University of Alberta

Computational Tools for Soft Sensing and State Estimation

by

Saneej Balakrishnapillai Chitralekha

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Process Control

Department of Chemical and Materials Engineering

Spring 2011
Edmonton, Alberta

Examining Committee

Dr. Sirish L. Shah, Chemical and Materials Engineering

Dr. Biao Huang, Chemical and Materials Engineering

Dr. Vinay Prasad, Chemical and Materials Engineering

Dr. Alan F. Lynch, Electrical and Computer Engineering

Dr. Japan J. Trivedi, Civil and Environmental Engineering
School of Mining and Petroleum Engineering, University of Alberta

Dr. Bjarne A. Foss, Engineering Cybernetics
Norwegian University of Science and Technology, Trondheim, Norway

Dedicated to

My Parents and Teachers

Abstract

The development of fast and efficient computer hardware technology has resulted in the rapid development of numerous computational software tools for making statistical inferences. The computational algorithms, which are the backbone of these tools, originate from distinct areas in science, mathematics and engineering. The main focus of this thesis is on computational tools which can be employed for estimating unmeasured variables in a process using all the available prior information. Specifically, this thesis demonstrates the application of a variety of tools for soft sensing of process variables and uncertain parameters of physio-chemical process models, using routine data available from the process. The application examples presented in this thesis come from broad areas where process uncertainty is inherent and includes petrochemical processes, mechanical valve actuators, and upstream production processes in petroleum reservoirs. The mathematical models that are employed in different domains vary significantly in their structure and their level of complexity. In the petrochemical domain, the focus was on developing empirical soft sensors which are essentially non-parametric mathematical models identified using routine data from the process. The Support Vector Regression technique was applied for identifying such non-parametric empirical models. On the other hand, in all the other application examples in this thesis the physical parametric models of the process were utilized. The latter application examples, which cover a major portion of this thesis, demonstrate the application of modern state and parameter estimation algorithms which are firmly grounded on Bayesian theory and Monte Carlo techniques. Prior to the chapters on the application of state and parameter estimation techniques, a

tutorial overview of the Monte Carlo simulation based state estimation algorithms is provided with an attempt to throw new light on these techniques. The tutorial is aimed at making these techniques simple to visualize and understand. The application case studies serve to illustrate the performance of the different algorithms. All case studies presented in this thesis are performed on processes that exhibit significant nonlinearity in terms of the relationship between the process input variables and output variables.

Acknowledgements

I would like to express my sincere gratitude to my supervisor Prof. Sirish Shah for guiding my research and acting as a mentor to my overall professional development. I would like to thank him for the constant motivation and imparting me the "big picture view" of my research work during the discussions we have had at the university. I would like to sincerely thank him for providing me excellent opportunities for attending international technical conferences and interacting with external research group members, which were quite instrumental in moulding my intellect. The freedom of thought that he gave me and the patience he showed as I was climbing the learning curve has made it a pleasure to have him as my supervisor. The critical reviews and suggestions that he provided for the improvement of my papers and presentations were very valuable. Also, I want to acknowledge the financial support he provided me throughout my graduate studies.

During the course of my thesis work, I have had close interactions with several external professors and researchers. I would like to thank Dr. J. Prakash, who spent a significant amount of time with me during his summer sabbatical visits to our research group. I would like to thank Dr. Bhushan Gopaluni and Dr. Harigopal Raghavan, who took the time out of their busy schedule to guide me and co-author a paper with me. I spent almost the entire final year of my PhD work learning about the field of petroleum engineering from Dr. Japan Trivedi at the University of Alberta. I would like to thank him, along with his graduate student Ali Gul for helping me with the petroleum reservoir simulation software.

I would like to thank my group of friends in Edmonton for making my graduate student life a wonderful experience, both professionally and personally. I would like to thank all the CPC group members for providing their critical reviews during my presentations in the group. I express my sincere thanks to all my other friends in Edmonton, outside the work group, who knowingly or unknowingly have supported me during my stay in Edmonton. The number of people in these groups seems

countless for me to enlist the names of all of them here.

Last, but not the least, I owe a lot of thanks to my parents, my sister and her family who have always given their encouragement and moral support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

The field of system identification deals with finding a mathematical model which can capture the relationship between a set of measured variables of a system with a reasonable degree of accuracy. Based on the basic nature of the dependency between the variables of the system, we can broadly classify identified models into two types: linear and nonlinear. It is well known that most real world systems are nonlinear by nature. However, the basic idea is to get a reasonably accurate approximation of the true system and because it is easier to do this if the process is assumed to be locally linear, we make some basic practical assumptions about the process so that a linear model is able to capture the dynamics sufficiently well. A common approach for modeling nonlinear processes is to use multiple linear models which when combined together can cover the entire operating range of the nonlinear process. The assumption here is that the process is locally linear within each of these operating regimes. However, increasingly tight specifications, environmental considerations and economic pressures are pushing the operational windows of many of the industrial processes to regions where the assumption of linearity breaks down (Rhodes & Morari, 1998). This necessitates the application of nonlinear modeling and control theory to many of today's industrial processes. Nonlinear Model Predictive Control (NMPC) is one such advanced control technique which can address the increasing demands of chemical industries. Reliable nonlinear dynamic models are key to the success of this methodology. As reported by Nagy & Allgöwer (2004), the number of NMPC applications in the chemical industry has been growing at a very fast rate. It is expected that NMPC will become more common with the development of new nonlinear model identification software tools that can make nonlinear models readily available (Camacho & Bordons, 2007).

The focus of this thesis is to study the application of nonlinear system identification tools that have so far been given less attention in the chemical process control literature. Based on the amount of *a priori* knowledge about the physical

and chemical properties of the process that is included in the model, we can broadly classify model structures into two categories (Sjöberg et al., 1995):

1. Black-box models

2. Gray-box models

Black-box models are data-based regression models which can efficiently capture the relationship between the process variables in the form of a mathematical function. Mostly, the function parameters are almost impossible to relate to the physical and chemical characteristics of the process. Soft sensors are a common type of data-based models, used in the industry as a cost-efficient substitute for hardware sensors. On the other hand, a complete set of differential equations which explicitly capture the physio-chemical properties of the process, with some unknown variables which have to be identified from data, are known as Gray-box models (Note: If all the model parameters are assumed to be perfectly known, they are known as White-box models). State space models belong to this category of models. Evidently, a significant amount of prior knowledge about the "physics and chemistry" of the process is required to arrive at such models. In all model structures (black or gray), there are certain unknown variables which have to be identified using data collected from the process. For the case of soft sensors, the parameters of the regression function will have to identified; for state space models, there will be unmeasured process states as well as unknown parameters which have to be estimated. There are a wide variety of computational tools that can be employed for identifying the unknown variables. For soft sensors, a deterministic optimization routine is the most common approach where the objective is to minimize the prediction errors. On the other hand, for state and parameter estimation a probabilistic route is more commonly taken since process uncertainty has to be accounted for while identifying the parameters of the first principles models.

1.2 Thesis Outline

This thesis has been prepared in the 'paper' format as per the guidelines from the Faculty of Graduate Studies and Research at the University of Alberta. The organization of the thesis is as follows. Chapter 2 presents the application of the Support Vector Regression technique for developing empirical soft sensors for nonlinear processes. The case studies discussed in this chapter serve to demonstrate the efficacy of the method for building steady state and dynamic models, using very little *a priori* knowledge about the process. In all the subsequent chapters, the focus is on advanced nonlinear state and parameter estimation tools, which require significant prior knowledge about the process. We assume that a first principles

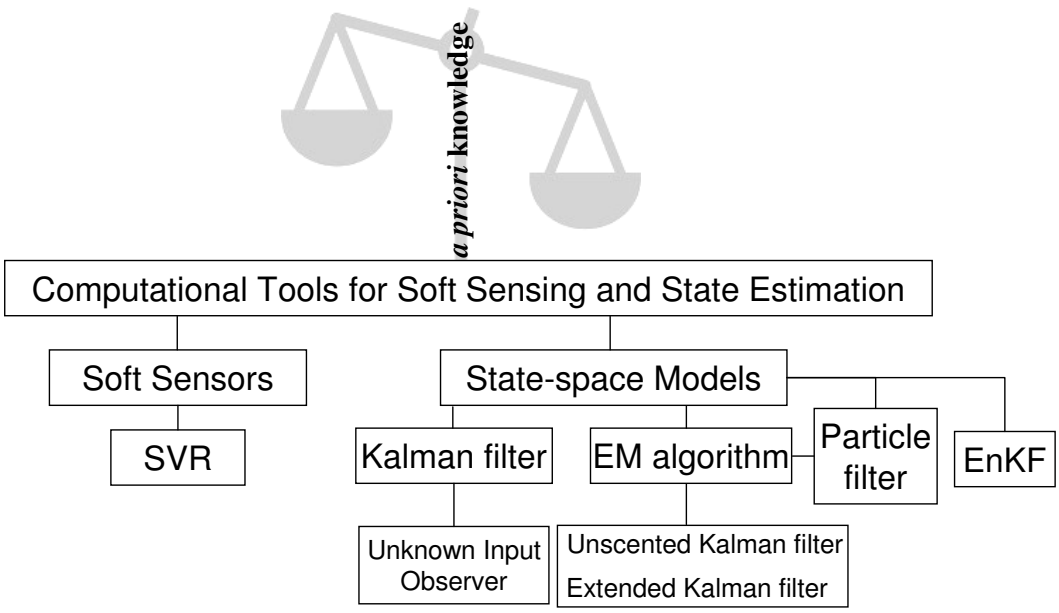


Figure 1.1: A snapshot view of this thesis.

based differential equation model of the process is available, with some unknown parameters and unmeasured states. Before dwelling into the application case studies, a tutorial overview of state and parameter estimation is first given in Chapter 3. The fundamentals of two state estimation algorithms, namely the ensemble Kalman filter (EnKF) and the particle filter, are discussed in detail in this chapter. Chapter 4 presents the application of a Kalman filter type of unknown input observer to detect and quantify valve stiction in process control loops. An advanced framework for combined state and parameter estimation, based on the expectation-maximization (EM) principle, is presented in Chapter 5. The application of the ensemble Kalman filter for online estimation of states and parameters is also presented in this chapter. Chapter 6 outlines the application of the EnKF algorithm for estimation of parameters of large scale petroleum reservoir simulation models. The efficacy of the methodology, more generally known as history matching, is demonstrated through two synthetic unconventional oil reservoir models. Each chapter in this thesis contains a broad overview of the subject matter, including a review of the pertinent references from the literature. To avoid repetition, the literature review is omitted in this introductory chapter.

Figure 1.1 shows a snapshot view of the various computational tools that are discussed in this thesis. As shown in this figure, the computational tools can be broadly classified into two categories based on the type of model structure identified, as follows: (1) Support Vector Regression for identifying soft sensor models, and (2) State and parameter estimation algorithms for identifying state-space models. The links connecting the blocks in the figure are representative of the organization of the

discussion in the remaining chapters of this thesis. The scales displayed at the top of the figure illustrate the amount of process *a priori* knowledge that is captured by the model structures in the two categories. Soft sensors are based on very little *a priori* knowledge about the process, whereas state-space models are based on a relatively higher amount of *a priori* knowledge about the physical and chemical characteristics of the process.

Bibliography

Camacho, E. & Bordons, C. (2007). Nonlinear model predictive control: An introductory review. In R. Findeisen, F. Allgöwer, & L. Biegler (Eds.), *Assessment and Future Directions of Nonlinear Model Predictive Control*, volume 358 of *Lecture Notes in Control and Information Sciences* (pp. 1–16). Springer, Berlin.

Nagy, Z. K. & Allgöwer, F. (2004). Nonlinear model predictive control: From chemical industry to microelectronics. In *43rd IEEE Conference on Decision and Control, Atlantis, Bahamas*, volume 4, (pp. 4249–4254).

Rhodes, C. & Morari, M. (1998). Determining the model order of nonlinear input/output systems. *AIChE Journal*, *44*(1), 151–163.

Sjöberg, J., Zhang, Q., Ljung, L., Benveniste, A., Delyon, B., Glorennec, P.-Y., Hjalmarsson, H., & Juditsky, A. (1995). Nonlinear black-box modeling in system identification: a unified overview. *Automatica*, *31*(12), 1691 – 1724.

Chapter 2

Application of Support Vector Regression for Developing Soft Sensors[1,2]

2.1 Introduction

Over the past decades, there have been significant advances in the field of nonlinear model based control. But many times, one of the key practical obstacles to the wider industrial adoption of such model based control strategies has been the difficulty of developing nonlinear models (Pearson, 2006). In general, the model should be estimated in combination with some *a priori* information about the process (often known as the gray-box approach). A first principles or a mechanistic description of the process in which all the parameters are assumed to be known *a priori* is known as a white-box model, whereas a model developed by a purely data-based modeling or identification exercise is known as a black-box model. In most cases, the lack of *a priori* knowledge about the nonlinearity of the process has made the black-box procedure very attractive. Even in cases where the structural information about the process is available, the effort put in for estimating parameters of such models makes the gray-box procedure less attractive. Most of the popular methods for the development of black-box nonlinear models are based on the theory of NARMAX/NARX models, built using neural networks or techniques involving kernel or logarithmic transformation followed by dynamic PCA or linear modeling approaches.

As a consequence of many different application areas, there has been a pronounced proliferation of methods, concepts and results in nonlinear systems identification (Ljung, 2006). One of the tools of data analysis which has recently shown

[1] A version of this chapter has appeared in the Canadian Journal of Chemical Engineering (Chitralekha & Shah, 2010a).

[2] A condensed version of this chapter was presented at the 18th IEEE Mediterranean Conference on Control and Automation, June 23 – 25, 2010, Marrakech, Morocco (Chitralekha & Shah, 2010b).

fairly wide spread applicability to a diverse class of real world problems is the Support Vector Machine algorithm, a kernel-based method, developed by Vapnik and co-workers in 1995. Originally developed for classification, the method was extended to regression and is known as Support Vector Regression Machine (SVR) (Drucker et al., 1997; Smola & Scholkopf, 2003). The main feature of this algorithm is the use of a nonlinear kernel transformation to map the input variables of the model into a feature space so that the input-output relationship can be approximated as a linear function in the feature space. A non-parametric regression model in the feature space is built by solving a constrained convex optimization problem. The objective function is a weighted combination of two distinct criterions, based on the idea of trade-off between low model prediction errors and low risk of overfitting the data (regularization). An important property of the optimization problem solution is that the estimated models represent sparse nonlinear approximations of the input-output relationship of the variables. The excellent generalization capabilities of the method to high dimensional nonlinear problems have made this method one of the best available methods for classification and regression in the past decade. The method has strong theoretical foundation and is based on the Statistical Learning theory or Vapnik-Chervonenkis (VC) theory developed by Vapnik and Chervonenkis (Vapnik & Chervonenkis, 1974; Vapnik, 1998). Recently, there has been a lot of interest on the application of this tool to nonlinear system identification. Suykens (2009) has presented a recent survey on the state-of-the-art of dynamical systems modeling using the SVR technique.

This chapter shows the application of the SVR technique for the development of nonlinear soft sensor models for chemical engineering processes. The versatility of this tool has been the main motivation behind the use of this technique in this work. The idea is to develop models that can capture the dynamics of the nonlinear process over a wide operating range. This would remove the need for developing multiple linear models at several operating regimes. In addition, a single nonlinear dynamic model can become very useful for control during process transitions between different operating regimes. For example, the problem of optimal grade transitions in a polymer manufacturing unit can be solved by having a good model which can accurately predict the dynamic changes required in the manipulative variables for achieving fast grade transitions. The main advantage of using the support vector methodology for modeling nonlinear systems is the ease of use of the algorithms. Unlike standard identification methods employed for nonlinear systems, in which the identification approach is recursive, the SVR approach is fairly straight forward.

This chapter has been organized as follows. In Section 2.2, a brief overview of the theory of Support Vector Regression is presented. Section 2.3 presents an application of SVR for developing a data based melt index soft sensor for an extruder

in an industrial EVA polymerization plant. The soft sensor predicts the melt index of different grades of polymer based on its steady state relationship with the secondary variables measured in the extruder such as temperature, pressure etc. The application of the SVR tool for developing dynamic nonlinear models is presented in Section 2.4. Using an automated procedure of model order and delay selection, an SVR-based nonlinear ARX model was developed for two nonlinear processes: a pH neutralization process which is a benchmark example of a process with nonlinear dynamics and a laboratory scale twin screw extrusion process. Finally, Section 2.5 summarizes the major concluding remarks from the different application studies of SVR conducted in this work.

2.2 Support Vector Regression: Overview

The basic idea of regression estimation is to approximate the functional relationship between a set of independent variables and a dependent variable by minimizing a "risk" functional which is a measure of prediction errors. More formally, we can describe the problem as follows (Smola & Scholkopf, 2003; Vapnik, 1998):
Suppose we are given training data $\{(x_1, y_1)....(x_l, y_l)\} \subset \chi \times \mathbb{R}$, where χ denotes the d dimensional space of the independent variable x (i.e., $\chi \in \mathbb{R}^d$) and y denotes univariate, random, independent observations. Consider a set of real valued functions given in a parametric form $\{f(x, w), w \in \Lambda\}$, where Λ is an arbitrary set which w belongs to. The dimension of w is equal to the number of parameters which have to be estimated in the function f. Staying in line with classical approaches, it is assumed that y results from a measurement with additive noise. This assumption is quite realistic and also gives more support to the choice of simple risk functional discussed below. The functional is defined as

$$R(w) = \frac{1}{l}\sum_{i=1}^{l} L(y_i, f(x_i, w))$$

where $L(y_i, f(x_i, w))$ is called the *Loss function*. The objective is to find $w = w^*$ that minimizes the functional, i.e., we seek the solution given by $w^* = \arg\min_{w\in\Lambda}(R(w))$. New predictions are then obtained as $\widehat{y} = f(x, w^*)$. In the classical approach the functional is Mean Square of Errors(MSE)

$$MSE = \frac{1}{l}\sum_{i=1}^{l}(y_i - f(x_i, w))^2 \tag{2.1}$$

where the loss function is quadratic and the method is called Least Square Regression. It is known that if the additive noise belongs to a Gaussian probability distribution, minimizing the MSE gives an efficient (unbiased) estimator of the regression $f(x, w)$. However, if the additive noise is generated by other laws, better

approximations are given by estimators based on other loss functions. For example, if one only knows that the density describing the noise is a symmetric smooth function, then the best choice of loss function is given by (see Huber (1964)):

$$L(y, f(x, w)) = |y - f(x, w)| \tag{2.2}$$

where $|\ .|$ denotes the one-norm. In Support Vector Regression (SVR), a new type of loss function, called ε-insensitive loss function is used, given by (see Vapnik (1998)):

$$M(y, f(x, w)) = L(|y - f(x, w)|_\varepsilon) \tag{2.3}$$

where we denote

$$|y - f(x, w)|_\varepsilon = \begin{cases} 0 & |y - f(x,w)| \leq \varepsilon \\ |y - f(x, w)| - \varepsilon & \text{otherwise} \end{cases} \tag{2.4}$$

These loss functions describe the ε-insensitive model: the loss is equal to 0 if the discrepancy between the predicted and the observed values is less than ε.

For explaining the basic idea and subsequent application examples, the Linear ε-insensitive loss function based SVR is considered here, referred to as $\varepsilon-$SVR given by

$$L(|y - f(x, w)|_\varepsilon) = |y - f(x, w)|_\varepsilon \tag{2.5}$$

For the case of linear regression, consider linear functions f of the form

$$f(x) = \langle w, x \rangle + b \text{ with } w \in \chi, b \in \mathbb{R} \tag{2.6}$$

where $\langle ., . \rangle$ denotes dot product in χ. In $\varepsilon-$SVR, the goal is to find (w^*, b^*) which gives a $f(x)$ that has the minimum risk w.r.t. the loss function in Eq. (2.5). In addition, a regularization penalty is imposed on w in order to prevent overfitting. This problem can be formulated as a constrained convex optimization problem as follows:

$$\underset{\text{w.r.t } w,\ b,\ \xi_i,\ \xi_i^*}{\text{minimize}} \quad \frac{1}{2} \|w\|_{\ell_2}^2 + C \sum_{i=1}^{l} (\xi_i + \xi_i^*) \tag{2.7a}$$

$$\underset{for\ i=1\ to\ l}{\text{subject to}} \begin{cases} y_i - (\langle w, x_i \rangle + b) \leq \epsilon + \xi_i \\ (\langle w, x_i \rangle + b) - y_i \leq \epsilon + \xi_i^* \\ \xi_i, \xi_i^* \quad \geq 0 \end{cases} \tag{2.7b}$$

The square of the ℓ_2 norm of the weight vector (model parameters), i.e., $\|w\|_{\ell_2}^2 = \langle w, w \rangle$, is the regularization term in the objective function. The main objective of regularization is to give more preference to solutions with lower norm of the model parameters. From a system identification point of view, this can be interpreted as the parsimony principle in which only parameters that have significant impact on the regression fit will be used in the final model. The constant $C > 0$ determines the

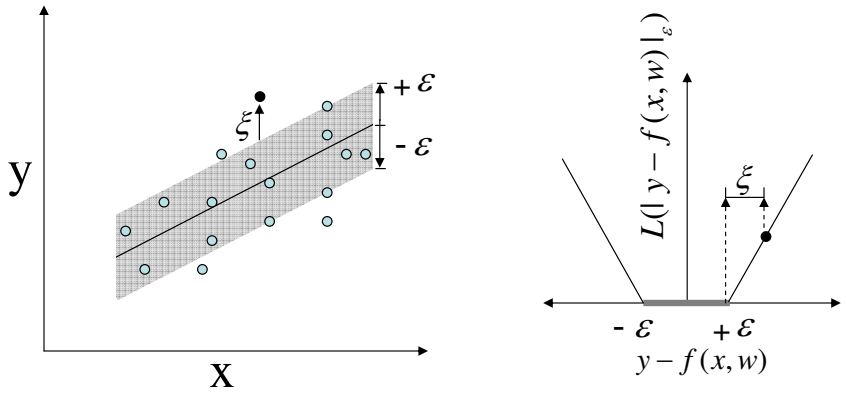


Figure 2.1: ε-insensitive loss function for a linear SVR.

trade-off between the penalty imposed on w and the tolerance on deviations larger than ε. The variables $\xi_i,\ \xi_i^*$ are *slack variables* which make the constraints feasible for training points having prediction errors more than ϵ and also penalize them in the objective function (also known as *soft constraints* in constrained optimization terminology). This is depicted graphically in Figure 2.1. On the left hand side of this figure, the straight line represents predictions from a linear SVR model. The shaded region depicts the epsilon tube around this straight line. The points, in blue and black color, represent the training data, $(x_l,\ y_l)$, available from a real world process. The data point shown in black color, lying outside the ε-tube, highlights the linear dependence of the loss function on prediction errors higher than ε.

The above defined problem is the basic formulation for the linear SVR. It can be solved using the Lagrangian method of constructing the Lagrange function with the introduction of dual variables for each of the $(4l)$ constraints in Eq. (2.7b). The dual problem formulation is summarized here and more details can be found in the book on statistical learning theory by Vapnik (1998) or the tutorial by Smola & Scholkopf (2003). For this, let $(\alpha_i, \alpha_i^*, \eta_i, \eta_i^*)_{i=1\ldots l}$ be the dual variables (Lagrange multipliers) corresponding to the inequality constraints in Eq. (2.7b). The Lagrangian is given by

$$\begin{aligned}\Gamma \quad \equiv \quad & \frac{1}{2}\left\|w\right\|_{\ell_2}^2 + C\sum_{i=1}^{l}(\xi_i + \xi_i^*) - \sum_{i=1}^{l}(\xi_i\eta_i + \xi_i^*\eta_i^*) \\ & -\sum_{i=1}^{l}\alpha_i(\epsilon + \xi_i - y_i + \langle w, x_i\rangle + b) - \sum_{i=1}^{l}\alpha_i^*(\epsilon + \xi_i^* + y_i - \langle w, x_i\rangle - b) \quad (2.8)\end{aligned}$$

For optimality, the partial derivative of the Lagrangian with respect to the primal

variables ($w,\ b,\ \xi_i,\ \xi_i^*$) should be equal to zero (saddle point conditions), i.e.,

$$\frac{\partial \Gamma}{\partial b} \triangleq \sum_{i=1}^{l}(\alpha_i - \alpha_i^*) = 0 \tag{2.9a}$$

$$\frac{\partial \Gamma}{\partial w} \triangleq w - \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)x_i = 0 \tag{2.9b}$$

$$\frac{\partial \Gamma}{\partial \xi_i} \triangleq C - \alpha_i - \eta_i = 0 \tag{2.9c}$$

$$\frac{\partial \Gamma}{\partial \xi_i^*} \triangleq C - \alpha_i^* - \eta_i^* = 0 \tag{2.9d}$$

As a result of the saddle point condition in Eq. (2.9b), the weight vector w takes the following important form:

$$w = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)x_i \tag{2.10a}$$

$$\text{thus,} \quad f(x) = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)\,\langle x_i, x\rangle + b \tag{2.10b}$$

From Eq. (2.10a), it is clear that the weight vector w can be described as a linear combination of the training patterns. Due to this form, it is not required to explicitly calculate w for evaluating $f(x)$ as shown in Eq. (2.10b). Moreover, it turns out that the complete algorithm can be described in terms of the dot product $\langle x_i, x_j\rangle$ between the input data. These observations become important for the direct extension of the algorithm to solve nonlinear regression.

One of the most important aspects of the solution is its sparsity which leads to the name *support vector*. Denoting $\beta_i = (\alpha_i - \alpha_i^*)$, the complementary slackness conditions force $\beta_i = 0$ to all training patterns inside the ϵ-tube (shaded region in Figure 2.1). A good choice of ϵ and C will lead to a majority of training patterns to have prediction errors $|f(x) - y| < \epsilon$ (inside the ϵ-tube). So only the smaller subset of training patterns that have nonzero β_i will be required to evaluate w or $f(x)$, which is therefore a sparse solution. These x_i's that come with nonzero coefficients are called *Support Vectors.* Also, the formulation is basically a convex optimization problem and thus is guaranteed to have a unique optimal solution that is also globally optimal. This is a great advantage when SVR is used for nonlinear regression over methods such as neural networks which usually suffer from the problem of local minima.

The advantage of sparse expansion of w in terms of the support vectors can be linked to the ε-insensitive loss function defined in Eqs. (2.3) and (2.4). Note that if $\varepsilon = 0$, the advantage of sparse decomposition is lost (Smola & Scholkopf, 2003). The

support vectors can be considered as a compressed set of samples which can be used to reconstruct or predict the output signal given a new set of input signals. In the nonlinear case, the $\varepsilon-$SVR can be utilized to yield sparse nonlinear approximations of the input-output relationship of the variables. Such approximations will be in terms of basis function dot products, called *kernel functions*, as will be explained later.

The key idea for the nonlinear extension is to perform simple preprocessing of training vectors x_i by mapping them into some feature space F and then applying the standard SV regression algorithm. Let $z = \phi(x)$ be such a mapping which maps x according to $\Phi : \chi \rightarrow F$. The assumption here is that $f(z)$ is linear in the feature space, i.e.,

$$\begin{aligned} f(z) &= \langle w, z \rangle + b \\ \text{where, } z &= \phi(x) \end{aligned}$$

Thus, the nonlinear SVR algorithm behaves like a linear one if the input vectors x_i are replaced by their corresponding feature vectors z_i. But, due to the exclusive dot product form in Eq. (2.10b), *it is not required to know the mapping* Φ *explicitly.* Rather, it is only required to know the dot product $\langle \phi(x_i), \phi(x_j) \rangle$, which can be represented by the kernel function

$$k(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle$$

This is commonly referred to as the "kernel trick" in the literature. Thus, for the nonlinear case we have:

$$\begin{aligned} w &= \sum_{i=1}^{l} (\alpha_i - \alpha_i^*) \phi(x_i) \qquad (2.11) \\ \text{thus,} \quad f(x) &= \sum_{i=1}^{l} (\alpha_i - \alpha_i^*) \langle \phi(x_i), \phi(x) \rangle + b \end{aligned}$$

For choosing the form of the kernel function $k(x_i, x_j)$, the requirement is that it should follow Mercer's condition (Mercer, 1909). Several kernel functions have been proposed which satisfy Mercer's condition (Vapnik, 1998). Table 2.1 lists a few commonly used kernels. The suitability of a kernel depends on the application and the choice is usually made on a trial & error basis. If it is known *a priori* that the process is linear then a linear kernel will suffice. For nonlinear processes, *a priori* knowledge about the type of nonlinearity can help in choosing the kernel. But usually this is not the case since very little knowledge is available about the structure of nonlinearity and hence the kernel choice is made by observing the prediction performance on validation data. The RBF kernel is a translation invariant kernel ($k(x, x') = k(x - x')$) and it is a very useful kernel whose use is

Table 2.1: Different types of Kernel functions.

Kernel type	Operation
Linear	$\langle x_i, x_j \rangle$
Polynomial	$(\langle x_i, x_j \rangle + c)^p$
Sigmoid	$\tanh(c + \gamma \langle x_i, x_j \rangle)$
Radial Basis Function(RBF)/Gaussian	$\exp(-\gamma \|x_i - x_j\|^2)$

quite widespread in applications of SVR cited in the literature (Smola & Scholkopf, 2003). The popularity of the RBF kernel is due to its practical success on many different nonlinear regression & classification problems. This is attributed to the property of RBF in that it is a "universal approximator". This means that RBF-based approximators can approximate any continuous function on a compact set with arbitrary accuracy. Park & Sandberg (1991) proved that RBF networks with one hidden layer are capable of universal approximation. Wang et al. (2004) have shown that an RBF kernel-based SVR also has the same property.

2.2.1 RBF-kernel SVR

The RBF network, which are a special class of neural networks, is one of the classical tools that is popular for developing nonlinear models. The activation functions in the RBF network are the radial basis functions which can be used as kernels (Gaussian kernel) in SVR. In the RBF network, the basis functions are centered at node centers which are either tuned manually for good validation results or found using some clustering technique. The number of such nodes and the parameters of the nodes have to be manually tuned along with the centers in the model building exercise, which can be a difficult task. The weights given to the nodes in the RBF network are computed as a solution to an optimization problem which is framed to minimize the model prediction errors.

The similarity between the RBF network model and the RBF-kernel SVR model can be realized if one visualizes the SVR model as a two-layered network, as shown in Figure 2.2 (Vapnik, 1998). The SVR model is essentially a weighted linear combination of the kernel function values evaluated at the support vectors, shown in Eq. (2.11). The support vectors are analogous to the node centers in the classical RBF network model. Both the weights and the number of nodes (support vectors) are automatically decided as the solution to the optimization problem in the SVR method. Thus, in contrast to the RBF network, an RBF-kernel based SVR model is much more easier to train; the node centers are automatically decided in SVR. In an interesting comparison of RBF-kernel SVM with the RBF network, Scholkopf et al. (1997) have demonstrated that the centers picked by the SVM algorithm give superior classification results on the US postal service handwritten digit recognition

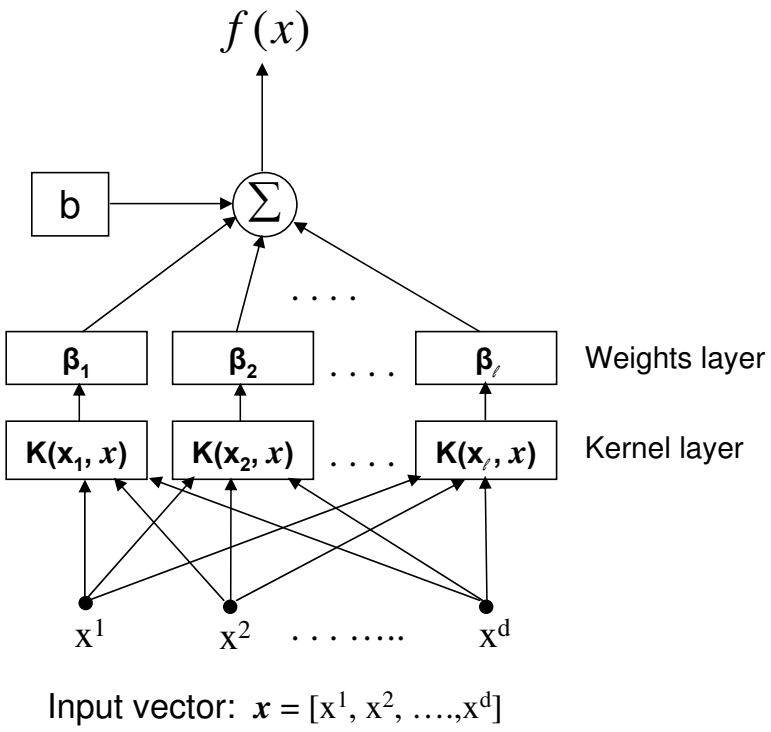


Figure 2.2: Two-layered network structure of SVR model.

data set which is a popular benchmark data set. The interpretation given by the authors is that the SVR algorithm ensures that the node centers (support vectors) are specifically suited to the data at hand. In the following sections, the application of RBF-kernel based SVR for developing models of real world processes which exhibit significant input-output nonlinearities will be demonstrated.

2.3 Soft Sensor Application

Soft sensors are inferential calculators which can output the value of a primary process variable or quality variable based on a model that captures its relationship with other measured secondary variables. In many cases the important process variables in process control systems are difficult or impossible to measure online due to the limitation of process technology or measurement techniques (Yan et al., 2004). These variables, which are sometimes the key indicators of process performance, have to be determined by off-line laboratory analysis (that typically introduce large time delays) or online product quality analyzers (which are expensive and have high maintenance cost). Soft sensors have become an attractive option over conventional hardware sensors or sensors based on laboratory analysis, because they do not suffer from the disadvantages mentioned above. They can give measurements at much faster sample rates than conventional hardware sensors and thus help in better

control. Joseph & Brosilow (1978) introduced the idea of inferential control of processes. They studied the use of several temperature and flow-rate measurements to estimate product composition in a distillation column. The estimator employs a linear combination of selected tray temperatures, steam and reflux flow rates to estimate product compositions. Such empirical models are the most popular ones used in soft sensors today.

In recent years, the use of SVR has attained wide popularity in the development of empirical models which can be solved as regression problems. Practical success has been demonstrated on both simulated data sets (Li et al., 2004; Rojo-Álvarez et al., 2004) and actual process data (Shi & Liu, 2006; Nandi et al., 2004; Vogt et al., 2004; Xi et al., 2007). The growing popularity can be mainly attributed to the ability of SVR in being able to tackle nonlinear regression problems through the use of kernels with relative ease. There is also considerable theoretical investigation on SVR in the literature. Pontil et al. (2000) have explicitly shown that the noise model corresponding to the ε-insensitive loss function is additive and Gaussian, where the variance and mean of the Gaussian density are random variables. The probability distribution of the mean and variance are also found explicitly; the variance is shown to follow a unimodal distribution that does not depend on ϵ and the mean to follow a distribution uniform in the interval $[-\epsilon, \epsilon]$. These probability distributions were justified by interpreting the objective function in Eq. (2.7a) from a Bayesian viewpoint. Cherkassky & Ma (2004) have proposed a practical analytical method for the choice of the SVR regression parameters (ϵ and C) directly from the training data. The authors suggest setting the value of ϵ proportional to the standard deviation of noise (σ, known or estimated from data) with the empirical dependency $\epsilon \sim \sigma\sqrt{\frac{\ln n}{n}}$, where n is the number of training samples. The setting for the C parameter is based on the assumption that the kernel is bounded in the input domain (e.g. RBF kernel) and is given by $C = \max(|\overline{y} - 3\sigma_y|, |\overline{y} + 3\sigma_y|)$ where $\overline{y}$ and σ_y are the mean and standard deviation of the y values of training data. Several modifications of the ϵ-SVR algorithm have also been proposed: Least Square SVM (for classification by Suykens & Vandewalle (1999), for regression by Saunders et al. (1998)) where the loss function is chosen as quadratic and inequality constraints are replaced by equality constraints; $\nu-$SVR (Scholkopf et al., 1998), where the ν parameter specifies *a priori* the maximum fraction of training samples to be support vectors and thus automatically minimizes ϵ, giving the best achievable accuracy for the chosen ν and C.

2.3.1 Melt index soft sensor

This section shows the application of SVR in the development of a soft sensor to predict the quality variable of polymer in an industrial polymerization plant based

on its dependency on a large number of other process variables. Ohshima & Tanigaki (2000) have given a comprehensive review of property estimation methods published for different polymerization processes. As evident from several papers listed in their work, the development of an on-line inferential system for polymer property is a very active research area in the field of polymerization reactor control. The majority of papers are reported to be using techniques such as Extended Kalman Filters (EKF), Artificial Neural Network (ANN), Nonlinear Parameter Estimation and Partial Least Squares (PLS) incorporating nonlinear models. Most of these methods use a mechanistic or physio-chemical model of the process. But the disadvantage of such approaches is the requirement of a good knowledge of the mechanism of polymerization process which is often very complex and the large computational effort required to come up with a meaningful set of model parameters. The Nonlinear SVR can be used to build a soft sensor in such situations where the output is nonlinearly related with the input variables. Han et al. (2005) present one of the first applications of SVR based soft sensor models for industrial polymerization processes. The quality variable of interest in their application was the melt index of the polymer in two polymerization plants (Styrene-Acrylo-Nitrile and Polypropylene). Process variables such as temperature, pressure, levels, which were measured more frequently than the melt index, were used as the dependent variables. The authors compare the performance of the SVR based model with other black-box models based on ANN & PLS and show that the SVR based model gives the best performance, especially when only a small sample size of data is available for the modeling exercise. Shi & Liu (2006) have shown the application of a weighted least square form of SVR to infer the melt index of polymer in a real propylene polymerization plant. The independent variables consist of upstream process variables of temperature, pressure and catalyst flow rate in the plant.

In the current work, a case study which demonstrates the application of SVR for soft sensing the melt index (MI) of polymer in an EVA polymerization plant will be presented. The data is obtained from the work by Alleyne (2006) at the AT Plastics Inc. site in Edmonton, Canada. The primary variable of interest is the melt index of the polymer after extrusion through an extruder installed downstream of the polymer reactor. In the previous work, an empirical model was built which related the polymer's melt index with the measured operating conditions of the extruder (Figure 2.3). The input data to the model consisted of extruder pressure (PI), extruder speed (SI) and temperature (TI) which are commonly measured on extruders. The online measurement for the melt index was determined by an online rheometer (AI-01). The significant issue, because of the large range of melt index (MI) measurement required, was that the rheometer required die changes and recalibration whenever a grade change occurred. This would result in the unit

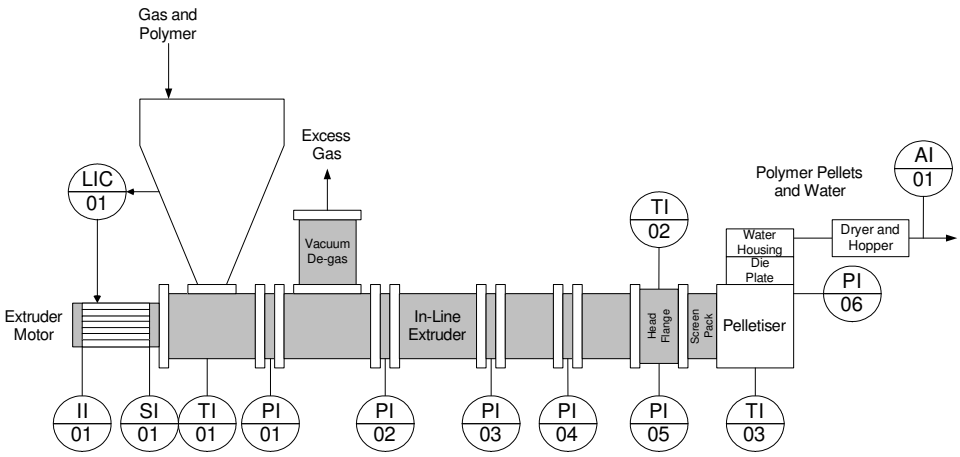


Figure 2.3: Extruder schematic.

having to be switched off or becoming unreliable for a short period while the die was being changed. Thus, it was clear that some new measurement device or sensor was required to give online polymer melt index values at regular time intervals. The empirical soft sensor used by Alleyne (2006) was based on the model in Eq. (2.12).

$$MI = \frac{\exp(a + b.S + c(\frac{S}{P^{\alpha}}) + d.P^{\alpha})}{T^2} \tag{2.12}$$

The parameters were found by nonlinear least square regression. In addition, a scheme for bias updating the soft sensor predictions was used. The bias in the soft sensor predictions was caused by fouling of the extruder and the die, which significantly changes the extruder operating conditions. An online bias was calculated every half an hour using the online analyzer reading. The laboratory measurements were available infrequently (generally every 2 hours or longer) and used to calculate laboratory bias. Both biases were combined (details given in Alleyne (2006)) and a unified bias update was applied to the soft sensor.

The Nonlinear Least Squares (NLS) approach had the disadvantages of slow training and requirement of good initial guesses for the parameters. The SVR tool can overcome these difficulties. It was also desirable to have a soft sensor that is less dependent on the bias update. Unlike the NLS, the SVR algorithm does not require any initial guesses and the convex nature guarantees the existence of a single optimal solution. It is only required to choose the tuning parameters, which can be easily done using the guidelines given in the literature, though some experience does help. Fast training of SVR could be achieved by the Sequential Minimal Optimization (SMO) algorithm, developed by Platt (1999) targeting the SVM and later extended to SVR by Flake & Lawrence (2002). The LIBSVM package (Chang & Lin, 2001) with the SMO implementation was used for training.

The objective was to build a single soft sensor model that can cover the entire

product range. The full data set was constructed by combining steady state plant measurements for around ten different polymer grades (25,793 samples at 1 minute sampling time). This covered a wide range of MI values over the range of 10-1000. The data was divided into training and validation set as shown in Figure 2.4. Note that the data represents steady state operation of the process and our objective is to build a corresponding steady state model covering the entire operating range. It is necessary to have a training data which includes at least a few points from each of the different steady state regions. Hence, the training and validation data were carefully constructed by ensuring that Melt Indices from all the operating regions are included in both subsets of the data as shown in Figure 2.4. Even though this results in merging of data from discontinuous time periods, the key assumption of steady state operation of the process makes this data preprocessing step a valid one.

The input or explanatory variables to the SVR based soft sensor comprised of 10 measured variables which monitor the extruder operating conditions: these were the 6 extruder pressure measurements (PI01-PI06), 3 temperature measurements (TI01-TI03), and the extruder screw speed (SI01). A log transformation was applied to the MI and given as target (or the dependent variable) to the SVR model. The RBF kernel was used for implicit nonlinear mapping of inputs to the feature space. Thus, the SVR model had the form shown below in Eq. (2.13).

$$\begin{aligned} \widehat{Y}(x) &= \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)\,\langle\phi(x_i),\phi(x)\rangle + b \qquad (2.13)\\ \text{where, } MI &= \exp(\widehat{Y}), \quad x = [\mathrm{PI}_{01-06}, \mathrm{TI}_{01-03}, \mathrm{SI}_{01}] \\ \langle\phi(x_i),\phi(x)\rangle &= \exp(-\gamma\,\|x_i - x\|^2) \end{aligned}$$

The SVR parameters were initially set using the guidelines mentioned in Cherkassky & Ma (2004), with some tuning to give good validation results. The parameter settings used were: $C = 100, \epsilon = 0.3, \gamma = 1e-6$. The SVR model predictions were compared with that of the NLS soft sensor on the validation data set. The Mean Square of Errors was used to assess model quality or performance. The raw predictions (without bias updating) from both models are shown in Figure 2.5. It can be observed that the predictions are able to track most of the global trends occurring in the MI reading. The high MSE of predictions can be attributed to an offset between the predictions and the actual values which is more or less constant within each grade. Thus, it is clear that bias updating would certainly improve the predictions for both soft sensors. It is not meaningful to compare the MSEs of the two sensors at this point as the raw predictions suffer from the bias problem.

As mentioned earlier, one of the objectives was to decrease the dependence of the soft sensor on the bias updating, i.e., to use less frequent bias updating than the existing 30 mins. update. To achieve this objective, a bias update was performed

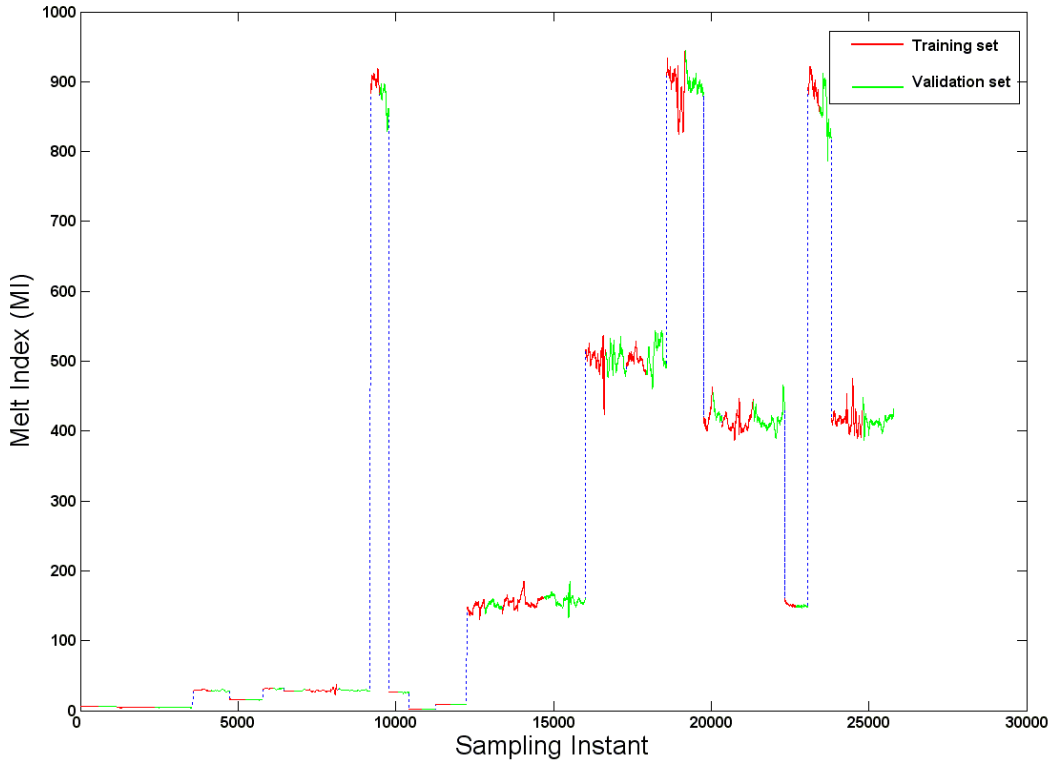


Figure 2.4: Online analyzer (AI01) MI data: Showing division of data into training and validation set.

every 8 hours and the predictions from both the soft sensors were compared as shown in Figure 2.6. As evident from the figure, the predictions from both soft sensors improve considerably with the bias update. Comparing the MSE of the prediction, it can be observed that the SVR based soft sensor performs better than the NLS soft sensor for the 8 hrs. bias updating. Noting that laboratory measurements of MI are usually available at least once for every 8 hrs., the SVR model can be used solely with lab measurement based bias updating for the entire range of grades. This would remove the issues of die changing and recalibration in the online analyzer during grade changes, both of which hinder the smooth running of the process. Another solution is to provide samples to the online analyzer every 8 hrs. and use the readings for bias updating. This gives several advantages: lab measurements can be completely avoided which naturally gives cost benefits; reduced measurement load on the on-line rheometer decreases the rate of fouling of the rheometer die making it reliable for longer periods; the 8 hrs. idle time in the rheometer can be utilized to make any die changes, without hindering the MI measurement which is always available from the soft sensor.

Remarks. *In light of the above industrial case study, there are a few comments*

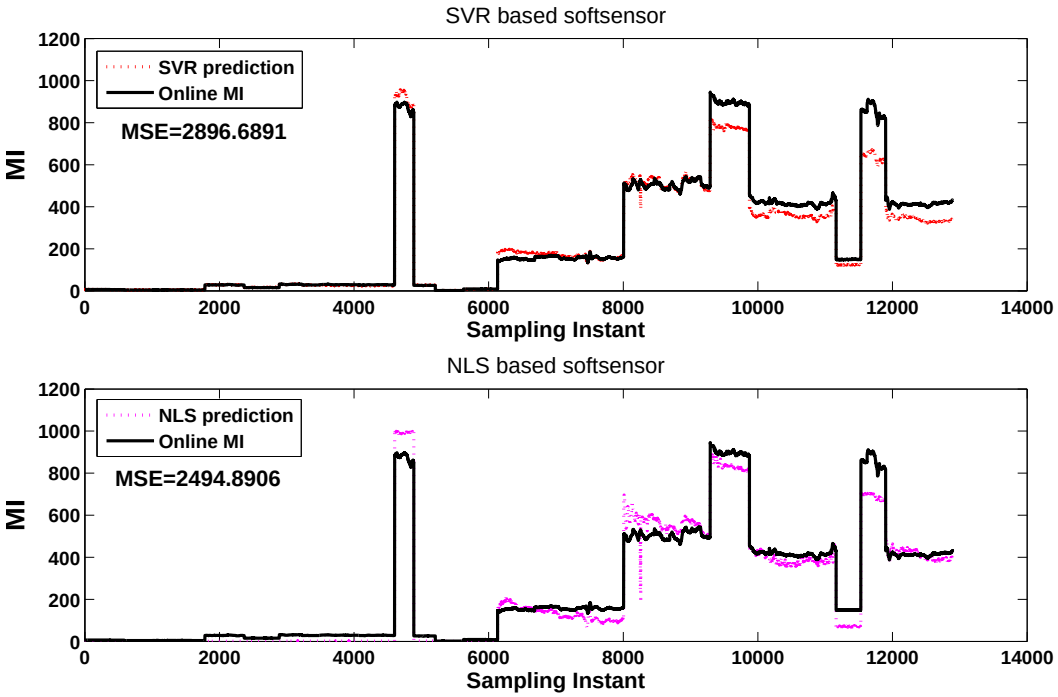


Figure 2.5: Soft sensor validation: Raw predictions (without bias updating).

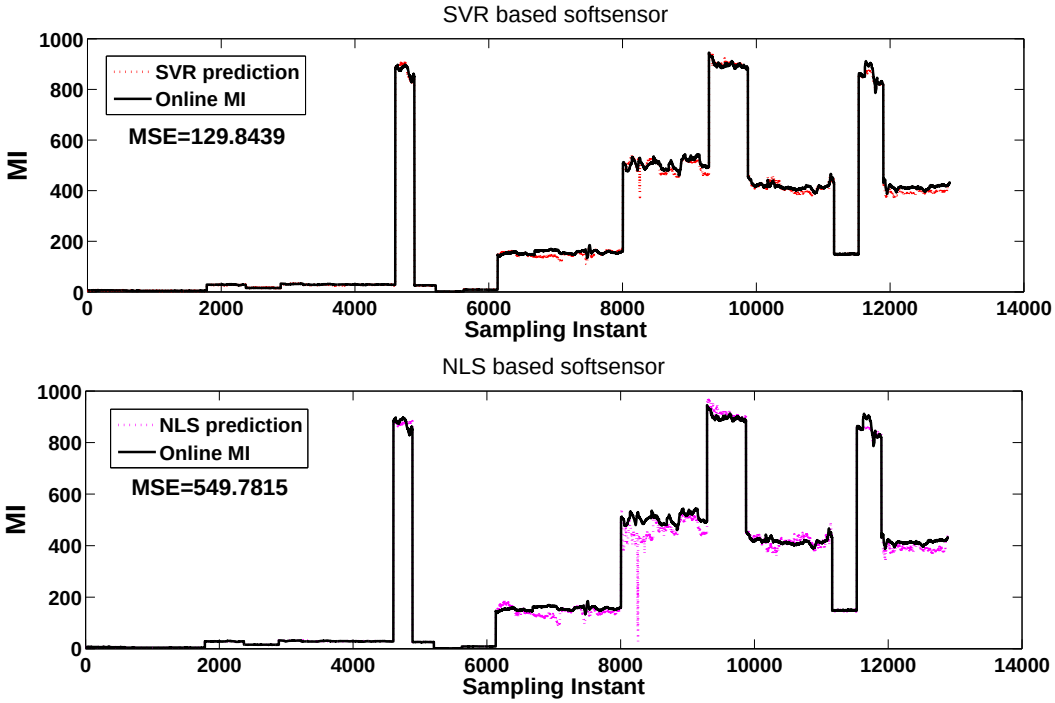


Figure 2.6: Soft sensor validation: 8 hrs bias updated predictions.

we would like to make regarding the success of the SVR soft sensor after the bias updating procedure was incorporated:

The success of the bias updating procedure will depend on the level of variation in the bias error with time. If the bias error varies very frequently, then a more frequent bias updating will be required to obtain a lower MSE. From Figure 2.5, we can see that the bias error in the SVR model predictions stays at a fairly constant value within each melt index grade. On the other hand, the NLS model predictions have bias errors which are fluctuating highly within each grade (often fluctuating between positive and negative values as can be observed from Figure 2.5). Hence, applying the bias updating procedure will prove more effective on the SVR model than the NLS model. Ideally a soft sensor model should be free from such bias errors; if not, at least the error should be a constant value which is easy to correct for without requiring re-calibration. In this comparative case study, the lower variation of the bias in the SVR soft sensor model makes it superior to the NLS soft sensor after the bias correction is incorporated.

2.4 Nonlinear Dynamic System Identification

From the case study on EVA polymerization plant, the application of SVR for soft sensor design based on steady state data from a nonlinear plant was clear. Here, the dynamic case is considered where the plant output is a nonlinear function of past outputs and inputs. The utility of the SVR algorithm for such problems is shown by constructing the input vector x appropriately. Assume that the dynamic plant can be approximated by the nonlinear ARX structure (NARX) of the following form:

$$y(t) = f(x_t) + \epsilon \tag{2.14}$$

$$x_t = [y(t-1:t-na), u_1(t-d_1:t-d_1-nb_1+1), .., u_p(t-d_p:t-d_p-nb_p+1)] \tag{2.15}$$

$$where,\ y(t-1:t-na) = \mathbb{R}^{1\times n_a},\ u_i(t-d_i:t-d_i-nb_i+1) = \mathbb{R}^{1\times n_{b_i}}$$

In Eq. (2.14), f is some unknown nonlinear function which will be approximated through regression on plant data; $(na, nb_{1\ to\ p})$ are model order parameters, $d_{1\ to\ p}$ are input delays and ϵ is the unknown additive noise contaminating the observed plant output. Here na and nb_i are parameters to be tuned to get good prediction results on the validation data set. Also, the SVR parameters have to be tuned in a similar fashion. In this work, the use of the SVR algorithm for nonlinear system identification is proposed, employing the procedure described in the flowchart shown in Figure 2.7 which is similar to the generic system identification loop given in Ljung (1999).

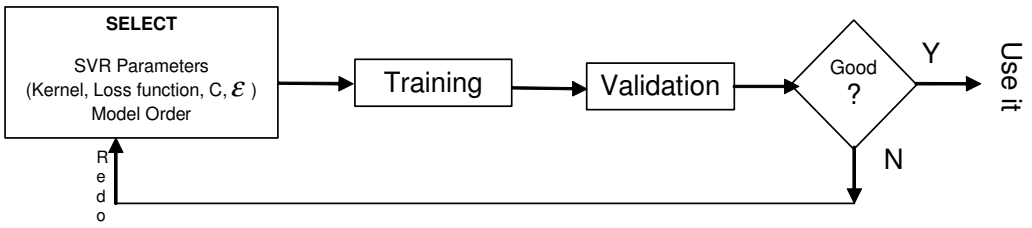


Figure 2.7: SVR based nonlinear system identification loop.

2.4.1 Simultaneous delay and order selection by a heuristic procedure

Based on the flowchart in Figure 2.7, an algorithm is proposed, which is more heuristic in nature, in order to automate the task of finding the optimal order and delay parameters. In all the subsequent discussions, the variables d and nb are used to denote a vector containing the respective delays and orders of all the inputs. The usual method of finding a suitable model order is to test ranks of sample covariance matrices or to use correlation plots for a range of lags which are manual procedures. The delay parameters are usually found from the step input on the open loop process. Here a unique heuristic approach is taken where a suitable choice of both delay and orders are made automatically from the identification data. The basic idea is simple: search for the best model (in terms of model fit) from a set of models with various delays and orders. The identification data available from the process can be used in all the steps of this algorithm. First, the data is divided into estimation and validation set. The input required from the user is a range of values of orders and delays to search for. While finding the best model, the parsimony of the model is also taken care of. A step by step approach is taken in this heuristic procedure as follows:

Step 1: Choose an appropriate kernel (linear/ nonlinear) based on the process knowledge.

Step 2: Find reasonable parameters (C, ϵ and kernel parameters) that give good fit on the validation data set for some arbitrary model orders. For the RBF kernel based SVR, it is difficult to know *a priori* which C and γ parameter values would give good prediction accuracy on the validation data set. Hence, a grid search method can be employed to arrive at a good pair of C and γ parameter values (Bao & Liu, 2006; Hsu et al., 2004). The search objective would be, for example, to minimize the MSE of predictions on the validation data set. For the ϵ parameter, a practical guideline is to choose a value proportional to the standard deviation of the measurement noise in the output variable (Cherkassky & Ma, 2004). Steps 1 and 2 are mostly coupled together and will require some experience with tuning of SVR parameters. Also, some prior knowledge of the process such as whether it is

linear or nonlinear will also help.

Step 3: Find delay: Fix upper bounds, D and B, below which the delay and input order is expected to lie. The lower bound is set at 1 for both delay and input orders. Using the above parameters and $na = 1$, estimate the delay as follows:

For $d = 1 \ldots .D$ train SVR models on the estimation data by using input order of $nb = 1 \ldots .B$ and output order of $na = 1$. Also, perform *infinite horizon predictions (simulation)* on the validation data and build a matrix of R^2 values of the predictions which is given by

$$R^2 = 1 - \frac{\sum_{i=1}^{N} (y_i - \widehat{y}_i)^2}{\sum_{i=1}^{N} (y_i - \overline{y})^2}$$

The R^2 value measures the model fitness by comparing the predictions with the measured output. Note that the R^2 value will be less than unity and hence can be used to compare the goodness of fit of the predictions from different models. The matrix of R^2 values is given by

$$J = \begin{bmatrix} R^2_{11} & R^2_{12} & . & . & . & R^2_{1D} \\ R^2_{21} & R^2_{22} & . & . & . & R^2_{2D} \\ . & . & . & R^2_{nb_i,d_j} & . & . \\ R^2_{B1} & R^2_{B2} & . & . & . & R^2_{BD} \end{bmatrix}$$

Then, estimate delay as follows:

$$d_{\max} = \arg\max_{d} \frac{1}{B} \sum_{i=1}^{B} R^2_{id}$$

, i.e., the column of J which has the maximum mean goodness of fit value gives the delay estimate.

For the Multi-Input case D and B will be vectors, rather than scalars, which contain the delay and order upper bounds for each of the inputs. Then d_j and nb_i will consist of all the possible combinations of delay and input orders which lie within the bounds. The computational cost will be high because many combinations of delays and nbs have to be tried out. A significant amount of computational cost can be saved by using a smaller set of nbs. The maximum of average model fit can be expected to occur at the correct delays even if the model orders are not optimal.

Step 4: Find optimal order (na, nb)

Specify A as the upper bound of na to search for. The lower bound is set at 1, similar to the previous step. Increase the B values to include more nbs, if required. For $delay = d_{max}$, build a matrix of R^2 values similar to the previous case for

$nb = 1, 2, \ldots..B$ and $na = 1, 2, \ldots..A$ as follows:

$$H = \begin{bmatrix} R^2_{11} & R^2_{12} & . & . & . & R^2_{1B} \\ R^2_{21} & R^2_{22} & . & . & . & R^2_{2B} \\ . & . & . & R^2_{na_i,nb_j} & . & . \\ R^2_{A1} & R^2_{A2} & . & . & . & R^2_{AB} \end{bmatrix}$$

The best order is given by the order $[na_{max}, nb_{max}]$ which maximizes the $R^2_{na_i,nb_j}$, i.e.,

$$\begin{aligned} [i^* \ j^*] &= \arg\max_{i,j} H_{i,j} \\ where, H_{i,j} &= R^2_{na_i,nb_j} \\ [na_{max}, nb_{max}] &= [na_{i^*}, nb_{j^*}] \end{aligned}$$

But the best model orders are usually high and this procedure will result in a very high order model. In order to obtain a parsimonious model, a model with a low order is required such that the predictions are within tolerable errors, i.e., R^2 does not decrease significantly. The *optimal orders* are therefore orders which are lower than $[na_{max}, nb_{max}]$ with R^2 higher than a defined threshold. The threshold can be implicitly defined as a percentage reduction from the maximum R^2 value obtained above. For the multi-input case, the sum of input order values given by the one-norm $|\ .|_1$ can be used to define parsimony w.r.t the input order. Thus the optimal orders are given by:

$$[na_{opt}, nb_{opt}] = \arg\max_{na_i,nb_j} \{H_{i,j} \ \forall \quad i,j : na_i < na_{i^*}, |nb_j|_1 < |nb_{j^*}|_1, \frac{H_{i^*,j^*} - H_{i,j}}{H_{i^*,j^*}} \leq 10\%\}$$

, i.e., a decrease of 10% of the best fit can be tolerated when order is decreased from $[na_{max}, nb_{max}]$ to $[na_{opt}, nb_{opt}]$.

Step 5: Choose the optimal model

The required SVR model is the one corresponding to the model parameters $[na_{opt}, nb_{opt}, d_{max}]$. The optimal model will be used for making predictions.

2.4.2 Case Study 1: Identification of a simulated pH neutralization process

The above described heuristic procedure was tested on data from a simulated pH neutralization process in a CSTR. The simulated data set is taken from the DaISy database (Database for Identification of Systems, De Moor). This is an example of a highly nonlinear process. The data was originally generated by simulation of the nonlinear dynamic model given in McAvoy et al. (1972) for the reaction between Acetic acid and Sodium hydroxide. The settings used for the simulation are shown in Table 2.2.

Table 2.2: Simulation settings for the pH neutralization process.

CSTR Volume	1100 l
Concentration of the acid solution (CH_3COOH)	0.0032 Mol/l
Concentration of the base solution ($NaOH$)	0.05 Mol/l
Sampling Time	10 $secs$

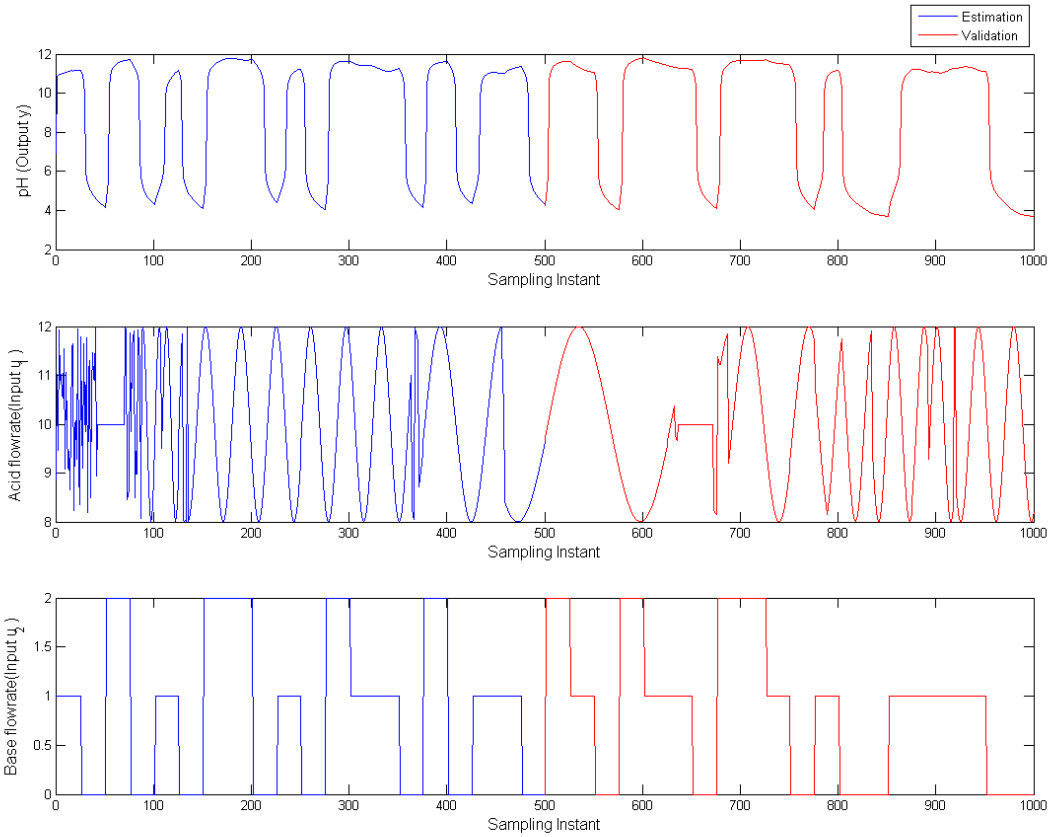


Figure 2.8: Simulated pH neutralization in a CSTR (showing division of data into estimation and validation set).

Figure 2.8 shows the plot of the full data set used for this case study along with the division into estimation and validation set. The output y is the pH of the solution inside the reactor. The inputs to the process are u_1=Acid flow rate and u_2=Base flow rate.

Results

The heuristic procedure was applied on the training data set to arrive at a suitable dynamic model. A log transform of the pH measurements was applied on the output, i.e., the target variable was $y = \log(\text{pH})$. The RBF kernel was used for kernel mapping of inputs (Step 1). The SVR parameter settings of $C = 10, \epsilon = 0.01, \gamma = 0.1$ were chosen based on satisfactory predictions on the validation data set (Step 2).

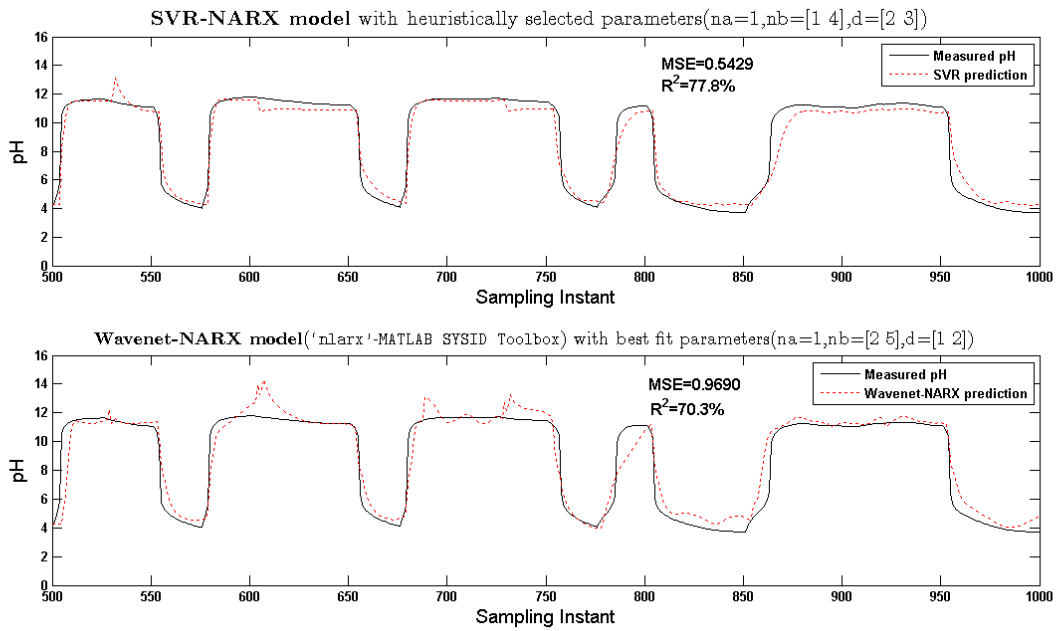


Figure 2.9: Validation of SVR-NARX model with the heuristic model parameter selection.

For finding appropriate input delay parameters (Step 3), the upper bounds on delay (D) and input order (B) were set as 5. The output order upper bound (A) was set as 10. The delay was estimated as 2 and 3 sampling instants for the two inputs respectively in Step 3. The output of Step 4 was as follows: $na = 1, nb = [1, 4]$. The optimal SVR-NARX model (Step 5) was validated using infinite horizon predictions on the validation data set.

For comparison with a benchmark, another NARX model was built using tools available in the "ident" GUI for System Identification in MATLAB developed by Ljung and co-workers, which uses different algorithms such as neural network, wavelet network etc. (Ljung et al., 2006). The Wavelet Network (wavenet) structure available in the toolbox was used for modeling the nonlinearity since it gave better results compared to neural and sigmoid networks. Wavenet-NARX models were first built with orders and delays over the range used in the heuristic method. The model with the best fit was chosen to compare with the SVR-NARX model. The validation results are shown in Figure 2.9. Comparing the MSE and R^2 values of the Wavenet-NARX model, we can conclude that the SVR-NARX model gives better predictions than the chosen benchmark. In order to evaluate the efficacy of the delay estimation step, the predictions were compared with that of a SVR-NARX model developed with a zero-order hold assumption, i.e., unit delay in both the inputs. The R^2 value of the SVR model based on the heuristic procedure was approximately 78%, where as that of the model based on zero-order hold assumption was much lower at 62%.

2.4.3 Case Study 2: Black-box modeling of a twin screw extrusion process

The MI soft sensor described earlier was developed using steady state data in a single screw extruder. In this section, the dynamic case is considered where an SVR based nonlinear black-box model is developed to capture the dynamics of a twin screw extrusion process. Similar to the case of single screw extruders, one of the purposes of twin screw extrusion is to compound a mixture of chemically different polymer materials in a controlled temperature and pressure environment so that the resulting polymer blend attains certain desired properties. A good overview of the twin screw extrusion process is given in Janssen (1977). The twin screw extrusion process differs from the single screw process in the mechanism of transport that is achieved with the rotating action of two parallel screws, as opposed to a single screw, in a closely fitting barrel. The twin screw extruder is widely used for blending applications in the polymer and the food industry (Iqbal et al., 2010).

The data used for this work was collected from a laboratory scale co-rotating twin screw extruder at the Department of Chemical and Materials Engineering, University of Alberta. A schematic diagram of the experimental setup is shown in Figure 2.10. The data was taken from the work of Iqbal et al. (2010), where a gray-box linear model was developed to capture the extruder dynamics. The feed to the extruder consisted of a binary mixture of High Density Polyethylene (HDPE) with different melt indices. The quality variables that were of interest were the melt index and the rheological properties of the blended polymer. But these variables were not measured online in this experimental setup. Hence for controlling these variables indirectly, two process variables were selected, namely the Melt Temperature (T_{melt}) and Pressure (P_{melt}) at the extruder die, which were found to have high correlation with the quality variables based on off-line sample analysis. The manipulated variables were the screw speed (N) and the feed rate (F), which are typical choices for control applications of the extrusion process (Janssen, 1977). The screw speed has a faster effect on the output response compared to the feed rate. In this study, the univariate relationship between the Melt Temperature and the screw speed, which has significant nonlinearity, is considered in order to show the efficacy of the methodology.

The previous work conducted on this data was based on a gray-box approach, where the physical knowledge about the dependence of T_{melt} on the screw speed was used to first capture the nonlinearity. A power law relationship was established based on steady state analysis of the output and input variables. For the dynamic model building, a power law transformation of the screw speed variable was performed to build an ARMAX model between the transformed input variable and T_{melt}. In the current work, the black-box route was taken where the identification data collected

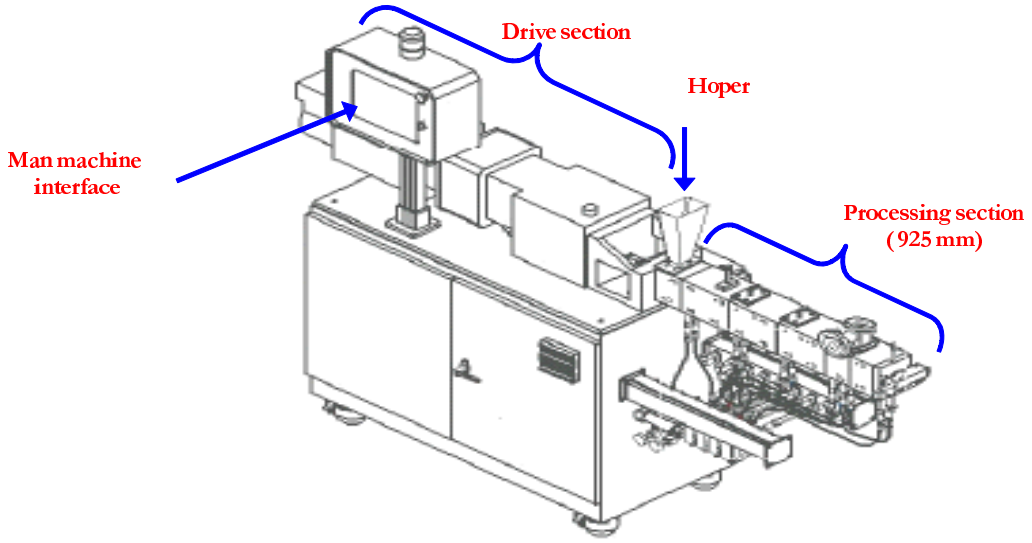


Figure 2.10: Twin screw extruder schematic.

from the process was used to directly arrive at a NARX model using the SVR technique. Based on the NARX structure shown in Eq. (2.14), we have $y \equiv T_{melt}$, $u \equiv N$. The heuristic methodology that was explained earlier was used to arrive at a model with suitable order and SVR parameters. The RBF kernel based nonlinear mapping was found to be appropriate.

Results

The input-output data collected from the extrusion process for a fixed value of feed rate is shown in Figure 2.11. The variables shown in this figure are in their detrended form which were used for model building and validation. The first half of the data was used as training data for the model building exercise in the heuristic procedure. As shown in the figure, a random binary variation of the screw speed was used to excite the process. The mean residence time of this extruder, which can be considered as the dominant time constant, was 98 sec (Iqbal et al., 2010). The measurements were initially recorded at a sampling rate of 1 sec and then down sampled to 10 sec sampling time. This sampling time will result in approximately 10 samples for one process time constant interval which is reasonable to capture the dynamics of the process. The SVR parameter settings of $C = 10, \epsilon = 0.05, \gamma = 0.05$ was used. In order to search for the optimal values of the order (na, nb) and delay, the range of values in the interval [1 3] was used. Application of the heuristic procedure resulted in the optimal order values of $na = 1, nb = 1$ and delay of 1. Figure 2.11 also shows the infinite horizon predictions (or simulations) using the model in order to validate the model. The goodness of fit of the model predictions

or the R^2 value was 98.2% which clearly indicates the validity of the model. The linear model developed in Iqbal et al. (2010) was reported to have an R^2 value of 88.5%, which shows that the SVR based model gives better predictions. Since the SVR based model gives good predictions in the simulation mode, it is well suited to be used in a typical model based control scheme.

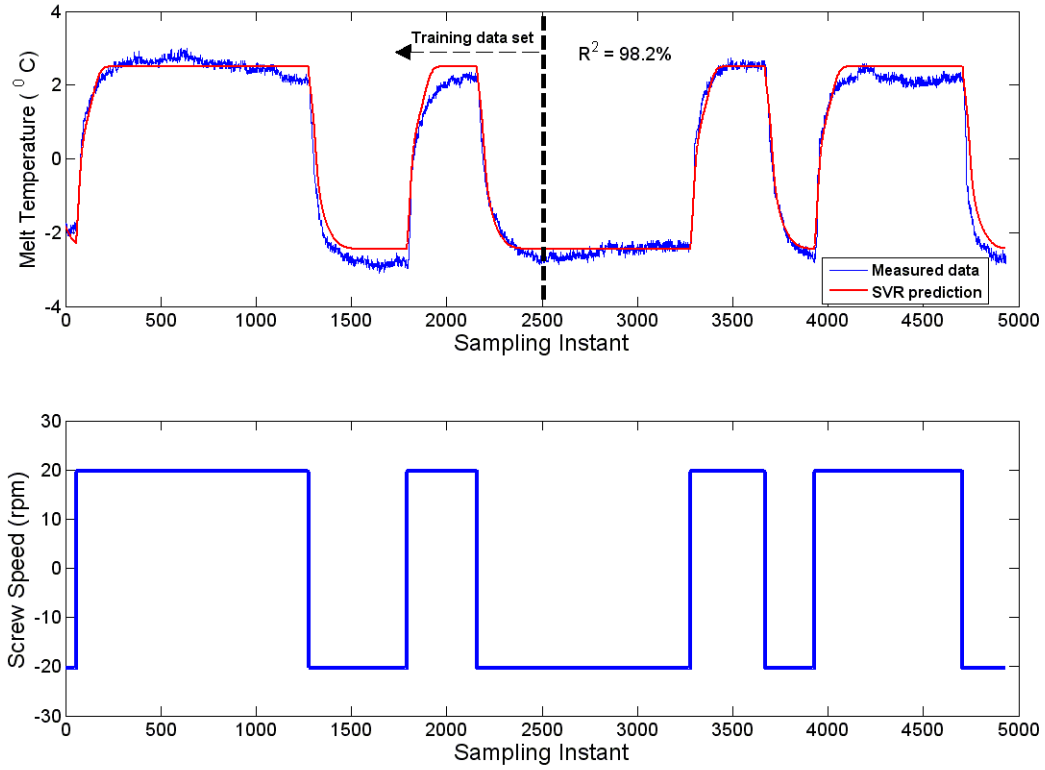


Figure 2.11: SVR model validation: Infinite horizon prediction on full data set.

2.5 Concluding Remarks

This chapter presented the application of the Support Vector Regression technique for designing soft sensors for nonlinear processes. The technique has several advantages over other nonlinear black-box modeling techniques such as the convex nature of optimization, few tuning parameters (compared to a neural network) and good generalization capabilities. For SVR-based nonlinear regression, there are many different types of kernel functions that can be chosen. It is difficult to tell *a priori* which kernel function would be appropriate for a given data set. A suitable kernel function has to be chosen using cross-validation of the trained model on a validation data set. The main difficulty is that for a given kernel function, the cross-validation results will highly depend on the values of the kernel parameters as well as the C and ϵ parameter values. Hence, a search for the optimal parameters

has to be performed for a given kernel function. The parameter search can be done using a grid search based method such as the one given in Bao & Liu (2006). The difficulty in choosing a feasible kernel function and the corresponding parameters is one downside of the SVR technique.

In this chapter, the efficacy of the SVR method for developing soft sensors was demonstrated through a Melt Index soft sensor for an industrial scale EVA polymerization plant. The SVR based regression technique was used to combine the information in the plant data, which covered several grades of polymers, into a single soft sensor model. The predictions from the SVR based model were shown to be more accurate than the previously installed nonlinear least square soft sensor in the plant. The application of the technique for developing nonlinear dynamic models was demonstrated through two case studies: a simulated pH neutralization process and a laboratory scale twin screw extrusion process. The optimal model order and delay vectors, which are required to define the dynamic relationships in the input-output model structures, were chosen using an automated heuristic procedure taking into account the aspects of goodness of fit and parsimony of the model.

Bibliography

Alleyne, I. R. (2006). *Optimal Grade Transition Policies for a High Pressure EVA Polymerization Plant*. PhD thesis, University of Alberta.

Bao, Y. & Liu, Z. (2006). A fast grid search method in Support Vector Regression forecasting time series. In E. Corchado, H. Yin, V. Botti, & C. Fyfe (Eds.), *Intelligent Data Engineering and Automated Learning* (pp. 504–511). Springer.

Chang, C.-C. & Lin, C.-J. (2001). *LIBSVM: a library for Support Vector Machines*. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Cherkassky, V. & Ma, Y. (2004). Practical selection of SVM parameters and noise estimation for SVM regression. *Neural Networks*, *17*(1), 113–126.

Chitralekha, S. B. & Shah, S. L. (2010a). Application of Support Vector Regression for developing soft sensors for nonlinear processes. *The Canadian Journal of Chemical Engineering*, *88*(5), 696–709.

Chitralekha, S. B. & Shah, S. L. (2010b). Support Vector Regression for soft sensor design of nonlinear processes. In *18th Mediterranean Conference on Control Automation, Marrakech, Morocco*, (pp. 569 – 574).

De Moor, B. L. R. (n.d.). *DaISy: Database for the Identification of Systems*. Department of Electrical Engineering, ESAT/SISTA, K.U.Leuven, Belgium.

http://homes.esat.kuleuven.be/~smc/daisy/,[Used dataset: pH Data, section: Process Industry Systems, code number:96-014].

Drucker, H., Burges, C., Kaufman, L., Smola, A., & Vapnik, V. (1997). Support Vector Regression Machines. In *Advances in Neural Information Processing Systems - Proceedings of the 1996 Conference*, volume 9, (pp. 155–161). MIT Press, Cambridge, MA.

Flake, G. W. & Lawrence, S. (2002). Efficient SVM regression training with SMO. *Machine Learning*, *46*(1-3), 271–290.

Han, I.-S., Han, C., & Chung, C.-B. (2005). Melt index modeling with Support Vector Machines, partial least squares, and artificial neural networks. *Journal of Applied Polymer Science*, *95*, 967–974.

Hsu, C.-W., Chang, C.-C., & Lin, C.-J. (2004). A practical guide to Support Vector classification. Technical report, Department of Computer Science and Information Engineering, National Taiwan University. Available at http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf.

Huber, P. J. (1964). Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, *35*(1), 73–101.

Iqbal, M. H., Sundararaj, U., & Shah, S. L. (2010). A new approach to develop dynamic grey box model for a plasticating twin screw extruder. *Industrial and Engineering Chemistry Research*, *49*(2), 648–657.

Janssen, L. P. B. M. (1977). *Twin screw extrusion*. Elseviers Scientific Publishing Company, New York.

Joseph, B. & Brosilow, C. B. (1978). Inferential control of processes: Part I. Steady state analysis and design. *AIChE Journal*, *24*(3), 485–492.

Li, H., Zhu, X., & Shi, B. (2004). Nonlinear identification based on least squares Support Vector Machine. In *8th Control, Automation, Robotics and Vision Conference (ICARCV 2004)*, volume 3, (pp. 2331–2335 Vol.3).

Ljung, L. (1999). *System Identification:Theory for the User*. Prentice Hall PTR, Upper Saddle River, NJ.

Ljung, L. (2006). Identification of Nonlinear Systems. In *9th International Conference of Control, Automation, Robotics and Vision (ICARCV 2006),*. IEEE.

Ljung, L., Zhang, Q., Lindskog, P., Juditsky, A., & Singh, R. (2006). An integrated system identification toolbox for linear and non-linear models. In *Proc. 14th IFAC Symposium on System Identification, Newcastle,Australia*.

McAvoy, T. J., Hsu, E., & Lowenthal, S. (1972). Dynamics of pH in controlled stirred tank reactor. *Industrial & Engineering Chemistry Process Design and Development*, *11*(1), 68–70.

Mercer, J. (1909). Functions of positive and negative type, and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society of London. Series A*, *209*, 415–446.

Nandi, S., Badhe, Y., Lonari, J., Sridevi, U., Rao, B., Tambe, S. S., & Kulkarni, B. D. (2004). Hybrid process modeling and optimization strategies integrating neural networks/support vector regression and genetic algorithms: study of benzene isopropylation on hbeta catalyst. *Chemical Engineering Journal*, *97*(2-3), 115–129.

Ohshima, M. & Tanigaki, M. (2000). Quality control of polymer production processes. *Journal of Process Control*, *10*(2-3), 135–148.

Park, J. & Sandberg, I. W. (1991). Universal approximation using radial-basis-function networks. *Neural Computation*, *3*(2), 246–257.

Pearson, R. K. (2006). Nonlinear empirical modeling techniques. *Computers and Chemical Engineering*, *30*, 1514–1528.

Platt, J. C. (1999). Fast training of Support Vector Machines using sequential minimal optimization. In B. Schölkopf, C. J. Burges, & A. J. Smola (Eds.), *Advances in Kernel Methods: Support Vector Learning* (pp. 185–208). MIT Press, Cambridge, MA.

Pontil, M., Mukherjee, S., & Girosi, F. (2000). On the noise model of Support Vector Machines Regression. In *Algorithmic Learning Theory, 11th International Conference, ALT 2000, Sydney, Australia, December 2000, Proceedings*, volume 1968, (pp. 316–324). Springer.

Rojo-Álvarez, J. L., Martínez-Ramón, M., de Prado-Cumplido, M., Artés-Rodríguez, A., & Figueiras-Vidal, A. R. (2004). Support Vector Method for robust ARMA system identification. *IEEE Transactions*, *52*(1), 155–164.

Saunders, G., Gammerman, A., & Vovk, V. (1998). Ridge regression learning algorithm in dual variables. In *Proc. 15th International Conf. on Machine Learning*, (pp. 515–521). Morgan Kaufmann, San Francisco, CA.

Scholkopf, B., Bartlett, P., Smola, A., & Williamson, R. (1998). Support Vector Regression with automatic accuracy control. In Niklasson, L., Boden, M., & Ziemke, T. (Eds.), *Proceedings of Eighth International Conference on Artificial Neural Networks*, (pp. 111–116).

Scholkopf, B., Sung, K.-K., Burges, C., Girosi, F., Niyogi, P., Poggio, T., & Vapnik, V. (1997). Comparing Support Vector Machines with Gaussian kernels to radial basis function classifiers. *IEEE Transactions on Signal Processing*, *45*(11), 2758–2765.

Shi, J. & Liu, X. (2006). Melt index prediction by weighted least squares Support Vector Machines. *Journal of Applied Polymer Science*, *101*(1), 285–289.

Smola, A. J. & Scholkopf, B. (2003). A tutorial on Support Vector Regression. Technical Report NC2-TR-1998-030, NeuroCOLT2, ESPRIT working group on Neural and Computational Learning Theory, Royal Holloway College, University of London.

Suykens, J. A. K. (2009). Support Vector Machines and kernel-based learning for dynamical systems modelling. In *Proceedings of the 15th IFAC Symposium on System Identification, Saint-Malo, France*, (pp. 1029–1037).

Suykens, J. A. K. & Vandewalle, J. (1999). Least squares Support Vector Machine classifiers. *Neural Processing Letters*, *9*(3), 293 – 300.

Vapnik, V. N. (1998). *Statistical Learning Theory*. John Wiley & Sons, New York.

Vapnik, V. N. & Chervonenkis, A. (1974). *Theory of Pattern Recognition (In Russian)*. Nauka, Moscow.

Vogt, M., Spreitzer, K., & Kecman, V. (2004). Identification of a high efficiency boiler by Support Vector Machines without bias term. In *System Identification 2003: A Proceedings Volume from the 13th IFAC Symposium on System Identification, Rotterdam, The Netherlands, 27 - 29 August 2003*, volume 1, (pp. 465–471).

Wang, J., Chen, Q., & Chen, Y. (2004). RBF kernel based Support Vector Machine with universal approximation and its application. In Yin, F., Wang, J., & Guo, C. (Eds.), *Advances in Neural Networks - ISNN 2004, Part I*, (pp. 512–517). Springer.

Xi, X.-C., Poo, A.-N., & Chou, S.-K. (2007). Support Vector Regression model predictive control on a HVAC plant. *Control Engineering Practice*, *15*(8), 897–908.

Yan, W., Shao, H., & Wang, X. (2004). Soft sensing modeling based on Support Vector Machine and Bayesian model selection. *Computers and Chemical Engineering*, *28*(8), 1489–1498.

Chapter 3

A Tutorial Overview of Sequential Monte Carlo Filters[1]

3.1 Introduction

The problem of state estimation and filtering has been one of the most active area of research in applied mathematics since the 1960's, when the Kalman filter was proposed by Kalman (1960). The Kalman filter algorithm is known to be the optimal filtering algorithm for linear systems with Gaussian process and measurement noise. Most of the filters that are used in real systems today are either direct implementation of the Kalman filter or practical extensions of it which are tailored to handle nonlinearity in the state and measurement equations. Some of the most popular extensions of the Kalman filter widely adopted in the research community are the Extended Kalman Filter (EKF), the Ensemble Kalman Filter (EnKF, Evensen (1994)), and the Unscented Kalman Filter (UKF, Julier & Uhlmann (1997)). Another class of nonlinear state estimation algorithm, which arose from a totally independent idea of using Monte Carlo methods for state estimation, are the ones based on the Particle Filter (PF) algorithm proposed by Gordon et al. (1993). A detailed overview of nonlinear state estimation algorithms, along with some of their practical applications, is given in Ristic et al. (2004). This chapter presents a tutorial overview of the Particle filter and the EnKF, which are both Monte Carlo simulation based filters. A comparative study of these filters and their relationship with the traditional Kalman filter is performed using simple examples of state space systems.

The chief objective of state estimation and filtering is to optimally fuse the data obtained through various sensors attached to the process in order to get an accurate estimate of the true state of the process (Julier & Uhlmann, 1997). Algorithms which perform this task are broadly termed as state estimation/filtering algorithms.

[1] A significantly abbreviated section of this chapter was presented at the IFAC Workshop on Automation in Mining, Mineral and Metal Industry, Viña del Mar, Chile, October 14 – 16, 2009.

A good introductory account on the topic of state estimation and filtering is given in Maybeck (1979). Figure 3.1 illustrates the general structure of the state estimation technique, as applied to process plants. The most important ingredient of any state estimator is a mathematical model of the process, which is usually developed based on physical insights and by conducting identification experiments. The model used in state estimators is always a stochastic system model which can account for the various uncertainties in our knowledge about the process, at least to some extent. These uncertainties, which are in fact the main reason for using a filter, result from mainly three sources (Maybeck, 1979). Firstly, the mathematical model we use to describe the process is always an approximate one, capturing only its most dominant characteristics known to us at the model development stage. The characteristics we assign to a process are usually based on past observations and fundamental laws of science. Secondly, the inputs affecting the process are never known completely and hence the effect of these unknown inputs (or disturbances) cannot be modeled. Finally, the measurement devices cannot give the perfect information we need, because they are always corrupted by noise and sometimes do not even exist for some process variables due to inherent measurement or process limitations. As a result of the factors mentioned above, random variables belonging to an appropriate probability distribution are included as part of the models to account for the uncertainties. As a result of these stochastic variables being present in the models, they are called 'stochastic' models.

3.2 Stochastic State Space Models

We briefly explain here the concept of stochastic state space models in a generic mathematical framework which will be used henceforth. The state space model of a process comprises of a state transition equation and measurement equation. We consider the discrete stochastic form of these equations, which are practically used for computer implementation purposes, as follows:

$$\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{v}_{k-1}) \tag{3.1a}$$

$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k, \mathbf{n}_k) \tag{3.1b}$$

where, $\mathbf{f}$ represents the state transition function and $\mathbf{h}$ the measurement equation. The variable $\mathbf{u}_k$ denotes the input vector which includes all the known inputs to the process. The random variable $\mathbf{v}_k$ denotes the process noise and captures disturbances and modeling errors. The random variable $\mathbf{n}_k$ denotes the measurement noise. These variables make this model a stochastic one. Information about the distribution of these random variables is required for filtering and this is usually specified based on some simplifying assumptions about the nature of the uncertainties. The most common choice is a zero mean Gaussian distribution with

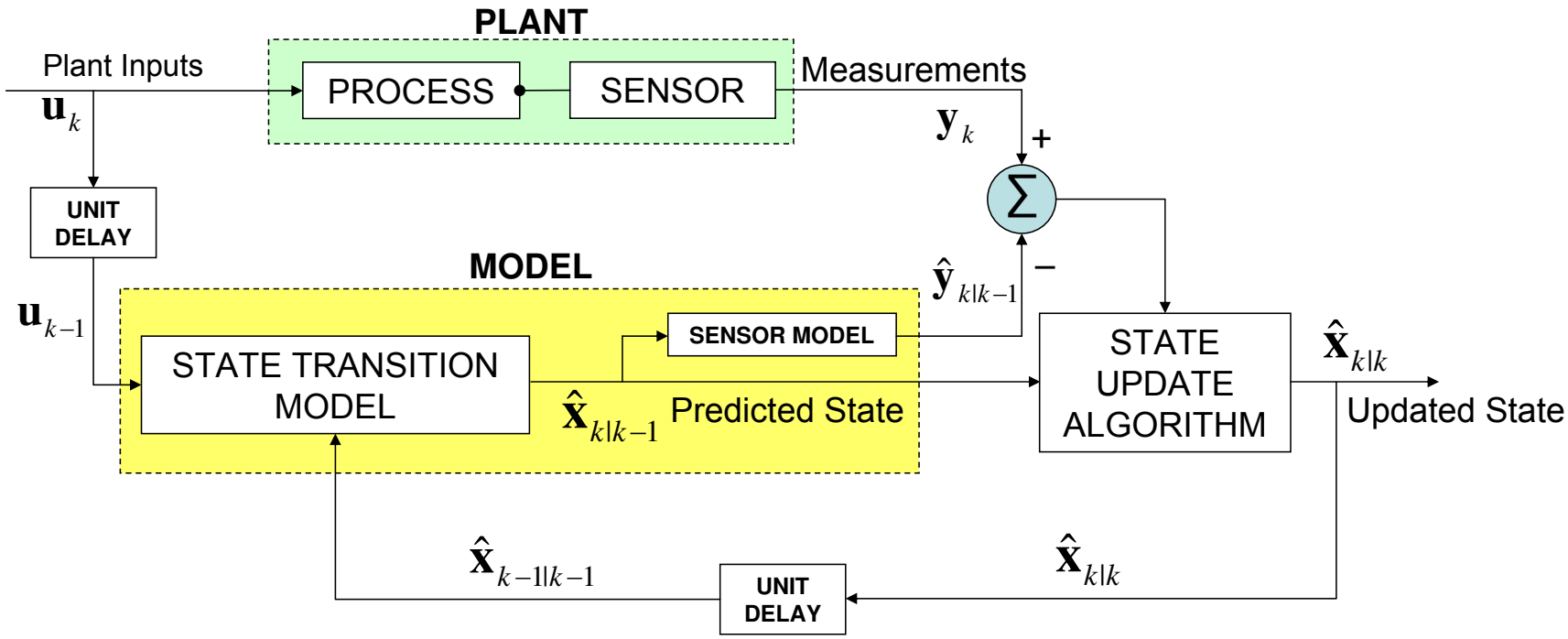


Figure 3.1: Block diagram to illustrate the general state estimation problem.

a known variance. Note that the state space form given above represents a Markov process, which simplifies to a large extent the mathematical formulation and solution of the estimation problem, as explained later in this chapter. By the definition of the Markov property, the state of the system at any instant is determined solely by the states, inputs and disturbances at the previous instant.

3.3 Bayesian Perspective

The Kalman filter was derived originally as a filter which minimizes the variance of estimates and achieves an 'optimal' fusion of noisy measurements in this sense. A more generic way of looking at the filtering problem is the Bayesian viewpoint using the theory of *conditional probability density*[2]. According to this viewpoint, the filter accomplishes the objective of finding "optimal" estimates by propagating the conditional probability density of the states, conditioned on all the data available from the sensors. Note that without the knowledge of the true values of $\mathbf{v}_k$ and $\mathbf{n}_k$, which is indeed the reality, we have to consider the states and measurements as random variables. Hence, we can define such a conditional probability density function associated with the states and measurements. Using this propagated conditional probability density, the "optimal" estimate can be defined. The most common choices are the *mean*, *mode* or *median* of this conditional density. It can be shown that the Kalman filter performs this propagation of conditional probability density and chooses the mean as the optimal estimate. For linear systems with Gaussian process and measurement noise, this conditional probability density can be shown to be always a Gaussian one, which can be propagated easily by using just the mean and covariance. Thus the Kalman filter is well justified as the optimal filter for such systems. A clear derivation of this result for a simple linear system with Gaussian noise will be shown shortly. Before that, we will mathematically derive the filtering problem from a Bayesian perspective.

The conditional probability of the system state in Eq. (3.1) at any instant k, conditioned on all the measurements available until time instant k is given by

$$p(\mathbf{x}_k|\mathbf{y}_{1:k}) = \frac{p(\mathbf{y}_{1:k}, \mathbf{x}_k)}{p(\mathbf{y}_{1:k})} \tag{3.2}$$

where $\mathbf{y}_{1:k}$ represents the set of all measurements from the sensors, available until time k. Noting that the measurement equation is "memory-less" ($\mathbf{y}_k$ depends only on $\mathbf{x}_k$ based on Eq. (3.1)), we can use the definition of conditional probability[2] to decompose the numerator in Eq. (3.2) as

$$\begin{aligned} p(\mathbf{y}_{1:k}, \mathbf{x}_k) &= p(\mathbf{y}_k|\mathbf{y}_{1:k-1}, \mathbf{x}_k)p(\mathbf{y}_{1:k-1}, \mathbf{x}_k) \\ &= p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{y}_{1:k-1}, \mathbf{x}_k) \end{aligned}$$

[2]For two events A and B, the conditional probability of A *given* B is defined as $P(A|B) = \frac{P(A,B)}{P(B)}$

and thus Eq. (3.2) becomes

$$p(\mathbf{x}_k|\mathbf{y}_{1:k}) = \frac{p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{y}_{1:k-1}, \mathbf{x}_k)}{p(\mathbf{y}_{1:k})}$$

Applying the definition of conditional probability to $p(\mathbf{y}_{1:k-1}, \mathbf{x}_k)$ and $p(\mathbf{y}_{1:k})$, we obtain

$$p(\mathbf{x}_k|\mathbf{y}_{1:k}) = \frac{p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{y}_{1:k-1})p(\mathbf{y}_{1:k-1})}{p(\mathbf{y}_k|\mathbf{y}_{1:k-1})p(\mathbf{y}_{1:k-1})}$$

Canceling out the common term $p(\mathbf{y}_{1:k-1})$ in the numerator and denominator, we obtain the following result to describe any filter from the Bayesian perspective:

$$p(\mathbf{x}_k|\mathbf{y}_{1:k}) = \frac{p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{y}_{1:k-1})}{p(\mathbf{y}_k|\mathbf{y}_{1:k-1})} \tag{3.3}$$

The relationship obtained in Eq. (3.3) can be interpreted by viewing each of the different elements as follows: $p(\mathbf{x}_k|\mathbf{y}_{1:k})$ in the L.H.S is the *posterior density of the state* at time k given all measurements until k (conditional probability density propagated in a filter); $p(\mathbf{y}_k|\mathbf{x}_k)$ is the *likelihood of the measurement* obtained at time k; $p(\mathbf{x}_k|\mathbf{y}_{1:k-1})$ is the *prior density of the state* at time k given all previous measurements (prior information only), and $p(\mathbf{y}_k|\mathbf{y}_{1:k-1})$ in the denominator is a normalizing constant which is independent of the state. The likelihood term is a conditional probability density of the measurement at the current instant, conditioned on a value of the state. This conditional density characterizes the stochastic relationship between $\mathbf{y}_k$ and $\mathbf{x}_k$, as defined by the measurement equation in Eq. (3.1). The likelihood can be evaluated using the probability density function (pdf) of the measurement noise distribution as

$$p(\mathbf{y}_k|\mathbf{x}_k) = \int \delta(\mathbf{y}_k - \mathbf{h}(\mathbf{x}_k, \mathbf{n}_k))p(\mathbf{n}_k)d\mathbf{n}_k \tag{3.4}$$

where, $\delta(.)$ is the Dirac-delta function. The prior density term $p(\mathbf{x}_k|\mathbf{y}_{1:k-1})$ defines the probability density of the current state conditioned on the set of all the past measurements until $(k-1)$. Unlike the likelihood, this prior density is not directly available to us using the equations in the state space model. If we assume that the filtering algorithm has completely processed all the information until the previous instant, then we can assume that $p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1})$ is available in some form. Then by the principle of induction, we would need a relationship between $p(\mathbf{x}_k|\mathbf{y}_{1:k-1})$ and $p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1})$ in order to establish a "recursive" relationship between the posterior density of the states at two consecutive instants. This relationship is known as the Chapman-Kolmogorov equation and is expressed as:

$$p(\mathbf{x}_k|\mathbf{y}_{1:k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1})dx_{k-1} \tag{3.5}$$

which uses the Markov property of the state space model. Here, $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ captures the stochastic relationship between $\mathbf{x}_k$ and $\mathbf{x}_{k-1}$ as defined by the state transition

equation in Eq. (3.1) and can be evaluated using the distribution of the process noise as (see Gordon et al. (1993))

$$p(\mathbf{x}_k|\mathbf{x}_{k-1}) = \int \delta(\mathbf{x}_k - \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{v}_{k-1}))p(\mathbf{v}_{k-1})d\mathbf{v}_{k-1} \tag{3.6}$$

Substituting Eq. (3.5) into (3.3), we obtain a very useful recursive form, generic enough to consider any filtering algorithm in terms of *propagation of conditional probability density*, given by

$$p(\mathbf{x}_k|\mathbf{y}_{1:k}) = \frac{p(\mathbf{y}_k|\mathbf{x}_k)\int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1})dx_{k-1}}{p(\mathbf{y}_k|\mathbf{y}_{1:k-1})} \tag{3.7}$$

This task of "propagation" can be translated into the two step procedure of *prediction* and *correction* as follows:

1. *Prediction step*:

$$p(\mathbf{x}_k|\mathbf{y}_{1:k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1})dx_{k-1} \tag{3.8}$$

2. *Correction/Update step*:

$$p(\mathbf{x}_k|\mathbf{y}_{1:k}) = \frac{p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{y}_{1:k-1})}{p(\mathbf{y}_k|\mathbf{y}_{1:k-1})} \tag{3.9}$$

The Chapman-Kolmogorov equation in Eq. (3.5), which gives the prior density of the states, constitutes the prediction step. The posterior density evaluation using Eq. (3.7) (note that the denominator is just a normalizing constant), is the correction or update step.

In order to better understand how the above equations can be useful, we can derive the Kalman filter from a Bayesian perspective. For this we consider a very simple linear state space system, with a scalar state variable as given in Eq. (3.10).

$$x_{k+1} = x_k + v_k, \quad v_k \sim N(0, \sigma_v^2) \tag{3.10a}$$

$$y_k = x_k + n_k, \quad n_k \sim N(0, \sigma_n^2) \tag{3.10b}$$

$$x_0 \sim N(\mu_0, \sigma_0^2) \tag{3.10c}$$

The dynamics of the system are modeled using a random walk model and the measurement of the state is also corrupted by noise. The process noise and measurement noise are assumed to be zero mean Gaussian random variables with variance σ_v^2 and σ_n^2 respectively. Also, the initial state x_0 is assumed to follow a Gaussian distribution with mean μ_0 and variance σ_0^2. Based on the state transition equation, we can say that prior to any measurements at $k = 1$, x_1 is Gaussian and follows $N(\mu_{1|0} = \mu_0, \sigma_{1|0}^2 = \sigma_0^2 + \sigma_v^2)$, i.e.,

$$p(x_1|y_0) = \frac{1}{\sqrt{2\pi}\sigma_{1|0}} \exp\left(\frac{-(x_1 - \mu_{1|0})^2}{2\sigma_{1|0}^2}\right) \tag{3.11}$$

Here, $\mu_{1|0}$ and $\sigma^2_{1|0}$ represent the conditional mean and variance of the state distribution at $k = 1$, conditioned on previous measurements. Note that since we are dealing with a Gaussian distribution for all variables, the mean and covariance suffice to completely define the pdf of the distribution. Let y_1 represent the value of measurement obtained at $k = 1$. Based on the measurement equation, the likelihood is given by

$$p(y_1|x_1) = \frac{1}{\sqrt{2\pi}\sigma_n} \exp\left(\frac{-(y_1 - x_1)^2}{2\sigma_n^2}\right) \tag{3.12}$$

Then, substituting the prior and likelihood into (3.3) (ignoring the denominator as constant and using proportionality instead of equality), we obtain the posterior of the state at $k = 1$ as

$$p(x_1|y_1) \propto \exp - \left(\frac{(x_1 - \mu_{1|0})^2}{2\sigma^2_{1|0}} + \frac{(y_1 - x_1)^2}{2\sigma_n^2}\right) \tag{3.13}$$

By completing the square and carrying out a few algebraic manipulations (detailed derivation given in Appendix A), we obtain

$$\begin{aligned} p(x_1|y_1) &= \frac{1}{\sqrt{2\pi}\sigma_{1|1}} \exp - \left(\frac{(x_1 - \mu_{1|1})^2}{2\sigma^2_{1|1}}\right) \\ \text{where, } \mu_{1|1} &= \mu_{1|0} + K_1(y_1 - \mu_{1|0}) \\ \sigma^2_{1|1} &= \sigma^2_{1|0} - K_1\sigma^2_{1|0} \\ K_1 &= \frac{\sigma^2_{1|0}}{\sigma^2_{1|0} + \sigma_n^2} \end{aligned} \tag{3.14}$$

Note that the posterior density of the state at $k = 1$, as given in Eq. (3.14), represents a Gaussian distribution with conditional mean $\mu_{1|1}$ and variance $\sigma^2_{1|1}$ which are conditioned on the measurement y_1. We can choose $\mu_{1|1}$, which is the mean of this conditional density, as the optimal estimate of the state. Note that the mean, median and the mode coincide for a Gaussian distribution and $\mu_{1|1}$ will also be the *maximum-a-posteriori* (MAP) estimate. The constant K_1 is the well known *Kalman update gain*, which applies a weighted correction to $\mu_{1|0}$ based on the difference between the obtained measurement and its predicted value. The expressions for $\mu_{1|1}$ and $\sigma^2_{1|1}$ represent the update step of the Kalman filter. In order to continue this procedure for subsequent time instants, we utilize the principle of mathematical induction. Assume that at time instant $(k - 1)$, we obtained the posterior pdf $p(x_{k-1}|y_{1:k-1})$ to be $N(\mu_{k-1|k-1}, \sigma^2_{k-1|k-1})$. Then, applying the Chapman-Kolmogorov equation we obtain the prior density as

$$
\begin{aligned}
p(x_k|y_{1:k-1}) &= \int p(x_k|x_{k-1})p(x_{k-1}|y_{1:k-1})dx_{k-1} \\
&= \int \frac{1}{\sqrt{2\pi}\sigma_v}\frac{1}{\sqrt{2\pi}\sigma_{k-1|k-1}} \times \\
&\quad \exp\left\{-\frac{1}{2}\left[\frac{(x_k - x_{k-1})^2}{\sigma_v^2} + \frac{(x_{k-1} - \mu_{k-1|k-1})^2}{\sigma_{k-1|k-1}^2}\right]\right\} dx_{k-1} \\
&= \frac{1}{\sqrt{2\pi(\sigma_v^2 + \sigma_{k-1|k-1}^2)}} \exp\left\{-\frac{1}{2}\frac{(x_k - \mu_{k-1|k-1})^2}{\sigma_v^2 + \sigma_{k-1|k-1}^2}\right\} \\
&= \frac{1}{\sqrt{2\pi\sigma_{k|k-1}^2}} \exp\left\{-\frac{1}{2}\frac{(x_k - \mu_{k|k-1})^2}{\sigma_{k|k-1}^2}\right\} \qquad (3.15) \\
\text{where, } \mu_{k|k-1} &= \mu_{k-1|k-1},\ \sigma_{k|k-1}^2 = \sigma_v^2 + \sigma_{k-1|k-1}^2
\end{aligned}
$$

Substituting Eq. (3.15) into (3.3), we obtain the posterior density of the state at any instant k as (using proportionality)

$$
p(x_k|y_{1:k}) \propto \exp\left(\frac{-(y_k - x_k)^2}{2\sigma_n^2}\right) \exp\left\{-\frac{1}{2}\frac{(x_k - \mu_{k|k-1})^2}{\sigma_{k|k-1}^2}\right\} \qquad (3.16)
$$

Again by completing the square, we obtain the posterior as

$$
\begin{aligned}
p(x_k|y_{1:k}) &\propto \exp\left\{-\frac{1}{2}\frac{(x_k - \mu_{k|k})^2}{\sigma_{k|k}^2}\right\} \\
\text{where, } \mu_{k|k} &= \mu_{k|k-1} + K_k(y_k - \mu_{k|k-1}) \\
\sigma_{k|k}^2 &= \sigma_{k|k-1}^2 - K_k\sigma_{k|k-1}^2 \\
K_k &= \frac{\sigma_{k|k-1}^2}{\sigma_{k|k-1}^2 + \sigma_n^2}
\end{aligned}
$$

3.4 Linear Systems: The Kalman Filter Algorithm

From the above derivations on a simple linear system, we see that the Bayesian perspective of the filtering problem can be used to analytically derive the posterior density of the state when the process and measurement noise are Gaussian. In fact, we observe that it is only required to propagate the mean and covariance for such systems since the posterior remains Gaussian at all times; these are the steps of the Kalman filter. For a general multivariable linear state space model, we can apply the above mentioned steps to arrive at the posterior mean and variance of the states. These equations form the basic computations involved in the Kalman filter.

Consider the following discrete time linear state space model

$$\mathbf{x}_k = \mathbf{A}\mathbf{x}_{k-1} + \mathbf{B}\mathbf{u}_{k-1} + \mathbf{v}_{k-1} \tag{3.17}$$

$$\mathbf{y}_k = \mathbf{H}\mathbf{x}_k + \mathbf{n}_k \tag{3.18}$$

$$\mathbf{v}_k \sim N(\mathbf{0}, \mathbf{Q}) \tag{3.19}$$

$$\mathbf{n}_k \sim N(\mathbf{0}, \mathbf{R}) \tag{3.20}$$

where, $\mathbf{x}_k \in \mathbb{R}^l$ (l = state space dimension), $\mathbf{y}_k \in \mathbb{R}^m$ (m = number of measurements), $\mathbf{u}_k \in \mathbb{R}^p$($p$ = number of inputs). Here, $\mathbf{A} \in \mathbb{R}^{l\times l}$ is the state transition matrix, $\mathbf{H} \in \mathbb{R}^{p\times l}$ is the measurement matrix and $\mathbf{B} \in \mathbb{R}^{l\times p}$ is the input matrix, all of which are assumed to be known. Similar to the simple linear system, we assume that the process and measurement noise are zero-mean Gaussian multivariate random variables and pure white noise sequences (uncorrelated in time). Also, we assume that the noise variables are uncorrelated with any of the other variables, i.e., $cov(\mathbf{v}_k, \mathbf{n}_j), cov(\mathbf{v}_k, \mathbf{x}_k), cov(\mathbf{v}_k, \mathbf{y}_k)$, $cov(\mathbf{n}_k, \mathbf{x}_k)$ and $cov(\mathbf{n}_k, \mathbf{y}_k)$ are all *null matrices*. The covariance matrices for process noise and measurement noise distribution are assumed to be $\mathbf{Q}$ and $\mathbf{R}$ respectively. Let the initial state distribution be Gaussian with mean $\boldsymbol{\mu}_{0|0}$ and covariance $\mathbf{P}_{0|0}$, i.e., $\mathbf{x}_0 \sim N(\boldsymbol{\mu}_{0|0}, \mathbf{P}_{0|0})$. Then, the posterior distribution for all subsequent time instants will follow a multivariate Gaussian distribution if all of the above mentioned assumptions hold true. In other words, all the distributions that we are interested in the prediction and correction steps, in equations (3.8) and (3.9) respectively, can be shown to be Gaussian, as follows:

1. *Prediction step*:

$$\mathbf{x}_{k|k-1} \sim p(\mathbf{x}_k|\mathbf{y}_{1:k-1}) \tag{3.21}$$

$$p(\mathbf{x}_k|\mathbf{y}_{1:k-1}) = N(\boldsymbol{\mu}_{k|k-1}, \mathbf{P}_{k|k-1}) \tag{3.22}$$

   where,

$$\begin{aligned} \boldsymbol{\mu}_{k|k-1} &= \mathbf{A}\boldsymbol{\mu}_{k-1|k-1} + \mathbf{B}\mathbf{u}_{k-1} \\ \mathbf{P}_{k|k-1} &= cov(\mathbf{x}_{k|k-1}) \\ &= \mathbf{Q} + \mathbf{A}\mathbf{P}_{k-1|k-1}\mathbf{A}^T \end{aligned}$$

2. *Correction/Update step*:

$$\mathbf{x}_{k|k} \sim p(\mathbf{x}_k|\mathbf{y}_{1:k}) \tag{3.23}$$

$$p(\mathbf{x}_k|\mathbf{y}_{1:k}) = N(\boldsymbol{\mu}_{k|k}, \mathbf{P}_{k|k}) \tag{3.24}$$

where,

$$\begin{aligned}
\boldsymbol{\mu}_{k|k} &= \boldsymbol{\mu}_{k|k-1} + \mathbf{K}_k(\mathbf{y}_k - \mathbf{H}\boldsymbol{\mu}_{k|k-1}) && (3.25)\\
\mathbf{P}_{k|k} &= cov(\mathbf{x}_{k|k}) = \mathbf{P}_{k|k-1} - \mathbf{K}_k\mathbf{H}\mathbf{P}_{k|k-1} && (3.26)\\
\mathbf{K}_k &= \mathbf{P}_{k|k-1}\mathbf{H}^T(\mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^T + \mathbf{R})^{-1} && (3.27)
\end{aligned}$$

In the above equations, $\boldsymbol{\mu}_{.|.} \in \mathbb{R}^l$ and $\mathbf{P}_{.|.} \in \mathbb{R}^{l\times l}$ represent the *conditional* mean and covariance of the appropriate multivariate Gaussian distributions. The Kalman update gain $\mathbf{K}_k$ will be a matrix of dimension $l \times m$. We can interpret the Kalman gain as a function of two covariance matrices. In order to do this, we define two random variables which are commonly known as *innovation* and *estimation error* respectively. The *innovation* is defined as

$$\boldsymbol{\varepsilon} \triangleq \mathbf{y}_k - \mathbf{H}\boldsymbol{\mu}_{k|k-1}$$

and the *estimation error* is defined as

$$\mathbf{e} \triangleq \mathbf{x}_k - \boldsymbol{\mu}_{k|k-1}$$

The innovation variable signifies the *new information* (hence the term innovation) contained in the measurement $\mathbf{y}_k$ compared to the predicted measurement $\mathbf{H}\boldsymbol{\mu}_{k|k-1}$. The estimation error, as the name suggests, signifies the error between the true process state $\mathbf{x}_k$ and the model prediction $\boldsymbol{\mu}_{k|k-1}$. Grouping the terms inside the inversion operation in Eq. (3.27) and denoting as $\mathbf{P}_k^{\varepsilon,\varepsilon}$, we can derive the following conditional covariance relationship:

$$\begin{aligned}
cov(\boldsymbol{\varepsilon}|\mathbf{y}_{1:k-1}) &= cov(\mathbf{y}_k - \mathbf{H}\boldsymbol{\mu}_{k|k-1}|\mathbf{y}_{1:k-1})\\
&= cov(\mathbf{H}\mathbf{x}_k + \mathbf{n}_k - \mathbf{H}\boldsymbol{\mu}_{k|k-1}|\mathbf{y}_{1:k-1})\\
&= cov\{\mathbf{H}(\mathbf{x}_k - \boldsymbol{\mu}_{k|k-1})|\mathbf{y}_{1:k-1}\} + cov(\mathbf{n}_k|\mathbf{y}_{1:k-1})\\
&= \mathbf{H}cov\{(\mathbf{x}_k - \boldsymbol{\mu}_{k|k-1})|\mathbf{y}_{1:k-1}\}\mathbf{H}^T + cov(\mathbf{n}_k|\mathbf{y}_{1:k-1})\\
&= \mathbf{H}cov(\mathbf{x}_k|\mathbf{y}_{1:k-1})\mathbf{H}^T + cov(\mathbf{n}_k)\\
&= \mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^T + \mathbf{R}\\
&\triangleq \mathbf{P}_k^{\boldsymbol{\varepsilon},\boldsymbol{\varepsilon}}
\end{aligned}$$

In the above derivation, we use the assumptions that the measurement noise covariance is equal to $\mathbf{R}$ irrespective of the measurements, i.e., $cov(\mathbf{n}_k|\mathbf{y}_{1:k-1}) = cov(\mathbf{n}_k) = \mathbf{R}$ and, that the measurement noise is uncorrelated with $\mathbf{x}_k$. In addition, the following can be derived for the conditional mean of the innovations and

estimation error random variables:

$$\begin{aligned} E\{\boldsymbol{\varepsilon}|\mathbf{y}_{1:k-1}\} &= E\{\mathbf{y}_k - \mathbf{H}\boldsymbol{\mu}_{k|k-1}|\mathbf{y}_{1:k-1}\} \\ &= E\{\mathbf{y}_k|\mathbf{y}_{1:k-1}\} - E\{\mathbf{H}\boldsymbol{\mu}_{k|k-1}|\mathbf{y}_{1:k-1}\} \\ &= E\{\mathbf{H}\mathbf{x}_k + \mathbf{n}_k|\mathbf{y}_{1:k-1}\} - \mathbf{H}E\{E\{\mathbf{x}_k|\mathbf{y}_{1:k-1}\}|\mathbf{y}_{1:k-1}\} \\ &= \mathbf{H}E\{\mathbf{x}_k|\mathbf{y}_{1:k-1}\} - \mathbf{H}E\{\mathbf{x}_k|\mathbf{y}_{1:k-1}\} \\ &= \mathbf{0} \end{aligned}$$

$$\begin{aligned} E\{\mathbf{e}|\mathbf{y}_{1:k-1}\} &= E\{\mathbf{x}_k - \boldsymbol{\mu}_{k|k-1}|\mathbf{y}_{1:k-1}\} \\ &= E\{\mathbf{x}_k|\mathbf{y}_{1:k-1}\} - E\{\boldsymbol{\mu}_{k|k-1}|\mathbf{y}_{1:k-1}\} \\ &= E\{\mathbf{x}_k|\mathbf{y}_{1:k-1}\} - E\{E\{\mathbf{x}_k|\mathbf{y}_{1:k-1}\}|\mathbf{y}_{1:k-1}\} \\ &= E\{\mathbf{x}_k|\mathbf{y}_{1:k-1}\} - E\{\mathbf{x}_k|\mathbf{y}_{1:k-1}\} \\ &= \mathbf{0} \end{aligned}$$

We can show that the term $\mathbf{P}_{k|k-1}\mathbf{H}^T$ in the Kalman gain calculation is the *cross-covariance* between the estimation error and innovations as follows:

$$\begin{aligned} cov(\mathbf{e}, \boldsymbol{\varepsilon}|\mathbf{y}_{1:k-1}) &= E\{[\mathbf{e}-E(\mathbf{e}|\mathbf{y}_{1:k-1})][\boldsymbol{\varepsilon}-E(\boldsymbol{\varepsilon}|\mathbf{y}_{1:k-1})]^T|\mathbf{y}_{1:k-1}\} \\ &= E\{\boldsymbol{\varepsilon}\mathbf{e}^T|\mathbf{y}_{1:k-1}\} \\ &= E\{[\mathbf{x}_k - \boldsymbol{\mu}_{k|k-1}][\mathbf{y}_k - \mathbf{H}\boldsymbol{\mu}_{k|k-1}]^T|\mathbf{y}_{1:k-1}\} \\ &= E\{[\mathbf{x}_k - \boldsymbol{\mu}_{k|k-1}][\mathbf{H}\mathbf{x}_k + \mathbf{n}_k - \mathbf{H}\boldsymbol{\mu}_{k|k-1}]^T|\mathbf{y}_{1:k-1}\} \\ &= E\{[\mathbf{x}_k - \boldsymbol{\mu}_{k|k-1}][\mathbf{H}(\mathbf{x}_k - \boldsymbol{\mu}_{k|k-1}) + \mathbf{n}_k]^T|\mathbf{y}_{1:k-1}\} \\ &= E\{[\mathbf{x}_k - \boldsymbol{\mu}_{k|k-1}][\mathbf{H}(\mathbf{x}_k - \boldsymbol{\mu}_{k|k-1})]^T + [\mathbf{x}_k - \boldsymbol{\mu}_{k|k-1}]\mathbf{n}_k^T|\mathbf{y}_{1:k-1}\} \\ &= E\{[(\mathbf{x}_k - \boldsymbol{\mu}_{k|k-1})][\mathbf{x}_k - \boldsymbol{\mu}_{k|k-1}]^T|\mathbf{y}_{1:k-1}\}\mathbf{H}^T + \\ &\quad E\{[\mathbf{x}_k - \boldsymbol{\mu}_{k|k-1}]\mathbf{n}_k^T|\mathbf{y}_{1:k-1}\} \\ &= \mathbf{P}_{k|k-1}\mathbf{H}^T + 0 \\ &= \mathbf{P}_{k|k-1}\mathbf{H}^T \\ &\triangleq \mathbf{P}_k^{\mathbf{e},\boldsymbol{\varepsilon}} \end{aligned}$$

Hence, we can interpret the Kalman gain as the product of two covariance matrices, as follows:

$$\begin{aligned} \mathbf{K}_k &= \mathbf{P}_k^{\mathbf{e},\boldsymbol{\varepsilon}}(\mathbf{P}_k^{\boldsymbol{\varepsilon},\boldsymbol{\varepsilon}})^{-1} \\ &= (\text{Cross-covariance between estimation error and innovations}) \\ &\quad \times(\text{Covariance of the innovations})^{-1} \end{aligned}$$

The covariance of the innovations is a measure of the measurement uncertainty and is proportional to the measurement noise variance $\mathbf{R}$. The cross-covariance between

$\mathbf{e}$ and $\boldsymbol{\varepsilon}$ is a measure of the impact of *innovation* on the *estimation error*, i.e., how the error in the predicted measurement will affect the error in state estimation. Note that in the update step, the Kalman filter corrects the predicted state mean $\boldsymbol{\mu}_{k|k-1}$ by a value proportional to the error in the predicted measurement $(\mathbf{y}_k - \mathbf{H}\boldsymbol{\mu}_{k|k-1})$. The Kalman update step applies a weighted correction, where the weighting is given by the ratio of $\mathbf{P}_k^{\mathbf{e},\boldsymbol{\varepsilon}}$ to $\mathbf{P}_k^{\boldsymbol{\varepsilon},\boldsymbol{\varepsilon}}$. A low magnitude for $\mathbf{P}_k^{\boldsymbol{\varepsilon},\boldsymbol{\varepsilon}}$ compared to $\mathbf{P}_k^{\mathbf{e},\boldsymbol{\varepsilon}}$, i.e., a high magnitude of $\mathbf{K}_k$ would signify that the measurement uncertainty is low and there is a high correlation between the estimation error and the error in the predicted measurement (innovation). In that case, the correction step will be very sensitive to the error in the predicted measurement $(\mathbf{y}_k - \mathbf{H}\boldsymbol{\mu}_{k|k-1})$. A useful heuristic to get good performance of the Kalman filter is to tune the magnitudes of the process and measurement noise covariances, $\mathbf{Q}$ and $\mathbf{R}$ respectively. For the simple univariate case we considered earlier, we can roughly say that

$$K_k \propto \frac{\sigma_v^2}{\sigma_v^2 + \sigma_n^2}$$

If $\sigma_n^2 \ll \sigma_v^2$ (measurement noise is negligible compared to disturbances), then the Kalman gain will be close to unity and the updated state will be equal to the measured value, i.e., we completely trust the measurements and model predictions do not affect our estimate. On the other hand, if $\sigma_n^2 \gg \sigma_v^2$, then the Kalman gain would be close to zero and the updated state will be equal to the predicted state. This means that we completely trust the process model and ignore the measurements. This is intuitive also, since a high σ_n^2 signifies significantly noisy measurements and hence the measured data is considered unreliable for estimation.

3.5 Nonlinear Systems: The Ensemble Kalman Filter and the Particle Filter

In the presence of any form of nonlinearity in the state space model, the analytical calculations involved in the Bayesian filtering steps become very complicated. This is because the posterior will no longer be a Gaussian distribution. The same is true if any of the noise distributions become non-Gaussian. In such scenarios, the integral involved in the Chapman-Kolmogorov equation becomes intractable and hence it is not possible to get an analytical expression for the posterior density. To get around this problem, some approximate suboptimal solutions exist such as EKF, UKF, EnKF and particle filters. The EKF is an analytical approximation method where a linearized version of the nonlinear state space model is used at each instant and subsequently a Kalman filter based prediction and update is implemented (Ristic et al., 2004). The linearization is done using first order Taylor series expansion by evaluating the Jacobian of the nonlinear model around the value of the filtered

states and measurements at the previous instant. Boutayeb et al. (1995) have shown through a theoretical convergence analysis of the EKF that if the covariance matrices Q_k and R_k satisfy certain conditions, then the state estimates will converge to their true values. But the main drawback of the EKF is that for systems with significant nonlinearity, it will give inconsistent estimates because of approximation errors arising from linearization (Ristic et al., 2004). Also, the method cannot be applied to nonlinear systems which do not have an analytical expression for the Jacobian. The UKF, proposed by Julier & Uhlmann (1997), is an improvement over the EKF and removes the need for evaluating the Jacobian for nonlinear state space systems. By using a deterministic sampling technique known as the *unscented transform*, the mean and covariance of the states at the prediction step are evaluated followed by the Kalman filter type of updating. The unscented transform can accurately give the mean and covariance of any random variable undergoing a nonlinear transformation. Similar to the EKF, the main assumption in UKF is that the posterior is always Gaussian. This is the basic limitation of the UKF because the true posterior of a nonlinear state space system will be in general non-Gaussian and hence the mean and covariance are not sufficient to characterize it completely. This limitation is alleviated, at least to some extent, if one uses the method of Monte Carlo simulation to represent the prior and posterior density in the filtering problem. This is the basic essence of the estimation algorithms based on sequential Monte Carlo methods.

3.5.1 What are Monte Carlo simulations ?

The method of Monte Carlo simulation is key to the development of the ensemble Kalman filter and particle filter algorithms. The terminology "Monte Carlo" was coined in the 1940's, at a time marked for the renaissance of the method of statistical sampling (Metropolis, 1987). The terminology is a general name for methods/algorithms that can generate random samples from a probability distribution of interest. We assume that such a computational algorithm exists which can generate random samples, at least when the distribution is completely defined. For example, in the MATLAB computational environment the function `randn` will generate random numbers from a standard normal distribution (mean = 0, variance = 1). Extending this to normal distributions with any given mean and variance is trivial. For performing a Monte Carlo simulation of a dynamic system described by a stochastic state space model, first we will use the Monte Carlo technique to generate samples of all the variables whose distribution is assumed to be known a priori. Considering the generic stochastic state space model given in Eq. (3.1), there are three probability distributions that we assume to be known a priori. These probability distributions are mathematical representations of the uncertainty in our knowledge about the true process. They are the following:

1. The distribution that represents the uncertainty in the initial conditions of the process (i.e., uncertainty in the initial value of the process states), $p(\mathbf{x}_0)$ at $k = 0$. For pedagogical reasons and ease of description of the EnKF and particle filter later in this chapter, let $k-1$ represent the initial time instant at which the Monte Carlo simulation would be started.

2. The distribution of the process noise/disturbance, $p(\mathbf{v}_k)$ for all k.

3. The distribution of the measurement noise, $p(\mathbf{n}_k)$ for all k.

Given the above three distributions, the first step is to draw a finite number of samples from the initial distribution $p(\mathbf{x}_{k-1})$. Let N_{mc} represent the number of samples and $\widehat{\mathbf{X}}_{k-1} \equiv (x_{k-1}^1, x_{k-1}^2, ..., x_{k-1}^{N_{mc}})$ represent the N_{mc} samples. In addition, let $\widehat{\mathbf{V}}_{k-1} \equiv (v_{k-1}^1, v_{k-1}^2, ..., v_{k-1}^{N_{mc}})$ represent N_{mc} samples drawn from the process noise distribution $p(\mathbf{v}_{k-1})$. Then, using each of the samples in $\widehat{\mathbf{X}}_{k-1}$ and $\widehat{\mathbf{V}}_{k-1}$, a one-step forward simulation can be performed using the state transition equation in Eq. (3.1). Note that the variable $\mathbf{u}_{k-1}$ is a deterministic input to the process and it is assumed that its exact value is known a priori. The Monte Carlo set of new values of the states obtained through this one-step forward simulation, which we collectively represent as $\widehat{\mathbf{X}}_{k|k-1}$, can be considered as *Monte Carlo samples from the prior distribution* $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ *at time instant* k. This is equivalent to the 'prediction step' of the Kalman filter, except that here we would have a distribution of N_{mc} predicted states. Summary statistics of the new samples such as the mean and variance can be useful for summarizing the predictive information. Also, measurement predictions can be simulated by transforming the samples in $\widehat{\mathbf{X}}_{k|k-1}$ using the measurement equation in Eq. (3.1b). These steps can be repeated for subsequent time steps and will yield N_{mc} "Monte Carlo paths" that the true process can be expected to take, had there been an exact match between the true process and the state space model. Figure 3.2 graphically illustrates the result of Monte Carlo simulation for a two-dimensional state space system. The black points represent the statistical samples generated through the Monte Carlo simulation. The red line denotes the trajectory of the true process. The bounding ellipses are representative of the spread of the statistical samples.

3.5.2 The ensemble Kalman filter algorithm

The Ensemble Kalman filter (EnKF), proposed by Evensen (1994), is a Monte Carlo simulation based variant of the Kalman filter designed for handling nonlinear systems. Note that one of the basic requirements for the Kalman filter computations is that the state transition model and the sensor model should be linear functions. Also, the process noise and measurement noise should follow a Gaussian distribution. We can relax these requirements in the EnKF computations. The main idea is to

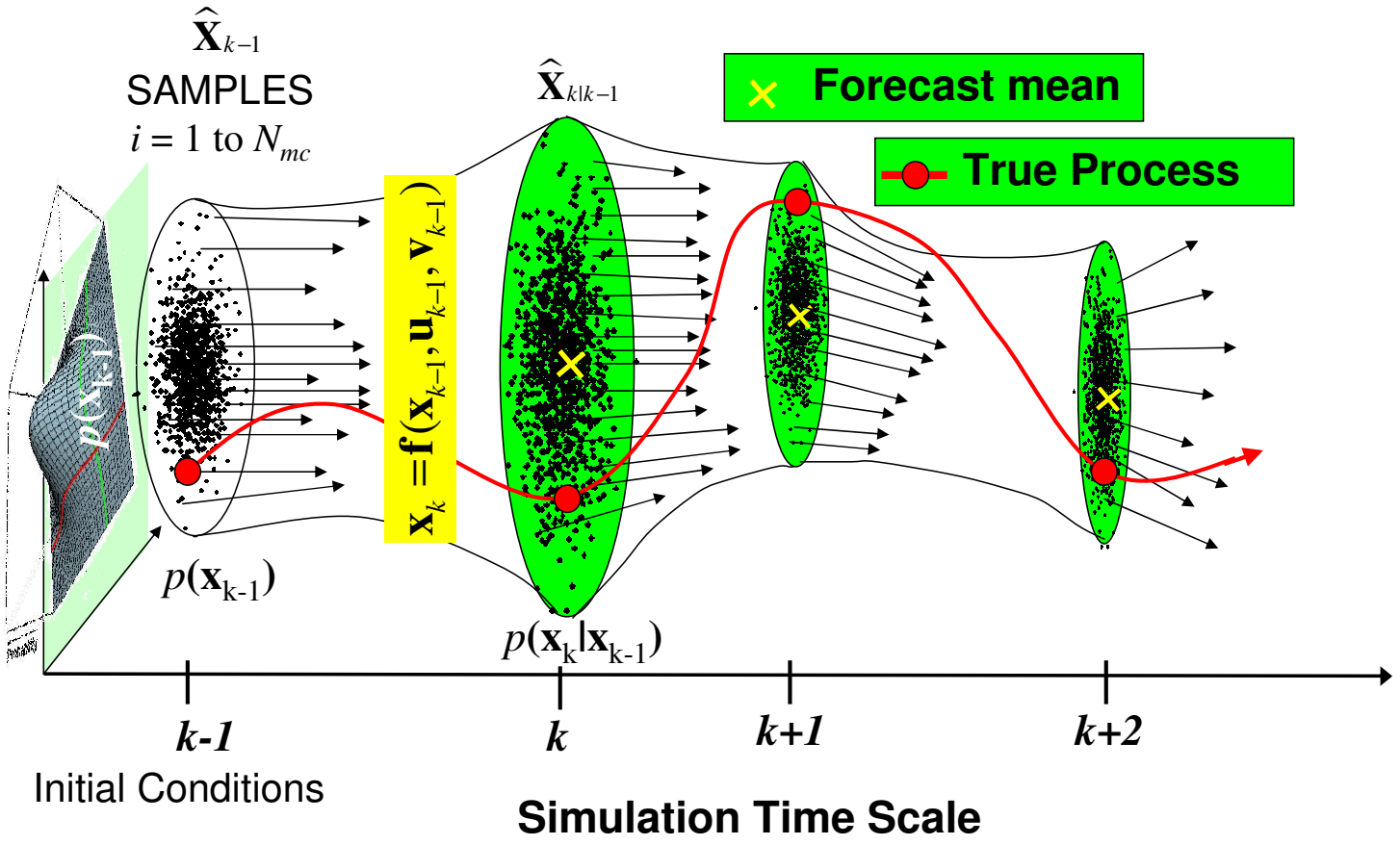


Figure 3.2: Illustration of Monte Carlo simulation for a two-dimensional state space system.

use a bank of Kalman filters via an ensemble of Monte Carlo simulations of the process. The covariance matrices $\mathbf{P}_k^{e,\varepsilon}$ and $\mathbf{P}_k^{\varepsilon,\varepsilon}$are calculated as sample covariances of the statistical samples in the ensemble. The Kalman gain is calculated using these sample covariance matrices and the Kalman update is applied to each sample in the predicted state ensemble. The corrected ensemble so obtained will be propagated forward using the state transition model. It is not required to store the error covariance matrix and no explicit covariance evolution equations are needed. The main assumption in the EnKF is that the state estimation error and the innovation random variables follow a multivariate normal distribution. These in turn imply that the pdf of the prior distribution of the state and the likelihood pdf are both assumed to be Gaussian. Since these assumptions do not hold true for nonlinear systems in general, the EnKF should be regarded as a sub-optimal nonlinear filter. Typical implementations of the EnKF in the literature suggest that an ensemble size of $O(50)$ is sufficient to get good estimates of the states.

Numerous applications of this method for nonlinear state estimation, especially for large scale systems, have been shown in the oceanography and meteorology literature (Evensen, 2003). A comprehensive overview of the EnKF is given in Evensen (2007). In recent years, this method has been demonstrated to be effective for parameter estimation of highly nonlinear large scale models of oil and gas reservoirs (Seiler et al., 2009). In this section, we briefly present the basic steps of the EnKF algorithm for state estimation.

Assume that a nonlinear state space model of the process is available in the form of Eq. (3.1). The EnKF uses an ensemble of Monte Carlo samples to represent the state prediction error and the output error (innovation) statistics. The sample covariance of the state prediction error and innovation samples will be used to calculate the Kalman gain. At time $k-1$, assume that we have an ensemble of state estimates given by:

$$\widehat{\mathbf{X}}_{k-1|k-1} \equiv (x^1_{k-1|k-1}, x^2_{k-1|k-1}, ..., x^{N_e}_{k-1|k-1})$$

where, in general, $x^i_{k|k} \in \mathbb{R}^l$, $\widehat{\mathbf{X}}_{k|k} \in \mathbb{R}^{N_e \times l}$, N_e is the ensemble size and $(.)_{k|k}$ denotes the updated value of the corresponding vector given the measurements up to the k^{th} time instant. Given $\widehat{\mathbf{X}}_{k-1|k-1}$, the EnKF steps for the subsequent time instant k are as follows:

1. Prediction step:

The *Prediction step* is a one-step Monte Carlo forward simulation of each ensemble member, carried out using the state transition model. We can represent the predicted state ensemble as $\widehat{\mathbf{X}}_{k|k-1} \equiv (x^1_{k|k-1}, x^2_{k|k-1}, ..., x^{N_e}_{k|k-1})$, where $x^i_{k|k-1}$ is obtained by the equation

$$x^i_{k|k-1} = f(x^i_{k-1|k-1}, u_{k-1}) + \mathbf{v}^i_{k-1} \tag{3.28}$$

Note that $\mathbf{v}_k^i$ is a random sample drawn from the distribution $p(\mathbf{v}_k)$. The *state estimation error* is approximated around the sample mean $\boldsymbol{\mu}_{k|k-1}^{\mathbf{x}} = \frac{1}{N_e}\sum_{i=1}^{N_e} x_{k|k-1}^i$ as

$$e_{k|k-1}^i = x_{k|k-1}^i - \boldsymbol{\mu}_{k|k-1}^{\mathbf{x}}$$

In order to calculate the *innovation covariance*, measurement samples are generated using the predicted state ensemble and the measurement equation. An ensemble of simulated measurements, $\widehat{\mathbf{Y}}_{k|k-1} \equiv (y_{k|k-1}^1, y_{k|k-1}^2, ..., y_{k|k-1}^{N_e})$ are generated using the measurement equation as follows:

$$y_{k|k-1}^i = h(x_{k|k-1}^i) \tag{3.29}$$

The innovation covariance matrix is defined as the sample covariance of $y_{k|k-1}^i$ around the sample mean, $\boldsymbol{\mu}_{k|k-1}^{\mathbf{y}} = \frac{1}{N_e}\sum_{i=1}^{N_e} y_{k|k-1}^i$, as follows:

$$\widehat{\mathbf{P}}_{k|k-1}^{\boldsymbol{\varepsilon},\boldsymbol{\varepsilon}} = \frac{1}{N_e - 1}\sum_{i=1}^{N_e}(\boldsymbol{\varepsilon}_{k|k-1}^i)(\boldsymbol{\varepsilon}_{k|k-1}^i)^T \tag{3.30}$$

where, $\boldsymbol{\varepsilon}_{k|k-1}^i = y_{k|k-1}^i - \boldsymbol{\mu}_{k|k-1}^{\mathbf{y}}$. Similarly, a cross-covariance matrix between the prediction error and output error is defined as follows:

$$\widehat{\mathbf{P}}_{k|k-1}^{\mathbf{e},\boldsymbol{\varepsilon}} = \frac{1}{N_e - 1}\sum_{i=1}^{N_e}(e_{k|k-1}^i)(\boldsymbol{\varepsilon}_{k|k-1}^i)^T \tag{3.31}$$

2. Correction/Update step:

The *Correction/Update step* is similar to the Kalman filter, where each of the samples in the predicted state ensemble is updated using the measurement y_k from the process. While using the measured data, it is required to treat them as random variables contaminated by noise (Burgers et al., 1998; Evensen, 2009). Hence, a set of N_e perturbed measurements around the measured data sample y_k are generated using random samples from the measurement noise distribution as follows:

$$y_k^i = y_k + \mathbf{n}_k^i \tag{3.32}$$

where, $\mathbf{n}_k^i$ is a random sample drawn from the distribution $p(\mathbf{n}_k)$. The purpose of carrying out the perturbation operation as shown above is to ensure that the sample covariance of y_k^i will be representative of the measurement error covariance. Burgers et al. (1998) and Evensen (2009) have shown analytically that this is a necessary step for the convergence of EnKF to the Kalman filter, in the limit of an infinite ensemble size for linear systems. The *Kalman gain* is computed using the error covariance matrices as

$$K_k = \widehat{\mathbf{P}}_{k|k-1}^{\mathbf{e},\boldsymbol{\varepsilon}}(\widehat{\mathbf{P}}_{k|k-1}^{\boldsymbol{\varepsilon},\boldsymbol{\varepsilon}} + \mathbf{R})^{-1} \tag{3.33}$$

where, $\mathbf{R}$ is the variance/covariance of the measurement noise distribution. The updated state ensemble is given by

$$\widehat{\mathbf{X}}_{k|k} \equiv (x^1_{k|k}, x^2_{k|k}, ..., x^{N_e}_{k|k}) \tag{3.34}$$

$$where,\ x^i_{k|k} = x^i_{k|k-1} + K_k(y^i_k - y^i_{k|k-1}) \tag{3.35}$$

The updated samples will be used to re-initialize the model for the subsequent time instant and the algorithm will proceed by following the above steps. The optimal estimate of the state vector can be defined as the mean of the updated state ensemble:

$$\widehat{x}^*_{k|k} = \frac{1}{N_e}\sum_{i=1}^{N_e} x^i_{k|k} \tag{3.36}$$

The sample covariance of the updated state ensemble, $\widehat{\mathbf{P}}^*_{k|k}$, can be computed as

$$\widehat{\mathbf{P}}^*_{k|k} = \frac{1}{N_e - 1}\sum_{i=1}^{N_e}(x^i_{k|k} - \widehat{x}^*_{k|k})(x^i_{k|k} - \widehat{x}^*_{k|k})^T \tag{3.37}$$

which will provide the quality of the state and parameter estimates generated by the EnKF.

Figure 3.3(a) graphically illustrates the prediction and correction steps of the EnKF for a two-dimensional state space system. The prediction step is similar to the Monte Carlo simulation illustrated earlier in Figure 3.2. Figure 3.3(b) illustrates the computational steps for a single time instant k, in more detail. As shown by the ellipses denoting *Corrected uncertainty*, the spread of the ensemble will reduce after the correction step. The reduced spread/variance of the ensemble is a result of the Kalman update step and signifies the reduction in uncertainty, *posterior to accounting for the information from the sensor measurements*. Also, there will be a shift in the mean of the ensemble after the Kalman update. This is shown in Figure 3.3(b) by the red arrows, which indicate the shifting of the ensemble members towards a new region in the state space after the correction step. The mean of corrected state ensemble can be conceptually expected to be closer to the true process state.

3.5.3 The particle filter algorithm

The main difference between the particle filter and the EnKF is in the correction/update step - in the EnKF algorithm, a Kalman gain based linear update is used for each ensemble member, whereas in the particle filter a *re-sampling step* is used to update the particles (Evensen, 2009). 'Particles' in the particle filter are equivalent to the realizations/ensemble members in the EnKF. Before describing the particle filter algorithm, we first go through some advanced Monte Carlo concepts which will help in better understanding of the algorithm. One of them is to visualize

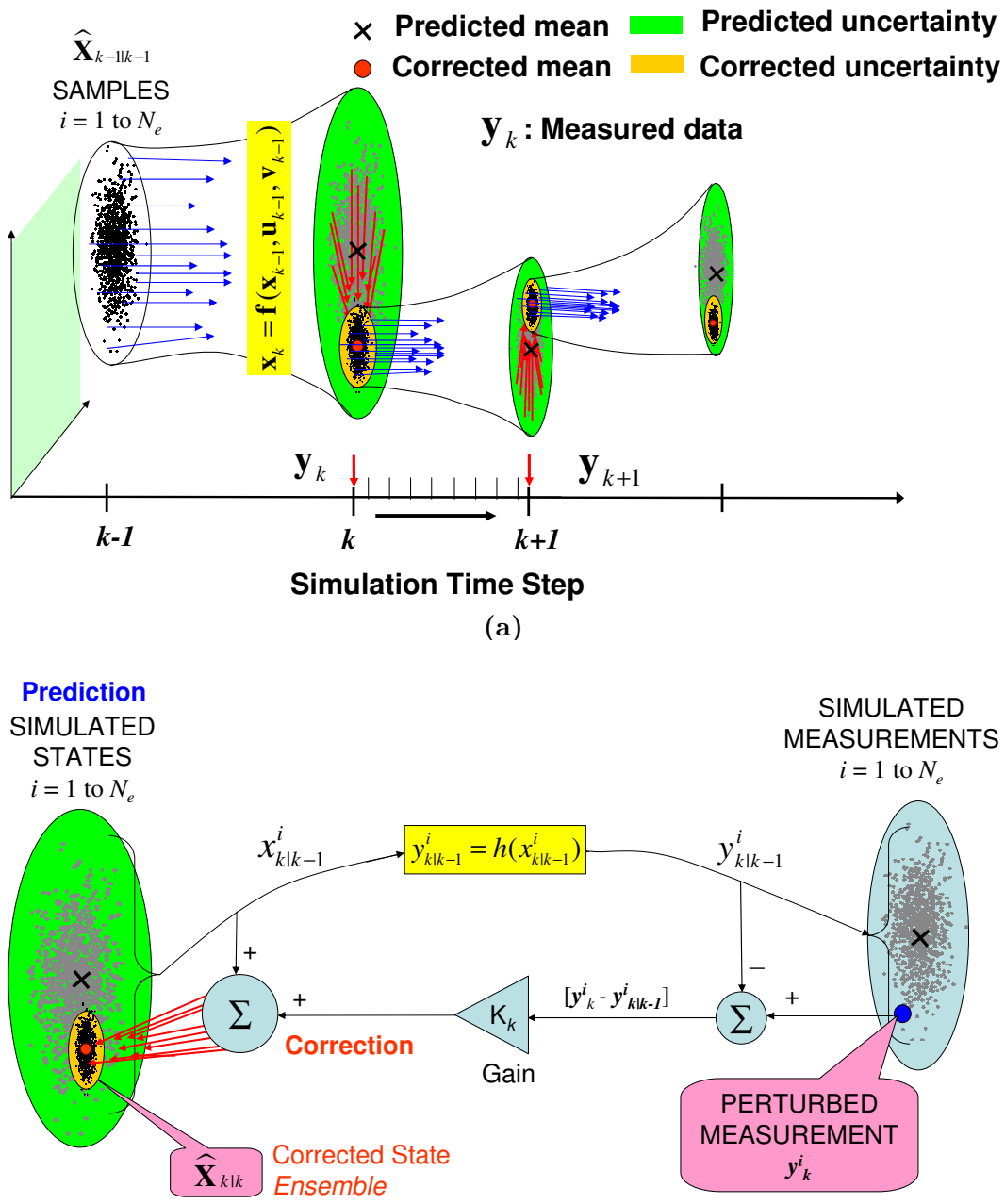


Figure 3.3: Illustration of the *prediction* and *correction* steps of the EnKF algorithm, showing: (a) the ensemble evolution with the progression of the update step for consecutive time instants k & $k+1$, and (b) the computational steps for one time instant k in more detail. The ensemble of states inside the green and orange ellipses shown in (b) correspond to the predicted and corrected ensemble at time instant k in (a), respectively.

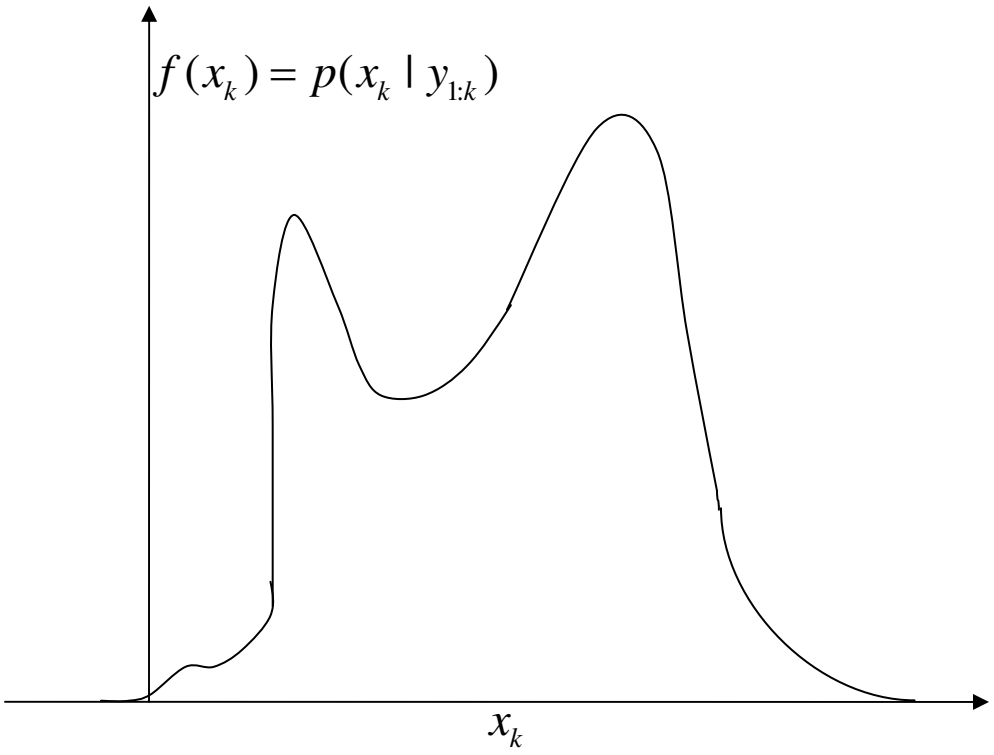


Figure 3.4: Illustration of the posterior density of the state for a scalar state space system.

any probability distribution as a plot of the density function against the values of the random variable. Since the conditional probability density is also a probability density, we can visualize the posterior probability density at any given instant as a function whose value can be evaluated at all points in the *state space* and plotted against the states. The word "conditional" means that the shape of this function depends upon the measurements available from the process until the current time instant and hence this function would evolve over time. For example, for a scalar state space system we can easily visualize this as a plot shown in Figure 3.4. Note that in general, this function can take any arbitrary shape and will vary from one system to another. For a linear system with Gaussian process and measurement noise, this plot will be the bell shaped curve of a Gaussian distribution as derived earlier for a univariate example.

The presence of distinct peaks in the posterior density at particular locations of the state space means that these regions have to be considered to have higher "weight" when performing a probabilistic inference. This introduces the concept of using weights associated with certain values of random variables in order to represent any distribution. In the context of state estimation, we are concerned

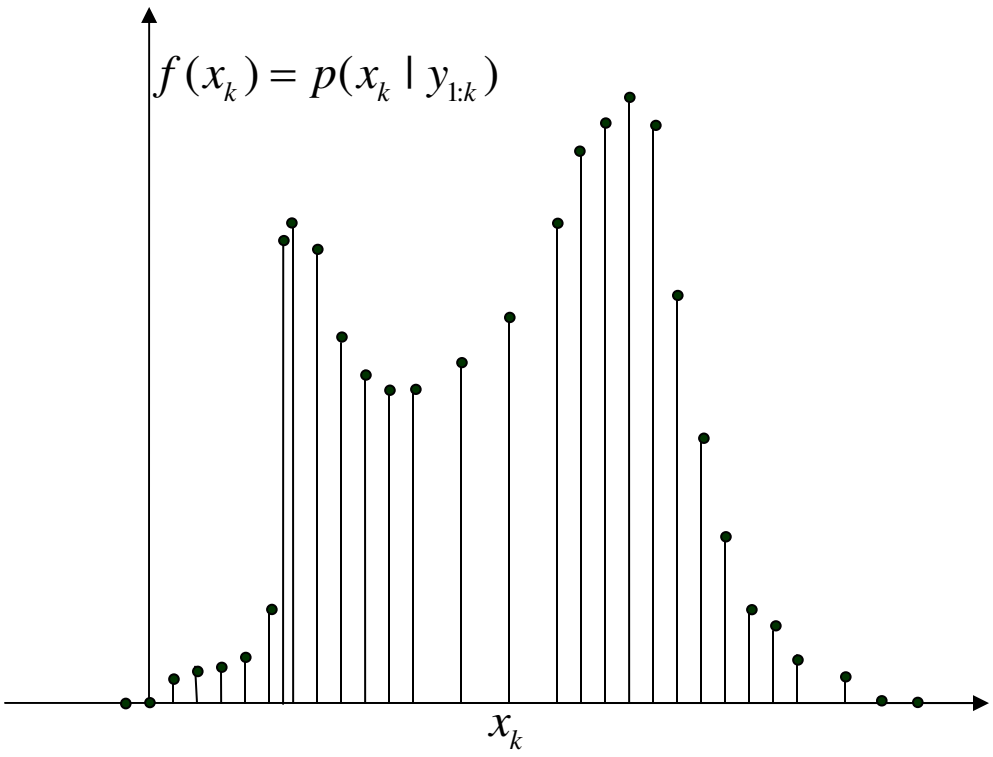


Figure 3.5: Approximation of the posterior density in Fig. 3.4 using discrete point weights.

with the random variable $\mathbf{x}_k$ and its posterior distribution defined by the density function $p(\mathbf{x}_k|\mathbf{y}_{1:k})$. Since the density function is defined in a continuous state space domain for most real world systems, a functional form is required to get an exact representation of the distribution. But this is not practically possible since the mathematics involved in its evaluation is tractable only for some specific systems as mentioned earlier. Taking the Monte Carlo route, we can solve this problem by using a discrete approximation of the distribution defined as

$$p(\mathbf{x}_k|\mathbf{y}_{1:k}) \approx \sum_{i=1}^{N_p} w^i \delta(\mathbf{x}_k - \mathbf{x}^i) \tag{3.38}$$

where, $\delta(\mathbf{x}_k - \mathbf{x}^i)$ represents the Dirac-delta function. In Eq. (3.38), we have used N_p points $\{\mathbf{x}^i\}_{i=1}^{N_p}$ in the entire state space, each with an associated weight w^i, together represented by the set $\{\mathbf{x}^i, w^i\}_{i=1}^{N_p}$, to approximately capture the shape of the density function $p(\mathbf{x}_k|\mathbf{y}_{1:k})$. For example, we can approximate the posterior density in Figure 3.4 using discrete points as shown in Figure 3.5. In order to satisfy the fundamental property of any discrete distribution, the weights are normalized so that the sum is equal to 1, i.e., $\sum_{i=1}^{N_p} w^i = 1$.

A further simplification of this approximation technique is to use *random samples* from the discrete distribution. This can be done by *resampling* (with replacement) from the discrete distribution $\{\mathbf{x}^i, w^i\}_{i=1}^{N_p}$ to obtain a set of N_p *random samples* $\{\mathbf{x}^{*i}\}_{i=1}^{N_p}$ each with weight $1/N_p$. We can discard the weights w^i after resampling, since the new set of samples form independent, identically distributed (i.i.d) samples which have uniform weights. For example, any Gaussian distribution can be represented by a set consisting of a large number of random samples drawn from the distribution. This can be verified by observing that as the number of samples is increased, the sample mean and variance will converge towards the corresponding values of the true population. The collection of random samples, which are called "particles", carry the information vital for the computations in the particle filter algorithm. The objective of the algorithm is to find these particles which can approximate the posterior at each instant. Then, an optimal estimate can be easily defined by using any of the aggregate measures of this discrete distribution such as *mean*, *median* or *mode*.

A particle filter is essentially a Monte Carlo simulation based method of finding where the process states are located in the state space at any instant. As mentioned earlier, in particle filtering a discrete representation of the posterior pdf of the states is also obtained at all time instants. Assume that at the time instant $(k-1)$, we have performed filtering and we have particles $\widehat{\mathbf{X}}_{k-1|k-1} \equiv \{\mathbf{x}^i_{k-1|k-1}\}_{i=1}^{N_p}$ which approximate the posterior density, $p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1})$. Using each of these particles as initial conditions and the known inputs $\mathbf{u}_{k-1}$, we can do a one step forward simulation of the process with the stochastic state space model, similar to the EnKF. This simulation will be done as follows: for every particle $\mathbf{x}^i_{k-1|k-1}$, a sample of the process noise $\mathbf{v}^i_{k-1}$ is drawn from the known distribution given by $p(\mathbf{v}_{k-1})$, and substituted in the state transition equation to get a new particle $\mathbf{x}^i_{k|k-1} = f(\mathbf{x}^i_{k-1|k-1}, \mathbf{u}_{k-1}, \mathbf{v}^i_{k-1})$. The new set of particles that is obtained, represented as $\{\mathbf{x}^i_{k|k-1}\}_{i=1}^{N_p}$, will represent i.i.d samples from the prior distribution defined by the density $p(\mathbf{x}_k|\mathbf{y}_{1:k-1})$. Note that no information about $\mathbf{y}_k$ was used for the simulation. The above mentioned one-step simulation constitutes the *prediction step* of the particle filter. In order to obtain an estimate of the state at the current time instant k, we require samples from the posterior distribution $p(\mathbf{x}_k|\mathbf{y}_{1:k})$ which incorporates the knowledge of $\mathbf{y}_k$. However, there is no direct way of drawing samples from the posterior since the density function $p(\mathbf{x}_k|\mathbf{y}_{1:k})$ is not available. By the recurrence relationship in Eq. (3.3), we have (ignoring the normalizing constant in the denominator and using proportionality)

$$p(\mathbf{x}_k|\mathbf{y}_{1:k}) \propto p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{y}_{1:k-1}) \tag{3.39}$$

Note that we already have samples $\{\mathbf{x}^i_{k|k-1}\}_{i=1}^{N_p}$ which represent $p(\mathbf{x}_k|\mathbf{y}_{1:k-1})$ in the R.H.S of the above equation. The likelihood $p(\mathbf{y}_k|\mathbf{x}_k)$ can be evaluated for any $\mathbf{x}_k$ by

substituting its value in Eq. (3.4) and knowing the pdf for the measurement noise. It turns out that using the samples from $p(\mathbf{x}_k|\mathbf{y}_{1:k-1})$ and the value of likelihood function evaluated at these sample points, one can obtain samples representing the posterior $p(\mathbf{x}_k|\mathbf{y}_{1:k})$. This is the *update step* in the particle filter and is carried out by a method called *sampling importance resampling*. This technique makes it possible to recursively propagate the particles based on the relationship in Eq. (3.39). This method will be explained through a simple example.

A simple example

Consider a process which has the following state space representation:

$$x_k = x_{k-1} + v_{k-1}, \quad v_k \sim N(0,5) \tag{3.40a}$$

$$y_k = x_k + n_k, \quad n_k \sim N(0,1) \tag{3.40b}$$

$$x_0 = 1 \ (Initial \ \ state) \tag{3.40c}$$

$$y_0 = No \quad measurement \quad available \ at \ k = 0 \tag{3.40d}$$

Simulation of the *true* process

Drawing a random sample from the process noise distribution using the MATLAB function `randn`, we obtain

$$v_1 = \texttt{sqrt(5)*randn} = \texttt{sqrt(5)} * 0.7826 = 1.7501$$

Using the above random sample of v_1 and performing a one-step forward simulation of the state space model, we obtain

$$x_1^{true} = x_0 + v_1 = 2.7501 \tag{3.41}$$

$$\begin{aligned} y_1^{meas} &= 2.7501 + \texttt{randn} \\ &= 2.7501 - 0.7204 \\ &= 2.0297 \end{aligned} \tag{3.42}$$

Estimation using 5 particles

Assuming the above realization of the state space model as the *true process*, the estimation of the state of the process at time instant k will be demonstrated using the particle filter algorithm. Here, N=5 particles will be used for the purpose of demonstrating the basic calculations 'by hand'. The basic steps of particle filtering will be shown along with the detailed computations involved in each step.

1. **At k=0:**

   Note that initially we do not have any measurements from the true process and hence the posterior for x_0 will be *assumed* to be of some appropriate form in

order to obtain i.i.d samples. These samples are the first set of particles in the particle filter algorithm which will be propagated forward through the state transition equation. Subsequently, the evolution of these particles through the prediction and update step will yield the samples required for all operations. Assuming $x_0 \sim N(1,1)$, consider the following 5 particles as a realization of a random draw from the distribution:

$$\widehat{x}_{0|0} = \begin{bmatrix} 0.3821 \\ 2.4085 \\ -0.4799 \\ 2.0195 \\ 0.3948 \end{bmatrix}$$

2. **At k=1:**

   (a) **Prediction step:** As explained earlier, in the prediction step the prior distribution particles will be obtained using the state model. For this we draw $N_p = 5$ random samples from the process noise distribution. A realization of this obtained using the MATLAB function `randn` was as follows:

$$\begin{aligned} \widehat{v}_{k-1} &= \texttt{sqrt(5)*randn(5,1)} \\ &= \begin{bmatrix} -2.3620 \\ 0.2305 \\ 0.7657 \\ 0.7584 \\ -4.7368 \end{bmatrix} \end{aligned}$$

   Using $\widehat{x}_{k-1}$ and $\widehat{v}_{k-1}$ for $k = 1$, we obtain the prior particles using the state transition equation as

$$\widehat{x}_{k|k-1} = \widehat{x}_{k-1} + \widehat{v}_{k-1} = \begin{bmatrix} -1.9799 \\ 2.6390 \\ 0.2858 \\ 2.7779 \\ -4.3420 \end{bmatrix}$$

   (b) **Likelihood evaluation:** Since we have Gaussian measurement noise, the likelihood at each of the prior particles is given by (based on Eq. 3.4)

$$p(y_k^{meas} |\ x_k = \widehat{x}_{k|k-1}^i) \propto \exp\left(\frac{-(y_k^{meas} - \widehat{y}_{k|k-1}^i)^2}{2\sigma_n^2}\right)$$

   where, y_k^{meas} is the value of the measurement obtained at time k (Eq. 3.42), $\widehat{y}_{k|k-1}^i$ is its predicted value based on the measurement model and

the i^{th} particle $\widehat{x}^i_{k|k-1}$. Thus, we have

$$\begin{aligned} \widehat{y}_{k|k-1} &= \widehat{x}_{k|k-1} \\ &= \begin{bmatrix} -1.9799 \\ 2.6390 \\ 0.2858 \\ 2.7779 \\ -4.3420 \end{bmatrix} \end{aligned}$$

Let,

$$w^i = \exp\left(\frac{-(y_k^{meas} - \widehat{y}^i_{k|k-1})^2}{2\sigma_n^2}\right)$$

Therefore,

$$\textbf{Likelihood} \propto w_i$$

Evaluating w^i for the 5 particles, we obtain the vector w as

$$w = \begin{bmatrix} \exp\left(\frac{-(\ 2.0297 - \widehat{y}^1_{k|k-1})^2}{2}\right) \\ \exp\left(\frac{-(\ 2.0297 - \widehat{y}^2_{k|k-1})^2}{2}\right) \\ \exp\left(\frac{-(\ 2.0297 - \widehat{y}^3_{k|k-1})^2}{2}\right) \\ \exp\left(\frac{-(\ 2.0297 - \widehat{y}^4_{k|k-1})^2}{2}\right) \\ \exp\left(\frac{-(\ 2.0297 - \widehat{y}^5_{k|k-1})^2}{2}\right) \end{bmatrix} = \begin{bmatrix} 0.0003 \\ 0.8306 \\ 0.2186 \\ 0.7559 \\ 0 \end{bmatrix}$$

The corresponding normalized weights, so that the sum of weights is equal to 1, are given by

$$w = \begin{bmatrix} 0.0002 \\ 0.4601 \\ 0.1211 \\ 0.4187 \\ 0 \end{bmatrix}$$

Figure 3.6 shows the stem plot of the normalized weights of the 5 particles.

(c) **Correction step:** *Sampling Importance Resampling (SIR)*

In the previous steps, we obtained samples from the prior distribution which corresponds to the prediction step in the particle filter. Also, we evaluated their respective likelihoods using the measurement equation and the measurement noise distribution (Step b). In order to obtain posterior samples (update step of the estimation problem), we require

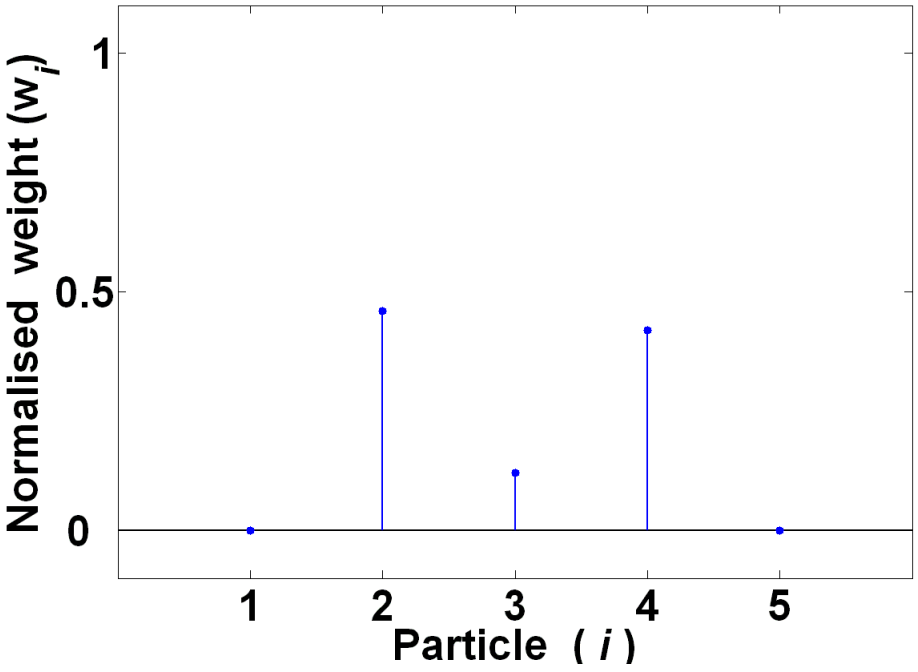


Figure 3.6: Normalized weights plotted against the particle index.

samples from the posterior distribution. Based on Eq. (3.39), the posterior pdf can be related to the likelihood function and the prior pdf as follows:

$$\underbrace{p(x_k|y_k^{meas})}_{\textbf{samples required}} \propto \underbrace{p(y_k^{meas}|x_k)}_{\textbf{normalized weights}} \times \underbrace{p(x_k|y_{1:k-1}^{meas})}_{\textbf{samples available}}$$

The Sampling Importance Resampling (SIR) algorithm can be employed to obtain samples from $p(x_k|y_k^{meas})$, given samples from $p(x_k|y_{1:k-1}^{meas})$. The SIR algorithm was first proposed in a Bayesian logistic regression application by Rubin (1983). The application of the algorithm for obtaining samples from posterior distributions was illustrated in Rubin (1988). The algorithm is useful when a good approximation of the posterior distribution exists from which it is easy to obtain samples. This distribution is called the *importance distribution*. The basic idea is to "obtain samples from this approximate distribution, and then resample from this finite sample with probability proportional to the 'importance ratios' to obtain the required posterior samples". In the current context it is assumed that the prior distribution, from which we have samples available, is an approximation of the posterior. This assumption results in a particular type of particle filter which is widely known as the *SIR-particle filter*. The choice of this importance distribution is one of the

main variations in the different types of particle filter algorithms in the literature. For the SIR-particle filter, the importance ratio $r(x)$ of any sample x is given by

$$r(x) \propto \frac{posterior}{prior}$$
$$\text{where, } \frac{posterior}{prior} \propto likelihood$$

Thus, we have the importance ratio as

$$r(\widehat{x}^i_{k|k-1}) \propto p(y^{meas}_k |\ \widehat{x}^i_{k|k-1})$$
$$\text{where, } p(y^{meas}_k |\ \widehat{x}^i_{k|k-1}) = w^i$$

i.e. the importance ratios are proportional to the likelihood values which were evaluated in the previous step (Step b).

By the SIR approach we resample from $\widehat{x}^i_{k|k-1}$ with probability proportional to w^i to obtain the posterior samples. Note that we have normalized w^i already and hence we can perform resampling considering w^i as the discrete probability associated with particle i. This resampling is just a standard problem of drawing samples from a finite discrete distribution, given the probability values of all the points in the discrete space. This can be done by the method of inversion of the cumulative distribution function (cdf). This method was first suggested by the mathematician John von Neumann in 1947 (Eckhardt, 1987; Metropolis, 1987). The cdf has the property that it is always uniformly distributed on the interval [0,1], independent of the pdf. The basic steps of sampling via inversion of the cdf can be described as follows:

i. Sample a uniform random number ξ from $U[0, 1]$
ii. Equate the cdf at a location x in the space of random variable X to ξ, i.e., $F(x) = \xi$
iii. Invert the cdf to obtain a sample x^k from the distribution of X:

$$x^k = F^{-1}(\xi)$$

iv. Go to Step i. until $k =$ required number of samples

The above steps is graphically depicted for a univariate continuous random variable in Figure 3.7. This method can be used for a discrete distribution also, where $F(x)$ will be discrete. For resampling from the prior samples, we have the discrete case and can be done using the above method which is popularly known as the 'Golden rule of Sampling'.

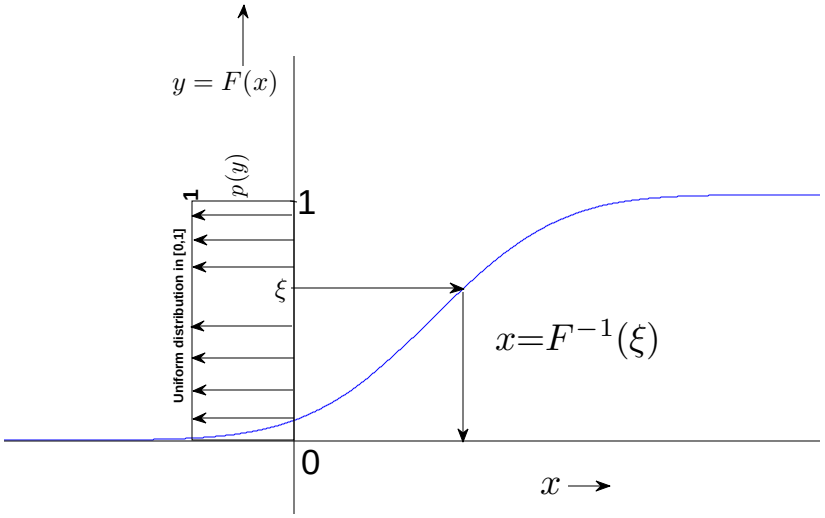


Figure 3.7: Sampling by inversion of the cumulative distribution function.

First, we have to construct the discrete cdf of particles using w as:

$$\begin{aligned} F_j &= \sum_{i=1}^{j} w_i, \\ for \;\; j &= 1 \;\; to \;\; 5 \end{aligned}$$

Here, we have $F = [0.0002 \quad 0.4602 \quad 0.5813 \quad 1.0000 \quad 1.0000]$ and this is plotted in Figure 3.8.

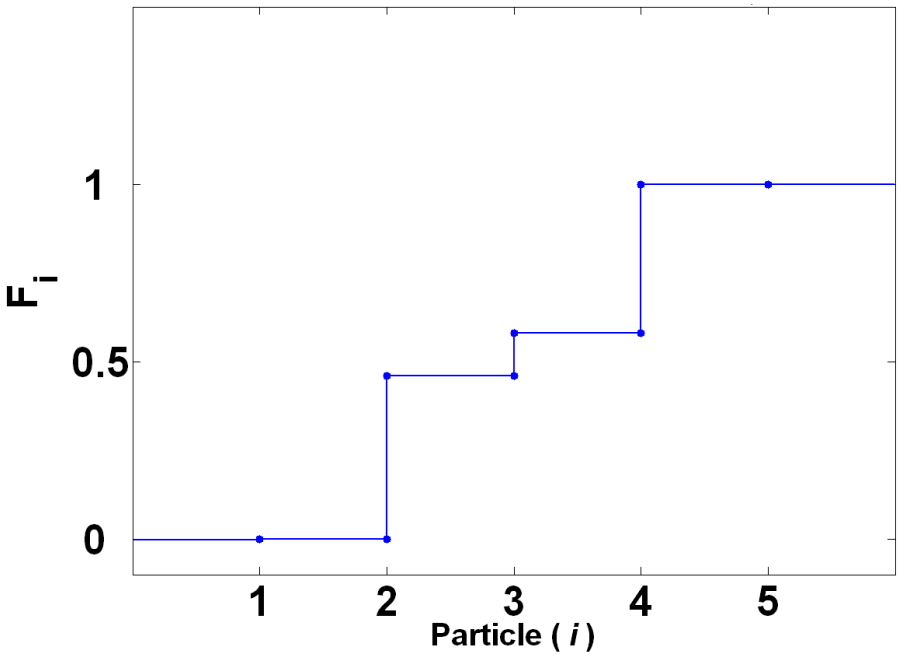


Figure 3.8: Cumulative distribution function based on w^i.

Resampling from the prior to obtain posterior samples ($\widehat{x}_{k|k}$):

i. Generate $N_p = 5$ uniform random numbers in the interval [0 1]:
$U = [0.4358 \quad 0.3631 \quad 0.8810 \quad 0.8012 \quad 0.0060]$

ii. *for* $i = 1$ *to* N_p

FIND $\quad j \ : F_{j-1} < U_i < F_j$

$\widehat{x}^i_{k|k} = jth\ particle \ \ in \ \ \widehat{x}_{k|k-1}$

end

With the above procedure, we obtain

$$\widehat{x}_{k|k} = \begin{bmatrix} \textbf{2nd particle in } \widehat{x}_{k|k-1} \\ \textbf{2nd particle in } \widehat{x}_{k|k-1} \\ \textbf{4th particle in } \widehat{x}_{k|k-1} \\ \textbf{4th particle in } \widehat{x}_{k|k-1} \\ \textbf{2nd particle in } \widehat{x}_{k|k-1} \end{bmatrix} = \begin{bmatrix} 2.6390 \\ 2.6390 \\ 2.7779 \\ 2.7779 \\ 2.6390 \end{bmatrix}$$

After resampling, we have equal weights for all particles, i.e.,

$$w_{\widehat{x}_{k|k}} = 1/N_p = 1/5$$

Comparing the histogram of resampled particles ($\widehat{x}_{k|k}$) in Figure 3.9 with the plot of normalized weights in Figure 3.6, it is clear that resampling preserves the distribution. [Note: The 2^{nd} and 4^{th} particles are the closest

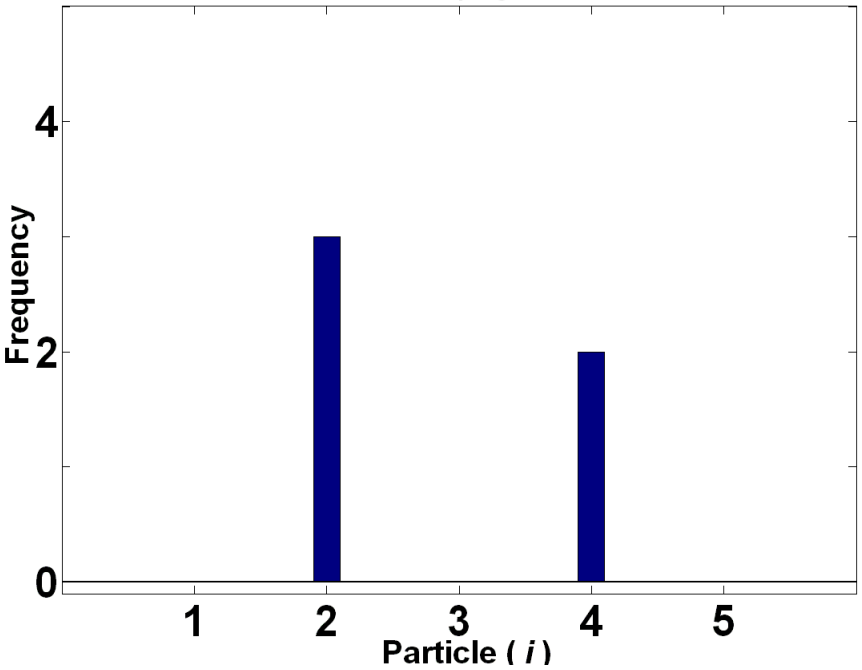


Figure 3.9: Histogram of the particles after resampling at $k = 1$.

to the true state ($x_1^{true} = 2.7501$) among all the particles in the set of

$\widehat{x}_{k|k-1}$. Hence, their higher weight (i.e. likelihood) is reflected in terms of higher frequency in the resampled set of $\widehat{x}_{k|k}$.]

(d) We can define the optimal estimate as either the sample *mean* or *mode* of the resampled particles, i.e.,

$$\begin{aligned} \textbf{Estimate} &= \textbf{Sample mean of } \widehat{x}_{k|k} \;\; OR \;\; \textbf{Mode of } \widehat{x}_{k|k} \\ &= \frac{1}{5}\sum \widehat{x}^i_{k|k} \; OR \;\; \textbf{2nd particle in } \;\; \widehat{x}_{k|k-1} \end{aligned}$$

Therefore,

$$x_k^{estimate} = 2.6946(mean) \;\; or \;\; 2.6390(mode)$$

Note that, here the mean of $\widehat{x}_{k|k}$ gives more accurate estimate. The choice of mean/mode as estimate is problem dependent.

3. **For the subsequent time instant k+1, we have:**

$$\underbrace{\widehat{x}_{k+1|k}}_{\substack{prior\ samples \\ at \quad (k+1)}} = \underbrace{\widehat{x}_{k|k}}_{\substack{posterior\ samples \\ at \quad k}} + \underbrace{\widehat{v}_k}_{\substack{new\ noise\ samples \\ at \quad k}}$$

Loop from steps 2(b) to 3

Running the algorithm for subsequent time instants, we can observe the quality of particle filtering by comparing the estimates with the true states of the system. Figure 3.10 shows an instance of running the particle filter algorithm on this example for k= **1** *to* **100**.

The choice of number of particles is problem dependent and the accuracy of the estimate can be expected to improve with increasing number of particles. This is because we always get a better approximation of the probability distribution with more representative samples. For a one dimensional problem such as the one discussed here, it is customary to use as many as 50 to 100 particles.

Estimation using N=50 particles:

Figure 3.11 shows the stem plots of the normalized weights (before resampling) and histogram of particles (after resampling) for another instance of the above example, using $N = 50$ particles. The sample mean statistic of the resampled particles was used for obtaining the estimate ($x_k^{estimate}$). As is evident from Figure 3.11, the use of more particles is able to give more meaningful weights to the particles (unimodal, normal distribution) for this linear state space system. Figure 3.12 shows

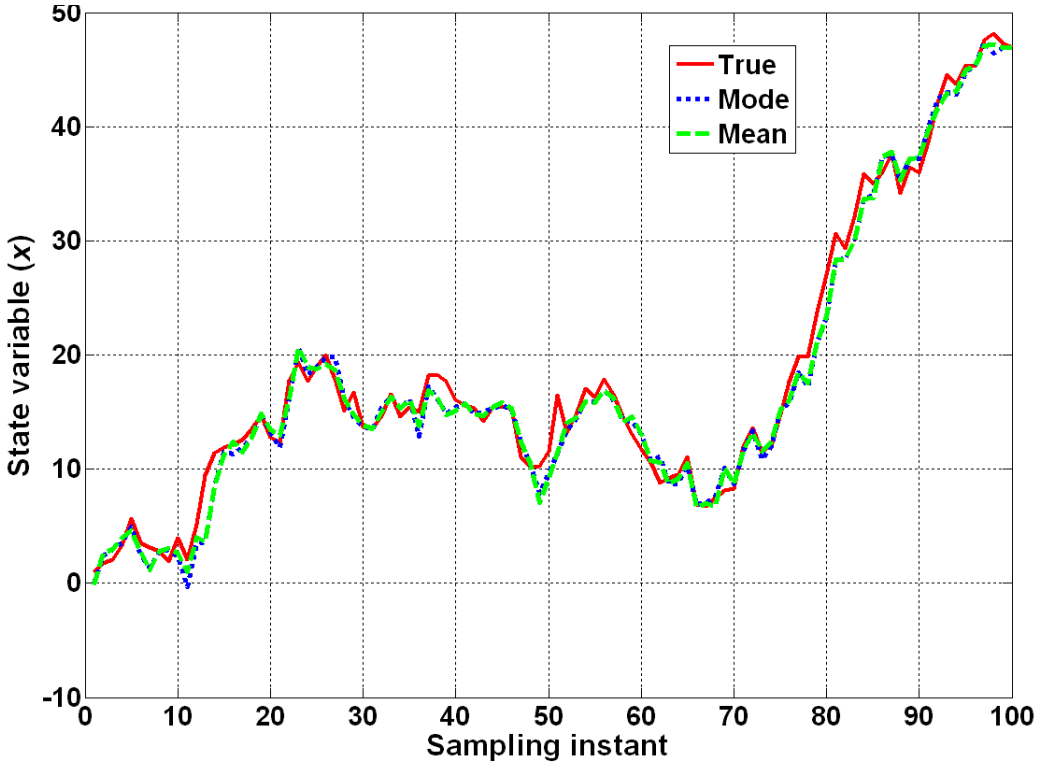


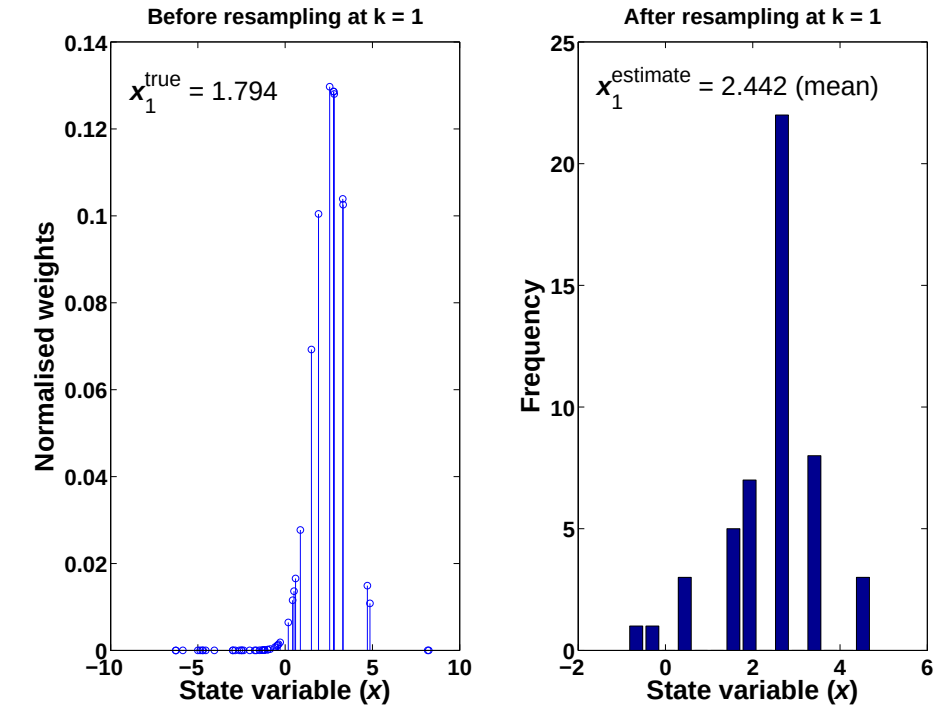
Figure 3.10: Particle filtering with N=5 particles: Comparison of true and estimated states.

an instance of the true states and the corresponding estimates using **N=50** particles for k= **1** *to* **100**.

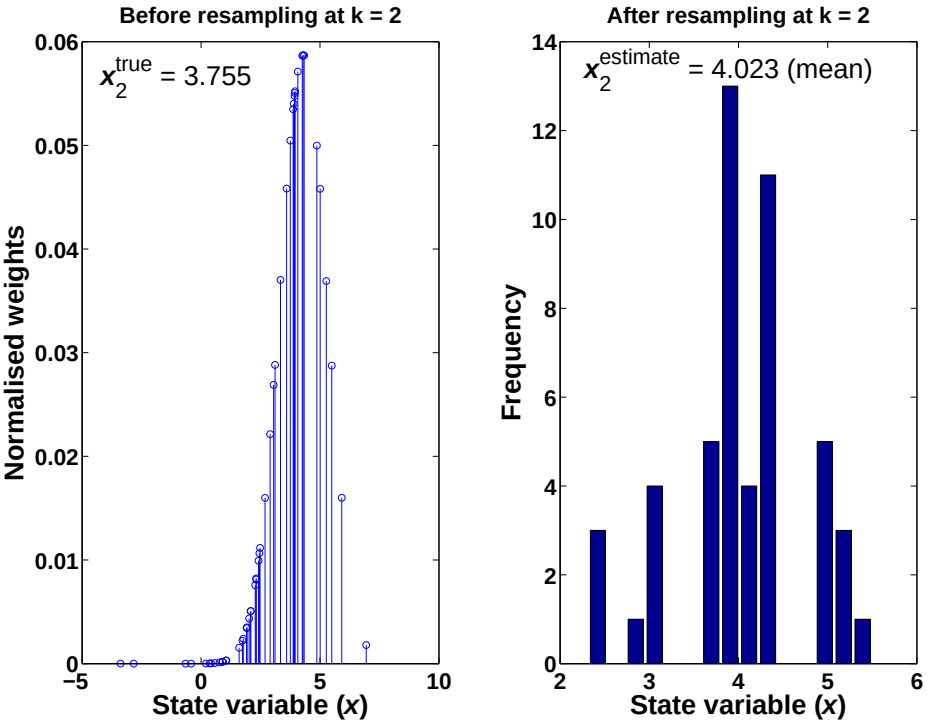
The basic computational steps involved in the SIR-particle filter algorithm for a single time instant k is graphically illustrated in Figure 3.13. The points in Figure 3.13 represent the particles in a two-dimensional state space. Figure 3.13(a) illustrates the prediction step, and 3.13(b)–(c) constitute the correction/update step in the particle filter algorithm. In Figure 3.13(b), the color contours illustrate how the state space spanned by the particles can be demarcated according to the value of their importance weights. After resampling, the particle population will be concentrated more towards the region of higher importance weight as shown in Figure 3.13(c).

The stochastic state space model of the process and the importance distribution completely defines the structure of the particle filter. In order to keep the discussion at a basic level, the different choices of importance distribution that can be made were not considered here. For a given structure of the filter, there are three basic requirements for designing the algorithm (see Gordon et al. (1993)):

1. The initial distribution of the states, $p(\mathbf{x}_0)$ at $k = 0$, should be available for sampling.



(a)



(b)

Figure 3.11: Comparison of the weights (*before resampling*) and histogram of particles (*after resampling*) at (a) k=1, (b) k=2.

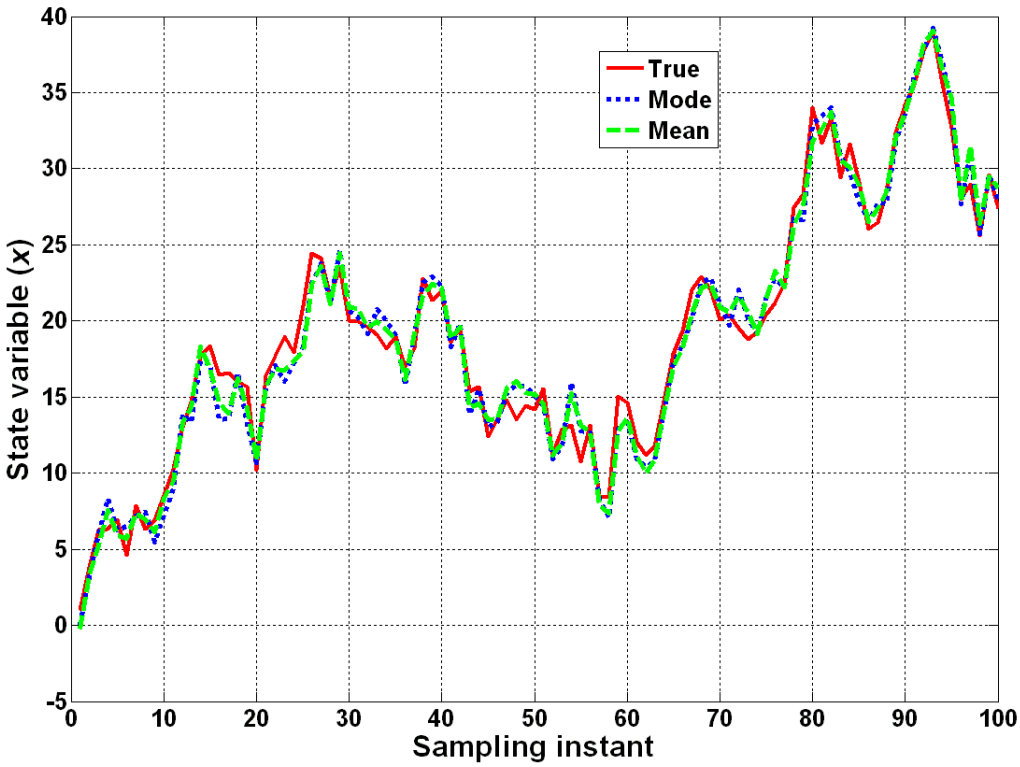


Figure 3.12: Particle filtering with N=50 particles: Comparison of true and estimated states.

2. The likelihood $p(y_k|x_k)$ should be available as a known functional form. This implies that the pdf of the measurement noise distribution should be explicitly known.

3. The process noise distribution $p(v_k)$ should be available for sampling.

The above requirements, which also partly decide the design of a particle filter, satisfy the mathematical formulation of the particle filter. The design parameters that completely defines a particle filter includes the choice of the distributions enlisted above and also the number of particles intended to be used. Like any other filtering algorithm, a lot of engineering insight and experience has to be used in "tuning" the particle filter so that it will become an effective operational filter. In the Kalman filter and the EnKF algorithm, this process of tuning is usually done by iteratively searching for the covariance of the process noise and measurement noise that yield the best estimation performance (Maybeck, 1979). A similar approach can also be taken for tuning the particle filter to give good estimation performance. Note that the particle filter imposes no restrictive assumptions of process linearity or Gaussian noise distributions and this procedure of tuning can be applied for any given stochastic model of a process.

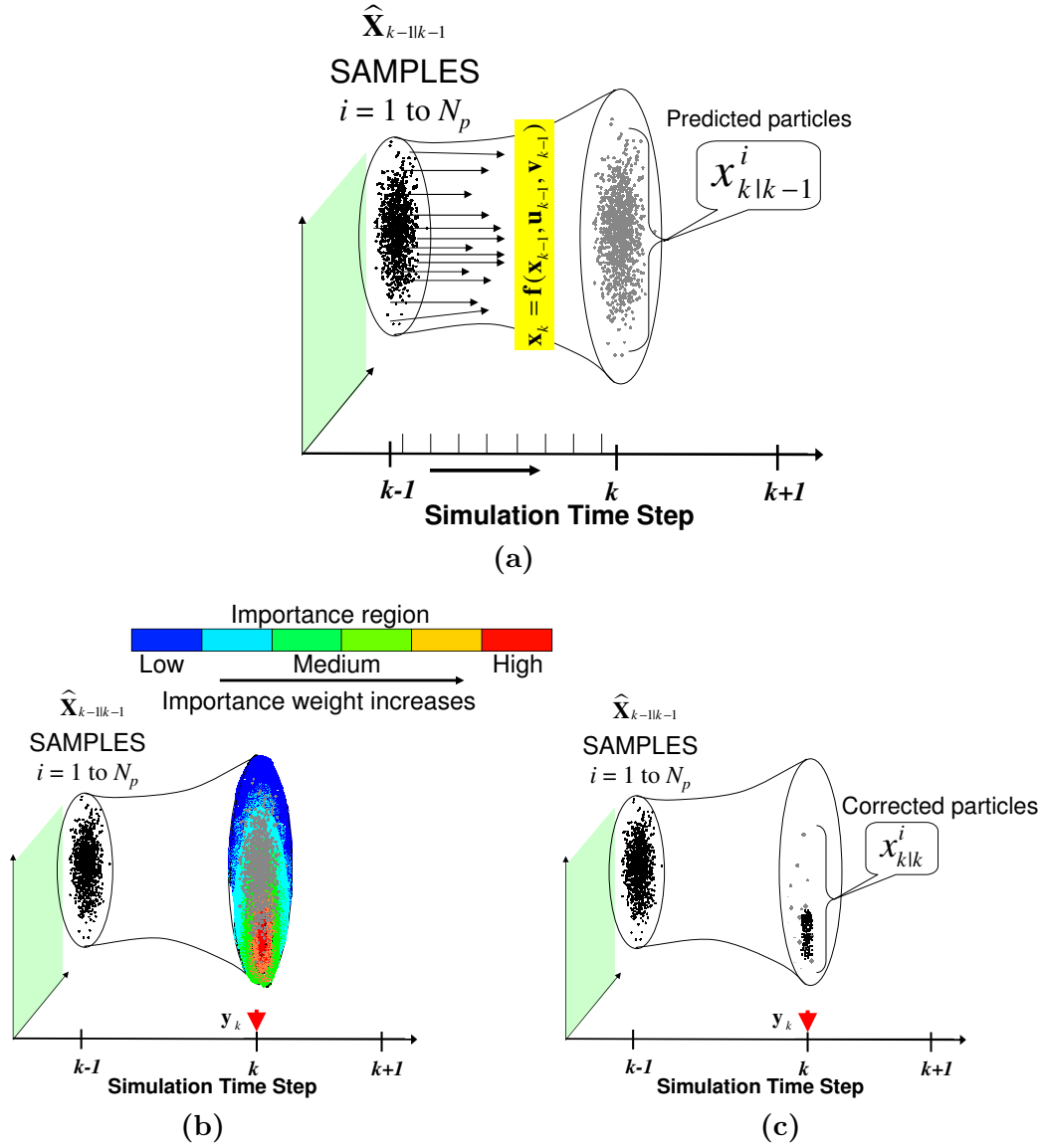


Figure 3.13: Illustration of the *prediction* and *correction* steps of the SIR-particle filter algorithm for one time instant k, showing: (a) prediction, (b) importance weight computation using the measurement $\mathbf{y}_k$, and (c) resampling of the particles using the importance weights. The predicted particles obtained in (a) are shown superimposed on the color contour regions in (b). The color contours are based on the magnitude of the importance weights. The correction/update step in the particle filter comprises of (b) and (c).

3.6 Comparison of Kalman Filter, Ensemble Kalman Filter and Particle Filter

In this section, the performance of the EnKF and the particle filter algorithm is analyzed, by comparing the estimation performance with Kalman filter, under various scenarios such as non-Gaussian process noise and measurement noise, nonlinear stochastic state space systems etc. In the previous example, our state-space system was linear with Gaussian process and measurement noise. In such scenarios, it is known that the Kalman filter is the optimal filter and gives the best estimate.

The Kalman filter (KF) assumes that the posterior density of the states is Gaussian at every time step and thus uses the mean and covariance value to characterize the distribution completely. Also, analytical expressions to recursively compute the covariance and mean of the Gaussian posterior density can be used, making the Kalman filter simple to implement. But the assumptions inherent in KF can be often highly restrictive in many practical situations. For example, if the system is nonlinear the above assumptions do not hold, even if the state and measurement noise processes are Gaussian. Another interesting case is when the distribution of the state/measurement noise is non-Gaussian. On the other hand, note that the EnKF/particle filter does not make any assumption of Gaussian noise or linearity of the system. The only requirement in such Monte Carlo filters is that the distribution of the noise should be known, and it should be possible to draw samples from this distribution. The particle filter can be considered as the most general solution for the state estimation problem because it practically makes no prior assumption about the nature of posterior/prior distribution. On the other hand, in the EnKF algorithm there is an inherent assumption that the distribution of the state prediction error and the innovations is Gaussian, and hence uses the ensemble covariance in the Kalman update equation. Hence, for any given system the particle filter can be expected to give better, if not similar, accuracy of estimation as that of the EnKF. However, the computational cost of the particle filter will be significantly high due to the large number of Monte Carlo simulations ($O(1000)$ particles) required to obtain reasonably accurate state estimates.

As mentioned earlier, we can expect that the accuracy of the estimate provided by the EnKF/particle filter algorithm will become better with more number of particles. This can be verified by doing MC simulations for different ensemble sizes/number of particles and observing the Root Mean Square Error (RMSE) of the estimation errors. Note that filters such as KF, EKF, UKF are deterministic and give a single RMSE value for one realization of the process. However, EnKF and PF being stochastic ones (utilize random sampling), the RMSE has to be evaluated by averaging over several Monte Carlo realizations of the filter on a single process

realization. The following sections present case studies comparing the performance of the EnKF and particle filter with that of KF as a function of the ensemble size/number of particles used. In each of the following case studies, one realization of the corresponding system (simulated for $k = 1$ *to* 100) was taken and used as data for state estimation.

3.6.1 Linear system

First, the system defined in Eq. (3.40) is considered, without changing any of the properties of the noise distributions. Figure 3.14 shows the plot of RMSE of estimation of the EnKF and particle filter plotted against the number of realizations in the ensemble/particles (N_e or N_p). Also, the RMSE obtained if the Kalman filter (KF) were used is shown. The RMSE of EnKF and the particle filter is obtained by averaging over 50 MC simulations. As one would expect, we can observe from Figure 3.14 that the KF gives the best performance for this linear system with Gaussian noise. Also, we can observe that the performance of KF is superior to the EnKF/particle filter when the number of particles is very low (<150). But as the number of particles is increased, the RMSE of the Monte Carlo filters are tending towards that of the Kalman filter. Clearly, this example illustrates the significance of using sufficient number of Monte Carlo realizations in an EnKF/particle filter to achieve near-optimal estimation accuracies.

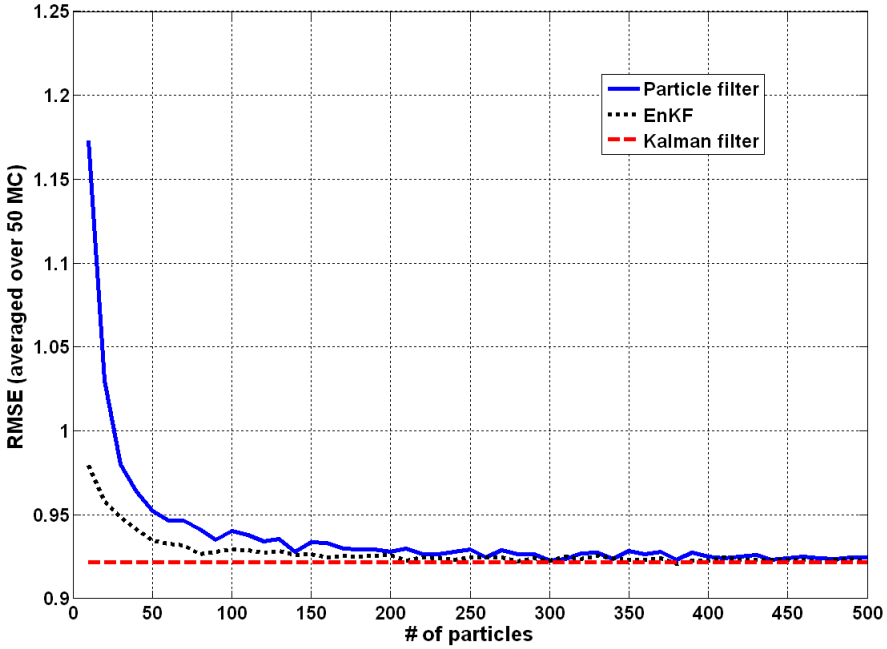


Figure 3.14: Performance comparison of particle filter, EnKF and KF for a linear process with Gaussian measurement noise.

Note that the 'curse of dimensionality' becomes a factor when the state space dimension is large. For this simple system, in order to get an estimation accuracy comparable to that of a KF, the particle filter requires more than 250 particles. As noted by Daum (2005), the computational complexity of filters such as particle filters increases exponentially with the number of states in the system. However, the computational complexity of KF grows only as cube of the state space dimension. Thus, one should prefer using the KF over a PF with large number of particles, for scenarios of linear systems with Gaussian noise such as the one considered here. The KF will also give the best optimal estimate in such cases.

The EnKF shows a much faster convergence to the optimal filter, with the RMSE reaching very close to that of the KF when the ensemble size is around 100. The main feature of the EnKF is that the solution space is limited to the span of the ensemble vectors. Hence, the dimension of the state estimation problem in the EnKF is given by the dimension of the ensemble space. Hence, even for large scale systems we can use a relatively small number of ensembles. However, the requirement is that the true states of the process should lie in the ensemble space. This can be satisfied by having a good knowledge about the initial conditions of the process states and also having better physical models of the process. This is true for large scale systems in earth, atmospheric and oceanographic sciences, thanks to the modeling effort by geophysicists, oceanographers and earth scientists. The EnKF has been shown to be a promising method in many practical applications in these areas of research. For example, the EnKF is today operational in the Canadian Meteorological Center (http://www.weatheroffice.gc.ca/ensemble/index_e.html) for obtaining ensemble weather forecasts through atmospheric data assimilation (Houtekamer & Mitchell, 2005).

In the previous study, it was assumed that the process noise and measurement noise is Gaussian, in which case the Kalman filter is known to be the optimal filter. In a second comparative study, this assumption was relaxed by assuming that the measurement noise $\mathbf{n}_k$ is non-Gaussian. The non-Gaussian density function assumed for the measurement noise, $p(\mathbf{n}_k)$, is shown in Figure 3.15. The normalized histogram of a realization of 100 samples drawn from the pdf is also shown superimposed in the figure. As shown in the figure, it was assumed that the measurement noise follows a bimodal distribution. Figure 3.16 shows the RMSE values of the particle filter and EnKF for different number of particles. The RMSE of the Kalman filter is also shown for comparison. The degradation in the performance of the KF caused by the non-Gaussian noise can be clearly observed from the figure. The EnKF performance converges to that of the Kalman filter for sufficiently high number of realizations in the ensemble. The particle filter gives the best performance compared to the EnKF and the KF, for approximately 50 or more number of particles. Note that the true

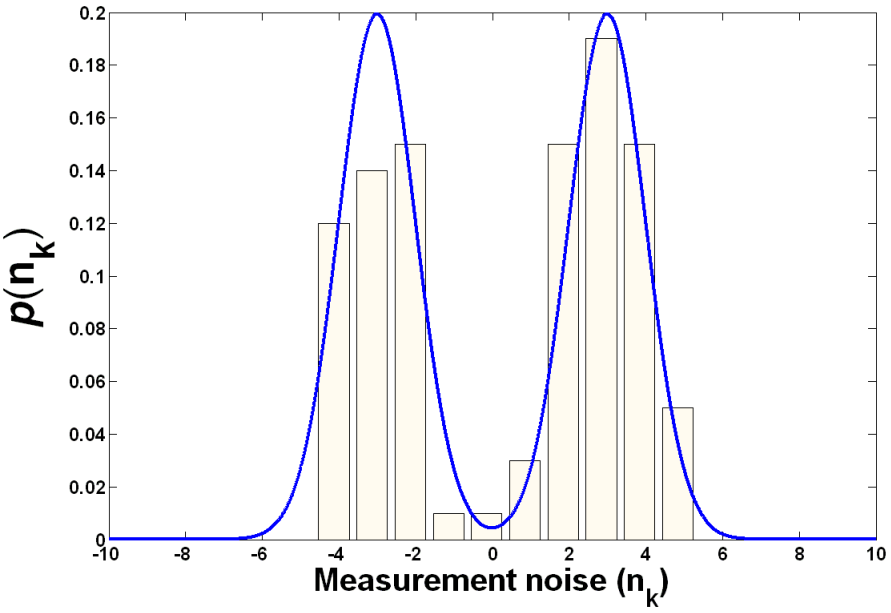


Figure 3.15: Bimodal pdf of measurement noise (n_k).

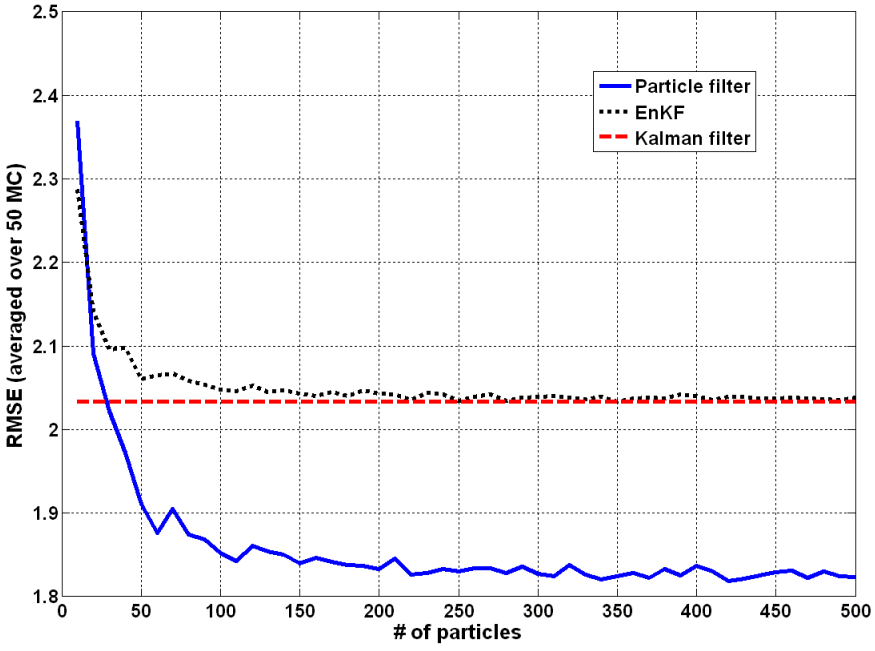


Figure 3.16: Performance comparison of particle filter, EnKF and KF for a linear process with non-Gaussian measurement noise.

bimodal noise distribution was used to draw noise samples in the implementation of the EnKF and the particle filter. This is a clear illustration that the EnKF algorithm is not able to completely account for the non-Gaussian likelihood resulting from the bimodal measurement noise distribution. This disadvantage is a result of the Kalman update used in the EnKF algorithm. However, in the particle filter we can properly incorporate the information about non-Gaussian likelihood at the sampling-importance-resampling step. It can be observed that when the number of particles is small (for example, when N_p<35), the performance of both Monte Carlo filters are worse than the Kalman filter, which once again illustrates the significance of using sufficient number of particles. The RMSE of the particle filter starts to improve significantly from that of the KF when $N_p > 50$. The advantage of using the particle filter is realized when the number of particles is around N_p =250, beyond which there is no significant improvement in the RMSE. There is no significant effect of the high number of particles on the computational complexity in this one dimensional case, making the particle filter as the best choice.

3.6.2 Nonlinear system

Next, the performance of the Monte Carlo filtering algorithms on a nonlinear system will be compared. First, a simple nonlinear process with Gaussian process and measurement noise will be considered, having the following state space representation:

$$x_{k+1} = x_k + v_k, \quad v_k \sim N(0, 5) \tag{3.43a}$$

$$y_k = 10^{-4} x_k^3 + n_k, \quad n_k \sim N(0, 1) \tag{3.43b}$$

$$x_0 = 1 \ (Initial \ \ state) \tag{3.43c}$$

$$y_0 = No \quad measurement \quad available \ at \ k = 0 \tag{3.43d}$$

The above system has the same random walk type of state transition model as the linear system considered earlier. However, the measurement equation is now nonlinear. The process and measurement noise are assumed to have the same variance as the linear system with Gaussian noise considered earlier. Note that the Kalman filter cannot be employed for this system because the measurement equation is nonlinear. Developing the PF and EnKF is straight forward for this nonlinear system, and only requires plugging in the state space equations to the respective prediction and correction steps. Similar to the previous studies, the comparison was performed by using one realization of the above system simulated for 100 sampling instants. Using the measurements from the simulation, the EnKF and PF algorithms were executed to estimate the states. The ensemble size/number of particles was varied in the range of 10 to 500. The RMSE of estimation of the

two filters, averaged over 100 MC, is shown in Figure 3.17 as a function of number of particles.

From Figure 3.17, it can be observed that the estimation performance of both the filters improve with increasing number of particles and are close to each other for $N_p \geq 300$ particles. Note that the PF does not make any assumption about the distribution of the states. Nevertheless, the performance of PF (with $N_p \geq 300$) and EnKF are similar for this particular nonlinear system. Although to some extent the EnKF shows a better performance than the PF, this result may not be generalized for any class of nonlinear systems. As will be shown in the next example, the performance of the EnKF will degrade compared to that of the PF when the likelihood becomes significantly non-Gaussian. Thus, the effect of nonlinearity on the posterior density of the states for different systems will have to be investigated on a case-by-case basis to understand under what scenarios the Gaussian assumption of the EnKF is reasonable to give a performance comparable to that of the PF.

The case of non-Gaussian measurement noise in the above system was also considered to compare the performance of the filters. Here, the same bimodal pdf for $\mathbf{n}_k$ shown earlier in Figure 3.15 was used. Figure 3.18 shows the plot of RMSE of estimation against the number of particles. It can be clearly seen that the PF outperforms the EnKF. The lower performance of the EnKF can be attributed to the presence of non-Gaussian measurement noise, which will make the distribution of the innovations significantly non-Gaussian and far away from the assumption of Gaussianity in the Kalman update rule.

3.7 Concluding Remarks

This chapter presented a tutorial overview of the concept of state estimation/filtering from a Bayesian viewpoint. The main objective of state estimation is to combine prior knowledge about the process along with information from the real process data measured through sensors, to estimate certain unknown quantities. Usually, the quantities so estimated are the ones necessary for decision making in the face of uncertainty. The uncertainty is mainly due to three reasons: (1) the lack of an exact physio-chemical model of the process, (2) unmeasured external disturbances affecting the process, and (3) random noise contaminating the measurements. A detailed description of two filtering algorithms, namely the particle filter and the ensemble Kalman filter (EnKF), which are suitable for state estimation of nonlinear systems was presented. Both these filters belong to the class of algorithms known as sequential Monte Carlo filtering algorithms. The ensemble Kalman filter can be defined as a Monte Carlo version of the Kalman filter algorithm. In the EnKF algorithm, the analytical solution provided by the Kalman filter for linear state space systems is extended for nonlinear systems through the use of statistical samples.

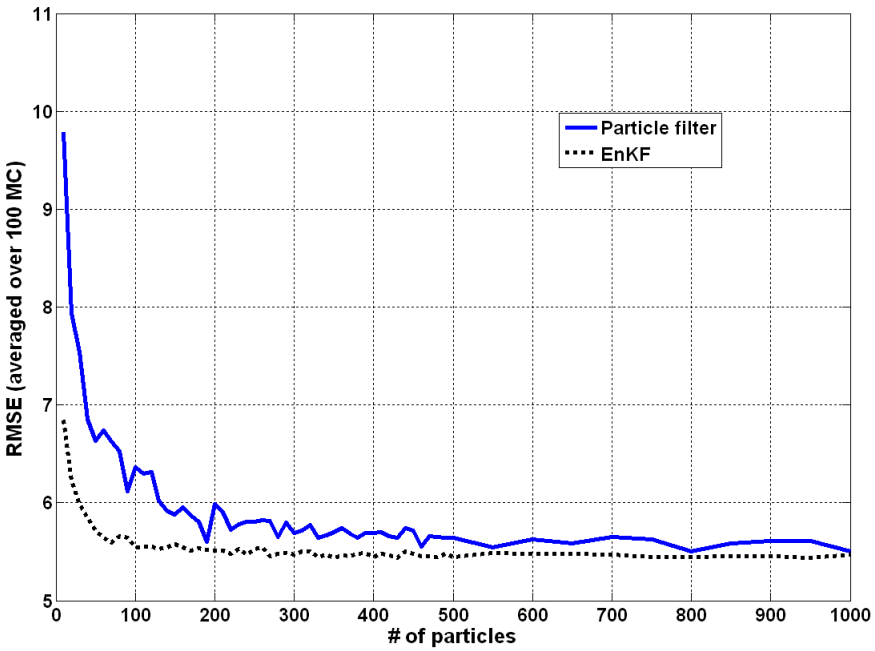


Figure 3.17: Performance comparison of the particle filter and the EnKF for a nonlinear process with Gaussian measurement noise.

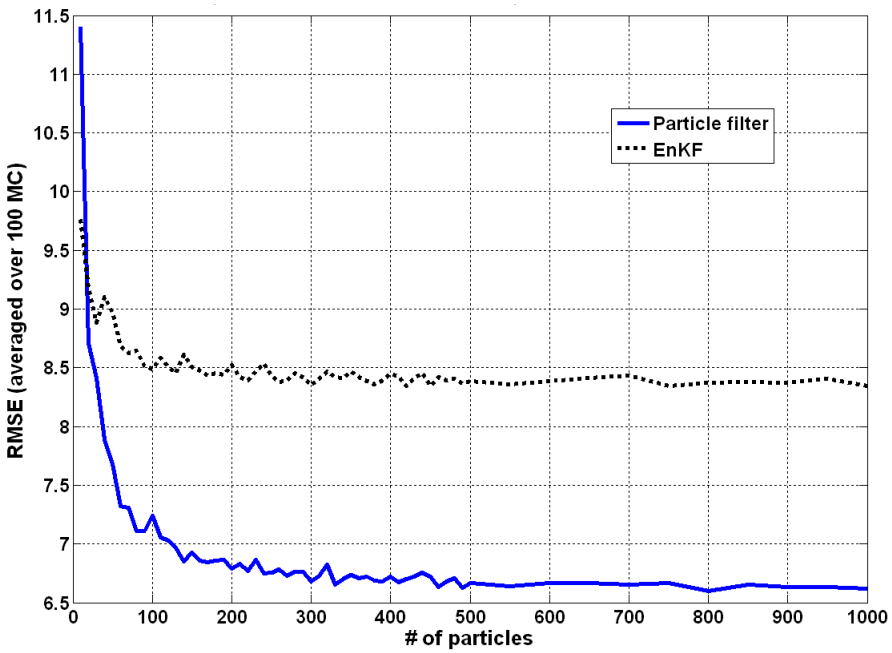


Figure 3.18: Performance comparison of the particle filter and the EnKF for a nonlinear process with non-Gaussian measurement noise.

The particle filter is a more general solution to the problem of state estimation for nonlinear systems. The particle filter is purely based on Bayes' rule whose analytical implementation is intractable for the state estimation problem. In the particle filter algorithm, the Bayes' rule is incorporated through Monte Carlo simulations by a method known as *sampling importance resampling* of statistical samples. Simple examples of univariate linear and nonlinear systems were used to demonstrate the application of the algorithms. The examples showed the performance of these filters in different scenarios, which were different combinations of linear/nonlinear systems with Gaussian/non-Gaussian process and measurement noise. The optimality of the particle filter was shown on a linear system with Gaussian process and measurement noise, where the RMSE of the estimated variable was observed to converge to that of the Kalman filter when the number of particles is close to 300. The EnKF also showed similar convergence on such systems. However, for systems in which the measurement noise was assumed to be non-Gaussian, the particle filter outperformed the EnKF/KF with a significantly lower RMSE for particle population sizes of 100 and above. The particle filter will require an enormous number of Monte Carlo simulations for systems whose state space dimension is high and this can cause an exponential rate of increase in the computational cost of the particle filter. This disadvantage limits the applicability of this method today to only low dimensional systems. For large scale nonlinear systems, the EnKF is a good choice and is today operational in some real world weather forecasting systems. An application of the EnKF algorithm for characterization of large scale oil and gas reservoirs will be discussed in a later chapter of this thesis.

Bibliography

Boutayeb, M., Rafaralahy, H. ., & Darouach, M. (1995). Convergence analysis of the extended Kalman filter as an observer for non-linear discrete-time systems. In *Proceedings of the 34th Conference on Decision & Control, New Orleans, LA*, (pp. 1555–1560).

Burgers, G., van Leeuwen, P. J., & Evensen, G. (1998). Analysis scheme in the ensemble Kalman filter. *Monthly Weather Review*, *126*, 1719–1724.

Daum, F. (2005). Nonlinear filters: Beyond the Kalman filter. *IEEE Aerospace and Electronic Systems Magazine*, *20*(8), 57–69.

Eckhardt, R. (1987). Stan Ulam, John von Neumann and the Monte Carlo method. *Los Alamos Science - Special Issue, Stanislaw Ulam 1909-1984*, *15*, 131–137.

Evensen, G. (1994). Sequential data assimilation with a nonlinear quasi-geostrophic

model using Monte Carlo methods to forecast error statistics. *Journal of Geophysical Research*, *99*, 10143–10162.

Evensen, G. (2003). The ensemble Kalman filer: theoretical formulation and practical implementation. *Ocean Dynamics*, *53*, 343–367.

Evensen, G. (2007). Estimation in an oil reservoir simulator. In *Data Assimilation: The Ensemble Kalman Filter* (pp. 239–248). Springer, Berlin.

Evensen, G. (2009). The ensemble Kalman filter for combined state and parameter estimation. *IEEE Control Systems Magazine*, *29*(3), 82–104.

Gordon, N. J., Salmond, D. J., & Smith, A. F. M. (1993). Novel approach to nonlinear/non-gaussian Bayesian state estimation. *Radar and Signal Processing, IEE Proceedings F*, *140*(2), 107–113.

Houtekamer, P. L. & Mitchell, H. L. (2005). Ensemble Kalman filtering. *Quarterly Journal of the Royal Meteorological Society*, *131*, 3269–3289.

Julier, S. & Uhlmann, J. (1997). New extension of the Kalman filter to nonlinear systems. In Kadar, I. (Ed.), *Signal Processing, Sensor Fusion, and Target Recognition VI*, volume 3068 of *SPIE proceedings*, (pp. 182–193).

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, *82*(Series D), 35–45.

Maybeck, P. S. (1979). *Stochastic models, estimation, and control*, volume 141 - 1 of *Mathematics in Science and Engineering*. Academic Press, London.

Metropolis, N. (1987). The beginning of the Monte Carlo method. *Los Alamos Science - Special Issue, Stanislaw Ulam 1909-1984*, *15*, 125–130.

Ristic, B., Arulampalam, S., & Gordon, N. (2004). *Beyond the Kalman Filter*. Artech House, Boston.

Rubin, D. B. (1983). Progress report on project for multiple imputation of 1980 codes. Technical report, U.S.: Census Bureau, NSF and SSRC.

Rubin, D. B. (1988). Using the SIR algorithm to simulate posterior distributions. *Bayesian Statistics*, *3*, 395–402.

Seiler, A., Evensen, G., Skjervheim, J.-A., Hove, J., & Vabo, J. (2009). Advanced reservoir management workflow using an EnKF based assisted history matching method. In *SPE Reservoir Simulation Symposium, Woodlands, TX*. Society of Petroleum Engineers, Inc.

Chapter 4

Detection and Quantification of Valve Stiction by the Method of Unknown Input Estimation[1,2]

4.1 Introduction

The problem of stiction in valves is well known to be one of the primary causes of oscillations in industrial process control loops (Ruel, 2000). Such oscillations can easily propagate to other control loops and degrade the overall closed loop performance of the process. The higher variability in the process variables due to such oscillatory loops will be reflected in the final product, in the form of larger variations in the product quality. Based on an industrial survey, Desborough & Miller (2001) have reported that control valve problems account for about one-third of the 32% of controllers classified as 'poor' or 'fair'. The detection and quantification of valve stiction in such loops is the first step towards a diagnosis scheme in order to alleviate the effect of a sticky valve on closed loop performance.

Jelali & Huang (2009) and Choudhury et al. (2008) have presented a comprehensive review of the state of the art on stiction detection and quantification methodologies. Horch (2006) has presented a detailed comparative study of the different methods of automatic detection of stiction that are available in the published literature. The main conclusion of this study was that different methods "have their strengths and application areas" , "rely on different assumptions" and "there is not one method which can cover all cases reliably". Jelali & Scali (2009) have presented an extensive comparison of different valve stiction techniques on a benchmark data set from 93 industrial loops. The aforementioned paper also provides detailed re-

[1] An almost full version of this chapter has been published in the Journal of Process Control (Chitralekha et al., 2010).

[2] Abbreviated sections of this chapter were also presented at the 7th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes, Barcelona, Spain, June 30 – July 3, 2009 (Chitralekha et al., 2009).

sults and conclusions about the merits of various techniques in terms of sensitivity to measurement noise, data requirements, implementation difficulties and many other practically relevant factors.

We can broadly classify the stiction detection methods into two categories: data-based and model-based approaches. Horch (1999) presents a data-based method which is based on the cross-correlation between the controller output (op) and process output (pv). This method cannot be applied to integrating processes, for which in a later work Horch & Isaksson (2001) have presented another method based on the distribution of the second derivative of the pv signal. Rengaswamy et al. (2001) and Yamashita (2006) have presented data based methods based on the qualitative shapes in the time trends of the op and pv signals. In a more recent work, Scali & Ghelardoni (2008) present an improved version of the shape based method by also including additional stiction patterns observed in industrial data sets. The shape based method was shown to be reliable for flow control loops. Peter He et al. (2007) and Ahmed et al. (2009) have proposed methods which are based on the triangular shape of the op or pv signals for detecting stiction in self-regulating and integrating processes. In another study, Choudhury et al. (2006) present a method for detecting and quantifying stiction for linear processes using the pv and op data. This method is based on the fact that for a linear process under closed loop control, a sticky valve would induce nonlinearity in the pv and op signals and hence stiction can be detected based on the nonlinearity in the control error signal. The quantification part was based on the pv vs. op plot, which was found to be elliptical in shape and the width of the ellipse was used to quantify stiction. This type of quantification is very useful since it expresses stiction as percentage of the valve travel which is the practice in the process industry for quantifying stiction. The disadvantage of this quantification methodology is that the width of the ellipse will be dependent on the effect of loop dynamics (mainly controller tuning) on the pv. Therefore, the estimated width of the ellipse is termed as 'apparent stiction'. On the other hand if the valve position (mv) data is available, then a simple plot of mv and op can be used to obtain the actual stiction. For a sticky valve, the mv vs. op plot will follow well defined patterns depending on the type of stiction as shown in an earlier work by Choudhury et al. (2005). The true stiction can be quantified using an appropriate width measure of this pattern.

Several authors have proposed model-based approaches for stiction detection and quantification. Stenman et al. (2003) has presented a model-based approach based on ideas from the field of change detection and multi-model mode estimation. A stiction model defined by a binary mode parameter was proposed and used in their approach. Stiction detection was performed through a combined identification of the process model parameters and the mode sequence. Srinivasan et al. (2005),

Choudhury et al. (2008), Jelali (2008) and Lee et al. (2009) have proposed model-based methods where a linear process with a sticky valve is considered to be a Hammerstein system, the sticky valve being the nonlinear element in the system. A suitable valve stiction model was used to define the structure of the nonlinear part. These methods involve explicit identification of the parameters of a stiction model, along with the process model parameters, through an optimization procedure. The validity of the stiction model structure chosen in this method is crucial to the success of the method and confirmation of this may be possible only through a post-results analysis.

In any of the methods mentioned hitherto, the *mv* vs. *op* plot was not used for the quantification part since for most practical cases the valve positioner data (available only for 'smart valves') is not available. Note that with the technological advancements of today, there are valve position sensors which can measure the stem position of the valve. However, the cost of such valves will be significantly high compared to the legacy valves which are still the most popular ones. In the current work, a novel method is proposed in which the valve position can be estimated by the method of unknown input estimation using a process model and the *pv*, *op* data available from the process. Such an approach can be described as a 'virtual smart valve'. Since most of the control loops with a valve have linear controllers, usually PI(D)s, which are capable of giving satisfactory regulation, the assumption of a linear process is reasonable. A major contribution of this work is the application of the unknown input observer (UIO) proposed by Gillijns & De Moor (2007) and its use in this particular study for estimating the *mv*. The two-parameter empirical model developed in Choudhury et al. (2005) was used in all the simulation studies in this work. However, note that the proposed stiction detection and quantification methodology does not assume any stiction model. The two-parameter stiction model was chosen over other models (which are physical models) since it was shown by the authors to be effective in capturing all the different modes of real sticky valves by specifying just two parameters of the model. Moreover, this model is generic in the sense that it requires only the knowledge of the type of stiction to simulate and hence it is more suitable for studying the performance of the proposed stiction detection and quantification algorithm.

This chapter is organized as follows. In Section 4.2, the formulation of the stiction detection problem as an unknown input estimation problem is explained. The phenomena of how stiction manifests on the *mv*-*op* data plot and how stiction quantification can be carried out using the plot is explained in Section 4.3. In Section 4.4, the application of the method is demonstrated on simulated examples. The simulation studies will show how the quantification is not significantly affected by model-plant mismatch, external disturbances, and controller tuning, which are

attractive features for applying this method to real world problems. In Section 4.5, the results of application of this method to a flow control loop in a pilot-scale process is presented. In Section 4.6, two industrial case studies are presented where the loop was exhibiting limit cycle behavior: one with a sticky valve and another with a normal valve, where oscillations were due to external causes in the latter case.

4.2 Problem Formulation

Consider the closed loop block diagram as shown in Figure 4.1, where a sticky valve is included between the process and the controller block. Typically, the controller will be P or PI. For the general case the *pv* can be the level, flow rate, temperature, pressure or composition measurement. The controller commands the valve to open to a particular position (units of % opening) through the *op* signal. The *mv* signal (units of % opening) represents the actual percentage opening performed by the valve. For a normal valve, the *mv* and *op* signal will be equal at all times. But if the valve is sticky, then there will be a clear difference between the two signals. In such a situation the valve acts like a nonlinear element transforming the *op* signal. If we model this transformation of the *op* by adding an additive, nonlinear, external signal which enters the loop just after the controller output, we get an equivalent representation as shown in Figure 4.2 with the valve block replaced by an external unknown input signal. Clearly, this external signal will be given by the difference between *mv* and *op* i.e., $\widetilde{u} = (u^V - u)$. This signal is defined as 'unknown' because in most real world systems we do not know the *mv* and the problem of getting an estimate of *mv* can be solved if we can estimate $\widetilde{u}$.

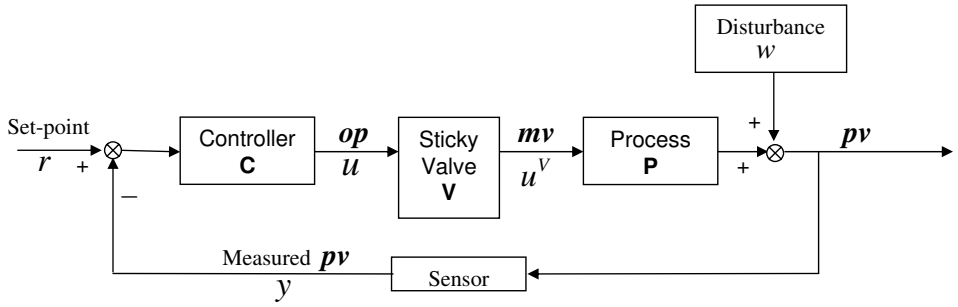


Figure 4.1: Closed loop with a "Sticky" valve.

The process in the above scenario can be assumed to be a univariate process with a P or PI controller to control it. Assume that the process can be described as a

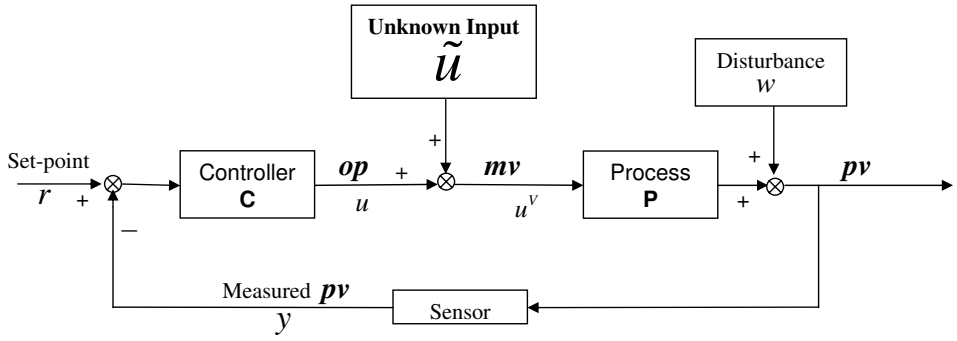


Figure 4.2: Equivalence between a sticky valve in a loop and an external unknown input signal.

univariate linear discrete time system with the following state-space representation

$$\begin{aligned} x_{k+1} &= a_k x_k + b_k u^V_{k-L+1} + w_k \\ y_k &= x_k + v_k \end{aligned} \tag{4.1}$$

where $x_k \in \Re$ is the process state, $u^V_{k-L+1} \in \Re$ is the input to the process (valve position), L accounts for process dead-time and $y_k \in \Re$ is the measurement available from the process. The process noise $w_k \in \Re$ and measurement noise $v_k \in \Re$ are assumed to be Gaussian with variance Q and R respectively. With the above system, we can use the method of unknown input estimation proposed by Gillijns & De Moor (2007) in order to obtain an estimate of u^V. This estimator is a Kalman filter type recursive estimator, which uses the process model in the form of Eq. (4.1) and the sequence of input and output measurements to estimate the unknown input. The main assumptions of the unknown input estimation method proposed by Gillijns & De Moor (2007) are the following: (1) the linear state space model in Eq. (4.1) is observable, and (2) an unbiased estimate of the initial state x_0 is available. The model parameters a_k and b_k have to be known/chosen *a priori*. They can be chosen based on the *a priori* knowledge about the gain and time constant of the process; also, an identified linear model of the process can be used. Any delay in the system is assumed to be known or estimated and should be incorporated in the state-space model of the process. In the following discussions, it is assumed that $L = 1$, i.e., a unit delay exists between the input and the state variables, without loss of generality. Q and R have to be considered as Kalman filter type tuning parameters in the unknown input estimator. The measurement error variance, R, can be estimated from past measurements from a steady state operation of the process. In the current context, the input is not completely known and is given by

$$u^V_k = u_k + \widetilde{u}_k \tag{4.2}$$

where $\widetilde{u}_k$ is the unknown portion which has to be estimated. Substituting Eq. (4.2) into Eq. (4.1), we obtain the state-space system with the unknown input explicitly as

$$\begin{aligned} x_{k+1} &= a_k x_k + b_k u_k + b_k \widetilde{u}_k + w_k \\ y_k &= x_k + v_k \end{aligned} \tag{4.3}$$

The recursive form of the unknown input estimator is then given by

$$\widehat{x}_{k|k-1} = a_{k-1}\widehat{x}_{k-1|k-1} + b_{k-1}u_{k-1} \tag{4.4}$$

$$\widehat{\widetilde{u}}_{k-1} = M_k(y_k - \widehat{x}_{k|k-1}) \tag{4.5}$$

$$\widehat{x}^*_{k|k} = \widehat{x}_{k|k-1} + b_{k-1}\widehat{\widetilde{u}}_{k-1} \tag{4.6}$$

$$\widehat{x}_{k|k} = \widehat{x}^*_{k|k} + K_k(y_k - \widehat{x}^*_{k|k}) \tag{4.7}$$

where M_k and K_k are obtained using the method given in Gillijns & De Moor (2007). In order to obtain a minimum-variance unbiased (MVU) estimate of $\widetilde{u}_{k-1}$, the gain matrix M_k is calculated based on a weighted Least Square (WLS) estimation scheme. For this, consider the following definition of the innovation:

$$\begin{aligned} \widetilde{y_k} &\triangleq y_k - \widehat{x}_{k|k-1} \\ &= b_{k-1}\widetilde{u}_{k-1} + e_k \end{aligned} \tag{4.8}$$

where, e_k is given by

$$e_k = a_{k-1}(x_{k-1} - \hat{x}_{k-1|k-1}) + w_{k-1} + v_k \tag{4.9}$$

In Gillijns & De Moor (2007), the weighting matrix $(E[e_k e_k^T])^{-1}$, i.e., inverse of $\mathrm{Cov}(e_k)$, was used for the WLS estimation of the input. Based on Eq. (4.9), we obtain

$$\begin{aligned} \tilde{R}_k &\triangleq E[e_k e_k^T] \\ &= a_{k-1}^2 P_{k-1|k-1} + Q + R \end{aligned} \tag{4.10}$$

where, $P_{k|k} \triangleq E[\tilde{x}_k \tilde{x}_k^T]$ and $\tilde{x}_k \triangleq x_k - \hat{x}_{k|k}$. For the univariate system in Eq. (4.8), we then obtain

$$\begin{aligned} M_k &= (b_{k-1}^2 \tilde{R}_k^{-1})^{-1} b_{k-1} \tilde{R}_k^{-1} \\ &= 1/b_{k-1} \end{aligned} \tag{4.11}$$

by applying the Theorem 2 given in Gillijns & De Moor (2007). Based on similar ideas and starting from Eq. (4.6), an optimal gain matrix K_k can be calculated which will give an MVU estimate of the state. The interested reader is referred to Gillijns & De Moor (2007) for a detailed derivation of the expressions for M_k and

K_k in a more general setting. In Gillijns & De Moor (2007), the authors have shown that the method will give a minimum-variance unbiased estimate if the process model exactly matches the plant and an unbiased estimate of the initial state x_0 is available. Another attractive feature of this method is that no prior knowledge about the nature of the unknown input is required (Gillijns & De Moor, 2007). Using the estimated unknown input signal $\widehat{\widetilde{u}}_k$, we can reconstruct the valve position signal by substituting for $\widetilde{u}_k$ in Eq. (4.2) to obtain,

$$\widehat{u}_k^V = u_k + \widehat{\widetilde{u}}_k \tag{4.12}$$

Note that one can use any type of unknown input observer that is available in the literature, or even a simultaneous state and parameter estimation algorithm, in order to do the above task of estimating the valve position from the process model.

4.3 Detection and Quantification: mv-op plot

A typical control valve consists of two major components, namely the valve body housing and the actuation unit (Kayihan & Doyle III, 2000). Figure 4.3 shows the basic parts of a typical pneumatic control valve. The valve body consists of a valve seat, a valve plug attached to a valve stem and a valve packing. The clearance between the valve plug and the seating, which determines the flow rate of the fluid, is varied by the motion of the valve stem. The stem is connected to a diaphragm in the actuation unit which applies the required force to move it in a controlled fashion. The friction force between the valve stem and the valve packing helps to keep the stem in position. Also, the packing prevents leaks from the valve body. During the course of its operation, when the valve stem becomes temporarily stationary, this friction force can often become fairly large due to static friction. Stiction is basically the phenomena of the valve stem becoming temporarily immobile until the force applied by the diaphragm is able to overcome this static friction.

The phenomena of valve stiction can be easily understood using the phase plot of input-output behavior as shown in Figure 4.4, as reproduced from Choudhury et al. (2005). In this figure, the portion AC and EF correspond to the region when the valve is sticking. It is composed of a dead band (AB) and a stick band (BC). The points A and E correspond to instances when the valve stem is changing its direction of motion and hence comes to rest momentarily. At these points, the static friction comes into play. In order to overcome the static friction, the controller output has to change by a value equal to the width AC or EF. At C (or F) the valve jumps (slip jump) to a new position, D (or G), and continues to move. A sticky valve under closed loop control will follow this cyclic pattern, causing the loop to have undesirable oscillations which will be reflected in the process variable.

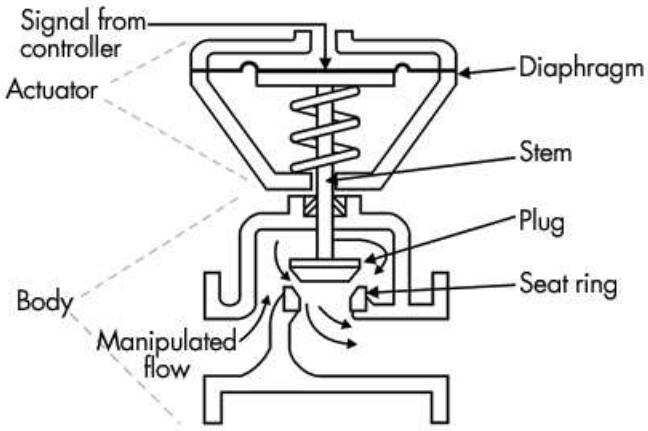


Figure 4.3: Control valve with actuator (Courtesy: InTech Magazine - April 2005, ISA).

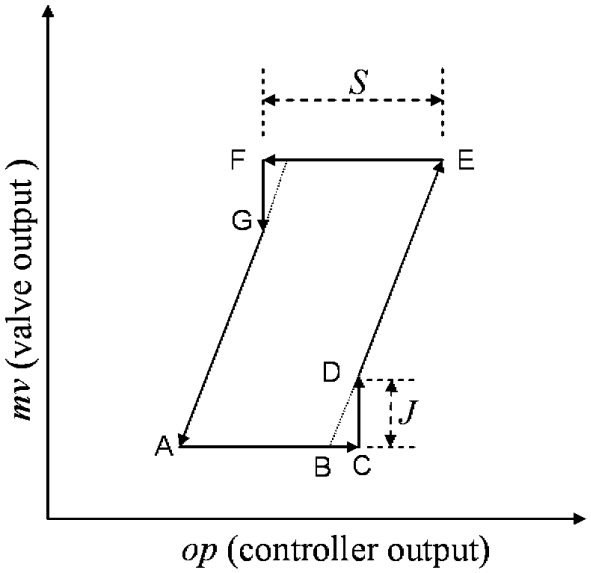


Figure 4.4: Sticky valve input-output behavior.

Based on this input-output behavior of a sticky valve, an empirical model containing two parameters was proposed in Choudhury et al. (2005). The first parameter quantifies the width of the region in which the valve is sticking (AC or EF) and is denoted by S. The second parameter captures the slip jump phenomena (CD or FG) and is denoted as J. Both S and J can be specified in terms of percentage travel of the valve. This model is used in the simulation examples discussed in the next section.

In order to detect whether a loop is oscillating due to valve stiction, one can visualize and check the *mv-op* plot for the above cyclic pattern. The width of the

cyclic pattern with respect to the *op* axis (= S) can be considered to be directly related to the amount of stiction present in the valve and hence can be used to quantify stiction (Choudhury et al., 2006). However, for the majority of the valves in industry the valve positioner data (*mv*) is not available. On the other hand, the unknown input estimator described in the previous section can be used to obtain an estimate of the *mv* and a pattern similar to Figure 4.4 can be identified in order to detect and quantify stiction. In the process industry the *op* signal is usually available in units of percentage opening of valve, which means that stiction will also be quantified as percentage of the valve travel. This type of quantification is intuitively understandable and also can be used for any valve irrespective of the valve type.

4.4 Simulation Study

In this section, the efficacy of the proposed stiction detection and quantification algorithm is demonstrated through simulation examples, where the stiction is introduced in the closed loop simulation studies using the stiction model proposed in Choudhury et al. (2005). First, a simple integrating process with the process gain being unity (i.e., $G(s) = 1/s$) and controlled by a discrete PI controller is considered. The two parameter stiction model was included in between the process and the controller in closed loop simulation.

A discrete time implementation of the PI controller represented by the transfer function $Kc(1 + \frac{1}{\tau_I(1-z^{-1})})$, where $Kc = 0.1$ and $1/\tau_I = 0.01$, was used in the simulation. A Gaussian white noise signal, with a variance of 0.01, was introduced as disturbance in the simulation. A sampling time of 1 sec. was used in all the simulation studies. The parameter values of $S = 4$ and $J = 2$ were used in the stiction simulation block. Due to the presence of stiction, the loop becomes oscillatory. Figure 4.5 shows the time trends of the *pv* and *op* signals, after the closed loop system has attained steady state.

In the unknown input estimator, the unity gain integrator model for the process was used and the unknown input was estimated to obtain the valve position signal $\widehat{u}_k^V$. The state noise variance (Q) and measurement noise variance (R) values of 1e-5 and 1e-3, respectively, were used in the estimator. Figure 4.6 shows the true and estimated *mv*-*op* plot. Note that in this simulation the true *mv* is naturally available. The noise in the simulation is causing the random variation in the estimated *mv*-*op* plot compared to the true phase plot. Nevertheless, we can clearly see the signature of stiction present in the estimated *mv*-*op* plot pattern and hence we can detect stiction visually from this figure. As mentioned earlier, the width of the *mv*-*op* plot quantifies stiction. In order to estimate the width of the pattern, a trapezoid was fitted to the *mv*-*op* data which is overlapped onto the estimated *mv*-*op* plot. The

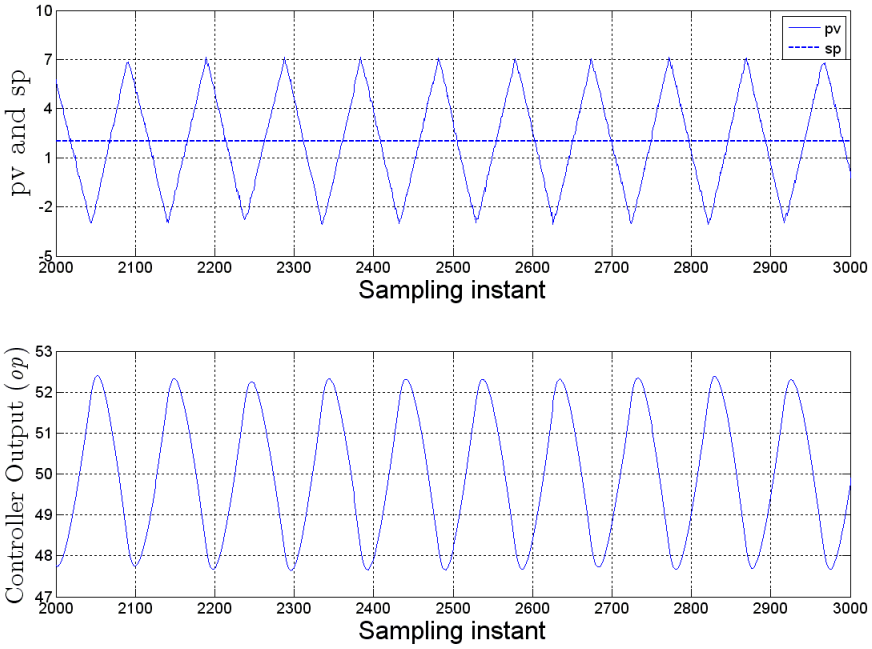


Figure 4.5: Simulated data set with stiction.

trapezoid fitting is done by solving a constrained optimization problem to find the four corner points of the polygon, where the objective is to find the least squares error of fit to the *mv-op* data. The corner points were constrained to fall within the range of the data. Constraints are also imposed on the points so that two of the edges will be parallel to the horizontal axis and these edges will correspond to the sticking region of the valve. The estimated width is denoted as $\widehat{S}$. For the quantification part, it is only the width of the *mv-op* plot that is of interest and in the proposed scheme, if the model is fairly precise, then $\widehat{S}$ will be close to S. In the simulated example, the estimated width matches the parameter S exactly and thus we can conclude that the quantification is accurate.

4.4.1 Model plant mismatch

The previous example demonstrated how the method can be applied for detecting stiction using input-output data and a process model. In this section, the performance of the algorithm when there is a mismatch between the coefficients of the plant and the model will be studied. Robustness to model-plant mismatch is an important feature of any model based scheme if it has to be applied in practice. The performance of the algorithm should not be significantly affected by model-plant mismatch (MPM). This is important because in the real world it is not practical to

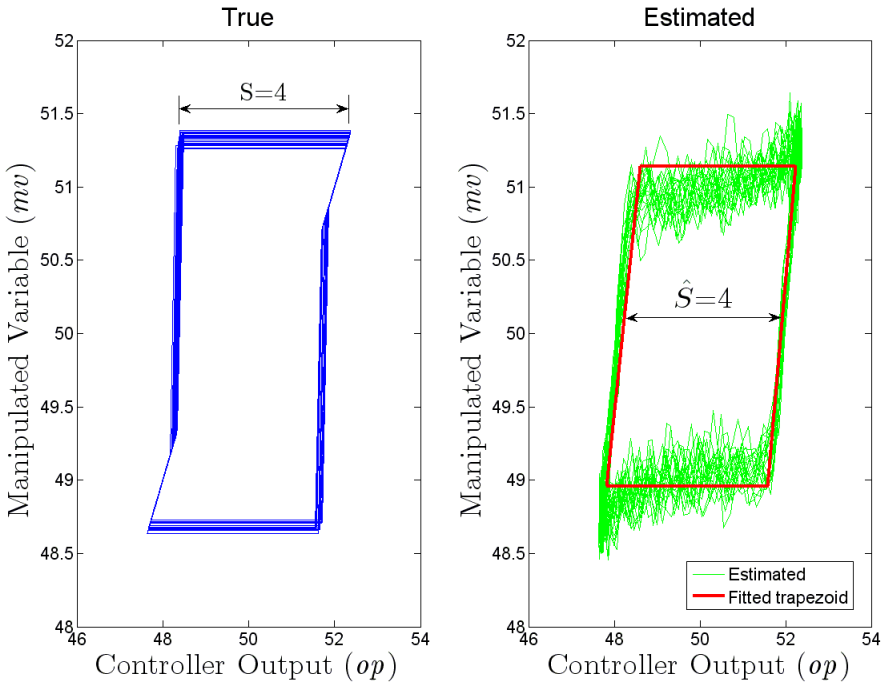


Figure 4.6: *mv-op* plot (Stiction model parameters: S=4, J=2).

obtain an exact model of the process.

In order to study the effect of MPM on the stiction quantification results, the performance of the method was evaluated on a first order process with time delay (FOPDT). The parameters of the FOPDT process and the model (with mismatch) are shown in Table 4.1. As shown in the table, the gain and time constant parameters have a 10% mismatch. A unit sample time (=1 sec.) mismatch was introduced in the delay. For each case of MPM, the S parameter was varied in the range of 0.5 to 10 (in steps of 0.5) and the percentage error in the stiction estimation was calculated. The J parameter was set at $0.5S$. In order to get a realistic value of the estimation error, the average estimation error was taken over 50 Monte Carlo simulations of the closed loop with different realizations of simulation noise.

Table 4.1: FOPDT parameters for process and model (with mismatch).

Parameters	Gain	Time constant	Delay (secs)
Process	**1**	**15**	**2**
Model (with mismatch)	**0.9**	**16.5**	**1**

Figure 4.7 shows the estimation error for different levels of stiction introduced in the loop, for the cases of with and without MPM. It can be observed that the error

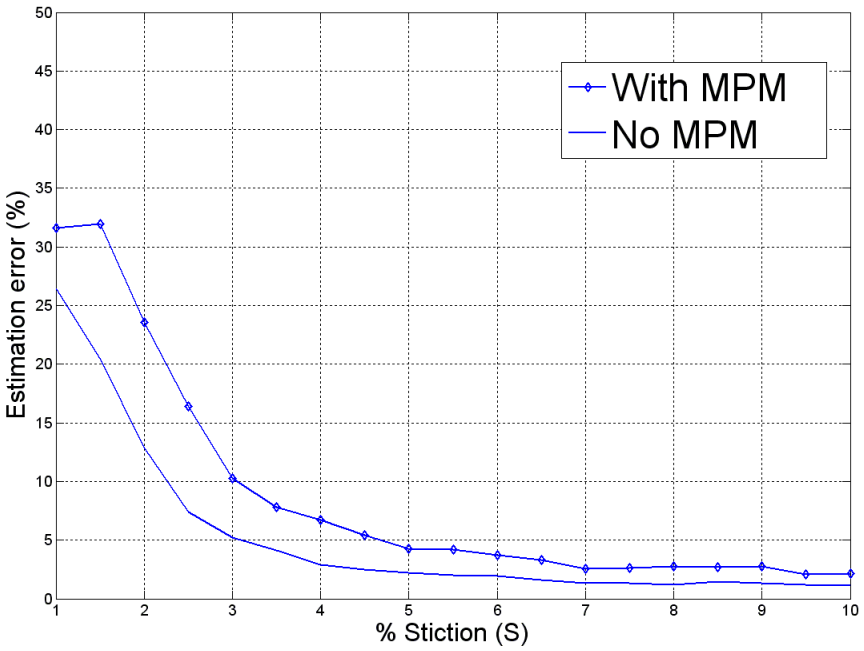


Figure 4.7: Stiction estimation error for a FOPDT process in the presence of MPM.

stays below 20% for most of the cases, except when the $S \leq 2$ in which case the error is around 30%. From the Fault Detection and Identification literature, it is well known that estimating any low magnitude fault, such as stiction with magnitude 2% or lower, with high accuracy is difficult to achieve; the low magnitude faults usually get confounded with process noise and disturbances.

The unit gain integrating process illustrated earlier was also considered here. Different levels of gain mismatch (through parameter b_k) was introduced. The results are summarized in Figure 4.8. The estimation error stays below 15% of the actual stiction value even at the high level of MPM considered here, for all stiction levels except for the case of a low stiction ($S = 1$).

Based on the above two simulation case studies, we can conclude that the proposed algorithm gives reliable estimates even in presence of model-plant mismatch. The robustness to MPM observed in the above simulation studies also indicates that the width of the estimated *mv-op* plot, which quantifies stiction, is fairly invariant to the effect of MPM.

4.4.2 External oscillatory disturbance

In a typical control loop shown in Figure 4.1, it is very likely that oscillations occur due to reasons other than stiction. A very common scenario is when an external

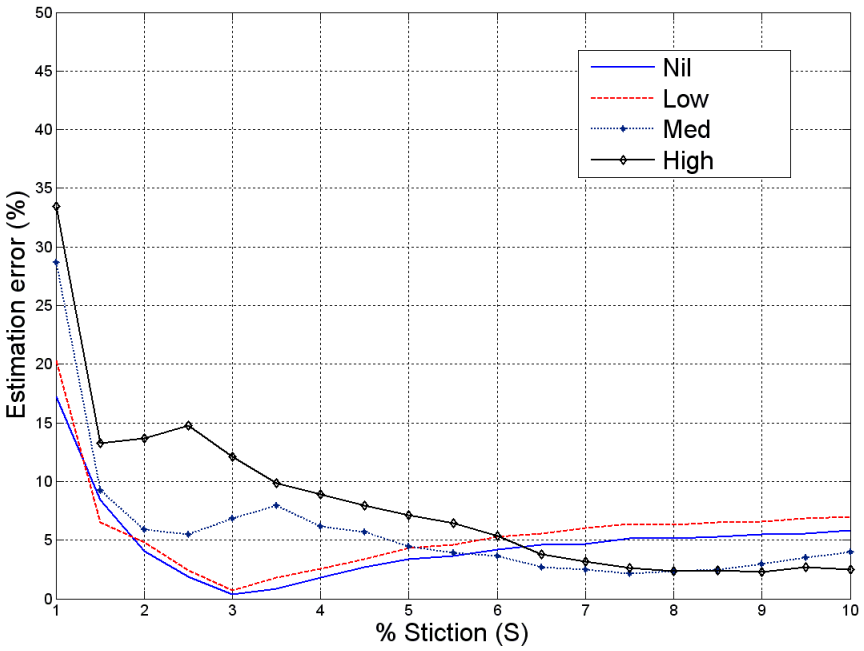


Figure 4.8: Stiction estimation error for an integrating process under different levels of MPM. Model gain (b_k) values were: 1 (No MPM), 0.9 (Low), 0.8 (Medium), 0.7 (High).

oscillatory disturbance enters the closed loop. For example, consider the case of level control in a tank where the outlet valve is the *mv* and the inlet flow rate (unmeasured disturbance) is fluctuating in an oscillatory manner. This would cause the level in the tank to also oscillate and hence the closed loop will have oscillatory *op*, *mv* and *pv*. In this section, a simulation example is used to demonstrate how the proposed algorithm will be able to correctly identify that the valve in a closed loop is normal when an external disturbance is causing persistent oscillations in the loop. If the algorithm had lacked this aspect, there would be false alarms whereby a normal valve will be wrongly reported as a sticky valve.

The same integrating process discussed earlier is considered once again here. In order to introduce an external oscillatory disturbance, a sinusoidal disturbance with amplitude 1 and frequency 0.05 rad/sec along with a random noise disturbance was added in the closed loop simulation. The stiction block was removed from the Simulink model to emulate the case of a normal valve. The controller tuning settings are the same as before. Due to the oscillatory disturbance, the *pv* and *op* variables oscillate. The simulation was run for 5000 sampling intervals. The time trend of the *pv* and *op* signals confirmed that they were oscillating. Using the input-output data and the model of the process, the unknown input estimator was executed and

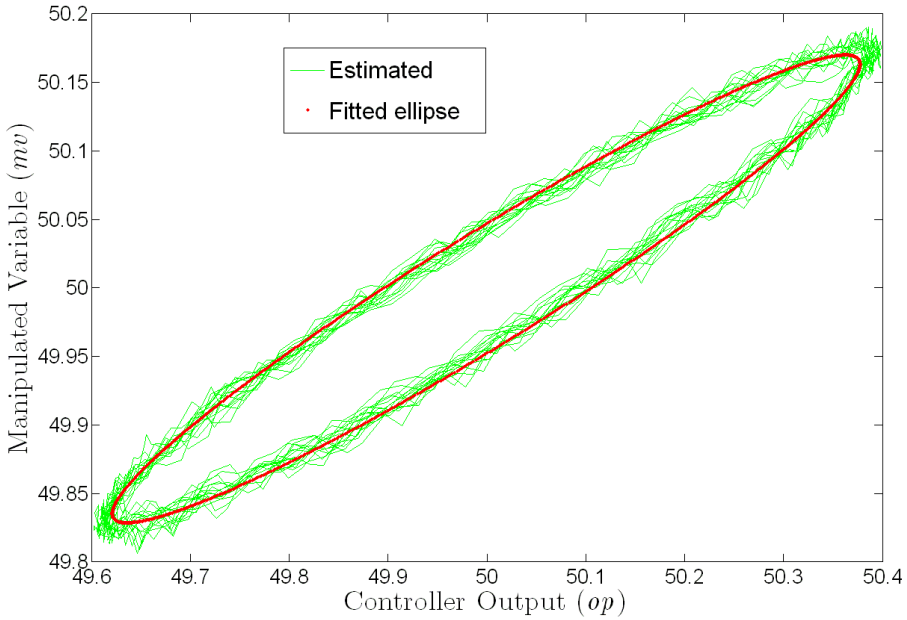


Figure 4.9: Diagnosis of external oscillatory disturbance using the *mv*-*op* plot.

the results were observed. The plot of the estimated *mv* vs. *op* is shown in Figure 4.9. This pattern is clearly an ellipse and is very different from that of a sticky valve. Hence, we can diagnose that the oscillations are not due to stiction, but are caused by some oscillatory external disturbance. We can automatically identify this pattern by fitting an ellipse and checking for the Root Mean Squared Error (RMSE) of fit to the ellipse. The RMSE value can be used as a test statistic to verify a null hypothesis that there is no stiction effect in the loop. Figure 4.9 also shows a fitted ellipse (using the direct least squares method proposed by Fitzgibbon et al. (1996)) and the RMSE of fit of the ellipse was 0.0078 which indicates quantitatively that the pattern is elliptical. Also, note that the width of the ellipse is very small and hence the valve can be considered as perfectly linear i.e., the *mv* and *op* are almost the same.

In the unknown input estimator, the basic assumption is that the unmeasured disturbance is random. If the disturbance is oscillatory, then the sinusoidal part will be unaccounted for by the model in the state estimator. This unaccounted portion of the disturbance signal will be captured in the unknown input signal ($\widehat{\widetilde{u}}$) by the estimator. Since the system is linear, $\widehat{\widetilde{u}}$ will also be sinusoidal. Note that the estimated process input (*mv* axis of Figure 4.9) is given by Eq. (4.12), which is the sum of *op* and $\widehat{\widetilde{u}}$. In fact, the estimated *mv* will not be the actual valve position (which is equal to the *op* for normal valve) but is an effective combination of the *op* and the disturbance signal. Since the *op* signal is also sinusoidal, plotting

Table 4.2: Aggressive controller tuning parameters.

Process	**Controller**: $C(z^{-1}) = K_c(1 + \frac{1}{\tau_I(1-z^{-1})})$	
	K_c	$1/\tau_I$
$\frac{1}{s}$	0.5	0.1
$\frac{1e^{-2s}}{15s+1}$	0.5	0.4

two sinusoid signal variables (which need not be in phase) against each other will yield the elliptical pattern. This explains the reason behind an elliptical pattern appearing in the above situation. Thus the proposed method can clearly detect if an external oscillatory disturbance is the root cause of oscillations in the loop.

4.4.3 Aggressive controller tuning

An aggressively tuned PID controller can be a cause for oscillations in many industrial control loops. In such a scenario, a good valve stiction detection algorithm should confirm that the root cause of the oscillations is not valve stiction. The efficacy of the current methodology will be demonstrated by deliberately introducing aggressive tuning specifications into the simulated control loops considered earlier. The stiction block was removed from the simulation so that the valve can be considered to be a normal one. The controller tuning parameters and the process model transfer functions used to generate oscillatory closed loop data are given in Table 4.2. The steady state time trends for a window of *sp*, *pv* and *op* data from the simulation of the two processes is shown in Figure 4.10.

With a normal linear valve in closed loop, we would expect the *mv*-*op* plot to be a straight line without any stiction pattern being depicted. Figure 4.11 shows the estimated and the true *mv*-*op* plots for the two processes. From the figure, we can observe that the estimated *mv vs. op* line coincides with the actual one and is clearly showing a linear dependence between the two. Thus, the absence of any stiction pattern in the estimated *mv*-*op* plot clearly shows that the valve is normal.

4.5 Pilot Plant Experimental Study

In this section, the application of the proposed methodology to experimental data from a laboratory scale level control system will be presented. This system is located in the Computer Process Control Laboratory in the Department of Chemical and Materials Engineering at the University of Alberta. The installed facility allows the level controller of the system to be configured as a master controller, with a slave

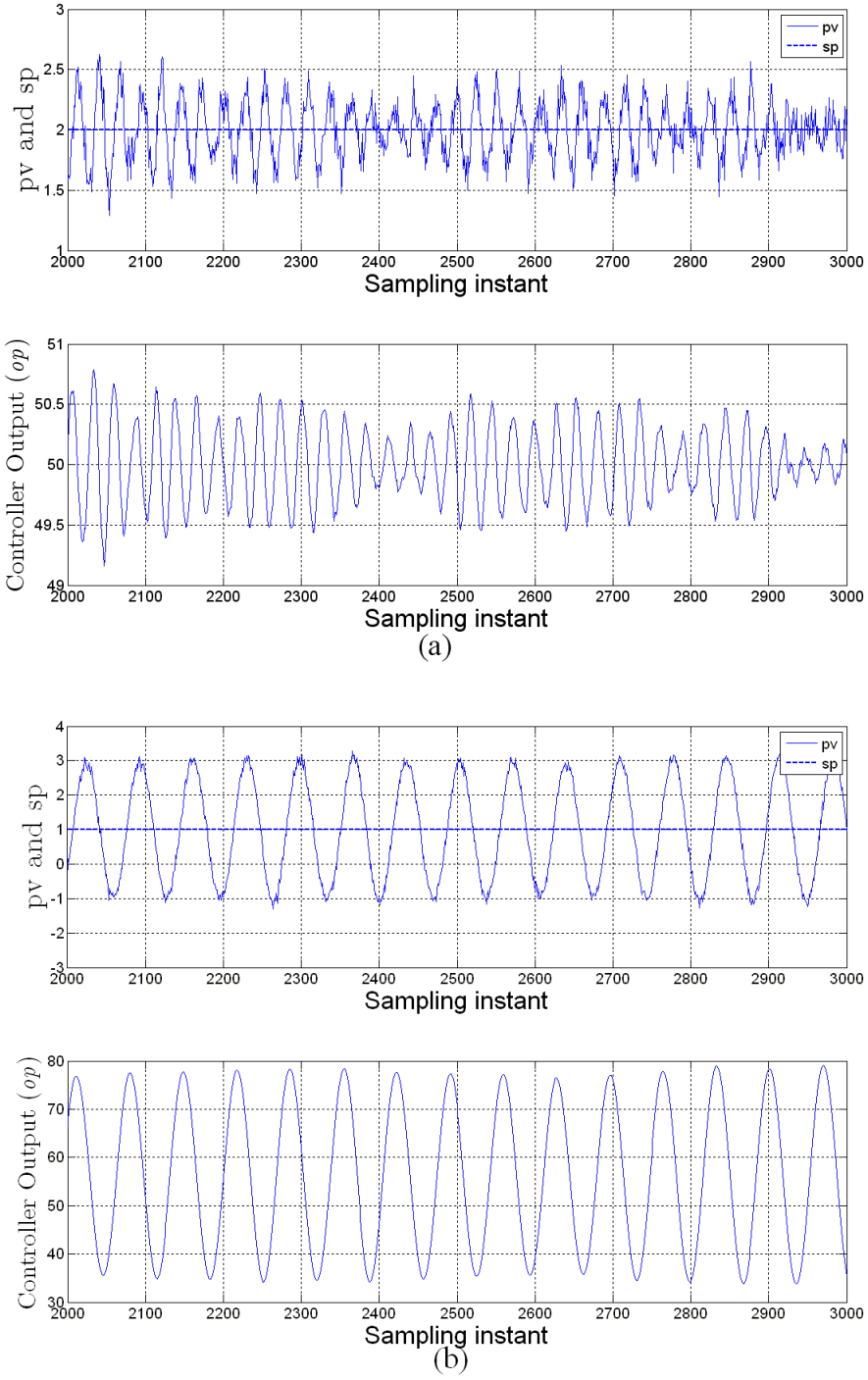


Figure 4.10: Oscillatory control loop simulated with aggressive controller tuning for (a) Integrating Process, (b) FOPDT Process.

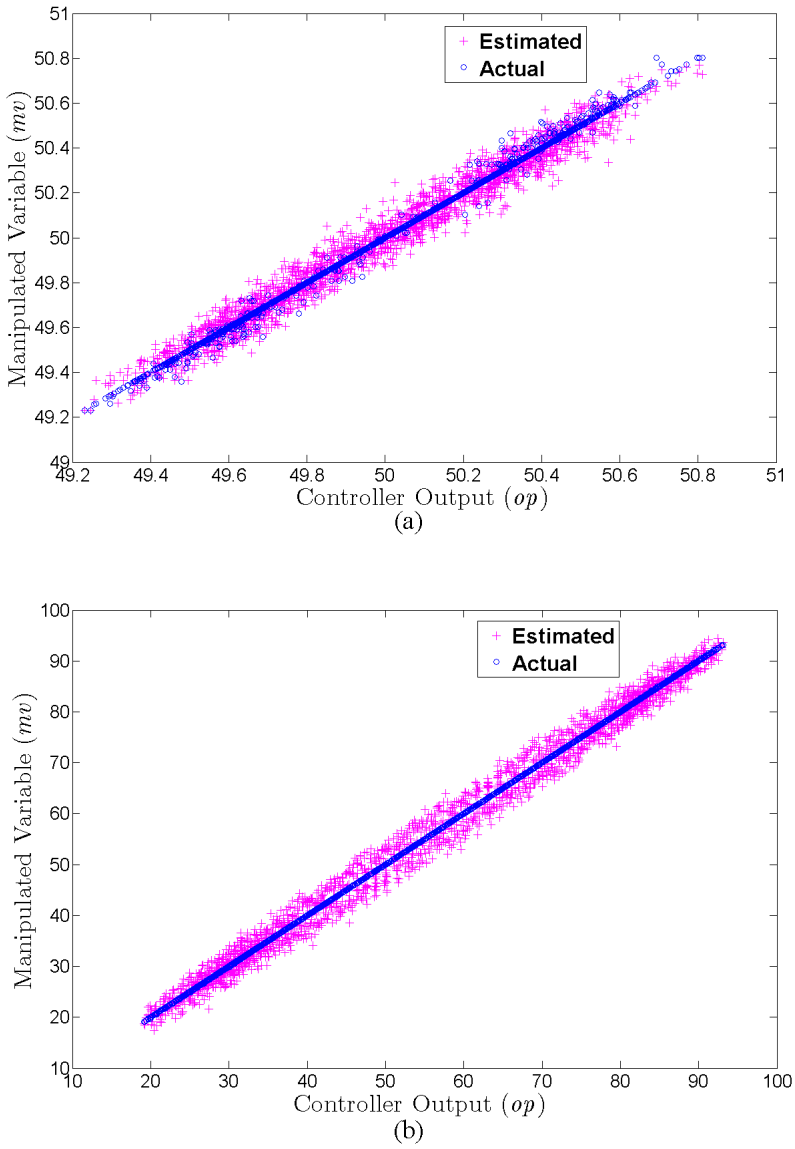


Figure 4.11: Diagnosis of oscillations due to aggressive controller tuning conditions: (a) Integrating Process, (b) FOPDT Process.

flow controller cascaded with the master. Hence both flow and level measurements, which will be part of the flow and level control loops, are available for this process. The flow control loop was considered for this study. The data for this study was obtained from the work by Choudhury et al. (2008). It was first confirmed that the valve was normal by doing bump tests. The two-parameter stiction model was used to transform the controller output signal before it reaches the valve to deliberately emulate valve stiction.

4.5.1 Flow control loop

Figure 4.12 shows the time trend of the *pv* and *op* signals which clearly indicate that the stiction is causing the process variable to oscillate. The parameters of the

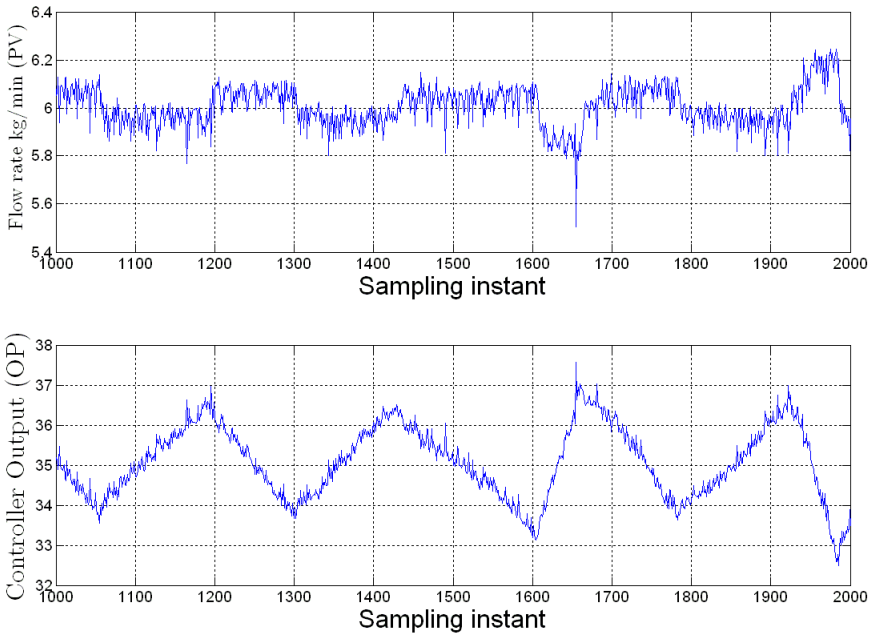


Figure 4.12: Laboratory scale flow control loop time trend data with stiction.

stiction model used for this experimental run was chosen to be (S, J) = (2,1). A first order model with a time constant of 5 secs. and gain of 0.1 was assumed for the flow process. Figure 4.13 shows the actual and the estimated *mv*-*op* plot when the proposed method was applied to the flow control loop. The stiction pattern is clearly visible in the estimated *mv*-*op* plot. The quantification of stiction is also found to be fairly accurate ($\widehat{S} = 2.13$).

4.6 Industrial Case Studies

In this section, the performance of the method on real industrial data sets is shown. The data was taken from the work by Choudhury et al. (2006). First the data obtained from a level control loop in a power plant was considered, in which the condenser level is controlled by manipulating the liquid flow rate out of the condenser. The flow control valve was "sticky" and was causing the loop to oscillate. One of the attractive features about this industrial example was that the valve positioner data (*mv*) was also available from the plant. This was useful in validating the performance of the proposed UIO-based *mv* estimation algorithm.

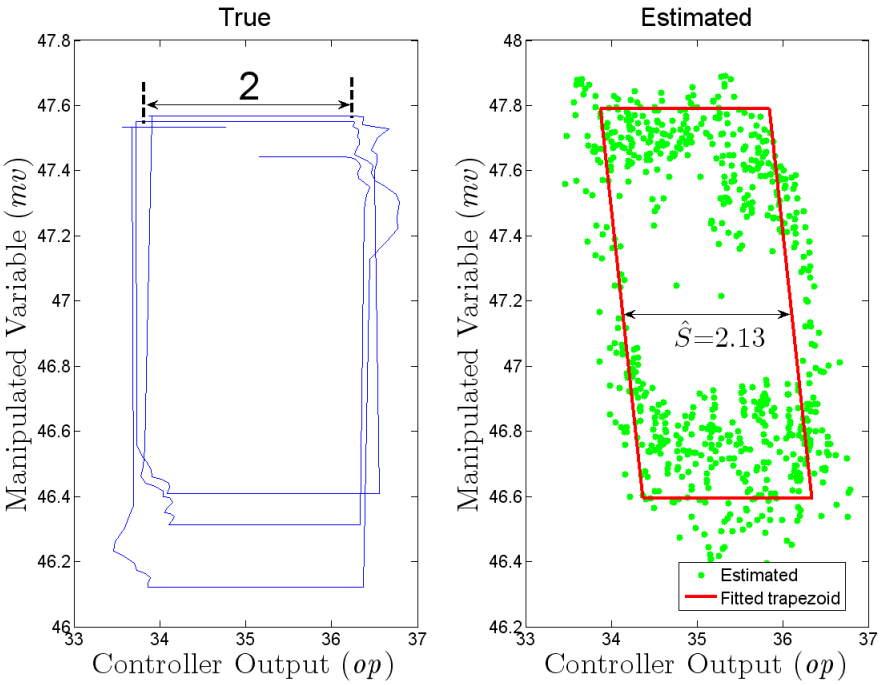


Figure 4.13: Laboratory scale flow control loop: *mv*-*op* plot (True and Estimated).

The full data set consists of 8640 samples for each variable, sampled at 5 sec sampling interval. In order to obtain data from the DCS at the 5 sec sampling rate, an explicit data acquisition loop was setup for investigating stiction. The time trend of the normalized values of all the variables in the data set for a window of the samples is shown in Figure 4.14.

Clearly, from Figure 4.14 we can observe that the loop is oscillating. Also, comparing the time trend of the *mv* and *op* data we can observe that the *mv* motion is restricted at positions where the *op* turns around because of stiction. For modeling this process, a discrete time first order model, with a negative gain to capture the relationship between the *pv* (level) and *mv*(outlet valve position), was used. Note that it is not realistic to use the *mv*, *pv* data to identify a model since the *mv* data is very rarely available. A model with a pole very close to unity was chosen by assigning a value of 0.99 to a_k in the state-space model Eq. (4.1). From Figure 4.14, we can observe that the *pv* deviation is roughly $1/10^{th}$ of the deviation in the *op*. Also, the outlet valve position will have an inverse effect on the level. Thus, by this comparison of the magnitudes of the *pv* and *op* deviation the parameter b_k was fixed at -0.1. Using these model parameters, the valve position was estimated. The true and estimated *mv*-*op* plot is shown in Figure 4.15.

From the true *mv*-*op* plot, we can clearly see that the valve is actually sticking and a similar pattern is present in the estimated *mv*-*op* plot. Also, from the true

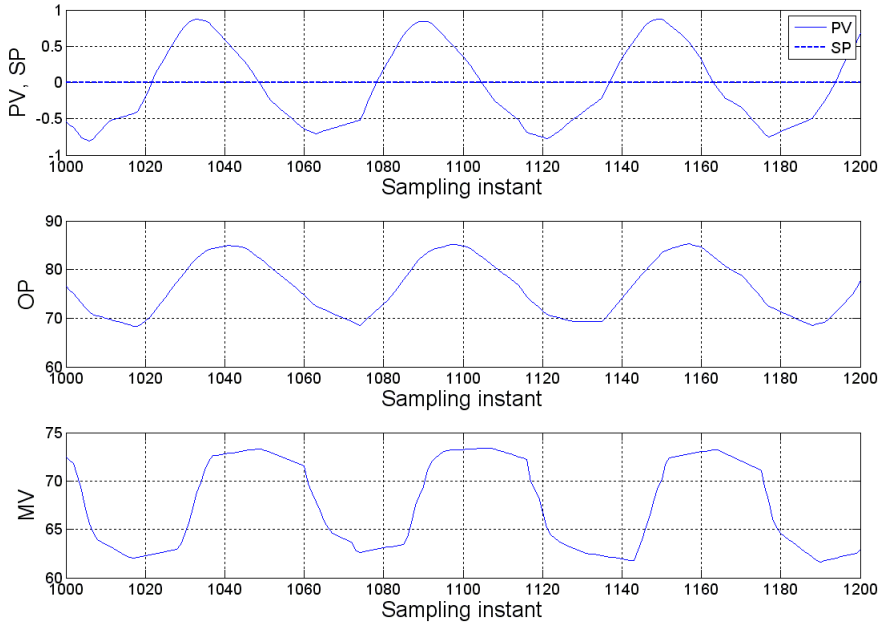


Figure 4.14: Industrial data from a level control loop with stiction.

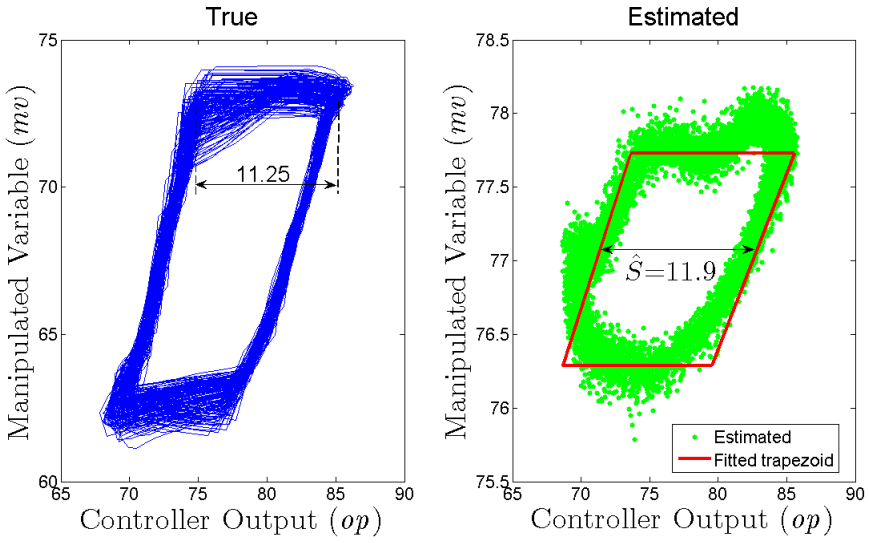


Figure 4.15: Industrial level control loop: *mv-op* plot (True and Estimated).

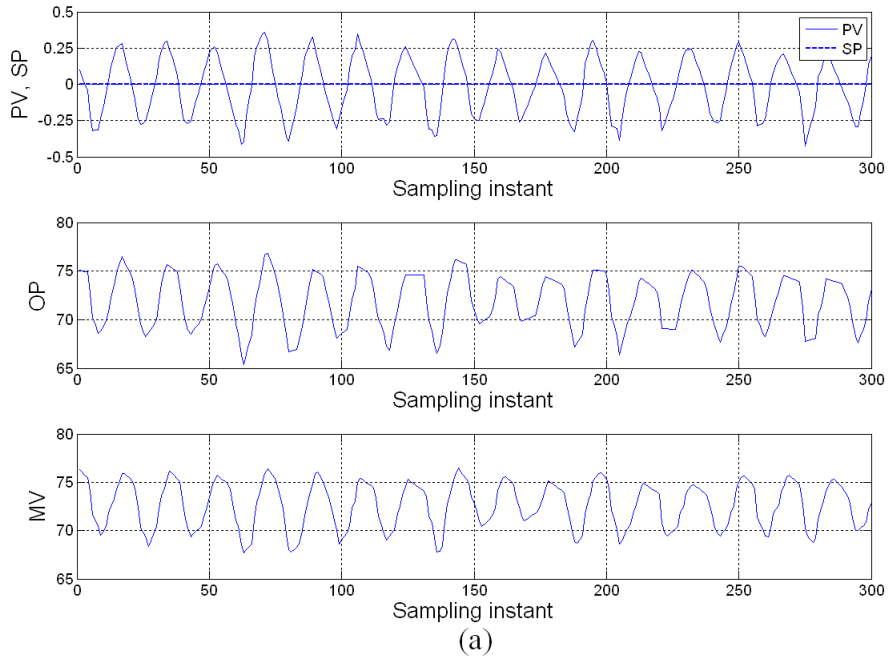
mv-op plot, the actual amount of stiction can be approximately quantified to be around 11.25%. The estimated stiction based on the width of the trapezoid fit is 11.9%, which is very close to the actual stiction amount. Note that there exists a bias between the values of the true and the estimated *mv* (compare the *y*-axis of the true and estimated plot). This can be attributed to the model plant mismatch

which occurs because an accurate model of the level process is not available. The quantification is not affected by the mismatch since the width of the pattern with respect to the *op* axis quantifies the amount of stiction and this '*op*' value is of course always available. Thus, we can conclude based on this industrial case study that the proposed stiction detection and quantification approach is fairly robust and therefore is useful for practical applications.

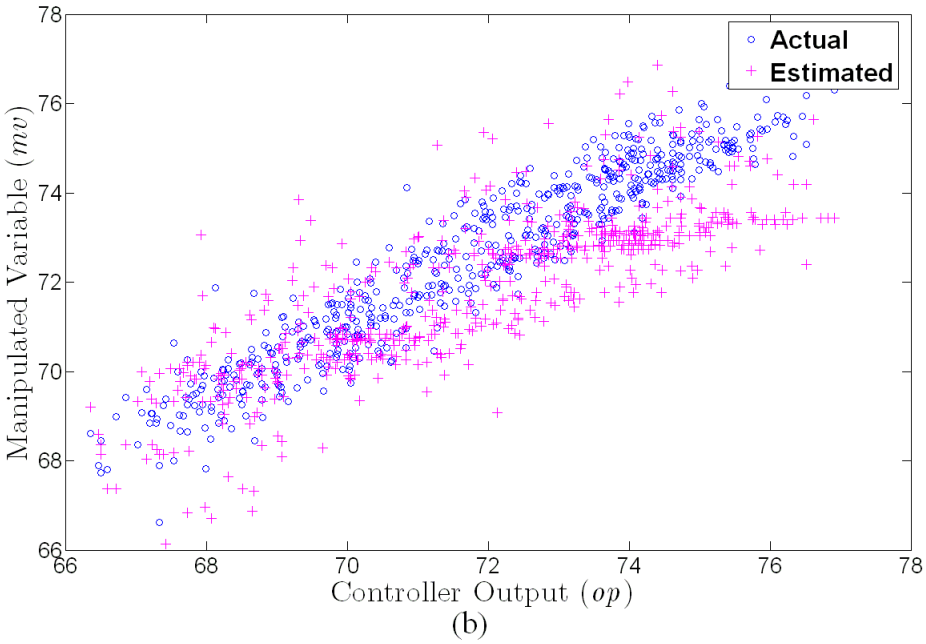
In order to demonstrate the efficacy of the method in preventing false alarms, the data from another loop (from the same power plant considered before) was used. The process variable is level in a condenser which was controlled by manipulating the flow rate of the liquid condensate. Figure 4.16(a) shows the time trend of the data sets. Though the time trend shows that the loop is oscillating, it was found that the valve was normal for this loop. Similar to the earlier case, the valve positioner (*mv*) data was available for this loop, which was useful to confirm that stiction was not present in the valve. Figure 4.16(b) shows the actual *mv*-*op* plot with the estimated values superimposed. A simple integrating process model was used to estimate the *mv* using the proposed methodology. The actual *mv*-*op* plot clearly depicts a normal valve, with the valve position following the controller signal. The estimated *mv*-*op* plot is clearly able to reproduce this fact and yielded a straightforward conclusion that the valve is in good condition. The random fluctuations in the estimated *mv* can be attributed to the random disturbances affecting the process variable.

Remarks. *In the light of industrial applications, there are a few comments we can make regarding the practical side of the proposed methodology:*

- The computational cost of the UIO is very low and is suitable for online implementation.
- In order to automate the technique, the pattern detection method should be very robust. In this work, a simple optimization based method of detecting the stiction pattern was employed. In real applications, the process disturbances can confound this pattern and make it difficult to get satisfactory result with this simple method.
- Nonlinear valve characteristics: A nonlinear flow characteristics of a normal valve is a case that was not analyzed in this work. Such nonlinearity will be reflected in the estimated *mv* vs. *op* plot as a nonlinear curve; this will eliminate the case of stiction. On the other hand, a nonlinear *mv* vs. *op* curve signature can also appear due to a nonlinear output disturbance affecting the process. It would then be difficult to diagnose the root cause of nonlinearity in the absence of stiction.



(a)



(b)

Figure 4.16: Industrial level control loop: Diagnosis of oscillations in the absence of stiction.

4.7 Concluding Remarks

This chapter presented a novel solution to the problem of detection and quantification of valve stiction in process control loops for linear processes. The method is based on the distinct signature that stiction phenomena exhibits in the *mv-op* plot of a sticky valve. In this method, the *mv* (valve position) was estimated by considering it as an unknown input and employing a Kalman filter type unknown input estimator which uses a linear model of the process. The simulation examples showed that the method can accurately detect and quantify stiction by using the estimated *mv-op* plot, even in the presence of significant model plant mismatch. For quantifying stiction, the width of the *mv-op* plot pattern was estimated by fitting a trapezoid in a least squares sense which makes it an automated quantification technique. The simulations also showed that the estimated *mv-op* plot obtained with this method can clearly indicate if the oscillations were caused purely by an external oscillatory disturbance or due to a controller tuning problem. The efficacy of the method was demonstrated on a laboratory scale flow control loop and also on industrial level control loops.

Bibliography

Ahmed, S., Shah, S. L., & Huang, B. (2009). Maintenance issues in oil and gas processes: Detection of valve stiction. In Alfadala, H., Reklaitis, G. V. R., & El-Hawagi, M. M. (Eds.), *Proceedings of the 1st Annual Gas Processing Symposium, Doha, Qatar*, (pp. 202–210).

Chitralekha, S. B., Shah, S. L., & Prakash, J. (2009). Detection and quantification of valve stiction by the method of unknown input estimation. In Quevedo, J., Escobet, T., & Puig, V. (Eds.), *7th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes, Barcelona, Spain*, (pp. 1432–1437).

Chitralekha, S. B., Shah, S. L., & Prakash, J. (2010). Detection and quantification of valve stiction by the method of unknown input estimation. *Journal of Process Control*, *20*(2), 206 – 216.

Choudhury, M. A. A. S., Jain, M., & Shah, S. L. (2008). Stiction - definition, modelling, detection and quantification. *Journal of Process Control*, *18*, 232–243.

Choudhury, M. A. A. S., Shah, S. L., & Thornhill, N. F. (2008). *Diagnosis of Process Nonlinearities and Valve Stiction - Data Driven Approaches*. Advances in Industrial Control. Springer.

Choudhury, M. A. A. S., Shah, S. L., Thornhill, N. F., & Shook, D. S. (2006).

Automatic detection and quantification of stiction in control valves. *Control Engineering Practice*, *14*, 1395–1412.

Choudhury, M. A. A. S., Thornhill, N. F., & Shah, S. L. (2005). Modelling valve stiction. *Control Engineering Practice*, *13*, 641–658.

Desborough, L. & Miller, R. (2001). Increasing customer value of industrial control performance monitoring-honeywell's experience. In *AIChE symposium series 2001*, volume 326, (pp. 172–192).

Fitzgibbon, A., Pilu, M., & Fisher, R. (1996). Direct least squares fitting of ellipses. In *Proceedings of the 13th International Conference on Pattern Recognition*, volume 1, (pp. 253–257).

Gillijns, S. & De Moor, B. (2007). Unbiased minimum-variance input and state estimation for linear discrete-time systems. *Automatica*, *43*, 111–116.

Horch, A. (1999). A simple method for detection of stiction in control valves. *Control Engineering Practice*, *7*, 1221–1231.

Horch, A. (2006). Oscillation diagnosis in control loops: Stiction and other causes. In *Proceedings of the 2006 American Control Conference*, (pp. 2086–2096).

Horch, A. & Isaksson, A. (2001). Detection of stiction in integrating processes. In *European Control Conference, Porto, Portugal*, (pp. 1327–1332).

Jelali, M. (2008). Estimation of valve stiction in control loops using separable least squares and global search algorithms. *Journal of Process Control*, *18*, 632–642.

Jelali, M. & Huang, B. (Eds.). (2009). *Detection and Diagnosis of Stiction in Control Loops - State of the Art and Advanced Methods*. Springer.

Jelali, M. & Scali, C. (2009). Comparative study of valve stiction detection methods. In M. Jelali & B. Huang (Eds.), *Detection and Diagnosis of Stiction in Control Loops* chapter 13. Springer.

Kayihan, A. & Doyle III, F. J. (2000). Friction compensation for a process control valve. *Control Engineering Practice*, *8*(7), 799–812.

Lee, K. H., Ren, Z., & Huang, B. (2009). Stiction estimation using constrained optimisation and contour map. In M. Jelali & B. Huang (Eds.), *Detection and Diagnosis of Stiction in Control Loops* chapter 11, (pp. 225–256). Springer.

Peter He, Q., Wang, J., Pottmann, M., & Joe Qin, S. (2007). A curve fitting method for detecting valve stiction in oscillating control loops. *Industrial and Engineering Chemistry Research*, *46*, 4549–4560.

Rengaswamy, R., Hagguland, T., & Venkatasubramaniam, V. (2001). A qualitative shape analysis formalism for monitoring control loop performance. *Engineering Applications of Artificial Intelligence*, *14*, 23–33.

Ruel, M. (2000). Stiction: The hidden menace. how to recognize this most difficult case of loop cycling. *Control Magazine*, *November*, 69–75.

Scali, C. & Ghelardoni, C. (2008). An improved qualitative shape anlaysis technique for automatic detection of valve stiction in flow control loops. *Control Engineering Practice*, *16*, 1501–1508.

Srinivasan, R., Rengaswamy, R., Narasimhan, S., & Miller, R. (2005). Control Loop Performance Assessment. 2. Hammerstein Model Approach for Stiction Diagnosis. *Industrial & Engineering Chemistry Research*, *44*(17), 6719–6728.

Stenman, A., Gustafsson, F., & Forsman, K. (2003). A segmentation-based method for detection of stiction in control valves. *International Journal of Adaptive Control and Signal Processing*, *17*, 625–634.

Yamashita, Y. (2006). An automatic method for detection of valve stiction in process control loops. *Control Engineering Practice*, *14*, 503–510.

Chapter 5

Simultaneous State and Parameter Estimation Schemes for a Continuous Fermentor Reactor[1,2]

5.1 Introduction

Stochastic nonlinear state space models are commonly used for modeling dynamic processes in many diverse fields. Some of the common examples are: the phenomenological models used to study small scale chemical and biological processes such as bio-reactors; stochastic volatility models used to model the dynamic changes in stock prices in the area of quantitative finance (Kim, 2005); reservoir simulation models used to describe the flow of oil and gas in a petroleum reservoir (Evensen, 2007; Lorentzen et al., 2005). In a majority of the cases, the model structure is based on fundamental laws of mass and energy conservation. While these conservation laws provide the deterministic part of a dynamic model, it is also necessary to account for unknown disturbances and measurement errors through a stochastic description of the model.

The following generic discrete time nonlinear state space model is considered:

$$\begin{aligned} x_{k+1} &= f(x_k, u_k, \theta) + v_k \\ y_k &= h(x_k, \theta) + n_k \end{aligned} \tag{5.1}$$

where, v_k denotes the process noise to take into account unknown process disturbances and n_k takes into account the noise in the measurement devices. The state vector, $x_k \in \mathbb{R}^l$, is generally considered to be 'hidden' and its effect is observed

[1] An almost full version of this chapter has been published in the Journal of Process Control (Chitralekha et al., 2010).

[2] Abbreviated sections of this chapter were also presented at the 15th IFAC Symposium on System Identification, Saint-Malo, France, July 6 – 8, 2009 (Chitralekha et al., 2009).

through the measurements, $y_k \in \mathbb{R}^m$. The measurements are related to the true states through the measurement function $h(x_k, \boldsymbol{\theta})$. For convenience, the process disturbance (v_k) and measurement noise (n_k) are assumed to follow zero-mean Gaussian distributions given by $\mathcal{N}(0, \mathbf{Q}_v)$ and $\mathcal{N}(0, \mathbf{R})$ respectively. However, this assumption is not necessary for the algorithm developed in this work. It is also assumed that the process disturbance and measurement noise are uncorrelated with each other, i.e., $E[v_k n_k^T] = 0$. This assumption is necessary for the standard state estimation algorithms discussed in this work. However, alternative state estimation algorithms which take into account cross-correlated noise variables can be incorporated into the proposed methodologies. The functions f and h are assumed to be known, nonlinear functions, and are parameterized by a constant but unknown vector θ.

If all the parameters of the model are known *a priori*, then the problem of state estimation can be directly solved using any of the standard algorithms such as the particle filter (Gordon et al., 1993; Arulampalam et al., 2002), unscented Kalman filter (Julier & Uhlmann, 1997), extended Kalman filter or their variants. The accuracy of state estimation will directly depend on the model quality, i.e., on the model's ability to accurately capture the process dynamics. The parameters in the model play a crucial role in determining the model quality. Hence, a good estimate of the parameters is required *a priori* for state estimation. Usually, in most physical systems there are at least a few parameters whose values are unknown *a priori*. Thus, for a chosen model structure the problem of system identification is to identify the unknown parameters of the model and simultaneously estimate the hidden states, using input-output data from the process. In this work, a Continuous Fermentor reactor is considered in which some of the yield parameters in the dynamic model are unknown. The substrate concentration is assumed to be unmeasured or hidden. Thus, the problem is to estimate the unknown parameters simultaneously with the states, using measured data from the process.

There are several techniques available in the literature that can be employed to solve this problem. Based on the approach taken for data processing, they can be broadly classified into batch and recursive techniques. Kantas et al. (2009) have presented a comprehensive review of the state of the art on joint state and parameter estimation techniques based on particle filters; a comparison of the pros and cons of the techniques is also presented. Several authors have proposed the technique of Expectation Maximization algorithm combined with the particle filter for combined state and parameter estimation of nonlinear systems (Andrieu & Doucet, 2003; Wills et al., 2008; Gopaluni, 2008b, 2010). Doucet & Tadic (2003), and Poyiadjis et al. (2005) have presented gradient based maximum likelihood estimation techniques, where a particle filter is used to numerically approximate the likelihood function derivatives. Chen et al. (2005) have presented the application of particle filter

for recursive state and parameter estimation of a simulated batch polymerization reactor. Recently, Evensen (2009) has proposed the Ensemble Kalman filter as an efficient Monte Carlo technique for recursive state and parameter estimation in large scale models used in atmospheric sciences and oil reservoir simulations. In general, most of these techniques are based on the maximum likelihood estimation (MLE) framework. The maximum likelihood framework is an attractive approach for state and parameter estimation due to the asymptotic consistency and efficiency of the resulting estimates. There are several ways of defining the likelihood function in order to solve the problem of combined state and parameter estimation (Maybeck, 1972). In this work, two types of likelihood functions is considered, one of them suitable for batch estimation and the other for online recursive estimation.

The following are the main contributions of this chapter: (a) Batch and recursive schemes for simultaneous state and parameter estimation, which are based on the theory of Expectation Maximization algorithm and nonlinear filters, is derived. (b) Various versions of the respective schemes, which employ state-of-the-art nonlinear filters such as particle filter, unscented Kalman filter (UKF), extended Kalman filter (EKF) and ensemble Kalman filter (EnKF), is presented. (c) An empirical comparison of batch parameter estimation efficacy and the associated computational overload is done on a simulated Continuous Fermentor reactor. (d) The efficacy of the recursive scheme for performing online estimation of time-varying parameters is demonstrated on the same reactor example.

This chapter is organized as follows. In Section 5.2, a batch parameter estimation technique based on the Expectation Maximization (EM) principle is explained. Three popular choices of nonlinear state estimation, which is an important step in the EM algorithm, are considered: particle smoother, unscented Kalman smoother and extended Kalman smoother. The effect of the choice of these smoothing algorithms on the overall computational cost of the EM algorithm is also analyzed. In Section 5.3, a recursive parameter estimation scheme is explained in which the parameters can be recursively estimated in a state estimation framework. For recursive estimation, the application of the ensemble Kalman filter and unscented Kalman filter is considered. Section 5.4 presents a case study using a Continuous Fermentor reactor; an empirical comparison of different methodologies is presented followed by some concluding remarks in Section 5.5.

5.2 Batch Parameter Estimation

Assume that a batch of data $\{\mathbf{Y}_T, \mathbf{U}_T\}$ where, $\mathbf{Y}_T = \{y_1, y_2, ...y_T\}$ and $\mathbf{U}_T = \{u_1, u_2, ...u_T\}$ are collected from a process whose dynamics are defined by Eq. (5.1). One common approach to estimating the parameters in Eq. (5.1) is to maximize the likelihood function of the measurements conditioned on the unknown parameters.

This parameter estimate can be written as

$$\widehat{\theta} = \arg\max_{\theta} \; p(\mathbf{Y}_T|\mathbf{U}_T, \theta) \tag{5.2}$$

where, $p(\mathbf{Y}_T|\mathbf{U}_T, \theta)$ represents the joint likelihood of the measurements. It is assumed that $\mathbf{U}_T$ are measured and known without any error. Hence, the dependence of the above likelihood on them will not be shown explicitly in the rest of this chapter. The above maximization problem requires either a functional form of the likelihood function or an approach to evaluate it for any θ. However, it is well-known that developing such a functional form is difficult for the general class of state space models described by Eq. (5.1). Moreover, this classical likelihood function is not suitable for the joint state and parameter estimation problem considered in this work.

On the other hand, it is straightforward to derive a functional form of the 'complete' likelihood function, $p(\mathbf{X}_T, \mathbf{Y}_T|\theta)$, that includes unmeasured and measured states. This complete likelihood function is related to the classical likelihood function through Bayes' rule as

$$p(\mathbf{X}_T, \mathbf{Y}_T|\theta) = p(\mathbf{X}_T|\mathbf{Y}_T, \theta)\; p(\mathbf{Y}_T|\theta) \tag{5.3}$$

where, $\mathbf{X}_T = \{x_1, x_2, ...x_T\}$. Based on Eq. (5.3), it is clear that the complete likelihood function inherently includes the information contained in the classical likelihood function. Then, a functional form of the complete likelihood can be derived using Bayes' rule and the Markov property of the model as follows

$$\begin{aligned} p(\mathbf{X}_T, \mathbf{Y}_T|\theta) &= p(\mathbf{X}_T|\theta)p(\mathbf{Y}_T|\mathbf{X}_T, \theta) \\ &= p(x_1|\theta)\prod_{k=2}^{T} p(x_k|x_{k-1}, \theta)\prod_{k=1}^{T} p(y_k|x_k, \theta) \end{aligned} \tag{5.4}$$

It turns out that by iteratively maximizing the expected value of the complete likelihood with respect to the parameter vector, it is possible to maximize the measurement likelihood function in Eq. (5.2). This iterative approach is called Expectation Maximization algorithm or the *EM algorithm* and was developed by Dempster et al. (1977). It is essentially an elegant way to maximize the likelihood function in problems where the data is incomplete or *missing* as is the case with state-space models. If we consider the state variables $\mathbf{X}_T$ as *missing data,* then the EM algorithm can be exploited for the simultaneous state and parameter estimation problem as described below.

5.2.1 Expectation Maximization algorithm

In this section, the EM algorithm is explained which will be used for the parameter estimation problem defined above. The basic idea of the algorithm is to iteratively

compute the parameter estimates by taking a two-step procedure at each iteration: the first step is called the Expectation step and is followed by a second Maximization step. Before elaborating on the two basic steps, the rationale behind this algorithm is explained.

Based on Eq. (5.3), the complete log-likelihood function and the classical log-likelihood function can be related as

$$\log\ p(\mathbf{Y}_T|\theta) = \log p(\mathbf{X}_T, \mathbf{Y}_T|\theta) - \log p(\mathbf{X}_T|\mathbf{Y}_T, \theta) \tag{5.5}$$

Consider a probability distribution function $p(\mathbf{X}_T|\mathbf{Y}_T, \theta')$ defined over the state variables, where θ' is any fixed value of the parameter vector. Taking the expectation of the terms in Eq. (5.5) w.r.t this distribution function (Note: The L.H.S. of Eq. (5.5) will be unaffected by this operation since $\int \log p(\mathbf{Y}_T|\theta)p(\mathbf{X}_T|\mathbf{Y}_T,\theta')d\mathbf{X}_T = \log p(\mathbf{Y}_T|\theta)$), we obtain

$$\begin{aligned}\log p(\mathbf{Y}_T|\theta) &= \int \log p(\mathbf{X}_T, \mathbf{Y}_T|\theta)p(\mathbf{X}_T|\mathbf{Y}_T,\theta')d\mathbf{X}_T \\ &\quad - \int \log p(\mathbf{X}_T|\mathbf{Y}_T,\theta)p(\mathbf{X}_T|\mathbf{Y}_T,\theta')d\mathbf{X}_T \\ &\triangleq Q(\theta, \theta') - H(\theta, \theta')\end{aligned} \tag{5.6}$$

The first term $Q(\theta, \theta')$, which is the expected value of the complete log-likelihood function, is called the Q-function. Since we are interested in finding θ in an iterative fashion, let us replace θ' by θ_i where i denotes the iteration count. Then, we have

$$\log p(\mathbf{Y}_T|\theta) = Q(\theta, \theta_i) - H(\theta, \theta_i) \tag{5.7}$$

Also, we can derive the following inequality for $\log p(\mathbf{Y}_T|\theta)$:

$$\begin{aligned}\log p(\mathbf{Y}_T|\theta) &= \log \left[\int p(\mathbf{X}_T, \mathbf{Y}_T|\theta)d\mathbf{X}_T\right] \\ &= \log \left[\int \frac{p(\mathbf{X}_T, \mathbf{Y}_T|\theta)}{p(\mathbf{X}_T|\mathbf{Y}_T,\theta_i)}p(\mathbf{X}_T|\mathbf{Y}_T,\theta_i)d\mathbf{X}_T\right] \\ &\geq \int \log \left[\frac{p(\mathbf{X}_T, \mathbf{Y}_T|\theta)}{p(\mathbf{X}_T|\mathbf{Y}_T,\theta_i)}\right] p(\mathbf{X}_T|\mathbf{Y}_T,\theta_i)d\mathbf{X}_T\end{aligned} \tag{5.8}$$

where, the last inequality follows from Jensen's inequality (Boyd & Vandenberghe, 2006) applied to the concave function $\log\left[\frac{p(\mathbf{X}_T,\mathbf{Y}_T|\theta)}{p(\mathbf{X}_T|\mathbf{Y}_T,\theta_i)}\right]$. Expanding the R.H.S of inequality (5.8), we obtain

$$\begin{aligned}\int \log \left[\frac{p(\mathbf{X}_T, \mathbf{Y}_T|\theta)}{p(\mathbf{X}_T|\mathbf{Y}_T,\theta_i)}\right] p(\mathbf{X}_T|\mathbf{Y}_T,\theta_i)d\mathbf{X}_T &= \int \log p(\mathbf{X}_T, \mathbf{Y}_T|\theta)p(\mathbf{X}_T|\mathbf{Y}_T,\theta_i)d\mathbf{X}_T \\ &\quad - \int \log p(\mathbf{X}_T|\mathbf{Y}_T,\theta_i)p(\mathbf{X}_T|\mathbf{Y}_T,\theta_i)d\mathbf{X}_T \\ &= Q(\theta, \theta_i) - H(\theta_i, \theta_i)\end{aligned} \tag{5.9}$$

From Eqs. (5.8) and (5.9), we have

$$\log p(\mathbf{Y}_T|\theta) \geq Q(\theta, \theta_i) - H(\theta_i, \theta_i) \tag{5.10}$$

Let θ_i be an estimate of the parameter vector. Consider a new parameter vector θ_{i+1} defined as

$$\theta_{i+1} = \arg\max_{\theta} \; Q(\theta, \theta_i) \tag{5.11}$$

Also, substituting for $\theta = \theta_{i+1}$ in Eq. (5.10), we have

$$\log p(\mathbf{Y}_T|\theta_{i+1}) \geq Q(\theta_{i+1}, \theta_i) - H(\theta_i, \theta_i) \tag{5.12}$$

From the definition of θ_{i+1}, we have

$$\begin{aligned} Q(\theta_{i+1}, \theta_i) &\geq Q(\theta_i, \theta_i) \\ \text{or} & \\ Q(\theta_{i+1}, \theta_i) - H(\theta_i, \theta_i) &\geq Q(\theta_i, \theta_i) - H(\theta_i, \theta_i) \end{aligned} \tag{5.13}$$

From Eq. (5.7) and (5.13), it follows that $Q(\theta_{i+1}, \theta_i) - H(\theta_i, \theta_i) \geq \log p(\mathbf{Y}_T|\theta_i)$. Based on this inequality and Eq. (5.12), we have

$$\log p(\mathbf{Y}_T|\theta_{i+1}) \geq \log p(\mathbf{Y}_T|\theta_i) \tag{5.14}$$

Therefore, by maximizing the Q-function w.r.t the parameter vector (Eq. 5.11), it is possible to obtain a new parameter estimate such that the likelihood function increases.

The distribution $p(\mathbf{X}_T|\mathbf{Y}_T,\theta_i)$ is the conditional distribution of the states conditioned on all the measurements available in the batch of data. It is also known as the distribution of smoothed states. Finding the expected value of the complete log-likelihood function with respect to this smoothed state distribution function constitutes the first step of the EM algorithm and is called the Expectation step or E-step. Using this smoothed state distribution function, we can evaluate the Q-function at any θ. Thus, using $p(\mathbf{X}_T|\mathbf{Y}_T,\theta_i)$ we can maximize $Q(\theta, \theta_i)$ w.r.t θ to obtain a better parameter estimate θ_{i+1} as was shown earlier. This step constitutes the Maximization step or M-step which is the second step of a given iteration in the EM algorithm. For the subsequent iterations, we replace θ_i by θ_{i+1} and repeat the E-step and M-step until the parameter converges to a local maximum of $\log p(\mathbf{Y}_T|\theta)$.

Algorithm: Based on the above theoretical derivations, the iterations involved in the EM algorithm can be summarized into the following steps. First, we choose an initial guess θ_0 for the parameters and proceed as follows:

1. Set $i = 0$.
2. E-step: Find the smoothed distribution $p(\mathbf{X}_T|\mathbf{Y}_T,\theta_i)$ by estimating the states $\mathbf{X}_T$ using a smoother algorithm with the parameter values in θ_i.
3. M-step: Maximize the objective function $Q(\theta, \theta_i)$ w.r.t θ to obtain

$$\widetilde{\theta} = \arg\max_{\theta} \; Q(\theta, \theta_i)$$

4. Set $i = i + 1$, $\theta_i = \widetilde{\theta}$
5. Repeat steps 2 to 4 until the change in the parameter vector between two subsequent iterations is below a specified tolerance level.

Figure 5.1 summarizes the above basic steps involved in the EM algorithm for parameter estimation. The idea behind this algorithm can be intuitively explained as follows: Using an initial guess of the parameter vector, the smoothed estimates of unmeasured states and missing observations are obtained. The Q-function is then estimated using these smoothed states and maximized with respect to the parameter vector. In the subsequent steps, the above expectation and maximization steps are repeated with the latest iterate of the parameter vector. This procedure is continued until the change in parameters through these iterations remains within a user specified tolerance limit. The maximization step in the EM algorithm can be performed numerically using any standard nonlinear optimization routine. Constraints can be imposed on the parameters, if required, to prevent the optimization from generating unrealistic parameter estimates.

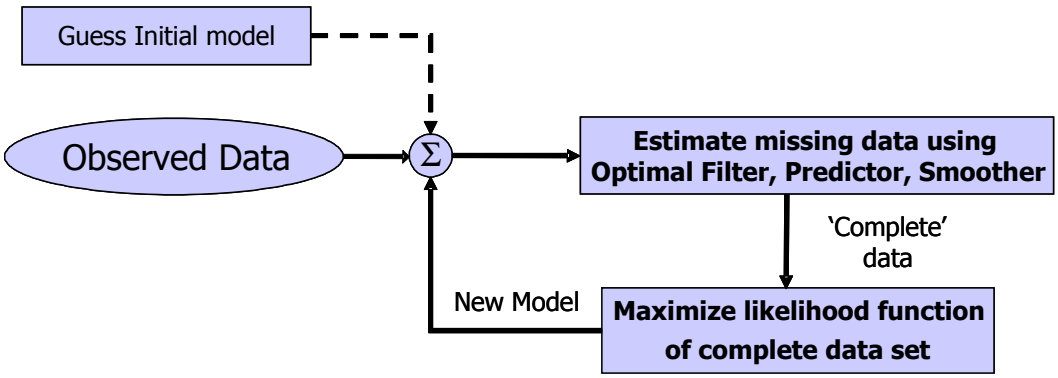


Figure 5.1: Flowchart of EM based parameter estimation scheme.

From the basic steps involved in the EM algorithm, it is clear that evaluation of the Q-function is required for maximizing it. However, in general, the complex multidimensional integrals in Q-function cannot be evaluated analytically for non-

linear state-space models of the form defined in Eq. (5.1). Hence, we have to resort to the elegant technique of using Monte-Carlo samples to approximate the integral equation in the Q-function. In its simplest form, this technique can be seen as a numerical approximation of integrals of the following form,

$$E[f(x)] = \int f(x)p(x)dx \approx \frac{1}{N}\sum_{i=1}^{N} f(x^i) \tag{5.15}$$

In Eq. (5.15), $f(x)$ is some function of the random variable x, and x^is are N independent samples from the probability distribution $p(x)$ of the random variable x. Noting that the integrals in Q-function are of the same form as the one in Eq. (5.15), one can use Monte-Carlo samples to approximate it. One straightforward way to approximate Q-function would be to use samples from the joint density $p(\mathbf{X}_T|\mathbf{Y}_T, \theta')$. But this method is inefficient as it requires sampling from a high dimensional density resulting in poor approximation when T is large (Andrieu et al., 2004). In order to overcome this issue, Gopaluni (2008b) has derived a simplified form of the Q-function involving only point-wise state densities given by

$$\begin{aligned} Q(\theta, \theta_i) \;=\; & \int \log[p(x_1|\theta)]p(x_1|\mathbf{Y}_T,\theta_i)dx_1 \\ & + \sum_{k=2}^{T} \int \log[p(x_k|x_{k-1},\theta)]p(x_k, x_{k-1}|\mathbf{Y}_T,\theta_i)dx_{k:k-1} \\ & + \sum_{k=1}^{T} \int \log[p(y_k|x_k,\theta)]p(x_k|\mathbf{Y}_T,\theta_i)dx_k \end{aligned} \tag{5.16}$$

From equations (5.15) and (5.16), it is clear that we can approximate the Q-function by using i.i.d samples from the distributions $p(x_k|\mathbf{Y}_T,\theta_i)$ and $p(x_k, x_{k-1}|\mathbf{Y}_T,\theta_i)$. These density functions are the smoothed density functions of the states whose approximations can be obtained using different types of nonlinear smoothing algorithms.

5.2.2 Forward-backward smoother algorithms

In this work, the forward-backward (Rauch–Tung–Striebel smoother) type implementation of smoothing algorithms was considered. The common feature of all such smoothing algorithms is that they are composed of a forward filter and a backward smoothing pass. From the Bayesian perspective, the filtering and smoothing steps in such smoothing algorithms is given by the following equations (see Gopaluni (2008a)):

1. Forward Filtering equation (for $k = 1$ to T, initialized with $p(x_0) =$ prior distribution of initial state)

   (a) Prediction step:

$$p(x_k|\mathbf{Y}_{k-1}, \theta_i) = \int p(x_k|x_{k-1}, \theta_i) p(x_{k-1}|\mathbf{Y}_{k-1}) dx_{k-1} \tag{5.17}$$

   (b) Update step:

$$p(x_k|\mathbf{Y}_k, \theta_i) = \frac{p(y_k|x_k) p(x_k|\mathbf{Y}_{k-1}, \theta_i)}{p(y_k|\mathbf{Y}_{k-1})} \tag{5.18}$$

2. Backward Smoothing equations (for $k = T - 1$ to 1, initialized with $p(x_T|\mathbf{Y}_T, \theta_i)$ from Forward Filtering)

   (a) Joint smoothed density of x_k and x_{k+1}:

$$p(x_k, x_{k+1}|\mathbf{Y}_T, \theta_i) = p(x_k|\mathbf{Y}_k, \theta_i) \times \frac{p(x_{k+1}|x_k, \theta_i) p(x_{k+1}|\mathbf{Y}_T, \theta_i)}{p(x_{k+1}|\mathbf{Y}_k, \theta_i)} \tag{5.19}$$

   (b) Smoothed density of x_k:

$$p(x_k|\mathbf{Y}_T, \theta_i) = \int p(x_k, x_{k+1}|\mathbf{Y}_T, \theta_i) dx_{k+1}$$

$$= p(x_k|\mathbf{Y}_k, \theta_i) \int \frac{p(x_{k+1}|x_k, \theta_i) p(x_{k+1}|\mathbf{Y}_T, \theta_i)}{p(x_{k+1}|\mathbf{Y}_k, \theta_i)} dx_{k+1} \tag{5.20}$$

The above filtering and smoothing equations are usually intractable for nonlinear systems and hence numerical approximations are required. In the current work, the performance of the EM algorithm is compared when three types of such approximate smoothing algorithms are employed: the particle smoother, the unscented Kalman smoother and the extended Kalman smoother.

Particle smoother:

Particle smoothers are based on the particle filter algorithm. The particle filter, as proposed by Gordon et al. (1993), is one of the most versatile filtering algorithms which uses Monte Carlo simulation and sampling methods. Though computationally expensive, this algorithm can handle any type of system nonlinearity and noise distribution. The main idea is to approximate all the required distributions using independent samples and perform the 'Bayesian steps' of filtering. In particle filter terminology these samples are referred to as 'particles'. This section explains how particles can be employed to arrive at an approximate solution to the Bayesian

filtering problem. Assume that at $(k-1)$, we have performed filtering and obtained particles $\{\mathbf{x}^i_{k-1|k-1}\}_{i=1}^N$, which approximate the posterior distribution $p(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1},\theta)$. Using each of these particles as initial conditions and the known inputs, $\mathbf{u}_{k-1}$, we can do a one step forward simulation of the process with a stochastic state space model of the process. The new set of particles that we obtain, represented as $\{\mathbf{x}^i_{k|k-1}\}_{i=1}^N$, will represent i.i.d samples from the prior distribution density function, $p(\mathbf{x}_k|\mathbf{Y}_{k-1},\theta)$. The above mentioned one-step simulation constitutes the *prediction step* of the particle filter. In order to obtain an estimate of the state at current time k, we require samples from the posterior distribution, $p(\mathbf{x}_k|\mathbf{Y}_k,\theta)$, which incorporates the knowledge of $\mathbf{y}_k$. Note that we do not have a direct way of drawing samples from the posterior distribution since the density function $p(\mathbf{x}_k|\mathbf{Y}_k,\theta)$ is not available. By the recurrence relationship in Eq. (5.18), we have (ignoring the normalizing constant in the denominator and using proportionality)

$$p(\mathbf{x}_k|\mathbf{Y}_k,\theta) \propto p(\mathbf{y}_k|\mathbf{x}_k,\theta)p(\mathbf{x}_k|\mathbf{Y}_{k-1},\theta) \tag{5.21}$$

The likelihood, $p(\mathbf{y}_k|\mathbf{x}_k,\theta)$, can be evaluated for any $\mathbf{x}_k$ using the density function for the measurement noise and the measurement equation. Using the samples from $p(\mathbf{x}_k|\mathbf{Y}_{k-1},\theta)$ and the likelihood function values evaluated at these sample locations, one can obtain samples representing the posterior $p(\mathbf{x}_k|\mathbf{Y}_k,\theta)$. This is done by a method called *sampling importance resampling* (Rubin, 1988) and constitutes the update step in the particle filter. This technique makes it possible to recursively propagate the particles based on the relationship in Eq. (5.21). The method is useful when a good approximation of the posterior distribution exists from which it is easy to obtain samples. This distribution is called the *importance distribution*. In the current context, it is assumed that the prior distribution, from which we have samples available, is a good approximation of the posterior. This assumption results in a particular type of particle filter which is widely known as *SIR-particle filter.* For the SIR-particle filter, the importance ratio $r(\mathbf{x})$ of any sample $\mathbf{x}$ can be defined as

$$r(\mathbf{x}) \propto \frac{posterior(\mathbf{x})}{prior(\mathbf{x})} \tag{5.22}$$

From Eq. (5.21), we have

$$\frac{posterior(\mathbf{x})}{prior(\mathbf{x})} \propto likelihood(\mathbf{x}) \tag{5.23}$$

Thus, we have the importance ratio at $\mathbf{x} = \mathbf{x}^i_{k|k-1}$ as

$$r(\mathbf{x}^i_{k|k-1}) \propto p(y_k|\ \mathbf{x}^i_{k|k-1},\theta)$$

i.e., the importance ratios are proportional to the likelihood values. By the SIR approach, we *resample* from $\{\mathbf{x}^i_{k|k-1}\}_{i=1}^N$ with probability proportional to $r(\mathbf{x}^i_{k|k-1})$ to obtain the posterior samples.

SIR-particle filter algorithm: Assume that we have $\{\mathbf{x}^i_{k-1|k-1}\}^N_{i=1}$ generated from $p(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1})$. Note that for $k = 1$, a suitable distribution given by $p(\mathbf{x}_0)$ will be used to generate the initial set of samples. Then, we can recursively obtain posterior samples for the next instant k through the following steps:

---

1. Generate samples $\{\mathbf{x}^i_{k|k-1}\}^N_{i=1}$ through N independent forward simulations of the stochastic state space model of the process as follows:

$$\mathbf{x}^i_{k|k-1} = \mathbf{f}(\mathbf{x}^i_{k-1}, \mathbf{u}_{k-1}, \mathbf{v}^i_{k-1}), for\ i = 1\ to\ N$$

   where, $\mathbf{v}^i_{k-1}$ is drawn from the process noise distribution. The samples $\{\mathbf{x}^i_{k|k-1}\}^N_{i=1}$ represent samples from the prior $p(\mathbf{x}_k|\mathbf{y}_{1:k-1})$

2. Calculate the weights $w^i = \frac{p(y_k|\ \mathbf{x}^i_{k|k-1})}{\sum_{i=1}^{N} p(y_k|\ \mathbf{x}^i_{k|k-1})}\ \forall\ i$

3. Resample N particles $\{\mathbf{x}^i_{k|k}\}^N_{i=1}$ from the discrete distribution $\{\mathbf{x}^i_{k|k-1}, w^i; i = 1 : N\}$. These represent samples from the posterior $p(\mathbf{x}_k|\mathbf{y}_{1:k})$.

Note that the Eqs. (5.19) and (5.20) constitute the steps for obtaining the smoothed distribution. In the particle smoother, these backward smoothing equations are applied on the filtered particles to obtain an approximation to the smoothed density. This will result in a sample based approximation of the smoothed density functions which can be represented as:

$$p(x_k|\mathbf{Y}_T, \theta_i) \approx \sum_{i=1}^{N} w^i_k \delta(x_k - x^i_k) \tag{5.24}$$

$$p(x_k, x_{k+1}|\mathbf{Y}_T, \theta_i) \approx \sum_{i=1}^{N} w^i_{k,k+1} \delta(x_k - x^i_k)\delta(x_{k+1} - x^i_{k+1}) \tag{5.25}$$

where, w^i_k is the weight computed by the smoothing algorithm for the ith particle. This weighted approximation of the smoothed density is computed using the forward-backward implementation of the particle smoother equations given in Gopaluni (2008a). The approximation of the Q-function, obtained by substituting Eqs. (5.24) and (5.25) into (5.16), is given by

$$\begin{aligned} Q(\theta, \theta_i) \approx & \sum_{i=1}^{N} w^i_1 \log[p(x^i_1|\theta)] \\ & + \sum_{k=2}^{T}\sum_{i=1}^{N} w^i_{k,k-1} \log[p(x^i_k|x^i_{k-1}, \theta)] \\ & + \sum_{k=1}^{T}\sum_{i=1}^{N} w^i_k \log[p(y_k|x^i_k, \theta)] \end{aligned} \tag{5.26}$$

Unscented Kalman smoother:

The unscented Kalman filter (UKF), from which the corresponding smoother is developed, is a sub-optimal and approximate extension of the Kalman filter for nonlinear systems. The main assumption in this technique is that all the required densities (posterior or smoothing distribution) can be approximated using just the mean and covariance. Since the system equations are nonlinear, the basic Kalman filter equations for propagating the covariance do not directly apply. Hence, in the UKF algorithm the unscented transform is used to form Gaussian approximations to random variables that result from nonlinear transformation of Gaussian random variables (Julier et al., 2000). In the current work, a forward-backward unscented Kalman smoother (UKS) proposed by Sarkka (2008) was employed and the performance of the EM algorithm was analyzed. Compared to the particle smoother, the UKS is computationally far less complex. Unlike the particle based methods, which require large number of Monte Carlo samples, the UKS uses a few deterministic samples ($2l + 1$ samples) for approximating the smoothed distributions. The Gaussian approximations that are obtained using the UKS are given by (see Sarkka (2008)):

$$p(x_k|\mathbf{Y}_T, \theta_i) \approx N(x_k|\mu_k^T, P_k^T) \tag{5.27}$$

$$p(x_k, x_{k+1}|\mathbf{Y}_T, \theta_i) \approx N(x_k|\mu_{k,k+1}^T, P_{k,k+1}^T) \tag{5.28}$$

where, μ_k^T, P_k^T are the mean and covariance of the respective density functions. Using random i.i.d samples from the above Gaussian distributions, we can approximate the Q-function using Eqs. (5.15) and (5.16) as

$$\begin{aligned} Q(\theta, \theta_i) \approx & \frac{1}{N}\sum_{i=1}^{N} \log[p(x_1^i|\theta)] \\ & + \frac{1}{N}\sum_{k=2}^{T}\sum_{i=1}^{N} \log[p(x_k^{*i}|x_{k-1}^{*i}, \theta)] \\ & + \frac{1}{N}\sum_{k=1}^{T}\sum_{i=1}^{N} \log[p(y_k|x_k^i, \theta)] \end{aligned} \tag{5.29}$$

where, x_k^{*i} are i.i.d samples from the Gaussian density in Eq. (5.27) and $[x_k^{*i}, x_{k-1}^{*i}]$ sampled similarly from the joint density in Eq. (5.28).

Extended Kalman smoother:

The application of Kalman smoother for parameter estimation using the EM algorithm for linear systems was presented in Shumway & Stoffer (1982). In the extended Kalman smoother approach, a local linearization of the nonlinear dynamic equations based on Taylor series is used to apply Kalman smoother equations on

nonlinear systems. In the current work, the Rauch–Tung–Striebel smoother type of the extended Kalman smoother (EKS) given in Haykin (2001) was employed. Similar to the UKS, the EKS obtains Gaussian approximation of the smoothed densities. Then, i.i.d samples can be drawn from the Gaussian densities to approximate the Q-function as shown in Eq. (5.29). The computational complexity of EKS is comparable to the UKS. However, the EKS suffers from additional approximation errors since the error covariance matrices are propagated through the linearized state space equations.

5.2.3 Computational cost

The particle smoother requires $O(N^2 lT)$ computations for each iteration of the EM algorithm, where l is the state space dimension and T is the data length (number of samples). The UKS and EKS require $O(l^3 T)$ computations, arising from the smoothed covariance matrix computation equation. Usually the number of particles in the particle smoother will be much higher compared to the state space dimension, i.e. $N \gg l$; a major factor which will make the EM algorithm based on particle smoother computationally more demanding than the one with UKS/EKS. One of the major factors that will contribute to the computational cost in all nonlinear filters and smoothers is the number of times the differential equations in the state space model have to be integrated forward in time for each sampling interval. The particle smoother will require N such integrations *per sampling interval*; the UKS will require $2l + 1$ and the EKS requires only one integration per sampling interval. The fact that $N \gg l$ will significantly increase the computational cost for the particle smoother compared to UKS and EKS. Also, the calculation of exponential likelihood functions is a computationally intensive step in the particle smoother. Hence, the overall cost will be highest for the particle smoother, with little differences between the UKS/EKS algorithms. However, note that the particle based filter/smoother is well suited for parallel implementation where different processors can be assigned a subset of the particles. Also, there is no need to store covariance matrices which is an advantage for large scale systems in terms of reduced memory usage.

5.3 Recursive Parameter Estimation

The batch estimation algorithm presented above is suitable for off-line parameter estimation and when the system parameters do not vary within the batch interval. The technique is computationally intensive due to the particle approximations of the integrals involved in the expectation step. For systems whose parameters significantly vary with time, the parameters have to be updated in a recursive fashion.

This section presents an approach where the unknown parameters are treated as *random variables* in the form of "hidden" states. The states of the process are augmented with the parameters to create a new state vector, $z_k = [x_k, \theta_k]$. Then, $\widehat{Z}_T = \{\widehat{z}_1, \widehat{z}_2 .... \widehat{z}_T\}$ can be estimated sequentially using a standard state estimation algorithm. For the case of static parameter estimation, θ_k can be either modeled as a process with no dynamics or can be considered to be a 'slowly' varying parameter. In terms of MLE, this method is equivalent to maximizing the likelihood function defined by the conditional probability density $p(z_k|\mathbf{Y}_k, \mathbf{U}_k)$. Note that this likelihood function represents the probability density of the unknown quantities conditioned on the observed quantities and belongs to the Bayesian framework. However, for implementing a state estimation scheme on the augmented state space system we have to depend on *a priori* information about the distribution of the parameters. Though suitable for online implementation, the disadvantage of this method is that the error in the estimate of the parameters will accumulate over time and can lead to divergence of the estimates (Andrieu et al., 2004).

In the current work, the evolution of the unknown parameters is modeled using a random walk model with additive zero-mean Gaussian noise. The evolution of the extended/augmented state vector for the nonlinear system in Eq. (5.1) will be given by

$$z_{k+1} \equiv \begin{bmatrix} x_{k+1} \\ \theta_{k+1} \end{bmatrix} = \begin{bmatrix} f(x_k, u_k, \Theta_k) \\ \theta_k \end{bmatrix} + \begin{bmatrix} v_k \\ w_k \end{bmatrix} \tag{5.30}$$

$$y_k = h(x_k, \Theta_k) + n_k$$

where, Θ_k represents the complete parameter vector and θ_k is the subset of the parameters which have to be estimated. Assume that $\theta_k \in \mathbb{R}^p$ and that the noise vector w_k is normally distributed as $N(0, \mathbf{Q}_w)$, $\mathbf{Q}_w \in \mathbb{R}^{p \times p}$. The parameter covariance matrix $\mathbf{Q}_w$ can be heuristically chosen to reflect prior knowledge about the variation in the parameters. In order to estimate the parameters in an online fashion we can employ any of the popular nonlinear state estimation algorithms, by providing an initial guess for the state and parameter distributions. A Gaussian distribution can be assumed as the initial guess for the parameters. A good initial guess for the mean value of the parameter distribution can be obtained through an EM-based parameter estimation algorithm on a batch of historical process operational data, if available. The update step in the nonlinear filter will perform a combined correction of the state as well as the parameters whenever measurements are available from the process. In this work two nonlinear filters are compared: the ensemble Kalman filter (EnKF) and the UKF. Both these filters are more suitable than the EKF, since in these methods the nonlinear state space equations are directly used for the error covariance propagation. A detailed description of the basic UKF

algorithm can be found in Julier & Uhlmann (1997), Wan & van der Merwe (2000). In the following section, a brief description of the ensemble Kalman filter algorithm is given.

5.3.1 The ensemble Kalman filter

The ensemble Kalman filter, proposed by Evensen (1994), is a Monte Carlo based variant of the Kalman filter. Numerous applications of this method for nonlinear state estimation, especially for large scale systems, have been shown in the oceanography and meteorology literature (Evensen, 2003). However, the application of this technique for parameter estimation of chemical and biological systems has not been investigated in the past literature. As is the case with the particle filter algorithm, in EnKF it is not required to store the error covariance matrix and no explicit covariance evolution equations are needed. The main difference between particle filter and the EnKF is in the state update step - in the EnKF algorithm, a Kalman gain based linear update is used for each particle (referred as 'ensemble members') whereas a re-sampling step is used in a particle filter (Evensen, 2009). Therefore, in the EnKF algorithm it is not required to calculate exponential density functions in the update step. This is a major reduction in computational cost compared to the particle filter. The main assumption in EnKF is that the prior distribution of the state follows a multivariate normal distribution. Therefore, the prior distribution covariance is used in the update step. Typical implementations of the EnKF in the literature suggest that an ensemble size of $O(100)$ is sufficient to get acceptable estimates of the states. A comprehensive overview of the EnKF algorithm is given in Evensen (2007). In the recent past, this method has been demonstrated to be effective for parameter estimation of highly nonlinear large scale models of oil and gas reservoirs (Seiler et al., 2009). In this section, the basic steps of the EnKF algorithm for state estimation is briefly presented.

Assume that a nonlinear state space model of the process is available in the form of Eq. (5.1). The augmented state vector form, as shown in Eq. (5.30), can be used in order to directly extend the algorithm for combined state and parameter estimation. The EnKF uses an ensemble of Monte Carlo samples to represent the state prediction error and the output error (innovation) statistics. The sample covariance estimate of these quantities will be used to calculate the Kalman gain. At time k, assume that we have an ensemble of augmented state estimates given by:

$$\widehat{\mathbf{Z}}_{k|k} \equiv (z^1_{k|k}, z^2_{k|k}, ..., z^N_{k|k})$$

where, $z^i_{k|k} = [x^i_{k|k}, \theta^i_{k|k}] \in \mathbb{R}^{(l+p)}$, $\widehat{\mathbf{Z}}_{k|k} \in \mathbb{R}^{N\times(l+p)}$, N is the ensemble size and $(.)_{k|k}$ denotes the updated value of the vector given the measurements up to kth time instant. For the subsequent time instant $k+1$, the EnKF proceeds as follows:

1. Prediction step:

The *Prediction step* is similar to that of particle filter: a one-step Monte Carlo forward simulation of each ensemble member is performed using the extended state evolution equation (5.30), by including the noise vectors v_t and w_t in the form of a random sample. We can represent the predicted state ensemble as $\widehat{\mathbf{Z}}_{k+1|k} \equiv (z^1_{k+1|k}, z^2_{k+1|k}, ..., z^N_{k+1|k})$, where $z^i_{k+1|k}$ is obtained by the equation

$$z^i_{k+1|k} = \begin{bmatrix} f(x^i_{k|k}, u_k, \Theta^i_{k|k}) \\ \theta^i_{k|k} \end{bmatrix} + \begin{bmatrix} v^i_k \\ w^i_k \end{bmatrix} \tag{5.31}$$

Note that v^i_k and w^i_k are random samples drawn from the distributions $N(0, \mathbf{Q}_v)$ and $N(0, \mathbf{Q}_w)$ respectively. The *state prediction error covariance* matrix, $\widehat{\mathbf{P}}^{\mathbf{z}}_{k+1|k}$, is approximated as the sample covariance of $\widehat{\mathbf{Z}}_{k+1|k}$ around the sample mean $\boldsymbol{\mu}^{\mathbf{z}}_{k+1|k} = \frac{1}{N}\sum_{i=1}^{N} z^i_{k+1|k}$:

$$\widehat{\mathbf{P}}^{\mathbf{z}}_{k+1|k} = \frac{1}{N-1}\sum_{i=1}^{N}(e^i_{k+1|k})(e^i_{k+1|k})^T \tag{5.32}$$

where, $e^i_{k+1|k} = z^i_{k+1|k} - \boldsymbol{\mu}^{\mathbf{z}}_{k+1|k}$ are the prediction error samples. In order to calculate *output error (innovation) covariance*, measurement samples are generated using the predicted state ensemble and the measurement equation in Eq. (5.30). An ensemble of *perturbed observations*, $\widehat{\mathbf{Y}}_{k+1|k} \equiv (y^1_{k+1|k}, y^2_{k+1|k}, ..., y^N_{k+1|k})$ are generated using a stochastic realization of the measurement equation in order to account for the measurement error variance, i.e.,

$$y^i_{k+1|k} = h(x^i_{k+1|k}, \Theta^i_{k+1|k}) + n^i_{k+1} \tag{5.33}$$

where, n^i_{k+1} is a random sample drawn from the distribution $N(0, \mathbf{R})$. The output error covariance matrix is defined as the sample covariance of $y^i_{k+1|k}$ around the sample mean $\boldsymbol{\mu}^{\mathbf{y}}_{k+1|k} = \frac{1}{N}\sum_{i=1}^{N} y^i_{k+1|k}$:

$$\widehat{\mathbf{P}}^{\mathbf{y}}_{k+1|k} = \frac{1}{N-1}\sum_{i=1}^{N}(\boldsymbol{\varepsilon}^i_{k+1|k})(\boldsymbol{\varepsilon}^i_{k+1|k})^T \tag{5.34}$$

where, $\boldsymbol{\varepsilon}^i_{k+1|k} = y^i_{k+1|k} - \boldsymbol{\mu}^{\mathbf{y}}_{k+1|k}$. Similarly, a cross-covariance matrix between prediction error and output error is defined as

$$\widehat{\mathbf{P}}^{\mathbf{zy}}_{k+1|k} = \frac{1}{N-1}\sum_{i=1}^{N}(e^i_{k+1|k})(\boldsymbol{\varepsilon}^i_{k+1|k})^T \tag{5.35}$$

2. Update step:

The *Update step* is similar to the Kalman filter, where each of the samples in the predicted state estimate ensemble is updated using the measurements y_{k+1} from

the process. The *Kalman gain* is computed using the error covariance matrices computed above as

$$K_{k+1} = \widehat{\mathbf{P}}^{\mathbf{zy}}_{k+1|k}(\widehat{\mathbf{P}}^{\mathbf{y}}_{k+1|k})^{-1} \tag{5.36}$$

The updated state ensemble is given by

$$\begin{aligned} \widehat{\mathbf{Z}}_{k+1|k+1} &\equiv (z^1_{k+1|k+1}, z^2_{k+1|k+1}, ..., z^N_{k+1|k+1}) \\ where,\ z^i_{k+1|k+1} &= z^i_{k+1|k} + K_{k+1}(y_{k+1} - y^i_{k+1|k}) \end{aligned} \tag{5.37}$$

These samples will be used to re-initialize the model for the subsequent time instant and the algorithm will proceed by following the above steps. The optimal estimate of the augmented state vector can be defined as the mean of the updated state ensemble:

$$\widehat{z}^*_{k|k} = \frac{1}{N}\sum_{i=1}^{N} z^i_{k|k} \tag{5.38}$$

The sample covariance of the updated state ensemble, $\widehat{\mathbf{P}}^*_{k|k}$, can be computed as

$$\widehat{\mathbf{P}}^*_{k|k} = \frac{1}{N-1}\sum_{i=1}^{N}(z^i_{k|k} - \widehat{z}^*_{k|k})(z^i_{k|k} - \widehat{z}^*_{k|k})^T \tag{5.39}$$

which will provide the quality of the state and parameter estimates generated by the EnKF.

5.4 Example: Continuous Fermentor

In this section, the performance of the different algorithms mentioned above is compared on a simulated continuous fermentation reactor example. Initially, the batch estimation methods are compared and then the recursive estimation methods using EnKF and UKF. The continuous time nonlinear state space model of the reactor is as follows (Henson & Seborg, 1997):

$$\frac{dX}{dt} = -DX + \mu(P,S)X \tag{5.40a}$$

$$\frac{dS}{dt} = D(S_f - S) - \frac{1}{Y_{x/s}}\mu(P,S)X \tag{5.40b}$$

$$\frac{dP}{dt} = -DP + (\alpha\mu(P,S) + \beta)X \tag{5.40c}$$

$$\mu(P,S) = \frac{\mu_m\left(1 - \frac{P}{P_m}\right)S}{K_m + S + \frac{S^2}{K_i}} \tag{5.40d}$$

where the states, measurements and the inputs are respectively defined as $x_k \equiv [X_k, S_k, P_k] \in R^3, y_k \equiv [X_k, P_k] \in R^2, u_k \equiv [D, S_f] \in R^2$. The variables in the reactor model equations in Eq. (5.40 a-d) represent the following reactor attributes: Biomass concentration (X), Substrate concentration (S), Product concentration

(P), Dilution rate (D), Feed substrate concentration (S_f), Specific growth rate ($\mu(P,S)$), Yield parameters ($Y_{x/s}, \alpha, \beta$), Maximum specific growth rate (μ_m) and reaction constants (P_m, K_m, K_i).

Table 5.1: Fermentor model parameters.

Parameter	**Value**
$\mu_m(=\theta_1)$	0.48 h^{-1}
$K_m(=\theta_2)$	1.2 g/L
$1/P_m(=\theta_3)$	0.02 L/g
$1/K_i(=\theta_4)$	0.0455 L/g
$1/Y_{x/s}(=\theta_5)$	2.5 g/g
$\alpha(=\theta_6)$	2.2 g/g
$\beta(=\theta_7)$	0.2 h^{-1}

The initial conditions for the reactor simulation were assumed to be $x_0 = [6, 5, 19.14]$. Table 5.1 shows the fermentor model parameter values that were used in the simulation study. A simple forward Euler integration with an integration time step of $0.01h$ was used to numerically solve the continuous time differential equations in MATLAB. In order to convert the continuous time state space model into discrete time state space model of the form shown in Eq. (5.1), a sampling time of 0.1 h was used. The covariance matrices of state noise and measurement noise were assumed as $\mathbf{Q}_v = [0.01, 0, 0; 0, 0.1, 0; 0, 0, 0.01]$ and $\mathbf{R} = [0.1, 0; 0, 0.1]$ respectively. A Random Binary Sequence (RBS) based variation in the inputs, around a nominal value of 0.2, 20 for D and S_f respectively, was used to excite the process for collecting identification data from the simulation model. The input D was varied between 0.1 h^{-1} and 0.3 h^{-1}, whereas S_f was varied between 10 g/L and 30 g/L. The simulation was carried out for 2000 sampling instants and the input-output data so generated was used for parameter identification.

The yield parameters in a typical reactor depend on the structure of the underlying reaction network, which are difficult to model for many complex biotechnological processes. Therefore, it is often desirable to identify the yield parameters using past data of the process operation. In this work, it was assumed that two of the yield parameters, θ_5 and θ_6, are not known. Our objective is to identify these two unknown parameters using data from the simulation. Note that an initial guess for the unknown parameter values is required in the batch and recursive estimation algorithms presented earlier. In all the empirical studies presented in the following sections, an initial guess of $[\theta_5, \theta_6] = [1, 1]$ was used. For all the estimation algorithms (batch and recursive), it was assumed that the initial conditions of the reactor are accurately known. Since the state S_k, corresponding to the substrate concentration, is not measured (θ_5 directly influences this state) and the measurements of the other two states are corrupted by noise, the parameters have to be estimated simultaneously

with the states.

5.4.1 Batch estimation results

The performance of different cases of the EM algorithm was compared, which differ in the particular smoother being employed, using the same realization of the simulation. In order to compare the computational complexity, the EM algorithm was stopped at the end of 100 iterations for all the cases. $N = 100$ particles/samples were used for the Q-function approximation shown in Eqs. (5.26) and (5.29). An initial state covariance matrix of $0.01I_{3\times3}$ was used in all the smoother algorithms, where $I_{3\times3}$ represents a 3×3 identity matrix. The sigma points ($2l + 1 = 7$) for the UKS were chosen using the scheme in Wan & van der Merwe (2000).

Figure 5.2 shows the parameter trajectories for 100 iterations. In order to analyze the convergence of the parameters relative to the true values, the ratio between the estimated and true parameters is shown in the figure. For the particle smoother based EM algorithm, the parameters converge in the neighborhood of their true values. However, there exists significant variations in the parameter trajectories and this is attributed to the Monte Carlo nature of the particle smoother. The EKS based estimation results show that there exists an offset in one of the parameters and this can be attributed to the approximation resulting from the linearization of the system model in the EKS algorithm. The parameter convergence is almost perfect for the UKS based EM algorithm and can be concluded to give the best performance. All the MATLAB implementations of the EM algorithm were done on identical computer platforms (Intel Core 2 CPU @ 2.13 GHz). It took 12 hours of CPU time for the execution of 100 iterations of the particle smoother based EM algorithm; whereas, for the UKS and EKS based implementations of the EM algorithm it took 2.5 hours only. Based on this, we can conclude that the particle smoother based EM algorithm is computationally very intensive. This higher computational cost is mainly due to the Monte Carlo simulation based smoothing.

5.4.2 Recursive estimation results

In this section, the results of recursive parameter estimation for the fermentor reactor is compared using EnKF and UKF. As mentioned earlier, the recursive estimation technique is useful for online implementation, especially when the process parameters are susceptible to change. For comparing the performance of the EnKF and UKF based recursive parameter estimation, it was assumed that the yield parameters in the actual fermentor reactor example undergo a step change during the simulation as follows:

$$\begin{aligned} \{\theta_5 &= 2.5, \theta_6 = 2.2 \}, \ for\ k < 1000 \\ \{\theta_5 &= 1.5, \theta_6 = 1 \}, \ for\ k > 1000 \end{aligned}$$

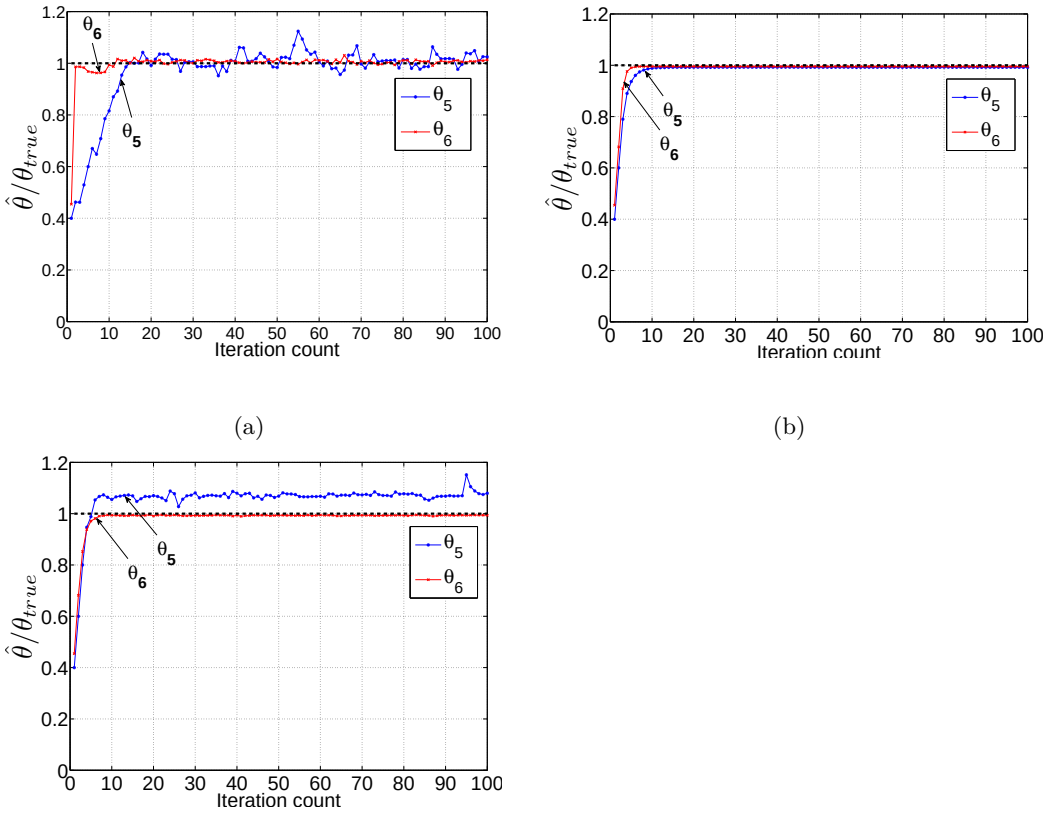
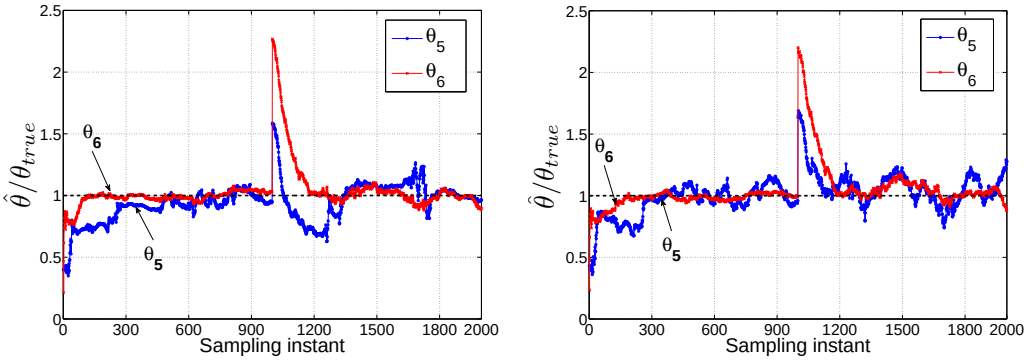


Figure 5.2: Comparison of parameter convergence of EM algorithms employing different smoothing techniques: (a) Particle smoother, (b) Unscented Kalman smoother, (c) Extended Kalman smoother.

Using the above parameter switching scheme and keeping all other simulation settings to be same as described earlier, a new realization of the reactor simulation was generated. This data set was used for evaluating the performance of EnKF and UKF based recursive parameter estimation.

The initial conditions x_0 of the reactor was assumed to be accurately known. Similar to the batch estimation study, an initial guess of $[\theta_5, \theta_6] = [1, 1]$ was given for the unknown parameters. In order to do the recursive estimation, the unknown yield parameters θ_5 and θ_6 were augmented to the state vector as shown in Eq. (5.30). The augmented state vector will have a dimension of 5 (3 process states and 2 unknown parameters). The initial covariance matrix of the augmented state was assumed to be $I_{5\times5}$. A covariance of $\mathbf{Q}_w = [1e-3,\ 0; 0,\ 1e-4]$ was assumed in the random walk model of the unknown parameters. The covariance matrices for the state noise ($\mathbf{Q}_v$) and measurement noise ($\mathbf{R}$) were same as described earlier. The weak noise variance in $\mathbf{Q}_w$ is a tuning setting for the filter which specifies that the dynamics of the parameters is much slower compared to that of the process states.

Figure 5.3 shows the parameter estimate trajectories using the UKF and EnKF (ensemble size of 500), for 2000 sampling instants. Note that the parameters are updated at every sampling instant of the simulation. Hence, the figures show the temporal evolution in real time with respect to the process simulation. From the figures, we can observe that the parameter estimates are not only converging in the neighborhood of the true values but are also tracking the step change that occurred after 1000 sampling instants. Both the algorithms give similar performance in terms of accuracy. The EnKF was found to be slower in execution due to the 500 ensemble model integrations required at each instant. However, this can be brought down by a parallel implementation of the ensemble operations.



(a) (b)

Figure 5.3: Comparison of parameter convergence for recursive estimation using: (a) UKF, (b) EnKF.

5.5 Concluding Remarks

In this work, an empirical comparison of different techniques for combined state and parameter estimation of a continuous fermentor reactor model was presented. Two methodologies were discussed: a batch parameter estimation scheme based on the EM algorithm and recursive parameter estimation using a parameter augmented state vector in a state estimation scheme. The performance of the batch estimation technique depended on the choice of the nonlinear smoothing algorithm. The UKS based EM parameter estimation technique was able to generate unbiased parameter estimates. The particle smoother based EM parameter estimates converged in the neighborhood of their true values, but had significant variance due to the Monte Carlo nature of the smoothing technique. The EKS based EM parameter estimates suffered from a bias problem due to the use of linearized state space models for finding the smoothing distributions. The particle smoother based technique was found to be computationally intensive compared to the UKS and EKS based

techniques. In general, the batch EM based estimation is computationally more intensive compared to a recursive estimation technique. The performance of EnKF and UKF algorithms, when employed for recursive estimation of the parameters, were compared. Both the filters gave similar performance for parameter estimation. The results also showed that the recursive estimation can track changes in the unknown parameter values in an online fashion.

Bibliography

Andrieu, C. & Doucet, A. (2003). Online expectation-maximization type algorithms for parameter estimation in general state space models. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 6, (pp. 69–72).

Andrieu, C., Doucet, A., Singh, S., & Tadic, V. (2004). Particle methods for change detection, system identification, and control. *Proceedings of the IEEE*, *92*(3), 423–438.

Arulampalam, S. M., Maskell, S., Gordon, N., & Clapp, T. (2002). A tutorial on particle filters for online nonlinear/non-gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, *50*(2), 174–188.

Boyd, S. & Vandenberghe, L. (2006). *Convex Optimization*. Cambridge University Press.

Chen, T., Morris, J., & Martin, E. (2005). Particle filters for state and parameter estimation in batch processes. *Journal of Process Control*, *15*, 665–673.

Chitralekha, S. B., Prakash, J., Raghavan, H., Gopaluni, R., & Shah, S. L. (2010). A comparison of simultaneous state and parameter estimation schemes for a continuous fermentor reactor. *Journal of Process Control*, *20*(8), 934 – 943.

Chitralekha, S. B., Prakash, J., Raghavan, H., Gopaluni, R. B., & Shah, S. L. (2009). Comparison of expectation-maximization based parameter estimation using particle filter, unscented and extended Kalman filtering techniques. In Walter, E. (Ed.), *15th IFAC Symposium on System Identification, Saint-Malo, France*, (pp. 804–809).

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, *39*(1), 1–38.

Doucet, A. & Tadic, V. B. (2003). Parameter estimation in general state-space models using particle methods. *Annals of the Institute of Statistical Mathematics*, *55*(2), 409–422.

Evensen, G. (1994). Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. *Journal of Geophysical Research*, *99*, 10143–10162.

Evensen, G. (2003). The ensemble Kalman filer: theoretical formulation and practical implementation. *Ocean Dynamics*, *53*, 343–367.

Evensen, G. (2007). Estimation in an oil reservoir simulator. In *Data Assimilation: The Ensemble Kalman Filter* (pp. 239–248). Springer, Berlin.

Evensen, G. (2009). The ensemble Kalman filter for combined state and parameter estimation. *IEEE Control Systems Magazine*, *29*(3), 82–104.

Gopaluni, R. B. (2008a). Identification of nonlinear processes with known model structure under missing observations. In *Proceedings of the 17th IFAC World Congress*, Seoul, South Korea.

Gopaluni, R. B. (2008b). A particle filter approach to identification of nonlinear processes under missing observations. *The Canadian Journal of Chemical Engineering*, *86*, 1081–1092.

Gopaluni, R. B. (2010). Nonlinear system identification under missing observations: The case of unknown model structure. *Journal of Process Control*, *20*(3), 314–324.

Gordon, N. J., Salmond, D. J., & Smith, A. F. M. (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *Radar and Signal Processing, IEE Proceedings F*, *140*(2), 107–113.

Haykin, S. (2001). Kalman filters. In *Kalman Filtering and Neural Networks*. John Wiley & Sons Inc., New York.

Henson, M. A. & Seborg, D. E. (1997). *Nonlinear process control*. Prentice-Hall Inc., USA.

Julier, S. & Uhlmann, J. (1997). New extension of the Kalman filter to nonlinear systems. In Kadar, I. (Ed.), *Signal Processing, Sensor Fusion, and Target Recognition VI*, volume 3068 of *SPIE proceedings*, (pp. 182–193).

Julier, S., Uhlmann, J., & Durrant-Whyte, H. (2000). A new method for the nonlinear transformation of means and covariances in filters and estimators. *IEEE Transactions on Automatic Control*, *45*(3), 477–482.

Kantas, N., Doucet, A., Singh, S. S., & Maciejowski, J. (2009). An overview of sequential Monte Carlo methods for parameter estimation on general state space models. In Walter, E. (Ed.), *Proceedings of the 15th IFAC Symposium on System Identification*, volume 15, (pp. 774–785).

Kim, J. (2005). *Parameter Estimation in Stochastic Volatility Models with Missing Data using Particle Methods and the EM algorithm*. PhD thesis, Department of Statistics, University of Pittsburgh.

Lorentzen, R., Naevdal, G., Valles, B., & Berg, A. (2005). Analysis of the ensemble Kalman filter for estimation of permeability and porosity in reservoir models. In *SPE Annual Technical Conference and Exhibition, Dallas, Texas*.

Maybeck, P. S. (1972). *Combined state and parameter estimation for online applications*. PhD thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology.

Poyiadjis, G., Doucet, A., & Singh, S. S. (2005). Maximum likelihood parameter estimation in general state-space models using particle methods. In *Proceedings of the American Statistical Association*, Minneapolis, USA.

Rubin, D. B. (1988). Using the SIR algorithm to simulate posterior distributions. *Bayesian Statistics*, *3*, 395–402.

Sarkka, S. (2008). Unscented Rauch-Tung-Striebel smoother. *IEEE transactions on Automatic control*, *53*, 845–849.

Seiler, A., Evensen, G., Skjervheim, J.-A., Hove, J., & Vabo, J. (2009). Advanced reservoir management workflow using an EnKF based assisted history matching method. SPE Reservoir Simulation Symposium, Woodlands, TX SPE 118906, Society of Petroleum Engineers, Inc.

Shumway, R. H. & Stoffer, D. S. (1982). An approach to time series smoothing and forecasting using the EM algorithm. *Journal of Time Series Analysis*, *3*, 253–264.

Wan, E. A. & van der Merwe, R. (2000). The unscented Kalman filter for nonlinear estimation. In *Proceedings of IEEE 2000 Adaptive Systems for Signal Processing, Communication and Control Symposium*, (pp. 153–156)., Alberta, Canada.

Wills, A. G., Schon, T., & Ninness, B. (2008). Parameter estimation for discrete-time nonlinear systems using EM. In *Proceedings of the 17th IFAC World Congress*, Seoul, South Korea.

Chapter 6

History Matching of Unconventional Oil Reservoirs using the Ensemble Kalman Filter[1,2]

6.1 Introduction

History matching is the art of calibrating reservoir simulation models to past production data available from the field. From a systems engineering point of view, this can be defined as identifying models that are consistent with data collected from the process. In the context of petroleum engineering, the process of interest is the flow of multi-component fluids (mainly mixture of oil, gas and water) through the heterogeneous geological layers of the reservoir. The reservoir simulation models are a set of highly nonlinear, partial differential equations in space and time. Typically, the differential equations are functions of several model parameters - this will include the spatially varying rock properties (permeability, porosity), initial conditions of the reservoir, relative permeability curves, and other physical and chemical properties of the reservoir strata and fluids. All these properties of the reservoir are collectively known as model parameters. Thanks to the modeling effort of reservoir engineers and physicists, today there are several sophisticated reservoir simulation packages available off-the-shelf. These packages can integrate the complex physics along with efficient numerical algorithms and simulate the reservoir, given the values of all the model parameters in the differential equations. From the solutions of the differential equations, we can obtain simulated production data. The predictions

[1] Some sections of this chapter have been accepted as a paper for presentation at the Canadian Unconventional Resources & International Petroleum Conference, to be held in Calgary, Alberta, Canada on October 19 – 21, 2010 (Chitralekha et al., 2010).

[2] A full version of this study is under review for publication in the Journal of Canadian Petroleum Technology since August, 2010.

that we obtain from the simulations will depend on these model parameters. If the objective of modeling the reservoir is to get reliable predictions of the future performance of the reservoir, then it is required to have a good knowledge of the values of these model parameters. However, in a practical scenario many of these model parameters will be uncertain or completely unknown because of our inability to directly measure them in the true reservoir. It is a well accepted fact that all reservoirs are heterogeneous with different degrees of heterogeneity exhibited by different properties. For example, the geological properties such as the permeability and porosity can vary in a spatial scale of a few meters. However, these properties may only be known for around 1% of the reservoir volume through core samples taken at well locations (Romeu, 2010). The geological uncertainty is inevitable in any reservoir simulation based decision making exercise (McLennan & Deutsch, 2006). Through the use of sophisticated geostatistical algorithms, such as GSLIB (Deutsch & Journel, 1998), SGeMS (Remy et al., 2009) etc., we can capture the spatial variation to some extent. However, the exact geological heterogeneity is never known and can only be approximately predicted in between wells through these stochastic realizations (McLennan & Deutsch, 2010). The geological uncertainty is usually represented by generating multiple stochastic realizations which honor the spatial variogram models. The traditional way of validating a particular geological realization is to manually compare the corresponding flow simulations with the production data from the true reservoir.

History matching is essentially an iterative model validation procedure - given some historical production data as the target, the objective is to find a set of model parameters which when plugged into the model equations will result in a simulated production data close to the historical production data of the wells. In other words, it is an iterative task of tuning the model parameters which will yield simulation results that are consistent with production history of the wells. The production data will include measurements that are routinely available from the wells such as oil/gas production rates, steam-to-oil ratios, well water cuts etc. It is well known that such production variables are sensitive to the geological heterogeneity. Traditionally, history matching was done by manually adjusting the model parameters, either directly or indirectly through the geostatistical models. Evidently, this is a trial-and-error approach and is known to be highly time-consuming, cumbersome and ineffective (Romeu, 2010). More importantly, the problem is known to be underdetermined and hence there will exist multiple plausible solutions (Oliver et al., 2008). A single history matched model will be biased towards a few model parameters. Hence, modern history matching techniques are aimed at arriving at a multiple set of history matched realizations. Such solutions will satisfy the needs of decision making in the face of uncertainty in the production forecasts. Moreover, the

advancement in sensor technologies have resulted in the use of permanent sensors which can provide reservoir production data more frequently than was possible with the legacy sensors. There is a growing need to develop fast history matching algorithms that can continuously update the reservoir models by integrating the most recent information available through such modern day sensors (Seiler et al., 2009).

Several approaches have been proposed in the literature with a focus on the modern day needs of continuous history matching of reservoir simulation models in the face of uncertainty. These techniques broadly fall into two categories: stochastic global optimization methods (Schulze-Riegert et al., 2001) and Ensemble Kalman filter based methods (Lorentzen et al. (2001), Evensen (2007), Aanonsen et al. (2009)). The stochastic optimization methods are based on the idea of minimizing an objective function which quantifies the mismatch/error between model predictions and the observed data. The objective function is usually chosen as a quadratic function, such as the sum of square of the error (SSE). Due to the highly nonlinear nature of the reservoir simulation models and the large number of optimization variables, gradient-free evolutionary optimization algorithms are used. The numerical cost associated with such optimization algorithms become unacceptably high because of the large number of forward simulations required to iteratively converge to the optimal solution, especially when there are large number of unknown parameters. Modern distributed computing technologies are employed to arrive at a solution within reasonable time frames for decision making in a history matching project (Schulze-Riegert & Ghedan, 2007).

Recently, the Ensemble Kalman filter (EnKF) has emerged as a very promising alternative because of its non-iterative nature and requirement of only O(50) forward simulations for estimating model parameters in real reservoir case studies (Evensen (2007), Haugen et al. (2008), Zhang & Oliver (2009)). The method is based on the idea of making probabilistic inference using the classical Bayes' theorem. It is based on the classical Kalman filter, which was proposed in the 1960's for estimating unmeasured variables in linear systems using a model and data collected from the real process (Kalman, 1960). The EnKF is an adaptation of the Kalman filter for handling systems characterized by nonlinear models. The history matching step in the algorithm is a one-step linear update rule proposed in the Kalman filter and does not require any iterative solution. The history matching step is given by the following equation:

$$\boldsymbol{\theta}_{k|k} = \boldsymbol{\theta}_{k-1|k-1} + \mathbf{K}_{gain}(\mathbf{y}_k^{obs} - \mathbf{y}_k^{pred}) \tag{6.1}$$

where $\boldsymbol{\theta}_{k|k}$ is the vector of unknown parameters, $\mathbf{y}_k^{obs}$ is the vector of newly observed production data and $\mathbf{y}_k^{pred}$ is the vector of predicted reservoir production based on the most recently history matched models. The subscript k indicates the update

time index and '$k|k$' means the value of the parameter at kth update time index given all the measurements until the kth time index. The vector $\mathbf{K}_{gain}$ is the Kalman gain proposed by Kalman (Kalman, 1960), the sole variable to be calculated, whose calculation is adapted for nonlinear systems in the EnKF (refer to Chapter 3, section 3.5.2). As seen in Eq. (6.1), the update magnitude is proportional to the magnitude of error between the predictions and measured data. The Kalman gain applies a weighted updating to the unknown parameters in an 'intelligent fashion', and depends on the degree of uncertainty in the models and the measurements. Based on Eq. (6.1), we can see another advantage of the method in that it is sequential in nature, i.e., the history matching step depends only on the new measurements and the most recently history matched set of models. This makes this method well suited for continuous updating of reservoir simulation models. More details about the EnKF algorithm steps will be discussed in the next section. Gu & Oliver (2007) have presented a gradient based optimization approach, combining the ideas of the EnKF algorithm and the maximum likelihood parameter estimation method. In this methodology, the prediction step is exactly similar to the EnKF algorithm, whereas the update step is formulated as a maximum likelihood estimation (MLE) problem. The main advantage of this approach is that it can efficiently account for the nonlinear dynamics of the multiphase flow models. Compared to the EnKF algorithm, the method incurs a high computational cost because of the requirement for an iterative solution at each update step. This means that at each update step the method will require $O(L \times N_e)$ reservoir simulations, where L is the number of iterations for convergence of the MLE optimization and N_e is the ensemble size. On the other hand, the update step in the general EnKF algorithm does not require any reservoir simulation.

Most of the applications of EnKF in the current petroleum engineering literature have concentrated on conventional black-oil reservoir models. In this work, the application of the EnKF algorithm for history matching of unconventional reservoir models is investigated. Two simulated case studies are presented: (1) a highly heterogeneous black-oil reservoir model (2-dimensional), and (2) a heterogeneous SAGD reservoir simulation model (50x10x5). The heterogeneous grid block permeabilities were considered as the unknown model variables in the history matching problem. In both cases, a set of realistically placed well-core measurements combined with a geostatistical variogram model is used to generate the initial ensembles for the EnKF algorithm. These application studies will show that using a set of geostatistical realizations as the initial ensemble is an important step to obtain realistic results when the heterogeneity information is available a priori. Otherwise, since the history matching problem is an under determined problem, there is a high chance for the ensembles to move into domains that are unrealistic. This is because

the solution space of the EnKF algorithm is limited by the span of the ensembles (Evensen, 2009).

This chapter is organized as follows. In Section 6.2, the basic methodology of the ensemble Kalman filter algorithm as applied to history matching is explained. Some modifications to the basic algorithm, that we have employed in this work based on previous works in the literature, are briefly discussed. Some insights on the critical tuning factors which can significantly affect the quality of the results are provided. Two quality measures for measuring the efficacy of the EnKF algorithm in any history matching application are proposed. In Section 6.3, two synthetic case studies on the application of the methodology for history matching and characterizing heterogeneous reservoirs is presented. Section 6.4 summarizes this chapter with some concluding remarks.

6.2 Methodology: The Ensemble Kalman Filter

The ensemble Kalman filter was first proposed as a novel method for estimating unmeasured variables, techniques more widely known as data assimilation algorithms, in highly nonlinear ocean models (Evensen, 1994). Numerous applications of this method for nonlinear state estimation, especially for large scale systems, have been shown in the oceanography and meteorology literature (Evensen, 2003). The methodology was introduced as an alternative method for history matching in petroleum engineering by Lorentzen et al. (2001). Here, the basic formulation and a few modifications proposed in earlier works in the literature are explained. The EnKF is a Monte Carlo simulation based variant of the Kalman filter designed for handling nonlinear systems. In order to explain the methodology, consider the following stochastic model of the reservoir:

$$\begin{aligned} \mathbf{x}_k &= \mathbf{f}(\mathbf{x}_{k-1}, \boldsymbol{\theta}) && (6.2) \\ \mathbf{y}_k &= \mathbf{g}(\mathbf{x}_k) + \mathbf{v}_k && (6.3) \\ \mathbf{v}_k &\sim N(\mathbf{0}, \mathbf{R}) \end{aligned}$$

where, $\mathbf{x}_k$ denotes the vector of variables which define the *reservoir state* at the kth instant. This will include the dynamic variables which directly represent the internal conditions at all the grid block locations of the reservoir such as pressure, saturations, temperature. The function $\mathbf{f}$ is the *state transition function* which will relate the current state of the reservoir to the previous state. The partial differential equations which define the dynamics of the reservoir simulation can be considered to be represented by $\mathbf{f}$, i.e.,

$$\mathbf{f} \triangleq \int_{k-1}^{k} \dot{\mathbf{x}} dt$$

In real reservoirs, the variables in $\mathbf{x}_k$ are not measured directly since it is not possible to sample/place online sensor devices at all the grid block locations of the true reservoir. However, all these variables will have an effect on the *production data* that are measured from the reservoir. The variable $\mathbf{y}_k$ represents the dynamic production variables that can be measured at well locations such as oil/gas/water flow rates, monthly/cumulative production, gas-to-oil ratio, steam-to-oil ratio etc. The production data that is obtained from the real reservoir will depend on the reservoir states through the *measurement function* $\mathbf{g}$, and will be contaminated with some sensor noise $\mathbf{v}_k$. It is assumed that $\mathbf{v}_k$ follows a zero-mean Gaussian distribution with covariance matrix $\mathbf{R}$. The variable $\boldsymbol{\theta}$ is the vector of all the static model parameters which are unknown such as permeability and porosity (3-D grids unfolded into a 1-D vector). Since we are uncertain about the true value of $\boldsymbol{\theta}$, we have to consider them to be a random variable following some distribution.

Both $\mathbf{f}$ and $\mathbf{g}$ are part of any commercial reservoir simulator which can be considered as a 'black-box' in the EnKF algorithm (Wen & Chen, 2007). In the EnKF algorithm it is not required to explicitly know any of the functions $\mathbf{f}$ and $\mathbf{g}$. It is only required to obtain the variables $\mathbf{x}_k$ and $\mathbf{y}_k$ through a forward simulation of the reservoir, starting from the initial conditions $\mathbf{x}_{k-1}$. The functions $\mathbf{f}$ and $\mathbf{g}$ are usually highly nonlinear due to the coupled nature of the partial differential equations and the nonlinear mass balance correlations (relating the well production data to the grid block saturations). Note that one of the basic requirements for the Kalman filter computations is that the state transition function and the measurement function be linear functions. Also, the uncertainty in the reservoir parameters and the measurement noise are required to follow a Gaussian distribution. These requirements can be relaxed in the EnKF computations. The main idea is to use a bank of Kalman filters on an *ensemble* of Monte Carlo simulations of the reservoir. The sample covariance/cross-covariance of the simulated ensemble of $\mathbf{x}_k$ and $\mathbf{y}_k$ are used in the calculation of the Kalman gain in the EnKF algorithm.

Note that some of the other extensions of the Kalman filter such as the extended Kalman filter (EKF) and unscented Kalman filter (UKF) may not be practical for history matching applications. The extended Kalman filter is not a preferred approach because of the need for explicitly calculating the Jacobian of the functions $\mathbf{f}$ and $\mathbf{g}$. The large scale nature of the reservoir simulation models makes the UKF a less practical approach for history matching. This is because the UKF will require $O(2\times$dimension of the parameter vector) reservoir simulations in each prediction step. Also, the UKF and EKF will be more memory intensive, if applied to reservoir simulation models since they will involve the computation and storage of covariance matrices of the high dimensional parameter vectors. The particle filter algorithm is the most general sequential Monte Carlo filter. The main difference between the

EnKF and the particle filter is in the update step. The update step of the particle filter is directly based on the Bayes' rule which relates the posterior to the likelihood and the prior distributions. In the update step of the particle filter, the predicted ensembles are resampled based on an approximation of the posterior probability distribution which is known as the 'proposal distribution'. The success of the particle filter highly depends on the quality of this approximation. In general, it is very difficult to get good proposal distributions for high dimensional state space systems such as reservoir simulation models. For poorly chosen proposal distributions, the particle filter will require enormous number of realizations (particles), which means enormous computational power requirements, also known as the 'curse of dimensionality' (Daum, 2005). Hence, the particle filter may not be practical for real world history matching applications. The idea of adding a resampling step to the EnKF algorithm, after the Kalman update step, is a possibility. However, the relatively few production measurements in typical reservoirs can cause rejection of many model realizations in the early update time instants of the algorithm. This would result in the degeneracy of the ensemble of model realizations and an underestimation of the production uncertainty after a few sequential updates of the reservoir parameters.

In the presence of uncertain model parameters and unmeasured state variables, the history matching problem will have to be considered as a combined state and parameter estimation problem using measured data. For this, the state vector can be augmented with the uncertain parameter vector as follows:

$$\mathbf{z}_k \equiv \begin{bmatrix} \mathbf{x}_k \\ \boldsymbol{\theta}_k \end{bmatrix} = \begin{bmatrix} \mathbf{f}(\mathbf{x}_{k-1}, \boldsymbol{\theta}_{k-1}) \\ \boldsymbol{\theta}_{k-1} \end{bmatrix} \tag{6.4}$$

where the augmented state vector $\mathbf{z}_k$ has to be estimated using the measurements. Note that the augmented state vector will contain both the dynamic and static variables. For the static variables, it is a common practice to assume some slow artificial dynamics in the form of a random walk/Brownian motion (Moradkhani et al., 2005)

$$\boldsymbol{\theta}_k = \boldsymbol{\theta}_{k-1} + \mathbf{w}_{k-1} \tag{6.5}$$

where $\mathbf{w}_{k-1}$ is a zero-mean Gaussian random variable with a very low variance $\mathbf{Q}$. A similar additive Brownian random noise is usually assumed for the state transition function, to account for modeling errors also (Burgers et al. (1998), Evensen (2009)). However, in this work it was assumed that the modeling error is captured entirely through the additive error in the parameter vector. Based on the augmented state vector $\mathbf{z}_k$, with the artificial parameter dynamics defined in Eq. (6.5), the EnKF algorithm will be explained.

As mentioned earlier, the EnKF will use an ensemble of Monte Carlo samples to represent the uncertain variables. Assume that at time $k = 0$, we have a

multivariate probability distribution $p(\boldsymbol{\theta})$ which represents the prior knowledge about the parameter vector. In reservoir engineering, this distribution will be defined by the geostatistical models which define the geological uncertainty, i.e.,

$$p(\boldsymbol{\theta}) \equiv \text{Prior Geological Uncertainty}$$

The first step in the EnKF is to generate N_e samples from the prior probability distribution $p(\boldsymbol{\theta})$. Let $\boldsymbol{\theta}_{0|0} = \{\boldsymbol{\theta}^1_{0|0}, \boldsymbol{\theta}^2_{0|0}, ...\boldsymbol{\theta}^{N_e}_{0|0}\}$ represent the set of N_e samples of the parameters. Here it is assumed that the initial conditions of the reservoir is accurately known and given by $\mathbf{x}_0$. If the initial conditions are also uncertain, then a set of initial conditions representing this uncertainty should also be generated. Let the production data from the field at the date instance k be represented as $\mathbf{y}^{obs}_k$. Then, the EnKF algorithm sequentially updates the reservoir as follows:

1. Prediction step:

The prediction step is a forward simulation of the reservoir using each of the N_e parameter samples. The predicted ensemble is represented as

$$\widehat{Z}_{k|k-1} \equiv \{\mathbf{z}^1_{k|k-1}, \mathbf{z}^2_{k|k-1}, ..., \mathbf{z}^{N_e}_{k|k-1}\}$$

where, $\mathbf{z}^i_{k|k-1}$ is generated using Eq. (6.4), i.e., the reservoir simulator $\mathbf{f}$ with the parameter value defined by $\boldsymbol{\theta}^i_{k-1|k-1}$. Thus, we get

$$\mathbf{z}^i_{k|k-1} = \begin{bmatrix} \mathbf{f}(\mathbf{x}^i_{k-1|k-1}, \boldsymbol{\theta}^i_{k-1|k-1}) \\ \boldsymbol{\theta}^i_{k-1|k-1} + \mathbf{w}^i_{k-1} \end{bmatrix} \tag{6.6}$$

Let,

$$\boldsymbol{\mu}^z_{k|k-1} = \frac{1}{N_e} \sum_{i=1}^{N_e} \mathbf{z}^i_{k|k-1} \tag{6.7}$$

be the mean of the predicted ensemble. The notation '$k|k-1$' is used to indicate that the information from the previous time step $(k-1)$ was used to obtain the current value of the variable. In the EnKF algorithm, the error/uncertainty in the predicted states of the reservoir is defined around the predicted state ensemble mean as

$$e^i_{k|k-1} = \mathbf{z}^i_{k|k-1} - \boldsymbol{\mu}^z_{k|k-1} \tag{6.8}$$

Note that for each $\mathbf{z}^i_{k|k-1}$, we will simultaneously obtain an ensemble of *simulated production data* in any commercial reservoir simulator. Let,

$$\widehat{Y}_{k|k-1} \equiv \{\mathbf{y}^1_{k|k-1}, \mathbf{y}^2_{k|k-1}, ..., \mathbf{y}^{N_e}_{k|k-1}\}$$

be the ensemble of predicted production data. Similar to Eq. (6.8), the error/ uncertainty in the predicted measurements is defined around the predicted measurement ensemble mean as

$$\varepsilon^i_{k|k-1} = \mathbf{y}^i_{k|k-1} - \boldsymbol{\mu}^y_{k|k-1} \tag{6.9}$$

where,

$$\boldsymbol{\mu}^{y}_{k|k-1} = \frac{1}{N_e}\sum_{i=1}^{N_e}\mathbf{y}^{i}_{k|k-1} \tag{6.10}$$

The uncertainty in the predicted states and measurements are calculated as sample covariance of the error ensemble generated in Eqs. (6.8) and (6.9). Two covariance matrices are calculated:

$$\widehat{\mathbf{P}}^{\varepsilon,\varepsilon}_{k|k-1} = \frac{1}{N_e-1}\sum_{i=1}^{N_e}(\varepsilon^{i}_{k|k-1})(\varepsilon^{i}_{k|k-1})^{T} \tag{6.11}$$

$$\widehat{\mathbf{P}}^{e,\varepsilon}_{k|k-1} = \frac{1}{N_e-1}\sum_{i=1}^{N_e}(e^{i}_{k|k-1})(\varepsilon^{i}_{k|k-1})^{T} \tag{6.12}$$

The covariance $\widehat{\mathbf{P}}^{e,\varepsilon}_{k|k-1}$ is the cross-covariance between the state prediction ensemble and measurement model ensemble. In other words, it is measure of how strongly correlated is the production data with respect to the uncertainty in the reservoir states/parameters. The main assumption in the EnKF is that the distribution of the state prediction error and measurement error is zero-mean multivariate Gaussian.

2. Correction step:

The correction step is the sequential history matching step in the EnKF algorithm. In the correction step, each of the samples in the predicted ensemble is updated using the production measurement $\mathbf{y}^{obs}_{k}$ from the actual reservoir. While using the production data, it is required to treat them as random variable contaminated by noise (Burgers et al., 1998). Hence, a set of N_e perturbed measurements around $\mathbf{y}^{obs}_{k}$ are generated using random samples from the measurement noise distribution defined in Eq. (6.3) ($\mathbf{v}_k \sim N(0,\mathbf{R})$), as follows:

$$\left\{\mathbf{y}^{i,obs}_{k} = \mathbf{y}^{obs}_{k} + \mathbf{v}^{i}_{k}\right\}_{\text{for } i=1 \text{ to } N_e} \tag{6.13}$$

The perturbed observations are used to update the reservoir states and the unknown parameters using the well-known Kalman update equation

$$\left\{\mathbf{z}^{i}_{k|k} = \mathbf{z}^{i}_{k|k-1} + \mathbf{K}_{gain}(\mathbf{y}^{i,obs}_{k} - \mathbf{y}^{i}_{k|k-1})\right\}_{\text{for } i=1 \text{ to } N_e} \tag{6.14}$$

Note that the subscript notation '$k|k$' is used to indicate that the information from observations at the current time instant k is assimilated into the updated variable. The Kalman gain $\mathbf{K}_{gain}$ is calculated using the sample covariance matrices in Eqs. (6.11) and (6.12), as follows:

$$\mathbf{K}_{gain} = \widehat{\mathbf{P}}^{e,\varepsilon}_{k|k-1}(\widehat{\mathbf{P}}^{\varepsilon,\varepsilon}_{k|k-1} + \mathbf{R})^{-1} \tag{6.15}$$

The Kalman update is applied to each of the predicted state ensemble to obtain the *corrected ensemble*, i.e., the history matched ensemble of reservoir states

and parameters represented as $\widehat{Z}_{k|k} \equiv \{\mathbf{z}^1_{k|k}, \mathbf{z}^2_{k|k}, ..., \mathbf{z}^{N_e}_{k|k}\}$. The history matched ensemble so obtained will be propagated forward through the prediction step until the next history matching date, then the update procedure is repeated, and so on. This sequential prediction and correction procedure is continued and stopped when the end of the period selected for history matching is reached. To our knowledge, selection of the optimal stopping point for the sequential history matching problem is not investigated in the current literature. A heuristic method, though computationally expensive, would be to evaluate the predictive accuracy of the reservoir model by restarting the simulation from $k = 0$ using the most currently updated parameter ensemble.

Note that in the update step, the EnKF adjusts the predicted ensemble by a value proportional to the error in the predicted measurement $(\mathbf{y}^{i,obs}_k - \mathbf{y}^i_{k|k-1})$, i.e.,

$$(\mathbf{z}^i_{k|k} - \mathbf{z}^i_{k|k-1}) = \mathbf{K}_{gain}(\mathbf{y}^{i,obs}_k - \mathbf{y}^i_{k|k-1}) \tag{6.16}$$

The Kalman gain applies a weighted correction where the weighting is roughly given by the ratio of $\widehat{\mathbf{P}}^{e,\varepsilon}_{k|k-1}$ to $(\widehat{\mathbf{P}}^{\varepsilon,\varepsilon}_{k|k-1} + \mathbf{R})^{-1}$. If $\widehat{\mathbf{P}}^{e,\varepsilon}_{k|k-1}$ is relatively high compared to $\widehat{\mathbf{P}}^{\varepsilon,\varepsilon}_{k|k-1}$, it would signify that (a) there is a high correlation between the measurement error and the estimation error, and (b) the measurement uncertainty is low. In that case, the Kalman gain will be high and the correction made to the reservoir states/parameters will be very sensitive to $(\mathbf{y}^{i,obs}_k - \mathbf{y}^i_{k|k-1})$. Thus, the Kalman gain in the EnKF can be interpreted similar to that of the classical Kalman filter for linear systems.

The general workflow of history matching using the EnKF algorithm is illustrated in Figure 6.1. As shown in this figure, prior knowledge about the reservoir geostatistics will be used to 'bootstrap' the first ensemble of reservoir model parameters. This initial ensemble of reservoir models will be used in the prediction step, which is essentially a one-step forward Monte Carlo simulation. This will be followed by the correction step, in which the predictions so obtained will be compared with the production measurements from the actual reservoir and an updated ensemble of reservoir models will be obtained. Subsequently, the updated ensemble will be used for predictions until the next scheduled date of history matching. A suitable post-processing step can be included after the update step to constrain the parameters, for example to honor the geologic realism of the reservoir. If no further update of the reservoir parameters is desired, i.e., no further history matching has to be carried out, then the most recent set of updated reservoir models can be used for making forecasts of the reservoir production.

6.2.1 EnKF with confirming option

One of the drawbacks of the conventional EnKF algorithm when applied to history matching is that the update step in Eq. (6.14) can cause the reservoir states to

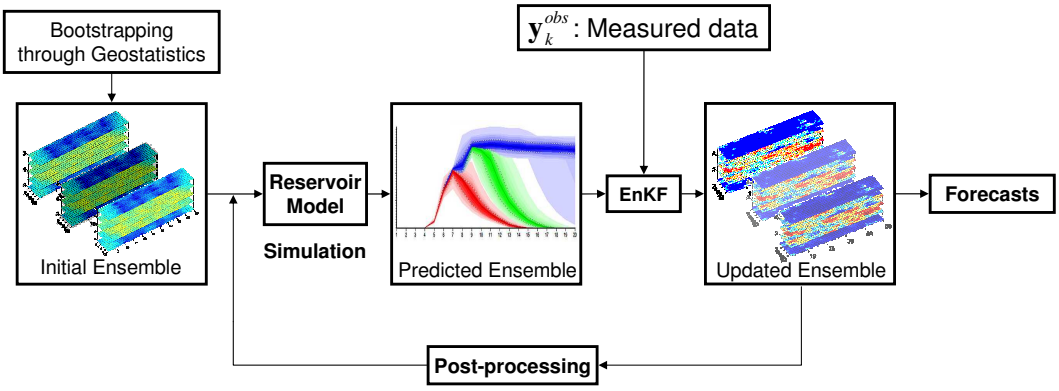


Figure 6.1: Block diagram to illustrate the general workflow of petroleum reservoir history matching using the Ensemble Kalman filter.

be unrealistic after the history matching. This is because the reservoir simulation equations are not explicitly used in any of the calculations of the update step. For example, the overall mass balance of the reservoir can be totally violated if the reservoir states are directly updated using the Kalman update rule. Moreover, due to the nonlinear nature of the combined state and parameter estimation problem, the updated reservoir states may not have a consistent relation to the updated parameters, i.e., if the reservoir is re-simulated with the updated parameters and the initial conditions, we may not get back the same updated states. In order to overcome this issue, it was suggested to use the Kalman update only on the parameters and indirectly update the states by re-simulating the reservoir starting from the previous history matched date (Wen & Chen, 2006). Following this suggestion, the EnKF algorithm with the confirming option was used in this work. With the confirming option, we can separate out the parameter vector from the augmented state to calculate the state prediction error and the Kalman gain, i.e.,

$$e^i_{k|k-1} = \boldsymbol{\theta}^i_{k|k-1} - \boldsymbol{\mu}^{\theta}_{k|k-1} \tag{6.17}$$

The update step can be divided into 2 sub-calculations:

2(a) Parameter update: Update the parameters using the Kalman update:

$$\left\{\boldsymbol{\theta}^i_{k|k} = \boldsymbol{\theta}^i_{k|k-1} + \mathbf{K}_{gain}(\mathbf{y}^{i,obs}_k - \mathbf{y}^i_{k|k-1})\right\}_{\text{for } i=1 \text{ to } N_e} \tag{6.18}$$

2(b) Confirming step (State update): Re-simulate the reservoir starting from the previous history matched date, but using the newly updated parameters:

$$\left\{\mathbf{x}^i_{k|k} = \mathbf{f}(\mathbf{x}^i_{k-1|k-1}, \boldsymbol{\theta}^i_{k|k}\right\}_{\text{for } i=1 \text{ to } N_e} \tag{6.19}$$

By performing the state update using the reservoir simulator model $\mathbf{f}$, we can ensure that all the updated reservoir states are consistent with the updated model parameters and also stay within realistic domains. The assumption is that the

uncertainty in the reservoir model is solely manifested through the parameter uncertainty. The EnKF algorithm is essentially used as a parameter estimation algorithm which is the main goal in history matching.

6.2.2 Localized EnKF

In large scale systems, the observations are scattered at various locations on the entire grid volume. For example, the production wells in a reservoir will be separated from each other by several grid blocks. We can assume that the heterogeneous geological properties at the grid block locations will only influence the production data of nearby wells. In that case, we can perform the update step in a piecewise fashion, where only the measurements from near-by wells will be used to update a particular grid block property. This is called localization technique in the EnKF literature. Later in this chapter, the application of the localized EnKF on a highly heterogeneous 2-D reservoir will be demonstrated. In order to do this, we have to first define the region of influence of a particular well *a priori*. One of the advantages of this localized analysis scheme is that it will reduce the impact of spurious correlations from measurements that are insensitive to a particular parameter. In addition, it is also expected that a lesser number of ensembles will be required than would be required with a global EnKF, especially for large scale systems. The localized EnKF described here is one of the several variants available in literature. Several types of the localization schemes and their respective advantages have been reviewed by Evensen (2009).

6.2.3 Geostatistical prior

Several authors have used geostatistical models of permeability and porosity to generate the initial ensembles (Naevdal et al. (2005), Wen & Chen (2007), Haugen et al. (2008), Zhang & Oliver (2009)). We consider this as a very important step for history matching using the EnKF algorithm. In all Bayesian estimation techniques, it is well known that the prior information is very valuable to inference the posterior. In the EnKF algorithm, the initial ensembles represent the prior information we have about the distribution of the static variables. The petroleum engineering literature is rich with techniques such as Kriging algorithms which can estimate the static variables at unsampled locations given hard data measured at well locations. Also, there are sophisticated techniques such as the sequential Gaussian simulation (Deutsch & Journel, 1998) which can perform conditional simulation to generate multiple plausible realizations of the static parameters. The geostatistical realizations obtained from such simulations can be conditioned to honor the histogram and spatial correlations of the random fields. However, the uncertainty in the predictions from these realizations can be very high if directly

used for simulating the reservoir. This is evident from the fact that the production data is not used to condition these simulations. The EnKF is a natural way forward to assimilate the production data from the wells into the geostatistical realizations. It is expected that by using geologically realistic covariance models to generate the initial ensemble, the EnKF algorithm will yield history matched results which are more realistic with respect to the geological heterogeneity of the reservoir.

6.2.4 Choice of observation variables

One of the critical factors which can affect the quality of any parameter estimation algorithm is the amount of information about the parameters that the data (vector $\mathbf{y}_k^{obs}$) carries. Hence, an important pre-processing step is to choose the production variables which contain meaningful information about the geostatistical properties. Usually production variables such as monthly production rate of oil/gas/water, steam-oil/gas-oil/gas-water ratios etc. are assumed to be sensitive to the reservoir permeabilities. However, the degree of sensitivity can vary from case-to-case. For example, the Bottom-Hole-Pressures may remain constant and unaffected by the permeability in a reservoir simulation model. Hence, a careful choice of production variables based on a sensitivity analysis can be expected to complement the history matching results. Sensitivity can be analyzed directly through Monte Carlo simulations of the reservoir model. The idea is to use random realizations of the static parameters from the geostatistical prior and check for variability in the simulated production variables. Higher variability in the simulations can be interpreted as higher sensitivity.

Another important guideline is to avoid redundancy in the data used for identifying the parameters. Note that the EnKF algorithm will fuse the information from different variables in a multivariate fashion through the correction step in Eq. (6.18). If two variables in $\mathbf{y}_k^{obs}$ contain redundant information, then it can cause spurious update of the parameters in the correction step. For example, the Monthly Production of Oil (bbl/month) will be highly correlated to the average Monthly Oil rate (bbl/day). If both these variables are part of the vector $\mathbf{y}_k^{obs}$, then their respective prediction errors will cause redundant update of the parameters. Hence, only one of these variables should be chosen in the objective for history matching. On the other hand, variables such as production rate of oil and that of gas will usually contain diverse information about the static properties and hence both these variables can be part of the vector $\mathbf{y}_k^{obs}$.

6.2.5 Choice of Q and R

For history matching applications, the favorable values for parameter noise variance $\mathbf{Q}$ usually fall in the range of 1e-4 to 1e-6. The low noise variance signifies that

the parameters are static properties such as ln(permeability) and porosity. A low variance will also ensure that the parameters are not perturbed too far away from the span of the updated ensembles from the previous time instant. The noise variance can be chosen to be spatially varying to represent the varying level of uncertainty at different spatial locations (E.g.: variance resulting from Kriging estimation). The matrix $\mathbf{R}$ should be used as a tuning parameter in the EnKF algorithm and their favorable values differ from case-to-case. Note that the Kalman gain has an inverse relationship with respect to the measurement noise matrix $\mathbf{R}$. Hence, the EnKF update can be expected to be more sensitive to those measurement variables for which the measurement noise is lower. A good choice for the measurement noise will require some experience of applying the EnKF on the particular reservoir model.

6.2.6 Quality of history match

In order to measure the quality of the ensemble and rank the realizations after history matching, a weighted mean square error for the ith realization in the ensemble is defined, as follows:

$$WMSE^i = \frac{1}{N} \sum_{k=t_1}^{t_N} (\mathbf{y}_k^{obs} - \widehat{\mathbf{y}}_k^i)^T \mathbf{R}^{-1} (\mathbf{y}_k^{obs} - \widehat{\mathbf{y}}_k^i) \tag{6.20}$$

Here, the variables $\mathbf{y}_k^{obs}$, $\widehat{\mathbf{y}}_k^i$ are assumed to be column vectors containing the production variables at time instant k. The vector $\widehat{\mathbf{y}}_k^i$ denotes the predictions from the ith model in the history matched ensemble of models. The measurement noise variance matrix applies a weighting that is inversely proportional to the amount of noise in the production variables. Since the EnKF was used as a parameter estimation algorithm, all predictions are based on simulations starting from the initial conditions. However, there are several options to choose the time period for measuring the history match performance (t_1 need not necessarily be at the initial conditions). The history match quality can be evaluated on data within the period of history matching. If the predictive capability of the model has to be evaluated, then a more reliable model validation procedure is to cross-validate against data falling outside the period of history matching. For the latter approach, it is required to divide the past production data into *estimation* (i.e. history matching) and *validation* data set (Ljung, 1999). The disadvantage would be that we cannot use all the available historical data for the model building exercise.

In order to evaluate the overall quality of history matching, a normalized measure of fit is defined which can be expressed in percentage scale. Note that the *WMSE* quality measure defined earlier will be dependent on the units of the production variables and the magnitude of measurement noise covariance matrix. A scale-independent measure for evaluating the overall quality of history matching will be

the commonly used *R-square* value of fit (Ljung, 1999), which is defined as

$$R^2_{i,j} = 1 - \frac{\sum_{k=t_1}^{t_N} (y_k^{obs_j} - \widehat{y}_k^{i,j})^2}{\sum_{k=t_1}^{t_N} (y_k^{obs_j} - \overline{y}^{obs_j})^2} \tag{6.21}$$

In Eq. (6.21), the R^2 value for the ith realization in the history matched ensemble and jth production variable is defined. The scalar $\overline{y}^{obs_j}$ in the denominator refers to the time average of the jth production variable, averaged over the time interval $k = t_1 \; to \; t_N$. The aggregate quality of the ith realization can be defined by averaging over all the j production variables, i.e.,

$$R^2_i = \frac{1}{N_{prod}} \sum_{j=1}^{N_{prod}} R^2_{i,j} \tag{6.22}$$

where, N_{prod} is the number of production variables selected for history matching. A value of $R^2 = 1$ should be interpreted as perfect fit between the observed data and the model predictions; a value close to 0 would mean that the predictions are just as good as the historical mean of production data (Note: Negative values would mean that the predictions are worse than forecasting just the average of the past data). The advantage of the R^2 value is that it is always guaranteed to be less than 1 and can be used as a scale-independent overall measure of history match quality. The percentage quality can be obtained by multiplying Eq. (6.22) by 100 %.

6.3 Case Studies on Heterogeneous Reservoirs

The methodology described in the previous section was applied to characterize and history match two synthetic heterogeneous reservoirs. The entire EnKF workflow was developed in the MATLAB computational environment by interfacing it with a commercial reservoir simulator. The IMEX and STARS modules of the CMG reservoir simulation software[3] were used to simulate the synthetic reservoir models. In the two case studies presented here, the reservoir model grid blocks were first populated with some known permeability values which are assumed as the truth case. The production data simulated from the truth case is treated as data available from the field for history matching. The truth case permeability is completely hidden from the reservoir model used in the EnKF workflow, except for the grid blocks with core hole measurements (hard data). In the case studies presented in this work, the algorithms available in the MATLAB geostatistical toolbox 'mGstat' (Hansen, 2009) was used to generate the initial ensembles conditioned on the hard data. Specifically, the mGstat toolbox interfaces to the geostatistical simulation packages

[3]Software developed by Computer Modeling Group Ltd., Calgary, AB, Canada

VISIM (Hansen & Mosegaard, 2008) and SGeMS (Remy et al., 2009) were utilized. The objective is to estimate all the grid block permeabilities using the production data from the truth case. Since ln(permeability) is usually assumed to follow a Gaussian distribution, all the EnKF calculations are based on the log-transformed values of the permeability, i.e., in Eqs. (6.17) and (6.18) we have

$$\boldsymbol{\theta}_{.|.} = \ln(\text{permeability})$$

For the reservoir simulator $\mathbf{f}$ in Eq. (6.19), the log-permeabilities were transformed back into the actual permeability values. Also, it was assumed that prior information about the upper and lower limit of the grid block permeability is available. These limits were used to constrain the permeability values at all times during the history matching process.

6.3.1 Heterogeneous black-oil reservoir (2-D case study)

The first case study was performed on a benchmark data set from the 2001 SPE Comparative solution project (Christie & Blunt, 2001). The model has a simple 2-dimensional vertical geometry, containing oil and gas (2-phase), modeled using a black-oil formulation. The fine-scale grid of 100x1x20 was used. The reservoir was assumed to be fully saturated with oil initially. In the original model, there was one injector (injecting gas) and one producer well. In this case study, this was modified with one injector well at the center of the reservoir, with two producer wells (Pro-1 & Pro-2) placed symmetrically on either side of the injector. The grid block permeability values in millidarcies (md) in the I direction (2000 values) in the original benchmark data set are shown in Figure 6.2(a). The J and K direction permeabilities were assumed to be equal to PERM-I. From the figure, this reservoir model can be observed to be highly heterogeneous with high permeability grids surrounded by extremely low permeability layers. The objective was to estimate all the 2000 grid block permeabilities using the well production data from the truth case. Figure 6.2(a) also shows the modified well placements. In addition to this, it was also assumed that two vertical core holes are drilled symmetrically in between the producer and injector wells (at i = 25, 75) to obtain hard data along these grid blocks. Figure 6.2(b) highlights the locations of the core holes C-1 and C-2. It was assumed that the grid block permeabilities along Pro-1, Pro-2, C-1, and C-2 are initially known (highlighted by the black lines in Figure 6.2(b)).

The regions 1 and 2 shown in Figure 6.2(b) show the localization regions. The region 1 permeability was updated using the production data from well Pro-1 only and the region 2 was updated using Pro-2 only. The remaining part of the reservoir (region between C-1 and C-2) was updated using both Pro-1 and Pro-2. The production data used for history matching consisted of the Monthly Gas and Oil

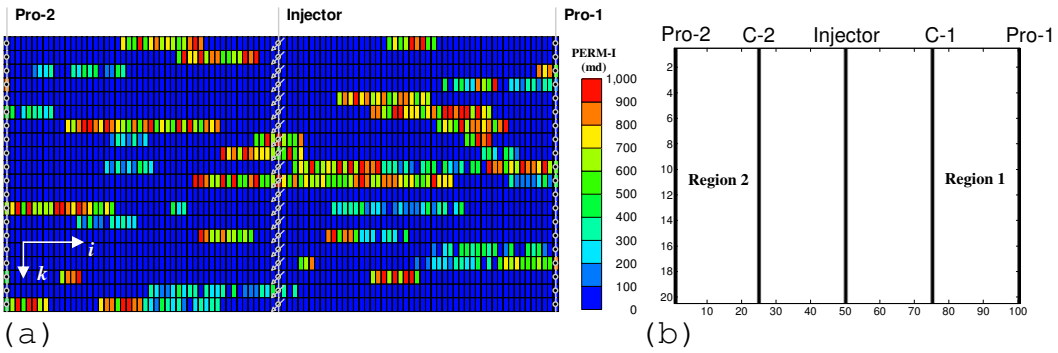


Figure 6.2: (a) Heterogeneous Black-oil reservoir (2-D) with one injector and two producers. (b) Permeability hard data locations.

productions (ft^3) for the two wells. Thus, the $\mathbf{y}_k^{obs}$ vector will be given by

$$\mathbf{y}_k^{obs} = \begin{bmatrix} \text{Gas Production of Pro-1} \\ \text{Gas Production of Pro-2} \\ \text{Oil Production of Pro-1} \\ \text{Oil Production of Pro-2} \end{bmatrix}_{k^{th} \text{ month}}$$

The measurement noise variance for Gas and Oil productions were chosen as 5 ft^3 and 10 ft^3 respectively. The parameter noise variance was chosen as 1e-4 at all grid block locations. These values were arrived through a tuning exercise to obtain a good history matched result. The fit of the history matched ensemble predictions to the production data was visually analyzed to tune these values.

The first step in the EnKF workflow is to generate the initial ensemble (ensemble size = 30 realizations) using prior knowledge of the geostatistics. Thirty realizations representing the initial ensemble were generated using a direct sequential simulation in the VISIM package, the simulation being conditioned on the hard data at the locations shown in Figure 6.2(b). An anisotropic spherical variogram function with a maximum correlation range of 75 grid blocks in the east-west direction was used in the conditional simulations. Figure 6.3(b) shows the E-type mean of the initial ensemble. As can be observed from the figure, the initial ensemble mean is very homogeneous because very few measurements of the static data is available. The localized EnKF algorithm was implemented to history match the production data falling in the period of August, 2001–March, 2013. The update step was carried out sequentially at 8 consecutive time points in this period. Counting the days from the beginning of simulation, the update was done sequentially after 250 (August, 2001), 400, 500, 1000, 1750, 2500, 3500, and 4500 (March, 2013) days respectively. The history matched ensemble mean is shown in Figure 6.3(c). It can be observed clearly that the mean permeability of the history matched ensemble shows a closer match to the true reservoir heterogeneity when compared to the initial ensemble.

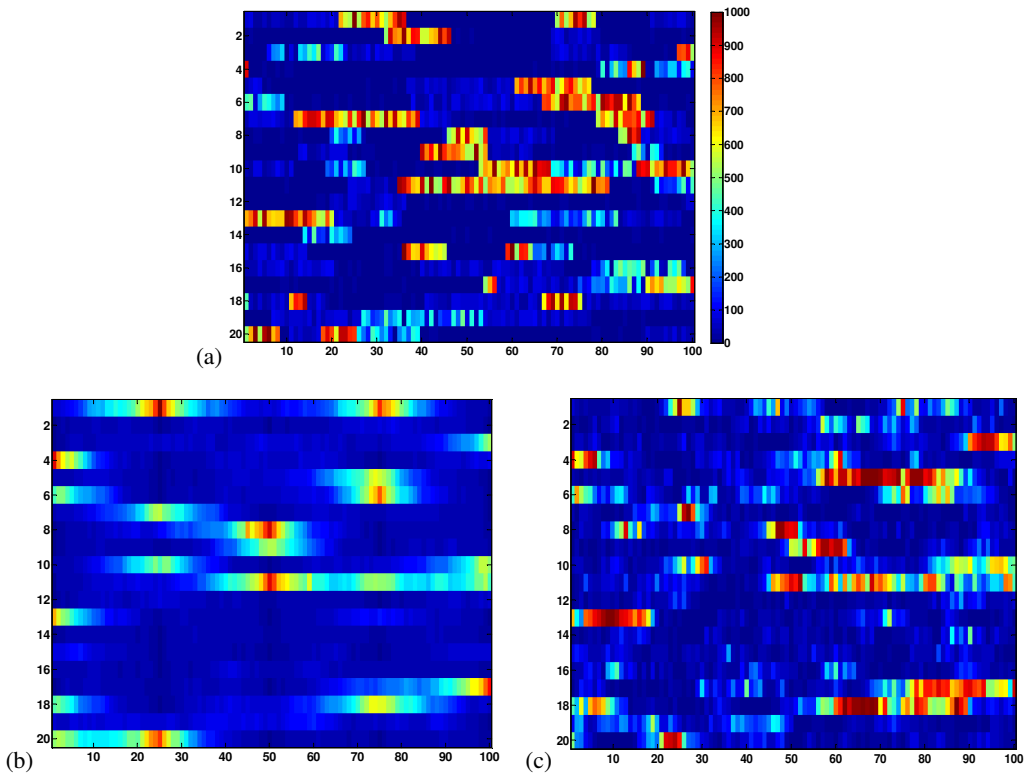


Figure 6.3: Grid block permeabilities of the 2-D heterogeneous reservoir showing (a) True case (SPE benchmark data set), (b) E-type mean of initial ensemble realizations (30) generated using a geostatistical model, and (c) E-type mean of history matched ensemble resulting from the localized EnKF algorithm.

Figures 6.4(a)–(d) show the comparison of ensemble predictions of the production data, *before* and *after* history matching. All the predictions (before and after) shown in the figure were generated through simulations starting from the initial conditions of the reservoir. The predictions after history matching were obtained through simulations initialized with the ensemble of permeabilities resulting from the last update of the EnKF algorithm (March, 2013). Clearly, the EnKF algorithm is able to estimate permeabilities that result in predictions matching the reservoir history more closely compared to the initial ensemble realizations. Note that the initial/prior ensembles were based on the prior geostatistical model only. However, the history matched ensembles encompass the information assimilated through the EnKF update step and hence yield predictions that closely track the true reservoir production. The error bars clearly indicate the reduction in uncertainty in terms of the smaller inter-quartile range of the realizations. There is also a significant reduction in the variance of the model predictions after the history matching. This is a result of the EnKF algorithm correction step, where each ensemble is updated by taking into account the error in the production data predictions.

The quality of the ensemble realizations in terms of their WMSE and R-square

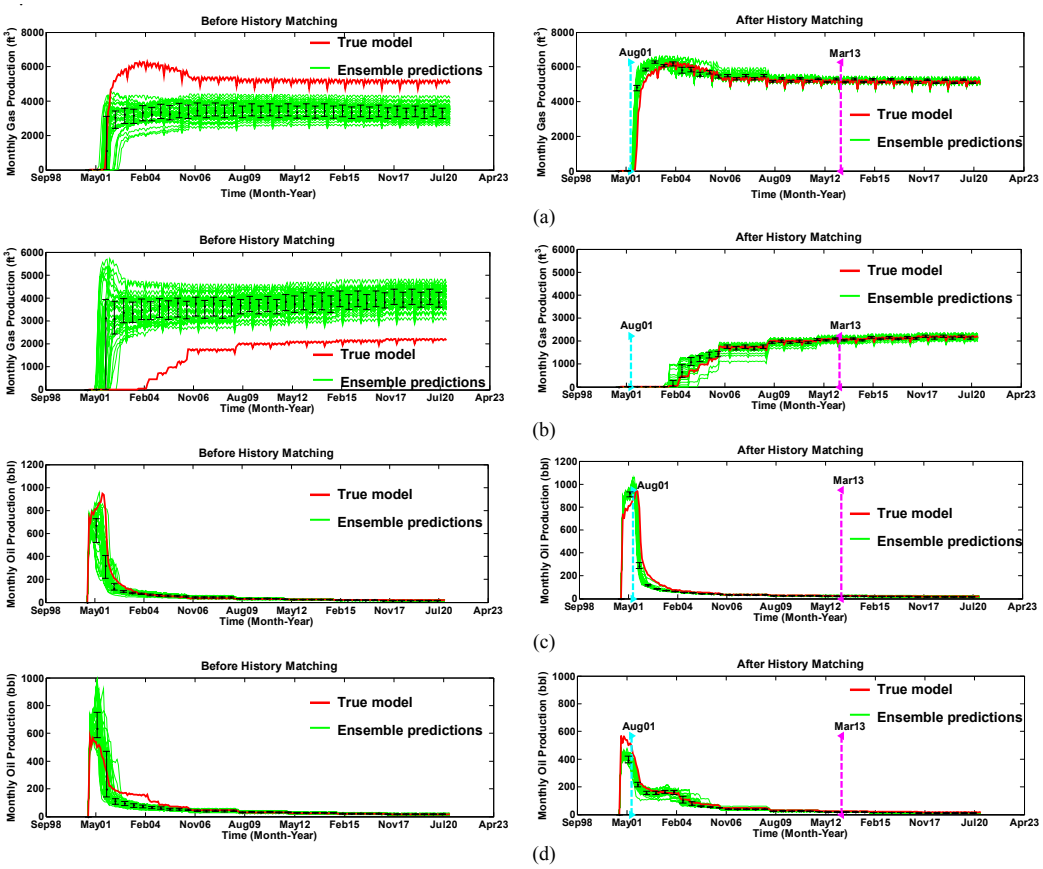
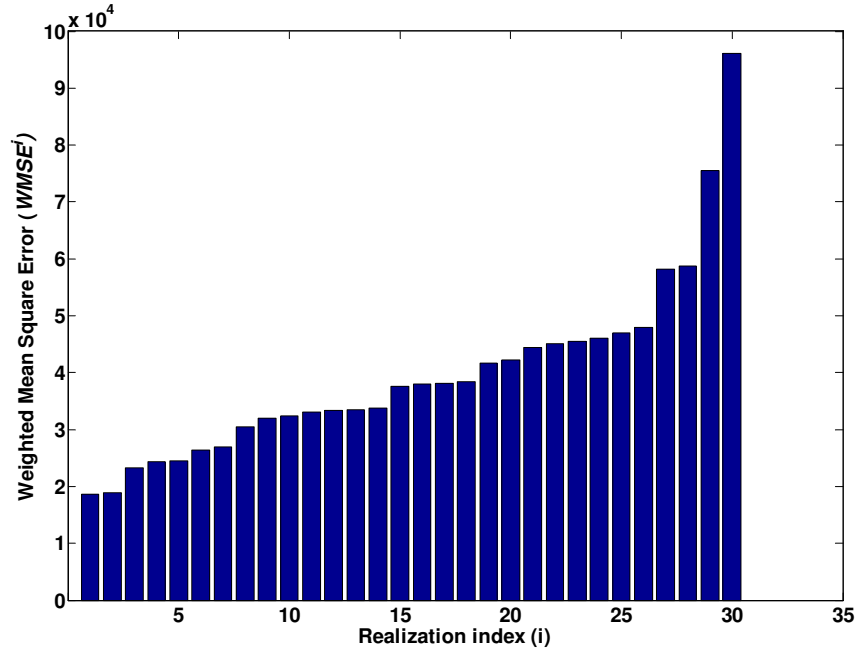
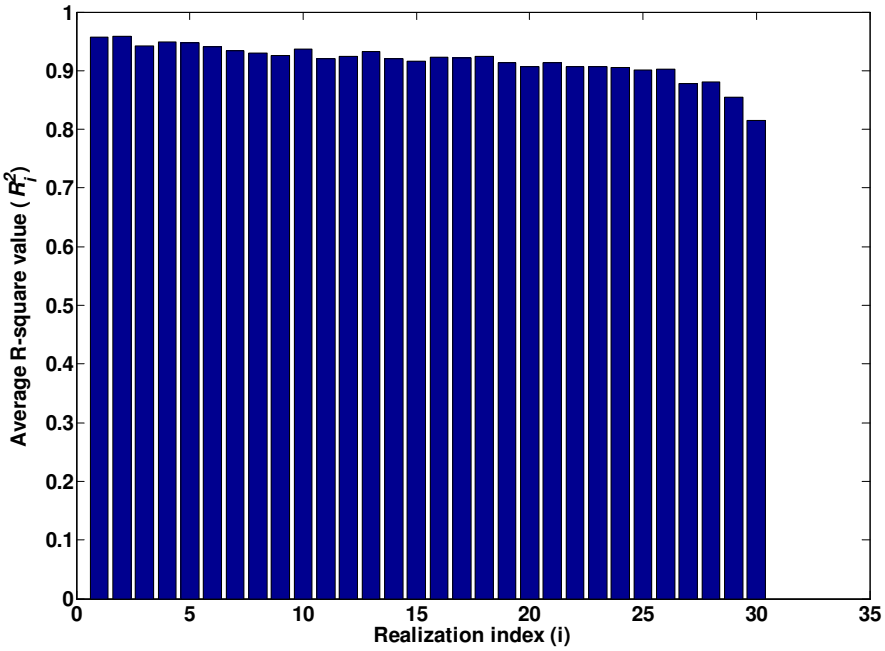


Figure 6.4: Comparison of ensemble predictions (30 realizations) *before* and *after* localized EnKF history matching for (a) Well Pro-1 gas production, (b) Well Pro-2 gas production, (c) Well Pro-1 oil production, and (d) Well Pro-2 oil production. The starting and ending period of data used for history matching are flagged using the cyan and magenta lines. Error bars indicate the inter-quartile range of the ensemble predictions (1st and 3rd quartile of the ensemble).



(a)



(b)

Figure 6.5: Quality of the ensembles (30 realizations) *after* localized EnKF history matching in terms of (a) WMSE: The ensembles are sorted in the ascending order of the WMSE and those with relatively poor WMSE can be discarded if necessary, and (b) R-square value: This will indicate the overall quality of the EnKF history match. For a reliable history match run, all the realizations will show high R^2 values.

values are shown in Figure 6.5. The realizations are sorted in the increasing order of their respective WMSE. From Figure 6.5(a), we can clearly see that there is a sharp decline in quality after the 26th ensemble realization. The high sensitivity of the WMSE plot makes it a good tool for ranking the ensemble realizations in terms of their prediction accuracy. On the other hand, the R-square error is less sensitive to the errors in the individual realizations and is a better indicator of the overall quality of the history match run. As shown in Figure 6.5(b), the R square value of most of the realizations is close to 0.9 indicating 90% fit between the true reservoir data and the ensemble predictions. In order to evaluate the advantage of localization in this case study, a global EnKF algorithm was also implemented where all the well measurements were used to update all the grid block permeabilities, irrespective of the grid block locations. Though there was no significant improvement in the quality of production data match, it was observed that the localized scheme yielded permeability values that had lower error compared to the truth case. This is depicted in Figure 6.6 where the root mean square error (RMSE) in the estimated permeability for the global and localized EnKF are shown for the 30 realizations in the ensemble. Note that the true reservoir permeability is known *a priori* for this synthetic example. From the figure we can observe that the RMSE in the estimated permeability of the localized EnKF is consistently lower than that of the global EnKF algorithm.

Figure 6.7 shows the evolution of the ensemble mean and standard deviation as the EnKF progressively updates the reservoir. Note that there is a significant change in the mean permeability of the ensembles from k = 1 to k = 4. This is an indication of significant correction applied to the permeability ensembles by the Kalman update step, to compensate for prediction errors with respect to the true reservoir production. We can also see that the standard deviation progressively decreases with each update step, which is a manifestation of the reduction in the uncertainty of the Kalman update technique (variance minimization). Beyond k = 5, the variance remains at the low range; this is an indication that the EnKF algorithm has converged and further updates will not bring significant decrease in the variance. The zero-variance along core hole locations signify that no update is performed at these locations because the true permeability is measured/available in the form of hard data along these grid blocks. Figure 6.8 shows the evolution of the aggregate mean of the E-type standard deviation, the averaging being done over the entire reservoir at each update time step. The reduction in the overall variance of the permeability ensemble with the progression of the Kalman updates is reiterated in this figure.

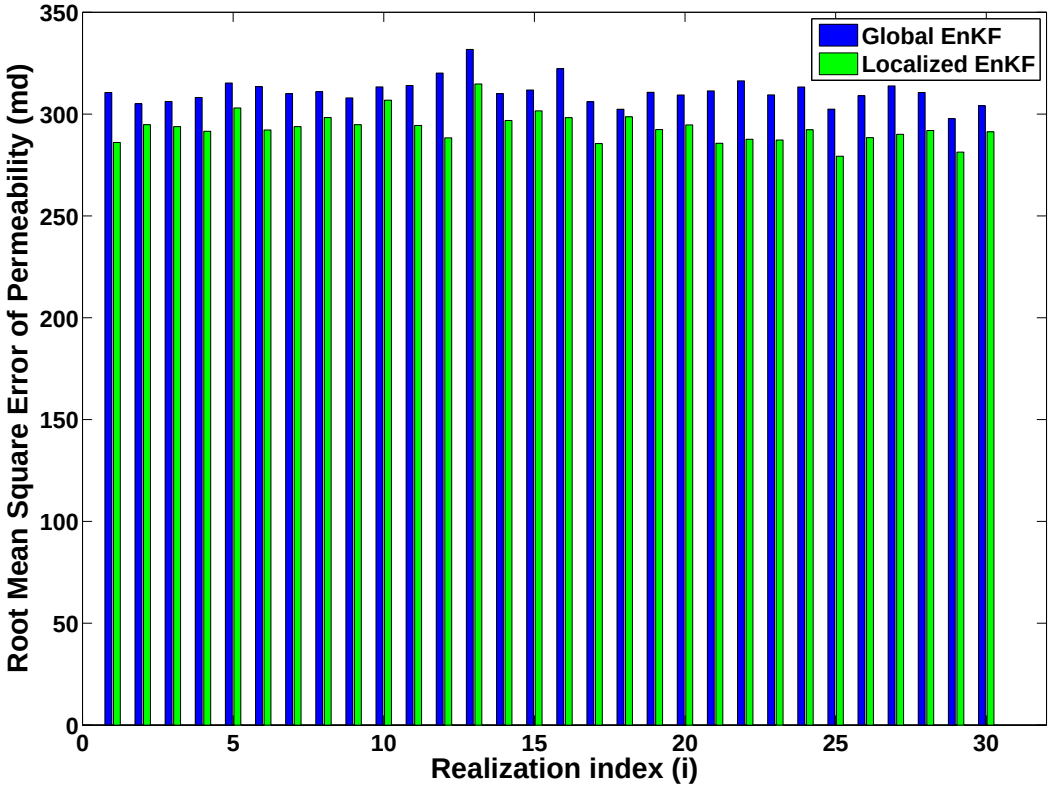


Figure 6.6: Comparison of the Root Mean Square Error of estimated permeability of localized and global EnKF algorithms.

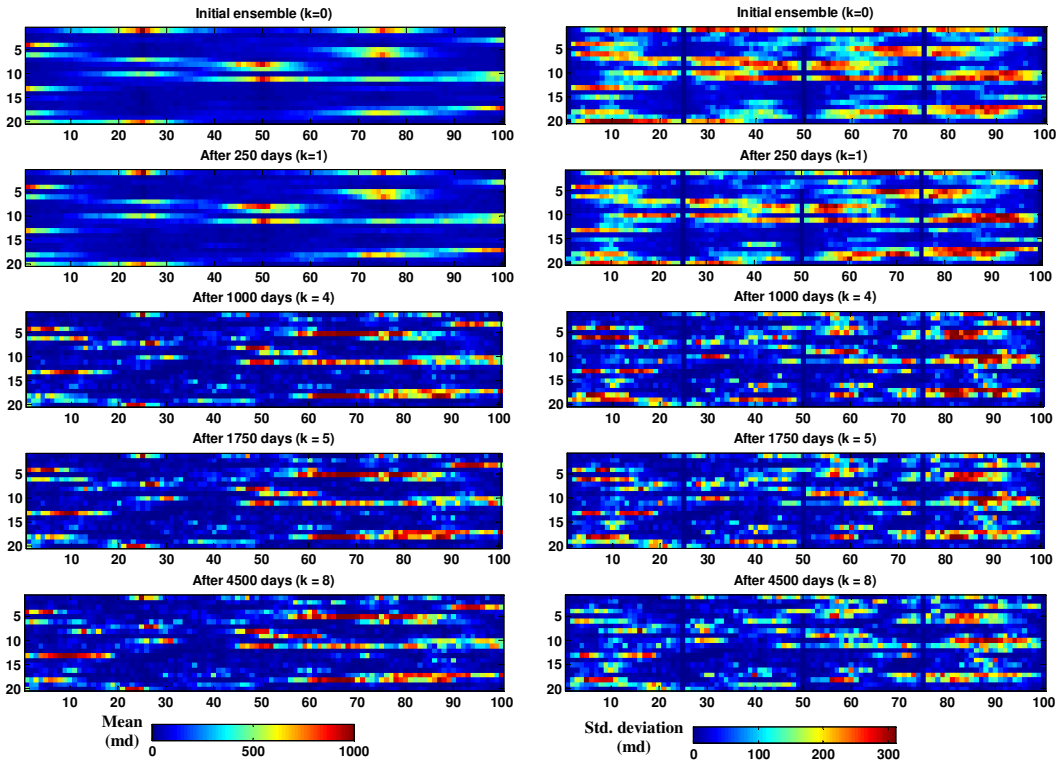


Figure 6.7: Evolution of E-type mean (*on the left*) and standard deviation (*on the right*) of the permeability ensembles with the progression of localized EnKF update steps.

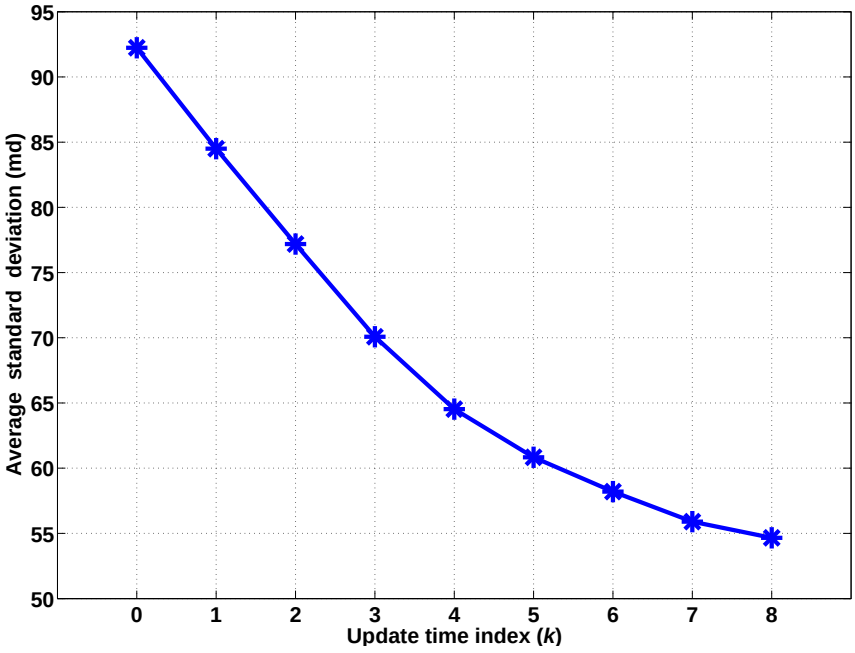


Figure 6.8: Evolution of the aggregate mean of permeability E-type standard deviation with the progression of localized EnKF update steps.

6.3.2 SAGD reservoir (3-D case study)

The second case study was performed on a synthetic 3-dimensional Steam Assisted Gravity Drainage (SAGD) reservoir with 80% initial heavy oil saturation. The dimensions of the reservoir were 5000x1000x150 ft, modeled using a coarse scale grid of size 100x100x30 ft in the respective dimensions (Number of grids = 50x10x5). Two pairs of Injector-Producer wells of length 30 grid blocks (length = 3000 ft) were assumed as shown in Figure 6.9(a). The horizontal trajectory of the injector wells (INJ-1, INJ-2) in layer 3, and that of the producer wells (P-1, P-1) in layer 4 are also shown in Figure 6.9(b). Figure 6.10(a) shows the spatial permeability distribution of the truth case, generated through a geostatistical simulation using the SGeMS package by assuming a geostatistical variogram model. The sequential Gaussian simulation (sgsim) technique was chosen as the simulation method and ln(permeability) was the simulated property. The overall variance of the random field (variogram sill value) was set at 1. An anisotropic spherical variogram model, with a maximum correlation range of 15 grid blocks was used. Steam at 650^0F was injected through the injection wells and 10 years of the reservoir production was simulated to generate the truth case production history. The producer wells were initially shut-in for a period of 1 year of steam injection, and opened for production from the 2^{nd} year onwards. The simulated data was used for estimating the grid block permeabilities using the EnKF, by assuming that the true grid-block permeability is not known.

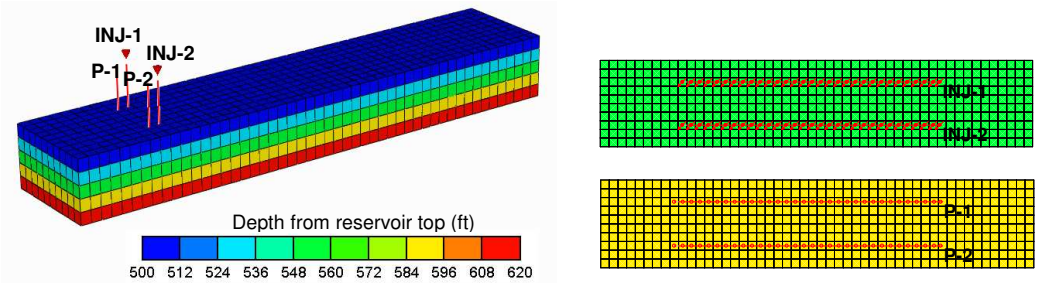


Figure 6.9: (a) Synthetic SAGD reservoir of size 50x10x5 containing two Injector-Producer well pairs. (b) Locations of the Injector wells (INJ-1, INJ-2) and Producer wells (P-1, P-2) in the 3rd and 4th layer of the reservoir respectively.

The production performance of the SAGD process can be evaluated using two production variables which are the oil production rate (OP_{rate}) and the cumulative steam-to-oil ratio (SOR). The geological heterogeneity will affect these production variables (McLennan & Deutsch, 2010) and hence they were used as the criteria for history matching in this case study. Thus, the measurement vector in the EnKF algorithm consists of the total OP_{rate} and the cumulative SOR for the entire field,

given by

$$\mathbf{y}_k^{obs} = \left[\begin{array}{c} \text{Cumulative SOR of entire field} \\ \text{OP}_{rate} \text{ of entire field} \end{array} \right]_{k^{th} \text{ month}}$$

A measurement noise variance of 1e-4 and 10 were assigned for SOR and OP_{rate} respectively in the EnKF algorithm. The permeability perturbation noise variance was chosen as 1e-4 at all grid block locations. Similar to the previous case study, it was assumed that hard data samples of the true permeability is initially available at a few core hole locations. Vertical core holes traversing the five layers were assumed and their spatial locations are as shown in Figure 6.10(b).

A conditional simulation, using the hard data as conditioning data and sgsim as the simulation algorithm, was performed to generate the initial ensemble realizations (ensemble size = 30). Figure 6.11(a) shows the E-type mean of the initial ensemble realizations. The variogram function of the ln(permeability) random field used for the initial ensemble generation was intentionally made different from the truth case. A variance of 2 and correlation range of 10 grid blocks were used, instead of the corresponding values of 1 and 15 in the true reservoir. The error in these variogram function parameters is a manifestation of the uncertainty/lack of exact knowledge about the reservoir geology which is the main motivation for characterizing the reservoir by assimilating the production data using the EnKF algorithm.

In this case study, an additional pre-processing of the initial ensembles was performed before executing the EnKF algorithm. This was necessary to account for the fact that the SAGD production variables are more sensitive to layers close to the well pairs (layers 2, 3 and 4). The variance of the permeability of the grid blocks in layer 1 and layer 5 after the sequential Gaussian simulation was extremely high (around 600 md). The pre-processing was done in a localized manner in layer 1 and layer 5 only in order to reduce their variance among the ensembles (E-type variance). The E-type variance of the layer 1 and layer 5 was reduced by replacing the geostatistical simulations with the E-type mean of these two layers, with the addition of a Gaussian random noise of standard deviation 50 md. Note that the Kalman update is more sensitive to those regions for which the cross-covariance $\widehat{\mathbf{P}}_{k|k-1}^{e,\varepsilon}$ is high. Without this pre-processing step, the high variance of layer 1 and 5 permeabilities was found to induce spurious correlations between the ensemble production data and these layers, which eventually resulted in false update of the permeabilities in these layers.

The history matching was performed for a period of 3 years and the production data in the $\mathbf{y}_k^{obs}$ vector was assimilated every 6 months, starting from June, 2004. Figure 6.11(b) shows the E-type mean of the history matched ensemble of grid block permeabilities. Comparing Figure 6.11(b) and the true reservoir permeability in Figure 6.10(a), we can observe that the EnKF updated ensembles contain

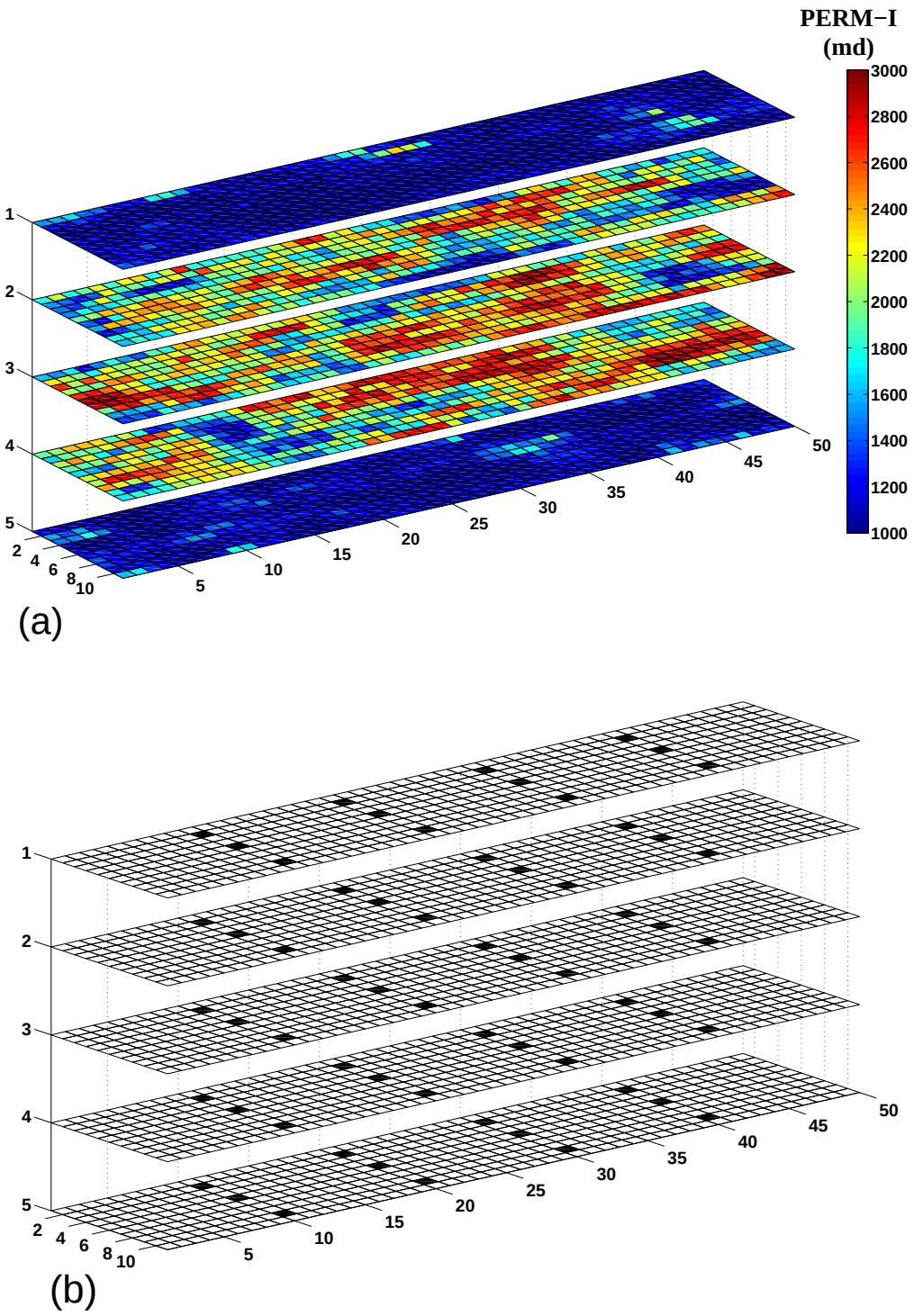
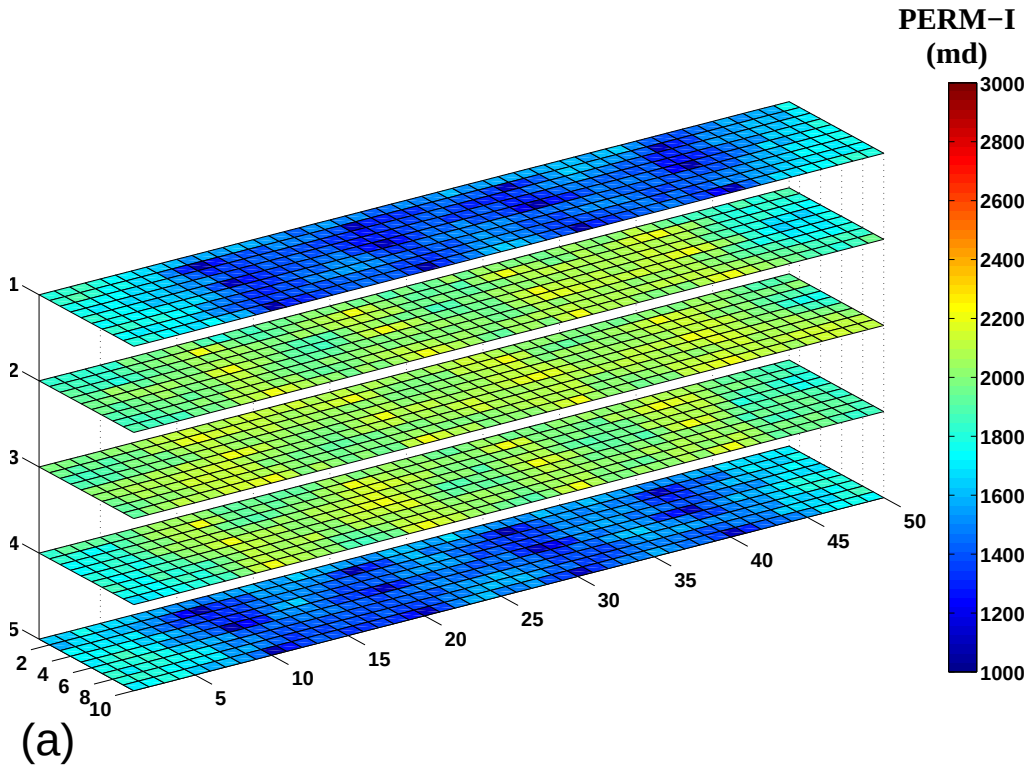


Figure 6.10: (a) Permeability of the true reservoir generated using a sequential Gaussian simulation. (b) Locations of 12 vertical core holes at which hard data of the true reservoir permeability were assumed to be available. The hard data at core holes were used as conditioning data for generating the initial ensembles and bootstrap the EnKF algorithm.



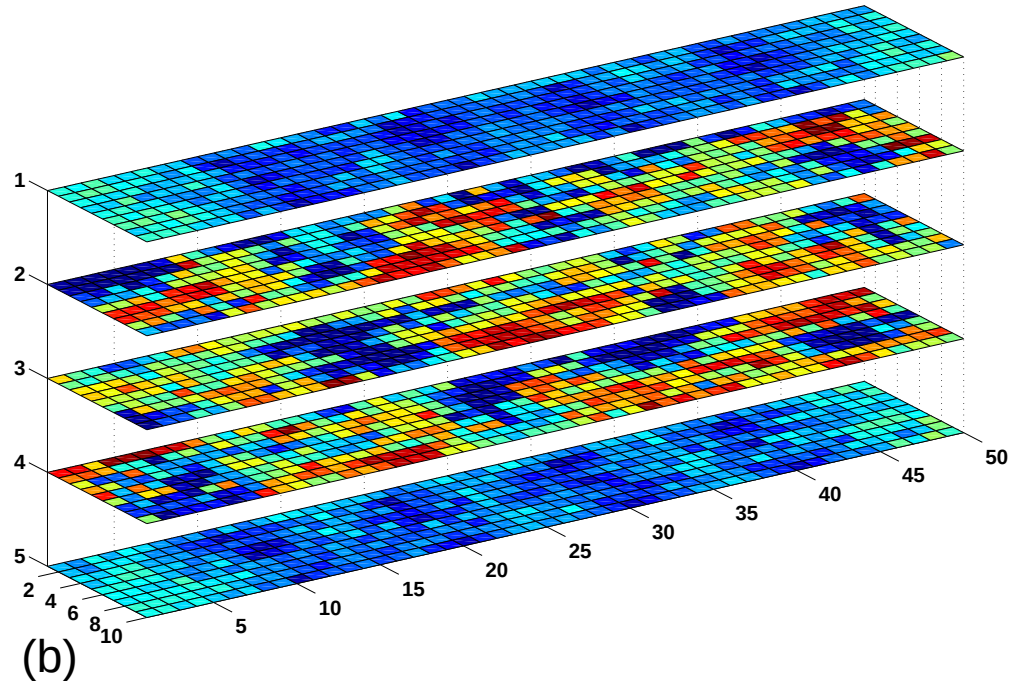


Figure 6.11: E-type mean of the ensemble realizations showing permeability (a) *before* history matching, and (b) *after* history matching using the EnKF algorithm.

more plausible information about the spatial heterogeneity. Figure 6.12 shows the comparison of ensemble predictions of the production data before and after history matching. Similar to the previous case study, the ensemble of history matched grid block permeabilities at the end of the last update step was used to simulate the *'After history matching'* case. The predictions so obtained were compared to the simulated predictions of the true reservoir. To compare the effectiveness of history matching, the initial ensembles were used in a separate simulation to produce the *'Before history matching'* case. From the comparison of the predictions, we can clearly observe that the realizations after the EnKF based history matching are much more plausible when compared to the initial ensembles. The quality of the ensembles in terms of the WMSE and R-square value is shown in Figure 6.13. The R-square value suggests more than 90% fit between the predictions and the observed production data for all the ensembles, thus showing that the history match is reliable.

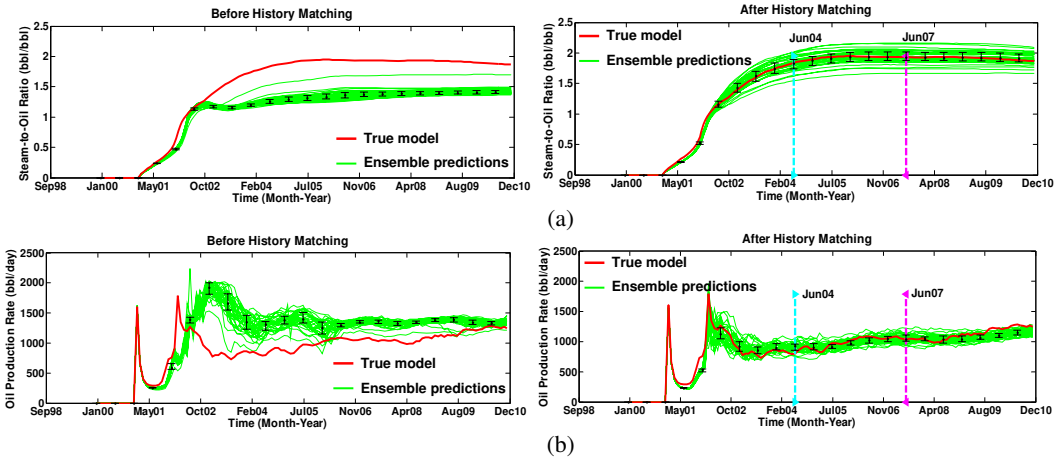


Figure 6.12: Comparison of ensemble predictions (30 realizations) *before* and *after* EnKF history matching for (a) Cumulative Steam-to-Oil Ratio (SOR), and (b) Oil Production Rate (OP_{rate}) for the entire field.

Figure 6.14 shows how the E-type mean and standard deviation of the ensemble permeability evolves as the EnKF algorithm sequentially updates the reservoir. It can be observed that a significant shift in the mean of the ensembles happens at the first update instance. This is due to large errors in the initial ensemble predictions, which calls for a higher error compensation in the Kalman update step. There is also a significant reduction in the ensemble variance that occurs at the first update step. After the third update onwards (June, 2005), the model prediction errors are significantly lower and hence only minute adjustments are required. The EnKF algorithm automatically infers this vital information based on the sample statistics (covariance and cross-covariance) of the ensemble and takes account of this information when making any parameter updates through the Kalman update

technique. The reduction in the overall variance of the permeability ensembles can be clearly observed in Figure 6.15.

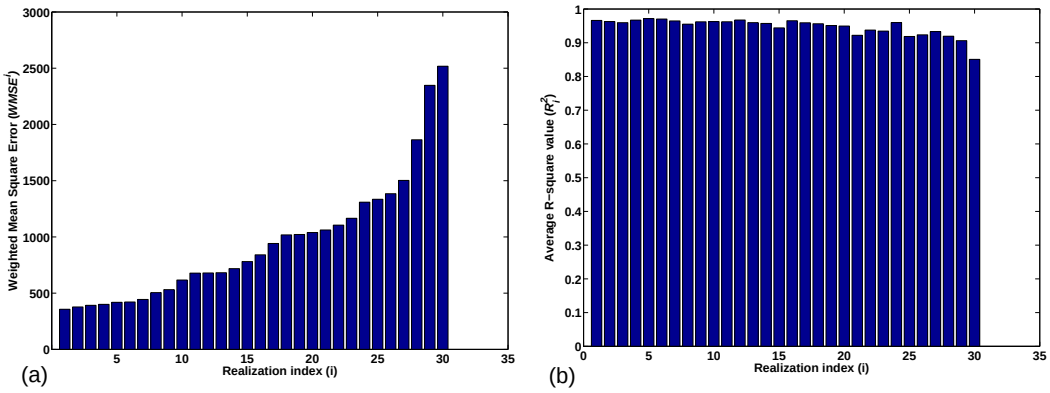


Figure 6.13: Quality of the ensembles (30 realizations) *after* EnKF history matching of the SAGD reservoir in terms of (a) WMSE: The realizations after i= 25 can be considered to have poor accuracy relative to the others, (b) R-square value: The high R-square value of all the ensembles indicate that the history match is reliable.

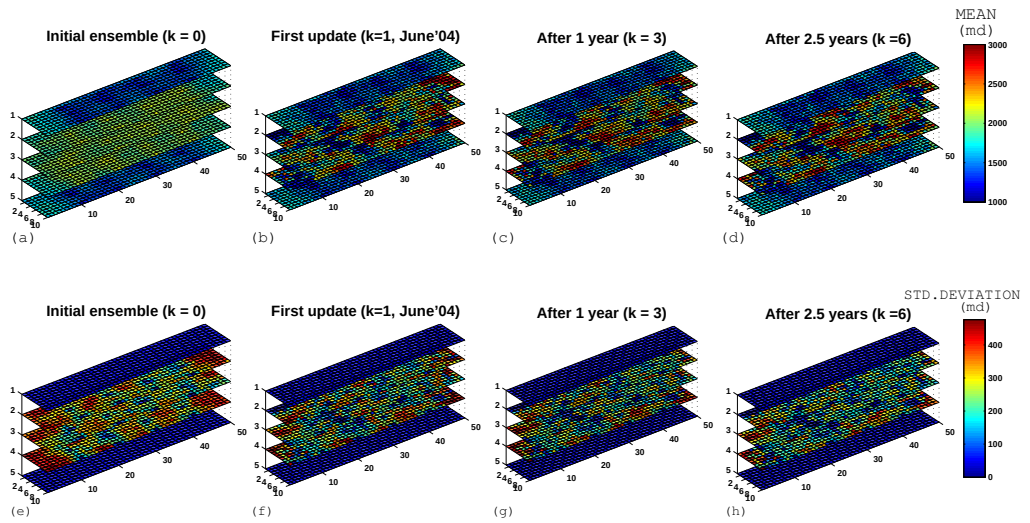


Figure 6.14: Evolution of E-type mean (*top row*) and standard deviation (*bottom row*) of the permeability ensembles with the progression of EnKF update steps, (a, e): Initial ensemble generated using geostatistical simulation, (b, f): After first update, (c, g): After 1 year from first update (3rd update), and (d, h): After 2.5 year from first update (6th update). The ensemble std. deviation significantly reduces after the first update and remains low thereafter showing significant convergence of the EnKF algorithm.

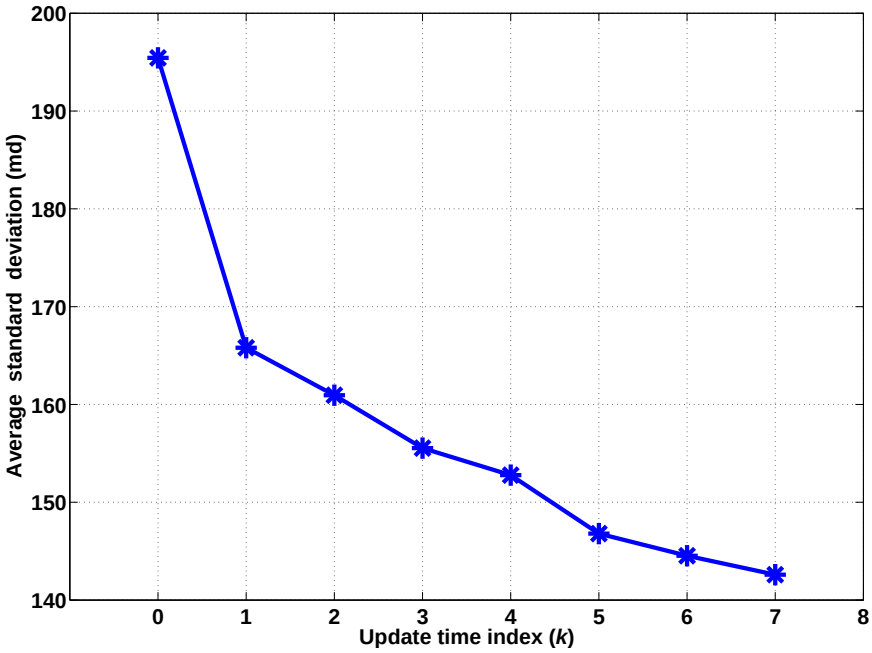


Figure 6.15: Evolution of the aggregate mean of permeability E-type standard deviation with the progression of EnKF update steps.

Figure 6.16 compares the steam chamber growth predictions after history matching with that of the truth case. For this comparison, the realization of the history matched ensemble that yielded the lowest WMSE was used, i.e., ranked best in terms of the production data history match. Overall, we can observe a good match between the truth case and the predicted spatial distribution of the oil saturation. Figure 6.17 shows the steam chamber growth in the 1st and 3rd layer for comparison. We can observe that the oil saturation in layer 3 shows a better match with the truth case, compared to the top layer. This can be attributed to the higher sensitivity of layer 3 to the production data compared to layer 1. The grid blocks with higher sensitivity will be updated more effectively in the EnKF update step.

6.4 Concluding Remarks

In this work, the efficacy of the EnKF algorithm for history matching of heterogeneous reservoirs, by taking into account the prior knowledge about the geological heterogeneity was demonstrated. Considering two synthetic case studies, the application of this technique for sequentially updating the heterogeneous permeability at grid block locations was demonstrated. Two quality measures for evaluating the efficacy of history matching were proposed, which could summarize the overall quality as well as the quality of individual ensembles at the end of history matching

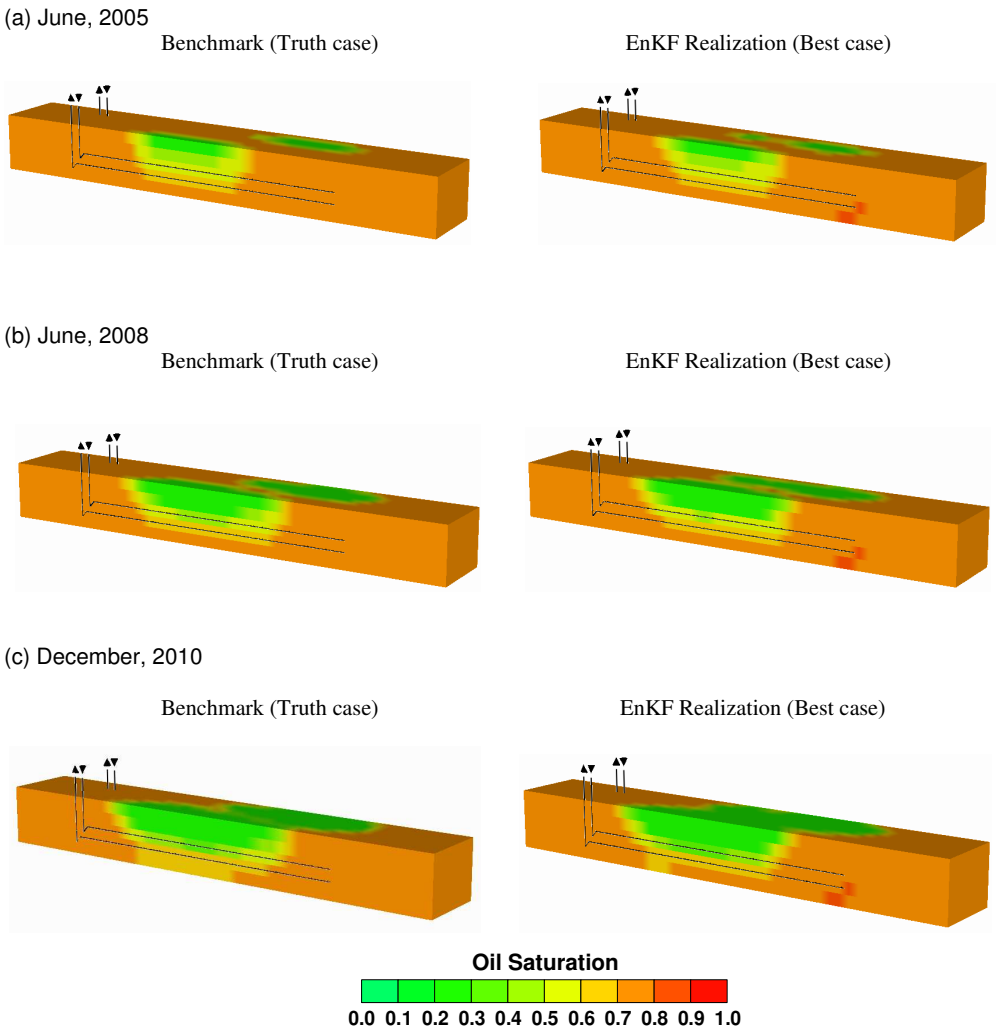


Figure 6.16: Comparison of the steam chamber growth of the True case (*on the left*) and the EnKF realization ranked highest after history matching (*on the right*), for the month of (a) June'05, (b) June'08, and (c) December'10. The figures show the oil saturation in a reservoir section cut out in the vertical plane containing one of the well pairs.

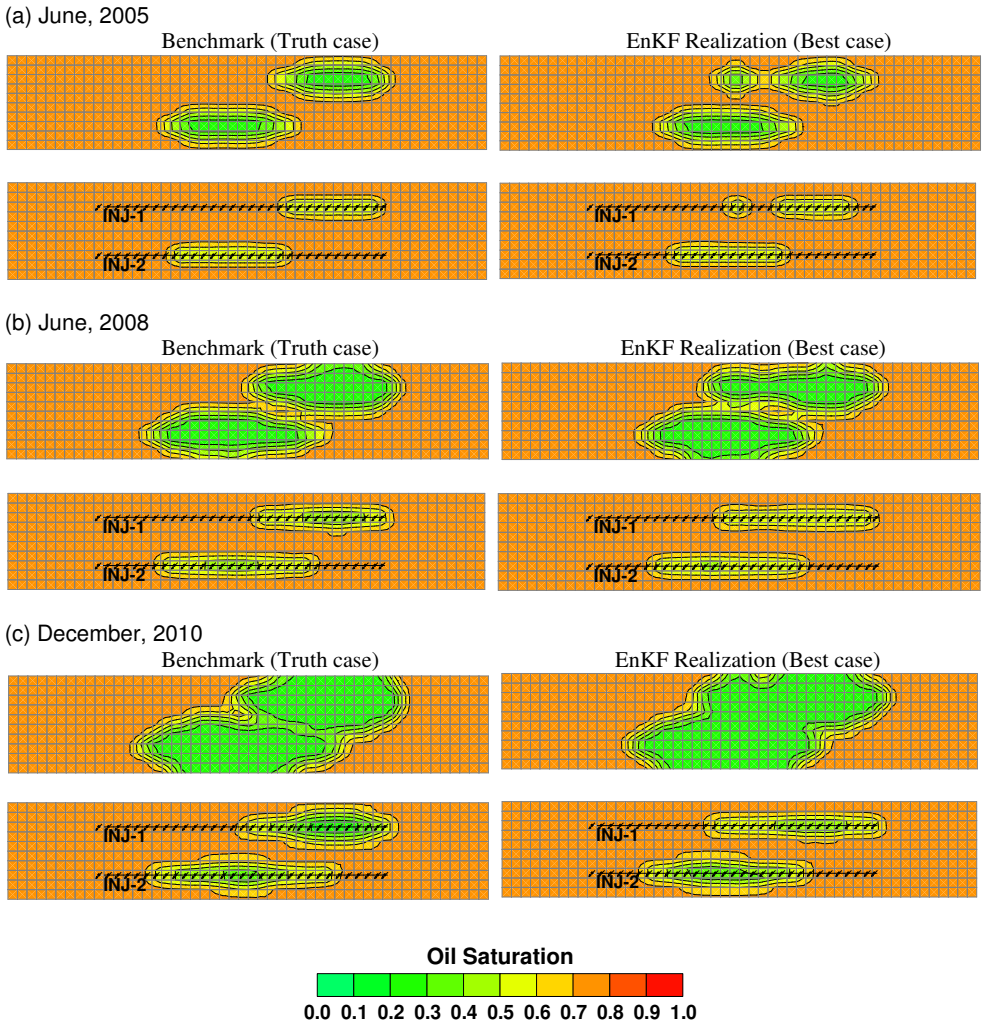


Figure 6.17: Comparison of the steam chamber growth in *layers 1 & 3* of the True case (*on the left*) and the EnKF realization ranked highest after history matching (*on the right*), for the month of (a) June'05, (b) June'08, and (c) December'10. For each month, the 1^{st} row shows *layer 1* and the 2^{nd} row shows *layer 3*.

in the EnKF algorithm. The prior knowledge about the geological heterogeneity can be easily integrated into the EnKF algorithm through geostatistical simulation techniques, which modern day reservoir engineers are accustomed to doing as part of any reservoir uncertainty evaluation project. The entire algorithm is non-iterative in nature and does not suffer from the limitations of expensive numerical optimization routines. One of the main advantages of the EnKF algorithm is that it can account for the uncertainty in the unknown parameters such as permeability and porosity using a few Monte Carlo realizations ($O(50)$). The algorithm allows for each of the Monte Carlo realizations to be simulated on any commercial reservoir simulator and the computations does not involve the partial differential equations of the reservoir simulator in any form. Moreover, the technique inherently allows for each of the simulations to be carried out independently from the others, in a parallel fashion. Significant reduction of overall time required for history matching is possible by implementing it on modern day multi-processor computing architectures.

Bibliography

Aanonsen, S. I., Naevdal, G., Oliver, D. S., Reynolds, A. C., & Valles, B. (2009). The ensemble Kalman filter in reservoir engineering - a review. *SPE Journal*, *14*(3), 393–412.

Burgers, G., van Leeuwen, P. J., & Evensen, G. (1998). Analysis scheme in the ensemble Kalman filter. *Monthly Weather Review*, *126*, 1719–1724.

Chitralekha, S. B., Trivedi, J. J., & Shah, S. L. (2010). Application of the ensemble Kalman filter for characterization and history matching of unconventional oil reservoirs. In *Canadian Unconventional Resources & International Petroleum Conference, Calgary, Alberta, Canada*. CSUG/SPE-137480.

Christie, M. & Blunt, M. (2001). SPE Comparative Solution Project. First data set (model-1), Society of Petroleum Engineers, Inc. Available from: http://www.spe.org/web/csp//datasets/set01.htm.

Daum, F. (2005). Nonlinear filters: Beyond the Kalman filter. *IEEE Aerospace and Electronic Systems Magazine*, *20*(8), 57–69.

Deutsch, C. V. & Journel, A. G. (1998). *GSLIB: Geostatistical Software Library and User's Guide*. Oxford University Press, New York.

Evensen, G. (1994). Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. *Journal of Geophysical Research*, *99*, 10143–10162.

Evensen, G. (2003). The ensemble Kalman filer: theoretical formulation and practical implementation. *Ocean Dynamics*, *53*, 343–367.

Evensen, G. (2007). Estimation in an oil reservoir simulator. In *Data Assimilation: The Ensemble Kalman Filter* (pp. 239–248). Springer Berlin.

Evensen, G. (2009). The ensemble Kalman filter for combined state and parameter estimation. *IEEE Control Systems Magazine*, *29*(3), 82–104.

Gu, Y. & Oliver, D. S. (2007). An iterative ensemble Kalman filter for multiphase fluid flow data assimilation. *SPE Journal*, *12*(4), 438–446.

Hansen, T. M. (2009). mgstat : A geostatistical Matlab toolbox; version 0.95. Available from http://mgstat.sourceforge.net/.

Hansen, T. M. & Mosegaard, K. (2008). VISIM: Sequential simulation for linear inverse problems. *Computers and Geosciences*, *34*(1), 53–76.

Haugen, V., Natvik, L. J., Evensen, G., Berg, A., Flornes, K., & Naevdal, G. (2008). History matching using the ensemble Kalman filter on a North Sea field case. *SPE Journal*, *13*(4), 382–391.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, *82*(Series D), 35–45.

Ljung, L. (1999). Comparing model structures. In *System Identification:Theory for the User*. Prentice Hall PTR, New Jersey, USA. (chap. 16, sec. 16.4).

Lorentzen, R. J., Fjelde, K. K., Froyen, J., Lage, A. C. V. M., Naevdal, G., & Vefring, E. H. (2001). Underbalanced and low-head drilling operations: Real time interpretation of measured data and operational support. In *SPE Annual Technical Conference and Exhibition, New Orleans, Louisiana*. Society of Petroleum Engineers Inc.

McLennan, J. & Deutsch, C. (2006). Best practice reservoir characterization for the Alberta oil sands. In *7th Canadian International Petroleum Conference, Calgary, AB, Canada*.

McLennan, J. & Deutsch, C. (2010). SAGD reservoir characterization using Geostatistics: Application to the Athabasca oil sands, Alberta, Canada. In *CHOA Handbook* (pp. 321–340). Canadian Heavy Oil Association.

Moradkhani, H., Sorooshian, S., Gupta, H. V., & Houser, P. R. (2005). Dual state-parameter estimation of hydrological models using ensemble Kalman filter. *Advances in Water Resources*, *28*, 135–147.

Naevdal, G., Johnsen, L. M., Aanonsen, S. I., & Vefring, E. H. (2005). Reservoir monitoring and continuous model updating using ensemble Kalman filter. *SPE Journal*, *10*(1), 66–74.

Oliver, D. S., Reynolds, A. C., & Liu, N. (2008). *Inverse Theory for Petroleum Reservoir Characterization and History Matching*. Cambridge University Press, New York.

Remy, N., Boucher, A., & Wu, J. (2009). *Applied Geostatistics with SGeMS: A User's Guide*. Cambridge University Press.

Romeu, R. K. (2010). History matching and forecasting. *Journal of Petroleum Technology*, *April*, 80.

Schulze-Riegert, R. & Ghedan, S. (2007). Modern techniques for history matching. In *9th International Forum on Reservoir Simulation, Abu Dhabi, United Arab Emirates*.

Schulze-Riegert, R. W., Axmann, J. K., Haase, O., Rian, D. T., & You, Y. L. (2001). Optimization methods for history matching of complex reservoirs. In *SPE Reservoir Simulation Symposium, Houston, Texas*. Society of Petroleum Engineers, Inc.

Seiler, A., Evensen, G., Skjervheim, J.-A., Hove, J., & Vabo, J. (2009). Advanced reservoir management workflow using an EnKF based assisted history matching method. In *SPE Reservoir Simulation Symposium, Woodlands, TX*. Society of Petroleum Engineers, Inc.

Wen, X.-H. & Chen, W. H. (2006). Real-time reservoir model updating using ensemble Kalman filter with confirming option. *SPE Journal*, *11*(4), 431–442.

Wen, X.-H. & Chen, W. H. (2007). Some practical issues on real-time reservoir model updating using ensemble Kalman filter. *SPE Journal*, *12*(2), 156–166.

Zhang, Y. & Oliver, D. S. (2009). History matching using a hierarchical stochastic model with the ensemble Kalman filter: A field case study. In *SPE Reservoir Simulation Symposium, Woodlands, TX*. Society of Petroleum Engineers, Inc.

Chapter 7

General Discussion and Concluding Remarks

7.1 Major Contributions

The main contributions of this thesis, as explained in each of the earlier chapters, are summarized below:

- **Chapter 2** presented the application of the Support Vector Regression technique for building nonlinear empirical models of three highly nonlinear processes: (a) an industrial scale EVA polymer extrusion process, (b) a simulated pH neutralization process, and (c) a laboratory scale twin screw extrusion process. In the first case study, a steady state soft sensor for the polymer melt index was successfully developed. Also, the soft sensor was shown to outperform a nonlinear least square regression based soft sensor which was previously installed in the plant. In the latter two case studies, dynamic nonlinear ARX (NARX) models were successfully built using the SVR technique. A novel heuristic procedure for automatically choosing the optimal order and delays of the NARX models was also proposed. The SVR-NARX models were shown to give better predictions than models which were based on the traditional least square regression criteria.

- **Chapter 3** presented a tutorial introduction of the basics of state estimation, with a special focus on sequential Monte Carlo algorithms for state estimation of nonlinear systems. Starting from the analytical solution of the Kalman filter, the ensemble Kalman filter was explained as a Monte Carlo version of the Kalman filter. Then, the basic computational steps in the particle filter algorithm were explained in detail through the use of a simple illustrative example. A comparative study of these algorithms were also presented through simulation case studies. These case studies serve to clearly bring out the distinct mathematical assumptions inherent in these filtering algorithms.

- **Chapter 4** presented a novel solution for detection and quantification of stiction in control valves using a Kalman filter type of unknown input observer. The formulation of valve stiction as an external unknown input affecting the control loop was the key idea. The process, excluding the valve, was assumed to be described by a linear state space model. Industrial case studies demonstrated the success of the methodology in terms of detection and also accuracy of quantification of the fault. Also, simulation studies demonstrated the robustness of the methodology in the presence of model-plant mismatch and external oscillatory disturbances.

- **Chapter 5** presented a comparative study of combined state and parameter estimation techniques. A batch estimation algorithm based on an expectation-maximization framework was presented. Three popular filtering and smoothing algorithms were considered in the comparative study: (a) particle smoother, (b) unscented Kalman smoother, and (c) extended Kalman smoother. An online recursive scheme for combined state and parameter estimation was also presented, with a focus on the ensemble Kalman filter. Using a simulated fermentor reactor, empirical case studies were performed to bring out the efficacy of these techniques in terms of estimation accuracy and computational cost.

- **Chapter 6** presented the application of the ensemble Kalman filter for the estimation of parameters of large scale petroleum reservoir systems, commonly referred to as history matching. Two synthetic, unconventional petroleum reservoir simulation models were considered: (a) a highly heterogeneous black-oil reservoir, and (b) a 3-dimensional SAGD reservoir. The objective was to characterize the heterogeneous grid-block permeability of the entire reservoir taking into account the geostatistical prior knowledge as well as the production data measurements at the well locations. Also, quantitative indices which can be used to summarize the 'history matched' quality after history matching were proposed. The case studies successfully demonstrated the efficacy of the methodology for fast, sequential updating of the heterogeneous reservoir permeability.

7.2 Recommendations for Future Work

During the course of the work in this thesis, some possible areas for future work were identified which will require additional work. A few of these areas, which merit additional investigation, are listed below.

7.2.1 Extensions to the Support Vector Regression technique

As mentioned in Chapter 2, one of the key features of the SVR technique is the sparse nonlinear approximation of the input-output relationship of the process variables. Recently, there has been a lot of interest in the literature on a novel sampling

paradigm called Compressive Sampling (CS) with the objective of obtaining sparse nonlinear approximations of naturally sparse signals (Candes & Wakin, 2008). The basis functions in CS are required in their explicit form and the ℓ_1 norm of the model weight vector is minimized to obtain the optimal model for signal reconstruction. For robust recovery of signals from noisy data, hard constraints are included in the optimization problem formulation in CS. Note that in $\varepsilon-$SVR, noise is handled through the ε-insensitive loss function which is a soft constraint in the optimization formulation. The use of ℓ_1 norm of w instead of the ℓ_2 norm (Weston et al., 1999) can be considered as a variation on the theme of $\varepsilon-$SVR along similar lines of the CS theory. However, the link between the CS theory and the SVR technique has not been explored in the literature. The idea of sparsity is prevalent in both $\varepsilon-$SVR and CS; a combination of the main ideas in these two formulations may pave ways to interesting results in future.

7.2.2 Applications of the Unknown Input Observer

In Chapter 3, an unknown input observer (UIO) was used to estimate the valve position. The estimated valve position was used to detect and quantify stiction. However, there are several other valve abnormalities which can be detected using the valve position information. Three of the important ones are (1) valve nonlinearity, (2) large deadband, and (3) valve hysteresis. All of the above mentioned abnormalities will be reflected as a distinct pattern in the $mv - op$ plot of the valve and hence can be used for fault detection. With some additional work, application of the unknown input observer technique for detecting such faults can be studied.

A more general area to explore the application of the UIO in the process industry is the detection of model-plant mismatch in process control loops. If there is a systematic mismatch between the true process and its best linear approximation, then the UIO can be applied to estimate the mismatch as an unknown input. Process nonlinearity can be one of the causes of model-plant mismatch and its detection is a significant step in the assessment of closed loop performance of a process (Choudhury et al., 2008).

7.2.3 Advanced techniques for history matching using the EnKF algorithm

The petroleum engineering literature has been vibrant with the application of the EnKF algorithm for history matching. However, the application of the EnKF for the history matching of SAGD reservoirs and other enhanced oil recovery reservoirs has not been significantly explored. In the SAGD reservoir, one of the important production variables which we have not considered in this thesis is the spatial temperature measurements. The inclusion of these measurements in the EnKF-

based history matching will add more information about the spatial heterogeneity. Another future challenge in this area is incorporating constraints in the EnKF to enforce the geologic realism in the updated model. For example, constraints can be included to efficiently define in some functional form the spatial continuity of the permeability and porosity. A modified form of the Kalman update rule, which will incorporate such constraints, will have to be developed.

7.3 Concluding Remarks

Overall, this thesis has presented novel case studies which demonstrate the success of modern computational tools in transferring the information available from process data into empirical and physio-chemical models of the process. From a practical stand point of process control today, it is evident that empirical soft sensors have already penetrated the work flow of practicing process control engineers (Kern, 2010). However, new computational tools which can make life easier for process control practitioners are always required as the chemical processes become more and more complex and nonlinear. An applied research based study of new computational tools is indispensable for translating the mathematical developments into practical engineering tools.

Bibliography

Candes, E. J. & Wakin, M. B. (2008). An introduction to compressive sampling. *IEEE Signal Processing Magazine*, *25*(2), 21–30.

Choudhury, M. A. A. S., Shah, S. L., & Thornhill, N. F. (2008). *Diagnosis of Process Nonlinearities and Valve Stiction - Data Driven Approaches*. Advances in Industrial Control. Springer.

Kern, A. (2010). An inferentials update: Use these techniques to achieve "high-bar" inferential performance. *InTech*, *May/June*, 14–18. ISA.

Weston, J., Gammerman, A., Stitson, M. O., Vapnik, V., Vovk, V., & Watkins, C. (1999). Support Vector density estimation. In B. Scholkopf, C. J. C. Burges, & A. J. Smola (Eds.), *Advances in Kernel Methods–Support Vector Learning* (pp. 293–305). MIT Press, Cambridge, MA.

Appendix A

Kalman filter from a Bayesian perspective

Recalling Eqs. (3.11) and (3.12) from Chapter 3, we have the following expressions for the prior density and the likelihood for the state at time instant $k = 1$:

$$p(x_1|y_0) = \frac{1}{\sqrt{2\pi}\sigma_{1|0}} \exp\left(\frac{-(x_1 - \mu_{1|0})^2}{2\sigma_{1|0}^2}\right) \tag{A.1}$$

$$p(y_1|x_1) = \frac{1}{\sqrt{2\pi}\sigma_n} \exp\left(\frac{-(y_1 - x_1)^2}{2\sigma_n^2}\right) \tag{A.2}$$

Consider the posterior probability density for the state, $p(x_1|y_1)$, at time instant $k = 1$. From Eq. (3.3), we can relate $p(x_1|y_0)$ and $p(y_1|x_1)$ to $p(x_1|y_1)$ as follows:

$$p(x_1|y_1) = \frac{p(x_1|y_0)p(y_1|x_1)}{p(y_1|y_0)} \tag{A.3}$$

Based on the measurement equation in Eq. (3.10), at time instant $k = 1$, we have the following:

$$y_1 = x_1 + n_1 \tag{A.4}$$

From Eq. (A.4), the conditional probability density $p(y_1|y_0)$ can be concluded to be a Gaussian one, with the following conditional mean and variance:

$$\begin{aligned} E(y_1|y_0) &= E(x_1 + n_1|y_0) \\ &= E(x_1|y_0) + E(n_1|y_0) \\ &= \mu_{1|0} + 0 \\ &= \mu_{1|0} \end{aligned}$$

$$\begin{aligned} E[\{y_1 - E(y_1|y_0)\}^2|y_0] &= E[(x_1 + n_1 - \mu_{1|0})^2|y_0] \\ &= E\{[n_1^2 + (x_1 - \mu_{1|0})^2 - 2(x_1 - \mu_{1|0})n_1]|y_0\} \\ &= E[n_1^2|y_0] + E[(x_1 - \mu_{1|0})^2|y_0] - 2E[(x_1 - \mu_{1|0})n_1|y_0] \\ &= \sigma_n^2 + \sigma_{1|0}^2 + 0 \\ &= \sigma_n^2 + \sigma_{1|0}^2 \end{aligned}$$

Note that $E[(x_1 - \mu_{1|0})n_1|y_0] = 0$, since the measurement noise is zero-mean and uncorrelated with the state variable. Thus, the density in the denominator of Eq. (A.3) is given by

$$p(y_1|y_0) = \frac{1}{\sqrt{2\pi}\sqrt{\sigma_n^2 + \sigma_{1|0}^2}} \exp\left(\frac{-(y_1 - \mu_{1|0})^2}{2(\sigma_n^2 + \sigma_{1|0}^2)}\right) \tag{A.5}$$

Substituting Eqs. (A.1), (A.2), and (A.5) into Eq. (A.3), we obtain the following

$$\begin{aligned} p(x_1|y_1) &= \frac{\frac{1}{\sqrt{2\pi}\sigma_{1|0}} \exp\left(\frac{-(x_1-\mu_{1|0})^2}{2\sigma_{1|0}^2}\right) \frac{1}{\sqrt{2\pi}\sigma_n} \exp\left(\frac{-(y_1-x_1)^2}{2\sigma_n^2}\right)}{\frac{1}{\sqrt{2\pi}\sqrt{\sigma_n^2+\sigma_{1|0}^2}} \exp\left(\frac{-(y_1-\mu_{1|0})^2}{2(\sigma_n^2+\sigma_{1|0}^2)}\right)} \\ &= \frac{\frac{1}{\sqrt{2\pi}\sigma_{1|0}} \frac{1}{\sqrt{2\pi}\sigma_n} \exp - \left(\frac{(x_1-\mu_{1|0})^2}{2\sigma_{1|0}^2} + \frac{(y_1-x_1)^2}{2\sigma_n^2}\right)}{\frac{1}{\sqrt{2\pi}\sqrt{\sigma_n^2+\sigma_{1|0}^2}} \exp\left(\frac{-(y_1-\mu_{1|0})^2}{2(\sigma_n^2+\sigma_{1|0}^2)}\right)} \end{aligned} \tag{A.6}$$

Ignoring all the terms which are independent of x_1 as a proportionality constant, we obtain the following relationship

$$p(x_1|y_1) \propto \exp - \left(\frac{(x_1 - \mu_{1|0})^2}{2\sigma_{1|0}^2} + \frac{(y_1 - x_1)^2}{2\sigma_n^2}\right) \tag{A.7}$$

The exponential factor $\left(\frac{(x_1-\mu_{1|0})^2}{2\sigma_{1|0}^2} + \frac{(y_1-x_1)^2}{2\sigma_n^2}\right)$ in the above equation will be used to perform some algebraic manipulations, as explained below. First, by performing a simple algebraic manipulation of the exponential factor, we obtain the following:

$$\frac{1}{2}\left(\frac{(x_1 - \mu_{1|0})^2}{\sigma_{1|0}^2} + \frac{(y_1 - x_1)^2}{2\sigma_n^2}\right) = \frac{1}{2\sigma_{1|0}^2\sigma_n^2}\left(\sigma_n^2(x_1 - \mu_{1|0})^2 + \sigma_{1|0}^2(x_1 - y_1)^2\right) \tag{A.8}$$

Expanding the squares in the above equation, we obtain

$$\begin{aligned} &\frac{1}{2\sigma_{1|0}^2\sigma_n^2}\left(\sigma_n^2(x_1 - \mu_{1|0})^2 + \sigma_{1|0}^2(x_1 - y_1)^2\right) \\ &= \frac{1}{2\sigma_{1|0}^2\sigma_n^2}\left[\sigma_n^2(x_1^2 + \mu_{1|0}^2 - 2x_1\mu_{1|0}) + \sigma_{1|0}^2(x_1^2 + y_1^2 - 2x_1y_1\right] \\ &= \frac{1}{2\sigma_{1|0}^2\sigma_n^2}\left[x_1^2(\sigma_n^2 + \sigma_{1|0}^2) + \sigma_n^2\mu_{1|0}^2 - 2x_1\mu_{1|0}\sigma_n^2 + \sigma_{1|0}^2y_1^2 - 2x_1y_1\sigma_{1|0}^2\right] \\ &= \frac{1}{2\sigma_{1|0}^2\sigma_n^2}\left[x_1^2(\sigma_n^2 + \sigma_{1|0}^2) + \sigma_n^2\mu_{1|0}^2 + \sigma_{1|0}^2y_1^2 - 2x_1(\mu_{1|0}\sigma_n^2 + y_1\sigma_{1|0}^2)\right] \end{aligned} \tag{A.9a}$$

By grouping the terms containing the random variable x_1 in Eq. (A.9a), we obtain

$$\begin{aligned}
&\frac{1}{2\sigma_{1|0}^2\sigma_n^2}\left[x_1^2(\sigma_n^2+\sigma_{1|0}^2)-2x_1(\mu_{1|0}\sigma_n^2+y_1\sigma_{1|0}^2)\right]+\frac{\sigma_n^2\mu_{1|0}^2+\sigma_{1|0}^2y_1^2}{2\sigma_{1|0}^2\sigma_n^2}\\
=\;&\frac{(\sigma_n^2+\sigma_{1|0}^2)}{2\sigma_{1|0}^2\sigma_n^2}\left\{x_1^2-2x_1\left(\frac{\mu_{1|0}\sigma_n^2+y_1\sigma_{1|0}^2}{(\sigma_n^2+\sigma_{1|0}^2)}\right)\right\}+\frac{\sigma_n^2\mu_{1|0}^2+\sigma_{1|0}^2y_1^2}{2\sigma_{1|0}^2\sigma_n^2}
\end{aligned}$$

By *completing the square* of the terms inside the curly braces in the above equation, we obtain

$$\begin{aligned}
&\frac{(\sigma_n^2+\sigma_{1|0}^2)}{2\sigma_{1|0}^2\sigma_n^2}\left[\left\{x_1-\left(\frac{\mu_{1|0}\sigma_n^2+y_1\sigma_{1|0}^2}{(\sigma_n^2+\sigma_{1|0}^2)}\right)\right\}^2-\left(\frac{\mu_{1|0}\sigma_n^2+y_1\sigma_{1|0}^2}{(\sigma_n^2+\sigma_{1|0}^2)}\right)^2\right]\\
&\quad+\frac{\sigma_n^2\mu_{1|0}^2+\sigma_{1|0}^2y_1^2}{2\sigma_{1|0}^2\sigma_n^2}\\
&=\frac{(\sigma_n^2+\sigma_{1|0}^2)}{2\sigma_{1|0}^2\sigma_n^2}\left\{x_1-\left(\frac{\mu_{1|0}\sigma_n^2+y_1\sigma_{1|0}^2}{(\sigma_n^2+\sigma_{1|0}^2)}\right)\right\}^2-\frac{(\sigma_n^2+\sigma_{1|0}^2)}{2\sigma_{1|0}^2\sigma_n^2}\left(\frac{\mu_{1|0}\sigma_n^2+y_1\sigma_{1|0}^2}{(\sigma_n^2+\sigma_{1|0}^2)}\right)^2\\
&\quad+\frac{\sigma_n^2\mu_{1|0}^2+\sigma_{1|0}^2y_1^2}{2\sigma_{1|0}^2\sigma_n^2} \qquad \text{(A.10)}
\end{aligned}$$

Consider the term $\frac{\mu_{1|0}\sigma_n^2+y_1\sigma_{1|0}^2}{(\sigma_n^2+\sigma_{1|0}^2)}$ in the above equation. By carrying out an algebraic manipulation, we obtain the following:

$$\begin{aligned}
\frac{\mu_{1|0}\sigma_n^2+y_1\sigma_{1|0}^2}{(\sigma_n^2+\sigma_{1|0}^2)} &= \frac{\mu_{1|0}\sigma_n^2+\mu_{1|0}\sigma_{1|0}^2-\mu_{1|0}\sigma_{1|0}^2+y_1\sigma_{1|0}^2}{(\sigma_n^2+\sigma_{1|0}^2)}\\
&= \frac{\mu_{1|0}(\sigma_n^2+\sigma_{1|0}^2)+\sigma_{1|0}^2(y_1-\mu_{1|0})}{(\sigma_n^2+\sigma_{1|0}^2)}\\
&= \mu_{1|0}+\frac{\sigma_{1|0}^2}{(\sigma_n^2+\sigma_{1|0}^2)}(y_1-\mu_{1|0})\\
&= \mu_{1|0}+K_1(y_1-\mu_{1|0}) \qquad \text{(A.11)}\\
&\triangleq \mu_{1|1} \qquad \text{(A.12)}
\end{aligned}$$

where, K_1 is the Kalman gain defined as

$$K_1=\frac{\sigma_{1|0}^2}{(\sigma_n^2+\sigma_{1|0}^2)} \qquad \text{(A.13)}$$

The final result of this derivation will prove that $\mu_{1|1}$ is the mean value of the posterior distribution of the state at time instant $k=1$. Next, consider the term $\frac{(\sigma_n^2+\sigma_{1|0}^2)}{2\sigma_{1|0}^2\sigma_n^2}\left(\frac{\mu_{1|0}\sigma_n^2+y_1\sigma_{1|0}^2}{(\sigma_n^2+\sigma_{1|0}^2)}\right)^2$ in Eq. (A.10). By doing a simple algebraic expansion we obtain

$$\frac{(\sigma_n^2+\sigma_{1|0}^2)}{2\sigma_{1|0}^2\sigma_n^2}\left(\frac{\mu_{1|0}\sigma_n^2+y_1\sigma_{1|0}^2}{(\sigma_n^2+\sigma_{1|0}^2)}\right)^2=\frac{1}{2\sigma_{1|0}^2\sigma_n^2}\frac{(\mu_{1|0}\sigma_n^2+y_1\sigma_{1|0}^2)^2}{(\sigma_n^2+\sigma_{1|0}^2)} \qquad \text{(A.14)}$$

Substituting Eq. (A.14) and Eq. (A.12) into the R.H.S. of Eq. (A.10), we obtain

$$\begin{aligned}
&\frac{(\sigma_n^2+\sigma_{1|0}^2)}{2\sigma_{1|0}^2\sigma_n^2}(x_1-\mu_{1|1})^2-\frac{1}{2\sigma_{1|0}^2\sigma_n^2}\frac{\left(\mu_{1|0}\sigma_n^2+y_1\sigma_{1|0}^2\right)^2}{(\sigma_n^2+\sigma_{1|0}^2)}+\frac{\sigma_n^2\mu_{1|0}^2+\sigma_{1|0}^2y_1^2}{2\sigma_{1|0}^2\sigma_n^2}\\
&=\frac{(\sigma_n^2+\sigma_{1|0}^2)}{2\sigma_{1|0}^2\sigma_n^2}(x_1-\mu_{1|1})^2+\frac{1}{2\sigma_{1|0}^2\sigma_n^2}\left(\sigma_n^2\mu_{1|0}^2+\sigma_{1|0}^2y_1^2-\frac{(\mu_{1|0}\sigma_n^2+y_1\sigma_{1|0}^2)^2}{(\sigma_n^2+\sigma_{1|0}^2)}\right)\\
&=\frac{(\sigma_n^2+\sigma_{1|0}^2)}{2\sigma_{1|0}^2\sigma_n^2}(x_1-\mu_{1|1})^2\\
&\quad+\frac{1}{2\sigma_{1|0}^2\sigma_n^2}\left(\frac{(\sigma_n^2\mu_{1|0}^2+\sigma_{1|0}^2y_1^2)(\sigma_n^2+\sigma_{1|0}^2)-(\mu_{1|0}\sigma_n^2+y_1\sigma_{1|0}^2)^2}{(\sigma_n^2+\sigma_{1|0}^2)}\right)\\
&=\frac{(\sigma_n^2+\sigma_{1|0}^2)}{2\sigma_{1|0}^2\sigma_n^2}(x_1-\mu_{1|1})^2+\frac{1}{2\sigma_{1|0}^2\sigma_n^2}\times\\
&\quad\left(\frac{\sigma_n^4\mu_{1|0}^2+\sigma_{1|0}^4y_1^2+\sigma_n^2\sigma_{1|0}^2y_1^2+\sigma_{1|0}^2\sigma_n^2\mu_{1|0}^2-\sigma_n^4\mu_{1|0}^2-\sigma_{1|0}^4y_1^2-2\mu_{1|0}\sigma_n^2y_1\sigma_{1|0}^2}{(\sigma_n^2+\sigma_{1|0}^2)}\right)\\
&=\frac{(\sigma_n^2+\sigma_{1|0}^2)}{2\sigma_{1|0}^2\sigma_n^2}(x_1-\mu_{1|1})^2+\frac{1}{2\sigma_{1|0}^2\sigma_n^2}\left(\frac{\sigma_n^2\sigma_{1|0}^2y_1^2+\sigma_{1|0}^2\sigma_n^2\mu_{1|0}^2-2\mu_{1|0}\sigma_n^2y_1\sigma_{1|0}^2}{(\sigma_n^2+\sigma_{1|0}^2)}\right)\\
&=\frac{(\sigma_n^2+\sigma_{1|0}^2)}{2\sigma_{1|0}^2\sigma_n^2}(x_1-\mu_{1|1})^2+\frac{1}{2}\left(\frac{y_1^2+\mu_{1|0}^2-2\mu_{1|0}y_1}{(\sigma_n^2+\sigma_{1|0}^2)}\right)\\
&=\frac{1}{2}\frac{(\sigma_n^2+\sigma_{1|0}^2)}{\sigma_{1|0}^2\sigma_n^2}(x_1-\mu_{1|1})^2+\frac{1}{2}\frac{(y_1-\mu_{1|0})^2}{(\sigma_n^2+\sigma_{1|0}^2)} \qquad \text{(A.15)}
\end{aligned}$$

Thus, we have the following intermediate result which gives an equivalent form for the exponential factor in Eq. (A.7):

$$\frac{1}{2}\left(\frac{(x_1-\mu_{1|0})^2}{\sigma_{1|0}^2}+\frac{(y_1-x_1)^2}{2\sigma_n^2}\right)=\frac{1}{2}\frac{(\sigma_n^2+\sigma_{1|0}^2)}{\sigma_{1|0}^2\sigma_n^2}(x_1-\mu_{1|1})^2+\frac{1}{2}\frac{(y_1-\mu_{1|0})^2}{(\sigma_n^2+\sigma_{1|0}^2)} \qquad \text{(A.16)}$$

Next, consider the term $\frac{(\sigma_n^2+\sigma_{1|0}^2)}{\sigma_{1|0}^2\sigma_n^2}$ in the above equation. Carrying out an algebraic manipulation of this term, we obtain the following result:

$$\begin{aligned}
\frac{\sigma_n^2+\sigma_{1|0}^2}{\sigma_{1|0}^2\sigma_n^2}&=\frac{\sigma_n^2+\sigma_{1|0}^2}{\sigma_{1|0}^2\sigma_n^2+\sigma_{1|0}^4-\sigma_{1|0}^4}\\
&=\frac{\sigma_n^2+\sigma_{1|0}^2}{\sigma_{1|0}^2(\sigma_n^2+\sigma_{1|0}^2)-\sigma_{1|0}^4}
\end{aligned}$$

$$
\begin{aligned}
&= \frac{1}{\sigma_{1|0}^2 - \frac{\sigma_{1|0}^4}{\sigma_n^2+\sigma_{1|0}^2}} \\
&= \frac{1}{\sigma_{1|0}^2 - \sigma_{1|0}^2 \frac{\sigma_{1|0}^2}{\sigma_n^2+\sigma_{1|0}^2}} \\
&= \frac{1}{\sigma_{1|0}^2 - K_1\sigma_{1|0}^2} \\
&\triangleq \frac{1}{\sigma_{1|1}^2} \qquad \text{(A.17)}
\end{aligned}
$$

where, $\sigma_{1|1}^2 = \sigma_{1|0}^2 - K_1\sigma_{1|0}^2$. The final result of this derivation will prove that $\sigma_{1|1}^2$ is the variance of the posterior distribution of the state at time instant $k = 1$. Substituting $\sigma_{1|1}^2$ into the R.H.S. of Eq. (A.16), we obtain

$$
\frac{1}{2}\left(\frac{(x_1-\mu_{1|0})^2}{\sigma_{1|0}^2} + \frac{(y_1-x_1)^2}{2\sigma_n^2}\right) = \frac{1}{2\sigma_{1|1}^2}(x_1-\mu_{1|1})^2 + \frac{1}{2}\frac{(y_1-\mu_{1|0})^2}{(\sigma_n^2+\sigma_{1|0}^2)}
$$

Thus, the exponential factor in Eq. (A.7) can be expressed in the following equivalent form:

$$
\left(\frac{(x_1-\mu_{1|0})^2}{2\sigma_{1|0}^2} + \frac{(y_1-x_1)^2}{2\sigma_n^2}\right) \equiv \frac{(x_1-\mu_{1|1})^2}{2\sigma_{1|1}^2} + \frac{(y_1-\mu_{1|0})^2}{2(\sigma_n^2+\sigma_{1|0}^2)} \qquad \text{(A.18)}
$$

Substituting the result obtained in Eq. (A.18) into Eq. (A.6) and carrying out a few algebraic manipulations, we obtain the following final result:

$$
p(x_1|y_1) = \frac{\frac{1}{\sqrt{2\pi}\sigma_{1|0}}\frac{1}{\sqrt{2\pi}\sigma_n}\exp - \left(\frac{(x_1-\mu_{1|1})^2}{2\sigma_{1|1}^2} + \frac{(y_1-\mu_{1|0})^2}{2(\sigma_n^2+\sigma_{1|0}^2)}\right)}{\frac{1}{\sqrt{2\pi}\sqrt{\sigma_n^2+\sigma_{1|0}^2}}\exp\left(\frac{-(y_1-\mu_{1|0})^2}{2(\sigma_n^2+\sigma_{1|0}^2)}\right)}
$$

$$
= \frac{\frac{1}{\sigma_{1|0}}\frac{1}{\sqrt{2\pi}\sigma_n}\exp - \left(\frac{(x_1-\mu_{1|1})^2}{2\sigma_{1|1}^2}\right)}{\frac{1}{\sqrt{\sigma_n^2+\sigma_{1|0}^2}}}
$$

$$
\begin{aligned}
&= \frac{\sqrt{\sigma_n^2+\sigma_{1|0}^2}}{\sqrt{2\pi}\sigma_{1|0}\sigma_n}\exp - \left(\frac{(x_1-\mu_{1|1})^2}{2\sigma_{1|1}^2}\right) \\
&= \frac{1}{\sqrt{2\pi}}\sqrt{\frac{\sigma_n^2+\sigma_{1|0}^2}{\sigma_{1|0}^2\sigma_n^2}}\exp - \left(\frac{(x_1-\mu_{1|1})^2}{2\sigma_{1|1}^2}\right) \\
&= \frac{1}{\sqrt{2\pi}}\sqrt{\frac{1}{\sigma_{1|1}^2}}\exp - \left(\frac{(x_1-\mu_{1|1})^2}{2\sigma_{1|1}^2}\right) \text{ (based on Eq. (A.17))}
\end{aligned}
$$

$$= \frac{1}{\sqrt{2\pi}\sigma_{1|1}} \exp - \left(\frac{(x_1 - \mu_{1|1})^2}{2\sigma_{1|1}^2} \right) \tag{A.19}$$

$$\propto \exp - \left(\frac{(x_1 - \mu_{1|1})^2}{2\sigma_{1|1}^2} \right) \tag{A.20}$$

Based on the above final result, we can conclude that the posterior probability density of the state is Gaussian, with mean $\mu_{1|1}$ and variance $\sigma_{1|1}^2$.