussian distributions. These sigma points are also propagated through the nonlinear function. From this,the mean and covariance which are approximated using a weighted sample mean and covariance of the posterior sigma points are obtained.

$$\hat{y} \approx \sum_{i=0}^{2n} W_i^{(m)} \mathcal{Y}_i \tag{6.22}$$

$$P_y \approx \sum_{i=0}^{2n} W_i^{(c)} \{\mathcal{Y}_i - \hat{y}\}\{\mathcal{Y}_i - \hat{y}\}^T \tag{6.23}$$

The Kalman filters update phase follows, where the Kalman gain and the *a posteriori* estimate are determined.

6.6.1 Predict phase

The predict phase includes two steps : one is choosing of sigma points and the other is propagation of sigma points.

$$\hat{x}_{k|k-1} = \sum_{i=0}^{2n} W_i^m X_{k|k-1}^i \tag{6.24}$$

$$P_{k|k-1} = \sum_{i=0}^{2n} W_i^c [X_{k|k-1}^i - \hat{X}_{k|k-1}][X_{k|k-1}^i - \hat{X}_{k|k-1}]^T \tag{6.25}$$

6.6.2 Update phase

The final stage is updating the sigma points. With the mean and covariance of the measurement noise given, the sigma vectors are propagated through the nonlinear function.

$$\gamma_k^i = h(X_{k|k-1}^i), i = 0, ...2n \tag{6.26}$$

To get the predicted measurement and covariance,

$$\hat{Y}_k = \sum_{i=0}^{2n} W_i^m \gamma_k^i \tag{6.27}$$

$$P_{y,k} = \sum_{i=0}^{2n} W_i^c [\gamma_k^i - \hat{Y}_k][\gamma_k^i - \hat{Y}_k]^T \tag{6.28}$$

The state and the measurement cross covariance matrix is,

$$P_{x,k} = \sum_{i=0}^{2n} W_i^c [X_{k|k-1}^i - \hat{X}_{k|k-1}][\gamma_k^i - \hat{Y}_k]^T \tag{6.29}$$

The Kalman gain is given as,

$$K_k = P_{y,k} y_k P_{y,k}^{-1} \tag{6.30}$$

$$\hat{X}_{k|k} = \hat{X}_{k|k-1} + K_k(Y_k - \hat{Y}_k) \tag{6.31}$$

$$P_{k|k} = P_{k|k-1} - K_k P_{y,k} K_k^T \tag{6.32}$$

The UKF is a faster algorithm as compared to EKF which involves lot of derivatives and reduces error in mean evaluation [9].

6.7 Nonlinear Filter - Particle Filter

Particle filter follows sequential Monte Carlo methods. A particular number of samples are chosen from the probability density function with initial mean and covariance. The samples are propagated through the system model, that is, the state and measurement equation Eq. 6.2 and Eq. 6.3, which are already shown in the problem statement. The assumption made in particle filtering is that a set of N samples and its corresponding weights represent the posterior probability density function.

There are many particle filter algorithms, but the most basic one for understanding and also the easiest algorithm for application purpose is the bootstrap particle filtering approach. For any particle filter there are two basic steps that needs importance: importance sampling and resampling [11].

Estimation of properties of a particular probability density function, by generating samples from a different probability density function is called importance sampling [10]. Consider $p(x)$ as a probability density function from which it is difficult to draw samples. Consider another density $q(x)$ which is easily sampled, on the condition $p(x) \propto q(x)$. The N samples are drawn from another density $q(x)$, which is called the *importance density*. The associated weights, are given as

$$w^i \propto \frac{p(x^i)}{q(x^i)} \tag{6.33}$$

where, $q(x^i)$ is the importance density [10]. The posterior density becomes:

$$p(x) \approx \sum_{i=1}^{N} w^i \delta(x - x^i) \tag{6.34}$$

The next time step is evaluated based on samples in hand. The weights are updated by,

$$w_k^i \propto \frac{p(x_{0:k}^i | y_{1:k})}{q(x_{0:k} | y_{1:k})} \tag{6.35}$$

The importance density is chosen such that,

$$q(x_{0:k}|y_{1:k}) = q(x_k|x_{0:k-1}, y_{1:k})q(x_{0:k-1}|y_{1:k-1}) \tag{6.36}$$

The posterior density can be calculated as,

$$p(x_{0:k}|y_{1:k}) = \frac{p(y_k|x_{0:k}, y_{1:k-1})p(x_{0:k}|y_{1:k-1})}{p(y_k|y_{1:k-1})} \tag{6.37}$$

$$= \frac{p(y_k|x_k)p(x_k|x_{k-1})}{p(y_k|y_{1:k-1})} \times p(x_{0:k-1}|y_{1:k-1}) \tag{6.38}$$

$$\propto p(y_k|x_k)p(x_k|x_{k-1})p(x_{0:k-1}|y_{1:k-1}) \tag{6.39}$$

The weight update equation is calculated as:

$$w_k^i \propto \frac{p(y_k|x_k^i)p(x_k^i|x_{k-1}^i)p(x_{0:k-1}^i|y_{1:k-1})}{q(x_k^i|x_{0:k-1}^i, y_{1:k})q(x_0 : k-1)^i|y_{1:k-1})} \tag{6.40}$$

$$= w_{k-1}^i \frac{p(y_k|x_k^i)p(x_k^i|x_{k-1}^i)}{q(x_k^i|x_{0:k-1}^i, y_{1:k})} \tag{6.41}$$

In case of a bootstrap particle filter, the transition density is considered to be the importance density, one kind of an assumption in choosing the importance density [10]. The assumptions are:

$$q(x_k|x_{0:k-1}, y_{1:k}) = q(x_k|x_{k-1}, y_k) \tag{6.42}$$

Thus weights are:

$$w_k^i \propto w_{k-1}^i \frac{p(y_k|x_k^i)p(x_k^i|x_{k-1}^i)}{q(x_k^i|x_{k-1}^i, y_k)} \tag{6.43}$$

Thus the posterior density:

$$p(x_k|y_{1:k}) \approx \sum_{i+1}^{N} w_k^i \delta(x_k - x_k^i) \tag{6.44}$$

The choice of an importance density is very important and must be chosen in such a way that it minimizes the covariance.

After importance sampling knowledge, knowledge on resampling is required. There are many resampling techniques like stratified resampling, residual resampling, etc. The samples are taken from a distribution. So, depending on the original probability density function some might fall out of contention. So, as long as there are samples that are not part of the distribution there exists higher probability of *degeneracy* [11]. This situation tells that at infinite time steps there will be some samples losing weights and after n iterations there is a possibility of just one sample to remain. When the samples are not important they need to be removed and replaced by samples with larger weights. Particle filter completely depends on the choice of samples. What resampling does is, removing the samples with less weight and replicating the samples with larger weight according to the importance weights. The Fig. 2 shows the importance of choosing the right important density. Poor choice of importance density may lead to little or no overlap of the transition prior with the likelihood.

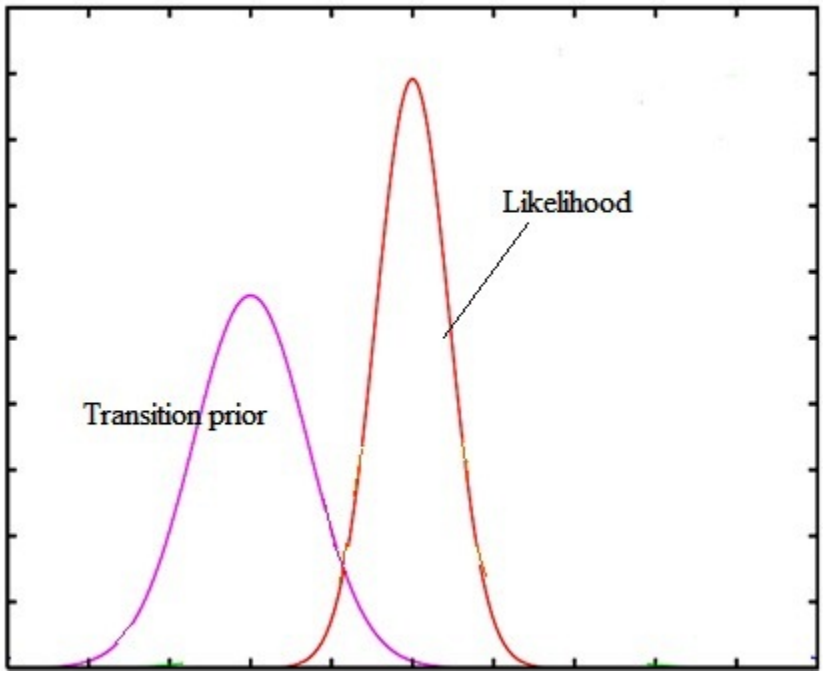
In general, the algorithm followed in particle filter can be explained in three steps:

1. Initialization phase - Consider a set of N random samples/particles $\{x^i_{k-1} : i-1, ..., N\}$ from the conditional probability density function: $p(x_{k-1}|y_{1:k-1})$

2. Predict phase - propagation of N values $\{v^i_{k-1} : i = 1, ..., N\}$, using the density function of process noise v_{k-1}, generation of new sample points $\{x^i_{k|k-1} : i = 1, ..., N\}$ using: $x^i_{k|k-1} = f(x^i_{k-1}, v^i_{k-1})$

3. Update phase - assign each a weight of $x^i_{k|k-1}$ with the measurement y_k, the calculated weight: $w^i_k = \frac{p(y_k|x^i_{k|k-1})}{\sum_{i=1}^{N} p(y_k|x^i_{k|k-1})}$, the posterior probability density function is $p(x_k|y_k) = \sum_{i=1}^{N} w^i_k \delta(x_k - x^i_{k|k-1})$

Figure 2: Poor choice of importance density: little overlap between transition prior and likelihood



There are other nonlinear state estimation technique like moving horizon estimation and cell filter etc. This thesis is mainly focused on moving horizon estimation which is discussed in the next chapter.

CHAPTER VII

Method of Least Squares and Moving Horizon Estimation

7.1 Background

As explained in literature review, some of the filters that were developed earlier are Kalman filter, extended Kalman filter, unscented Kalman filter, particle filter. One of the latest and robust technique is moving horizon estimation. MHE is formulated from least squares estimation. The nonlinearity and prior knowledge of measurements can be addressed by considering them as a least squares problem, which is how MHE has been formulated. This makes MHE a stable and reliable mode of approach towards nonlinear systems.

The current challenges in state estimation are handling nonlinear system dynamics, allowing non-Gaussian distributions for noise parameters and constraints. MHE is one filter that can cope with all these challenges.

7.2 Least Squares

Least squares is a mathematical solving technique which is used in cases where, the number of unknown parameters in a set of equation is more than the number of equation itself. The method of least squares when applied to general state estimation problems offers the advantages of being able to incorporate nonlinear models. In case of MHE, as the number of measurements grows, the size of the optimization increases too. The least squares objective can be modified to employ the fixed size fixed size moving window.

7.3 Introduction to MHE

Moving horizon estimation is one of the nonlinear filters which follows mode estimate or maximum *a posteriori* estimate. As the name suggests the estimation is done based on a horizon that is propagated through time steps. A fixed horizon of measurements are taken into consideration at the beginning. The fixed horizon represents a particular horizon chosen of any length, given the time steps. This horizon moves in time for the next data set of points by discarding the past measurement and taking in new measurement. This follows for complete set of data. In each data set of the horizon, the optimization of the cost takes place where estimate is obtained. This is done through mode estimation.

Moving horizon estimation is an optimization approach to state estimation. It is one of the nonlinear filtering techniques that have been evolved recently and used for on-line estimation. All filters that were explained in the previous chapters are minimum variance estimate, meaning, the estimate depends on the mean of the pdf where as moving horizon estimation depends on the mode of the pdf, this is called as maximum *a posteriori* estimate [15]. The estimation is done as follows:

- Initialization of moving horizon estimation is done through any of the existing nonlinear filters. The nonlinear filters predict the mean and covariance of the system model, given, its state and measurement and also the initial conditions. Using the predicted mean and covariance the MHE cost function can be evaluated.

- The cost function of the MHE consists of two parts arrival cost and stage cost. Arrival cost can also be called as a penalty term and it is the error caused in the predicted mean and covariance. Depending on the nonlinear filter used it varies and it causes poor initialization to optimization. Stage cost is evaluated based on the noise covariance in the state and measurement models.

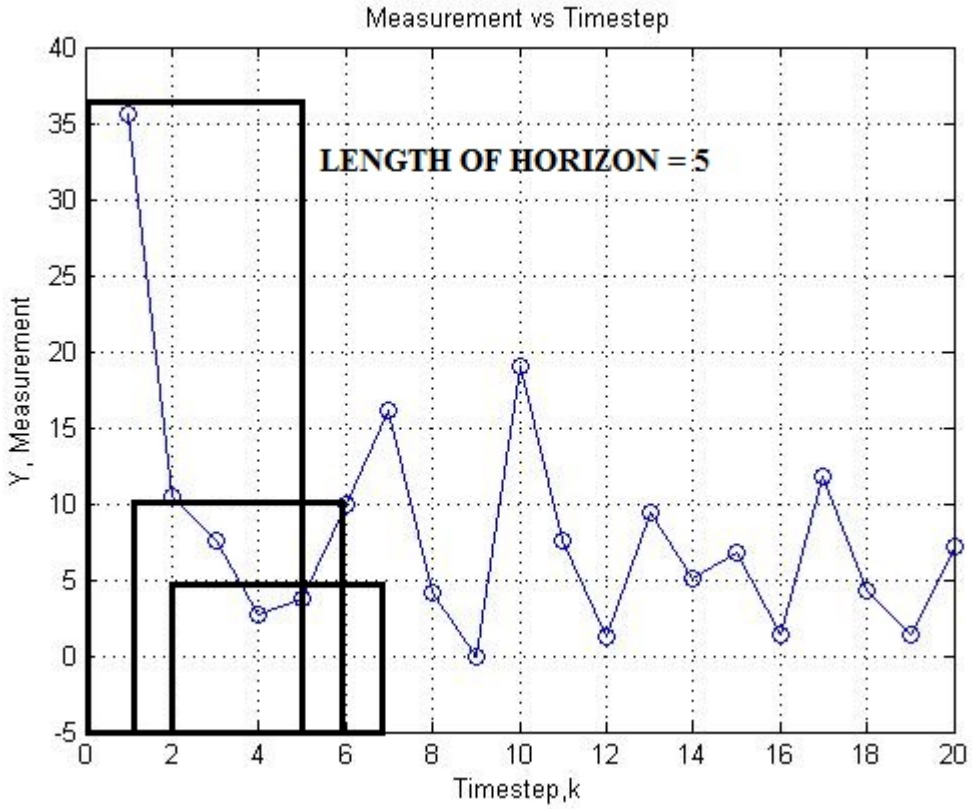
- The optimization of the cost function or the objective function, through nonlinear programming algorithm, is used in MHE. The optimization strategy minimizes the cost and using this evaluates the *a posteriori* estimate.

The number of measurements increases with every time step, along with which the size of optimization also increases. In order to have the size of optimization constant, the least squares objective is employed for a fixed horizon, in which the number of measurements always remains a constant.

In moving horizon estimation how horizon movement takes place is that, when the first set of measurements are processed, the set is appended by discarding the earliest measurement which is shown in Fig. 3.

Approximation of conditional pdf is of importance in MHE. Assuming the conditional pdf to be uniform density is one way, as it takes into account only measurement information of the most recent horizon and the past is discarded. In order to compensate for this lack of information there is need for bigger horizon lengths. This is done because, the choice of horizon length is of at most importance to determine

Figure 3: Moving Horizon Strategy: The horizon length and the movement of the horizon window



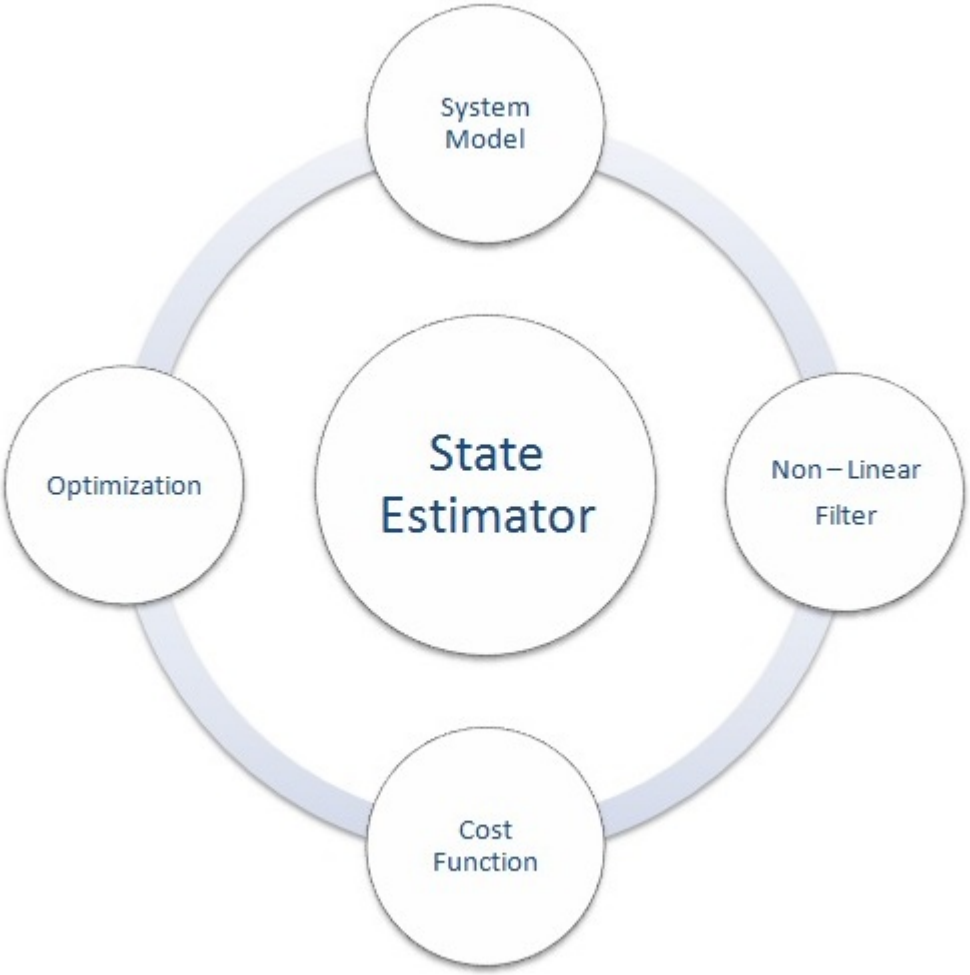
how well MHE works [15]. If the conditional pdf is inconsistent with measurement then horizon length needs to be large enough to overcome them [21].

In Fig. 3 the horizon length is considered to be 5 time steps. The movement of horizon window with respect to each time step is explained in terms of the three boxes accordingly. This also shows how the measurements are discarded at each time step.

Given the system model, that is, the state and measurement equation as specified in the problem statement, the steps involved in moving horizon estimation are specified in Fig. 4.

In Fig. 4 the first step is obtaining the system model. The nonlinear filter is initialized with the system parameters.The cost function is evaluated and finally

Figure 4: Steps in MHE Algorithm: System model propagation in the nonlinear filter, evaluating the cost function and optimization



the cost is optimized. The obtained estimate is again considered for initialization. This algorithm gives the basic approach of MHE. The state estimator, given any system model has to use any one of the approximate non-linear filter to propagate and update two parameters of the probability density function, the mean and the covariance. Using this mean and covariance evaluate the penalty term known as arrival cost from the moving horizon estimation objective function.

7.4 Cost

The two costs that are calculated for optimizing in moving horizon estimation are stage cost and arrival cost [21]. The negative logarithm of the probability density function is the cost function of the moving horizon estimation that is minimized for the state estimates,

$$\min_{x_m} \sum_{j=k-m+1}^{k} ||y_j - h(x_j)||^2_{R^{-1}} + \sum_{j=k-m+1}^{k-1} ||x_{j+1} - f(x_j)||^2_{Q^{-1}} + \Gamma(x_{k-m+1}) \qquad (7.1)$$

The summations together are called the stage costs and the last term $\Gamma(.)$ is called as the penalty term or the arrival cost. The stage cost is the uncertainty caused in the system state and measurement. The arrival cost is the uncertainty in the *a priori* estimate, that is, the mean and covariance.

7.5 Initialization of Moving Horizon Estimation

Moving horizon estimation requires other nonlinear filters in order to provide the mean and covariance. These filters are required to propagate the mean and covariance through the system and measurement equation. Thus, the predicted mean and covariance are obtained. This is important in calculating the arrival cost.

Initialization through extended Kalman filter has some concerns when the non-linearity is high due to its already explained disadvantages caused through Taylor's series expansion. The deterministic sampling method UKF uses sampled sigma points and associated weights to represent state of the system. Since linearization is avoided in UKF, significant results are demonstrated when compared to EKF. Particle filter is a sampling based approach, where the sample size is large, which is a good motivator for MHE. Also, mean and covariance converge independent of the state, which again is good for arrival cost estimation.

7.6 Arrival Cost

The most commonly used arrival cost approximation in moving horizon implementation is that *a priori* probability density function at the start of the horizon is a multi variate Gaussian, which can be represented by the first two moments. The arrival cost is expressed as

$$\Gamma(x_{k-m+1}) = ||x_{k-m+1} - \tilde{x}_{k-m+1}||^2_{\tilde{P}^{-1}_{k-m+1}} \tag{7.2}$$

To know more about how a particular nonlinear filter affects the computation of arrival cost, computing the arrival cost using extended Kalman filter is explained.

7.6.1 Arrival cost using Extended Kalman Filter

The extended Kalman filter is explained briefly in Chapter 2 from which it is understood that the mean and covariance determined from the update phase is the conditional mean and covariance. Those from predict stage are the predicted mean and covariance. Always, extended Kalman filter is prone to show divergence in estimation, which may result in poor arrival cost. Similarly,the arrival cost can be computed using other existing nonlinear filters.

7.7 Optimization

Once the predicted mean and covariance are obtained, the cost is evaluated, given the objective function, through a matlab nonlinear programming algorithm called the unconstrained algorithm. There are two matlab in-built functions, one is *fminunc* and other is *fminsearch*.

The traditional MHE algorithm is explained in this chapter. In order to optimize the cost function, a technique called dynamic programming is to be understood.

The following chapters explain dynamic programming in detail.

CHAPTER VIII

Dynamic Programming

8.1 Background to Dynamic Programming

There are some disadvantages when implementing MHE. One of those disadvantages is when using particle filter for initialization of MHE, where the cost function needs to be optimized. As the sample size of the particle filter increases, the nonlinearity increases too and degeneracy becomes an issue. In order to simplify the optimization strategy, modifications are made to the algorithm of MHE. One of the best ways is to look for new algorithms for optimization strategy. There are two possible methods, one using kernel density estimation and the other using dynamic programming. This thesis focuses on choosing dynamic programming. The reason behind the choice is the simplicity and also the convenience in optimizing the cost function.

The goal here is to find another way to optimize the approach as well as increase the sample size and decrease the computational time. Dynamic programming has been widely used for solving complex problems. The nonlinearity of the system tends

to be in those lines of complexity, whereby, the need for dynamic programming is highly essential.

Dynamic programming examines the possible ways of solving a complex problem and arrives at the best possible solution available. Dynamic programming splits the complex problem into simpler subproblems in order to evaluate the problem easily, as well as, arrive at the best possible solution.

8.2 Introduction to Dynamic Programming

Dynamic programming is an algorithm for efficiently solving a broad range of search and optimization problems, which exhibit the characteristics of overlapping subproblems and optimal substructure. A problem is said to have overlapping subproblems if it can be broken down into subproblems, which are reused multiple times. This is closely related to recursion. A problem is said to have optimal substructure if the global optimal solution can be constructed from local optimal solution to subproblems. In simpler terms, it is the method to solve complex problems by reducing them to simpler subproblems. It follows the principle of optimality where the initial decision is not important. Despite this all remaining decisions need to be optimal. Depending on the recurrence relations there are two methods: forward approach and backward approach, depending on which optimization is done in backward and forward direction respectively [22]. The dynamic programming is an optimization strategy that can be helpful in replacing the already existing optimization in moving horizon estimation. Dynamic programming can minimize as well as maximize any given function. This idea comes from the implementation of dynamic programming in least squares estimation. The cost function of moving horizon estimation is similar to the least squares. Therefore it can be implemented in optimizing the cost function in moving horizon estimation.

This method cannot be applicable for all existing moving horizon estimation because moving horizon estimation requires the nonlinear filter to evaluate the predicted mean and covariance. In order to implement dynamic programming, the samples or particles at a particular time step of the estimation along with the mean and covariance of the predict phase. So, to initialize moving horizon estimation, a nonlinear filter such as particle filter in necessary. For this method, at every time step there are particles and length of horizon gives room for structure of dynamic programming.

In order to see the difference, consider the factorial function, defined as follows

$$def factorial(n): \tag{8.1}$$

$$if n == 0 : return 1 \tag{8.2}$$

$$return n * factorial(n-1) \tag{8.3}$$

Thus, in order to calculate $factorial(n)$ calculating the subproblem $factorial(n-1)$ is necessary. This problem does not exhibit overlapping subproblems, since factorial is called, exactly once for each positive integer less than n. A problem is said to have optimal substructure if the globally optimal solution can be constructed from locally optimal solutions to subproblems. The general form of problem in which optimal substructure plays a role is as represented. Consider a collection of objects called A. For each object o in A we have a "cost", $c(o)$. The aim is to find the subset of A with the maximum (or minimum) cost, perhaps subject to certain constraints. The brute-force method would be to generate every subset of A, calculate the cost, and then find the maximum (or minimum) among those values. The more the number of elements the harder it becomes to compute the cost.

8.3 Principle of Optimality and Algorithm

The concept of principle of optimality is as follows. Whatever the initial state the remaining decisions must be optimal, with regard to the state, following from the first decision. No matter what the first decision is, if it is removed, all other remaining decisions need to be optimal. There are two approaches in solving dynamic programming problem, forward and backward approach and in order to obtain the solution it is solved in the opposite direction of the recursive problem statement. Depending on the approach, the solution algorithm can be written in four steps [22]:

- Initialization: For $1 \leq i \leq N$

$$\delta_1(i) = \log f(x_1^{(i)}) + \log g(y_1|x_1^{(i)}) \tag{8.4}$$

- Recursion: For $2 \leq k \leq t$ and $1 \leq j \leq N$

$$\delta_k(j) = \log g(y_k|x_k^{(j)}) + \max_i[\delta_{k-1}(i) + \log f(x_k^{(j)}|x_{k-1}^{(i)})] \tag{8.5}$$

$$\psi_k(j) = arg\max_i[\delta_{k-1}(i) + \log f(x_k^{(j)}|x_{k-1}^{(i)})] \tag{8.6}$$

- Termination:

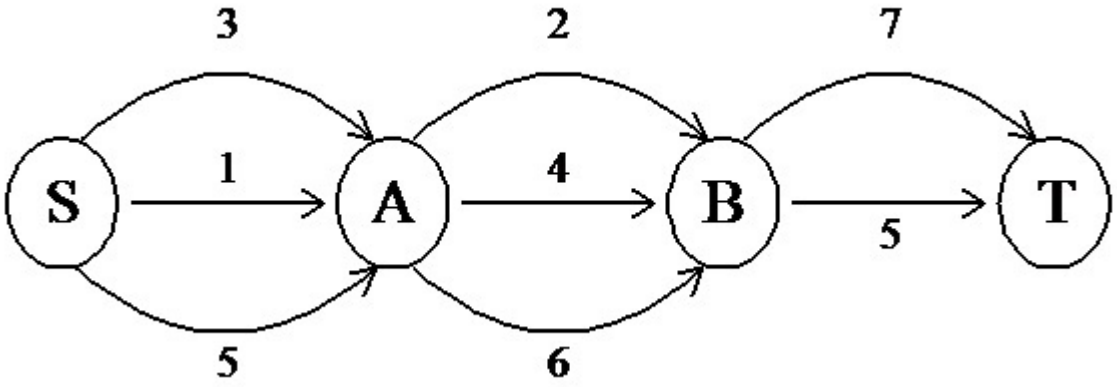
$$i_t = arg\max_i \delta_t(i)\hat{x}_t^{MAP}(t) = x_t^{i_t} \tag{8.7}$$

- Backtracking: For $k = t-1, t-2, ..., 1$

$$i_k = \psi_{k+1}(i_{k+1})\hat{x}_k^{MAP}(t) = x_k^{i_k} \tag{8.8}$$

The computational complexity of this algorithm can develop to huge orders. The optimization here can be both minimization and maximization of the given objective function.

Figure 5: Example 1: S, A, B, T are different points, the distance between them is the number shown, the shortest distance between S and T is to be found



8.4 Examples

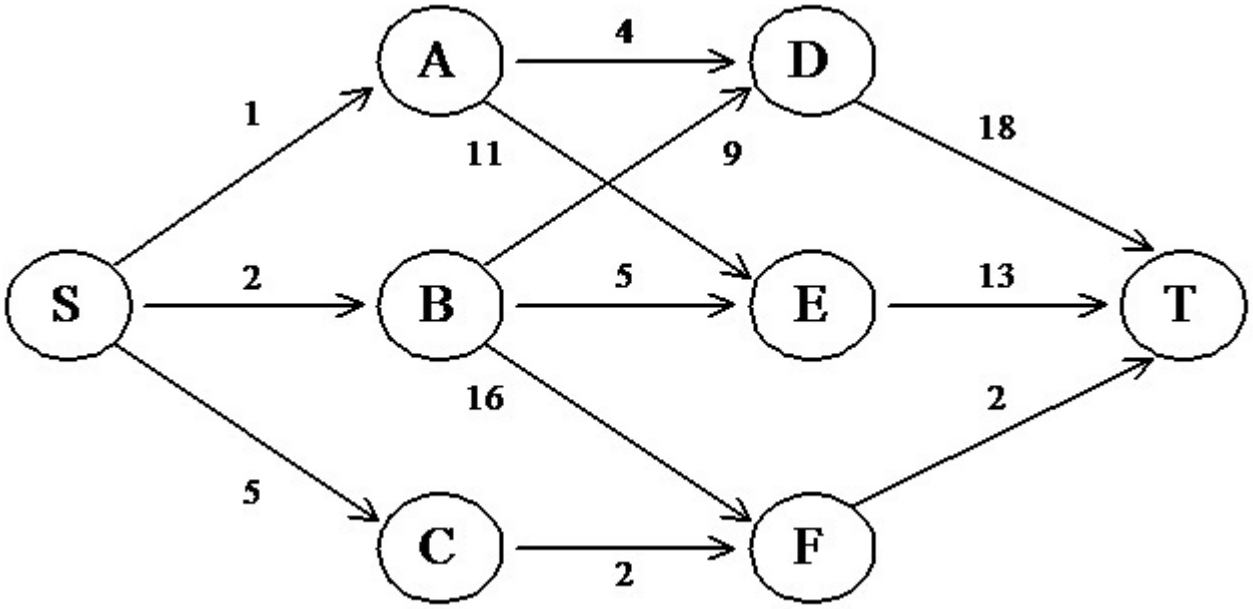
Fig. 5 is an example for implementation of dynamic programming. In the example, finding the shortest path from S to T is the goal. Out of the many ways, assuming the shortest path to be $1+4+5=10$ is usual. But the shortest path found using algorithm is $1+2+5=8$

The example in Fig. 6 can be elaborated as follows. This figure represents a traveling sales man, traveling from S to T, in the shortest path. Since it is forward approach the solution will be obtained in the backward direction. The goal of salesman is to start from point S and reach T in the shortest possible, covering all other intermediate points.

From the Fig. 7 it is deterministically shown that shortest distances from A to T, B to T, C to T defines what the shortest distance is from S to T.

$$d(S,T) = min[1 + d(A,T), 2 + d(B,T), 5 + d(C,T)] \tag{8.9}$$

Figure 6: Traveling Sales Man Problem: the shortest distance between S and T is found using DP



$$d(A,T) = min[(4 + d(D,T)), (11 + d(E,T))] \tag{8.10}$$

$$= min[(4 + 18), (11 + 13)] = 22 \tag{8.11}$$

$$d(B,T) = min[(9 + d(D,T)), (5 + d(E,T)), (16 + d(F,T))] \tag{8.12}$$

$$= min[(9 + 18), (5 + 13), (16 + 2)] = 18 \tag{8.13}$$

$$d(C,T) = min[2 + d(F,T)] = 2 + 2 = 4 \tag{8.14}$$

$$d(S,T) = min[(1 + d(A,T)), (2 + d(B,T)), (5 + d(C,T))] \tag{8.15}$$

$$= min[(1 + 22), (2 + 18), (5 + 4)] = 9 \tag{8.16}$$

Here d(A,T) is the distance from A to T. It is calculated by finding the minimum distance from A through D and E, which is explained by the Fig. 8. The d(B,T) and d(C,T) are obtained through the same minimization optimization of distance. This finally minimizes the distance from S to T. Here, the backward approach is used. Firstly, the minimum distance to reach A, B, C are obtained. Therefore,

Figure 7: Dynamic Programming Approach

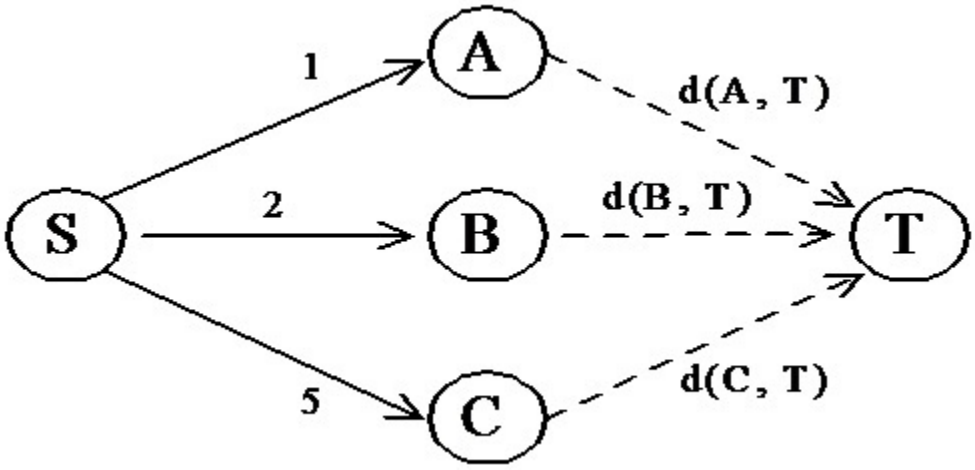
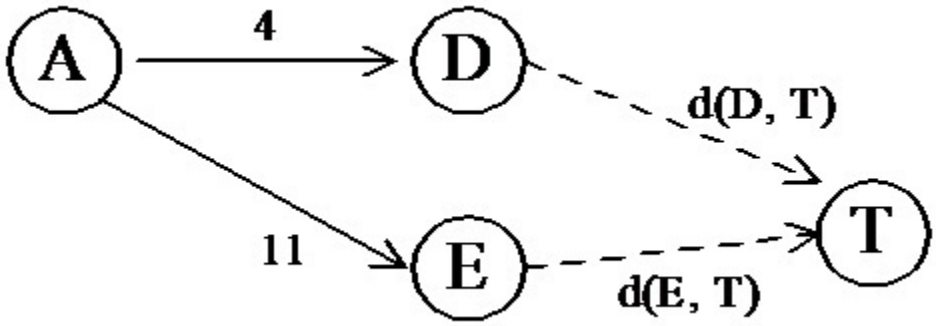


Figure 8: Dynamic Programming d(A,T)



evaluating the minimum distance from these points to S, gives the overall minimum distance between S and T.

8.5 Advantages and Disadvantages

Dynamic programming computes recurrences efficiently by storing partial results. Thus, dynamic programming can only be efficient when there are not too many partial results to compute.

There are $n!$ permutations of an n-element set. Dynamic programming cannot store the best solution for each sub permutation. (There are 2^n subsets of an n-

element set)

There are two main disadvantages of dynamic programming: one is the curse of dimensionality, as the size of the problem increases, the increase in computation complexity occurs and the other is the menace of expanding grid, by which estimation of number of variables increases, so storage of those in a computer becomes a problem.

CHAPTER IX

Examples

9.1 Introduction

To provide a concrete evidence as to how these filters work, certain examples need to be illustrated and solved. The comparison of these filters can be made by finding the mean squared error(MSE) value. The MSE can be evaluated as

$$MSE = \frac{1}{N \times n} \sum_{i=1}^{N} (x_i - \hat{x}_i)^2. \tag{9.1}$$

where N is the number of time steps, n is the dimension of state vector, x_i is the simulated value and $\hat{x}_i$ is the estimated value from the filters.

9.2 Example 1

A challenging example and bench mark problem in nonlinear estimation research is given in Eq. 9.2 and Eq. 9.3 . The dynamic model and measurement

equation of the nonlinear system is given as

$$x_{k+1} = \frac{x_k}{2} + \frac{25x_k}{1+x_k^2} + 8cos(1.2k) + w_k \tag{9.2}$$

$$y_k = \frac{x_k^2}{20} + \nu_k \tag{9.3}$$

where w_k has mean 0 and process noise covariance $Q = 10$ and ν_k has mean 0 and measurement noise covariance $R = 1$. The important aim here is to have a look at how each filter works depending on which their performance can be analyzed. Therefore, for initialization of moving horizon estimation we use nonlinear filters. Due to approximations, extended Kalman filter, as well as unscented Kalman filter are not suitable for moving horizon estimation initialization. The filter initial condition is $\hat{x}_0 = 1$ same as the true initial condition, the variance, $\hat{P}_0 = 1$ and time steps N=64.

9.2.1 Model and Measurement

The true initial conditions are used in order to create the working model. The Fig. 9 represents a plot between time and simulated values of state model. A comparison between these values and estimated values obtained from the filter will be made in following sections.

The Fig. 10 is a plot between time and measurement values.

9.2.2 Performance of Nonlinear Filters

The performance of EKF and PF are explained through this example. The Fig. 11 shows the performance of EKF. Here the true simulated values of the state are expressed as lines and estimated values are expressed as circles. How good the filter works is dependent on how close enough are the estimated and true values which is represented as MSE.

Table I: MSE values of nonlinear filters: extended Kalman filter and particle filter filter are compared

Nonlinear Filter	**MSE**	**CPU time** in seconds
EKF	$2.3E+08$	4.11
PF	$2.17E+01$	3.11

The Fig. 12 shows the performance of particle filter for the example. The particle filter is initiated with 512 samples/particles. Here the true simulated values are expressed as lines and estimated values are expressed as circles. The performance is evaluated based on the number of samples needed and how good is the MSE value.

From Fig. 11 and Fig. 12 the performance of the particle filter is better than EKF. This is further proved by their MSE values obtained over 100 realizations which is listed in Table I.

Both the EKF anf PF are used for initializing the MHE and the performance of MHE is recorded.

In Fig. 13 the performance of MHE when initialized with EKF, is shown as a plot between the time step and the true simulated value, EKF estimate and MHE initialized with EKF estimate.

In Fig. 14 the performace of MHE when initialized with PF, is shown as a plot between the time step and the true simulated value, PF estimate and MHE initialized with PF estimate.

From Fig. 13 and Fig. 14 it is clear that the performance of MHE initialized through PF is better than initialization through EKF. This is significant from the fact that MHE initialized through PF estimate is close to the true simulated values.

Thus, it is understood how EKF can be a poor arrival cost initialization when the nonlinear function is of higher order and how particle filter works effectively.

Table II: MSE values of PF and EKF initialized MHE: the performance of the filters are significant from the MSE values

MSE values		
Nonlinear Filter	**MHE - EKF**	**MHE - PF**
D = 2	$1.95E+04$	$3.54E+01$
D = 3	$1.64E+04$	$3.43E+01$
D = 5	$1.41E+04$	$3.13E+01$
D = 6	$1.32E+04$	$3.16E+01$
D = 9	$1.25E+04$	$3.08E+01$
D = 10	$1.23E+04$	$3.02E+01$

The Table II gives the MSE values over 100 realizations for different horizon lengths. As the length of horizon increases the MHE value decreases which shows a better performance. There cannot be a justification for which filter is better in comparison to MSE values as one is mean estimate and other is mode estimate.

Figure 9: Simulation of the process: simulated values of the state model are plotted against the discrete time interval

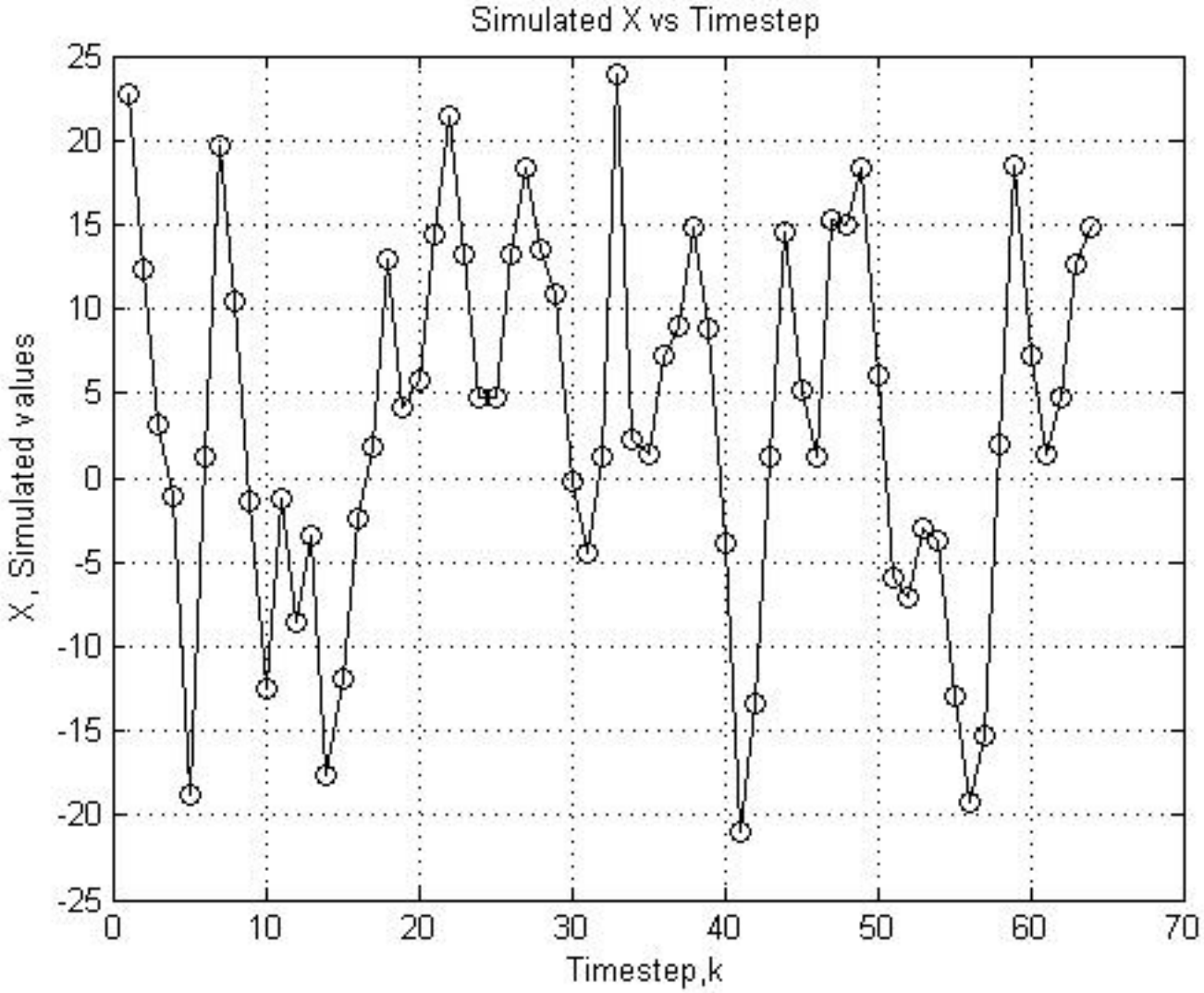


Figure 10: Simulation of the measurement: simulated values of the measurement model are plotted against the discrete time interval

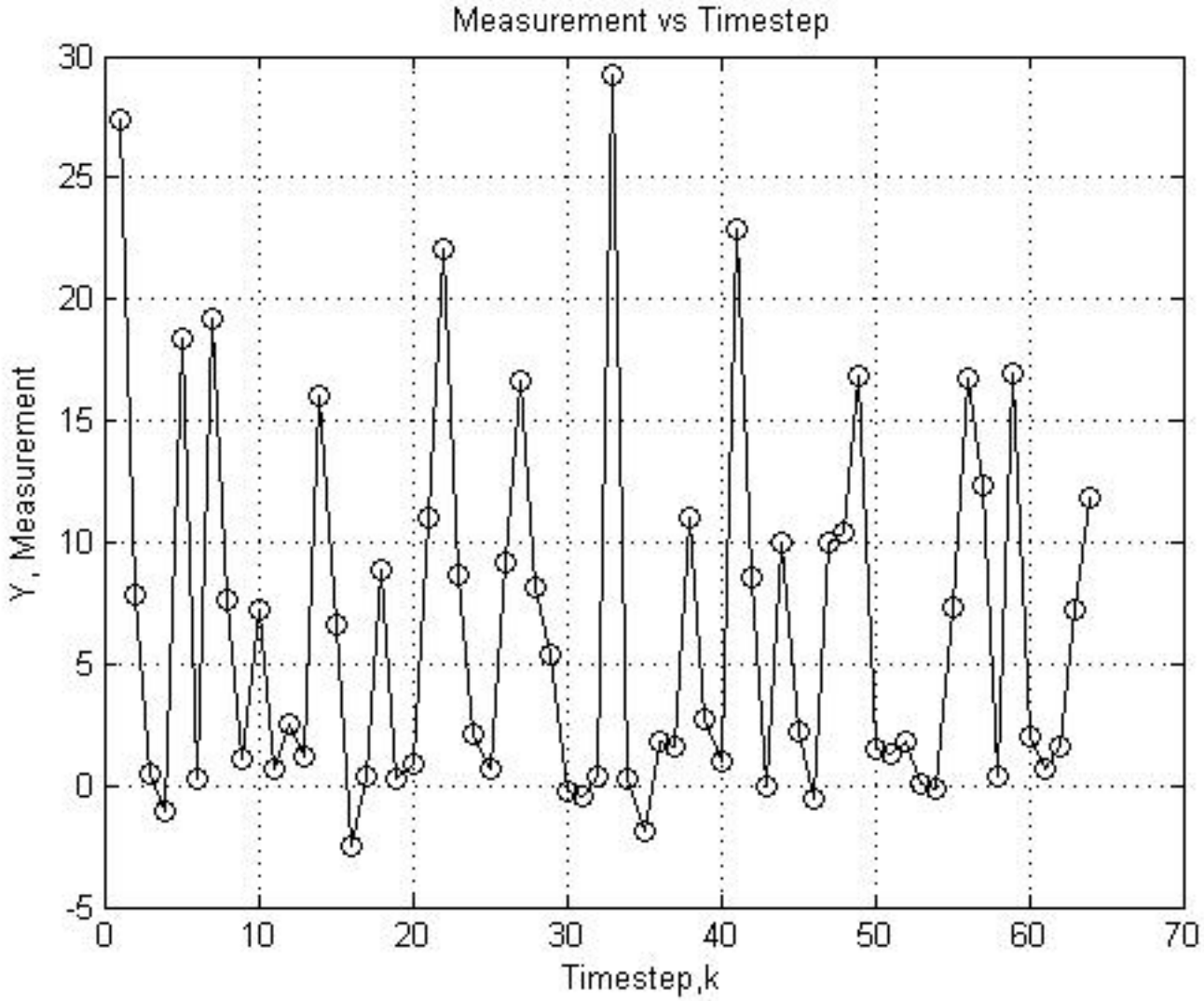


Figure 11: Performance of extended Kalman filter: the simulated values and the estimated values of the model are plotted against the discrete time interval

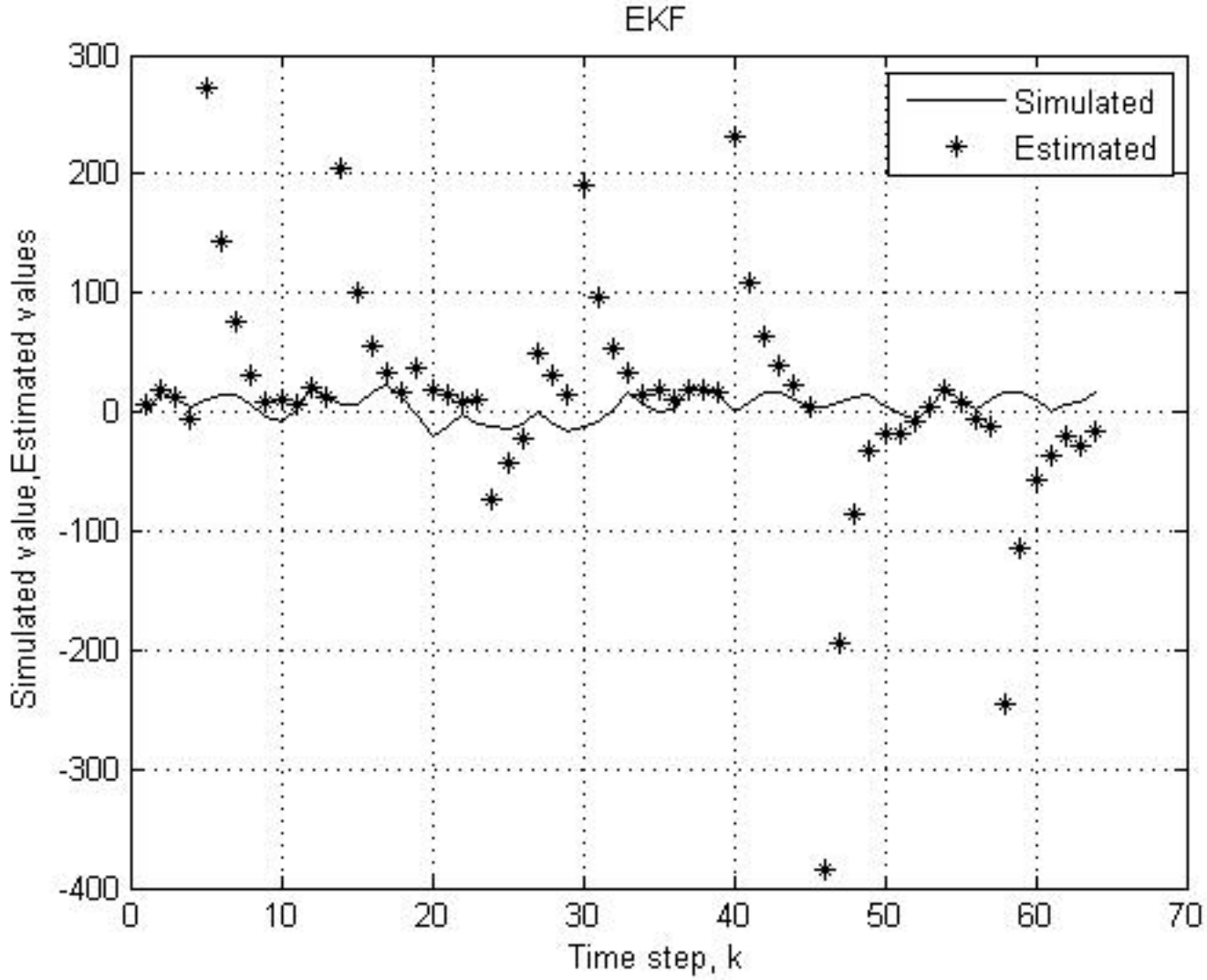


Figure 12: Performance of Particle Filter: the comparison between the simulated and estimated values of the model are shown which demonstrates the performance of the filter

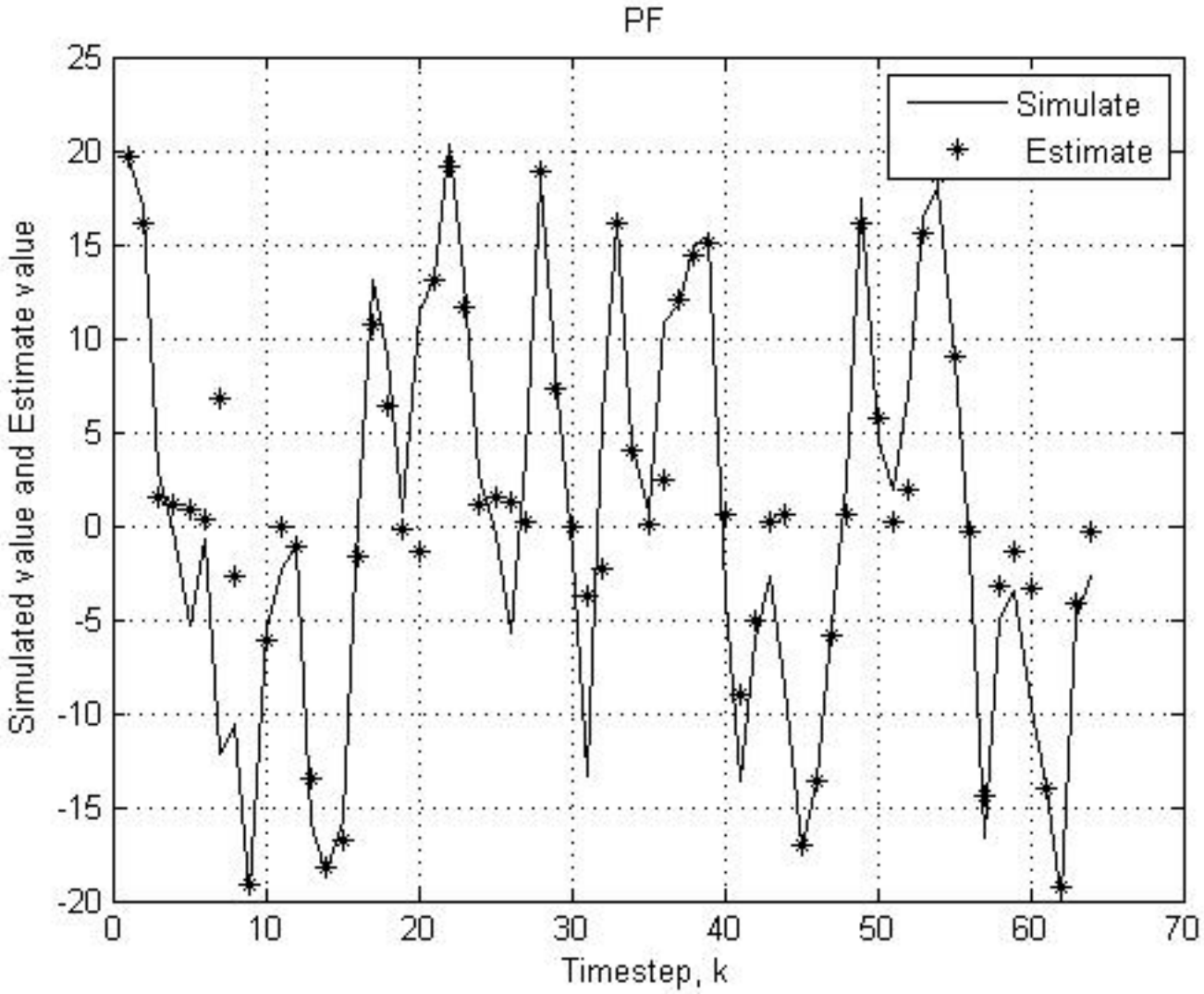


Figure 13: Performance of MHE using EKF initialization, the MHE estimates are closer to the simulated values

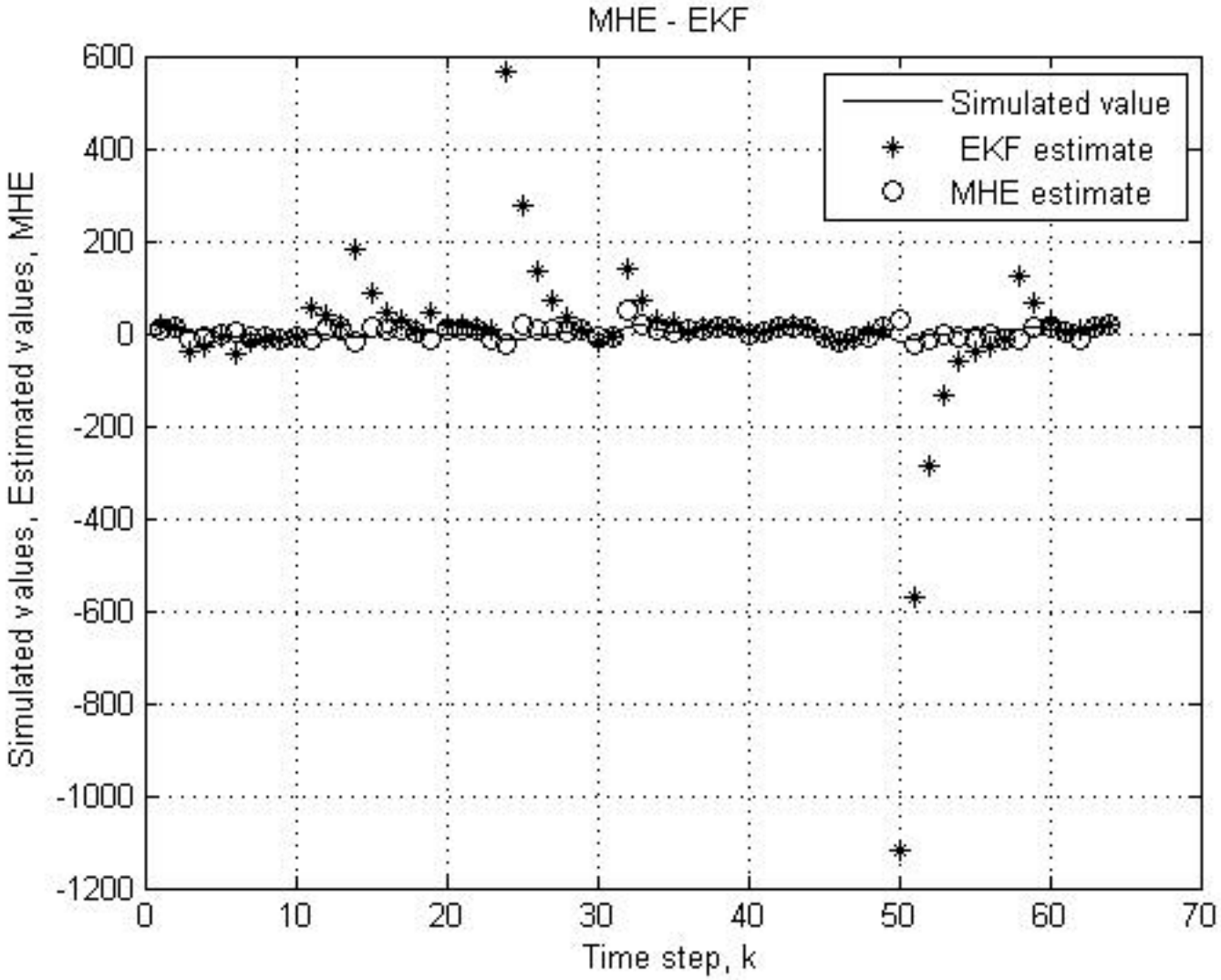
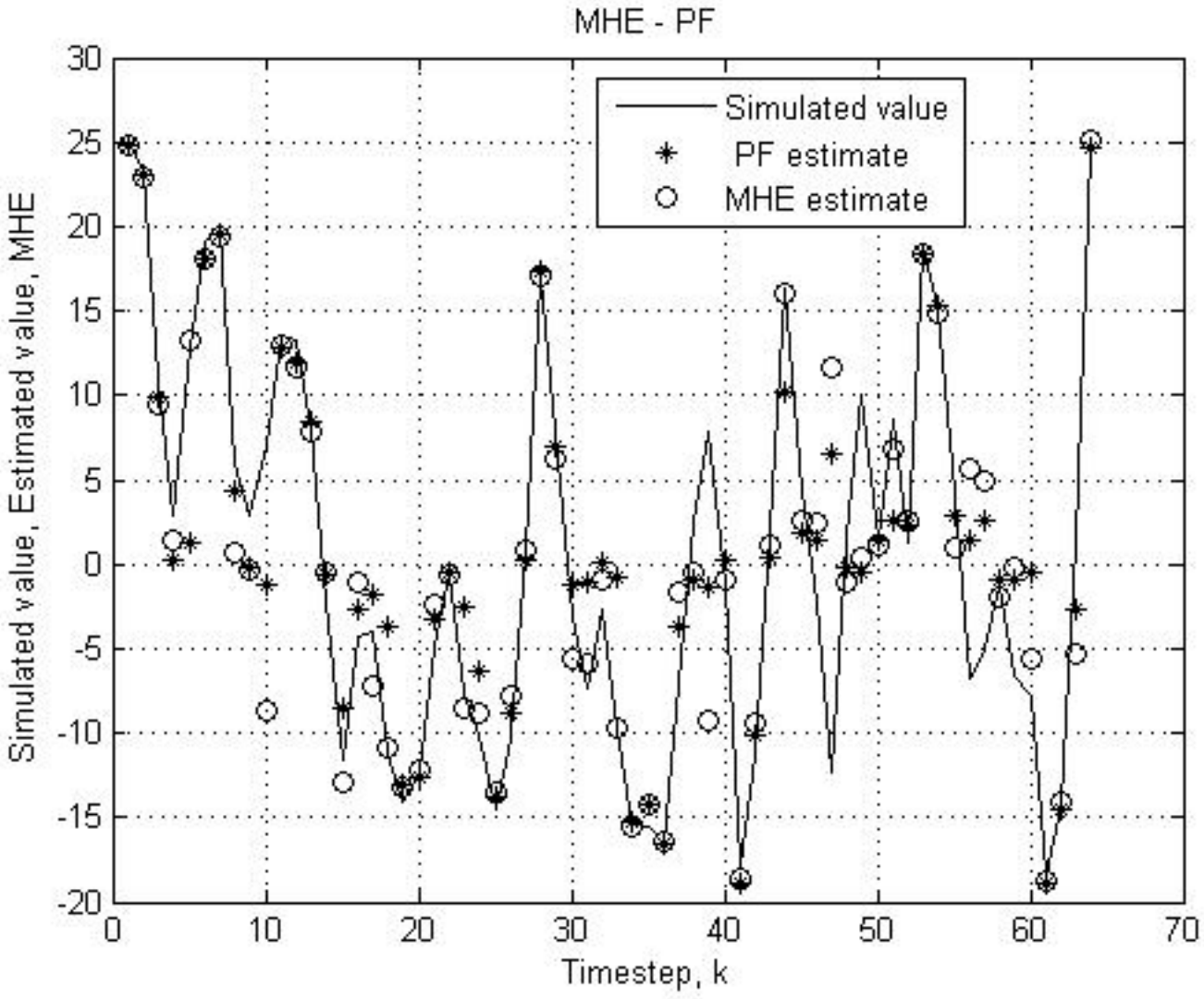


Figure 14: Performance of MHE using PF initialization



CHAPTER X

Dynamic Programming in MHE

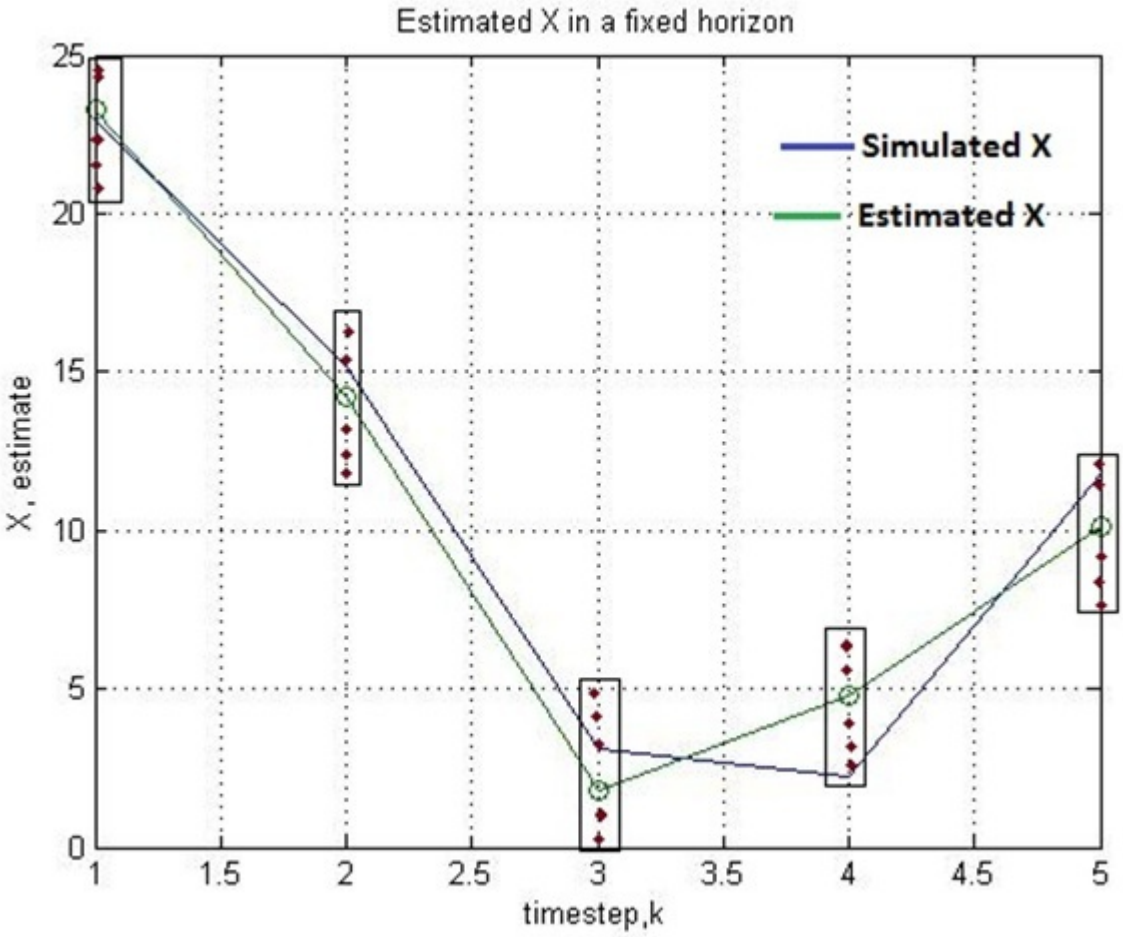
10.1 Introduction

The main aim of this thesis is to change the optimization strategy already existing in the MHE algorithm. Instead of using the existing method of Nonlinear Programming algorithm, use of the dynamic programming algorithm with MHE is done, but it can be only used with particle filter initialization because only in this nonlinear filter we have samples by which dynamic programming algorithm can perform its optimization strategy on cost function.

10.2 How it works?

Take a particular horizon length D =5 and now dynamic programming algorithm takes effect. At every time step k there are particles or samples, using which the optimization is done by the backward solution for forward approach in dynamic programming. It is explained in Fig. 15.

Figure 15: Implementation of DP in MHE, the samples close to the estimate are chosen at each time step using DP algorithm



In Fig. 15 at each time step there are 5 samples considered. Out of the 5 samples at each time point dynamic programming aims at finding the sample which is closest to the estimate thus minimizing the cost. In this way the estimate which is closest to the true simulated value is found at each time step.

The plot shown in Fig. 15 indicates how dynamic programming leads to better optimization strategy.

The MSE values of the particle filter and the DP implemented MHE are tabulated.

From Fig. 16 the performance of MHE with NLP algorithm using PF initial-

Table III: MSE values of MHE with DP and NLP algorithm using PF initialization: the modified MHE - DP algorithm generated MSE values are compared with the original MHE algorithm generated MSE values

	MSE values	
Nonlinear Filter	**MHE - DP - PF**	**MHE - PF**
D = 2	$3.8E+01$	$3.54E+01$
D = 3	$3.6E+01$	$3.43E+01$
D = 5	$3.5E+01$	$3.13E+01$
D = 6	$3.01E+01$	$3.16E+01$
D = 9	$2.78E+01$	$3.08E+01$
D = 10	$2.54E+01$	$3.02E+01$

ization is compared with the performance of MHE modified with DP algorithm, with particle filter initialization. In the Fig. 16 the circles represent the estimate value of MHE modified with DP algorithm and green line represents estimate value of MHE with NLP and true simulated values are represented in blue line.

In Table III the MSE values are shown. They vary according to the length of the horizon. Comparing the MSE values of these two algorithm, the performance of MHE modified through DP algorithm is better than MHE with NLP algorithm at higher horizon lengths. Even though the performance cannot be determined with just MSE values alone but for all practical purposes it is expected that the estimated values are close to the simulated values.

10.3 Performance

In order to explain more on the results of this modified MHE, there is a need to analyze the method.

The modified MHE with dynamic programming algorithm incorporated and can only be used, if MHE is initialized with PF. So, the efficiency of MHE with DP incorporated algorithm majorly depends on how well PF works in estimating the initial mean and covariance. PF initialization is based on the assumption that the random noise error is additive on state and measurement equations.

Consider the example in Eq. 9.2 and Eq. 9.3, the PF can work better if R and Q values are changed.

Here, the PF has similar difficulties. Hence, at lower horizon lengths the estimate values are little off than the MHE with NLP algorithm.

Also, the 100 realizations done in order to minimize any error has a different effect on this method.

The studies are explained in Fig. 17, Fig. 18 and Fig. 19. Fig. 17 is a plot between number of realizations and MSE for the PF.

Fig. 18 is a plot between number of realizations and MSE for MHE algorithm with PF initialization.

Fig. 19 is a plot between number of realizations and MSE for the MHE algorithm modified with DP, with PF initialization.

This shows at over 100 realizations in majority of instances PF fails or degeneracy occurs which results in higher MSE values. When the horizon lengths are large, it removes the past data and gives good results.

Figure 16: Performance of MHE-DP with paricle filter initialization: the modified MHE algorithm is represented in this performance curve

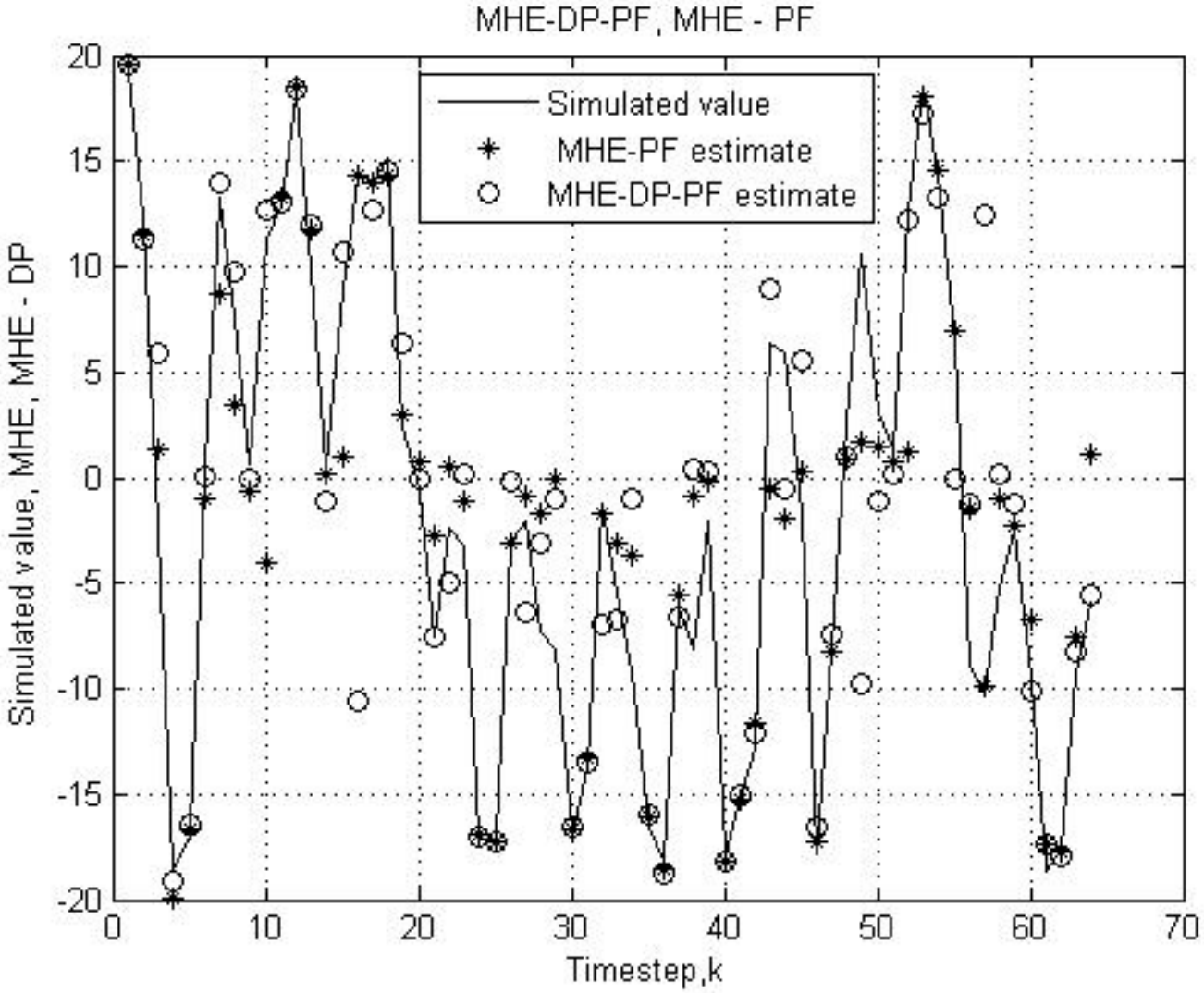


Figure 17: MSE vs No. of Realizations for PF

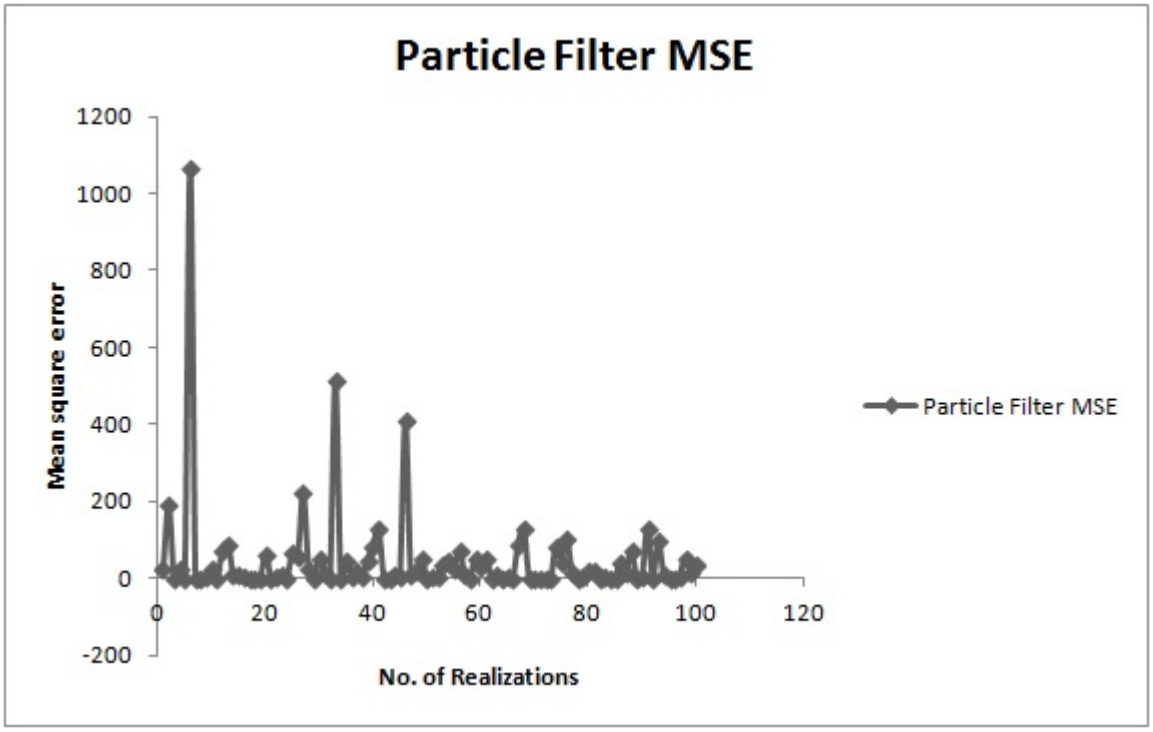


Figure 18: MSE vs No. of Realizations for MHE- PF

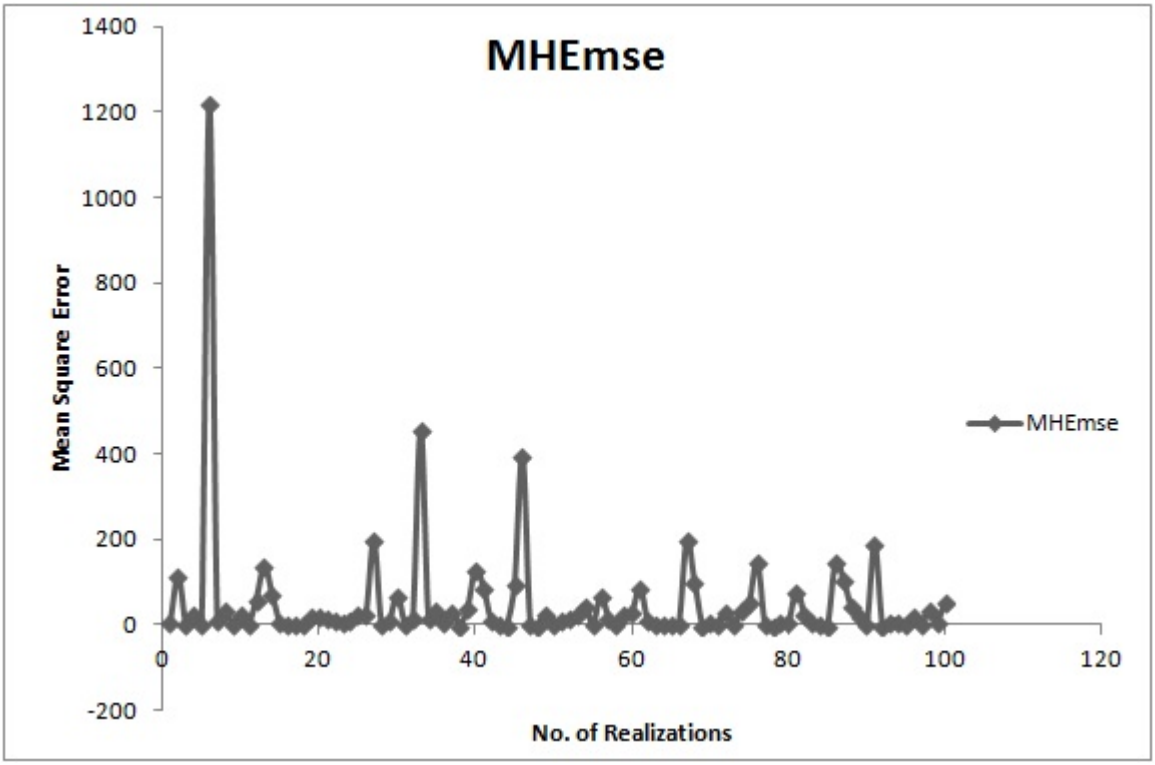
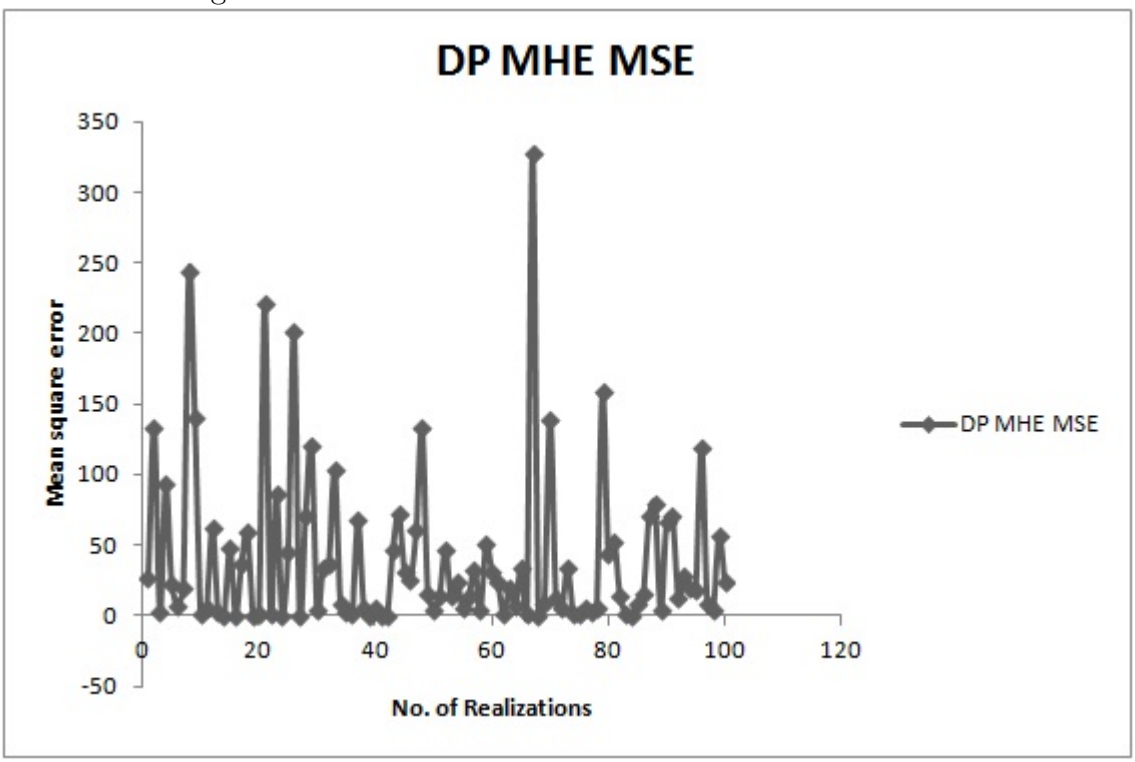


Figure 19: MSE vs No. of Realizations for MHE-DP-PF



CHAPTER XI

Conclusion

The objective of the thesis is to find an alternative approach to the existing MHE approach. The optimization of cost function is one of the gray areas in MHE and there are many number of ways to optimize the cost function. This thesis focuses on one such method, dynamic programming.

The concept of state estimation, filtering and smoothing are explained briefly in order to understand the major portion of the thesis. Bayesian state estimation and probability theory are explained in detail as they are the basis of many existing filters. Kalman filter, extended Kalman filter, unscented Kalman filter and particle filter are explained in detail as these filters are the important developments in the estimation field. The performance of EKF and PF are explained with examples.

The three specific aims of the thesis are achieved. Moving horizon estimation is explained in detail along with its algorithm and the concept of cost optimization is also briefly explained. The concept of dynamic programming is also explained with suitable examples.

The dynamic programming algorithm is implemented in MHE and a new al-

gorithm is developed which can be implemented instead of the conventional MHE algorithm. A mathematical example is stated and performance comparison between the new MHE algorithm and old method is analyzed.

Though this method has both its advantages and disadvantages the modified MHE algorithm can be implemented on nonlinear models due to cost optimization.

BIBLIOGRAPHY

[1] Ching, J., Beck, J.L., Porter, K.L., Shaikhutdinov, R., Bayesian state estimation method for nonlinear systems and its application to recorded seismic response, Journal of Engineering Mechanincs, vol. 132, No. 4, 396-410, April 2006.

[2] Kalman, R.E., A new approach to linear filtering and prediction problems, J. Basic Eng., vol. 82, 35-45, March 1960.

[3] Jazwinski, A.H., Stochastic process and filtering theory, Academic Press, 1970.

[4] Jazwinski, A. H., Filtering for Nonlinear Dynamical Systems, *IEEE Trans. Automatic Control*, vol. 11, 1966.

[5] Julier, S.J., Uhlmann, J.K., Durrant-Whyte, H.F., A new method for the nonlinear transformation of means and covariances in filters and estimators, IEEE Trans. Autom. Control., vol. 43, No. 3, 477-482, 2000.

[6] Julier, S. J., Uhlmann, J. K., New extension of the Kalman filter to nonlinear systems, *Proc. SPIE*, vol. 3068, no. 182, 1997.

[7] van der Merwe, R., de Freitas, N., Doucet, A., and Wan, E.A., The unscented particle filter, Rep. CUED/F-INFENG/TR380, Dept. of Engineering, Cambridge Univ., 349-354, 2000.

[8] Wan, E.A., van der Merwe, R., The unscented Kalman filter for nonlinear estimation, Proc. Symposium 2000 on Adaptive systems for signal Processing, Communication and Control, IEEE, 153-158, 2000.

[9] Julier, S.J., Uhlmann, J.K., Unscented Filtering and Nonlinear Estimation, *Proceedings of the IEEE.*, vol.92, no. 3, Mar. 2004.

[10] Gordon, N.J., Salmond, D. J., Smith, A.F.M., Novel approach to nonlinear/non-Gaussian Bayesian state estimation, IEE Proceedings, vol. 140, No. 2, 107-113, April 1993.

[11] Arulampalam, M., Maskell, S., Gordon, N., Clapp, T., A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking, *IEEE Trans. on Signal Processing.*, vol. 50, no. 2, Feb. 2002.

[12] Simon, D., *Optimal State Estimation: Kalman, H-Infinity and Nonlinear Approaches.* John Wiley and Sons, 2006.

[13] Haseltine, E.L., Rawlings, J.B., Critical evaluation of extended Kalman filtering and moving horizon estimation, Ind. Eng. Chem. Res., vol. 44, 2451-2460, 2005.

[14] Rawlings, J. B., Bakshi, B. R., Particle filtering and moving horizon estimation, *Computers and Chemical Engineering*, vol. 30, 2006.

[15] Zavala, V.M., Laird, C.D., Biegler, L.T., A fast moving horizon estimation algorithm based on nonlinear programming sensitivity, Journal of Process Control, vol. 28, 876-884, 2008.

[16] Moraal, P.E., Grizzle, J.W., Observer design for nonlinear systems with discrete time measurements, IEEE Transactions on Automatic Control, vol. 40, n0.3, 395-404, 1995.

[17] Zimmer, G., State estimation by online minimization, International Journal of Control, vol. 60, no. 4, 595-606, 1994.

[18] Alamir, M., Optimization based nonlinear observes revisited, International Journal of Control, vol. 72, no.13, 1024-1217, 1999.

[19] Michalska, H., Mayne, D.Q., Moving horizon observers and observer-based control, IEEE Transactions on Automatic Control, vol. 6, no.6, 995-1006, 1995.

[20] Rao, C.V., Rawlings, J.B., Mayne, D.Q., Constrained state estimation for nonlinear discrete-time systems: stability and moving horizon approximations, IEEE Transactions on Automatic Control, vol. 48, no. 2, 246-257, 2003.

[21] Ungarala, S., Computing the arrival cost parameters in moving horizon estimation using sampling based filters, Journal of Process Control, vol. 19, 1576-1588, 2009.

[22] Bellman, R., Dynamic Programming, Princeton University Press, 1957.

APPENDIX

1 Matlab files for EKF, PF

Code for EKF

```
 MSE = [];
 for count = 1:100
% Initial state
x =1;
n = length(x);
m = 1;
% Number of time steps
N = 64;
t = [1:N];
X = x;
% Noise covariances
Q = 10;
R = 1;
y = (x^2/20);
Y = y;
for k = 2:N
% Non-linear equations
x = 0.5*x + 25*x/(1+x^2) + 8*cos(1.2*(k-1)) + Q^0.5*randn(size(x));
y = (x^2/20) + R^0.5*randn;
X = [X,x];
Y = [Y;y];
end
% ESTIMATION
% EKF
```

```
xf = 1;
Pf = 1;
% Storage
Xf = xf;
% Filter loop
for k = 1:N
    F = 1/2 + 25/(1+xf^2)-(50*xf^2)/(1+xf^2)^2;
    xf = xf/2 + 25*xf/(1+xf^2)+ 8*cos(1.2*(k-1))+((Q)^0.5)*randn(size(xf));
    Pf = (F^2)*Pf + Q;
    H = xf/10;
    K = (Pf/H)*inv((H^2)*Pf+R);
    xf = xf+K*(Y(k)-(xf^2/20));
    Pf = Pf-K*H*Pf;
    Xf(k) = xf;
end
mse = sum(sum((X-Xf).^2)/R)/(n*N)
MSE = [MSE; mse];
end
m = MEAN(MSE);
plot (t,X,'k-',t,Xf, 'k*')
xlabel('Time step, k');
ylabel('Simulated value,Estimated values');
grid;
title('EKF');
legend('Simulated','Estimated');
```

Code for PF

```
t = [1:64];

n = length(t);

x_i = [1];

x_0 = [1];

P_0 = 1;

u_n = 10;

v_n = 1;

k=1;

Y = zeros(size(v_n,1),n);

X = zeros(size(u_n,1),n);

X(:,1) = 0.5*x_i + 25*x_i/(1+x_i^2) + 8*cos(1.2*(k-1)) + u_n^0.5*randn;

for k = 2:n

    X(:,k) = 0.5*X(:,k-1) + 25*X(:,k-1)/(1+X(:,k-1)^2) + 8*cos(1.2*(k-1)) + u_n^0.

end

for k = 1:n

    Y(k) = X(:,k).^2/20 + v_n^0.5*randn;

end

% Partilce Filter

M = x_0;

P = P_0;

n_particles = 512;

MM_BS = zeros(size(M,1),size(Y,2));

PP_BS = zeros(size(M,1),size(Y,2));

BS_particles =[];

Xh_BS = MM_BS;

SX = M + P^0.5*randn(1,n_particles)
```

```
tt = clock;
for k = 1:size(Y,2)
    SX = 0.5*SX + 25*SX./(1+SX.^2) + 8*cos(1.2*(k-1)) + u_n^0.5*randn(1,size(SX,2)
    SY = SX.^2/20;
    w = 1/sqrt(2*pi*v_n) *exp(-(Y(:,k)-SY).^2 / (2 * v_n));
    w = w/sum(w);
     [ind] = resampleResidual(w);
    SX = SX(ind);
    BS_particles(:,k) = SX';
    M = mean(SX');
    P = cov(SX');
    MM_BS(:,k) = M';
    PP_BS(:,k) = P;
end
BS_CPUT = etime(clock,tt);
BS_MSE = sum(X-MM_BS).^2/n
plot(t,X,'k-',t,MM_BS, 'k*')
xlabel('Timestep, k');
ylabel('Simulated value and Estimate value');
grid;
title('PF');
legend('Simulate',' Estimate')
```

Code for MHE-PF

```
BS_MSE = [];
MHE_MSE1 = [];
MHE_MSE2 = [];
```

```
MHE_MSE3 = [];
for count = 1:100
t = [1:64];
global xf Pf k D u_n v_n Y
n = length(t);
x_i = [1];
x_0 = [1];
P_0 = 1;
u_n = 10;
v_n = 1;
k=1;
Y = zeros(size(v_n,1),n);
X = zeros(size(u_n,1),n);
X(:,1) = 0.5*x_i + 25*x_i/(1+x_i^2) + 8*cos(1.2*(k-1)) + u_n^0.5*randn;
for k = 2:n
    X(:,k) = 0.5*X(:,k-1) + 25*X(:,k-1)/(1+X(:,k-1)^2) + 8*cos(1.2*(k-1)) + u_n^0.
end
for k = 1:n
    Y(k) = X(:,k).^2/20 + v_n^0.5*randn;
end
% Partilce Filter
M = x_0;
P = P_0;
n_particles = 512;
MM_BS = zeros(size(M,1),size(Y,2));
PP_BS = zeros(size(M,1),size(Y,2));
```

```
BS_particles =[];
Xh_BS = MM_BS;
D =2;
SX = M + P^0.5*randn(1,n_particles);
tt = clock;
for k = 1:size(Y,2)
    SX = 0.5*SX + 25*SX./(1+SX.^2) + 8*cos(1.2*(k-1)) + u_n^0.5*randn(1,size(SX,2)
    SY = SX.^2/20;
    xf= MEAN(SX);
   Pf = VAR(SX);
    w = 1/sqrt(2*pi*v_n) *exp(-(Y(:,k)-SY).^2 / (2 * v_n));
    w = w/sum(w);
    [ind] = resampleResidual(w);
    SX = SX(ind);
    BS_particles(:,k) = SX';
    M = mean(SX');
    P = cov(SX');
    MM_BS(:,k) = M';
    PP_BS(:,k) = P;
     if k >= D
        [xmhe,fval] = fminunc('MHE_PFfunction_cost_SU', MM_BS(k-D+1:k));

        Xmhe(k-D+1:k) = xmhe;
     end
end
BS_CPUT = etime(clock,tt)
```

```
bs_MSE = sum((X-MM_BS).^2)/n;

mhe_MSE = sum((X-Xmhe).^2)/n;

BS_MSE(:,count) = bs_MSE;

MHE_MSE(:,count) = mhe_MSE;

end

PFMSE = mean(BS_MSE)

MHEMSE1 = mean(MHE_MSE)

plot(t,X,'k-',t,MM_BS,'k*',t, Xmhe, 'ko')

xlabel('Timestep, k');

ylabel('Simulated value, Estimated value, MHE');

grid;

title('MHE - PF');

legend('Simulated value',' PF estimate', 'MHE estimate')
```

Code for MHE-DP-PF

```
BS_MSE = [];

MHE_MSE = [];

for count =1:100

t = [1:64];

global xf Pf k D u_n v_n Y

n = length(t);

x_i = [1];

x_0 = [1];

P_0 = 1;

u_n = 10;

v_n = 1;

k=1;
```

```
Y = zeros(size(v_n,1),n);

X = zeros(size(u_n,1),n);

X(:,1) = 0.5*x_i + 25*x_i/(1+x_i^2) + 8*cos(1.2*(k-1)) + u_n^0.5*randn;

for k = 2:n

    X(:,k) = 0.5*X(:,k-1) + 25*X(:,k-1)/(1+X(:,k-1)^2) + 8*cos(1.2*(k-1)) + u_n^0.

end

for k = 1:n

    Y(k) = X(:,k).^2/20 + v_n^0.5*randn;

end

f_func = @singlestate;

% Partilce Filter

M = x_0;

P = P_0;

n_particles = 512;

MM_BS = zeros(size(M,1),size(Y,2));

PP_BS = zeros(size(M,1),size(Y,2));

BS_particles =[];

Xh_BS = MM_BS;

D =10;

XPART =[];

SX = M + P^0.5*randn(1,n_particles);

tt = clock;

for k = 1:size(Y,2)

    SX = 0.5*SX + 25*SX./(1+SX.^2) + 8*cos(1.2*(k-1)) + u_n^0.5*randn(1,size(SX,2)

    SY = SX.^2/20;

   xf= MEAN(SX);
```

```
   Pf = VAR(SX);
   XPART = [XPART, SX'];
    if k>=D
        Yh = Y(:,k-D+1:k);
        [r,c] = size(XPART);
        Xpart = XPART(:, c-D+1:c);
        aa = (Xpart(:,1) - xf).^2/Pf;
        YY = repmat(Yh,n_particles,1);
       for i = 1: D
           ba(:,i) = (YY(:,i) - Xpart(:,i).^2./20).^2/v_n;
       end
        bb = aa + ba(:,1);
       th =k-D;
        for j = 2:D
            for i = 1:n_particles
                ca = (Xpart(i,j) - feval(f_func,Xpart(:,j-1),th)).^2/u_n;
                da = bb(:,j-1) + ca;
                [DA, indx] = min(da);
                bb(i,j) = ba(i,j) + DA;
                phi(i,j) = indx;
                th = th+1;
            end
        end
    [ea,I] = min(bb(:,D));
    fa(:,D) = Xpart(I,D);
for j = D-1:-1:1
```

```
fa(:,j) = Xpart(phi(I,j+1),j);
Xmhe(:,k-D+1:k) = fa;
end
    end
    w = 1/sqrt(2*pi*v_n) *exp(-(Y(:,k)-SY).^2 / (2 * v_n));
    w = w/sum(w);
     [ind] = resampleResidual(w);
    SX = SX(ind);
    BS_particles(:,k) = SX';
    M = mean(SX');
    P = cov(SX');
    MM_BS(:,k) = M';
    PP_BS(:,k) = P;
end
BS_CPUT = etime(clock,tt);
bs_MSE = (sum(X-MM_BS)).^2/n;
mhe_MSE = (sum(X-Xmhe)).^2/n;
BS_MSE(:,count) = bs_MSE;
MHE_MSE(:,count) = mhe_MSE;
end
PFMSE = mean(BS_MSE)
MHEMSE = mean(MHE_MSE)
plot(t, X,'k-', t,MM_BS,'k*' ,t, Xmhe,'ko')
xlabel('Timestep,k');
ylabel('Simulated value, MHE, MHE - DP ');
grid;
```

```
title('MHE-DP-PF, MHE - PF');
legend('Simulated value',' MHE-PF estimate', 'MHE-DP-PF estimate')
```