UNIFYING CONSENSUS AND COVARIANCE INTERSECTION FOR EFFICIENT DISTRIBUTED STATE ESTIMATION OVER UNRELIABLE NETWORKS

A Thesis

by

AMIRHOSSEIN TAMJIDI

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Suman Chakravorty
Committee Members,	Dylan Shell
	John Hurtado
	John Valasek
Head of Department,	Rodney Bowersox

May 2017

Major Subject: Aerospace Engineering

ABSTRACT

This thesis studies the problem of recursive distributed state estimation over unreliable networks. The main contribution is to fuse the independent and dependent information separately. Local estimators communicate directly only with their immediate neighbors and nothing is assumed about the structure of the communication network, specifically it need not be connected at all times. The proposed estimator is a Hybrid one that fuses independent and dependent (or correlated) information using a distributed averaging and iterative conservative fusion rule respectively. It will be discussed how the hybrid method can improve estimators's performance and make it robust to network failures.

The content of the thesis is divided in two main parts. In the first part I study how this idea is applied to the case of dynamical systems with continuous state and Gaussian noise. I establish bounds for estimation performance and show that my method produces unbiased conservative estimates that are better than Iterative Covariance Intersection (ICI). I will test the proposed algorithm on an atmospheric dispersion problem, a random linear system estimation and finally a target tracking problem.

In the second part, I will discuss how the hybrid method can be applied to distributed estimation on a Hidden Markov Model. I will discuss the notion of conservativeness for general probability distributions and use the appropriate cost function to achieve improvement similar to the first part. The performance of the proposed method is evaluated in a multi-agent tracking problem and a high dimensional HMM and it is shown that its performance surpasses the competing algorithms.

ACKNOWLEDGEMENTS

First, I would like to thank my advisor, Dr. Suman Chakravorty and my co-advisor Dr. Dylan Shell. Suman has been an excellent teacher, mentor, and a trustworthy friend. He was very patient and supportive at the time that I was having several surgeries on my eyes. He always inspires me to get a deeper understanding of my research topic and to become independent. Dylan was a great co-advisor, an encouraging and enthusiastic professor who taught me to look at the same problem from different angles. I greatly value the time and constructive discussions that Suman and Dylan had with me during our meetings. Without their guidance, I could have not finished this thesis. I would like also to thank Dr. Valasek and Dr. Hurtado who accepted to be in my thesis committee.

My colleagues and friends in EDPLab and the DNC group of the aerospace department made my experience very pleasant. I had the privilege to work with Saurav Agarwal, Dan Yu, Mohammadhussein Rafieisakhaei, Reza Oftadeh during my studies. Their invaluable friendship will always be remembered. My wife, Yasaman, supported me throughout writing this thesis and her unassuming love remained incessant like always. My old friend Ali and Negar have impacted my life in a very positive way and no word can do justice to my appreciation of their friendship.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a thesis committee consisting of Dr. Suman Chakravorty from Aerospace Department (advisor), Dr. Dylan Shell from CSCE Department (co-advisor), Dr. Valasek and Dr. Hurtado, both from the Aerospace Department. A code provided by Dan Yu is used in the experiment section of chapter 2. Reza Oftadeh was a collaborator in the work presented in chapter 3.

Funding Sources

Graduate study was supported by a Graduate Assistantship from Texas A&M University and funding from NSF.

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

1. INTRODUCTION

1.1 Definitions

Some theoretical background that will be used throughout the rest of this thesis is reviewed in this chapter. The content is taken mainly from [9, 7, 25] and the interested reader can find further details therein.

Definition 1. A *graph* is an ordered pair $G = \langle \mathcal{V}, \mathcal{E} \rangle$ where, $\mathcal{V}$ and $\mathcal{E}$ are the set of graph nodes and edges respectively. If $(v_i, v_j) \in \mathcal{E}$, it means nodes v_i and v_j are connected. *in-Neighbors* of node v_i are defined as[1]

$$\mathcal{N}^i = \{\forall v_j \in \mathcal{V}, (i,j) \in \mathcal{E}\}. \tag{1.1}$$

Also, $|\mathcal{N}^i|$ is the *cardinality* of $\mathcal{N}^i$. The degree of a node v_i, denoted d_i, is the number of edges incident to it.

Definition 2. If a weight is assigned to each edge of the graph $G = \langle \mathcal{V}, \mathcal{E} \rangle$, the result is called a *weighted graph*. An example weighted graph is shown in Fig. 1.1.

1.2 Consensus Algorithm

One of the fundamental problems in a network of agents is to reach consensus over a decision or opinion. Any solution is constrained by a number of factors including, network communication topology and the agent's knowledge about it. Agents can be static or dynamic (moving) and represent sensors, robots, UAVs, etc. One example application is depicted in Fig. 1.2 where a flock of heterogeneous robots

[1]The traditional definition for the node v_if neighbors in graph theory excludes node v_i. I choose this definition since it results in more condensed formulas.

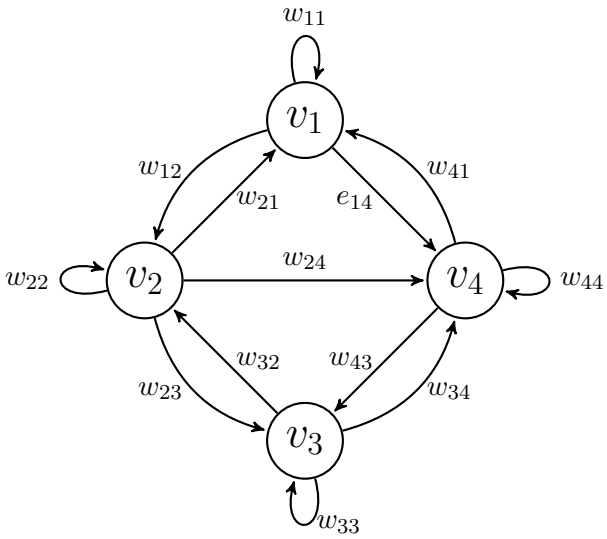


Figure 1.1: An example directed graph with four nodes

are moving towards a goal. They are communicating to each other and making decisions collectively. The ground robots get disconnected from the Quadrotors at some point and they get back together later. The group is trying to keep a specific formation while moving toward a goal position.

The distributed consensus problem's objective is to devise algorithms for local processing and messaging protocol such that by following them, nodes reach an agreement over an opinion (usually represented by a scalar, vector or a matrix). One of the main categories of consensus problems is the *distributed averaging problem*.

1.3 Distributed Averaging

Definition 3. Distributed Averaging: Assume that we have a network with n nodes whose time varying communication topology is denoted by $G_k = \langle \mathcal{V}, \mathcal{E}_k \rangle$. The i'th component of vector $\mathbf{x}(0) = [x^1(0), \cdots, x^n(0)]$ represents the initial value at node i. Let $\mathbf{x}_{ave} = \sum_{i=1}^{n} x^i$ be the average node value. The goal in *distributed averaging* is to

Figure 1.2: A typical robotics scenario in which reaching to a target point is the objective.

calculate x_{ave} in a distributed manner.

The following linear update formula is utilized to update node values

$$x^i(k+1) = \sum_{j=1}^{n} \gamma_{ij}(k) x^j(k). \tag{1.2}$$

In [25] the conditions for convergence of this update rule to $\mathbf{x}_{ave}$ is studied. The following assumptions and theorem summarizes the results relevant to this thesis.

Assumption 1.3.1. There exists a positive constant α such that:

(a) $\gamma_{ii}(k) \geq \alpha$, for all i, t.

(b) $\gamma_{ij}(k) \in \{0\} \cup [\alpha, 1]$, for all i, j, t.

(c) $\sum_{j=1}^{n} \gamma_{ij}(k) = 1$, for all i, k.

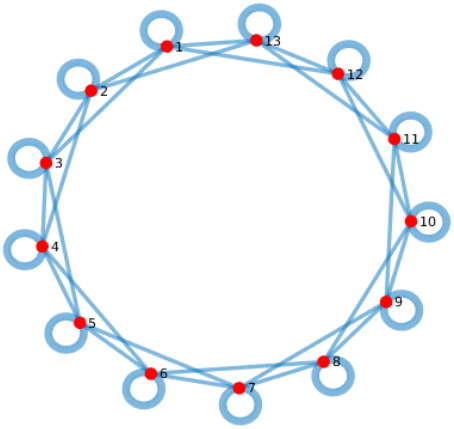
Assumption 1.3.2 (*Bounded interconnectivity times*)**.** There is some B such that for all k, the graph $\langle \mathcal{V}, \mathcal{E}(kB) \cup \mathcal{E}(kB+1) \cup \cdots \cup \mathcal{E}((k+1)B-1) \rangle$ is strongly connected.

Theorem 1.3.3. *Under Assumptions 1.3.1 and 1.3.2, the update rule 1.2 guarantees asymptotic consensus, that is, there exists some c (depending on $\mathbf{x}(0)$ and on the sequence of graphs $G(\cdot)$) such that $\lim_{k\to\infty} x^i(k) = c$, for all i.*

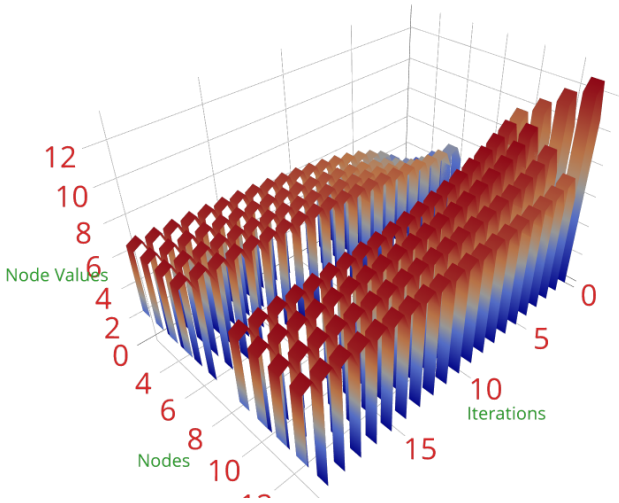
If the network topology is fixed, i.e., $G(k) = G$ for all k, one can associate a Markov Chain with network G, and assign transition probabilities of the Markov Chain as edge weights. Then, the consensus update rule can be written as

$$\mathbf{x}(k+1) = \Gamma \mathbf{x}(k), \tag{1.3}$$

where Γ is the transition probability matrix. It can be shown that if Γ is a *doubly stochastic matrix*, i.e. its rows and columns sum to one, the consensus algorithm is guaranteed to converge to $\mathbf{x}_{ave}$. A particular choice of a doubly stochastic matrix based on a Metropolis Hastings Markov Chain (MHMC) has been proposed in [32, 34]. The advantage of MHMC based distributed averaging is that the weights are determined only based on local information and no global knowledge of network topology is required. An example application has been shown in Fig. 1.3. In a network consisting of $n = 13$ nodes with a topology shown in 1.3(a), nodes can reach a consensus over the average initial node values through the MHMC-based consensus method. The evolution of the node values is depicted in 1.3(b). Nodes converge to the same value only using local message passing.



(a) Network Topology



(b) The evolution of node values from their initial value to $\mathbf{x}_{ave}$

Figure 1.3: Evolution of node values for a distributed averaging algorithm.

2. LINEAR/LINEARIZED SYSTEMS AND GAUSSIAN NOISE

This chapter expands on the material published in reference [28][1]

2.1 Related Work

This chapter studies distributed estimation using multiple robotic agents with applications to the estimation of a dynamic random field. When the field dynamics can be described by a linear, lumped-parameter model, the classical solution is the Kalman filter (KF). However, bandwidth and energy constraints may preclude the centralized implementation of such a filter and necessitate the design of a distributed estimator.

In general, a distributed sensor network cannot achieve the estimation quality of a centralized estimator but is inherently more flexible and robust to network failure and consequently is advantageous in certain applications [36].

In distributed estimation settings, the system comprises a set of nodes connected to each other through a communication network with some topology. Nodes are assumed to make noisy observations of a global state from which the full state of the system cannot necessarily be recovered. The goal is to design local estimators that can recursively calculate an estimate of the global state with access only to the information locally available to nodes. We desire that estimates be conservative and the estimator be consistent. No prior knowledge about the network topology is assumed.

When the topology of the network is known *a priori* and it remains connected throughout, some existing methods recover the centralized estimator's performance

[1] Reprinted, with permission, from Amirhossein Tamjidi, Suman Chakravorty, and Dylan Shell, "Unifying consensus and covariance intersection for decentralized state estimation" In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 125–130, 2016.

[23, 24] for dynamic state estimation. However, such methods are not applicable for the case where the network does not remain connected all the time.

For static state/parameter estimation Xia, et al., introduce a method based on distributed averaging that can converge to the global state estimator provided that the infinitely occurring communication graphs are jointly connected [33]. This method relies on the distributed averaging property of Metropolis-Hastings Markov Chains (MHMC). The advantage of it is that the network topology can be dynamically changing and need not be connected at all times. The local estimators exchange information only with their immediate neighbors and remain agnostic about the topology of the rest of the network. Their work is limited to static field/parameter estimation. In the dynamic state estimation, when the network becomes disconnected, the estimate priors can drift away while they still have some mutual information. Performing distributed averaging on those priors is incorrect since it results in multiple counting of mutual information. In order to solve this problem one would have to resort to distributed estimators that account for the correlations between local estimates.

In [11], a Distributed Delayed-State Extended Information Filter (DDSEIF) is described that handles the correlation between local estimates. This method only works in directed networks that do not have any loops. It is claimed that under certain assumptions local estimates would converge to the centralized estimate. However, the method requires a large amount of data communication, storage memory, and book-keeping overhead, and therefore, does not lend itself to online resource constrained recursive distributed state estimation.

Another approach to deal with the correlation of local estimates is to use Covariance Intersection (CI) methods [14] that produce conservative estimates in the absence of correlation knowledge. The work in references [30, 14, 12, 20, 16, 15] falls into this category. They propose different optimization criteria to perform CI and/or use

different iterative CI schemes for distributed state estimation.

The downside of distributed CI based methods is that they produce overly conservative estimates by unnecessarily performing the covariance intersection on generally uncorrelated new information at the current step. This incurs significant performance loss compared to MHMC-based distributed averaging, which is a superior way to reach consensus on uncorrelated information.

2.2 Motivating Example

In fig. 2.1 a motivating example is given for the method proposed in this chapter. Consider an atmospheric dispersion scenario as an example where there exists 6 pollutant sources and 8 receptor distributed in the field connected to each other through a time varying graph. At first, all receptors are connected and all the nodes reach a consensus over the field estimate. Later, for a time interval, we have two disconnected groups. The sensors in each group continue receiving new information and calculate their local estimates to the best of their knowledge, After some time the network becomes connected again and agents in each group will get access to

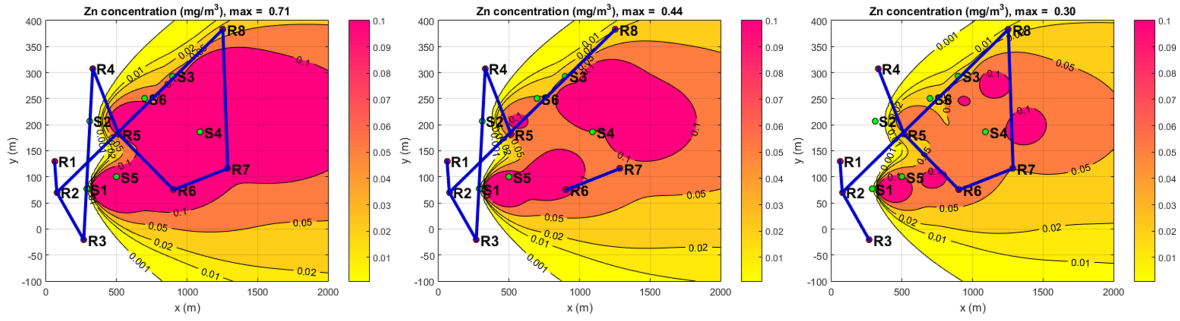


Figure 2.1: A motivating example: In an atmospheric dispersion scenario there exists 6 pollutant sources and 8 receptor distributed in the field connected to each other through a time varying graph. At first all receptors are connected and for a time interval we have two disconnected groups. The question is how to handle the consensus over estimates after reconnection.

the information accumulated in the other group during the disconnection time. As explained earlier, since the priors of the two groups become different, simple averaging is no longer applicable, and using Covariance Intersection results in too conservative estimates. The question is how to handle the consensus over estimates when agents are connected, during the disconnection time, and after reconnection.

In this work we strive to bring together the best of MHMC based distributed averaging and CI. The former is suitable for reaching consensus over uncorrelated information and the later is useful for combining estimates whose correlations are unknown or difficult to keep track of. We propose a hybrid scheme that has comparable performance to MHMC consensus while being robust to network failures. Albeit the method is explained with respect to the dynamic field estimation example, it is generally applicable to most distributed estimation scenarios.

In Section 2.3, the notation used in this chapter is explained as well as assumptions and system model. Section 2.4 discusses some preliminaries on distributed estimation which paves the way for introducing our problem objective and method. Our proposed method is presented in Section 2.4 along with its theoretical performance analysis. We extensively evaluate our method's performance in Section 2.7.

2.3 Modeling

2.3.1 State Space Modeling of the Dynamic Field

In this chapter the atmospheric dispersion problem [27] is considered as a case-study. The three-dimensional advection-diffusion equation describing the contaminant transport in the atmosphere is:

$$\frac{\partial c}{\partial t} + \nabla \cdot (cu) = \nabla \cdot (K \nabla c) + Q\delta(X - X_s), \tag{2.1}$$

where $c(x, y, z, t)$, the parameter of interest, is the mass concentration of the pollutant at location $X = (x, y, z)$. Other parameters and boundary conditions are explained in [27]. With proper discretization of the above PDE, one can define a state vector by stacking the values of the field at a given time k over all sites of the discretization lattice. The PDE model then becomes a lumped parameter, discrete-time linear (LTI) state equation of the form

$$x(k+1) = Ax(k) + Bu(k), \tag{2.2}$$

where $x(k) = [F_k(1,1,1) \cdots F_k(n,n,n)]$ and $F_k(i_x, i_y, i_z) = c(i_x\Delta_x, i_y\Delta_y, i_z\Delta_z, k\Delta t)$.

2.3.2 Stochastic Field Model

Since we consider the case where we have noise and the system is stochastic, we model the evolution of the field using the following equation which relates the state at time step k to $k+1$:

$$x(k+1) = Ax(k) + Bu(k) + w(k). \tag{2.3}$$

In the above equation $u(k) \in \mathbb{R}^m$ accounts for m input variables and the vector $w(k) \sim \mathcal{N}(0, Q(k))$ represents additive white noise used to model unknown perturbations.

2.3.3 Network Topology

Assume that we have N homogeneous agents associated with nodes of a graph. These agents can communicate with each other under a time-varying network topology $G_k = \langle \mathcal{V}_k, \mathcal{E}_k \rangle$ where $\mathcal{V}_k$ and $\mathcal{E}_k$ are the set of graph nodes and edges respectively. If $(i, j) \in \mathcal{E}_k$, it means agents i and j can communicate. The node corresponding to the

i-th agent is denoted by v_i. Neighbors of node v_i are defined as

$$\mathcal{N}^i = v_i \cup \{\forall v_j \in \mathcal{V}, (i,j) \in \mathcal{E}\}. \tag{2.4}$$

Also $|\mathcal{N}^i|$ is the cardinality of $\mathcal{N}^i$.

Each agent has a processor and a sensory package on-board. Sensors make observations every Δt seconds and processors and the network are fast enough to handle calculations based on message passing among agents every δt seconds. We assume that $\delta t \ll \Delta t$. We also assume that the agents exchange their information over the communication channel which is free of delay or error.

We assume that $x(k)$ denotes the state of the field at time-step k. Each agent retains a local version of $x(k)$ which is denoted by $x_i(k)$. For random variables we use the following notation: $\hat{x} = \mathbb{E}(x)$ and $P_x = \mathbb{E}[x - \hat{x}]^2$ are the expected value and the covariance of the random variable x respectively.

2.3.4 Observation Model

We assume that each agent has a sensor that produces noisy observations that are functions of the state of the field. The observation model of the i'th sensor is

$$z_i(k) = H_i(k)x(k) + v_i(k), \tag{2.5}$$

$$v_i(k) \sim \mathcal{N}(0, R_i(k)). \tag{2.6}$$

2.4 Distributed Filtering Preliminaries

Filtering is the process of recursively computing the posterior probability of a random dynamic process $\mathbf{x}(k)$ conditioned on a sequence of measurements $Z^k = \{z(1), z(2), \ldots, z(k)\}$, where $z(k)$ denotes the observation vector at the time-step k. Under the Gaussian assumption, the Kalman Filter (KF) is the optimal recursive

filter for linear state space systems. We denote the predicted and estimated mean and covariance at time k by $(\hat{\mathbf{x}}^-(k), P^-(k))$ and $(\hat{\mathbf{x}}(k), P(k))$.

2.4.1 Centralized Kalman Filter

The KF steps are generally formulated based on the mean and covariance matrix representation of Gaussian random variables involved; however, an alternative representation, called the information form of the KF is more useful and intuitive in the development of the decentralized filter. In this representation we define

$$\mathbf{y}(k) = P_{\mathbf{x}}^{-1}(k)\mathbf{x}(k), \tag{2.7a}$$

$$\mathbf{Y}(k) = P_{\mathbf{x}}^{-1}(k), \tag{2.7b}$$

where $\mathbf{y}(k)$ and $\mathbf{Y}(k)$ are the information vector and information matrix respectively. The prediction step of the KF can then be written as

$$M(k) = (A^{-1})^T \mathbf{Y}(k-1)A^{-1}, \tag{2.8a}$$

$$P(k) = M(k) + Q(k)^{-1}, \tag{2.8b}$$

$$\mathbf{Y}^-(k) = M(k) - M(k)P(k)^{-1}M(k), \tag{2.8c}$$

$$\mathbf{y}^-(k) = \mathbf{Y}^-(k)A\mathbf{Y}(k-1)\mathbf{y}(k-1). \tag{2.8d}$$

The information content of an observation $z_j(k)$ is $\delta i_j(k) = H_j^T(k)R_j(k)^{-1}z_j(k)$ along with the information matrix $\delta I_j(k) = H_j^T(k)R_j(k)^{-1}H_j(k)$. Assuming that information from all agents is available to a central processor, the update step of KF can be carried out by adding the information from different observations to the predicted

values.

$$\mathbf{y}(k) = \mathbf{y}^{-}(k) + \sum_{j=1}^{N} \delta i_j(k) \tag{2.9a}$$

$$\mathbf{Y}(k) = \mathbf{Y}^{-}(k) + \sum_{j=1}^{N} \delta I_j(k) \tag{2.9b}$$

This formulation is called the Centralized Information Filter (CIF).

An assumption underlying the CIF is that there is a central processor which has access to all the information available. However, when there is no central processor and each agent can only communicate with its neighbors, we want to formulate a decentralized version of the information filter. When run by all agents they should converge to the centralized estimate of the field state.

2.4.2 Decentralized Estimator Designs

2.4.2.1 Consensus Based Estimator

We start with CIF procedure outlined in previous section. Looking at Eqs. 2.9a–2.9b, one can see that

$$[\delta I, \delta i](k) \triangleq N.\tfrac{1}{N} \sum_{j=1}^{N} [\delta I_j, \delta I_j](k) \triangleq N[\overline{\delta I}, \overline{\delta i}](k).$$

Now if all the agents have the same prior information and if via a distributed averaging method the agents can reach a consensus over $\overline{\delta i}(k)$ and $\overline{\delta I}(k)$, they can use Eqs. 2.9a–2.9b to get a decentralized estimate whose results asymptotically converge to the centralized estimate.

Fortunately, such a method exists. The distributed averaging method of [33] makes minimal assumptions about the network topology and only relies on local information exchange between neighboring nodes of a graph to reach a consensus over the average initial value of the nodes. The method uses an iterative linear consensus

filter based on the weights calculated from an MHMC. Throughout this chapter, to avoid confusion, we use superscript l to indicate the consensus iterations. Consider communication graph G^l. One can use the message passing protocol of the form $\mathbf{x}^{l+1} = \sum_{j=1}^{|\mathcal{N}^l|} \gamma_{ij}^l x_j^l$ to calculate the average of the values on the graph nodes in which $d_i^l = |\overline{\mathcal{N}}_i^l|$ is the degree of the node v_i, and

$$\gamma_{ij}^l = \begin{cases} \frac{1}{1+\max\{d_i^l, d_j^l\}} & \text{if } (i,j) \in \mathcal{E}^l, \\ 1 - \sum_{(i,m)\in\mathcal{E}^l} \gamma_{im} & \text{if } i = j, \\ 0 & \text{otherwise .} \end{cases} \tag{2.10}$$

Note that for each node i, γ_{ij}'s only depend on the degrees of its neighboring nodes. Also, due to the averaging property of MHMC weights, after reaching consensus, MHMC estimates converge to the centralized estimator's results. Therefore, given the ideal centralized estimate $(\hat{\mathbf{x}}^{\text{CTR}}, P_{\mathbf{x}}^{\text{CTR}})$, we have $\hat{\mathbf{x}}_i^{\text{MH}} = \hat{\mathbf{x}}^{\text{CTR}}$ and $P_{\mathbf{x}_i}^{\text{MH}} = P_{\mathbf{x}}^{\text{CTR}}$ in the limit.

In practice the priors become different as a result of network disconnection. In those cases agents have some shared information (from the time they were connected to each other) but will accumulate new information whilst disconnected from one another. Their priors will differ after reconnection so, consequently, their consensus must be handled with care.

2.4.2.2 Covariance Intersection Based Estimator

It follows from the above discussion that if the priors are not the same among the network nodes, distributed averaging alone will not produce consistent estimates. One way of handling such a scenario is using Covariance Intersection (CI) methods. We may use an iterative CI method to reach a consensus over the local estimates when

the priors differ, either owing to disconnection or termination of the consensus process over-early. In iterative CI, the goal is to fuse different estimates of a random variable without having any knowledge about the cross covariance between such estimates. Iterative CI, iteratively solves the following optimization problem and updates local estimates accordingly until it reaches consensus.

2.4.3 Iterative CI (ICI)

At initial iteration $l = 0$, for each agent, assign the local estimate, $[\mathcal{Y}_i^0, \boldsymbol{y}_i^0]$, to be

$$\mathcal{Y}_i^0 \triangleq \mathbf{Y}_i(t_0) + \delta I_i(t_0), \quad \boldsymbol{y}_i^0 \triangleq \mathbf{y}_i(t_0) + \delta i_i(t_0).$$

Then for each iteration afterward solve for w^* such that

$$\begin{aligned} \omega^* &= \underset{\omega}{\operatorname{argmin}} \mathcal{J}\big([\textstyle\sum_{j\in\mathcal{N}_i^l} \omega_j \mathcal{Y}_j^l]^{-1}\big), \\ s.t. \quad & \textstyle\sum_{j=1}^{|\mathcal{N}_i^l|} \omega_j = 1, \quad \forall j \quad \omega_j \geq 0, \end{aligned} \tag{2.11}$$

where $\mathcal{J}(\cdot)$ is an optimization objective function; we consider $trace(\cdot)$ or $\log\det(\cdot)$. Local estimates are then updated for the next iteration

$$[\mathcal{Y}_i^{l+1}, \boldsymbol{y}_i^{l+1}] = \textstyle\sum_{j\in\mathcal{N}_i^l} \omega_j^* [\mathcal{Y}_j^l, \boldsymbol{y}_j^l]. \tag{2.12}$$

As discussed in [14], CI and consequently iterative CI (ICI) generate conservative estimates which means that $\mathbb{E}[\mathbf{x} - \hat{\mathbf{x}}_i^{\text{ICI}}] = \mathbb{E}[\mathbf{x} - \hat{\mathbf{x}}^{\text{CTR}}] = 0$ and $P_{\mathbf{x}_i}^{\text{ICI}} \geq P_{\mathbf{x}}^{\text{CTR}}$ for the local estimates and the consensus value. The disadvantage of CI is that it generates overly conservative estimates by continually neglecting the cross correlation information.

2.4.4 Problem Objective

Our goal is to design a network agnostic recursive decentralized estimator to calculate the local estimate $\mathbf{x}_i^{\text{HYB}}$ along with an associated covariance $P_{\mathbf{x}_i}^{\text{HYB}}$such that the following properties hold:

$$\begin{aligned} &\mathbb{E}[\mathbf{x} - \hat{\mathbf{x}}_i^{\text{ICI}}] = \mathbb{E}[\mathbf{x} - \hat{\mathbf{x}}_i^{\text{HYB}}] = \mathbb{E}[\mathbf{x} - \hat{\mathbf{x}}^{\text{CTR}}] = 0, \\ &\mathcal{J}(P_{\mathbf{x}}^{\text{CTR}}) \leq \mathcal{J}(P_{\mathbf{x}_i}^{\text{HYB}}) \leq \mathcal{J}(P_{\mathbf{x}_i}^{\text{ICI}}), \end{aligned} \tag{2.13}$$

i.e., we are looking for an unbiased estimate whose covariance is less than that of CI.

2.5 Hybrid CI Consensus

We propose a hybrid approach that uses ICI to reach consensus over priors and the MHMC based consensus filter for distributed averaging of local information updates. Our method is summarized in Algorithm 1. We explain the flow of the proposed method using a simple scenario with two agents. Generalization to more than two agents is straightforward and follows similar steps.

Imagine a scenario consisting of two agents, observing a dynamic field with state vector $\mathbf{x}$, that are communicating with each other through a time-varying network topology. At time t_0, the agents start with priors $[\mathbf{y}_1^-(t_0), \mathbf{Y}_1^-(t_0)]$ and $[\mathbf{y}_2^-(t_0), \mathbf{Y}_2^-(t_0)]$ respectively.

At time t_1 the field evolves to the new state $\mathbf{x}(t_1)$ and agents calculate their own local prediction (line 1 in the algorithm). Then they make observations $z_1(t_1)$ and $z_2(t_1)$, respectively, and compute the local information updates $[\delta i_1(t_1), \delta I_1(t_1)]$ and $[\delta i_2(t_1), \delta I_2(t_1)]$ (lines 2 and 3 of the algorithm).

Algorithm 1: Hybrid Method

Input : $[\mathbf{y}_j(t_0), \mathbf{Y}_j(t_0)]$

Use Eqs. 2.8c – 2.8d to calculate predicted values $[\mathbf{y}_j^-(t_1), \mathbf{Y}_j^-(t_1)]$ given $[\mathbf{y}_j(t_0), \mathbf{Y}_j(t_0)]$

Collect local observation $z_j(t_1)$ and calculate jacobian and noise covariance $[H_j(t_1), R_j(t_1)]$

Calculate the local information update

$$\delta i_j(t_1) = H_j^T(t_1) R_j^{-1}(t_1) z_j(t_1)$$
$$\delta I_j(t_1) = H_j^T(t_1) R_j^{-1}(t_1) H_j(t_1)$$

Initialize consensus variables ($l = 0$)

$$[y_j^0, \mathcal{Y}_j^0] = [\mathbf{y}_j^-, Y_j^-](t_1) \quad [\overline{\delta i}_j^0, \overline{\delta I}_j^0] = [\delta i_j, \delta I_j](t_1)$$

while *NOT CONVERGED* **do**

  BROADCAST$[y_j^l, \mathcal{Y}_j^l, \overline{\delta i}_j^l, \overline{\delta I}_j^l]$

  RECEIVE$[y_k^l, \mathcal{Y}_k^l, \overline{\delta i}_k^l, \overline{\delta I}_k^l] \quad \forall k \in \overline{\mathcal{N}}_j^l$

  Collect received data

$$\mathcal{C}_j^l = \{y_{k\in\mathcal{N}_j^l}^l, \mathcal{Y}_{k\in\mathcal{N}_j^l}^l\} \quad \mathcal{M}_j^l = \{\overline{\delta i}_{k\in\mathcal{N}_j^l}^l, \overline{\delta I}_{k\in\mathcal{N}_j^l}^l\}$$

  Do one iteration of CI on consensus variables for local prior information $\mathcal{C}_j^l$

$$[y_j^{l+1}, \mathcal{Y}_j^{l+1}] = \texttt{CI}(\mathcal{C}_j^l)$$

  Do one iteration of MHMC on consensus variables for new information $\mathcal{C}_j^l$

$$[\overline{\delta i}_j^{l+1}, \overline{\delta I}_j^{l+1}] = \texttt{MHMC}(\mathcal{M}_j^l)$$

  $l \leftarrow l + 1$

Calculate the posteriors according to:

$$\mathbf{Y}_j(t_1) = \mathcal{Y}_j^l + n_{\text{CG}} \overline{\delta I}_j^l$$
$$\mathbf{y}_j(t_1) = y_j^l + n_{\text{CG}} \overline{\delta i}_j^l$$

return $[\mathbf{Y}_j(t_1), \mathbf{y}_j(t_1)]$

The two agents, if performing ICI, would find a fused estimate

$$Y^{\text{ICI}} = w^{\text{ICI}}(\mathbf{Y}_1^- + \delta I_1) + (1 - w^{\text{ICI}})(\mathbf{Y}_2^- + \delta I_2),$$

where w^{ICI} is obtained from solving the optimization problem in Eq. 2.11. Note that doing MHMC alone is not possible here since $\mathbf{Y}_1^-$ and $\mathbf{Y}_2^-$ are different. In our hybrid method we do the following:

$$Y^{\text{HYB}} = \underbrace{w^{\text{HYB}}\mathbf{Y}_1^- + (1 - w^{\text{HYB}})\mathbf{Y}_2^-}_{\substack{\text{CI to reach} \\ \text{consensus over priors}}} + \underbrace{\delta I_1 + \delta I_2}_{\substack{\text{consensus over} \\ \text{the incremental} \\ \text{information}}} .$$

It can be seen that $\delta I_1 + \delta I_2 \geq w^{\text{CI}}\delta I_1 + (1 - w^{\text{CI}})\delta I_2$ and $\mathcal{J}(w^{\text{HYB}}\mathbf{Y}_1^- + (1 - w^{\text{HYB}})\mathbf{Y}_2^-) \geq \mathcal{J}(w^{\text{CI}}\mathbf{Y}_1^- + (1 - w^{\text{CI}})\mathbf{Y}_2^-)$ due to the fact that the optimization problem for $\mathbf{Y}_2^-$ and $\mathbf{Y}_2^-$ has the optima w^{HYB}. If $\mathcal{J}(\cdot)$ has the property that if $\mathcal{J}(\mathcal{Y}_1) \geq \mathcal{J}(\mathcal{Y}_2)$ and $\mathcal{I}1 \geq \mathcal{I}_2$ then $\mathcal{J}(\mathcal{Y}_1 + \mathcal{I}_1) \geq \mathcal{J}(\mathcal{Y}_2 + \mathcal{I}_2)$, then our method is guaranteed to outperform CI.

For an N-agent system with the i'th agent having prior Y_i^-, the ICI approach is used to find a consensus over the priors using Eq. 2.11 recursively. The MHMC approach is used to form the consensus over the new information, i.e., $\sum_{j=1}^{N} \delta I_j$ (Eq. 3.10). In line 12 of the algorithm, n_{CG} is the number of agents that form a connected group, and it can be determined by assigning unique IDs to the agents and passing these IDs along with the consensus variables. Each agent keeps track of unique IDs it receives and passes them to its neighbors. The following propositions hold.

2.6 Analysis

2.6.1 Proof of Convergence

Proposition 1. *If the objective function $\mathcal{J}(\cdot)$ in Eqn. 2.11 is strictly convex, the ICI process is guaranteed to reach a consensus over the priors, i.e., $\exists \mathcal{Y}_\star$, such that $\forall i$*

$\lim_{l\to\infty} \mathcal{Y}_i^l = \mathcal{Y}_\star$. *The same result holds for the information vector as well.*

Proof. At each iteration 'l' and for each agent 'j', ICI solves an instance of the optimization problem in Eq. 2.11. Local variables $\mathcal{Y}_i(l), \forall i \in 1, \cdots, N$ are then updated according to

$$\mathcal{Y}_i(l+1) = \sum_{j\in\mathcal{N}^i(l)} \omega_j^* \mathcal{Y}_j(l). \tag{2.14}$$

Performing ICI is equivalent to a mapping $\mathcal{F}$ that maps the set of local information matrices at step l to a new set of information matrices at step $l+1$. Defining $\mathcal{I}(l) = \{\mathcal{Y}_1(l), \cdots, \mathcal{Y}_n(l)\}$, we can write

$$\mathcal{I}(l+1) = \mathcal{F}\big(\mathcal{I}(l)\big). \tag{2.15}$$

The very definition of the optimization problem in Eq. 2.11 requires that[2]

$$\mathcal{J}(\mathcal{Y}_i^{-1}(l+1)) \leq \mathcal{J}(\mathcal{Y}_j^{-1}(l)) \quad \forall j \in \mathcal{N}^i(l) \tag{2.16}$$

Lets define $V(\mathcal{Y}_i, l) = \mathcal{J}(\mathcal{Y}_i^{-1}(l))$. Take the Lyapunov function of the whole network at iteration l to be

$$\mathcal{V}(\mathcal{I}(l)) = \sum_{i=1}^{N} V(\mathcal{Y}_i, l). \tag{2.17}$$

If $\mathcal{J}(X)$ is a positive function over the set of $\{X \in \mathbb{S}_{++}^n \triangleq$ Symmetric Positive Definite matrices$\}$, then $\forall l, \quad \mathcal{V}(\mathcal{I}(l)) > 0$.

Now define the set $\Omega = \{\mathcal{I} | \quad \mathcal{V}(\mathcal{I}) = \mathcal{V}\big(\mathcal{F}(\mathcal{I})\big)\}$. Due to the strict convexity of $\mathcal{J}$, $\mathcal{V}$ is also strictly convex and $\forall \mathcal{I} \in \Omega$, all $\mathcal{Y}_i$'s should be equal; otherwise, $\mathcal{V}(\mathcal{I}(l+1)) < \mathcal{V}(\mathcal{I}(l)$ due to the strictly convex property of $\mathcal{J}$ which results in a contradiction. This proves that $\mathcal{V}(\mathcal{I})$ is a decreasing function unless all the local

[2]Can be easily proved by contradiction.

information matrices are equal.

Since $\mathcal{V}(\mathcal{I})$ is decreasing, and it is a positive function, it has a lower bound $\mathcal{V}_L > 0$. When the network reaches this lower bound, the value of $\mathcal{V}(\mathcal{I})$ does not change and $\mathcal{I} \in \Omega$. According to the above discussion, all the local information matrices should be equal then.

Therefore, by performing ICI, the Lyapunov function of the network is guaranteed to reach a lower bound in which all $\mathcal{Y}_i$'s are equal. We conclude that if there exists a $\mathcal{Y}_\infty$, then the network is guaranteed to converge to it. □

Strict convexity of $\mathcal{J} \triangleq \operatorname{trace} Y(w)^{-1}$ in w is straightforward to show. Next proof shows that $\log\det Y(w)^{-1}$ is also strictly convex in w. By establishing strict convexity, the convergence of ICI process is guaranteed by proposition 1.

Proof. For the objective function $\mathcal{J}(w) = \log\det Y(w)^{-1}$, the gradient vector with respect to the elements of w is

$$g_i(w) = \frac{\partial \log\det Y(w)^{-1}}{\partial w_i} = -\operatorname{trace}(Y(w)^{-1}Y_i) \tag{2.18}$$

$$= -\operatorname{trace}(Y(w)^{-1/2}Y_iY(w)^{-1/2}) \leq 0 \tag{2.19}$$

for $i = 1, \cdots n$.

Similar to the above calculation, for the Hessian matrix we have

$$H_{ij}(w) = \frac{\partial^2 \log\det Y(w)^{-1}}{\partial w_i \partial w_j} = -\operatorname{trace}(Y(w)^{-1}Y_i) \tag{2.20}$$

$$= -\operatorname{trace}(Y(w)^{-1}Y_i Y(w)^{-1}Y_j) \tag{2.21}$$

$$= -\operatorname{trace}((Y(w)^{-1/2}Y_i Y(w)^{-1/2})$$

$$(Y(w)^{-1/2}Y_j Y(w)^{-1/2})) \tag{2.22}$$

for $i, j = 1, \cdots n$. We can verify that $H_{ik}(w)$ is strictly convex for $y \in \mathbb{R}^n$.

$$y^T H(w) y$$

$$= \sum_{i,j=1}^{n} -y_i y_j \operatorname{trace}(Y(w)^{-1/2}Y_i Y(w)^{-1/2})$$

$$(Y(w)^{-1/2}Y_j Y(w)^{-1/2})) \tag{2.23}$$

$$= \operatorname{trace}\Big(Y(w)^{-1/2}\big(\sum\nolimits_{i=1}^{n} y_i Y_i\big)Y(w)^{-1/2}\Big)^2 \tag{2.24}$$

$$= \Big\|\Big(Y(w)^{-1/2}\big(\sum\nolimits_{i=1}^{n} y_i Y_i\big)Y(w)^{-1/2}\Big)\Big\|_F^2 \geq 0 \tag{2.25}$$

which establishes that $\log\det Y(w)^{-1}$ is convex in w. From Eq. 2.23 one can see that $y^T H(w) y = 0$ only if $\sum_{i=1}^{n} y_i Y_i$ which will not happen for $y \neq 0$ due the independence of $Y_1, \cdots, Y_n$. Therefore, $\log\det Y(w)^{-1}$ is strictly convex in w. This guarantees that there is a unique solution w^* for the CI problem with the assumptions in proposition 1.

A counter example for Hybrid method

In order to show the superiority of the Hybrid method to CI, one should be able to show that

$$\begin{aligned} &\det\big[\sum_{j\in\mathcal{N}^i}\omega_j^{\text{ICI}}Y_j+\sum_{j\in\mathcal{N}^i}\omega_j^{\text{ICI}}\delta I_j\big]\\ &\leq\det\big[\sum_{j\in\mathcal{N}^i}\omega_j^{\text{HYB}}Y_j+\sum_{j\in\mathcal{N}^i}\delta I_j\big] \end{aligned} \tag{2.26}$$

We know that $\sum_{j\in\mathcal{N}^i}\delta I_j \geq \sum_{j\in\mathcal{N}^i}\omega_j^{\text{ICI}}\delta I_j$ and $\det\left[\sum_{j\in\mathcal{N}^i}\omega_j^{\text{HYB}}Y_j\right] \geq \det\left[\sum_{j\in\mathcal{N}^i}\omega_j^{\text{ICI}}Y_j\right]$. However, as the following example shows, for PSD matrices A, B, C, and D, $A \geq C$ and $det[B] \geq det[D]$ does not guarantee that $det[A+B] \geq det[C+D]$. This can be seen by assuming the following assignments.

$$\begin{aligned} A=\begin{bmatrix}2&0\\0&2\end{bmatrix}, B=\begin{bmatrix}1&0\\0&1\end{bmatrix},\\ C=\begin{bmatrix}1&0\\0&1\end{bmatrix}, D=\begin{bmatrix}10&0\\0&0.01\end{bmatrix} \end{aligned} \tag{2.27}$$

With this assignments

$$det(A+B)=9 \leq det(C+D)=11.11$$

2.6.2 Performance Analysis

With the counter example in the previous section, one cannot show that in general for $\mathcal{J}(\cdot)=\log\det(\cdot)$, $\mathcal{J}(P_{x_i}^{\text{HYB}}) \leq \mathcal{J}(P_{x_i}^{\text{ICI}})$ In the following we show that if hybrid method uses weights calculated by ICI, it outperforms ICI. Also, if the objective is to

reduce the trace , i.e, $\mathcal{J}(\cdot) = \text{trace}(\cdot)$, then hybrid method always outperforms ICI.

Proposition 2. *For the distributed estimation problem with the network topology* $G = \langle \mathcal{V}, \mathcal{E} \rangle$*, and the objective function* $\mathcal{J}(\cdot) = \log\det(\cdot)$*, if at the beginning, the estimate priors satisfy* $\forall i, Y_i^{\text{HYB}}(0) \geq Y_i^{\text{ICI}}(0)$ *and if at each iteration of the consensus process ICI weights are used to fuse priors in the Hybrid method, then, for all agents* $i = 1, \cdots, N$ *and for all iterations* $l = 1, \cdots, L$

$$Y_i^{\text{HYB}}(l) \geq Y_i^{\text{ICI}}(l), \tag{2.28}$$

and after convergence

$$P_i^{\text{HYB}}(+\infty) \leq P_i^{\text{ICI}}(+\infty). \tag{2.29}$$

Proof. For $l = 0$, the inequality in Eq. 2.28 holds by default. Now suppose that it holds at step l for agent $i, \forall i \in [1, \cdots, N]$. Then, since $\forall i, Y_i^{\text{HYB}}(l) \geq Y_i^{\text{ICI}}(l)$,

$$\sum_{j \in \mathcal{N}^i} \omega_j^{\text{ICI}}(l) Y_j^{\text{HYB}}(l) \geq \sum_{j \in \mathcal{N}^i} \omega_j^{\text{ICI}}(l) Y_j^{\text{ICI}}(l)$$

and therefore, $\forall i, Y_i^{\text{HYB}}(l+1) \geq Y_i^{\text{ICI}}(l+1)$. Invoking the method of prove by induction, the first claim of the proposition is proved. The second claim is obtained by invoking the first claim as $l \to \infty$. □

2.6.3 Properties of ICI Weights

For the second part, note that ICI iteration for each agent i starts by making a convex combination of $\mathcal{Y}_i(0) \triangleq Y_i^{\text{ICI}}(0) + \delta I_i(0), \forall i \in \mathcal{N}^i$. In the second iteration,

$$\mathcal{Y}_{j_2}^{\text{ICI}}(1) = \sum_{j_1 \in \mathcal{N}^{j_2}} \omega_{j_1}^{\text{ICI}}(1) \mathcal{Y}_{j_1}^{\text{ICI}}(1) \tag{2.30}$$

$$= \sum_{j_1 \in \mathcal{N}^{j_2}} \omega_{j_1}^{\text{ICI}}(1) \Big[\sum_{j_0 \in \mathcal{N}^{j_1}} \omega_{j_0}^{\text{ICI}}(0) \Big[Y_{j_0}^{\text{ICI}}(0) + \delta I_{j_0}(0) \Big] \Big] \tag{2.31}$$

$$= \sum_{j_1 \in \mathcal{N}^{j_2}} \sum_{j_0 \in \mathcal{N}^{j_1}} \omega_{j_1}^{\text{ICI}}(1) \omega_{j_0}^{\text{ICI}}(0) Y_{j_0}^{\text{ICI}}(0) + \sum_{j_1 \in \mathcal{N}^{j_2}} \sum_{j_0 \in \mathcal{N}^{j_1}} \omega_{j_1}^{\text{ICI}}(1) \omega_{j_0}^{\text{ICI}}(0) \delta I_{j_0}(0). \tag{2.32}$$

One can rewrite the ICI iterations as multiplication of time varying stochastic matrices by the results from the previous iteration. The multiplication of two stochastic matrices is also a stochastic matrix. Therefore, by dropping the ICI superscript for better clarity, the following can be said about $\{m, n\}'th$ element of $\mathcal{Y}_i$'s:

$$\mathcal{Y}^{\{m,n\}}(l+1) = \begin{bmatrix} \mathcal{Y}_1^{\{m,n\}}(l+1) \\ \mathcal{Y}_2^{\{m,n\}}(l+1) \\ \vdots \\ \mathcal{Y}_N^{\{m,n\}}(l+1) \end{bmatrix}$$

$$= \begin{bmatrix} w_{1,1}(l) & \cdots & w_{1,N}(l) \\ w_{2,1}(l) & \cdots & w_{2,N}(l) \\ \vdots & \cdots & \vdots \\ w_{N,1}(l) & \cdots & w_{N,N}(l) \end{bmatrix} \begin{bmatrix} \mathcal{Y}_1^{\{m,n\}}(l) \\ \mathcal{Y}_2^{\{m,n\}}(l) \\ \vdots \\ \mathcal{Y}_N^{\{m,n\}}(l) \end{bmatrix}, \tag{2.33}$$

in which $w_{i,j}(l) \triangleq 0$ if $\{i,j\} \notin \mathcal{E}$ and for the rest of the elements in $i'th$ row where $\{i,j\} \in \mathcal{E}$ at least one of them is non-zero and the non-zero elements always add up to one. In a more compressed way,

$$\mathcal{Y}^{\{m,n\}}(l+1) = W^G(l)\mathcal{Y}^{\{m,n\}}(l), \tag{2.34}$$

in which $W^G(l)$ is the graph topology dependent weight matrix for ICI at iteration l. $W^G(l)$ is a stochastic matrix and since the multiplication of two stochastic matrices is also a stochastic matrix, the ICI process is equivalent to performing a convex combination of priors and local information matrices.

$$\mathcal{Y}^{\{m,n\}}(\infty) = \lim_{l\to\infty} \prod_{l=1}^{\infty} W^G(l)\mathcal{Y}^{\{m,n\}}(0). \tag{2.35}$$

We have shown that under ICI, all estimates converge to a unique matrix and given that the ICI is equivalent to a convex combination of initial values over all the nodes, it can be concluded that

1. The matrix $W^G(\infty) = \prod_{l=1}^{\infty} W^G(l)$ is a stochastic matrix and has an eigen value of 1

2. The corresponding eigen vector for eigen value 1 is a vector of all ones.

3. The ICI estimate is a convex combination of priors and additional information over all the network nodes, i.e., $\exists w = (w_1, \cdots, \omega_n) \in \mathbb{R}^N$, where $\forall i, 0 \leq \omega_i \leq 1$, $\sum_{i=1}^{N} \omega_i = 1$ and

$$\mathcal{Y}^{\text{ICI}}(+\infty) = \sum_{j=1}^{N} \omega_j \mathcal{Y}_j^{\text{ICI}}(0) + \sum_{j=1}^{N} \omega_j \delta I_j(0). \tag{2.36}$$

Now given the assumption that hybrid method uses the weights of ICI and given that

$$\sum_{j=1}^{N}\omega_j\delta I_j(0) \leq \sum_{j=1}^{N}\delta I_j(0), \tag{2.37}$$

it can be seen that

$$\begin{aligned}&\big[\sum_{j=1}^{N}\omega_j\delta Y_j^{\text{HYB}}(0) + \sum_{j=1}^{N}\delta I_j(0)\big]^{-1}\\ \leq\ &\big[\sum_{j=1}^{N}\omega_j\delta Y_j^{\text{ICI}}(0) + \sum_{j=1}^{N}\omega_j\delta I_j(0)\big]^{-1}.\end{aligned} \tag{2.38}$$

Therefore, with the assumptions made in the proposition, for the converged value of covariance

$$P^{\text{HYB}}(+\infty) \leq P^{\text{ICI}}(+\infty).$$

Proposition 3. *For the distributed estimation problem with the network topology* $G = \langle \mathcal{V}, \mathcal{E}\rangle$*, and the objective function* $\mathcal{J}(\cdot) = \text{trace}(\cdot)$*, if at the beginning, the estimate priors satisfy* $\forall i, Y_i^{\text{HYB}}(0) \geq Y_i^{\text{ICI}}(0)$*, then, for all agents* $i = 1, \cdots, N$ *and for all iterations* $l = 1, \cdots, L$

$$\text{trace}(Y_i^{\text{HYB}}(l)) \geq \text{trace}(Y_i^{\text{ICI}}(l)), \tag{2.39}$$

and after convergence

$$\text{trace}(P_i^{\text{HYB}}(+\infty)) \leq \text{trace}(P_i^{\text{ICI}}(+\infty)). \tag{2.40}$$

Proof. For $l = 0$, the inequality in Eq. 2.39 holds by default. Now suppose that it holds at step l for agent $i, \forall i \in [1, \cdots, N]$. Then, since $\forall i, \text{trace}(Y_i^{\text{HYB}}(l)) \geq \text{trace}(Y_i^{\text{ICI}}(l))$,

and

$$\begin{aligned}&\text{trace}\Big(\sum_{j\in\mathcal{N}^i}\omega_j^{\text{HYB}}(l)Y_j^{\text{HYB}}(l)\Big)\\&\geq \text{trace}\Big(\sum_{j\in\mathcal{N}^i}\omega_j^{\text{ICI}}(l)Y_j^{\text{ICI}}(l)\Big)\end{aligned}\tag{2.41}$$

should hold. If it does not, then

$$\begin{aligned}&\text{trace}\Big(\sum_{j\in\mathcal{N}^i}\omega_j^{\text{HYB}}(l)Y_j^{\text{HYB}}(l)\Big)\\&< \text{trace}\Big(\sum_{j\in\mathcal{N}^i}\omega_j^{\text{ICI}}(l)Y_j^{\text{ICI}}(l)\Big),\end{aligned}\tag{2.42}$$

and one can combine use ICI weights and get a strictly larger trace value compared to the combination with $\omega_j^{\text{HYB}}(l)$'s. This contradicts with the fact that $\omega_j^{\text{HYB}}(l)$'s minimize the trace of the convex combination of $Y_j^{\text{HYB}}(l)$'s.

Using the method of proof by induction, the above discussion shows that the inequality in 2.39 holds for all consensus iterations. Once this is proved, for converged estimates we have

$$\begin{aligned}&\text{trace}\Big(\sum_{j=1}^{N}\omega_j^{\text{HYB}}Y_j^{\text{HYB}}(0)\Big)\\&\geq \text{trace}\Big(\sum_{j=1}^{N}\omega_j^{\text{ICI}}Y_j^{\text{ICI}}(0)\Big)\end{aligned}\tag{2.43}$$

and

$$\begin{aligned}&\text{trace}\left(\sum_{j=1}^{N}\omega_j^{\text{HYB}}Y_j^{\text{HYB}}(0)+\sum_{j=1}^{N}I_j(0)\right)\\ \geq\ &\text{trace}\left(\sum_{j=1}^{N}\omega_j^{\text{ICI}}Y_j^{\text{ICI}}(0)+\sum_{j=1}^{N}\omega_j^{\text{ICI}}\delta I_j(0)\right)\end{aligned}\tag{2.44}$$

which proves that

$$\text{trace}(P_i^{\text{HYB}}(+\infty))\leq\text{trace}(P_i^{\text{ICI}}(+\infty)).\tag{2.45}$$

□

2.6.4 Properties of CI as an Operator

Covariance intersection as a function accepts more than two inputs. However, investigating the mathematical properties of it as a binary operator gives us insight into relationship between distributed CI and centepsralized CI. The former iteratively updates estimates in a network through successive local CIs till it converges. The latter performs the optimization on all the estimates in the network at once. Important question is if these two converge to the same value. In our experiments we found that most of the time both methods converge to the same value. However there are cases where that is not the case. The example in Fig. 2.2 illustrates one such case where the order of doing covariance intersection affects the final outcome. As it can be seen, among the three possible combinations considered, only $\mathcal{CI}(\mathcal{CI}(B,C),A)$ generates the same result as the global CI. The scenario that generates this example is as follows. A network with three nodes i, j and, k starts with a topology in which node i is isolated from the two other at first and j and k are connected to each other. Initial covariances are C_{i0}, C_{j0} and, C_{k0}. After doing local CI, the covariance of node i does not change while the covariance of nodes j and k become $\mathcal{CI}(C_{j0},C_{k0})$. In second step all nodes

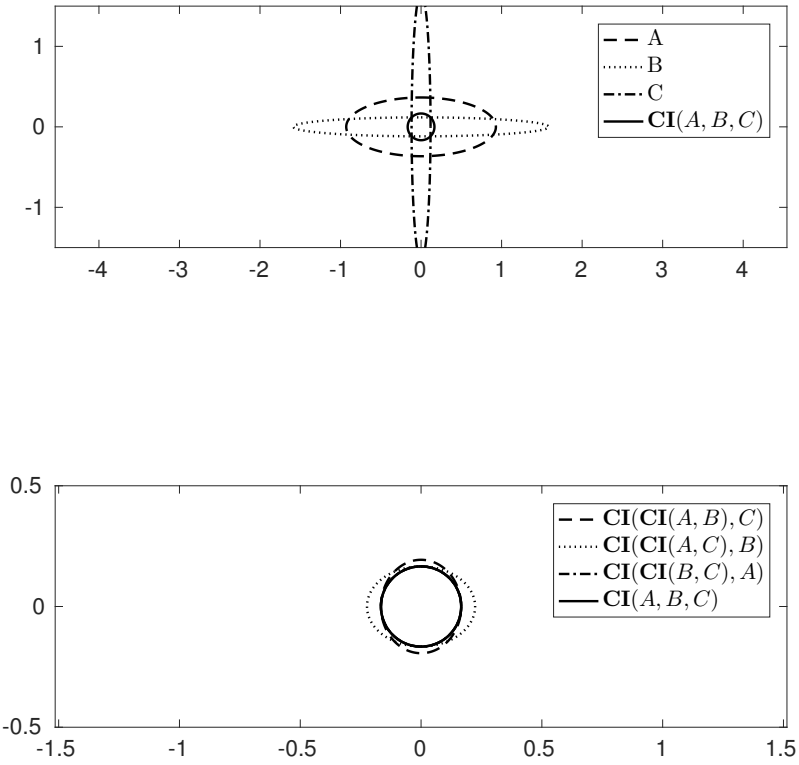


Figure 2.2: The result of Iterative CI procedure does not always converge to the solution of global CI.

get connected and they all have covariance matrix $\mathcal{CI}(\mathcal{CI}(C_{j0}, C_{k0}), C_{i0})$. The example in Fig. 2.2 shows that $\mathcal{CI}(\mathcal{CI}(C_{j0}, C_{k0}), C_{i0})$ is not equal to $\mathcal{CI}(C_{i0}, C_{j0}, C_{k0})$ in two of the three cases considered there.

2.6.5 The Price of Not Knowing the Cross-Covariance

We consider an example in which the difference between knowing and not knowing the correlation between two estimates becomes noticeable. Consider the following

scenario. Two agents a and b start with the same prior information and make observations z_a and z_b in turn. They calculate local covariance estimates P_{aa} and P_{bb} respectively. Now consider three different cases

1. Local estimates are uncorrelated.
2. Local estimates are correlated but the correlation matrix $\tilde{P}_{ab}$ is unknown.
3. Local estimates are uncorrelated and the correlation matrix $\tilde{P}_{ab}$ is known.

Suppose now that they become in contact with each other and suppose that they want to fuse their information according to

$$c = K_1 a + K_2 b$$

The first two cases can be handled by finding the solution to the following problem.

$$P_{cc} = \begin{bmatrix} K_1 & K_2 \end{bmatrix} \begin{bmatrix} P_{aa} & \tilde{P}_{ab} \\ \tilde{P}_{ab}^T & P_{bb} \end{bmatrix} \begin{bmatrix} K_1^T \\ K_2^T \end{bmatrix}. \tag{2.46}$$

The optimal solution of K_1 and K_2 yields a P_{cc} in the following form:

$$P_{cc}^{-1} = \begin{bmatrix} I & I \end{bmatrix} P^{-1} \begin{bmatrix} I \\ I \end{bmatrix} \tag{2.47}$$

$$\begin{aligned} =& P_{aa}^{-1} + \left(P_{aa}^{-1} \tilde{P}_{ab} - I \right) \\ & \times \left(P_{bb} - \tilde{P}_{ab}^T P_{aa}^{-1} \tilde{P}_{ab} \right)^{-1} \left(\tilde{P}_{ab}^T P_{aa}^{-1} - I \right). \end{aligned} \tag{2.48}$$

The case of unknown correlation is handled by CI method. The three cases are compared in figures 2.3, 2.4 and 2.5. In these examples

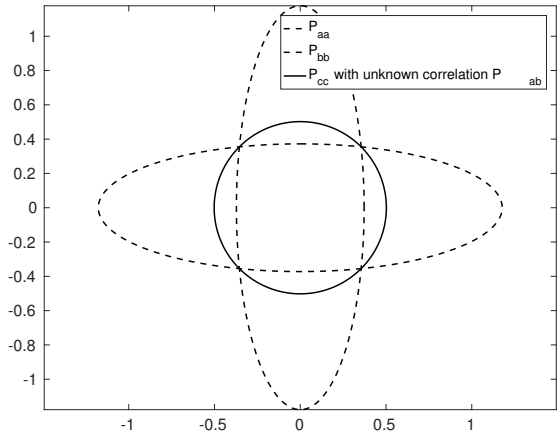


Figure 2.3: Fusion of two estimates with unknown correlation between them.

$$P_{aa} = \begin{pmatrix} 1.0 & 0 \\ 0 & 0.1 \end{pmatrix}, \quad P_{bb} = \begin{pmatrix} 0.1 & 0 \\ 0 & 1 \end{pmatrix}$$

and for the known correlation case

$$\tilde{P}_{ab} = \begin{pmatrix} -0.3162 & 0 \\ 0 & -0.3162 \end{pmatrix}.$$

As it can be seen Fig. 2.4, knowing the correlation can make a big difference in some cases. The uncertainty is considerably smaller compared to two other cases.

2.6.6 Realistic Evaluation Criteria

Comparing the performance of the distributed algorithm with centralized algorithm is insightful. However, when because of network disconnection, realization of a centralized estimator is impossible, comparing its performance with the distributed estimation algorithm is unfair. Instead, the comparison should be made with respect

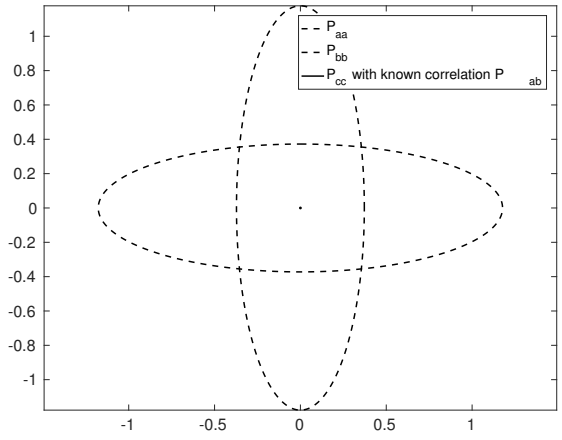


Figure 2.4: Fusion of two estimates with known correlation between them.

to the best possible estimator given the network connectivity constrains throughout time. Such an ideal estimator is called *Gold Standard* from now on. It is described as follows:

Each agent keeps track of its own observations and all the observations that it receives at each iteration from other agents connected to it. Lets denote this by $\mathcal{H}_i^t$. If the memory and communication constrains are not of concern, at each time step agents can share their history with each other and update their history according to the shared information. The update rule for $\mathcal{H}_i^t$ is as follows

$$\mathcal{H}_i^t = \bigcup_{\forall j, \mathbf{1}_{i\to j}} \mathcal{H}_j^{t-1} \bigcup_{\forall j, \mathbf{1}_{i\to j}} z_j^t \tag{2.49}$$

where $\mathbf{1}_{i\to j}$ is an indicator function which is 1 when there is a path between node i and j under current network topology. Obviously $\mathbf{1}_{i\to i}$.

In Gold Standard, at each step the best possible estimate for each agent is obtained by updating the history and then rerunning the filer from scratch. If the network

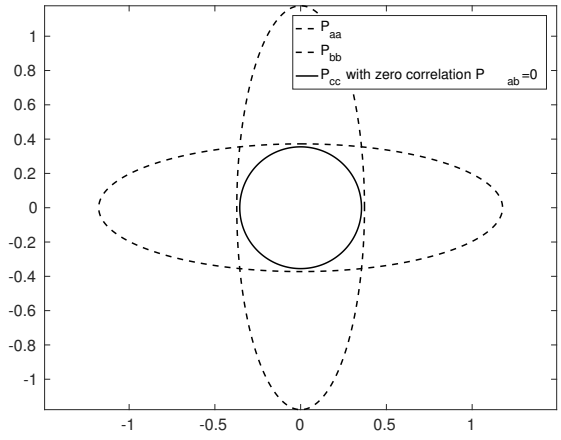


Figure 2.5: Fusion of two estimates with zero correlation between them.

remains connected, the output is equal to the centralized estimator. If the network gets disconnected, Gold Standard gives the best possible estimate.

2.6.7 Complexity Analysis

Consider the the problem of distributed estimation of a state vector of dimension n by a system consisting of M agents connected to each other through a network $G = \langle \mathcal{V}, \mathcal{E} \rangle$.

Complexity of the ICI method: The core of CI is a determinant maximization problem and according to [29] the number of iterations required to solve the optimization is of order $O(\sqrt{n} f(\epsilon))$ where ϵ is a convergence parameter. For each iteration of the optimization algorithm and for each agent i, cost (considering the objective function -log det$(\cdot)$) and gradient calculations are of orders $O(n^3 + d_i n^2)$ and $O(d_i n^2)$ respectively, where d_i is the node i's degree. Therefore, the complexity of CI optimization step is $O(\sqrt{n}(n^3 + d_i n^2))$.

Assuming T_{ICI} to be the number of iterations until ICI converges, the computational complexity requirement for each agent can be summarized as

$$T_{\text{ICI}} \times O\left(\sqrt{n}(n^3 + d_i n^2)\right). \tag{2.50}$$

ICI relies on passing messages of size $d_i(n^2+n)$ which is independent of the size ofthe network and only depends on the number of agent i's neighbors.

Complexity of the Hybrid method: The computational cost of ICI is already calculated. For the Hybrid method the cost of doing MHMC consensus should also be considered. MHMC consensus iterations update local covariance with order $O(d_i n^2)$. The convergence times of these algorithms are different in general. Assuming T_{MH} to be the number of iterations until MHMC converges, the computational complexity requirement for each agent can be summarized as

$$T_{\text{ICI}} \times O\left(\sqrt{n}(n^3 + d_i n^2)\right) + T_{MH} \times O(d_i n^2) \tag{2.51}$$

The Hybrid method relies on passing messages of size $2d_i(n^2+n)$ for exchanging information with neighbors.

Complexity of the Gold Standard Algorithm: Assuming that observations and histories that are passed around in the network have unique ID numbers and timestamps, finding the union in Eq. 2.49 for each agent takes

$$O\left(t^2(M-1)^2 + t(M-1)n^2\right)$$

for worst case scenario and $O(d_i n^2)$ at least. The best implementation of union algorithm would only look for the observations that the agent does not have at the moment. That is, it will look for timestamps and agents that are not present in the current history. Assuming that history is saved as a look up table, the worst case would be to search in the history of all agents for the N_{miss} missing

observations, which takes $O(N_{miss}(M-2))$ for the worst case scenario. This results in $O(N_{miss}(M-2)) + O(N_{obs}n^3)$ Establishing an average computational requirement for the union in Eq. 2.49 is more involved and even without considering that, for each agent the Gold Standard algorithm has to run the estimation starting from the latest time for which the whole history of all other agents is at its disposal. This will become computationally prohibitive as time goes on for a network that might get disconnected. The upper bound for computational cost is $O(tMn^3)$. Even without considering the computational cost of performing the union and the prohibitive memory size and communication requirement for passing messages, the full history estimation cost is larger than the hybrid algorithm for large t.

The memory and message passing requirement of keeping the full history also grows linearly in time which finally will make Gold Standard algorithm impossible to implement for real world applications. We will only use the Gold Standard algorithm for comparison purposes as it is the best achievable performance under the network topology constraints.

2.7 Experiments

We perform two sets of experiments on an atmospheric dispersion problem to show the effectiveness of our method and evaluate its performance during disconnection and after reconnection. This is a three dimensional problem and after proper discretizing of its Partial Differential Equation (PDE), we get a system in the format of Eq. 2.3.

For our experiments after discretization, the dimension of the state is 80. We assume that there are 10 sources emitting pollutant Zinc (referred to as Zn from now on) into atmosphere. There are also 9 receptors making noisy measurements of the concentrations of Zn around their location in space. We assume that receptors can communicate to each other through a time varying network which does not remain

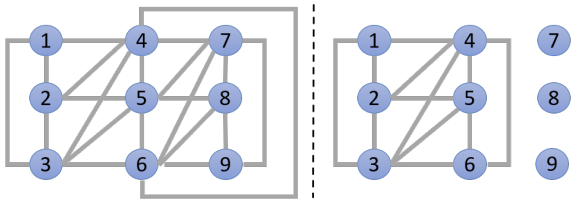


Figure 2.6: Topology of the Network when all receptors are connected (left) and when receptors 7,8 and 9 get disconnected from the rest of the group (right).

connected at all times. Receptors receive information only from their immediate neighbors. They all have access to the sources' locations and the source emission is modeled as a white noise process with known covariance.

2.7.1 The Effect of Disconnection on Estimation Performance

In this experiment we intend to evaluate the performance of the proposed method during the phase where some receptors become disconnected from the rest of the group and get connected again after some interval. The topology of the network takes one of the forms depicted in Fig. 2.6. The network starts fully connected and starting from timestep 3, receptors 7, 8 and 9 become isolated and remain in this situation for 2 steps, then they are connected back to the rest of the receptors. Similarly, disconnection happens in intervals $[17-20]$ and $[23-30]$.

In order to make a comparison we obtain the estimation result using pure CI, our method and also a centralized estimator to see how much of its performance can be recovered. Note that the MHMC consensus cannot be done here due to disconnection. The results are depicted in Fig. 2.7.

We use three measures to evaluate the estimates.

As it can be seen, the proposed method outperforms pure CI as expected and is able to get the performance very close to centralized estimator results after reconnection. Based on Bhattacharyya distance, closeness between centralized and distributed

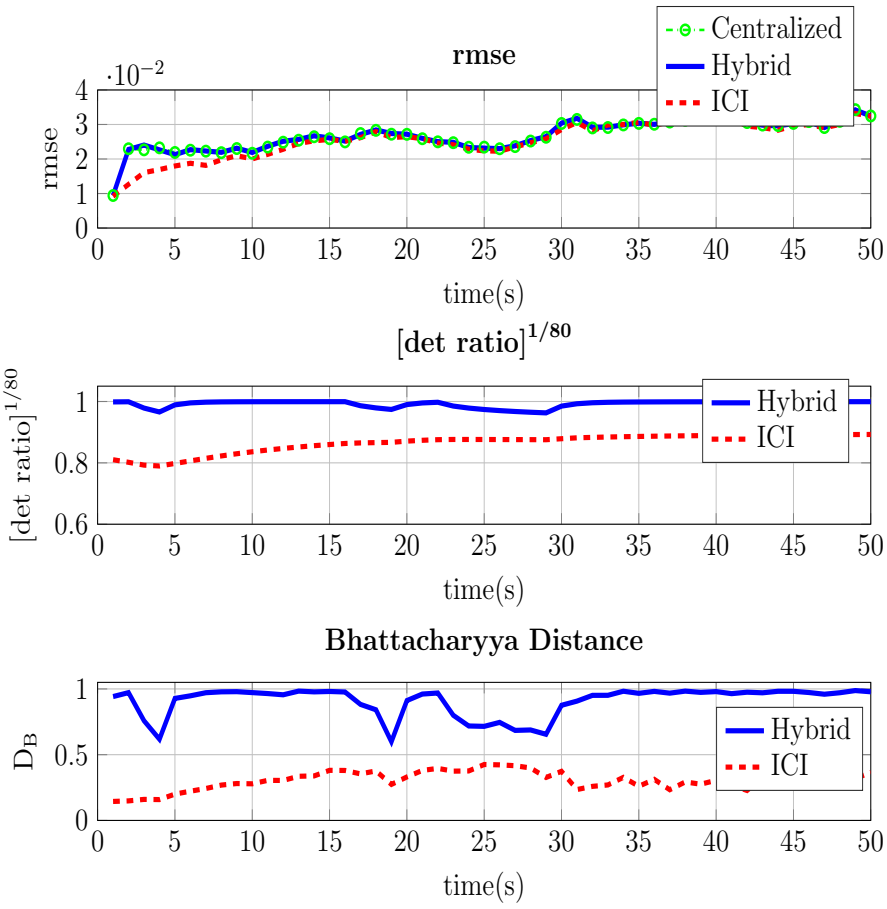


Figure 2.7: Comparison of the estimation results using centralized kalman filter, pure covariance intersection, and our method.

estimators drops during disconnection interval as expected since receptors do not have access to all the information available to the centralized estimator. While the proposed method is able to immediately recover after reconnection, pure CI continues to have lower performance even after re-connection due to the fact that it ignores the correlations.

Fig. 2.8 takes a closer look at the performance of the proposed method and compares the estimation results of receptor 5 and 8 during two different time steps. The vertical axes represent consensus steps not time. Based on Fig. 2.6, receptor

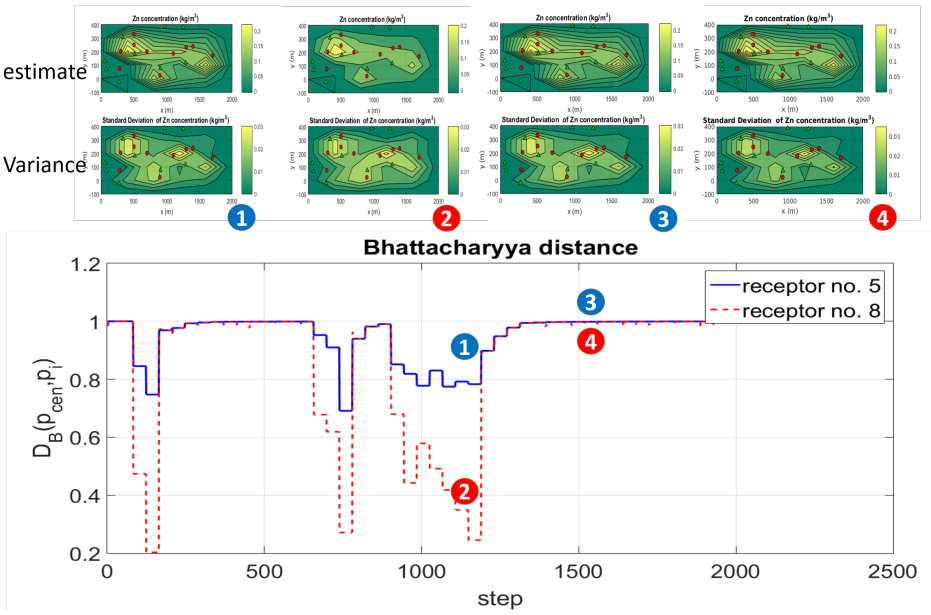


Figure 2.8: Estimation performance comparison among receptors.

5 remains in a group of size 6 during disconnection period whereas receptor 8 remains totally isolated in that period. The higher difference between centralized and distributed estimate for this receptor can be explained based on the fact that it has less information at its disposal. However, after reconnection both receptors are able to converge to the same value which is very close to the centralized estimator.

2.7.2 Performance Analysis and Robustness to Link Failure

In this experiment we evaluate the performance of our method in a systematic way to establish its usefulness and robustness to networks with high probability of link failure. We consider the same system as in the first experiment and simulate it for 50 time steps. At the beginning of each step a 4 regular graph with 9 nodes is generated, and given a probability of failure for each link, some links in the graph will randomly be disconnected. Depending on the regularity degree, and probably of failure, in some percentage of times, the graph still remains connected. However, if

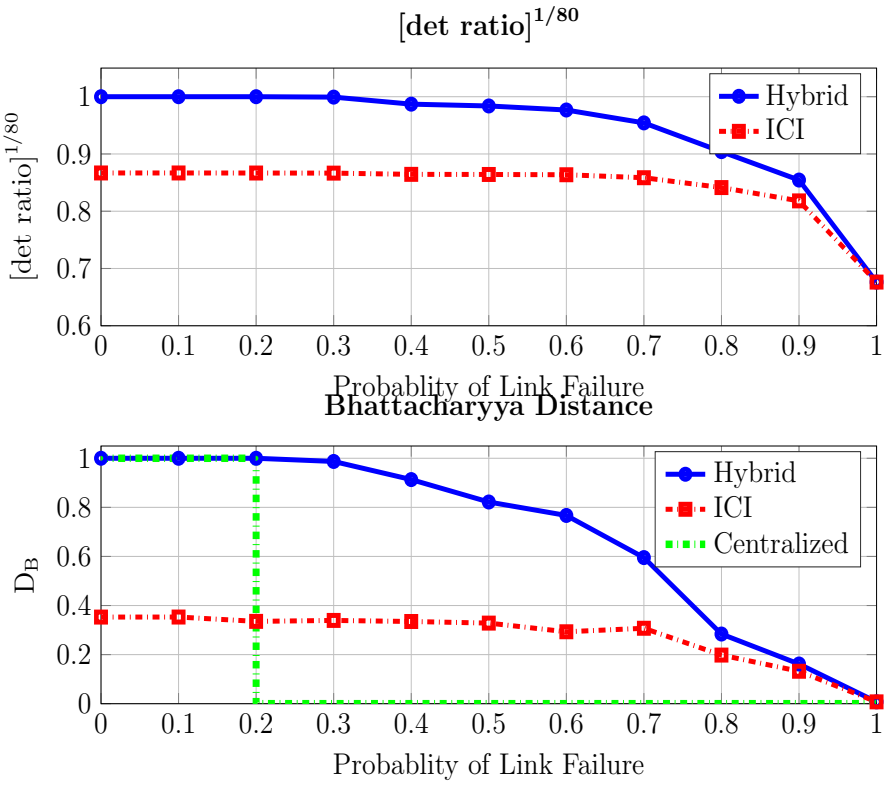


Figure 2.9: Composite diagram for performance comparison for different probablity of link failure.

the regularity degree goes down or the probability of failure increases, more often than not, the graph becomes disconnected.

In practice, for $p \geq 0.2$, consensus methods are no longer guaranteed to succeed since the network almost always get disconnected at some point in time.

We ran our method for 50 steps, as explained earlier, for each probability of link failure and compared its performance with the ideal centralized result (which is obtained by assuming full connectivity at all times). The performance is evaluated by calculating the average value for Bhattacharyya distance and determinant ratio measure at all steps and for all receptors. Based on Fig. 2.9, for the case considered in this experiment, our distributed estimator performs very similar to the ideal

centralized one for $p \in [0.0, 0.4]$ while drastically outperforming pure CI all the time. This means that in the case considered here, our method can perform almost as well as the ideal estimator for an unreliable network. Obviously, the performance can vary from one system to another and under different network topologies. However, our method can recover the performance of the centralized method when the network is unreliable and substantially outperforms pure CI always as it has already been established theoretically.

2.7.3 Comparison with Gold Standard

We performed a comparison with Gold Standard for the reduced order atmospheric example. We reduced the dimension of the system reduced from 10^5 to 40 using RPOD (A Randomized Proper Orthogonal Decomposition Technique) [35] and simulated the reduced order system for 80 steps. The performance comparison with the gold standard is shown in Fig. 2.10. The gap between the results of the Hybrid algorithm and the GS (Gold Standard) is the price of not keeping all the information. Given the unsubstantial computational requirement of the GS method, one might resort to using the Hybrid algorithm with some loss of performance.

2.7.4 Tracking Example

Figures 2.11 and 2.12 illustrate an example of running the proposed method on a tracking problem. Nine agents observe a maneuvering target.

2.7.5 Convergence Time

The convergence time of distributed averaging and ICI are compared in Fig. 2.13. The convergence time is faster for the ICI compared to MHMC distributed averaging part.

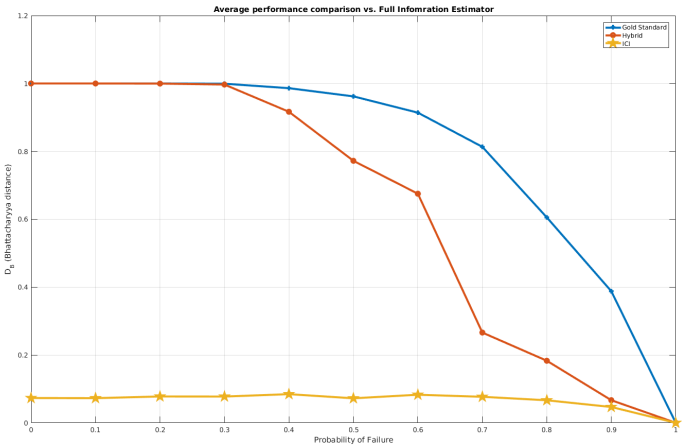


Figure 2.10: Comparison with Gold Standard

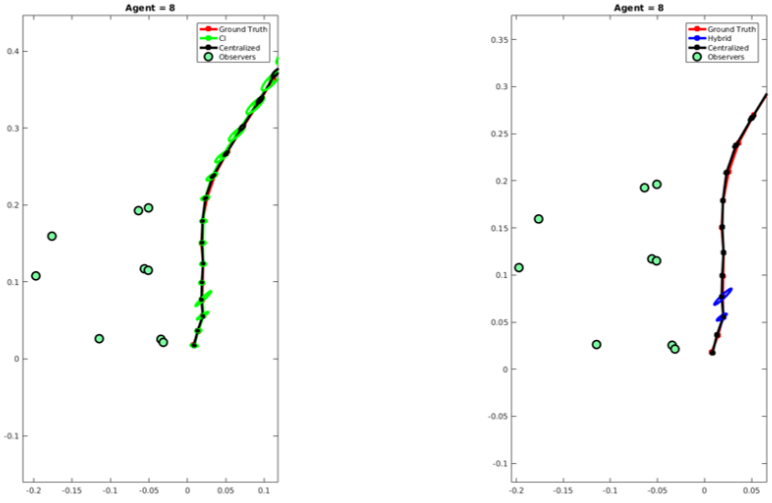


Figure 2.11: Tracking example.

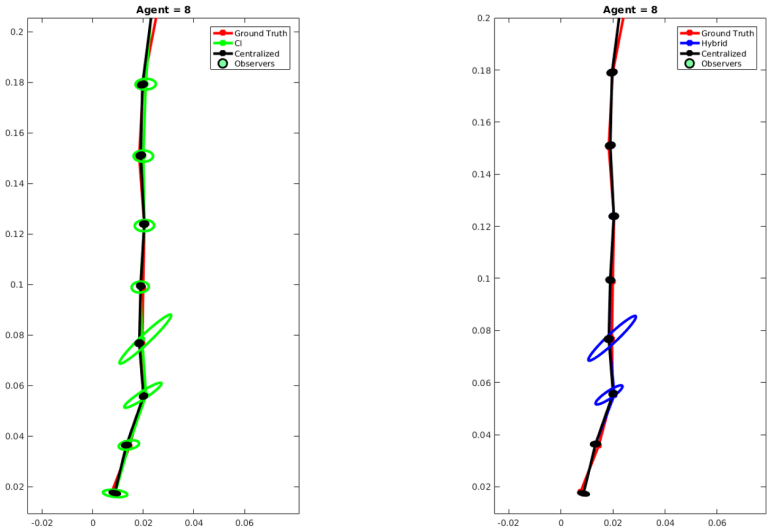


Figure 2.12: Tracking example zoomed version.

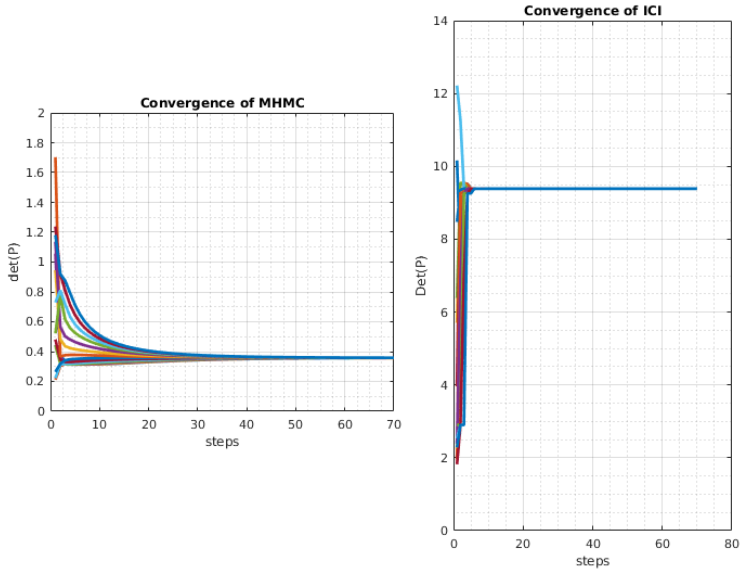


Figure 2.13: Convergence time of distributed averaging and ICI.

3. GENERALIZED HYBRID DISTRIBUTED ESTIMATION.

3.1 Related Work

Estimation on sensor networks has many applications and, thus, has been extensively studied in recent years [17, 10]. In a sensor network, nodes represent sensors that make noisy observations of the state of an underlying system of interest. The estimation is considered centralized if all the nodes send their raw observations to a central node who then calculates an estimate based on the collective information [1]. This is not always possible due to link failures as well as bandwidth and energy constraints [36]. One viable alternative is Distributed State Estimation (DSE).

In DSE, the processor on each node fuses local information with the incoming information from neighboring nodes and redistributes the fused result on the network. The objective is to design both a protocol for message passing between nodes and local fusion rules so that the nodes reach a consensus over their collective information. Although the DSE algorithms are not guaranteed to match the performance of the centralized estimator all the time, their scalability, modularity and, robustness to network failure motivates the ongoing research. These features are important for the envisioned applications of such algorithms like multi-agent localization [26] and cooperative target tracking [31].

DSE algorithms can be categorized based on the assumptions they make. Any DSE method makes assumptions about one or more of the following aspects: the state (static [34] or dynamic [26]), state transition model (linear [23] or non-linear [6]), type of noise (Gaussian [34, 23] or non-Gaussian [18]), topology of the network (constant or changing [28, 34]), connectivity of the network (always [6] or intermittent connection [28, 34]), agents' knowledge about the network topology (global or local

[28, 34, 6]) and finally the treatment of mutual information between local estimate (exact solution through bookkeeping [17] or conservative solutions that avoid double counting [30, 20]).

The research on DSE for linear systems with Gaussian noise is extensive (see [23, 13] for reviews). For nonlinear systems with Gausssian noise, the distributed versions of Extended Kalman Filters (EKF), Extended Information Filters (EIF) and Unscented Kalman Filter (UKF) have been proposed by [5, 12, 6], respectively. For nonlinear systems with non-Gaussian noise, different flavors of Distributed Particle Filter (DPF) methods were proposed by [22]. In order to avoid scalability problems and the need for synchronized random generators, DPF methods make approximations that result in loss of performance compared to a centralized PF [18].

For dynamic systems, the connectivity constraint is a determining factor for choosing the proper DSE method. If the network remains connected, DSE methods can keep the node priors the same and perform consensus only on likelihoods [19, 21]. We refer to this approach as Consensus on Likelihoods (CL). The advantage of CL is it can match the centralized estimator's performance. However, if the network becomes disconnected, priors become different and CL methods fail. For those scenarios, one approach is to perform Iterative Conservative Fusion (ICF) on node posteriors [4, 30, 20]. ICF methods are inherently sub-optimal as a result of their conservative fusion rule that avoids double counting at the expense of down weighting the uncorrelated information.

Recently, researchers have suggested combining ICF and CL methods to benefit from their complementary features [5, 6, 28]. CL can reach a consensus over uncorrelated new information and ICF can handle the correlated prior information. Such hybrid methods have been shown to outperform pure ICF's performance and remain robust to link failure [28].

In this chapter we extend the method of [28] to finite-state systems with non-Gaussian noise. We adopt Hidden Markov Model (HMM) to model the system. Unlike most methods we do not require the network to remain connected all the time. In Section 3.2, the notation used in this chapter is explained as well as assumptions and system model. Section 3.3 discusses some preliminaries on distributed state estimation, paving the way for our method. Our proposed method is presented in Section 3.4. We report on evaluation of our method's performance in Section 3.5.

3.2 Modeling

3.2.1 The Network Topology

Assume that we have n homogeneous agents associated with the nodes of a graph. These agents can communicate with each other under a time-varying undirected network topology $G_k = \langle \mathcal{V}, \mathcal{E}_k \rangle$ where $\mathcal{V}$ and $\mathcal{E}_k$ are the set of graph nodes and edges at step k respectively. The node corresponding to the i^{th} agent is denoted by v_i. If $(v_i, v_j) \in \mathcal{E}_k$, it means that agents i and j can communicate at step k. The neighbors of node v_i are defined as $\overline{\mathcal{N}}_i$; those agents connected by an edge to v_i. The set $\mathcal{N}_i = \overline{\mathcal{N}}_i \cup \{v_i\}$ will also be used in some of the equations. $|\mathcal{N}_i|$ is the cardinality of $\mathcal{N}_i$. For an arbitrary integer a, we define the index set $\boldsymbol{I_a}$ to be

$$\boldsymbol{I_a} = \{1, 2, \cdots, a\}, \tag{3.1}$$

and therefore, index sets $\boldsymbol{I_n}$, and $\boldsymbol{I_k}$, index the agents and time steps, respectively.

3.2.2 The Hidden Markov Model

Consider a finite state hidden markov model (HMM) with the following specification:

- The HMM has n_s possible states $\mathcal{X} = \{S_1, \cdots, S_{n_s}\}$ and also, there are n_z

possible observation symbols $\mathcal{Z} = \{O_1, \cdots, O_{n_z}\}$.

- The random variables $\mathbf{X}_k \in \mathcal{X}$ and $\mathbf{Z}_k^i \in \mathcal{Z}$ represent the state and observation made by agent i at step k, respectively. The realizations of those random variables at step k are denoted as $\mathbf{x}_k$ and $\mathbf{z}_k^i$.

- The transition model is a $n_s \times n_s$ matrix denoted as $\mathcal{P}_{k|k-1} \triangleq \mathrm{p}(\mathbf{X}_k|\mathbf{X}_{k-1})$. All the agents on the network have a copy of this model.

- Each agents has an observation model, which is a $n_s \times n_z$ matrix denoted as $\mathrm{p}(\mathbf{Z}_k^i|\mathbf{X}_k), i \in \boldsymbol{I_n}$. The observation models of different agents can be different.

- The prior, prediction, and posterior probabilities are $1 \times n_s$ random vectors

$$\begin{aligned}
\pi_{k-1} &\triangleq \mathrm{p}\left(\mathbf{X}_{k-1}|\{\mathbf{z}_k^i\}_{k\in\boldsymbol{I_{k-1}}}^{i\in\boldsymbol{I_n}}\right), \\
\tilde{\pi}_k &\triangleq \mathrm{p}\left(\mathbf{X}_k|\{\mathbf{z}_k^i\}_{k\in\boldsymbol{I_{k-1}}}^{i\in\boldsymbol{I_n}}, \mathbf{X}_{k-1}\right), \\
\pi_k &\triangleq \mathrm{p}\left(\mathbf{X}_k|\{\mathbf{z}_k^i\}_{k\in\boldsymbol{I_k}}^{i\in\boldsymbol{I_n}}\right),
\end{aligned}$$

  respectively.

The above HMM is well defined for many distributed estimation applications including ones with dynamic state and time varying observation models. For example, the following transition and observation models can be represented in the above form.

$$\mathbf{X}_{k+1} = f(\mathbf{X}_{k+1}, \mathbf{w}_k) \quad \mathbf{w}_k \sim \mathrm{p}(\mathbf{W}_k), \tag{3.2}$$

$$\mathbf{Z}_{k+1}^i = h^i(\mathbf{X}_{k+1}, \mathbf{v}_k) \quad \mathbf{v}_k \sim \mathrm{p}(\mathbf{V}_k). \tag{3.3}$$

In which, $\mathbf{W}_k$ and $\mathbf{V}_k$ are random variables representing the noise in the model and the observation. We further assume that each agent has a processor and a sensor

on-board. Sensors make observations every Δt seconds and the processors and the network are fast enough to handle calculations based on message passing among agents every δt seconds. We assume that $\delta t \ll \Delta t$. We also assume that the agents exchange their information over the communication channel which is free of both delay and error. Note that the above specification for the HMM and the model may easily be extended to include control inputs but they are omitted as they are not the focus this chapter.

Hence, $\{\mathbf{Z}_k^i\}_{k\in\boldsymbol{I_k}}^{i\in\boldsymbol{I_n}}$ is the indexed family of all the observations made by all the agents up to step k. Moreover, for each agent i, the variable $\mathbf{R}_k^{ij}, j \in \overline{\mathcal{N}}_i$ denotes the information that node i receives from its neighbor, node j, at time k. The set $\mathbf{R}_k^i$ contains all the information that node i has received from its neighbors up to step k and $\mathbf{I}_k^i = \mathbf{R}_k^i \cup \mathbf{Z}_k^i$ represents all the information content that is available to agent i at time k. In general, in this chapter a superscript i denotes that the information in the variable that bears the superscript is a version local to the i^{th} agent. Moreover, symbol η with or without any sub/superscript is a normalizing constant.

3.3 Distributed State Estimation Preliminaries

In the context of HMMs, a Recursive State Estimation is the process of recursively computing the posterior probability of a random dynamic process $\mathbf{X}_k$ conditioned on a sequence of measurements $\{\mathbf{z}_k^i\}_{k\in\boldsymbol{I_k}}^{i\in\boldsymbol{I_n}}$. Bayesian recursive filtering, in a process with the Markov assumptions, has the form

$$\begin{aligned} \mathrm{p}(\mathbf{X}_k|\mathbf{z}_k) &= \frac{1}{\eta}\mathrm{p}(\mathbf{z}_k|\mathbf{X}_k)\mathrm{p}(\mathbf{X}_k|\mathbf{z}_{k-1},\mathbf{X}_{k-1}) \\ &= \frac{1}{\eta}\prod_{i=1}^{n}\mathrm{p}(\mathbf{z}_k^i|\mathbf{X}_k)\int \mathrm{p}(\mathbf{X}_k|\mathbf{X}_{k-1})\mathrm{p}(\mathbf{X}_{k-1}|\mathbf{z}_{k-1})\mathrm{d}\mathbf{X}_{\mathbf{k-1}}. \end{aligned} \tag{3.4}$$

Performing recursive estimation in a sensor network setting and for an HMM can be done in one of the following ways:

3.3.1 Centralized Estimation

In this approach there is a single distinguished node in the network that receives observations $\mathbf{z}_k^{\boldsymbol{I_n}} \triangleq \{\mathbf{z}_k^i\}^{i\in\boldsymbol{I_n}}$ from the rest. The above Bayesian filtering recursion for step k of a finite state HMM consists of first calculating the prediction $\tilde{\pi}_k$ according to

$$\tilde{\pi}_k = \pi_{k-1}\mathcal{P}_{k|k-1}, \tag{3.5}$$

then, updating via:

$$\pi_k = \frac{1}{\eta}\tilde{\pi}_k\mathcal{O}_k, \tag{3.6}$$

where $\mathcal{O}_k$ is an $n_s \times n_s$ diagonal matrix of likelihoods, $\mathrm{p}(\mathbf{z}_k^{\boldsymbol{I_n}}|\mathbf{X}_k)$.

3.3.1.1 Distributed Consensus Based Filtering

Looking at (3.4) one can see that if all agents share the same prior information, they can recover the centralized estimator's performance if they can reach a consensus over the product of measurement probabilities. Distributed averaging methods can be applied here as the nodes need to reach a consensus over the log of the joint measurement probabilities (log-of-likelihood):

$$\tilde{l}_k = \frac{1}{n}\log\prod_{i=1}^{n}\mathcal{O}_k^i = \frac{1}{n}\sum_{i=1}^{n}\log\mathcal{O}_k^i = \frac{1}{n}\sum_{i=1}^{n}\tilde{l}_k^i. \tag{3.7}$$

Once consensus is reached, the updated estimate is

$$\pi_k = \frac{1}{\eta}\overbrace{\underbrace{\pi_k}_{\text{prior}}\mathcal{P}_{k|k-1}}^{\text{prediction}}\underbrace{e^{n\tilde{l}}}_{\text{likelihood}}. \tag{3.8}$$

Reaching a consensus over likelihoods can be implemented for the discrete state variables using a distributed averaging method based on Metropolis-Hastings Markov Chains (MHMC). To avoid confusion we use m to indicate consensus iterations throughout this chapter. On a communication graph G one can use a message passing protocol of the form

$$\begin{aligned} &\psi^i(m+1) = \textstyle\sum_{j=1}^{|\mathcal{N}_i|} \gamma_{ij}(m)\psi^j(m), \\ &\text{such that } \sum_m \gamma_{ij}(m) = 1,\ \psi^i(0) = \tilde{l}_k^i, \end{aligned} \tag{3.9}$$

to calculate the average of the values on the graph nodes in which $d_i(m) = |\mathcal{N}^i|$ is the degree of the node v_i, and

$$\gamma_{ij}(m) = \begin{cases} \frac{1}{1+\max\{d_i(m), d_j(m)\}} & \text{if } (i,j) \in \mathcal{E}_m, \\ 1 - \sum_{(i,n)\in\mathcal{E}} \gamma_{in} & \text{if } i = j, \\ 0 & \text{otherwise.} \end{cases} \tag{3.10}$$

With this messaging passing protocol,

$$\lim_{m\to\infty} \psi^i(m) = \tilde{l}_k.$$

Note that for each node i, the γ_{ij}'s depend only on the degrees of its neighboring nodes. As stated earlier, once a consensus has been reached over likelihoods, the centralized estimate can be recovered. The prerequisite for this method to work is that the network remains connected. This however is too restrictive for many applications.

3.3.1.2 Conservative Approximate Distributed Filtering

In this approach, instead of putting effort into keeping the dependencies between agents' information, a fusion rule is designed to guarantee that no double counting of mutual information occurs. This usually results in replacement of independent information with some form of conservative approximation. Such a treatment results in inferior performance with respect to the exact distributed filter's output.

3.3.1.3 Conservative approximation of a Probability Mass Function (PMF)

The authors in [3] introduced a set of sufficient conditions for a Probability Mass Function (PMF) $\tilde{\mathrm{p}}(\mathbf{X})$ to satisfy in order to be a conservative approximation of another PMF $\mathrm{p}(\mathbf{X})$. The conditions are

- Non-decreasing entropy property:

$$\mathrm{H}(\mathrm{p}(\mathbf{X})) \leq \mathrm{H}(\tilde{\mathrm{p}}(\mathbf{X})).$$

- The order preservation property that,

$$\mathrm{p}(\mathbf{x}_i) \leq \mathrm{p}(\mathbf{x}_j) \text{ iff } \tilde{\mathrm{p}}(\mathbf{x}_i) \leq \tilde{\mathrm{p}}(\mathbf{x}_j),\ \forall \mathbf{x}_i, \mathbf{x}_j \in \mathbf{X}$$

3.3.1.4 Conservative Fusion of two PMFs (CF)

Two probability distribution functions $\mathrm{p}_a(\mathbf{X}|\mathbf{I_a})$ and $\mathrm{p}_b(\mathbf{X}|\mathbf{I}_b)$ can be fused together with the *Geometric Mean Density Rule (GMD)*

$$\begin{aligned} \mathrm{p}_c(\mathbf{X}) &= \frac{1}{\eta_c}\mathrm{p}_a(\mathbf{X}|\mathbf{I_a})^{\omega}\mathrm{p}_b(\mathbf{X}|\mathbf{I_b})^{1-\omega} \\ &= \frac{1}{\eta_c}\mathrm{p}_a(\mathbf{X}|\mathbf{I_a} \setminus \mathbf{I}_b)^{\omega}\mathrm{p}_b(\mathbf{X}|\mathbf{I_b} \setminus \mathbf{I}_b)^{1-\omega}\mathrm{p}_a(\mathbf{X}|\mathbf{I_a} \cap \mathbf{I}_b), \end{aligned} \tag{3.11}$$

in which, $0 \leq \omega \leq 1$. Note that in the above equation the PMFs are raised to the power of ω and multiplied together element-wise. As it can be seen this rule never double counts the mutual information and replaces the independent components with a conservative approximation of them. The interesting property of this fusion rule is that it works without the knowledge of the dependence of two initial PMFs. This fusion rule can be generalized to more than two PMFs. For example, in the context of this chapter, node i calculates a conservative approximation of the centralized estimate and stores it in $\tilde{\pi}^i$. The GMD fusion of these estimates is also a conservative approximation of the centralized estimate.

$$\tilde{\pi}_k = \frac{1}{\eta} \prod_{i=1}^{n} (\tilde{\pi}_k^i)^{\omega_i}, \quad s.t. \quad \sum_{i=1}^{n} \omega_i = 1. \tag{3.12}$$

Remark 1. Several criteria have been proposed to choose the ω_i. One such criterion is [2]:

$$\tilde{\pi} = \arg \min_{\pi} \max_{i} \{\mathcal{D}(\pi \| \tilde{\pi}^i)\}, \tag{3.13}$$

in which the $\mathcal{D}(\pi \| \tilde{\pi}^i)$ is the Kullback-Leibler divergence between π and $\tilde{\pi}^i$.

Remark 2. In [3], the authors have shown that raising a PMF to the power of $\omega \leq 1$ reduces its entropy. From (3.12) it can be seen that applying the GMD rule reduces the entropy of the likelihood probabilities that are independent. This is undesirable and can be avoided by treating the likelihood probabilities separately.

3.3.1.5 Iterative CF (ICF)

At each first iteration of consensus, $m = 0$, for each agent j, initialize the local consensus variable to be

$$\phi^j(0) = \frac{1}{\eta_j} (\tilde{\pi}_k^j \mathcal{O}_k^j).$$

Solve for ω^* such that

$$\begin{aligned} \omega^* &= \arg\min_{\omega} \mathcal{J}\big(\frac{1}{\eta}\prod\nolimits_{j\in\mathcal{N}^i(m)}\big[\phi^j(m)\big]^{\omega_j}\big), \\ s.t. \quad &\prod\nolimits_{j\in\mathcal{N}^i(m)}\omega_j = 1, \quad \forall j, \quad \omega_j \geq 0, \end{aligned} \tag{3.14}$$

where η is the normalization constant and $\mathcal{J}(\cdot)$ is an optimization objective function. Specifically, it can be entropy $H(\cdot)$ or the criteria in (3.13). Estimates are then updated locally for the next iteration

$$\phi^i(m+1) = \frac{1}{\eta^*}\prod\nolimits_{j\in\mathcal{N}^i(m)}\big[\phi^j(m)\big]^{\omega_j^*}. \tag{3.15}$$

3.4 Hybrid ICF and CL

We propose a hybrid approach that uses CF to reach consensus over priors and the MHMC-based consensus filter for distributed averaging of local information updates. Our method is summarized in Algorithm 2. We explain the flow of the proposed method using a simple scenario with two agents. Generalization to more than two agents is straightforward and follows similarly.

Imagine a scenario consisting of two agents, observing, $\mathbf{x}_k$, the state of a Markov chain at time k, that are communicating with each other through a time-varying network topology. Initially, the agents start with priors π_0^1 and π_0^2 respectively.

At step k the chain transitions to the new state $x(k)$ and agents calculate their own local prediction $\tilde{\pi}_k^1$ and $\tilde{\pi}_k^2$ respectively (line 1 in the algorithm). Then they make observations z_k^1 and z_k^2, respectively, and compute the local likelihood matrices $\mathcal{O}_k^1$ and $\mathcal{O}_k^1$ (line 2 of the algorithm).

Clearly, if there was a centralized agent that received all the observations at all

times the fused estimate would be

$$^{\text{CEN}}\pi_{k+1} = \frac{1}{\eta}\ {}^{\text{CEN}}\tilde{\pi}_k \mathcal{O}_k^1 \mathcal{O}_k^2. \tag{3.16}$$

However, if the two agents were to perform ICF, would find a fused estimate

$$^{\text{ICF}}\pi_{k+1} = \frac{1}{\eta}\big({}^{\text{ICF}}\tilde{\pi}_k^1 \mathcal{O}_k^1\big)^{\omega^{\text{ICF}}}\big({}^{\text{ICF}}\tilde{\pi}_k^2 \mathcal{O}_k^2\big)^{1-\omega^{\text{ICF}}}, \tag{3.17}$$

where w^{ICF} is obtained from solving the optimization problem in (3.14). Note that doing MHMC alone is not possible here since ${}^{\text{ICF}}\tilde{\pi}_k^1$ and ${}^{\text{ICF}}\tilde{\pi}_k^2$ differ. In our hybrid method we do the following:

$$^{\text{HYB}}\pi_{k+1} = \frac{1}{\eta_1} \underbrace{\big({}^{\text{HYB}}\tilde{\pi}_k^1\big)^{\omega^{\text{HYB}}}\big({}^{\text{HYB}}\tilde{\pi}_k^2\big)^{1-\omega^{\text{HYB}}}}_{\substack{\text{ICF to reach} \\ \text{consensus over priors}}} \underbrace{\mathcal{O}_k^1 \mathcal{O}_k^2}_{\substack{\text{consensus over} \\ \text{the incremental} \\ \text{information}}}. \tag{3.18}$$

The above equations demonstrate why the hybrid method is capable of recovering the centralized estimate if, after a disconnection, the network stays path connected long enough. Assume that the agents are disconnected at time $k-1$and resume connection at time k. Then the agents have different priors at time k. Therefore, for the ICF method Eq. 3.17 and for the hybrid method Eq. 3.18 should be used for their update step. At this point one of the two posteriors could be closer to the centralized estimate. However, as time goes forward and the two agents remain connected, due to the forgetting property of Markov chains, the priors of the centralized estimate and the hybrid method reconcile and therefore, Eq. 3.18 becomes the same as Eq. 3.16.

For an n-agent system with the i^{th} agent having prior ${}^{\text{HYB}}\tilde{\pi}_k^i$, the ICF approach is used to find a consensus over the priors using (3.14) recursively. The MHMC approach is used to form the consensus over the new information, i.e., $\sum_{j=1}^{n} \tilde{l}_k^i$ (3.10).

In line 12 of the algorithm, n_{CG} is the number of agents that form a connected group, and it can be determined by assigning unique IDs to the agents and passing these IDs along with the consensus variables. Each agent keeps track of the unique IDs it receives and passes them to its neighbors. The following propositions hold.

Proposition 4. *The ICF process is guaranteed to reach consensus over the priors, i.e.,* $\exists\ \phi_\infty$,

$$\lim_{m\to\infty} \phi^i(m) = \phi_\infty \quad \forall i.$$

The same result is already established for the distributed averaging process, i.e., $\exists \psi_\infty$

$$\lim_{m\to\infty} \psi^i(m) = \psi_\infty \quad \forall i.$$

3.5 Experiments

The first experiment is concerned with a distributed target pose estimation problem in a grid using multiple observers connected through a changing topology network. Fig. 3.1 depicts the 2D grid in which a target performs a random walk while six observers are trying to estimate its position. Each white cell is modeled as a single state of our HMM representing the position of the target on the grid. The observers' motion is deterministic; four of them are rooks moving along the borders and the other two are bishops moving diagonally on the grid. In order to detect the target, each observer sends a straight beam normal to its direction of motion as shown in the figure. The beam hits either the target, or a wall (that can be an obstacle or a border). In the former case the observer senses the position of the target based on a discrete one dimensional Gaussian distribution over the states that the beam has traveled. In the latter case, under the assumption of no "false positives", the observer produces the "no target" symbol as an additional state incorporated into

Algorithm 2: Hybrid Method

Input : π_{k-1}^j

Use (3.6) to calculate $\tilde{\pi}_k^j$

Collect local observation z_k^j and calculate $\mathcal{O}_k^j$ and $\tilde{l}_k^j$

Initialize consensus variables

$$\phi^j(0) = \tilde{\pi}_k^j, \quad \psi^j(0) = \tilde{l}_k^j$$

$m = 0$

while *NOT CONVERGED* **do**

  BROADCAST$[\psi^j(m), \phi^j(m)]$

  RECEIVE$[\psi^i(m), \phi^i(m)] \quad \forall i \in \mathcal{N}^j$

  Collect received data

$$\mathcal{C}^j(m) = \{\phi^{i\in\mathcal{N}^j}(m)\}, \quad \mathcal{M}^j(m) = \{\psi^{i\in\mathcal{N}^j}(m)\}$$

  Do one iteration of CI on consensus variables for local prior information $\mathcal{C}_m^j$

$$\phi^j(m+1) = \texttt{ICF}(\mathcal{C}^j(m))$$

  Do one iteration of MHMC on consensus variables for new information

$$\psi^j(m+1) = \texttt{MHMC}(\mathcal{M}^j(m))$$

  $m = m + 1$

Calculate the posteriors according to:

$$\pi_k^j = e^{n_{\mathrm{CG}}\psi^j(m)}\phi^j(m)$$

the observation model by setting zero probabilities in the likelihood matrix for those states that beam has traveled until it has hit a wall.

At each Markov transition step, each observer carries out its distributed estimation step for the position of the target, which is shared with other connected observers through a communication network. The network topology has two components; one has the rook observers and the other one has the bishop. The observers in each

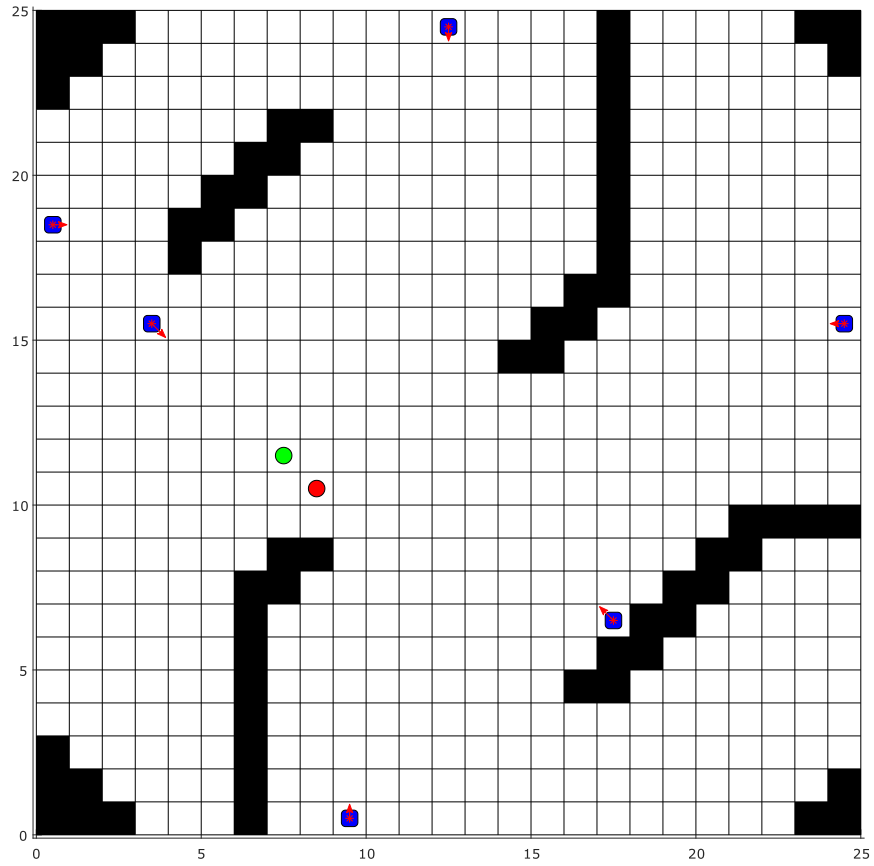


Figure 3.1: The grid map of the environment, dark cells depict obstacles; blue circles are trackers and the red circle is the ground truth location of the maneuvering target; the green circle depicts the observation an agent.

component are connected all the time while the link between the two components gets connected and disconnected intermittently. All communications occur at a higher rate than Markov transition steps, which allows the connected nodes to reach consensus over the shared information.

We evaluate the performance of the proposed method during the phase where the rooks become disconnected from bishops and are reconnected again after some interval. In order to make a comparison, each node performs three instances of estimation. In one instance it uses our hybrid method to fuse its prior along with the received priors

while in the second instance it uses ICF methods to fuse its posterior along with the received posteriors. The third instance concerns a hypothetical god's-eye-view centralized estimator, for comparison purposes. To make such a comparison we use the Bhattacharyya distance [8] between the estimation results and the centralized estimator. The Bhattacharyya distance can be used to evaluate the similarity of two probability mass functions, $\pi_1(\mathbf{X}), \pi_2(\mathbf{X})$ as:

$$D_B(\pi_1(\mathbf{X}), \pi_2(\mathbf{X})) = -\ln(\textstyle\sum_{\mathbf{x}\in\mathbf{X}} \sqrt{\pi_1(\mathbf{x})\pi_2(\mathbf{x})}). \tag{3.19}$$

In the case of $P_1 = P_2$, complete similarity, $D_B(p_1, p_2) = 1$, while $D_B(p_1, p_2) = 0$ means complete dissimilarity.

In Fig. 3.2 we compare the performance of both hybrid and ICF methods in terms of their respective Bhattacharyya distances to the ideal centralized case. As it can be seen, the proposed hybrid method outperforms CF and is able to get the performance very close to centralized solution after reconnection. Based on the Bhattacharyya distance, closeness between centralized and distributed estimates drops during disconnection interval, as is expected, since observers do not have access to all the information available to the centralized estimator. While the hybrid method is able to immediately recover after reconnection, ICF continues to have worse performance even after reconnection due to the fact that it ignores the correlations.

Fig. 3.2 also gives a detailed view of the performance of the hybrid method and compares the estimation results of observers 3, a rook, and 5, a bishop, during three different time steps. The shaded area shows the time during which rooks are disconnected from the bishops. The higher difference between centralized and distributed estimate for the fifth observer, which is a bishop, can be explained based on the fact that it has less information at its disposal. However, after reconnection both

groups are able to converge to the same value which is very close to the centralized estimator.

In second experiment we have evaluated the robustness of the proposed method for networks with different likelihoods of link failure. We report the Bhattacharyya coefficient, and Helinger distance vs. link failure probability for a general distributed HMM with a network of size 20 and state size 30 with each node roughly connected to 10% of the other nodes. We simulate the system multiple times, each time for 150 time steps but with different probability of link failure. At the beginning of each step, a 2 regular graph with 15 nodes is generated and, given a probability of failure for each link, some links in the graph will randomly be disconnected. The graph still remains connected some portion of the time, but this depends on the degree and probability of failure. If the regularity degree goes down or the probability of failure increases, more often than not, the graph becomes disconnected. In practice, for $p \geq 0.05$, consensus methods that rely on full connectivity are no longer succeed since the network almost always suffers disconnection at some point in time.

We ran our method for 150 steps for a range of probabilities of link failure and compared performances with the ideal centralized result (which is obtained by assuming full connectivity at all times). The performance is evaluated by calculating the average value for the Bhattacharyya coefficient and determinant ratio measure at all steps and for all receptors. Based on Fig. 3.3, for the case considered in this experiment, our distributed estimator performs close to the ideal centralized one for $p \in [0.0, 0.1]$, and drastically outperforms ICF all the time. This means that in the case considered here, our method can perform almost as well as the ideal estimator for an unreliable network. Obviously the performance can vary from one system to another and under different network topologies, but, our method can recover the performance of the centralized method when the network is unreliable

and substantially outperforms ICF as has already been established theoretically.

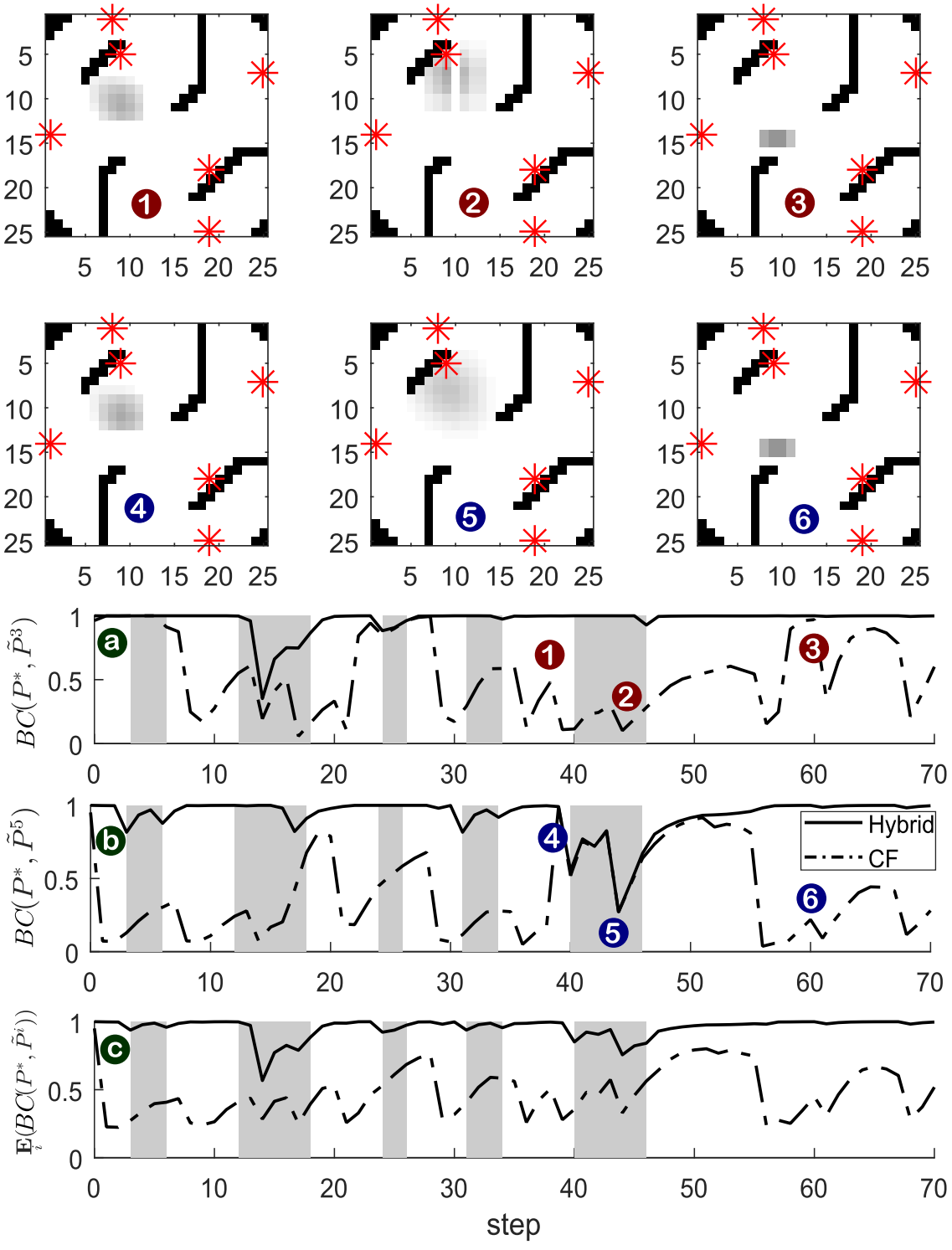


Figure 3.2: Estimation performance in the tracking example

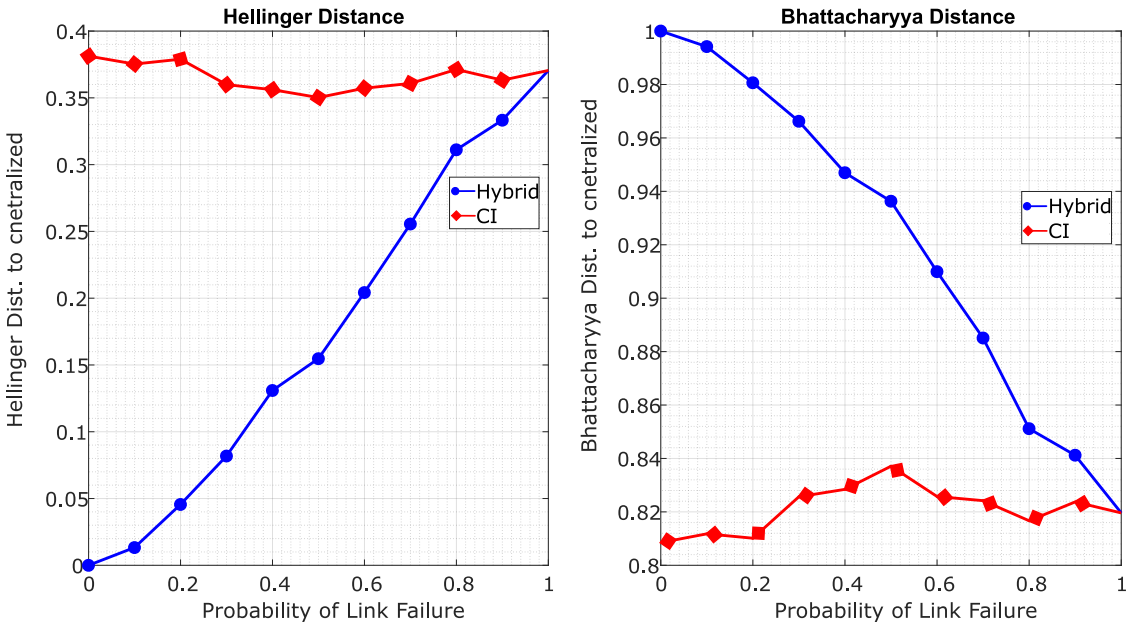


Figure 3.3: Performance comparison between the proposed method and ICF.

4. CONCLUSION

In chapter two I introduced a distributed estimator for dynamic systems in networks with changing topology and those that do not remain connected all the time. Separating the process of consensus for the correlated and uncorrelated information was the key to achieve a better performance compared to Covariance Intersection alone. Evaluating the proposed method on an 80-dimensional estimation problem showed substantial performance improvement compared to CI and also the ability to recover after a disconnection interval occurs.

Chapter three proposes a distributed state estimator for discrete-state dynamic systems with non-Gaussian noise in networks with changing topology and those that do not remain connected all the time. Separating the process of consensus for the correlated and uncorrelated information was the key to achieving better performance compared to ICF alone. Evaluating the proposed method on a multi-agent tracking application and a high-dimensional HMM distributed state estimator problem showed substantial performance improvement compared to the state of the art. We are able to achieve robustness and recover performance after a disconnection interval occurs.

REFERENCES

[1] Nisar R Ahmed, Simon J Julier, Jonathan R Schoenberg, and Mark E Campbell. Decentralized bayesian fusion in networks with non-gaussian uncertainties. *Multisensor Data Fusion: From Algorithms and Architectural Design to Applications*, page 383, 2015.

[2] Jiří Ajgl and Miroslav Šimandl. Design of a robust fusion of probability densities. In *American Control Conference (ACC), 2015*, pages 4204–4209. IEEE, 2015.

[3] Tim Bailey, Simon Julier, and Gabriel Agamennoni. On conservative fusion of information with unknown non-gaussian dependence. In *2012 International Conference on Information Fusion (FUSION)*, pages 1876–1883, 2012.

[4] Giorgio Battistelli and Luigi Chisci. Kullback–leibler average, consensus on probability densities, and distributed state estimation with guaranteed stability. *Automatica*, 50(3):707–718, 2014.

[5] Giorgio Battistelli and Luigi Chisci. Stability of consensus extended kalman filter for distributed state estimation. *Automatica*, 68:169–178, 2016.

[6] Giorgio Battistelli, Luigi Chisci, and Claudio Fantacci. Parallel consensus on likelihoods and priors for networked nonlinear filtering. *IEEE Signal Processing Letters*, 21(7):787–791, 2014.

[7] Dimitri P Bertsekas and John N Tsitsiklis. Parallel and distributed computation, prentice hall inc., 1989.

[8] Anil Bhattacharyya. On a measure of divergence between two multinomial populations. *Sankhyā: The Indian Journal of Statistics*, pages 401–406, 1946.

[9] F. Bullo. *Lectures on Network Systems*. Version 0.95, 2017. With contributions by J. Cortes, F. Dorfler, and S. Martinez.

[10] Mark E Campbell and Nisar R Ahmed. Distributed data fusion: Neighbors, rumors, and the art of collective knowledge. *IEEE Control Systems*, 36(4):83–109, 2016.

[11] J. Capitan, L. Merino, F. Caballero, and A. Ollero. Delayed-state information filter for cooperative decentralized tracking. In *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pages 3865–3870, May 2009.

[12] D.W. Casbeer and R. Beard. Distributed information filtering using consensus filters. In *American Control Conference, 2009. ACC '09.*, pages 1882–1887, June 2009.

[13] Federico S Cattivelli and Ali H Sayed. Diffusion strategies for distributed kalman filtering and smoothing. *IEEE Transactions on Automatic Control*, 55(9):2069–2084, 2010.

[14] Lingji Chen, Pablo O Arambel, and Raman K Mehra. Estimation under unknown correlation: covariance intersection revisited. *IEEE Transactions on Automatic Control,*, 47(11):1879–1882, 2002.

[15] Andrea Cristofaro, Alessandro Renzaglia, and Agostino Martinelli. Distributed information filters for mav cooperative localization. In *Distributed Autonomous Robotic Systems*, pages 133–146. Springer, 2013.

[16] Zili Deng, Peng Zhang, Wenjuan Qi, Jinfang Liu, and Yuan Gao. Sequential covariance intersection fusion kalman filter. *Information Sciences*, 189:293–309, 2012.

[17] Hugh Durrant-Whyte, Mike Stevens, and E Nettleton. Data fusion in decentralised sensing networks. In *Proceedings of the 4th International Conference on Information Fusion*, pages 302–307, 2001.

[18] Ondrej Hlinka, Franz Hlawatsch, and Petar M Djuric. Distributed particle filtering in agent networks: A survey, classification, and comparison. *IEEE Signal Processing Magazine*, 30(1):61–81, 2013.

[19] Ondrej Hlinka, Ondrej Sluciak, Franz Hlawatsch, Petar M Djuric, and Markus Rupp. Likelihood consensus and its application to distributed particle filtering. *IEEE Transactions on Signal Processing*, 60(8):4334–4349, 2012.

[20] Jinwen Hu, Lihua Xie, and Cishen Zhang. Diffusion kalman filtering based on covariance intersection. *IEEE Transactions on Signal Processing,*, 60(2):891–902, 2012.

[21] Wenling Li and Yingmin Jia. Distributed consensus filtering for discrete-time nonlinear systems with non-gaussian noise. *Signal Processing*, 92(10):2464–2470, 2012.

[22] Lin Mao and Da Wei Yang. Distributed information fusion particle filter. In *Intelligent Human-Machine Systems and Cybernetics (IHMSC), 2014 Sixth International Conference on*, volume 1, pages 194–197, Aug 2014.

[23] Reza Olfati-Saber. Distributed kalman filter with embedded consensus filters. In *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC'05. 44th IEEE Conference on*, pages 8179–8184. IEEE, 2005.

[24] Reza Olfati-Saber and Jeff S Shamma. Consensus filters for sensor networks and distributed sensor fusion. In *Decision and Control, 2005 and 2005 European*

Control Conference. CDC-ECC'05. 44th IEEE Conference on, pages 6698–6703. IEEE, 2005.

[25] Alex Olshevsky and John N Tsitsiklis. Convergence speed in distributed consensus and averaging. *SIAM Journal on Control and Optimization*, 48(1):33–55, 2009.

[26] A. Simonetto, T. Keviczky, and R. Babuka. Distributed nonlinear estimation for robot localization using weighted consensus. In *IEEE International Conference on Robotics and Automation*, pages 3026–3031, May 2010.

[27] John M Stockie. The mathematics of atmospheric dispersion modeling. *SIAM Review*, 53(2):349–372, 2011.

[28] Amirhossein Tamjidi, Suman Chakravorty, and Dylan Shell. Unifying consensus and covariance intersection for decentralized state estimation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 125–130, 2016.

[29] Lieven Vandenberghe, Stephen Boyd, and Shao-Po Wu. Determinant maximization with linear matrix inequality constraints. *SIAM journal on Matrix Analysis and Applications*, 19(2):499–533, 1998.

[30] Yimin Wang and X.R. Li. Distributed estimation fusion with unavailable cross-correlation. *IEEE Transactions on Aerospace and Electronic Systems,* , 48(1):259–278, Jan 2012.

[31] William W Whitacre and Mark E Campbell. Cooperative estimation using mobile sensor nodes in the presence of communication loss. *Journal of Aerospace Information Systems*, 10(3):114–130, 2013.

[32] Lin Xiao and S. Boyd. Fast linear iterations for distributed averaging. In *2nd IEEE Conference on Decision and Control, 2003. Proceedings. 4*, volume 5, pages 4997–5002 Vol.5, Dec 2003.

[33] Lin Xiao, S. Boyd, and S. Lall. A scheme for robust distributed sensor fusion based on average consensus. In *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*, pages 63–70, April 2005.

[34] Lin Xiao, Stephen Boyd, and Sanjay Lall. A scheme for robust distributed sensor fusion based on average consensus. In *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks*, page 9. IEEE Press, 2005.

[35] Dan Yu and Suman Chakravorty. A randomized proper orthogonal decomposition technique. In *American Control Conference (ACC), 2015*, pages 1137–1142. IEEE, 2015.

[36] Haotian Zhang, J.M.F. Moura, and B. Krogh. Dynamic field estimation using wireless sensor networks: Tradeoffs between estimation error and communication cost. *IEEE Transactions on Signal Processing,*, 57(6):2383–2395, June 2009.