Distributed Estimation of a class of Nonlinear Systems

by

Derek Heungyoul Park

A thesis submitted to the Graduate Program in Chemical Engineering
in conformity with the requirements for the
Degree of Master of Applied Science

Queen's University
Kingston, Ontario, Canada
December 2012

Abstract

This thesis proposes a distributed observer design for a class of nonlinear systems that arise in the application of model reduction techniques. Distributed observer design techniques have been proposed in the literature to address estimation problems over sensor networks. In large complex sensor networks, an efficient technique that minimizes the extent of the required communication is highly desirable. This is especially true when sensors have problems caused by physical limitations that result in incorrect information at the local level affecting the estimation of states globally. To address this problem, scalable algorithms for a suitable distributed observer have been developed. Most algorithms are focussed on large linear dynamical systems and they are not directly generalizable to nonlinear systems. In this thesis, scalable algorithms for distributed observers are proposed for a class of large scale observable nonlinear system.

Distributed systems models multi-agent systems in which each agents attempts to accomplish local tasks. In order to achieve global objectives, there should be agreement regarding some commonly known variables that depend on the state of all agents. These variables are called consensus states. Once identified, such consensus states can be exploited in the development of distributed consensus algorithms. Consensus algorithms are used to develop information exchange protocols between agents such that global objectives are met through local action. In this thesis, a higher order observer is applied in the distributed sensor network system to design a distributed observer for a class nonlinear systems. Fusion of measurement and covariance information is applied to the higher order filter as the first method. The consensus filter is embedded in the local nonlinear observer for fusion of data. The second method is based on the communication of state estimates between neighbouring sensors rather than fusion of data measurement and covariance. The second method is found to reduce disagreement of the states estimation between each sensor. The performance of these new algorithms is demonstrated by simulation, and the second method is effectively applied over the first method.

Acknowledgement

First and foremost, I would like to express my deep gratitude to Professor Martin Guay for his patient guidance, enthusiastic encouragement and useful critiques of this research work. Professor Guay gives his students plenty freedom and space, and always encourages us to find and appreciate the challenges that our work offers. His approach to discipline for students is unparalleled and creates a great research environment in which high standards are both expected and achieved. I really appreciate the trust that he has shown me for the past two years. His inspiring perspectives, the coffees and the laughs have helped me more than I can say. I would also like to thank Professor Abdol-Reza Mansouri for wonderful instruction and valuable insights, and for inspiring my research. I thank Professor James McLellan for his patient guidance throughout his course as well.

Thanks, too, to my officemates in G37 - especially to Ehsan, Sean and Scott. Thank you all for sharing your passion, life, excitement and even frustration. You have all encouraged me to learn something new to expand my view widely. Thanks also to Calista and Jennifer: You made me feel home in Kingston. Thanks to Walter, Kai, and Key for fun conversations and making the office such a comfortable place to work and learn.

I also want to thank Kevin and Robert, who have helped me to explore my academic life, to understand North American culture and to develop my writing and communication skills in English. Thanks to Dennis and Noah for cheering me up on those days when life in Canada seemed trying. To Tim, Gerry, Dan, Chad, Adrian, Daniel, Alisha, Stacy, James, and Pat for making Kingston and Canada an enjoyable place to live and travelâĂŤfor making it seem like my home. Thanks to all the volleyball members for have a great time with playing together. Thanks to all

I thank my family and friends back home for their support and trust in me, and especially to my parents, who have supported me with unconditional love. Without their support, this would not have been possible. I hope they are proud of me. To my friends, thank you for trusting me and giving me valuable encouragement. Your friendship is valued beyond measure.

Table of Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

There are considerable challenges associated with the estimation of fluid flow governed by the Navier Stokes (NS) equations. If one considers the problem of estimating velocity fields from point measurement, it is difficult (or even impossible) to guarantee the observabilty of NS flow. One way to handle this problem is to use model reduction techniques. By using this approach, the more challenging problem of guaranteeing observabilty of NS flow can be circumvented if the finite dimensional model can be shown to be observable. In [Guay et al., 2010], the design of an observer was introduced for the estimation of reduction fields in Navier-Stokes flow in building system. This method was based on a Proper Orthogonal Decomposition (POD). The POD technique identifies particular modes of the flow dynamics that can approximate the overall behaviour. Projection of the NS equation of those modes yields a finite dimensional approximation that is amenable to observer design. The use of sensors is a primary concern in the design of an observer. Utilizing sensors, however, presents

challenges that include limited energy sources to operate, determining optimal location of sensors, the number of sensors, and various communication problems that can arise between sensors.

In recent years, sensor networks have garnered increased attention from researchers. This technology has been used in multiple industries that include military, law enforcement, agricultural and forestry-based projects, surveillance, and even information collection. Typically sensor networks are based on a distributed system over a spatial area with multiple sensors at various locations. Each sensor communicates with its neighbours to exchange data and collect information. In the ideal scenario, all sensors are required to communicate to all other sensors to guarantee accurate measurements. This has advantages that include low dimension and fast implementation. In spite of the advantages, when there are a significant number of sensors, it is not efficient to communicate all-to-all links. This thesis attempts to develop new consensus distributed algorithms for a class of nonlinear system. This method, first suggested in [Olfati-Saber and R.M.Murray, 2003], is applied to develop the distributed consensus observer.

The main purpose of this thesis is to design a nonlinear observer for a distributed system. A second order observer with an embedded consensus filter for a class of nonlinear distributed systems is developed. An alternative distributed second order filtering algorithm is presented that directly uses consensus on state estimation.

1.2 Organization of the Dissertation

CHAPTER 2: In this chapter, we introduce technical preliminaries and set up the basic notations. The topics include proper orthogonal decomposition (POD) modes,

Lyapunov stability, observability, state observers, and graph theory. A review of the previous and present research in the field of state estimation for a class of distributed nonlinear systems is also presented.

CHAPTER 3: In this chapter, a finite dimensional approximation of velocity field, contaminant field and energy field in NS flow is developed. The Galerkin projection approach based on POD modes is used to develop a finite dimensional approximation of NS flow dynamics, component balance and energy balance. A higher order observer is designed to estimate the states of the nonlinear system subject to discrete noisy measurement. The simulation result demonstrates the effectiveness of proposed technique.

CHAPTER 4: In this chapter, a decentralized estimation in distributed systems is considered to develop distributed estimation algorithms with multiple sensors. We consider the distributed Kalman filter (DKF) systems with embedded consensus filters. We consider low-pass and band-pass consensus filters for fusion data. We also consider a DKF that builds consensus on state estimation in linear systems. Using these techniques, we develop a new class of consensus observers for nonlinear systems. Simulation results are presented to show the performance of the distributed observer.

CHAPTER 5: We summarize the design procedure developed in Chapter 3 and 4. The conclusion of the procedure is addressed, and suggestions for future research, such as the designing a controller based on this state estimation are discussed.

Chapter 2

Preliminaries

In this chapter, we present some preliminaries and background information. We also present a brief literature review of distributed observers, and nonlinear observers. In section 2.1, we introduce the necessary technical preliminaries which include concepts from linear systems theory, nonlinear observer design, decentralized and distributed systems. In section 2.2, we review the literature on the design of the nonlinear observers and distributed observers with consensus filters used to estimate unmeasured states in a distributed system.

2.1 Technical Preliminaries

2.1.1 Proper Orthogonal Decomposition (POD) Modes

POD is a method used to compute a low-dimensional approximation of large data sets such as those arising from image processing, fluid flow or large-scale system dynamical systems.

Let $\mathcal{H}$ be a Hilbert space (i.e complete inner product space) with inner product $\langle \cdot, \cdot \rangle$. We set a representative of the system dynamics as a data ensemble, $\{u_k\}$ $(k = 1, \ldots, m)$ in which each element to a Hilbert space. The data ensemble is expressed by $\{u_k \in \mathcal{H} \mid k = 1, \ldots, m\}$ where the ensemble $\{u_k\}$ consists of experiments that highlight different features of the dynamical system's behaviour. These experiments are developed by a set of snapshots of the states u_k at time t_k.

POD generates an orthonormal basis of dimension n $(n < m)$ that generates a low-dimensional subspace of $\mathcal{H}$, denoted by $\mathcal{S}$. The projection of the data ensemble u_k is written as $P_{\mathcal{S}} u_k$ where $P_{\mathcal{S}}$ is the orthogonal projection operator onto $\mathcal{S}$. To find the subspace $\mathcal{S}$, one minimizes the error $\| u_k - P_{\mathcal{S}} u_k \|$, where $\| \cdot \|$ is the induced norm. Given a data ensemble $\{u_k\}$, the POD basis can be simply computed by the solving the following eigenvalue-eigenvector decomposition, [Lumley, 1970].

$$U\phi = \lambda\phi \tag{2.1}$$

where $U : \mathcal{H} \to \mathcal{H}$ is the linear operator defined by:

$$U = u_k \otimes u_k^* \tag{2.2}$$

The variable $u_k^* \in \mathcal{H}^*$ is the adjoint of $u \in \mathcal{H}$ and $\mathcal{H}^*$ is the space of functionals $u^*(\cdot) = \langle \cdot, u \rangle$. Since $\otimes$ is the standard tensor product, the following property, $(u \otimes v^*)(w) = u\langle w, v \rangle$ for any u, v and w in $\mathcal{H}$, is satisfied. In general, the snapshots u_k are taken at time t_k at a finite number of sensors N over the spatial domain Ω where N can be a significantly large number.

POD modes $\phi(x)$ are taken as linear combinations of the elements of the ensemble

$\{u_k\}$ (with $k = 1, \ldots, m$ where $m << N$) .

$$\phi(x) = \sum_{k=1}^{m} c_k u_k \tag{2.3}$$

This combination is not arbitrary since the linear operator U is constructed by elements in the span of the ensemble $\{u_k\}$. Eq. (2.1) can be expressed as follows:

$$Rc = \lambda c \tag{2.4}$$

where R is a m by m matrix with elements $R_{ij} = \frac{1}{m}\langle u_i, u_j \rangle$. The problem is then reduced to the solution of an m dimensional eigenvalue-eigenvector decomposition. Normally snapshots can be obtained using CFD simulation of the process, or through experiments placing observed by sensors at predefined locations. In most studies, the snapshots are generated by CFD models.

2.1.2 Lyapunov Stability

Stability theory plays a critical role in dynamical systems theory and engineering. In general, the method of Lyapunov can be widely used to large classes of nonlinear and linear dynamical systems, including time-invariant or time-varying systems for both continuous-time and discrete-time systems.

An equilibrium point is said to be Lyapunov stable if all solutions of the dynamical system that starts at nearby points stay nearby permanently; otherwise, it is unstable. It is asymptotically stable if all solutions of the dynamical system that starts at nearby points converge to the equilibrium point [Khalil, 2002].

Consider a dynamical system which satisfies

$$\dot{x} = f(t, x) \qquad x(t_0) = x_0 \qquad x \in \mathscr{R}^n \tag{2.5}$$

Assuming that $f(t, x)$ satisfies the standard conditions for the existence and uniqueness of solutions. At equilibrium point x_e, the condition $f(t, x_e) = 0$ is satisfied.

Definition 2.1.1. *[Khalil, 2002] Let the equilibrium point be x_e,*

- *stable if there is $\delta = \delta(\epsilon) > 0$ for each $\epsilon > 0$ such that*

$$\|x(x_e)\| < \delta \Rightarrow \|x(t) < \epsilon\|, \forall t \geq 0$$

- *unstable, otherwise*

- *asymptotically stable if it is stable and δ can be satisfied as,*

$$\|x(x_e)\| < \delta \Rightarrow \lim_{t\to\infty} x(t) = x_e$$

Denote that $V(\cdot) : D \to R$ is a non-negative differentiable function defined in a domain $D \subset R^n$. The derivative of V along the trajectories of (2.5) is written,

$$\dot{V}(t, x) = \frac{\partial V}{\partial t} + \frac{\partial V}{\partial x} f(t, x) \tag{2.6}$$

By this definition, the following theorem allows us to determine stability for a system. Briefly described, the theorem indicates that when $V(t, x)$ is a locally positive definite function and $\dot{V}(t, x) \leq 0$, we can then come to the conclusion that the equilibrium point is stable.

Theorem 2.1.1. ***Basic theorem of Lyapunov***

Let x_e be an equilibrium point and $D \subset R^n$ be a domain with x_e. Let $V : D \to R$ be a continuously differentiable function

$$V(x_e) = 0 \text{ and } V(x) > 0 \text{ in } D - \{0\} \tag{2.7}$$

$$\dot{V} \leq 0 \text{ in } D \tag{2.8}$$

then, $x = x_e$ is stable, if

$$\dot{V}(x) < 0 \text{ in } D - \{0\} \tag{2.9}$$

then $x = x_e$ is asymptotically stable

2.1.3 Observability

The study of observer design is of great importance in the study of dynamical systems. The key property of a dynamical system that is required for the design of observers is the observability of the system. In what follows, we provide a definition of observability and provide some conditions that are necessary and sufficient for the observability of a widely applied class of discrete-time linear systems systems with imperfect measurements. This class of systems can be written in general form as follows:

$$x_{k+1} = Ax_k + Bu_k \tag{2.10a}$$

$$y_k = Cx_k \tag{2.10b}$$

where $x \in \mathbb{R}^n$ is a state vector, $u \in \mathbb{R}^n$ is the control input, $y \in \mathbb{R}^m$ are the outputs. The matrices, A, B and C are of appropriate dimensions and define the system's dynamics. Observability is a property of a dynamical system, first introduced by [Kalman, 1960], that expresses the ability to reconstruct or make inferences regarding the values of unmeasured state variables using available measurements.

Definition 2.1.2. *[Simon, 2006] A linear discrete-time system* (2.10) *is observable if for any initial state* x_0 *and some final time* t *the initial state* x_0 *can be uniquely determined by knowledge of the input* u_k *and output* y_k *for all* $k \in [0, t]$.

Observability is a property of the system which states that the values of the internal states can be inferred by the output y over some time interval. If a system is observable then the initial state x_0 can be determined. If the initial state can be known then all states for the time interval $k \in [0, t]$ can be calculated. Therefore, observability implies that all states between the initial and final times, $k \in [0, t]$ can be reconstructed as long as the inputs and outputs are known exactly.
Observability can be checked by a matrix rank test performed on the system's *observability matrix* or *observability Gramian* as below:

Theorem 2.1.2. *The m-state discrete linear time-invariant system* (2.10) *has the observability* F *defined by*

$$F_{C,A} = [C^T, (CA)^T, \ldots, (CA^{n-1})^T]^T$$

The system (2.10) *is observable if and only if* $rank(F) = n$.

Theorem 2.1.3. *The n-state discrete linear time-invariant system* (2.10) *is observable if and only if the observability Gramian defined by*

$$\sum_{k=0}^{t}(A^T)^k C^T C A^k$$

is positive definite for some $t \in (0, \infty)$.

2.1.4 State Observers

In control theory, the estimation of unmeasured states from past values of the output y is an important consideration for controller design [Dullerud and Paganini, 2000]. A state observer is a dynamical system derived from a system model that is used to estimate its internal states. In theory, all the states variable of the system can be estimated using an observer if and only if the system is observable. Let $\hat{x}_k$ represent an estimate of the state x_k. An observer is designed to ensure that the estimation error $e_k = (x_k - \hat{x}_k)$ tends to zero as k goes to infinity. Assuming that the dynamics of the plant is a linear time-variant system, an observer can be expressed as,

$$\hat{x}_{k+1} = A\hat{x}_k + L(y_k - \hat{y}_k) + Bu_k \tag{2.11}$$

$$\hat{y} = C\hat{x}_k + Du_k \tag{2.12}$$

This observer given by (2.11) and (2.12) is known as a *Luenberger observer* [Dullerud and Paganini, 2000]. The matrix L is designed such that the matrix $(A - LC)$ has all eigenvalues inside the unit circle. The stability of $(A - LC)$ guarantees the stability of the observer's error dynamics. The rate of the convergence of the estimation error e_k to the origin can be arbitrarily chosen by modifying the observer gain matrix

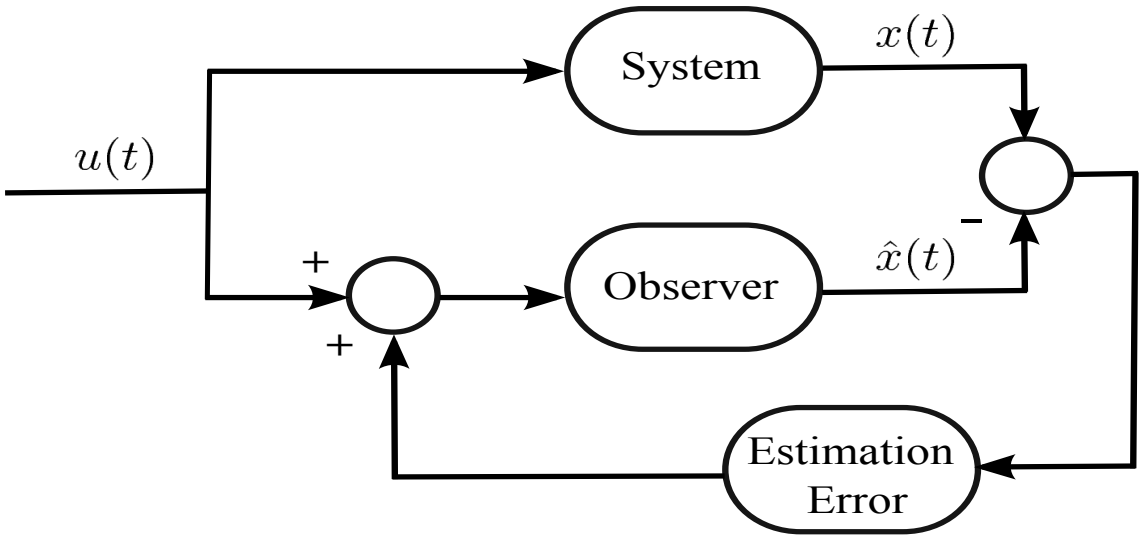

Figure 2.1: [Tin and Poon, 2005]General Structure of the Luenberger Observer

L. This is valid for any initial condition $x(0)$ and any matrix A such that (C, A) is an observable pair. Figure 2.1 shows how a Luenberger observer is implemented to estimate unknown states. The difference between the measured outputs and the predicted outputs are used to modify the state estimates.

2.1.5 Graph Theory

A graph consists of a collection of vertices and edges. Each vertex is linked to other vertices by edges. The pictorial representation of a graph, as depicted in Figure 2.2, is not particularly meaningful in sensor networks, but the relationship between vertices and edges provides some valuable information in the design of consensus algorithms. An edge provides the relationship between vertices, and it is necessary to have vertices at the end of each edge. However, the vertices are not necessarily connected to all of the edges.[Godsil and Royle, 2001] The vertex set is denoted by $V(G)$, and the edge

set is denoted by $E(G)$. The degree of a graph G is the number of incoming edges to the vertices. When G has a well defined degree, we assume that all edges are directed and the graph is said to be directed. If the graph is not directed, its order does not exist.

At a discrete time, t_k, a directed graph can be described as $G_n(k) \triangleq (V_n, E_n(k))$ where $V_n = \{1, \ldots, n\}$and $E_n \subseteq V_n \times V_n$ are vertex set and the edge set respectively and $|V_n|$ and $|E_n|$ are called scale and size for complex networks. [Olfati-Saber and Shamma, 2005]. A *spanning tree* of a connected and directed graph G_n is a tree consisting of all the vertices and some (or perhaps all) of the edges of G_n. When all the vertices are connected, a graph is said to be *strongly connected* and it is *balanced* when each vertex has the same number of incoming edges. For example, if there are five vertices denoted *a,b,c,d,e* in Fig 2.2, the graph has a directed spanning tree, but it is not strongly connected.

An adjacency matrix is a matrix which denotes the relationship between vertices that are adjacent to each other. It is denoted by $\mathbf{A}_n(k) \in \mathbb{R}^{n \times n}$ where entry a_{ij} is 1 or 0 depending on whether vertices are connected or not. We write $a_{ij}(k) = 1$ if there is an incoming edge from j to i, otherwise $a_{i,j}(k) = 0$.

Definition 2.1.3. *For a graph G_n with vertices $v_1, v_2, \ldots, v_n$, the adjacency matrix of G_n is the $n \times n$ matrix $\boldsymbol{A}$ whose (i,j) entry is:*

$$\boldsymbol{A}_{i,j}(k) = \begin{cases} 1 & \text{if } v_i \text{ and } v_j \text{ are adjacent} \\ 0 & \text{otherwise} \end{cases}$$

The graph in Figure 2.2 has 5 vertices, such that the adjacency matrix, $\mathbf{A}$ is

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

Through the adjacency matrix, we can obtain the Laplacian matrix $\mathbf{L}_n(k) \in \mathbb{R}^{n \times n}$ whose the element *i,j*th are defined as follows.

Definition 2.1.4. *[Godsil and Royle, 2001; Casbeer, 2009] For a graph G_n with vertices $v_1, v_2, \ldots, v_n$, the Laplacian matrix of G_n is the $n \times n$ matrix $\boldsymbol{L}$ whose (i,j) entry is given by:*

$$\boldsymbol{L}_{i,j}(k) = \begin{cases} deg(V_i) & \text{if } i{=}j \\ -1 & \text{if } i{\neq}j \text{ and } v_i \text{ is adjacent to } v_j \\ 0 & \text{otherwise} \end{cases}$$

the Laplacian matrix of G_n in Figure 2.2, $\mathbf{L}$ is

$$\mathbf{L} = \begin{bmatrix} 3 & -1 & -1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 2 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 \\ 0 & -1 & 0 & -1 & 2 \end{bmatrix}$$

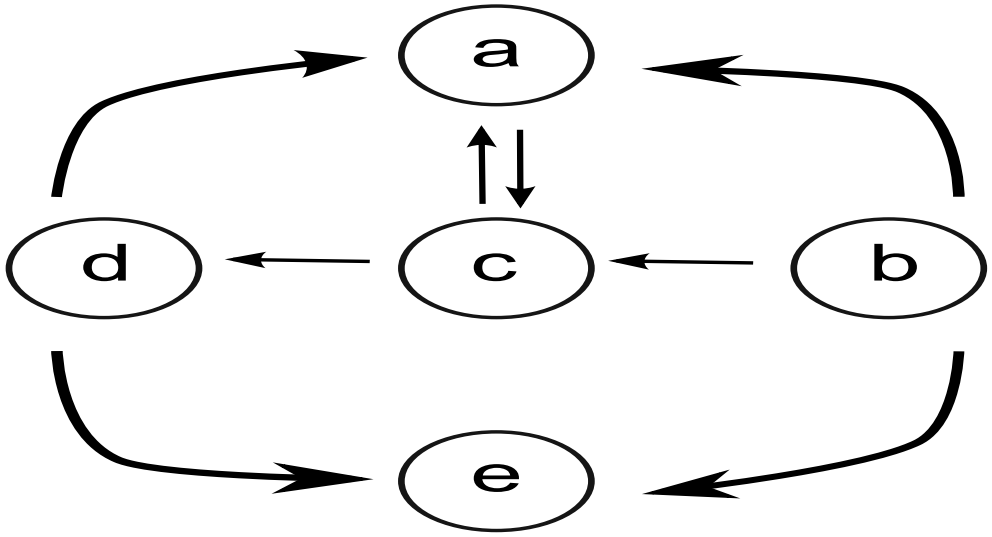

Figure 2.2: A five vertices graph : There is a path from vertices to every other vertices in the graph, however there is no path from vertices [e] to any other node. Hence, the graph has a directed spanning tree, but it is not strongly connected.

2.2 State Estimation in Distributed System

The estimation of states requires measurements by sensors. Occasionally, sensors may have problems caused by physical restrictions that include an ineffective power

source, or a lack of accessibility. These problems can seriously affect the estimation of states. This is especially true in a distributed system, where all sensors need to communicate to achieve common goals. To improve efficiency of the estimation of the states, new algorithms have been developed for sensor networks in a distributed system. Distributed sensor networks are systems that allow for the sharing of information between sensors. Such networks are amenable to the reconstruction of network wide information without the need for all-to-all sensor information. The reconstruction of network wide information can be achieved through the design of local filters that exploit the structure of the network. In a number of applications, sensor networks are based on a distributed system over a spatial area with multiple sensors. In such cases, the sensors communicate with each other to exchange data, process the system, and collect information. One takes advantage of the exchange of information to improve local information while generating a more accurate global representation of the dynamics. This property can be preserved even in the presence of loss of information, uncertainties, and disturbances. This class of distributed observers could have multiple applications for military and forestry-based projects, as well as developing greater surveillance and information collection methods. For example, we can use RFID chips as sensors to monitor all the condition of crops based on distributed sensor network. Also, there are many foreseeable potential applications in chemical engineering, such as fluid flow estimation.

While extensive research has been conducted with respect to flow control and optimization, flow estimation by an observer has not received the same attention, and is not broadly used in applications. The knowledge of estimation of the velocity field however, is useful to control air quality or monitor contaminant. Since flow systems,

typically described by Navier-Stoke(NS) equations, are nonlinear, a linearized version of the equation is usually considered. The principle of using a linearized version of NS for the purpose of estimation was introduced in [Hoepffner et al., 2005, 2006], which allowed one to design an infinite dimensional Kalman filter, that can be discretized to produce a finite dimensional filter. The discretization of an infinite dimensional Kalman filter can lead to large and complex finite-dimensional dynamics. Additionally, since the NS equations are nonlinear, information related to nonlinearity may be lost. Although the use of the linearized NS can yield an effective way to estimate the states, one must also consider the problem of nonlinear design for NS flow in an attempt to capture its nonlinear behaviour.

Proper Orthogonal Decomposition(POD) techniques was first introduced in [Lumley, 1970]. This approach avoids the need for linearization, and preserves the nonlinearity of the system. The infinite dimensional nonlinear system is simplified by providing a low dimensional representation that approximates the local behaviour of the system. The finite dimensional dynamical system is obtained by using the Galerkin Projection technique based on the projection of NS equations onto the finite dimensional space identified by POD, as demonstrated in [Berkooz and Titi, 1993; Rowley et al., 2004]. The Galerkin Projection technique on POD modes has been extensively studied. The resulting simplified nonlinear model can be used to express the distributed state variables of the reduced order equations as a linear combination of the POD modes. The Galerkin Projection is an effective way to address the nonlinearity of the complex system in the form of a low dimensional set of nonlinear ordinary differential equations.

POD is one of the most well-known methods for reducing the order of a model.

It has been applied in many fields such as control, optimization, and flow estimation [Rowley et al., 2004]. [Guay and Hariharan, 2008] applied the POD based estimation technique to flow estimation in a building system. Some modification of the POD methods have been introduced in [Bleris and Kothare, 2005; Noak et al., 2004, 2005].

In [Guay and Hariharan, 2008; Guay et al., 2009, 2010], POD based approaches for the estimation of velocity field and contaminant flow in a building system was proposed. This estimation provides a possible method for the design of distributed observers in complex large-scale flow. In [Yu et al., 2009], a centralized scheme and a decentralized observer/controller design technique were discussed. This research suggested that these decentralized methods can have several advantages in a large-scale system that include low dimension, fast implementation, and low cost compared to centralized approaches. This however, is not an efficient method for distributed systems. Earlier research conducted by [Mutambara, 1998; Speyer, 1978] focused on the combination of the state estimates of a system with multiple sensors in decentralized estimation into a single central estimate. The technique required that all sensors needed to communicate to come up with a single estimate. This intensive communication between sensors is not the most appropriate approach for distributed systems with multiple sensors. In [Acikmese and Mandic, 2011; Olfati-Saber and Shamma, 2005], distributed estimation method in a linear system was introduced with a large number of sensors. This research considered a fusion of data of measurements or state estimates from neighbouring sensors by using a Kalman filter. The distributed filter consists of a network of micro-Kalman filters, each embedded with a low-pass and a band-pass consensus filter that can be used to collect and compare local information in order to generate network-wide consensus.

Chapter 3

Nonlinear filter

This chapter presents the development of a second order observer that estimates the states of a nonlinear plant based on discrete noisy measurements. This second order observer is suitable for application in the estimation of NS flow using POD based techniques. As discussed in chapter 2, this technique provides low dimensional subspace including the original nonlinearity. This reduction technique results in a nonlinear ordinary differential equations that can be effectively treated using a particular class of higher order filters as proposed in [Athans et al., 1968].

In section 3.1, we consider the application of POD based techniques for the approximation of NS flow, with component and energy balance. The nonlinear system of ODEs resulting from this technique can be effectively dealt with using the higher order filter presented in section 3.2. A simulation study is presented in section 3.3 to demonstrate the effectiveness of this technique.

3.1 States Estimation

3.1.1 Velocity Field Estimation

In this chapter, we present the estimation of the velocity fields in flow system in this section based on POD modes.It is assumed that the flow velocity is governed by the compressible Navier-Stokes equation below [Bird et al., 1960]:

$$\begin{aligned} div(v) &= 0 \\ \frac{\partial v}{\partial t} &= -(v \cdot \nabla)v + \nu \nabla^2 v - \nabla p \end{aligned} \tag{3.1}$$

where $v : \Omega \times \mathbb{R} \to \mathbb{R}^3$ indicates velocity field on spatial domain Ω, p is the pressure term, $\nu = 1/Re$, Re is the Reynolds number. It is assumed that velocity and pressure field are defined on closed-subset of $\mathbb{R}^3$. The equation (3.1) has a scaled formulation of the Navier-Stokes equation where the velocities are scaled by a factor V, time by V/L, pressure by ρV^2 where ρ is the density and the viscosity by $\rho V L$, V and L are nominal velocities and length. Although the assumption of incompressible flow is considered for the modeling of the airflow in a commercial building, a similar approach can be applied to model reduction in compressible flow (as mentioned in [Rowley et al., 2000, 2004; Rowley, 2005]).

POD based Model Reduction

In [Guay et al., 2010], the velocity field, v represented as an expansion form, $v(t,x)$ in POD modes $\alpha(x)$ defined on the spatial domain Ω is written as:

$$v(t,x) = \sum_{i=1}^{n} a_i(t)\alpha_i(x) \tag{3.2}$$

The projection onto the modes requires the definition of an inner product over a Hilbert space $\mathcal{H}$ defined by:

$$< v_i, v_j >= \int_{\Omega} v_i(x) \cdot v_j(x) dV \tag{3.3}$$

where $v_i(x) \cdot v_j(x)$ is the standard inner product between two vectors $v_i(x)$ and $v_j(x)$ in Euclidean space, dV is a volume element on $\mathbb{R}^3$

The basic concept of the POD based model reduction is to recast the Navier-Stokes equation using the expression (3.2). This gives by substitution of (3.2) in (3.1):

$$\begin{aligned} \frac{\partial v}{\partial t} = \sum_{i=1}^{n} \dot{a}_i(t)\alpha_i(x) = &- \left(\sum_{j=1}^{n} a_j(t)\alpha_j(x) \cdot \nabla \right) \sum_{i=1}^{n} a_i(t)\alpha_i(x) \\ &+ \nu \sum_{i=1}^{n} a_i(t)\nabla^2\alpha_i(x) - \nabla p \end{aligned} \tag{3.4}$$

Projection of (3.4) onto the POD modes $\alpha_i(x)$ by applying inner product yields:

$$\begin{aligned} < \frac{\partial v}{\partial t}, \alpha_i(x) > =& < \sum_{k=1}^{n} \dot{a}_k(t)\alpha_k(x), \alpha_i(x) > \\ &= - < \left(\sum_{j=1}^{n} a_j(t)\alpha_j(x) \cdot \nabla \right) \sum_{k=1}^{n} a_k(t)\alpha_k(x), \alpha_i(x) > \\ &+ \nu < \sum_{k=1}^{n} a_k(t)\nabla^2\alpha_k(x), \alpha_i(x) > - < \nabla p, \alpha_i(x) > \end{aligned} \tag{3.5}$$

The modes are such that,

$$< \alpha_i(x), \alpha_j(x) >= \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \tag{3.6}$$

and $div(\alpha_i) \equiv 0$ for i=1, ...,n.

Therefore, (3.5) reduces to:

$$\begin{aligned} \dot{a}_i(t) < \alpha_i(x), \alpha_k(x) > =& - \sum_{i=1}^{n}\sum_{j=1}^{n} a_i(t)a_j(t) < (\alpha_j(x) \cdot \nabla)\alpha_i(x), \alpha_k(x) > \\ &+ \nu \sum_{i=1}^{n} a_i(t) < \nabla^2\alpha_i(x), \alpha_k(x) > - < \nabla p, \alpha_k(x) > \end{aligned} \tag{3.7}$$

The following dynamical system is obtained:

$$\begin{aligned} \dot{a}_k(t) =& - \sum_{i=1}^{n}\sum_{j=1}^{n} a_i(t)a_j(t) < (\alpha_j(x) \cdot \nabla)\alpha_i(x), \alpha_k(x) > \\ &+ \nu \sum_{i=1}^{n} a_i(t) < \nabla^2\alpha_i(x), \alpha_k(x) > - < \nabla p, \alpha_k(x) > \end{aligned} \tag{3.8}$$

Equation (3.8) is the decomposition model of the velocity field on the POD modes. This finite dimensional approximation provides the basis for the design of an observer

for fluid flow dynamic system.

Reduced order dynamical Velocity State

The dynamical system (3.8) is a quadratic differential equation of the general form:

$$\dot{a}_k(t) = N_k a(t) + a(t)^T P_k a(t), \quad k = 1, \ldots, n \tag{3.9}$$

where N_k is an n-dimensional row vector and P_k is an $n \times n$ matrix for $k = 1, \ldots, n$,

$$N_{k_i} = \nu < \nabla^2 \alpha_i(x), \alpha_k(x) > \tag{3.10}$$

$$P_{k_{ij}} =< (\alpha_j(x) \cdot \nabla)\alpha_i(x), \alpha_k(x) > \tag{3.11}$$

The pressure term vanishes on the closed boundary since $\alpha_k(x) = 0$ on the boundary of $\Omega, \partial\Omega$,

$$\int_\Omega \nabla p \cdot \alpha_k(x) dV = \int_{\partial\Omega} p\alpha_k(x) \cdot \mathbf{n}_\Omega dS \tag{3.12}$$

where $\mathbf{n}_\Omega$ is the unit vector normal to the spatial domain Ω.

Measurement

Assume that velocity field measurements,v_0, are available at predefined locations,x_0. These measurements can be expressed using the expansion (3.2). For example, consider the measurement of the average velocity, $v_{avg}(t, x_0) = v(t, x_0) + u(t, x_0) + w(t, x_0)$

at the location x_0, then the expansion form yields:

$$v_{avg}(t, x_0) = \sum_{i=1}^{n} a_i(t)(\alpha_i^1(x_0) + \alpha_i^2(x_0) + \alpha_i^3(x_0)) \tag{3.13}$$

where $\alpha_i^j(x)$ is the j^{th} element of the i^{th} POD modes. Since the POD modes are assumed to be time-independent, equation (3.13) is expressed in the form

$$v_{avg}(t, x_0) = Ca(t) \tag{3.14}$$

where C is $1 \times n$ measurement matrix and $a(t)$ is the n-dimensional vector of time varying coefficients of the Galerkin approximation of $v(t, x)$. The measurement output is re-expressed in the general form as

$$y_{velo}(t) = Ca(t). \tag{3.15}$$

The complete dynamical system of velocity with the available measurements is given by:

$$\begin{aligned} \dot{a}_k(t) &= N_k a(t) + a(t)^T P_k a(t), \ k = 1, \cdots, n, \\ y_{velo}(t) &= Ca(t) \end{aligned} \tag{3.16}$$

If we assume that the POD modes give an accurate explanation of the character of the flow fluid then the Galerkin coefficient $a(t)$ provide an estimate of the velocity field via the expansion (3.2). If the initial value of the coefficients $a_i(0)$ are known, then the actual values $a_i(t)$ can be calculated. The expansion (3.2) can then be used to approximate the velocity field. Since the initial values of the coefficients

are generally unknown, an observer is required to estimate them. In [Rowley and Juttijudata, 2005], an observer for the estimation of the Galerkin coefficients was introduced. Rowley and Juttijudata [2005] considered the linear approximation of (3.16) to design an observer. In [Guay et al., 2010], a nonlinear observer is designed that can improve the performance of the resulting estimation scheme.

3.1.2 Estimation of Contaminant Flow

The principle of contaminant flow estimation is similar to the estimation of velocity fields. We estimate concentration fields using the velocity field estimation. Assuming that diffusivity of the contaminant is constant, the concentration field is governed by the advection-diffusion equation given by:

$$\begin{aligned} div(v) &= 0 \\ \frac{\partial c}{\partial t} &= -(v \cdot \nabla)c + \kappa \nabla^2 c + J_s \end{aligned} \tag{3.17}$$

where $c : \Omega \times \mathbb{R} \to \mathbb{R}$ indicates the concentration field on spatial domain Ω, $J_s : \Omega \times \mathbb{R} \to \mathbb{R}$ describes "sources" or "sinks" of the quantity c. $\kappa = D/UL$ with D, diffusivity coefficient.

The equation (3.17) is a scaled formulation of the advection-diffusion equation. The velocities are scaled by a factor V and time is scaled by V/L, where V and L are nominal velocities and length. The concentration and sources/sinks fields are dimensionless since they are treated as the mass fraction of the contaminant.

POD based Model Reduction

The concentration field is expanded $c(t,x)$, in the POD modes $\phi(x)$ defined on the spatial domain Ω as follows:

$$c(t,x) = \sum_{i=1}^{p} b_i(t)\phi_i(x) \tag{3.18}$$

As in the treatment of the velocity field, we consider the projection over a Hilbert $\mathcal{H}$ with inner product:

$$< c_i, c_j >= \int_\Omega c_i(x) \cdot c_j(x) dV \tag{3.19}$$

where $c_i(x) \cdot c_j(x)$ is the standard dot product between two vectors $c_i(x)$ and $c_j(x)$ in Euclidean space and dV is a volume element.

Substituting both equations (3.2) and (3.18) into (3.17), one obtains

$$\begin{aligned}\frac{\partial c}{\partial t} = \sum_{i=1}^{p} \dot{b}_i(t)\phi_i(x) = &- \left(\sum_{j=1}^{n} a_i(t)\alpha_i(x)\right) \cdot \nabla \left(\sum_{i=1}^{p} b_i(t)\phi_i(x)\right) \\ &+ \kappa\nabla^2 \left(\sum_{i=1}^{p} b_i(t)\phi_i(x)\right) + J_s \end{aligned} \tag{3.20}$$

Projecting (3.20) onto POD modes $\phi_i(x)$ yields:

$$\begin{aligned} < \frac{\partial c}{\partial t}, \phi_l(x) > =& < \sum_{i=1}^{p} \dot{b}_i(t)\phi_i(x), \phi_l(x) > \\ =& < \sum_{i=1}^{n}\sum_{j=1}^{p} a_i(t)b_j(t)\nabla\phi_j(x) \cdot \alpha_i(x), \phi_l(x) > \\ &+ \kappa < \nabla^2 \left(\sum_{i=1}^{p} b_i(t)\phi_i(x)\right), \phi_l(x) > + < J_s, \phi_l(x) > \end{aligned} \tag{3.21}$$

By the linearity of the inner product and the orthogonality of the POD modes, we can compute the following reduced form:

$$\dot{b}_l(t) = -\sum_{i=1}^{n}\sum_{j=1}^{p} a_i(t)b_j(t) < \nabla\phi_j(x)\cdot\alpha_i(x), \phi_l(x) > + \kappa\sum_{i=1}^{p} b_i(t) < \nabla^2\phi_i(x), \phi_l(x) > + < J_s, \phi_l(x) > \tag{3.22}$$

Equation (3.22) is the decomposition model of concentration field in POD modes. Equations (3.8) and (3.22) are the basis of designing an observer for both the fluid flow velocity field, $v(t,x)$ and the concentration field, $c(t,x)$.

Reduced order Component Balance

The dynamical system (3.22) is simplified to a quadratic differential equation of the form:

$$\dot{b}_l(t) = M_l b(t) + a(t)^T Q_l b(t) \quad l = 1,\ldots,p \tag{3.23}$$

where M_l is a p-dimensional row vector and Q_l is an $p\times p$ matrix for $l = 1,\ldots,p$,

$$M_{l_i} = \kappa < \nabla^2\phi_i(x), \phi_l(x) > \tag{3.24}$$

$$Q_{l_{ij}} =< \nabla\phi_j(x)\cdot\alpha_i(x), \phi_l(x) > \tag{3.25}$$

In general, the source/sink term is ignored since the POD modes are such that div(ϕ)=0 below,

$$\int_{\Omega} J_s \cdot \phi_l(x) dV = \int_{\partial\Omega} J_s \phi_l(x) \cdot \mathbf{n}_{\Omega} dS \tag{3.26}$$

where $\mathbf{n}_{\Omega}$ is the unit vector normal to the spatial domain Ω. The source/sink term vanishes on the closed boundary such that $\phi_l(x) = 0$ on the boundary of $\Omega, \partial\Omega$.

Measurement

Assuming that concentration field measurements, $c_0(t, x_0)$, are available at predefined location, x_0, one can express these measurements by using the expansion form as follows:

$$c(t, x_0) = \sum_{i=1}^{p} b_i(t)\phi_i(x_0). \tag{3.27}$$

Since POD modes are time independent, equation (3.27) is expressed in the form

$$y_{conc}(t) = \bar{C} b(t) \tag{3.28}$$

where $\bar{C}$ is $1 \times p$ measurement matrix and $b(t)$ is the p-dimensional vector of time varying coefficients of the Galerkin approximation of $c(t, x)$. Therefore, the complete

dynamical system of the concentration of the contaminants with available measurements is as follows:

$$\begin{aligned} \dot{b}_l(t) &= M_l b(t) + a(t)^T Q_l b(t),\ l = 1, \cdots, p, \\ y_{conc}(t) &= \bar{C} b(t). \end{aligned} \tag{3.29}$$

3.1.3 Energy Field Estimation

In this section, we extend the model reduction technique used in the treatment of the velocity and concentration fields to the development of a reduced order model of the energy balance. Assuming that there is no heat of mixing and the change in potential and kinetic energy between inlet and outlet streams is negligible. The temperature field dynamics are governed by the energy balance equation:[Bird et al., 1960]

$$\begin{aligned} div(v) &= 0 \\ \frac{\partial T}{\partial t} &= -(v \cdot \nabla)T + \varsigma \nabla^2 T + D_c \end{aligned} \tag{3.30}$$

where $T : \Omega \times \mathbb{R} \to \mathbb{R}$ indicates the fluid temperature field on spatial domain Ω, $D_c : \Omega \times \mathbb{R} \to \mathbb{R}$ describes "chemical source" of the temperature T. $\varsigma = k/\rho C_p$ with k, thermal conductivity coefficient, fluid density, ρ and heat capacity $\hat{C}_p$ at constant pressure per unit mass. As in the previous sections, the velocity and time are scaled by factors, V and V/L . We assume that the temperature and the source term are dimensionless.

POD based Model Reduction

The temperature field, $T(t,x)$ is expressed as an expansion form in the POD modes $\theta(x)$ defined on the spatial domain Ω as follows:

$$T(t,x) = \sum_{i=1}^{q} h_i(t)\theta_i(x) \tag{3.31}$$

The following standard inner product is considered:

$$< T_i, T_j >= \int_{\Omega} T_i(x) \cdot T_j(x) dV \tag{3.32}$$

where $T_i(x) \cdot T_j(x)$ is the standard dot product between two vectors $T_i(x)$ and $T_j(x)$ in Euclidean space, dV is a volume element.

Substituting both equations (3.2) and (3.31) into (3.30) yields:

$$\begin{aligned}\frac{\partial T}{\partial t} = \sum_{i=1}^{q} \dot{h}_i(t)\theta_i(x) = &- \left(\sum_{j=1}^{n} a_i(t)\alpha_i(x)\right) \cdot \nabla \left(\sum_{i=1}^{q} h_i(t)\theta_i(x)\right) \\ &+ \varsigma \nabla^2 T \left(\sum_{i=1}^{q} h_i(t)\theta_i(x)\right) + D_c. \end{aligned} \tag{3.33}$$

Projecting (3.33) onto POD modes $\theta_i(x)$ by applying inner product yields:

$$\begin{aligned}< \frac{\partial T}{\partial t}, \theta_m(x) > &=< \sum_{i=1}^{q} \dot{h}_i(t)\theta_i(x), \theta_m(x) > \\ &=< \sum_{i=1}^{n}\sum_{j=1}^{q} a_i(t)h_j(t)\nabla\theta_j(x) \cdot \alpha_i(x), \theta_m(x) > \\ &+ \varsigma < \nabla^2 \left(\sum_{i=1}^{q} h_i(t)\theta_i(x)\right), \theta_m(x) > + < D_c, \theta_m(x) > \end{aligned} \tag{3.34}$$

According to the linearity of the inner product and orthogonality of the POD modes, the reduced order dynamic equation is as follows:

$$\dot{h}_m(t) = -\sum_{i=1}^{n}\sum_{j=1}^{q} a_i(t)h_j(t) < \nabla\theta_j(x)\cdot\alpha_i(x), \theta_m(x) > + \varsigma\sum_{i=1}^{q} h_i(t) < \nabla^2\theta_i(x), \theta_m(x) > + < D_c, \theta_m(x) > \quad (3.35)$$

Equation (3.35) is the reduced order model for the approximation of temperature field using POD modes. Equation, (3.8), (3.22), and (3.35) are the basis for the design of an observer for three states of fluid flow, velocity filed $v(t,x)$, concentration field $c(t,x)$, and energy field $T(t,x)$.

Reduced order Energy Balance

The energy field system is simplified to a quadratic differential equation of the form:

$$\dot{h}_m(t) = L_m h(t) + a(t)^T E_m h(t) \quad m = 1,\ldots,q \quad (3.36)$$

where L_m is a row vector and E_m is an $q \times q$ matrix for $m = 1,\ldots,q$,

$$L_{m_i} = \varsigma < \nabla^2\theta_i(x), \theta_m(x) > \quad (3.37)$$

$$E_{m_{ij}} =< (\nabla\theta_j(x)\cdot)\alpha_i(x), \theta_m(x) > \quad (3.38)$$

In general, the source term is ignored since the POD modes are such that div(θ)=0 below,

$$< D_c, \theta_m(x) >= \int_\Omega D_c \cdot \theta_m(x) dV = \int_{\partial\Omega} D_c \theta_m(x) \cdot \mathbf{n}_\Omega dS \qquad m = 1, \ldots, q \tag{3.39}$$

where $\mathbf{n}_\Omega$ is the unit vector normal to the spatial domain Ω. The source term vanishes on the closed boundary which means that $\theta_m(x) = 0$ on the boundary of $\Omega, \partial\Omega$ as well.

Measurement

As for the measurements of the velocity and concentration derived in the previous sections, temperature field measurements, T_0, available at predefined locations, x_0, can be expressed in the expansion form:

$$h(t, x_0) = \sum_{i=1}^{q} h_i(t)\theta_i(x_0). \tag{3.40}$$

Since the POD modes are time independent, equation (3.40) is expressed in the form,

$$y_{temp}(t) = \tilde{C}h(t) \tag{3.41}$$

where $\tilde{C}$ is $1 \times u$ measurement matrix and $h(t)$ is the u-dimensional vector of time varying coefficients of the Galerkin approximation of T(t,x). The complete dynamical

system of the energy with available measurements is given by

$$\begin{aligned} \dot{h}_m(t) &= L_m h(t) + a(t)^T E_m h(t), m = 1, \cdots, q \\ y_{temp}(t) &= \tilde{C} h(t) \end{aligned} \tag{3.42}$$

The overall dynamical system, combining (3.16), (3.29) and (3.42), is summarized below:

$$\begin{aligned} \dot{a}_k(t) &= N_k a(t) + a(t)^T P_k a(t), & k &= 1, \ldots, n \\ \dot{b}_l(t) &= M_l b(t) + a(t)^T Q_l b(t), & l &= 1, \cdots, p \\ \dot{h}_m(t) &= L_m h(t) + a(t)^T E_m h(t), & m &= 1, \cdots, q \\ m(t) &= \begin{bmatrix} y_{velo}(t) \\ y_{conc}(t) \\ y_{temp}(t) \end{bmatrix} = \begin{bmatrix} C & 0 & 0 \\ 0 & \bar{C} & 0 \\ 0 & 0 & \tilde{C} \end{bmatrix} \begin{bmatrix} a(t) \\ b(t) \\ h(t) \end{bmatrix}. \end{aligned} \tag{3.43}$$

3.2 Observer Design

The purpose of this section is to design an observer for a nonlinear system of the general form (3.43). In [Guay and Hariharan, 2008], the application of an extended Kalman filter(EKF) was considered for linear approximation of velocity field estimation to reduce the complexity of the observer design. In order to improve the performance of the observer for the nonlinear system, we consider the higher order Kalman filter proposed in [Athans et al., 1968]. The Galerkin projection of three

states (3.43) is rewritten in the following form:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + \sum_{k=1}^{n+p+q} e_k x(t)^T F_k x(t) \\ m(t) &= Cx(t) + \varrho_1(t) \end{aligned} \tag{3.44}$$

where the vectors $e_k, (k = 1, \ldots, N)$, are the basis vectors on $\mathbb{R}^{n+p+q}$,

$$e_1 = \begin{bmatrix} 1 \\ 0 \\ \cdots \\ 0 \end{bmatrix}, e_2 = \begin{bmatrix} 0 \\ 1 \\ \cdots \\ 0 \end{bmatrix}, \ldots, e_{n+p+q} = \begin{bmatrix} 0 \\ 0 \\ \cdots \\ 1 \end{bmatrix}. \tag{3.45}$$

Considering $x(t) = [a(t), b(t), h(t)]^T \in \mathbb{R}^{n+p+q}$, the matrices A and $F_k (k = 1, \ldots, N)$ are of the following form

$$A = \begin{bmatrix} N_k & 0 & 0 \\ 0 & M_l & 0 \\ 0 & 0 & L_m \end{bmatrix} \tag{3.46}$$

and

$$F_k = \begin{cases} \begin{bmatrix} P_k & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} & k = 1, \ldots, n \\ \begin{bmatrix} 0 & \frac{1}{2}Q_{k-n} & 0 \\ \frac{1}{2}Q_{k-n} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} & k = n+1, \ldots, n+p \\ \begin{bmatrix} 0 & 0 & \frac{1}{2}E_{k-n-l} \\ 0 & 0 & 0 \\ \frac{1}{2}E_{k-n-l} & 0 & 0 \end{bmatrix} & k = n+p+1, \ldots, n+p+q \end{cases} \tag{3.47}$$

Let $N = n + p + q$, then the following assumptions are required for the application of the Kalman filter.

Assumption 3.2.1. *The initial condition $x(0)$ is a Gaussian random variable with mean $E\{x(0)\} = x_0$ and covariance $E\{(x(0) - x_0)(x(0) - x_0)^T\} = S_0$ where S_0 is a positive definite matrix.*

It is assumed that the measurements are available at discrete times, $t_1, \ldots, t_k, t_{k+1}, \ldots$, measurements. The state variables and measurement noise variables at time t_k are expressed by

$$m(t_k) = m_k,\ x(t_k) = x_k,\ \varrho(t_k) = \varrho_k$$

It is assumed that the measurements are subject to Gaussian noise, independent of x_0 with mean $E\{\varrho_k\} = 0$ and covariance $\text{Cov}\{\varrho_k\} = R_k$. At time t_k, the measurement sequence is

$$m_k = Cx_k + \varrho_k, \ (k = 1, 2, \ldots)$$

Let the state estimate be given by:

$$\omega(t_k) = \hat{x}_k.$$

In [Guay et al., 2010], the dynamical nonlinear filter is introduced using the following differential equations:

$$\dot{\omega}(t) = A\omega(t) + \sum_{i=1}^{N} e_i \omega(t)^T F_i \omega(t) + \sum_{j=1}^{N} e_j tr(F_j S(t)) \tag{3.48}$$

where tr is the trace of a square matrix. Define the $N \times N$ symmetric matrix $S(t)$ by

$$S(t) \triangleq E\{e(t)e^T(t)\}. \tag{3.49}$$

$S(t)$ is the positive definite covariance matrix of the state estimates given as the solution of matrix differential equation given by:

$$\dot{S}(t) = (A + 2\sum_{i=1}^{N} e_i \omega(t)^T F_i)S(t) + S(t)(A + 2\sum_{i=1}^{N} e_i \omega(t)^T F_i)^T \tag{3.50}$$

with initial condition $S(0) = S_0$. The covariance matrix at time t_k, $S(t_k) = \Sigma_k$. At the next time step t_{k+1}, let $y_{k+1} = Cx_{k+1}, \omega_{k+1} = \omega(t_{k+1})$ and $S_{k+1} = S(t_{k+1})$. The

estimate of the state is updated at t_{k+1} by the recursion:

$$\hat{x}_{k+1} = \omega_{k+1} + G_{k+1}\left[y_{k+1} - C\omega_{k+1}\right] \tag{3.51}$$

where the gain matrix G_{k+1} is given by,

$$G_{k+1} = S_{k+1}C^T\left[CS_{k+1}C^T + R_{k+1}\right]^{-1}. \tag{3.52}$$

The covariance matrix of the state estimates is updated as follow,

$$\Sigma_{k+1} = S_{k+1} = S_{k+1}C^T\left[CS_{k+1}C^T + R_{k+1}\right]^{-1}CS_{k+1}. \tag{3.53}$$

The second order filter (3.48) is used for the estimation of the nonlinear system (3.43) with discrete-time measurements. We require that the nonlinear system (3.43) is N-mode observable.

Definition 3.2.1. *[Guay et al., 2010] The dynamical system* (3.1)*,* (3.17) *and* (3.30) *with measurements y_{velo}, y_{conc} and y_{temp} is N-mode observable if the finite-dimensional Galerkin projection* (3.43) *is observable with output $y(t) = Cx(t)$*

Remark 3.2.2. *[Guay et al., 2010] N-mode observability is not guaranteed for any given combination of snapshots and measurements. The property must be checked in each case.*

The second order observer (3.48) estimates the states of the nonlinear system if the reduced order nonlinear system is stable and N-mode observable. A simulation example is presented in the next section.

3.3 Simulation Example

To illustrate the effectiveness of the proposed second order observer performance, we consider the following system:

$$\left.\begin{aligned} \dot{x}_1 &= -2x_1 - x_2x_1 \\ \dot{x}_2 &= -x_2 + x_1^2 \\ \dot{x}_3 &= -x_3 \\ y &= x_1 \end{aligned}\right\} \quad \text{Main System} \tag{3.54}$$

which is stable. It assumed the general form:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + \sum_{k=1}^{n+p+q} e_k x(t)^T F_k x(t) + \varrho_1(t) \\ m(t) &= Cx(t) + \varrho_2(t) \end{aligned}$$

where state $x(t)$ is a vector which has three elements (n=3), C is a measurement matrix, $[1 \quad 0 \quad 0]^T$, and the measurement $m(t)$ is observed at a discrete moment of time. $\varrho_1(t)$ and $\varrho_2(t)$ are discrete white Gaussian noise signals with zero mean $E\{\varrho(t)\} = 0$ and constant covariance $\text{Cov}\{\varrho(t)\} = R$. However, we only consider measurement noise. A is 3×3 constant matrix, F_1, F_2, and F_3 are also 3×3 matrices given by:

$$A = \begin{bmatrix} -2 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}, \quad F_1 = \begin{bmatrix} 0 & -\frac{1}{2} & 0 \\ -\frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad F_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad F_3 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

Let the continuous-time state estimate dynamics, $\omega(t) = \hat{x}(t)$. be described by the following system of ordinary differential equations:

$$\dot{\omega} = A\omega(t) + \sum_{k=1}^{n} e_k \omega(t)^T F_k \omega(t) + \sum_{k=1}^{n} e_k tr(F_k S(t)) \tag{3.55}$$

where the 3×3 covariance matrix $S(t)$ is the positive definite solution of the following matrix ordinary differential equation:

$$\dot{S}(t) = (A + 2\sum_{k=1}^{n} e_k \omega(t)^T F_k) S(t) + S(t)(A + 2\sum_{k=1}^{n} e_k \omega(t)^T F_k)^T \tag{3.56}$$

At time t_k, let $S(t_k) = S_k$, $R(t_k) = R_k$, $m(t_k) = m_k$, and $\omega(t_k) = \omega_k$. The state estimate at time t_k is given by:

$$\hat{x}_k = \omega_k + G_k[m_k - C\omega_k] \tag{3.57}$$

where the gain matrix G_k is given by

$$G_k = S_k C^T [C S_k C^T + R_k]^{-1} \tag{3.58}$$

The covariance matrix of the state variables is given by,

$$\Sigma_k = S_k - S_k C^T [C S_k C^T + R_k]^{-1} C S_k \tag{3.59}$$

3.3.1 Simulation Results

The dynamical equations of the plant (3.54) and the filter (3.55) were numerically integrated using the ODE15s Matlab function. The system states $x(t)$, filter variable

$\omega(t)$ and covariance $S(t)$ are solved in continuous-time, and the estimates, $\hat{x}_k$, are generated in discrete-time. The observations were taken every second, and the running time used was from $t = 0$ to $t = 10000$. The Gaussian measurement noise is generated using a normal random number generator from a normal distribution with mean parameter, 0 and standard deviation parameter 0.1. At time $t = 0$, the initial states were set to be Gaussian random variables and independent. The measurement matrix C, initial state variables and covariance matrix are taken as:

$$C = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad x(0) = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}, \quad S(0) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \omega(0) = \begin{bmatrix} 2 \\ 1 \\ 4 \end{bmatrix},$$

Figure 3.1 shows the nominal trajectory of the three states (solid line) and the estimation of states (dotted line). The results show that the estimation of states x_1, x_2 and x_3 follow the unknown state variable as required. In Figure 3.2, the errors are shown to vanish, although x_2 and x_3 have relatively large gaps initially. This simulation demonstrates the performance of the nonlinear observer for estimating the three states. This observer will be applied in the design of a distributed system in the next chapter.

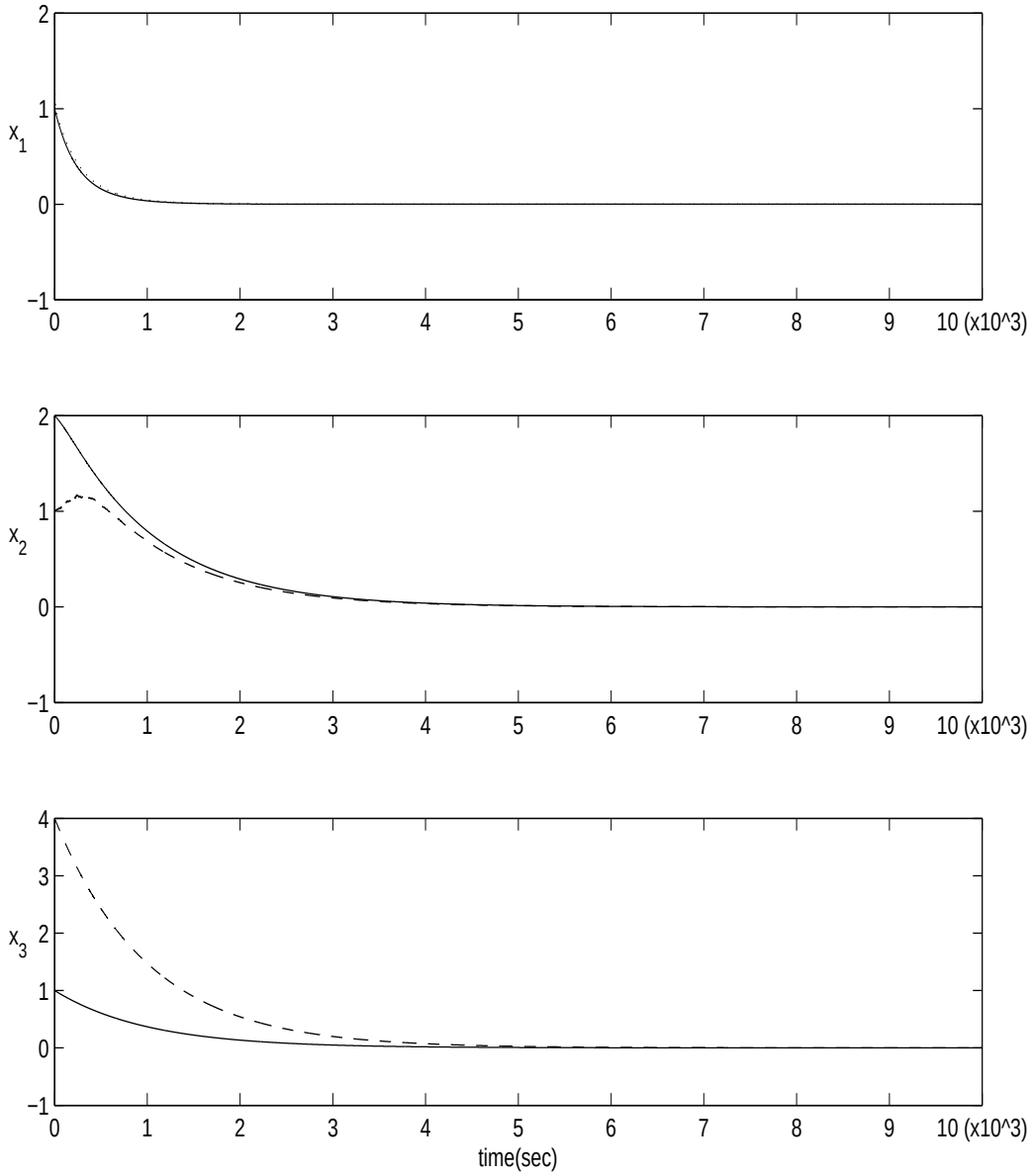

Figure 3.1: The performance of second order filter of nonlinear system. (states "—", estimation of states "- - -")

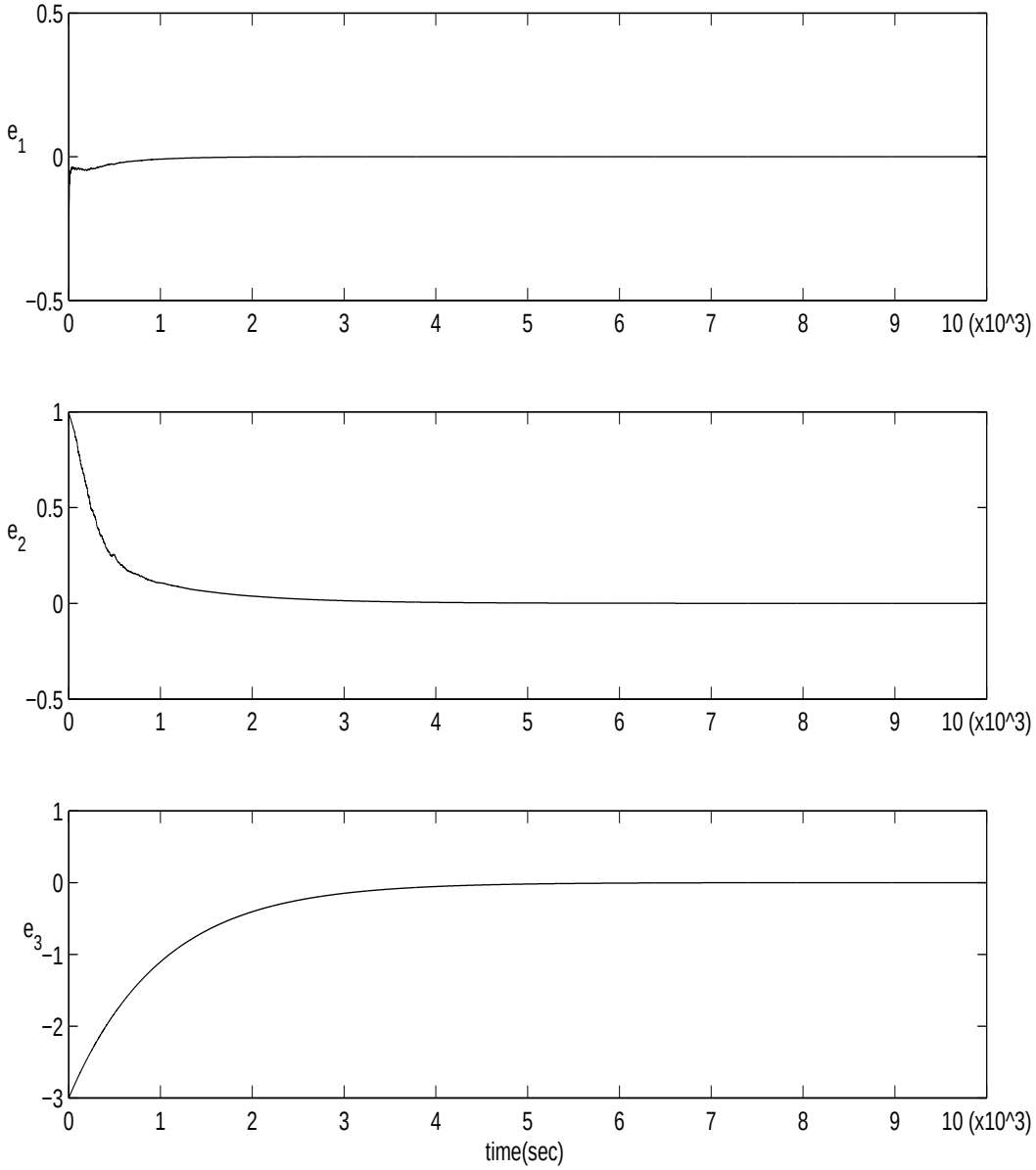

Figure 3.2: Time course plot of the state estimation error $e = x - \hat{x}$

Chapter 4

Consensus

One of the major research problems associated with the study of sensor networks is to develop scalable detection and estimation algorithms. In order to address this complex problem, one should develop distributed algorithms that yield computable real-time algorithms. Distributed systems are usually represented by a collection of agents. Each agent attempts to accomplish set number of tasks that can require agreement on some commonly known variables that depend on the state of all agents. These variables are called consensus states. They must be communicated between agents to ensure that global tasks are achieved. Consensus algorithms are interaction methods that are based on information exchange between a sensor and all its neighbouring sensors.

Decentralized Kalman filtering proposed in [Rao et al., 1993], uses a set of local Kalman filters that communicate with all other sensors. Since this decentralized filtering requires all-to-all connected topology, it is not scalable. Hence, we focus on scalable or distributed Kalman filtering algorithms in which each sensor only communicates with neighbouring sensors within a certain radius. [Olfati-Saber, 2007]

This chapter will provide information regarding consensus algorithms, as well as two methods of developing a distributed estimation algorithm system. The first method is a data fusion based on a local higher order Kalman filter coupled to consensus filters. The second method provides an alternative way to develop the distributed algorithm based on local state estimates.

4.1 Introduction

Sensor networks are used in a number of different fields because they are an effective means of data fusion and information collection. According to [Arampatzis et al., 2005], a sensor network is a type of computer network with numerous sensor nodes. These nodes are utilized in a number of industries such as meteorology, forestry, and agriculture. This technology can also be adapted to military uses. Regardless of their utility in various fields, all sensor networks must perform their tasks in a efficient manner and the development of distributed algorithm will help to address these efficiencies.

A decentralized algorithm was developed by [Speyer, 1978] where an all-to-all sensor network architecture is required. Since the information flow all-to-all link has n^2 communication complexity, where n is the number of sensors, it is not scalable for sensor networks. A distributed Kalman filtering algorithm is required to limit communication information such that only from nearest neighbours is needed. If the Graph G associated with the sensor is connected, then one can develop such scalable algorithms by exploiting the graph properties. The distributed filter is more flexible and provides accurate information to develop average consensus states at each node,

as proposed in [Olfati-Saber, 2005b].

According to [N.A.Lynch, 1997; Anderson and J.B.Moore, 1979], the development of a distributed algorithm primarily depends on the properties of the observer considered in the design. In [Olfati-Saber and R.M.Murray, 2004, 2003; Olfati-Saber, 2005a], a Distributed Kalman Filter(DKF) is proposed that generates average-consensus states in linear systems. This chapter develops distributed algorithms using two version of the consensus filter of [Olfati-Saber and R.M.Murray, 2004, 2003; Olfati-Saber, 2005a]. The first version involves data collection of measurement and covariance. The second version involves finding appropriate consensus filters for each of them. Once these two problems are addressed, we apply a higher order filter in the consensus filter to develop distributed algorithms for nonlinear systems.

In order to address these problems, we will firstly talk about the concept of the consensus filter in networks in Section 4.2. Section 4.3 discusses the distributed Kalman filter design problem. Two dynamical consensus problems are considered to provide an effective fusion of the measurement, and covariance information in the distributed Kalman filter. Appropriate filters are proposed to achieve consensus on the measurements and the covariance data. A direct consensus filter is also considered to provide consensus on state estimates. The resulting methods are applied to a nonlinear system in Section 4.4 and a simulation example is given in Section 4.5. [Olfati-Saber, 2005b]

4.2 Consensus in Networks

In section 2.1.5, we defined a graph $G_n(k) \triangleq (V_n, E_n(k))$, the corresponding adjacency matrix $\mathbf{A}_n(k)$ and Laplacian matrix $\mathbf{L}_n(k)$ as models of the relation between sensors in

a network of dynamical agent systems. Let us first describe a linear consensus protocol in networks using basic background from graph theory. Let $N_i = \{i \in V_n : a_{ij} \neq 0\}$ be the set of neighbours of sensor i and $J_i = N_i \cup \{i\}$ the set of inclusive neighbours of sensor i.

Theorem 4.2.1. *Each sensor of a connected graph G_n has the following consensus linear protocol. [Olfati-Saber and R.M.Murray, 2003]*

$$\dot{x}_i(t) = \sum_{j \in N_i} (x_j(t) - x_i(t)), \quad x(0) \in \mathbb{R}^n \tag{4.1}$$

Let $z_1, \ldots, z_n \in \mathbb{R}$ be n constants that define the initial condition of each agent, $x_i(0) = z_i$. When graph G_n is connected, all sensors globally come to an average value asymptotically $\bar{z} = \frac{1}{n}\sum_i z_i$ and identical elements yield an average consensus,z. For example, let $x^* = \lim_{t \to +\infty} x(t)$ then, $x_i^* = x_j^* = avg\ (x(0)) = z$. The vector of the value of the sensors can be expressed as

$$x^* = (z, \ldots, z)^T, \qquad z \in \mathbb{R} \tag{4.2}$$

The linear consensus algorithm is associated with the Laplacian matrix. The system can be expressed in a compact form as

$$\dot{x} = -\mathbf{L}x$$

where $L = L(G_n)$ is the Laplacian matrix of graph G_n. It is defined as $L = D - A$ where D is diagonal degree matrix of G with diagonal elements $d_i = |N_i| = \sum_j a_{ij}$ of

sensor i. Let λ is eigenvalue of Laplacian $\mathbf{L}$, according to the definition of Laplacian, all row-sums of $\mathbf{L}$ are zero since $\sum_j L_{ij} = 0$. Hence, the Laplacian matrix always has a zero eigenvalue $\lambda_1 = 0$, and the remaining eigenvalues are such that

$$\lambda_1 \leq \lambda_2 \leq \cdots \lambda_n$$

The zero eigenvalue corresponds to the aligned state of the graph $G_n, \mathbf{1} = (1, 1, \ldots, 1)^T$ since $\mathbf{1}$ belongs to the null-space of L ($i.e. L\mathbf{1}$=0). If the graph G_n is connected, then eigenvalue λ_2 is greater than zero, $\lambda_2 > 0$. The eigenvalue λ_2 is called the algebraic connectivity of the graph [Godsil and Royle, 2001] and it is a measure of the speed of convergence (or performance) of the consensus algorithm. Algebraic connectivity fulfils the following inequality:

$$\lambda_2(G_n) \leq \nu(G_n) \leq \eta(G_n)$$

where $\nu(G_n)$ is the node connectivity and $\eta(G_n)$ is the edge connectivity of G_n. Using this inequality, it can be shown that the system is robust in terms of both node-failures and edge-failures when the network has relatively high algebraic connectivity [Olfati-Saber, 2005a]

4.3 Distributed Kalman Filter

4.3.1 Kalman Filter

A Kalman filter is an optimal estimator that minimizes the mean square error of the state esimates for a class of linear dynamical systems subject to Gaussian noise signals.

Using a Kalman filter, one can infer the estimate of states from the measurement data using a recursive process. Assuming there are n sensors in an interconnected network G_n, we can consider the general problem of trying to estimate the state $x \in \mathbb{R}^n$ in a discrete-time model of form.

$$x_k = A_{k-1}x_{k-1} + f_{k-1} \tag{4.3}$$

with measurement $y \in \mathbb{R}^m$ obtained from m sensors at time t_k,

$$y_k = B_k x_k + v_k \tag{4.4}$$

Assumption 4.3.1. *The process noise covariance Q and measurement noise covariance R are constant.*

Assumption 4.3.2. *The $n \times n$ matrix A in the system* (4.3) *and the $m \times n$ matrix B in the measurement* (4.4) *are constant.*

The random variables $\{f_k\}, \{v_k\}$ are white Gaussian noise(WGN) for the process and measurement respectively. They are independent, zero-mean and each noise covariance matrix equals to Q_k and R_k respectively in normal probability distributions. It is assumed that:

$$
\begin{aligned}
f_k &\sim (0, Q_k) \\
v_k &\sim (0, R_k) \\
E[f_k f_j^T] &= Q_k \xi_{k-j} \\
E[v_k v_j^T] &= R_k \xi_{k-j} \\
E[v_k f_j^T] &= 0 \\
x_0 &= \mathscr{N}(\bar{x}_0, P_0)
\end{aligned} \tag{4.5}
$$

where ξ_{k-j} is the Kronecker delta function; when $k = j$, $\xi_{k-j} = 1$, and when $k \neq j$, $\xi_{k-j} = 0$. The estimation of x_k is based on the information of the dynamical system (4.3) and noisy measurement (4.4). If we have all the information of the measurement up to step t_k is available, we can then form an *a posteriori* estimate, denoted as $\hat{x}_k^+$. The"+" indicates that the estimate is *a posteriori* and the *a posteriori* estimate can be expressed as the expected value of x_k with all measurements including step t_k:

$$\hat{x}_k^+ = E[x_k \mid m_1, m_2, \cdots, m_k]. \tag{4.6}$$

If we consider all the information of the measurement before (not including) step t_k available to estimate x_k, then we can obtain an *a priori* estimate, denoted by $\hat{x}_k^-$. The superscript "$^-$" indicates that the estimate is *a priori* and the *a priori* estimate can be expressed as the expected value of x_k with all of measurements up to (not

including) step t_k:

$$\hat{x}_k^- = E[x_k \mid m_1, m_2, \cdots, m_{k-1}]. \tag{4.7}$$

Both $\hat{x}_k^-$ and $\hat{x}_k^+$ are estimates of x_k, depending on before we process the measurement at step t_k, $(\hat{x}_k^-)$. Once the measurement at step t_k is processed, one can compute the estimate $\hat{x}_k^+$. It is assumed (without loss of generality) that $\hat{x}_0^+$ provides an initial value of the estimate of x_0 despite the fact that no information is available until time t_1. Therefore, we can indicate the initial estimate x_0 as the expected value of the initial state x_0 :

$$\hat{x}_0^+ = E(x_0) \tag{4.8}$$

Let P_k be the covariance of the estimation error. P_k^- represents the covariance of the estimation error of $\hat{x}_k^-$, and P_k^+ indicates the covariance of the estimation error of $\hat{x}_k^+$. They are defined as follows:

$$\begin{aligned} P_k^- &= E[(x_k - \hat{x}_k^-)(x_k - \hat{x}_k^-)^T] \\ P_k^+ &= E[(x_k - \hat{x}_k^+)(x_k - \hat{x}_k^+)^T] \end{aligned} \tag{4.9}$$

Figure 4.1 shows the relationship between the estimates of the state and the covariance between before and after measurement at time t_{k-1} and t_k. After we process the measurement and covariance at time t_{k-1}, we have estimates of the state $x_{k-1}$and covariance P_{k-1} represented by $\hat{x}_{k-1}^+$ and P_{k-1}^+ respectively. When the system moves to the next step t_k then we compute an estimate of the state $(\hat{x}_k^-)$ and the covariance

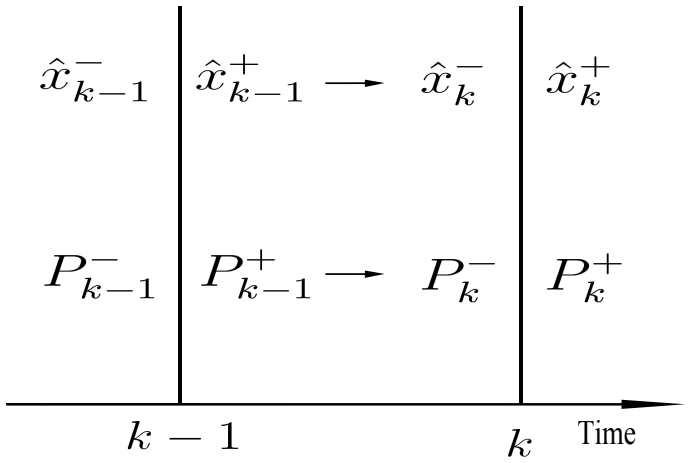

Figure 4.1: Timeline showing *a posteriori* and *a priori* state estimates and estimation error covariances [Simon, 2006]

(P_k^-). Then at step t_k, using the measurement (m_k), we compute a new estimate of the state ($\hat{x}_k^+$) and its covariance (P_k^+). The discrete-time Kalman filter iterations are summarized as follows:

$$P_k^- = A_{k-1}P_{k-1}^+A_{k-1}^T + Q_{k-1} \tag{4.10}$$

$$\begin{aligned} K_k &= P_k^- B_k^T (B_k P_k^- B_k^T + R_k)^{-1} \\ &= P_k^+ B_k^T R_k^{-1} \end{aligned} \tag{4.11}$$

$$\hat{x}_k^- = A_{k-1}\hat{x}_{k-1}^+ \tag{4.12}$$

$$\hat{x}_k^+ = \hat{x}_k^- + K_k(m_k - B_k\hat{x}_k^-) \tag{4.13}$$

$$\begin{aligned} P_k^+ &= (I - K_k B_k)P_k^- \\ &= (I - K_k B_k)P_k^-(I - K_k B_k)^T + K_k R_k K_k^T \\ &= [(P_k^-)^{-1} + B_k^T R_k^{-1} B_k]^{-1} \end{aligned} \tag{4.14}$$

The first form of P_k^+ (4.14) ensures that P_k^+ will always be symmetric positive definite,

as long as P_k^- is symmetric positive definite. The first one is more stable and robust than the third form. However, the third form of P_k^+ (4.14) does not guarantee that P_k^+ will always be a symmetric positive definite matrix. The second form is used in the information filter below. Also, we can see that (4.10),(4.11),(4.14) are not related to the measurement m_k, and only depend on the system parameters A_k, B_k, Q_k, and R_k. As a result, one can calculate the Kalman filter gain K_k *a priori*. The estimate of system $\hat{x}_k$ can be implemented during real-time operation. This is advantageous as we can evaluate the performance of the filter prior to its implementation. In contrast, the filter gain and covariance for a nonlinear system depend on the measurement and cannot be computed *a priori*.

Kalman Filter : Information Form

An information filter form of the Kalman filter is presented in [Simon, 2006]. It is an implementation of the Kalman filter that propagates the inverse of S, S^-, instead of S. The information matrix of the system is defined as:

$$P = E[(x - \hat{x})(x - \hat{x})^T] \tag{4.15}$$

$$S = P^{-1} \tag{4.16}$$

The covariance, P is a measure of the uncertainty in the estimation of the state. A large value of P indicates significant uncertainty in the state estimates. When $P \to 0$, there is sufficient information to estimate x accurately. Conversely, the information matrix S indicates the certainty in the estimation of the state. The second form of

the equation (4.14) can be expressed as:

$$(P_k^+)^{-1} = (P_k^-)^{-1} + B_k^T R_k^{-1} B_k \tag{4.17}$$

and in terms of S, as follows:

$$S_k^+ = S_k^- + B_k^T R_k^{-1} B_k \tag{4.18}$$

One can also write the measurement-update equation (4.10) for the information matrix as

$$S_k^- = [A_{k-1}(S_{k-1}^+)^{-1} A_{k-1}^T + Q_{k-1}]^{-1} \tag{4.19}$$

Using the matrix inversion lemma from [Simon, 2006], it follows the

$$(A + BD^{-1}C)^{-1} = A^{-1} - A^{-1}B(D + CA^{-1}B)^{-1}CA^{-1} \tag{4.20}$$

As a result, one can restate (4.19) as:

$$S_k^- = Q_{k-1}^{-1} - Q_{k-1}^{-1} A_{k-1}(S_{k-1}^+ + A_{k-1}^T Q_{k-1}^{-1} A_{k-1})^{-1} A_{k-1}^T Q_{k-1}^{-1} \tag{4.21}$$

which becomes a time-update equation for information matrix, S. Now, let $P_k^+ = M_k$, $P_k^- = N_k$, $\hat{x}^+ = \hat{x}$ and $\hat{x}^- = \omega$ simply.

When the dynamical systems and the measurements in each agent are the same, the

information filter is given as:[Olfati-Saber, 2005b]

$$M_k^{-1} = N_k^{-1} + B_k^T R_k^{-1} B_k \tag{4.22}$$

$$K_k = M_k B_k^T R_k^{-1} \tag{4.23}$$

$$\hat{x}_k = \omega_k + K_k(m_k - B_k\omega_k) \tag{4.24}$$

$$\begin{aligned} N_k &= Q_{k-1}^{-1} - Q_{k-1}^{-1}A_{k-1}(M_{k-1}^{-1} + A_{k-1}^T Q_{k-1}^{-1} A_{k-1})^{-1} A_{k-1}^T Q_{k-1}^{-1} \\ &= A_{k-1} M_{k-1} A_{k-1}^T + Q_{k-1} \end{aligned} \tag{4.25}$$

$$\omega_k = A_{k-1}\hat{x}_{k-1} \tag{4.26}$$

One of the main advantages of the information filter is that when r is the number of measurements, r measurements can be filtered by summing their information matrices at each step. While the regular Kalman filter equations require the inversion of an $r \times r$ matrix, the information filter requires $n \times n$ matrix when n is the number of the states. As a result, the information filter is computationally simpler.

Micro Kalman Filters

[Olfati-Saber, 2005b] shows how the information of a central Kalman filter in a sensor network can be expressed in consensus form using *micro*-Kalman filters(μKF) with measurement vectors embedded in each sensors. The *micro*-Kalman filter at each sensor collects the information on the measurements and the covariance and calculates the state estimate $\hat{x}$ in a distributed way.

Assuming that there is a sensing model at sensor i,

$$m_i(k) = B_i(k)x(k) + v_i(k) \tag{4.27}$$

where n sensors with $p \times m$ measurement matrices B_i. Let the central measurement definition with observation noise and matrix be denoted by:

$$m_c = [m_1; m_2; \cdots ; m_n], \tag{4.28}$$

$$v_c = [v_1; v_2; \cdots ; v_n], \tag{4.29}$$

$$B_c = [B_1; B_2; \cdots ; B_n], \tag{4.30}$$

where the subscript "c" means "central" and B_c is a column block matrix. Using this notation, one can write the network measurements with B_i along its diagonal as

$$m_c(k) = B_c(k)x(k) + v_c(k) \tag{4.31}$$

The information form of the Kalman filter is then used as a *micro*-Kalman filter (4.22) (4.23) (4.24) form for each sensor. It can be expressed as:

$$M = (N^{-1} + B_c^T R_c^{-1} B_c)^{-1} \tag{4.32}$$

$$K_c = M B_c^T R_c^{-1} \tag{4.33}$$

$$\begin{aligned} \hat{x} &= \omega + K_c(m_c - B_c\omega) \\ &= \omega + M(B_c^T R_c^{-1} m_c - B_c^T R_c^{-1} B_c \omega). \end{aligned} \tag{4.34}$$

Define the $m \times m$ average inverse-covariance matrix and the m-vector of average

measurements as:

$$S(k) = \frac{1}{n}\ B_c^T(k)R_c(k)^{-1}B_c(k) = \frac{1}{n}\ \sum_{i=1}^{1}\ B_i^T(k)R_i(k)^{-1}B_i(k) \tag{4.35}$$

$$y_i(k) = B_i^T(k)R_i(k)^{-1}m_i(k), \quad y = \frac{1}{n}\sum_{i=1}^{n} y_i, \tag{4.36}$$

respectively.

We can get the state propagation equation (4.34) as:

$$\hat{x}(k) = \omega(k) + M_\mu(k)(y(k) - S(k)\omega(k)) \tag{4.37}$$

where $M_\mu(k)$ is the *micro*-Kalman gain, given by

$$\begin{aligned} M_\mu(k) &= nM(k) \\ &= ((nN(k))^{-1} + S(k))^{-1}. \end{aligned} \tag{4.38}$$

We can express the *micro*-Kalman filter at each node as $N\mu(k) = nN(k)$ and $Q_\mu(k) = nQ(k)$. The update equation (4.25) for a μKF are given by:

$$N_\mu^+(k+1) = AM_\mu(k)A^T + Q_\mu(k) \tag{4.39}$$

The average inverse covariance matrix S and the average measurement y are calculated at each step. They are used to estimate the state $\hat{x}$ using the local *micro* Kalman filter (4.37) $\sim$ (4.39) at each sensor. The estimate of the states by the *micro* Kalman filter approximate the state estimates from a centralized Kalman filter when there exists n sensors with connected graph, topology G_n with dimension m sensor

measurement. Having reduced the Distributed Kalman filter problem to the solution of two dynamic consensus problems by the fusion of data for the measurement and covariance, we apply appropriate consensus filters for each of them. A distributed low-pass consensus filter [Olfati-Saber and Shamma, 2005] is a useful tool for sensor fusion of noisy measurement y_i. A band-pass consensus filter is recommended for the calculation for the inverse-covariance matrices, S. The result of the implementation of the DKF is that it yields a sensor network where the local information in each agent enters certain neighbourhoods of the consensus value called a ε-consensus. That is, it can be shown that all sensors reach the small neighbourhoods of radius, $\varepsilon \ll 1$ of the consensus values $\hat{S}_i$ and $\hat{y}_i$ which are column-wise for node i [Olfati-Saber and Shamma, 2005]. The corresponding state and covariance update equations for sensor ith μKF are given by:

$$M_i(k) = (N_i(k)^{-1} + \hat{S}_i(k))^{-1}, \tag{4.40}$$

$$\hat{x}_i(k) = \omega_i(k) + M_i(k)(\hat{y}_i - \hat{S}_i\omega_i(k)), \tag{4.41}$$

$$N_i(k+1)^+ = A_k M_i(k) A_k^T + Q_\mu(k), \tag{4.42}$$

$$\omega_i(k+1) = A_k \hat{x}_i(k) \tag{4.43}$$

4.3.2 Distributed Kalman Filter Type [I] : Consensus-Based Fusion of Sensory Data

In [Olfati-Saber, 2005b], distributed algorithms using Kalman filtering in sensor networks are proposed, in which each sensor is quarantined to reach consensus states for measurement, m and covariance, S. This algorithm associates a *micro*-Kalman filter (μ KF) with each embedded low-pass and band-pass consensus filter. Two dynamic

consensus problems are solved by appropriate consensus filters where the filters work by fusing data at each sensor. There are three kinds of consensus filters: a low-pass filter, a band-pass filter, and a high-pass filter applied to each filter depending on the bandwidth of the system dynamics . In Olfati-Saber [2007], high-pass filters are generally recognized to have some critical weakness and are not recommended. Low-pass filters are recommended for the fusion of measurement data and a band-pass for the fusion of covariance data.

- **Low-Pass Consensus Filter**

Consider that c_i is the state of consensus filter and that u_j is the input of sensor j with m-dimension respectively. The dynamic consensus algorithm is given by:

$$\dot{c}_i = \beta \sum_{j \in N_i} (c_j - c_i) + \beta \sum_{j \in N_i \cup \{i\}} (u_j - c_i) \tag{4.44}$$

It can be expressed in compact form as:

$$\dot{c} = -\hat{L}c - \hat{L}u + (I_n + \hat{A})(u - c) \tag{4.45}$$

where c is the fusion of sensor data, denoting $c = [c_1, \ldots, c_n]$ and $\hat{A} = A \otimes I_m$ and $\hat{L} = L \otimes I_m$. Since $\hat{y}_i$ is calculated using this filter (4.44) through fusion of measurements, we consider $B_i^T R_i^{-1} m_i$ as the input of sensor i. The gain $\beta > 0$ is relatively large for random ad hoc topologies. [Olfati-Saber, 2007]

- **Band-Pass Consensus Filter**

Consider that $(v_i, b_i) \in \mathbb{R}^{2m}$ is the state, U_i is the input, and b_i is the output. The dynamic consensus algorithm is given by:

$$\dot{v}_i = -\hat{L}v_i - \hat{L}U_i, \tag{4.46}$$

$$g_i = v_i + U_i, \tag{4.47}$$

$$\dot{b}_i = \alpha \sum_{j \in N_i} (b_j - b_i) + \alpha \sum_{j \in N_i \cup \{i\}} (g_j - b_i) \tag{4.48}$$

Since $\hat{S}_i$ is calculated by $[(4.46) \sim (4.48)]$, we consider $B_i^T R_i^{-1} B_i$ as the input of sensor i in vectorized form. The gain $\alpha > 0$ is required to be large enough for the topologies for communication directly. In [Olfati-Saber, 2007], the following Algorithm A describes the DKF with consensus filtering. It indicates that sensor i sends the information such as states of measurement and covariance, c_i and b_i respectively, and receives input from the neighbouring consensus filters u_i and U_i. Note that Algorithm A is based on Assumption 4.3.3

Assumption 4.3.3. *[Olfati-Saber, 2007] Distributed Kalman Filter (DKF) is only valid for sensors with identical sensing models. It is applicable only to homogeneous multi-sensor fusion.*

$$x(k+1) = A_k x(k) + f_i(k)$$

$$m_i(k) = B_i(k) x_(k) + v_i(k)$$

Here, x_k denotes the state of a dynamic process, $B_i = B, \forall i$, we get

$$m_i(k) = B(k)x(k) + v_i(k)$$

[**Algorithm A**] Distributed Kalman Filtering Algorithms with consensus filtering of the measurement and covariance information. [Olfati-Saber, 2007]

1 : Initial State : $c_i = 0$, $b_i = 0$, $N_i = nN_0$, $\omega_i = x(0)$

2 : Produces new data in neighbourhoods, sensors i and j at time t_k.

3 : Invite measurement data from neighbour sensors and update it:

$$\begin{aligned} u_j &= B_j^T R_j^{-1} m_j, \quad \forall j \in N_i \cup \{i\} \\ c_i &\leftarrow c_i + \epsilon \sum_{j \in N_i} [(c_j - c_i) + (u_j - c_i)] \\ \hat{y}_i &= c_i + u_i \end{aligned}$$

4 : Invite new covariance data from neighbour sensors and update it:

$$\begin{aligned} U_j &= B_j^T R_j^{-1} B_j, \quad \forall j \in N_i \cup \{i\} \\ \dot{v}_i &= -\hat{L} v_i - \hat{L} U_i, \\ g_i &= v_i + U_i \\ b_i &\leftarrow b_i + \epsilon \sum_{j \in N_i \cup \{i\}} [(b_j - b_i) + (g_j - b_i)] \\ \hat{S}_i &= b_i + U_i \end{aligned}$$

5 : Estimate the state by μKF:

$$\begin{aligned} M_i(k) &= (N_i(k)^{-1} + S_i(k))^{-1}, \\ \hat{x}_i(k) &= \omega_i + M_i(k)(y_i(k) - S_i(k)\omega_i), \end{aligned}$$

6 : Update the state for time t_{k+1}

$$\begin{aligned} N_i(k+1) &\leftarrow A_k M_i(k) A_k^T + Q(k), \\ \omega_i(k+1) &\leftarrow A_k \hat{x}_i(k) \end{aligned}$$

7 : (End) Move to step 2 for time t_{k+1}

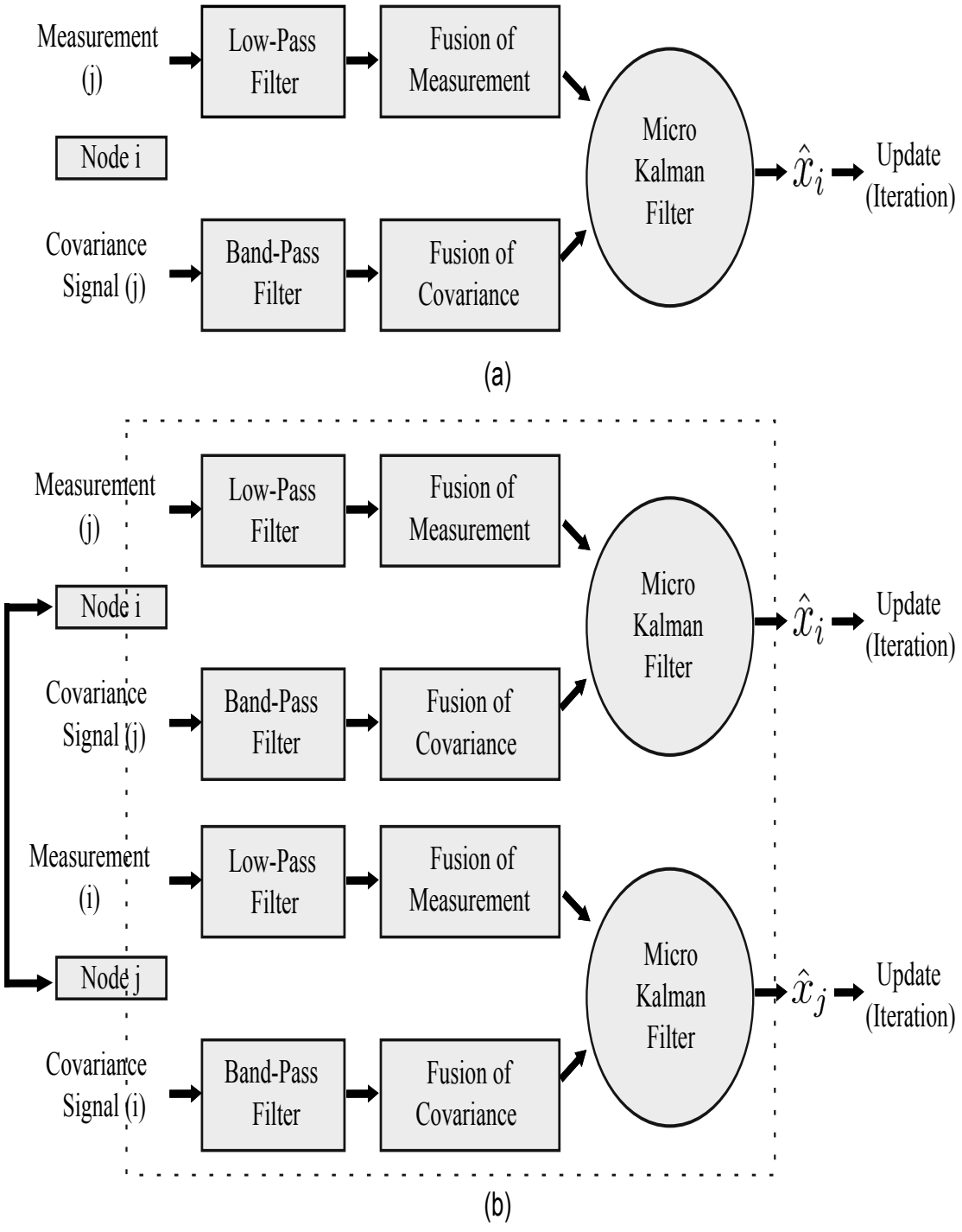

Figure 4.2: The architecture for distributed Kalman filtering:(a)description of consensus filters and μKF at node i and (b) communication way between different nodes of consensus filters in neighbourhoods. [Olfati-Saber, 2005b]

Fig. 4.2 (a) describes how two dynamic consensus problems are solved using different consensus filter and μKF at sensor i. Two types of information, measurement and covariance data, from sensor j are transferred into sensor i and are used to collect information. Once this has occurred, the local *micro*-Kalman filter then calculates the state estimate. Fig. 4.2 (b) shows how measurements and covariance data exchange at two different sensors i and j. Each sensor invites two types of data from other sensors to exchange information. Then, each local *micro*-Kalman filter is calculated. As a result, each sensor gets the same state estimate, $\hat{x}$, which is the centralized estimate when consensus is reached.

4.3.3 Distributed Kalman Filter Type [II] : Consensus on Estimation

Algorithm A has a key weakness related to Assumption 4.3.3. Since each sensor is subject to its own white Gaussian noise, it cannot yield agreement of measurement between neighbour sensors. The limitations of Algorithm A motivates the development of alternative distributed Kalman filters Algorithm B and Algorithm C. The new algorithms are based on the following assumption.

Assumption 4.3.4. *[Olfati-Saber, 2007] Distributed Kalman Filter (DKF) type [II] is valid for sensors with different sensing models. It is applicable to heterogeneous multi-sensor fusion.*

$$x(k+1) = A_k x(k) + f_i(k)$$

$$m_i(k) = B_i(k) x_(k) + v_i(k)$$

The measurement matrix B_i's are different across the whole network and the state of a dynamic process $x(k)$ is collectively observable, $B_i \neq B, \forall i$

A different approach is considered here, where the distributed Kalman filtering is based on the exchange of state estimates between neighbouring sensors rather than on the measurement and covariance. Olfati-Saber [2007] named this second class of the DKF algorithms a type II algorithm. Each sensor calculates states and estimates by local Kalman filter, and then exchange the information between neighboring sensors. It updates the state of the local Kalman filter before moving to the next step. The new algorithm is shown below. Recall that local Kalman filter iterations are based on (4.40) $\sim$ (4.43)) where sensor i locally compute $y_i(k)$ and $S_i(k)$.

[**Algorithm B**] Distributed Kalman Filtering Algorithms with consensus based on state estimate. [Olfati-Saber, 2007]

1 : Initial State : $N_i = N_0$, $\chi_i = x(0)$

2 : Produces new data in neighbourhoods, sensors i and j at time t_k.

3 : Combine measurement data and covariance matrices locally from neighbour sensors:

$$u_j = B_j^T R_j^{-1} m_j, \quad \forall j \in N_i \cup \{i\}$$
$$U_j = B_j^T R_j^{-1} B_j, \quad \forall j \in N_i \cup \{i\}$$

4 : Estimate the state by local Kalman filter:

$$M_i(k) = (N_i(k)^{-1} + S_i(k))^{-1},$$
$$\psi_i(k) = \chi_i(k) + M_i(k)(y_i(k) - S_i(k)\chi_i(k))$$

5 : Estimate the state by consensus filter.

$$\hat{x}_i(k) = \psi_i(k) + \epsilon \sum_{j \in N_i} (\psi_j(k) - \psi_i(k))$$

6 : Update the state of the local Kalman filter for time t_{k+1}:

$$N_i(k+1) \leftarrow A_k M_i(k) A_k^T + Q(k),$$
$$\chi_i(k+1) \leftarrow A_k \hat{x}_i(k)$$

7 : (End) Move to step 2 for time t_{k+1}

Assuming that ψ_i is the intermediate state estimate, each sensor exchanges the information of input, u_i and U_i, and intermediate state estimate ψ_i to its neighbour. As a result, Algorithm B attempts to compute values of the state estimates that are close between neighbour sensors by decreasing the differences caused by measurement noise. It implements a consensus step right after the estimation step of two local Kalman filter. One can combine the two different steps such as consensus step after the local Kalman filtering step into a single one. This new "Consensus on estimates" is a more rigorous approach. Additionally, Algorithm C tends to be faster than Algorithm B because the information from local Kalman filter does not have to be

communicated to the next step to establish consensus states. It is implemented simultaneously in the distributed observer.

[**Algorithm C**] Iterative Kalman - Consensus filter : Distributed Kalman filtering Algorithm with an observer. [Olfati-Saber, 2007]

---

1 : Initial State : $N_i = N_0$, $\omega_i = x(0)$

2 : Produces new data in neighbourhoods, sensors i and j at time t_k.

3 : Combine measurement data and covariance matrices locally from neighbour sensors:

$$u_j = B_j^T R_j^{-1} m_j, \quad \forall j \in N_i \cup \{i\}$$
$$U_j = B_j^T R_j^{-1} B_j, \quad \forall j \in N_i \cup \{i\}$$

4 : Estimate the state by computing Kalman-Consensus filter:

$$M_i(k) = (N_i(k)^{-1} + S_i(k))^{-1},$$
$$\hat{x}_i(k) = \omega_i + M_i(k)(y_i(k) - S_i(k)\omega_i) + \epsilon M_i \sum_{j \in N_i} (\omega_j(k) - \omega_i(k)),$$

5 : Update the state of the Kalman-Consensus filter for time t_{k+1}:

$$N_i(k+1) \leftarrow A_k M_i(k) A_k^T + Q(k),$$
$$\omega_i(k+1) \leftarrow A_k \hat{x}_i(k)$$

6 : (End) Move to step 2 for time t_{k+1}

In Algorithm C, each sensor exchanges the information of input, u_i and U_i, and prediction ω_i to its neighbour. In the simulation results in [Olfati-Saber, 2007], the performance of Algorithm C is shown to be improved.

4.4 Consensus-second order filter

In this section, we develop distributed algorithms for a class of nonlinear systems. In section 4.3, we presented some relevant techniques for a class of distributed linear dynamical systems using local Kalman filters and consensus filters. We propose to

develop a distributed nonlinear estimation technique that combines the second order observer and consensus filters. The approach is similar to the technique proposed in sections 4.2 and 4.3. The second order filtering iterations for sensor i are of the form:

$$\dot{x}(t) = Ax(t) + x^T(t)Fx(t) \tag{4.49}$$

$$m(t) = Cx(t) + v(t) \tag{4.50}$$

We design the algorithms that consider the continuous time nature of (4.49) and (4.50), while inferring estimates in discrete-time to account for fixed sampling rates. The continuous-time second order filters are given by:

$$\dot{\omega}(t) = A\omega(t) + \omega(t)^T F_i \omega(t) + tr(FS(t)) \tag{4.51}$$

$$\dot{S}(t) = (A + 2\omega(t)^T F)S(t) + S(t)(A + 2\omega(t)^T F)^T \tag{4.52}$$

The filter variable $\omega(t)$ is an estimate of $x(t)$ between measurements and it is expressed as $\omega(t_k) = \hat{x}_k$.

4.4.1 Distributed second order Filter Type I : Consensus-Based Fusion of Sensory Data

In a distributed algorithm, we use the discrete-time version of the measurement and covariance. We first compute two sets of data at each sensor i and j at time t_k based on initial conditions. Each sensor invites data from neighbouring sensors to collect the measurement data for the low-pass consensus filter and the covariance data for the band-pass consensus filter. The output $y_i(k)$ and $S_i(k)$ of the consensus filter

gives the consensus state expressed as $y(k)$ and $S(k)$ in Algorithm A-I. Using the second order filter, we estimate the state using the discrete-time recursion at time t_k as follows:

$$G_k = S_k C^T [C S_k C^T + R_k]^{-1} \tag{4.53}$$

$$\hat{x}_k = \omega_k + G_k [y_k - C\omega_k] \tag{4.54}$$

$$\Sigma_k = S_k - S_k C^T [C S_k C^T + R_k]^{-1} C S_k \tag{4.55}$$

Algorithm A-I and Algorithm A-II are distributed algorithms of the second order filter (4.51) $\sim$ (4.55) with consensus filter for nonlinear systems.

[**Algorithm A-I**] Distributed second order Filtering Algorithms with consensus filtering of the state information.

1 : Set initial states at time t_0 : x_0, ω_0, S_0, m_o

2 : Produces new data in neighbourhoods, sensors i and j at time t_k. : $x(k), \omega(k), S(k), m(k)$

3 : Invite measurement data from neighbour sensors and update it:

$$\begin{aligned} u_j &= Cx_j(k) + v_j(k) \quad \forall j \in N_i \cup \{i\} \\ c_i &\leftarrow c_i + \epsilon \sum_{j \in N_i} [(c_j - c_i) + (u_j - c_i)] \\ y_i &= c_i + u_i \end{aligned}$$

4 : Invite new covariance data from neighbour sensors and update it:

$$\begin{aligned} U_j &= S_j, \quad \forall j \in N_i \cup \{i\} \\ \dot{v}_i &= -\hat{L}v_i - \hat{L}U_i, \\ g_i &= v_i + U_i \\ b_i &\leftarrow b_i + \epsilon \sum_{j \in N_i} [(b_j - b_i) + (g_j - b_i)] \\ S_i &= b_i + U_i \end{aligned}$$

5 : Calculate the gain and estimate of the state:

$$\begin{aligned} G_i(k) &= S_i(k)C^T[CS_i(k)C^T + R_i(k)]^{-1} \\ \hat{x}_i(k) &= \omega_i(k) + G_i(k)[y_i(k) - C\omega_i(k)] \\ \Sigma_i(k) &= S_i(k) - S_i(k)C^T[CS_i(k)C^T + R_i(k)]^{-1}CS_i(k) \end{aligned}$$

6 : Update the state for time t_{k+1}

$$\begin{aligned} S_i(k+1) &\leftarrow \Sigma_i(k) \\ \omega_i(k+1) &\leftarrow \hat{x}_i(k) \end{aligned}$$

7 : (End) Move to step 2 for time t_{k+1}

Algorithm A-I enables the fusion of measurement and covariance data. It is assumed that each sensor has white Gaussian noise. In this case, collecting measurement data to come up with the consensus states is not necessary since the measurement at each sensor is meant to be different. Therefore, we keep each different measurement value

to estimate the state. Algorithm A-II shows that one can compute the average $S(k)$ without calculating $y(k)$.

[**Algorithm A-II**] Distributed second order Filtering Algorithms with consensus filtering of the covariance information only.

---

1 : Set initial states at time t_0 : x_0, ω_0, S_0, m_o

2 : Produces new data in neighbourhoods, sensors i and j at time t_k. : $x(k), \omega(k), S(k), m(k)$

3 : Invite new covariance data from neighbour sensors and update it:

$$
\begin{aligned}
U_j &= S_j, \quad \forall j \in N_i \cup \{i\} \\
\dot{v}_i &= -\hat{L} v_i - \hat{L} U_i, \\
g_i &= v_i + U_i \\
b_i &\leftarrow b_i + \epsilon\alpha \sum_{j \in N_i} [(b_j - b_i) + (g_j - b_i)] \\
S_i &= b_i + U_i
\end{aligned}
$$

4: Calculate the gain and estimate of the state:

$$
\begin{aligned}
G_i(k) &= S_i(k) C^T [C S_i(k) C^T + R_i(k)]^{-1} \\
\hat{x}_i(k) &= \omega_i(k) + G_i(k)[m_i(k) - C\omega_i(k)] \\
\Sigma_i(k) &= S_i(k) - S_i(k) C^T [C S_i(k) C^T + R_i(k)]^{-1} C S_i(k)
\end{aligned}
$$

5 : Update the state for time t_{k+1}

$$
\begin{aligned}
S_i(k+1) &\leftarrow \Sigma_i(k) \\
\omega_i(k+1) &\leftarrow \hat{x}_i(k)
\end{aligned}
$$

6 : (End) Move to step 2 for time t_{k+1}

In Algorithm A-II, sensor i sends data of the state and input of its consensus filter (band-pass consensus filter). The output $S_i(k)$ of the band-pass filter attempts to provide a consensus state $S(k)$.

4.4.2 Distributed second order Filter Type II : Consensus on Estimation

As discussed in the alternative approach in section 4.4.3, one can also consider a distributed nonlinear system based on the state estimate in situations where the sensor data yields different measurements. This is a type II second order filter.

[**Algorithm B-I**] Distributed Kalman Filtering Algorithms with consensus based on state estimate. [Olfati-Saber, 2007]

---

1 : Set initial states at time t_0 : x_0, ω_0, S_0, y_o

2 : Produces new data in neighbourhoods, sensors i and j at time t_k. : $x(k), \omega(k), S(k), y(k)$

3 : Combine measurement data and covariance matrices locally from neighbour sensors:

$$u_j = Cx_j(k) + v_j(k) \quad \forall j \in N_i \cup \{i\}, \quad y_i = \sum_{j \in J_i} u_j$$

$$U_j = S_j, \quad \forall j \in N_i \cup \{i\}, \quad S_i = \sum_{j \in J_i} U_j$$

4: Calculate the gain and intermediate estimate of the state:

$$G_i(k) = S_i(k)C^T[CS_i(k)C^T + R_i(k)]^{-1}$$
$$\psi_i(k) = \omega_i(k) + G_i(k)[y_i(k) - C\omega_i(k)]$$
$$\Sigma_i(k) = S_i(k) - S_i(k)C^T[CS_i(k)C^T + R_i(k)]^{-1}CS_i(k)$$

5 : Estimate the state by consensus filter.

$$\hat{x}_i(k) = \psi_i(k) + \epsilon \sum_{j \in N_i} (\psi_j(k) - \psi_i(k))$$

6 : Update the state for time t_{k+1}

$$S_i(k+1) \leftarrow \Sigma_i(k)$$
$$\omega_i(k+1) \leftarrow \hat{x}_i(k)$$

7 : (End) Move to step 2 for time t_{k+1}

Assuming that ψ_i is the intermediate state estimate, each sensor exchanges the

information of input, u_i and U_i, and intermediate state estimate ψ_i to its neighbours. As a result, Algorithm B-I tries to approximate state estimates that minimize the differences caused by measurement noise. It implements a consensus step right after the estimation step of the local second order filter. Since measurements and covariance are not collected at each sensor, the process of estimation is faster. In a large-scale sensor network, the local state estimates can provide global estimates that are subject to consensus constraints. Algorithm C-I is a combination of a consensus step a local second order filtering step. This new "Consensus on estimate" is a more rigorous, as well as faster than Algorithm B-I since the information from the local second order filter does not have to be sent to the next step to determine consensus states. We propose to use this algorithm in concert with the higher order filter from Chapter 3. Estimation of states in a linear system is determined relatively easily such that the distributed observer is quite accurate. The performance of the nonlinear distributed observer as designed in this thesis is tested through simulation in the next section.

[**Algorithm C-I**] Iterative second order - Consensus filter : Distributed second order filtering Algorithm based on state estimate.[Olfati-Saber, 2007]

1 : Set initial states at time t_0 : x_0, ω_0, S_0, y_o

2 : Produces new data in neighbourhoods, sensors i and j at time t_k. : $x(k), \omega(k), S(k), y(k)$

3 : Combine measurement data and covariance matrices locally from neighbour sensors:

$$u_j = Cx_j(k) + v_j(k) \quad \forall j \in N_i \cup \{i\}, \quad y_i = \sum_{j \in J_i} u_j$$

$$U_j = S_j, \quad \forall j \in N_i \cup \{i\}, \quad S_i = \sum_{j \in J_i} U_j$$

4: Estimate the state by 2nd order - Consensus filter.:

$$G_i(k) = S_i(k)C^T[CS_i(k)C^T + R_i(k)]^{-1}$$

$$\hat{x}_i(k) = \omega_i(k) + G_i(k)[y_i(k) - C\omega_i(k)] + \epsilon S_i(k) \sum_{j \in N_i} (\omega_j(k) - \omega_i(k))$$

$$\Sigma_i(k) = S_i(k) - S_i(k)C^T[CS_i(k)C^T + R_i(k)]^{-1}CS_i(k)$$

5 : Update the state for time t_{k+1}

$$S_i(k+1) \leftarrow \Sigma_i(k)$$

$$\omega_i(k+1) \leftarrow \hat{x}_i(k)$$

6 : (End) Move to step 2 for time t_{k+1}

4.5 Simulation Example

In this section, we test the performance of the observer for the nonlinear distributed system using Algorithms A-II, Algorithms B-I, and Algorithms C-I. Consider the

nonlinear dynamical system given by:

$$\left.\begin{aligned} \dot{x}_1 &= -x_1 - x_2 x_1 \\ \dot{x}_2 &= -x_3 + x_1^2 \\ \dot{x}_3 &= x_2 \end{aligned}\right\} \quad \text{Main System} \tag{4.56}$$

This system is Lyapunov stable and belongs to the class of systems considered in Chapter 3. That is,

$$\dot{x}(t) = Ax(t) + \sum_{k=1}^{n} e_k x^T(t) F_k x(t)$$
$$m(t) = Cx(t) + \varrho(t)$$

where state $x(t) \in \mathbb{R}^3$, C_i and C_j are different measurement matrices at each sensor, and the measurement $m(t)$ is observed at fixed sampling times. $\varrho(t)$ is discrete white Gaussian noise with zero mean $E\{\varrho(t)\} = 0$ and constant covariance $Cov\{\varrho(t)\} = R(t)$. The measurement matrices are:

$$C_i = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}^T, \quad C_j = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}^T,$$

The matrices A, F_1, F_2, and F_3 are:

$$A = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}, \quad F_1 = \begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad F_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad F_3 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

Let the state estimate be $\omega(t) = \hat{x}(t)$ between observations given by the following system of ordinary differential equations:

$$\dot{\omega} = A\omega(t) + \sum_{k=1}^{n} e_k \omega(t)^T F_k \omega(t) + \sum_{k=1}^{n} e_k tr(F_k S(t)) \tag{4.57}$$

where $S(t)$ is the positive definite solution of the following differential equation:

$$\dot{S}(t) = (A + 2\sum_{k=1}^{n} e_k \omega(t)^T F_k) S(t) + S(t)(A + 2\sum_{k=1}^{n} e_k \omega(t)^T F_k)^T. \tag{4.58}$$

At time t_k, let $S(t_k) = S_k$, $R(t_k) = R_k$, $m(t_k) = m_k$, and $\omega(t_k) = \omega_k$.

[Algorithm A-II]

Invite new covariance data from neighbour sensors and update it:

$$\begin{aligned} U_j &= S_j, \quad \forall j \in N_i \cup \{i\} \\ \dot{v}_i &= -\hat{L} v_i - \hat{L} U_i, \\ g_i &= v_i + U_i \\ b_i &\leftarrow b_i + \epsilon\alpha \sum_{j \in N_i} [(b_j - b_i) + (g_j - b_i)] \\ S_i &= b_i + U_i \end{aligned} \tag{4.59}$$

Calculate estimate of the state

$$\hat{x}_i(k) = \omega_i(k) + G_i(k)[m_i(k) - C\omega_i(k)] \tag{4.60}$$

The state estimates for the next two algorithms at time t_k are given by:

[**Algorithm B-I**]

$$\psi_i(k) = \omega_i(k) + G_i(k)[m_i(k) - C_i\omega_i(k)] \tag{4.61}$$

$$\hat{x}_i(k) = \psi_i(k) + \delta \sum_{j \in N_i} (\psi_j(k) - \psi_i(k)) \tag{4.62}$$

[**Algorithm C-I**]

$$\hat{x}_i(t_k) = \omega_k + G_k[m_k - C_i\omega_k] + \delta S_i(t_k) \sum_{j \in N_i} (\omega_j(t_k) - \omega_i(t_k)) \tag{4.63}$$

where the δ is step size and the gain matrix G_k is given by

$$G_k = S_k C_i^T [C_i S_k C_i^T + R_k]^{-1} \tag{4.64}$$

The covariance matrix of the state variables is described by,

$$\Sigma_k = S_k - S_k C^T [C S_k C^T + R_k]^{-1} C S_k. \tag{4.65}$$

4.5.1 Simulation Results

The dynamical equations of the plant (4.56) and of the filter (4.57) were integrated using ODE15s function. The system $x(t)$, filter variable $\omega(t)$ and covariance $S(t)$ are updated in continuous-time. The state estimates are updated at each sampling time. The observations were taken every second, and the running time as from t=0 to t=10000. During the simulation, the observation white Gaussian noise is generated by random numbers sampled from normal distribution with mean parameter, 0 and

standard deviation parameter 0.1. At time t=0, the initial states were set to be Gaussian random variables and independent. The initial state variable, $x(0)$, the initial filter variables $\omega(0)$ at sensor i and j, and the initial covariance are as follows:

$$x(0) = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}, \quad S(0) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \omega_i(0) = \begin{bmatrix} 4 \\ 3 \\ 2 \end{bmatrix}, \quad \omega_j(0) = \begin{bmatrix} 2 \\ 1 \\ 3 \end{bmatrix},$$

[Algorithm A-II]

Figure 4.3 shows that the estimation of states has very rough behavior, although they estimate states eventually. The solid line is the plant behavior and the two dotted lines show the behavior of the neighbouring sensors. In figure 4.5 and 4.6, the estimation error goes to zero slowly compared to Algorithm B-I and Algorithm C-I. Therefore, the performance of distributed estimation based on fusion of data is relatively less efficient than distributed estimation based on consensus on estimation.

[Algorithm B-I]

In figure 4.7, the state, x_1 converges to zero value while state, x_2 and x_3 display a sinusoidal response. Through the communication between sensor i and j based on estimates as we developed in Algorithm B-I, x_2 and x_3 can be estimated appropriately. Solid line gives the plant behaviour, and two dotted lines are behaviour of neighbouring sensors. They behave in a very similar manner. After the estimation of states by the nonlinear observer at each sensor, a consensus filter is used to reach equilibrium on the consensus states. The overall process yields a fast local estimate with slow consensus. In figure 4.8, we see that the trajectory of nonlinearity do not track properly initially. They are shown to become appropriately as consensus is

reached. Also, figures 4.9 ,4.10 indicate that the errors $x_k - \hat{x}_k$ converge to zero, but relatively slowly.

[**Algorithm C-I**]

Figure 4.11 shows the plant behaviour always with the distributed estimates from the different sensors. The solid line estimates the plant behaviour, and the two dotted lines show the behaviour of the neighbouring sensors. In contrast to figure 4.8, the estimation of the states converge very quickly to the consensus value as shown in figure 4.12. It demonstrates that Algorithm C-I provides an effective estimation process. In comparison to Algorithm B-I, this algorithm provides a faster and simpler mechanism to achieve consensus of the state estimates. In addition, as shown in the figures 4.13 and 4.14, the errors between the states and the estimation, $x_k - \hat{x}_k$, converge to zero values despite relatively large initial errors.

4.5.2 Discussion of Results

A sensor network consists of a large number of sensor nodes. Each sensor computes some level of communication, collects data, and conducts signal processing, which can build up a sensing network. Due to technical problems, such as limited sensor power and computational ability associated with communications between sensors in large-scale networks, one attempts to design a distributed algorithm that will reduce the communication level required between all-to-all links, thus addressing this problem. We applied the second order observer to the distributed system using consensus filter.

We set two different sensor nodes, i and j, and applied two different methods Algorithm B-I and Algorithm C-I to a nonlinear system. The purpose of this method was to check the feasibility for two nodes to communicate with each other to come

up with consensus states in the nonlinear system. As demonstrated in the simulation results, two different sensor nodes communicated properly to get the same estimates. Using these algorithms, if we assume that there exists a significant number of sensor nodes in a system, estimation of states should occur. These algorithms also allowed one to solve problems related to unlimited single sensor energy, as well as communication capacity for sensors. The proposed method has the advantage that it can provide consensus in the face of sensor malfunction and failure.

To achieve these results, we first applied the second order filter and the consensus filter. This process proved to be effective, yet inefficient due to the long processing time required to display the results. We therefore, combined two filters to design the distributed observer for a class of nonlinear system. This method proved more efficient and rigorous due to the fact that the information from the local second order filter does not have to be sent to the next step to determine consensus states.

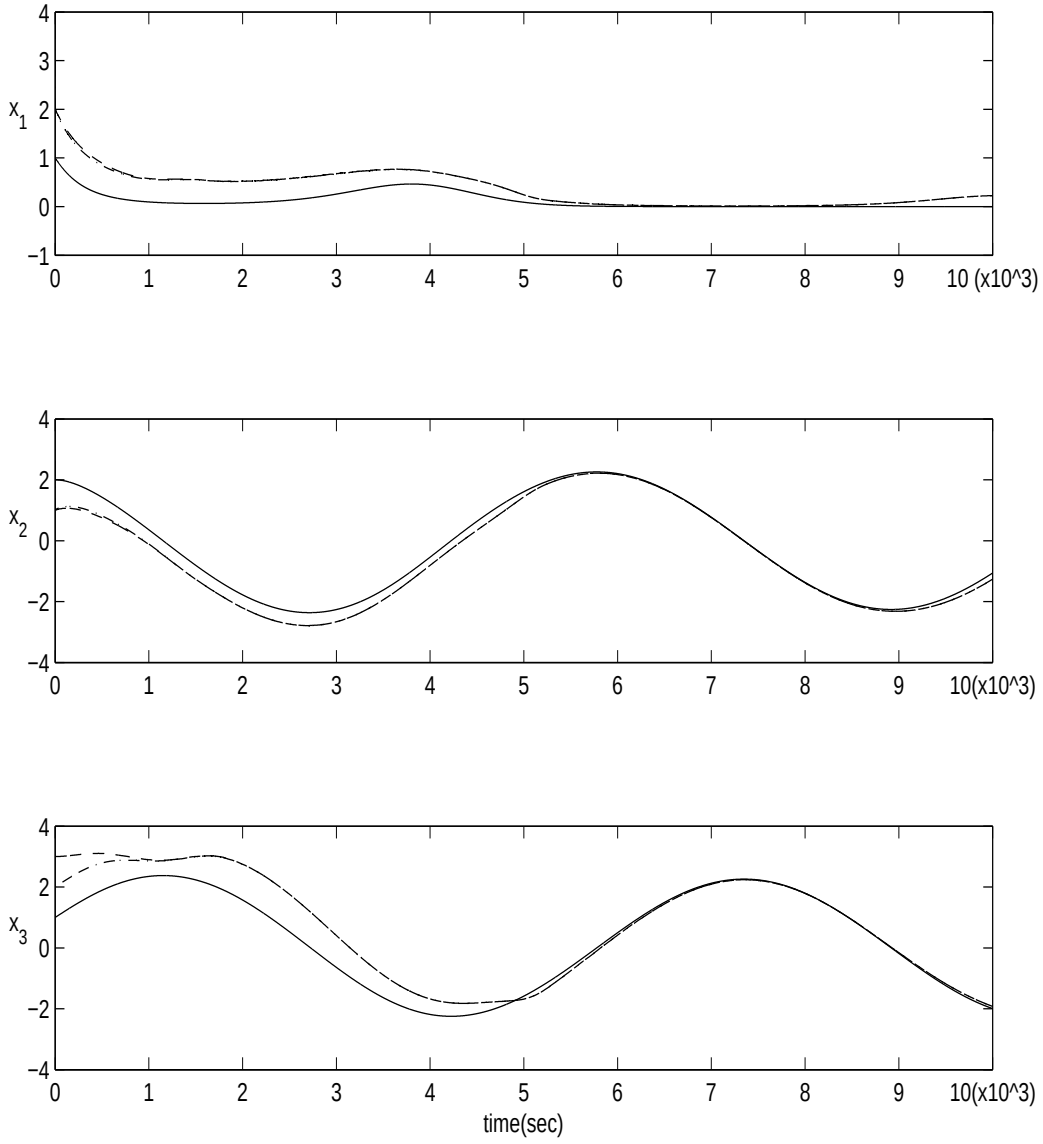

Figure 4.3: [Algorithm A-II] Comparison of the performance of distributed second order kalman filter at different sensor i and j (states "—", estimation (i node) of states "- - -", estimation (j node) of states "- · -")

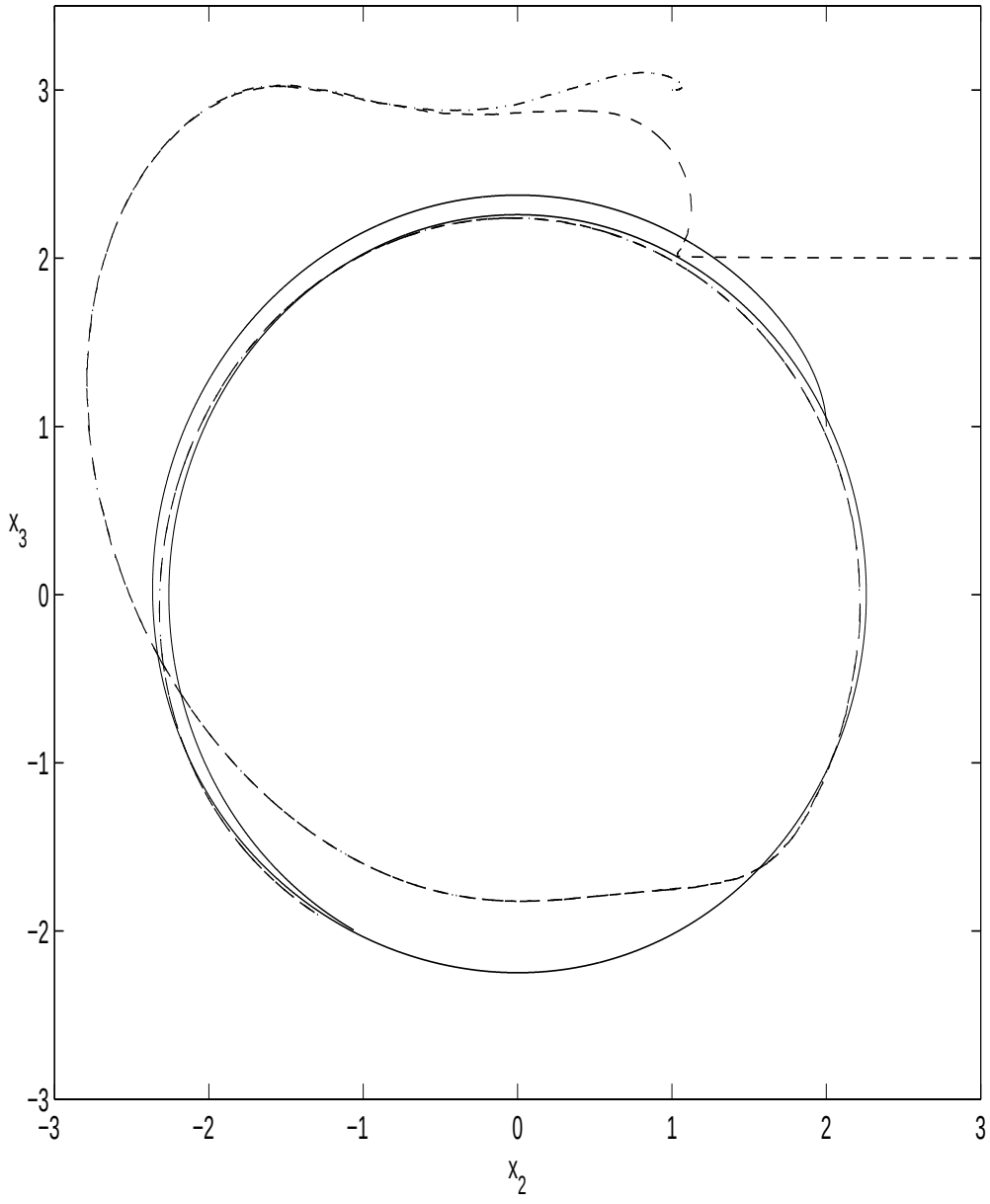

Figure 4.4: [Algorithm A-II] The trajectory behaviour of the nonlinear system for the states x_2, x_3 (states "—", estimation (i node) of states "- - -", estimation (j node) of states "- · -")

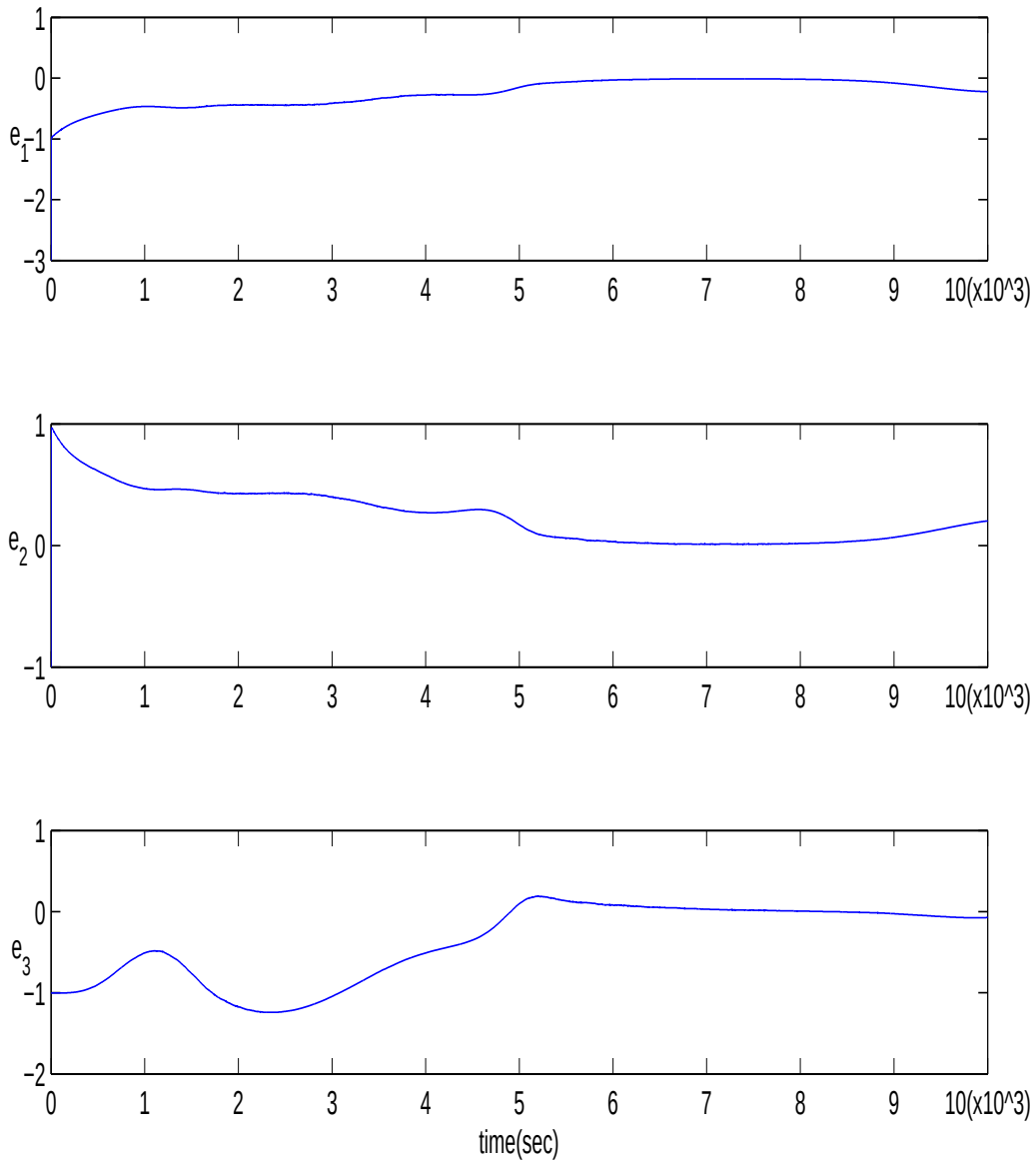

Figure 4.5: [Algorithm A-II] Time course plot of the state estimation error $e = x - \hat{x}$ at sensor i

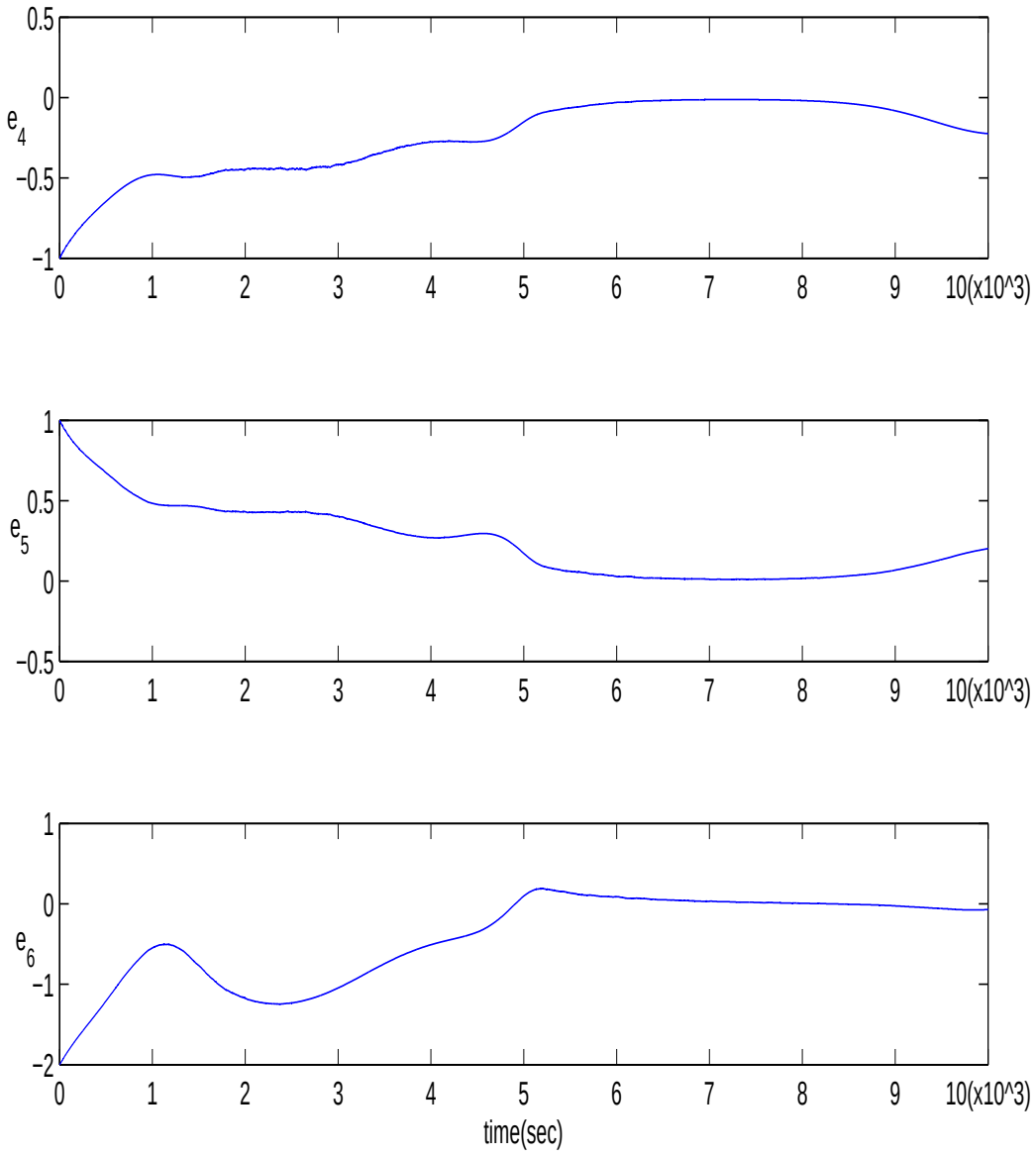

Figure 4.6: [Algorithm A-II] Time course plot of the state estimation error $e = x - \hat{x}$ at sensor j

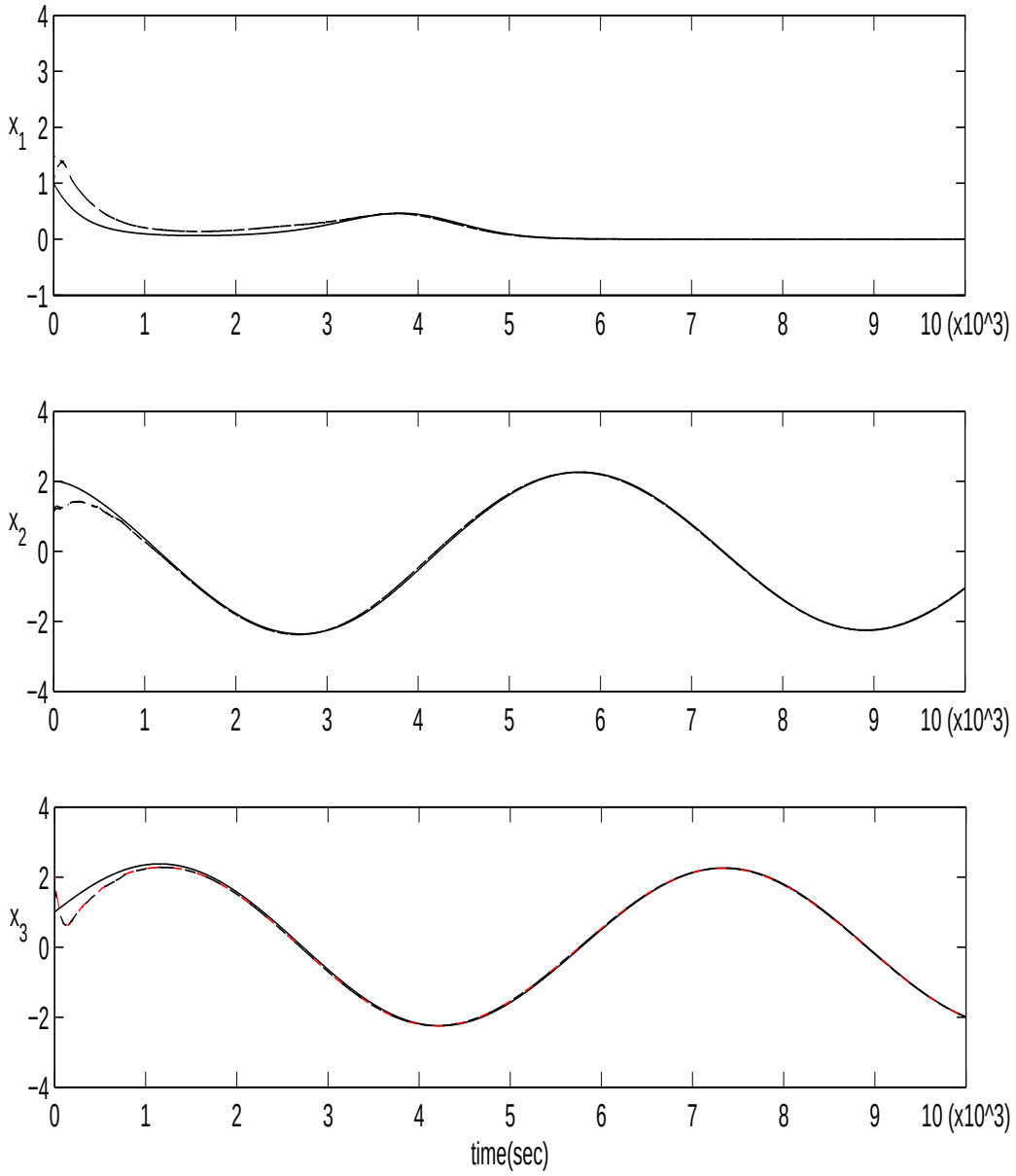

Figure 4.7: [Algorithm B-I] Comparison of the performance of distributed second order kalman filter at different sensor i and j (states "—", estimation (i node) of states "- - -", estimation (j node) of states "- · -")

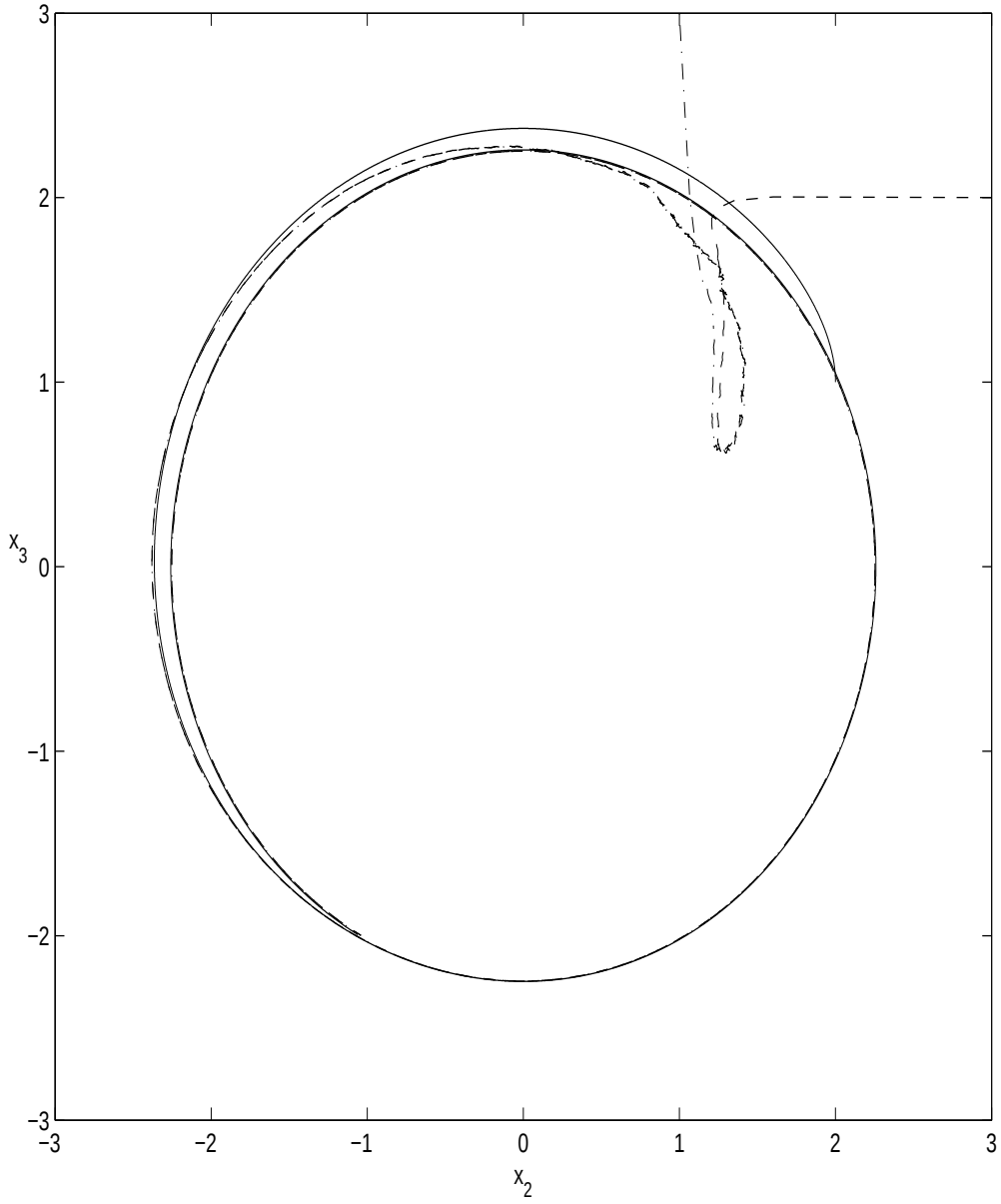

Figure 4.8: [Algorithm B-I] The trajectory behaviour of the nonlinear system for the states x_2, x_3 (states "—", estimation (i node) of states "- - -", estimation (j node) of states "- · -")

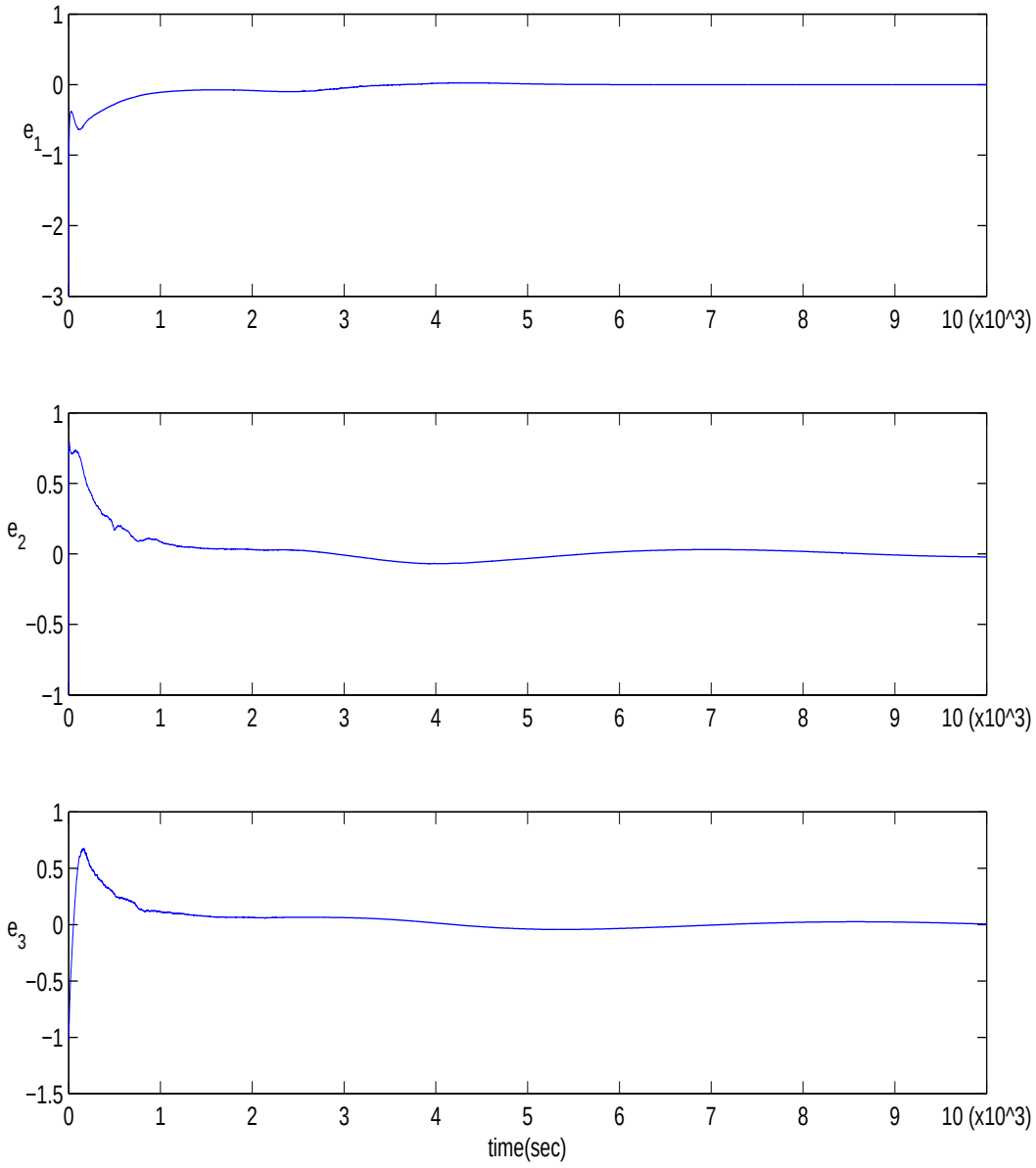

Figure 4.9: [Algorithm B-I] Time course plot of the state estimation error $e = x - \hat{x}$ at sensor i

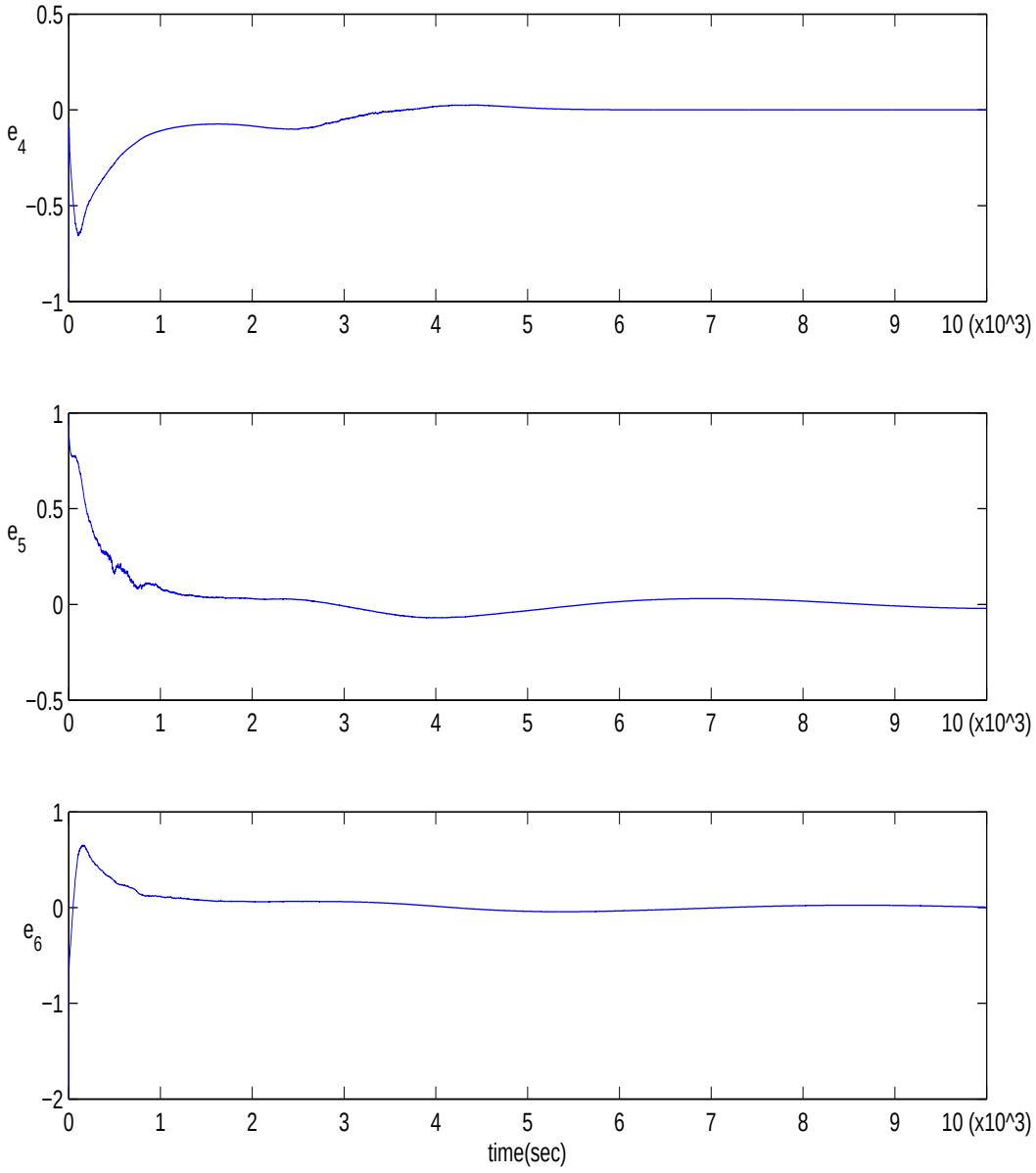

Figure 4.10: [Algorithm B-I] Time course plot of the state estimation error $e = x - \hat{x}$ at sensor j

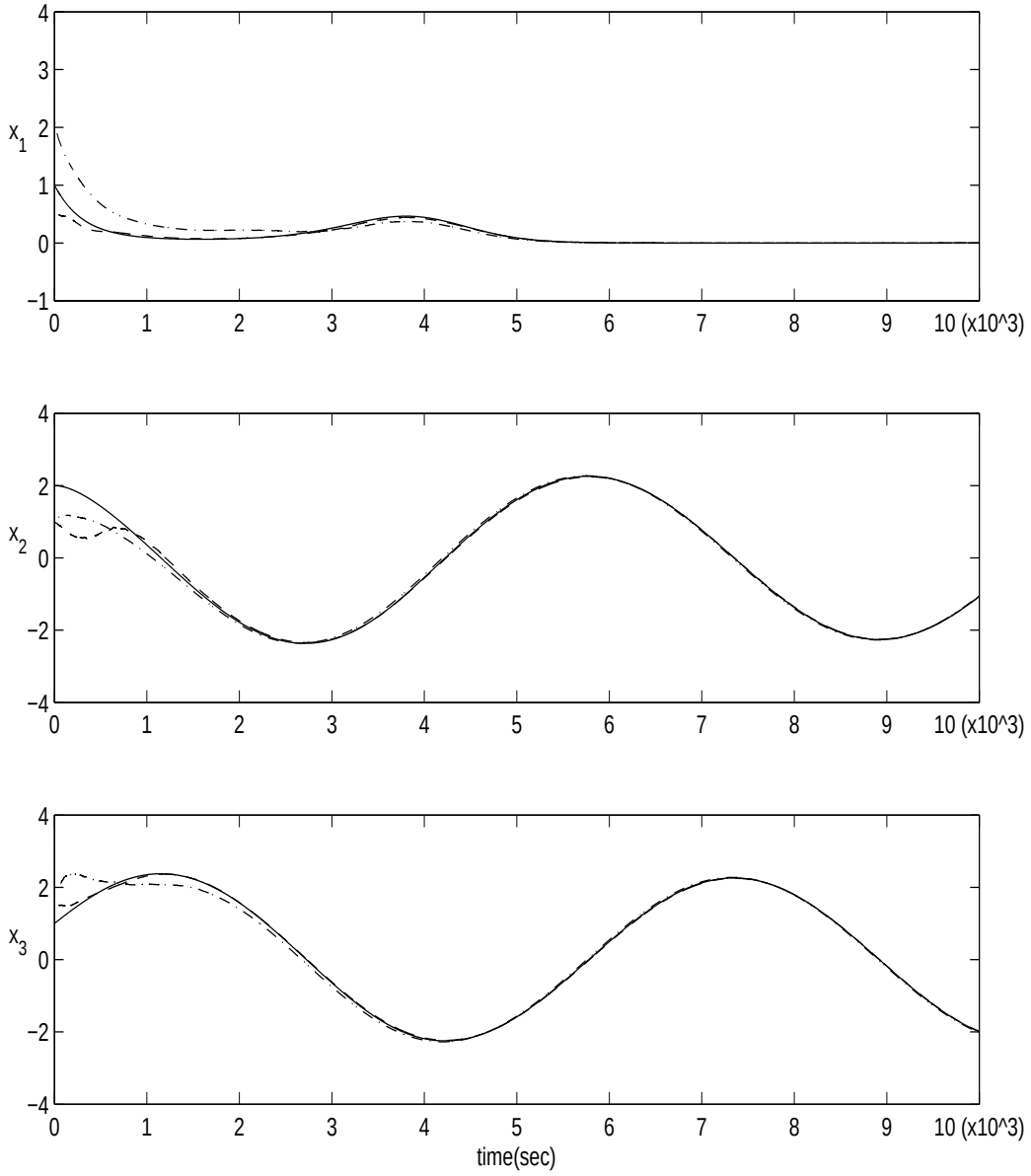

Figure 4.11: [Algorithm C-I] Comparison of the performance of distributed second order kalman filter at different sensor i and j (states "—", estimation (i node) of states "- - -", estimation (j node) of states "- · -")

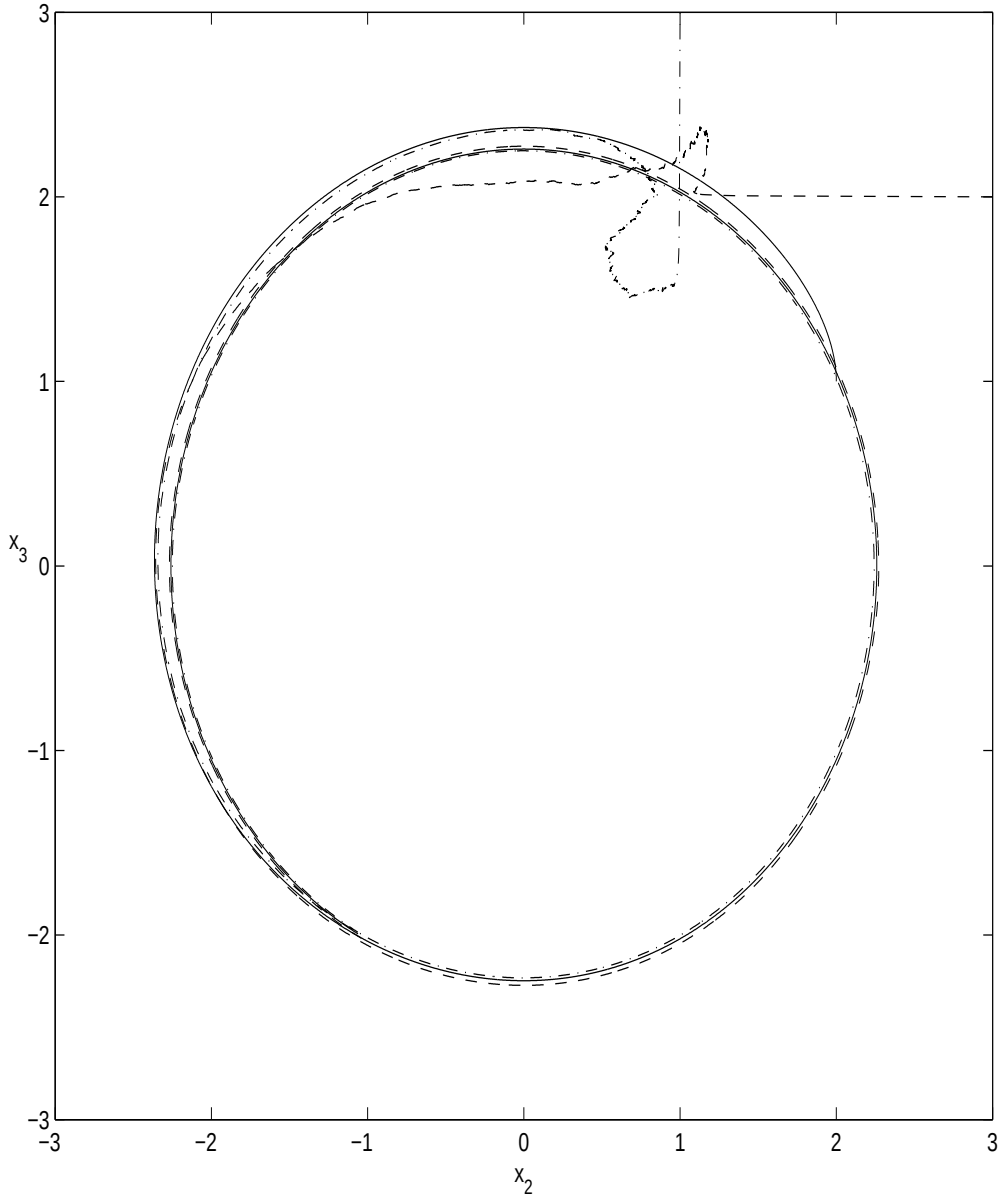

Figure 4.12: [Algorithm C-I] The trajectory behaviour of the nonlinear system for the states x_2, x_3 (states "—", estimation (i node) of states "- - -", estimation (j node) of states "- · -")

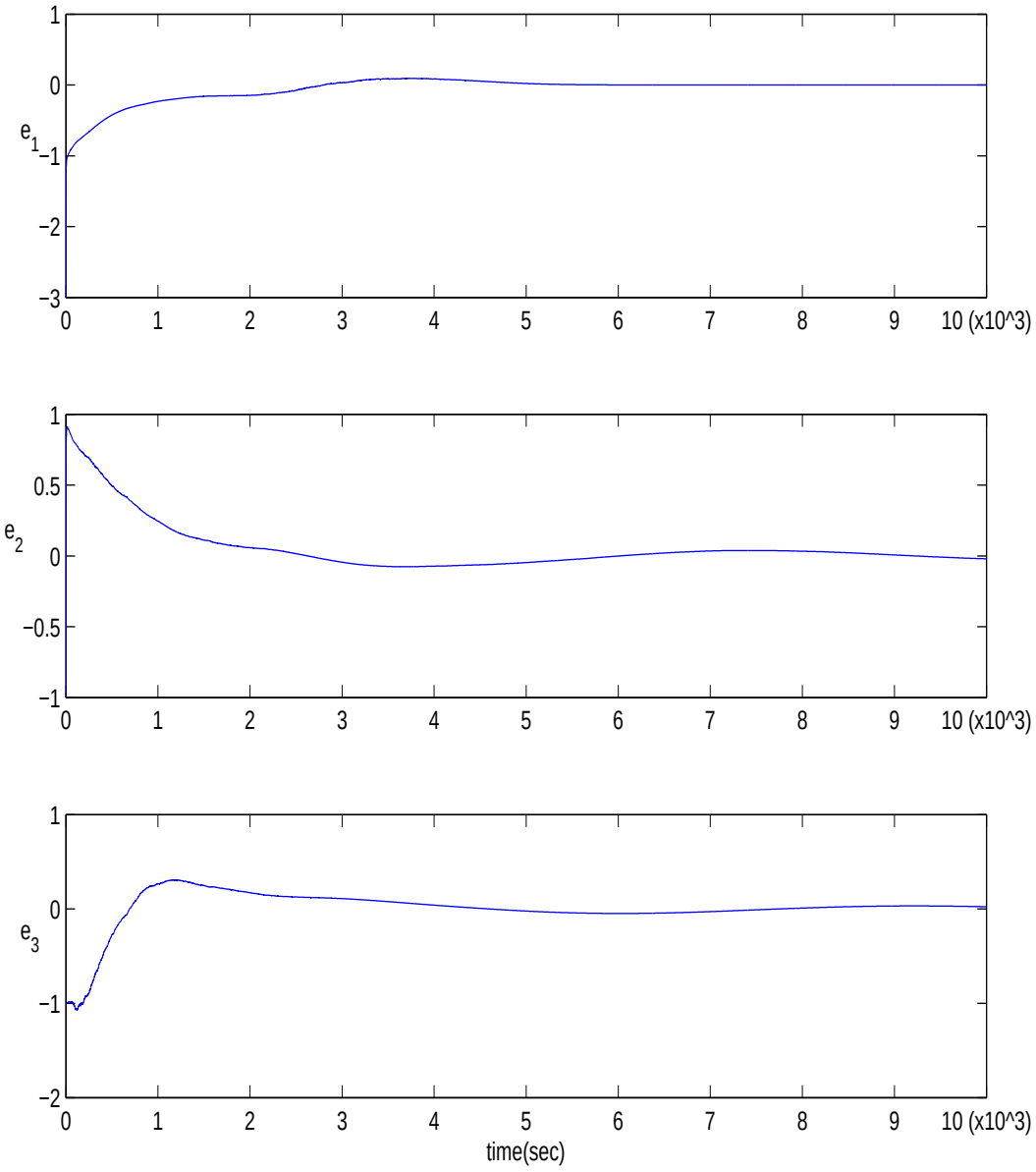

Figure 4.13: [Algorithm C-I] Time course plot of the state estimation error $e = x - \hat{x}$ at sensor i

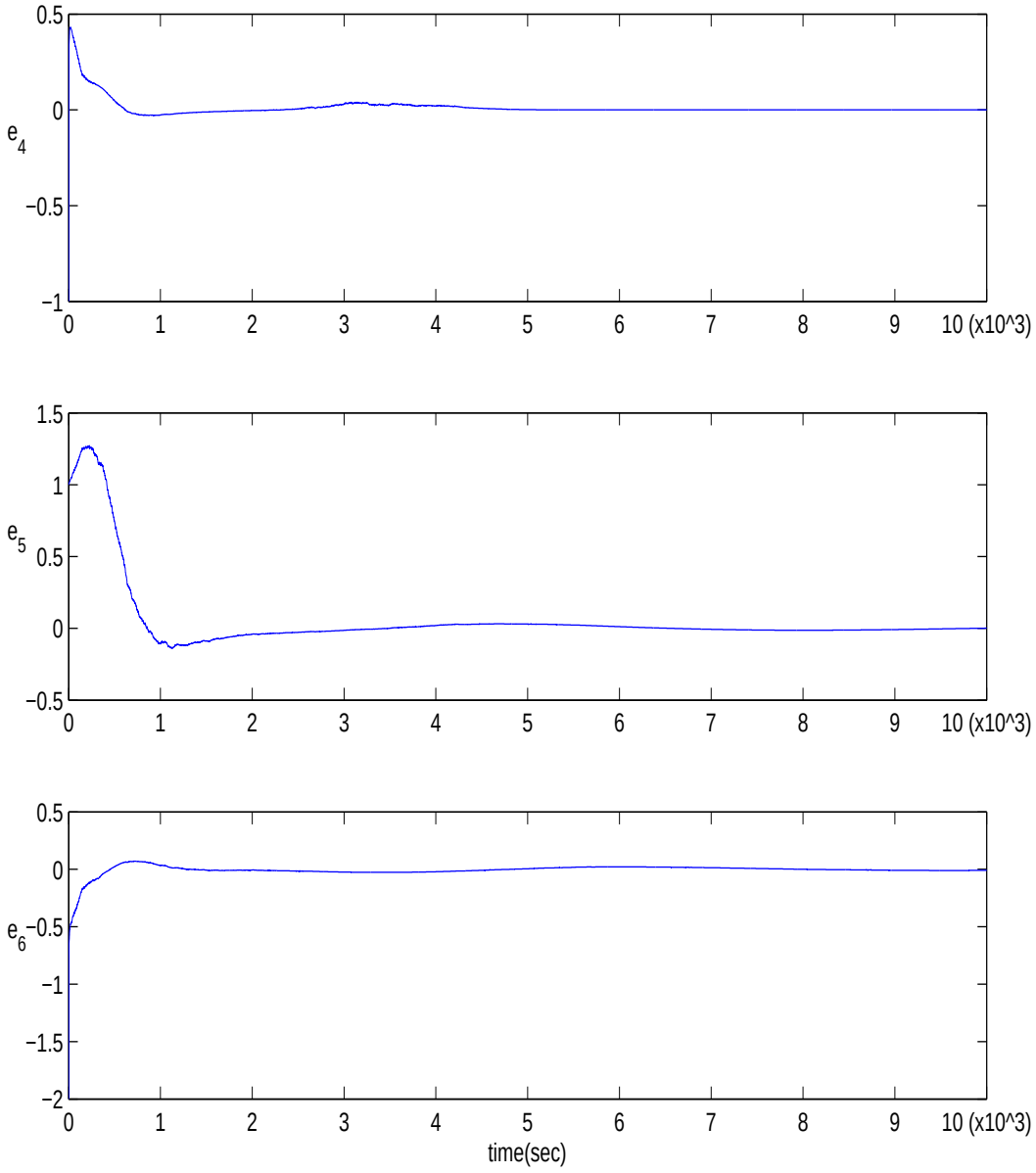

Figure 4.14: [Algorithm C-I] Time course plot of the state estimation error $e = x - \hat{x}$ at sensor j

Chapter 5

Conclusion and Future Work

This thesis has introduced and detailed a new distributed state estimation for a class of a nonlinear systems. The estimation technique was developed in three steps. The first step involved the design of an observer for nonlinear systems that arise in the application of POD-based model reduction techniques. The POD-based technique preserves the nonlinearity of the Navier-Stokes equation. It is shown that we can design nonlinear observer for the combined estimation of velocity field, contaminant field, and energy field.

The second step involved the design of a distributed observer with multiple sensors. The consensus filter was embedded in the local nonlinear observer for the fusion of data of measurement and covariance. Through exchange of information from each neighbouring sensor that exploits the availability of multiple measurements, local state estimates can be obtained. The final step resulted in the design of distributed observer with consensus based on state estimation. In this last step, the distributed observer was added to a consensus filter in the nonlinear observer. The observer from the last step is shown to provide better performance than the nonlinear observer

embedded consensus filter.

The design for an observer for a nonlinear system has emerged from extensive and varied research processes. Extended Kalman filter(EKF) and Unscented Kalman filter(UKF) were considered as an observer for the nonlinear system in [Guay and Hariharan, 2008] and [Guay et al., 2009]. Since EKF is based on a time-varying linear approximation of the nonlinear system, the performance limitations of the EKF are well understood and widely reported. Although the UKF is a nonlinear filter that is able to handle the nonlinearities, the UKF does not significantly outperform an EKF for the dynamical system,nor does it result in a faster process. In contrast, the second order filter does not require any approximation of dynamics other than the model reduction step. As a result, the second order filter provides a much larger region of stability. Therefore, the nonlinear observer can handle larger perturbations and larger errors between the states and the state estimates.

In a distributed system, where a number of sensors are available, one needs to develop new scalable algorithms to estimate states efficiently. Since it appears to be nearly impossible to communicate all-to-all link sensors in a decentralized system, a nonlinear distributed observer is required in application by adding consensus filter. The advantage of the consensus filter is that all sensors do not have to communicate with one another. Only neighbouring sensors within a certain distance can exchange their information to come up with the average states estimation. We demonstrated the new design of distributed observer through developing algorithms. The distributed second order filter type I is collecting data of measurements and covariance from each sensor by communicating. The simulation results show that the estimation of the states is accurate as well as reliable. The objective of the distributed second order

filter of type II is to reduce disagreement of the states estimation by each sensor. The new Algorithms B-I and C-I have better performance than Algorithms A-I & II and the local POD based second order observer. As the simulation results are shown, the last Algorithm C-I provides the best performance.

In future work, the proposed distributed observer should applied to design of a POD-based distributed observer to Navier-Stokes flow and contaminant flow from CFD model. As discussed in Chapter 3, the application of POD-based model reduction techniques yield nonlinear systems that can be readily handled using the proposed second order observer. In most applications, the estimation of flow characteristics rely on a spatially distributed sensor system. Using the results of the thesis, one could conceive a distributed observer design in complex flows in which each sensor can exchange information with its neighbourhoods to obtain an overall estimate of the flow characteristics.

As mentioned earlier, each sensors possesses its own limited energy source. To reduce the consumption of energy, we should consider the optimization of sensor network and sensor ability and capability. In this thesis, the simulation study was limited to a system with two sensor nodes potentially placed at a certain distance of each other. The question of sensor placement and optimization of distance between each sensor was not considered. In addition to the relative location of sensors, one must also consider the connectivity of the communication between sensors. Future work should focus on how best to prioritize the information exchange in order to save energy for activity time of the sensor networks.

Bibliography

B. Acikmese and M. Mandic. Decentralized observer with a consensus filter for distributed discrete-time linear systems. *Proceedings of the American Control Conference*, pages 4723 – 4730, 2011.

B.D.O Anderson and J.B.Moore. *Optimal Filtering*. Prentice-Hall. Englewood Cliffs. NJ, 1979.

T. Arampatzis, J. Lygeros, and S. Manesis. A survey of applications of wireless sensors and wireless sensor networks. *Proceedings of the IEEE Mediterranean Conference on Control and Automation*, 3:719 – 724, 2005.

M. Athans, R.P. Wishner, and A. Bertolini. Suboptimal state estimation for continuous-time nonlinear systems from discrete noisy measurements. *IEEE Transactions on Automatic Control*, 13:504 – 514, 1968.

G. Berkooz and E. S. Titi. Galerkin projections and the proper orthogonal decomposition for equivariant equations. *Physics Letters A*, 174:94 – 102, 1993.

R. B. Bird, W. E. Stewart, and E. N. Lightfoot. *Transport Phenomena*. Wiley, 1960.

L.G. Bleris and M.V. Kothare. Low-order empirical modeling of distributed parameter system using temporal and spatial eigenfunctions. *Computers and Chemical Engineering*, 29:817 – 826, 2005.

D.W. Casbeer. Decentralized estimation using information consensus filters with a multi-static uav radar tracking system. *Phd Thesis*, 2009.

G. E. Dullerud and F. Paganini. *A Course in Robust Control Theory: A Convex Approach*. Springer, 2000.

C. D. Godsil and G. Royle. *Algebraic Graph Theory*. Springer, 2001.

M. Guay and N. Hariharan. Airflow velocity estimation in building systems. *Proceedings of the American Control Conference*, pages 908 – 913, 2008.

M. Guay, T. John, and N. Hariharan. Pod based observer for contaminant flow estimation in building systems. *Proceedings to the American Control Conference*, pages 4642 – 4647, 2009.

M. Guay, T. John, N. Hariharanc, and S. Naranayan. Pod-based observer for estimation in navier - stokes flow. *Computers and Chemical Engineering*, 34:965 – 975, 2010.

J. Hoepffner, M. Chevalier, R. Bewley, and D.S. Hennington. State estimation in wall-bounded flow systems. part 1. perturbed laminar flow. *Journal of Fluid Mechanics*, 534:263–294, 2005.

J. Hoepffner, M. Chevalier, R. Bewley, and D.S. Hennington. State estimation in wall-bounded flow systems. part 2. tubular flow. *Journal of Fluid Mechanics*, 552: 167–187, 2006.

R. E. Kalman. On the general theory of control systems. *IRE Transactions on Automatic Control*, 4:110, 1960.

H. K. Khalil. *Nonlinear Systems*. Prentice-Hall, New Jersey, 2002.

J. L. Lumley. *Stochastic Tools in Turbulence*. Academic Press, New York, 1970.

A. G.O. Mutambara. *Decentralized Estimation and Control for Multisensor Systems*. Taylor & Francis, 1998.

N.A.Lynch. *Distributed Algorithms*. Morga Kaufmann Publishers, Inc, 1997.

B.R. Noak, P. Pappas, P.A. Monkewitz, M. Morzynski, and G. Tadmor. Empirical galerkin models for incompressible flow pressure-term and subgrid turbulence representations. *XXI International Congress of Theoretical and Applied Mechanics, Warsaw, Poland*, 2004.

B.R. Noak, P. Pappas, and P.A. Monkewitz. The need for a pressure-term representation in empirical galerkin models of incompressible shear flows. *Journal of Fluid Mechanics*, 523:339 – 365, 2005.

R. Olfati-Saber. Ultrafast consensus in small-world networks. *Proceedings of the American Control Conference*, 4:2371 – 2378, 2005a.

R. Olfati-Saber. Distributed kalman filter with embedded consensus filters. *Proceegings of the 44th IEEE Conference on Decision and Control*, pages 8179 – 8184, 2005b.

R. Olfati-Saber. Distributed kalman filtering for sensor networks. *Proceedings of the 46th IEEE Conference on Decision and Control*, pages 5492 – 5498, 2007.

R. Olfati-Saber and R.M.Murray. Consensus protocols for networks of dynamic agents. *Proceedings of the American Control Conference*, 2:951–956, 2003.

R. Olfati-Saber and R.M.Murray. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Transactions on Automatic Control*, 49 , Issue: 9:1520 – 1533, 2004.

R. Olfati-Saber and J. S. Shamma. Consensus filters for sensor networks and distributed sensor fusion. *Proceedings to the 44th IEEE Conference on Decision and Control*, pages 6698 – 6703, 2005.

B.S.Y. Rao, H.F. Durrant-Whyte, and J.A Sheen. A fully decentralized multi-sensor system for tracking and surveillanc. *Interntational Journal of Robotics Research,*, 12:20 – 44, 1993.

C. W. Rowley. Model reduction for fluids, using balanced proper orthogonal decomposition. *International Journal of Bifurcation and Chaos*, pages 997 – 1013, 2005.

C. W. Rowley and V. Juttijudata. Model-based control and estimation of cavity flow oscillations. *44th IEEE Conference on Decision and Control*, pages 512 – 517, 2005.

C. W. Rowley, T. Colonius, and R. M. Murray. Pod based models of self-sustained oscillations in the flow past an open cavity. *Proceedings of the 6th AIAA/CEAS Aeroacoustics Conference*, pages 2000 – 1969, 2000.

C. W. Rowley, T. Colonius, and R. M. Murray. Model reduction for compressible flows using pod and galerkin projection. *Physica D, Nonlinear Phenomena*, 189(1Ű2): 115 – 129, 2004.

D. Simon. *Optimal State Estimation*. Wiley, 2006.

J. L. Speyer. Computation and transmission requirements for a decentralized linear-quadratic-gaussian control problem. *Proceedings of the IEEE Conference on Decision and Control*, 17:1126 – 1131, 1978.

C. Tin and C.-S. Poon. Internal models in sensorimotor integration: Perspectives from adaptive control theory. *Journal of Neural Engineering*, 2:S147 – S163, 2005.

W. Yu, G. Chen, Z. Wang, and W. Yang. Distributed consensus filtering in sensor networks. *IEEE Transaction on Systems, Man, and Cybernetics, Part B : Cybernetics*, 39(6):1568 – 1577, 2009.