Robust Distributed Parameter Estimation in Wireless Sensor Networks

by

Jongmin Lee

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved August 2017 by the
Graduate Supervisory Committee:

Cihan Tepedelenlioğlu, Co-Chair
Andreas Spanias, Co-Chair
Konstantinos Tsakalis
Martin Reisslein

ARIZONA STATE UNIVERSITY

December 2017

ABSTRACT

Fully distributed wireless sensor networks (WSNs) without fusion center have advantages such as scalability in network size and energy efficiency in communications. Each sensor shares its data only with neighbors and then achieves global consensus quantities by in-network processing. This dissertation considers robust distributed parameter estimation methods, seeking global consensus on parameters of adaptive learning algorithms and statistical quantities.

Diffusion adaptation strategy with nonlinear transmission is proposed. The nonlinearity was motivated by the necessity for bounded transmit power, as sensors need to iteratively communicate each other energy-efficiently. Despite the nonlinearity, it is shown that the algorithm performs close to the linear case with the added advantage of power savings. This dissertation also discusses convergence properties of the algorithm in the mean and the mean-square sense.

Often, average is used to measure central tendency of sensed data over a network. When there are outliers in the data, however, average can be highly biased. Alternative choices of robust metrics against outliers are median, mode, and trimmed mean. Quantiles generalize the median, and they also can be used for trimmed mean. Consensus-based distributed quantile estimation algorithm is proposed and applied for finding trimmed-mean, median, maximum or minimum values, and identification of outliers through simulation. It is shown that the estimated quantities are asymptotically unbiased and converges toward the sample quantile in the mean-square sense. Step-size sequences with proper decay rates are also discussed for convergence analysis.

Another measure of central tendency is a mode which represents the most probable value and also be robust to outliers and other contaminations in data. The proposed distributed mode estimation algorithm achieves a global mode by recursively shifting conditional mean of the measurement data until it converges to stationary points of estimated density function. It is also possible to estimate the mode by utilizing grid vector as well as kernel density estimator. The densities are estimated at each grid point, while the points are updated until they converge to a global mode.

To My Parents and Family.

ACKNOWLEDGMENTS

I would like to begin with my special thanks to God for guiding me and always being with me. I would also like to thank my dear family - parents, wife Jae Yoon An, brother Jongyoung Lee, and my parents-in-law, for their unconditional love, support, patience, and prayers.

I would like to express my deepest gratitude to my co-advisors, Dr. Cihan Tepedelenlioğlu and Dr. Andreas Spanias, for their excellent guidance, encouragement, patience, and continuous support. I would never been able to complete my dissertation without their support. I am very fortunate to work with them, experiencing diverse research opportunities in industry as well as academia. They are always willing to help and support.

I would like to express my sincere appreciation to Dr. Konstantinos Tsakalis and Dr. Martin Reisslein who were willing to participate in my defense committee. Also I would like to thank Michael Stanley at NXP Semiconductors for his help and support. I am fortunate to work with him, exploring sensors and machine learning algorithms with my knowledge. I always learn new things and extend my knowledge which make me exciting. His insightful discussion and enthusiasm at work always motivate me. I would like to thank Dr. Mahesh Banavar for his help and guidance when he was with Arizona State University.

I would like to thank my colleagues, Xue Zhang, Sai Zhang, Henry Braun, Shwetang Peshin, David Ramirez, Sunil Rao, Huan Song, Uday Shankar Shanthamallu, Jie Fan, Abhinav Dixit, Sameeksha Katoch for their help and various experience at SenSIP center. Also I would like to extend my thanks to Ahmed Ewaisha, Ruochen Zeng, and Xiaofeng Li for useful discussions and presentations in Signal Processing and Communications group under the guidance of Dr. Cihan Tepedelenlioğlu.

TABLE OF CONTENTS

Page

APPENDIX

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

Figure Page

Chapter 1

INTRODUCTION

1.1 Distributed Wireless Sensor Networks

A wireless sensor network (WSN) is a network of spatially distributed sensor devices to monitor or measure physical phenomena observable over a certain region. Sensor devices are typically small, inexpensive, memory-limited, and lightweight [1]. Sensors are deployed in difficult-to-access locations with limited battery power. A radio is implemented for wireless communication, which is essential to transfer sensed data for fusion or processing with the other collected data. Depending on applications and type of sensors used, actuators may be incorporated in sensor devices [2], in order to control or monitor the network of sensors and the sensors themselves. More general overviews and surveys about WSNs can be found in [1–3].

It is also important to address how to fuse the sensing data gathered from sensor networks. There are generally two types of sensor networks: centralized and distributed. In centralized networks, a fusion center is located in the network and all the data observed by sensor nodes are transmitted to the fusion center. This type of sensor fusion is easy to control and fast to obtain data processing outcomes. However, if the sensor nodes are deployed in a large area with limited power resources, lifetime of sensor nodes far from the fusion center can be shortened. On the other hand, when there is no fusion center in distributed networks, each sensor node communicates only with neighbors and probably in-network processing can be a viable solution because data is processed by the local sensor nodes themselves.

A traditional problem in distributed networks is to estimate an arithmetic mean of measurement data by iteratively averaging the states with neighboring ones and achieving consensus on the global average of the initial measurements [4, 5]. This has influenced many distributed estimation applications due to the broad use of the arithmetic mean in signal processing techniques. Consensus on linear system parameter estimation of adaptive learning algorithms (e.g., least-mean-square, LMS) can be achieved by iteratively averaging intermediate states of the estimates with neighboring ones, even in fully distributed networks [6]. However, the linear averaging process can be impractical and inefficient when intermediate states are transmitted to neighbors, and therefore constrained transmission may be needed. Consensus on an average of measurement data can be used for a measure of central tendency of the sensed data in monitoring applications. However, when the data distribution is skewed or has even a small number of outliers, the average can be highly biased, and robust metrics such as median (or quantiles for more general metric) and mode may be needed.

This dissertation studies consensus-based estimation methods in fully distributed wireless networks. When real-time data are observed at each node, individual nodes can locally estimate system parameters using adaptive learning algorithms [7–11]. Collaboration with neighboring nodes during the adaptive learning process is beneficial because all the nodes have a common objective - the system parameter estimation, which can be obtained by achieving *consensus* on the estimates. Nonlinearity is motivated by the necessity for bounded transmission power when the intermediate estimates of parameters are averaged with neighbors.

Another use of consensus-based estimation is a robust measure of central tendency in sensed data. An average consensus of data is achievable by distributed averaging schemes [4, 12], and may be used to estimate central tendency of the data such as average temperature over a network. However, an average can be highly sensitive to outliers and skewness of the data distribution. Among the many statistical metrics for central tendency, one can consider median which represents 50% of data. More general metric than median is quantile that can be also used for outliers removal or trimmed mean. Mode is another metric for a measure of central tendency. A mode is obtained by searching the most densest region of data distribution. Arguably the mode is the closest one to the intuitive understanding of the measure of central tendency, as it represents the most probable value of sensed data. This dissertation describes consensus-based *quantile* and *mode* estimation methods to measure the central tendency of sensed data in fully distributed wireless networks.

In the remain of this chapter, we briefly review background knowledge relevant to this dissertation. Graph theory and distributed network structure is explained, and then related works are introduced by unifying various works based consensus estimation approaches. Related applications are introduced and contributions of this dissertation are addressed.

1.2 Notations and Conventions

Vectors and matrices are denoted by boldface lower-case and upper-case, respectively. $\lambda_n(\mathbf{L})$ denotes the n-th smallest eigenvalue of matrix $\mathbf{L}$. The vector $\mathbf{1}$ denotes a column vector of all ones and $\mathbf{I}$ denotes an identity matrix. The symbol $\|\cdot\|$ denotes the l_2 norm for real vectors and spectral norm for symmetric matrices. $\operatorname{diag}[a_1, a_2, \ldots, a_N]$ denotes a $N \times N$ diagonal matrix $\mathbf{A}$ with the n-th element a_n. $E[\cdot]$ denotes the expectation operator. Sets are denoted by blackboard bold upper-case: for example, a set of N nodes is denoted by $\mathbb{N} = \{1, 2, \ldots, N\}$. Calligraphic symbols denote distribution: $\mathcal{N}$ represents normal (Gaussian) distribution and log-normal distribution is denoted by $\ln \mathcal{N}$. Bold calligraphic symbols denote block matrices and vectors with Kronecker product $\otimes$: for example, $\boldsymbol{\mathcal{L}} \triangleq \mathbf{L} \otimes \mathbf{I}$ where $\boldsymbol{\mathcal{L}}$, $\mathbf{L}$, and $\mathbf{I}$ are $NM \times NM$, $N \times N$, and $M \times M$ respectively.

1.3 Background Review

1.3.1 Graph Theory and Distributed Network

Network graph theory is briefly summarized in this section. There is an undirected graph $\mathbb{G} = (\mathbb{N}, \mathbb{E})$ containing a set of nodes $\mathbb{N} = \{1, \ldots, N\}$ and a set of edges $\mathbb{E}$. The neighbors of node n is denoted by $\mathbb{N}_n = \{l | \{l, n\} \in \mathbb{E}\}$ where $\{n, l\}$ is an edge between the nodes n and l [13]. Each node communicates with neighbors via the edges. The degree d_n at node n denotes the number of neighbors at n, and $d_{\max} = \max_n d_n$. It is called that a graph is *connected* if there exists at least one path between every pair of nodes. The graph structure is described by adjacency matrix $\mathbf{B}$, which is an $N \times N$ symmetric matrix. The element $b_{nl} = 1$ of $\mathbf{B}$ if $\{n, l\} \in \mathbb{E}$. The diagonal matrix $\mathbf{D} = \mathrm{diag}\,[d_1, d_2, \ldots, d_N]$ represents the degrees of all the nodes in the network. The Laplacian matrix is given by $\mathbf{L} = \mathbf{D} - \mathbf{B}$.

The graph Laplacian characterizes a number of useful properties of the graph. The eigenvalues of $\mathbf{L}$ are non-negative and the number of zero eigenvalues denotes the number of distinct components of the graph. When the graph is connected, $\lambda_1(\mathbf{L}) = 0$, and $\lambda_n(\mathbf{L}) > 0$, $2 \leq n \leq N$, so that the rank of $\mathbf{L}$ for a connected graph is $N - 1$. The vector $\mathbf{1}$ is the eigenvector of $\mathbf{L}$ associated with the eigenvalue 0, i.e., $\mathbf{L1} = 0$. The eigenvalue $\lambda_2(\mathbf{L})$ characterizes how densely the graph is connected and the performance of consensus algorithms depend on this eigenvalue [14].

In this dissertation we consider *connected* networks where at least one path connecting any two arbitrary nodes exists. The nodes may be connected directly by an

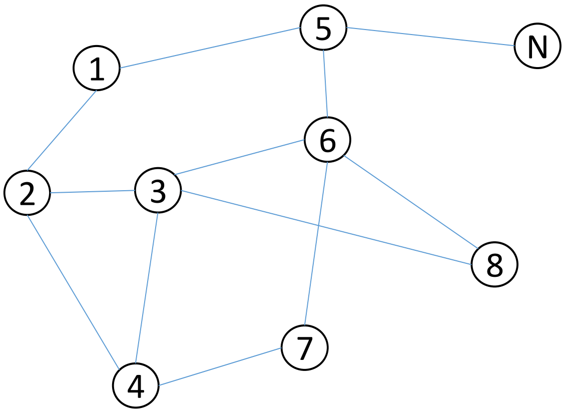


Figure 1.1: A Distributed Network Topology.

edge if they are neighbors, or they may be connected by a path that passes through other intermediate nodes. Fig. 1.1 illustrates a graphical view of a connected network with N nodes. The neighborhood is defined as the set of nodes that are connected to it by edges, excluding the node itself. For example, the neighborhood at node 6 is defined as $\mathbb{N}_6 = \{3, 5, 7, 8\}$ and the node 6 has degree $|\mathbb{N}_6| = 4$. Since the network does not have a fusion center, the network is also called *distributed* network.

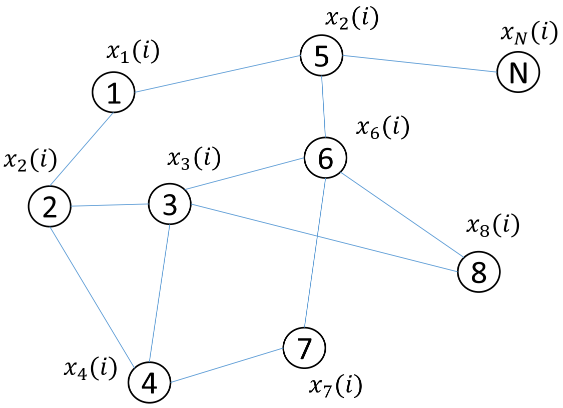
Each sensor node n observes a certain type of data in this network, and it shares the observed data with its neighbors. The aim of sensing at each node n is to collect and interpret the data in order to provide monitoring systems with desired information. Thus, it is assumed that the nodes have a common goal, and they are expected to cooperate for achieving the goal. One straightforward method to achieve the common goal is a flooding scheme. Each node maintains a table that is consists of the sensed data collected from all nodes. At each iteration time instance, the nodes exchange their own table information with their neighbors over the network. This process is iteratively continued until every node obtains the entire sensed data. However, there is a *distributed iteration* method that can achieve more efficient and faster computation because each node shares its own data only with neighbors. In such a distributed process, every node seeks to *consensus* of desired information by iterative communications.

Depending on *consensus* applications of interests, we may classify the observed data into *static* or *dynamic*. Let x and y denote the static and dynamic real scalar data, respectively. Let N denote the number of nodes in the network. With the static data, each node observes x_n, where $n = 1, 2, \ldots, N$, initially and wishes to converge to a function of all the values $\{x_n\}_{n=1}^{N}$ after a number of iterations. With the dynamic data, on the other hand, each node observes $y_n(i)$, where $n = 1, 2, \ldots, N$, at time i and has ability to adapt and learn the network in response to changes in the statistical properties of the data. Interestingly, learning a network parameter that is the common goal of all the sensor nodes can be improved by cooperation over the network.

1.3.2 Distributed Average Consensus

Fully distributed sensor networks are scalable and energy efficient, as each node shares its sensing data with neighboring nodes only. A traditional problem in this domain is to estimate an average of measurements by iteratively averaging the states with neighboring ones, and achieves a *consensus* on the global average of the initial measurements [4, 5, 15]. This influenced the problem of distributed estimations because the broad use of the arithmetic mean in signal processing techniques.

Distributed average consensus algorithms asymptotically achieve an average of the initial measurement data at sensors. As shown in Fig. 1.2, each node maintains a sensor measurement, denoting $x_n(0)$ at node n time $i = 0$. In a connected graph network, every node n transmits its measurement $x_n(i)$ at time i through the links connected to its neighbors. After a number of iterations, as $i \to \infty$, a consensus on the global average of the initial measurements $\{x_n(0)\}_{n=1}^{N}$ is achieved. There are



Avg. consensus after i iterations:

$$x_n(i) = \frac{1}{N}\sum_{n=1}^{N} x_n(0)\,,\forall n$$

Figure 1.2: Consensus to an Average Value over the Network.

significant number of works based on this idea. We may unify some of the existing works with the following scenarios.

- *Fixed* topologies without noisy links [4]
- *Fixed* topologies with *noisy* links [16–19]
- *Random* topologies without noisy links [14, 20–25]
- *Random* topologies with *noisy* links [26]
- *Nonlinear bounded transmission* for transmit power saving, whereas the above scenarios are all linear [27]

The fixed topologies mean that the networks are fixed and do not change over time. Reference [4] designed the optimal link weights for this type of network where the links between nodes are noiseless. When the links between nodes are connected via wireless communication channels, transmitted data is corrupted by random noise. Refs [16–19, 21] consider consensus algorithm for such fixed network and noisy links. To achieve a consensus, decreasing weight sequences are used. The random noises are assumed temporally white noise. The consensus is not the global average of initial measurement data but a random variable. The network topologies can be time-varying due to random link failure. When the network links fail at random but they are noiseless, the impacts of network topologies on distributed average consensus algorithms were studied in [14]. A sufficient condition for mean-square convergence of the distributed average consensus algorithm is provided, in terms of a moment of the network graph Laplacian matrix $\mathbf{L}$. Random link failures but noiseless also have been considered in [20, 22–25, 28]. Refs [20, 22, 28] assume an erasure model: the network links fail independently in space and time. Refs [23, 25] study directed topologies with only time i.i.d. link failure, but impose distributional assumptions on the link formation process. In [24], the link failures are i.i.d. Laplacian matrices, the graph is directed, and no distributional assumptions are made on the Laplacian matrices. In time-varying links of network with additive random noise, [26] studies the algorithm that forces the parameter α to decay to zero to guarantee the convergence when there are random link failures and noisy links. In distributed systems, it is often assumed that the power amplifiers used are perfectly linear over the entire range of the sensed observations. In practice, however, the amplifiers exhibit nonlinear behavior when the amplitude of the sensed data is relatively high [29, 30]. Thus, a distributed average consensus algorithm in which every sensor transmits with bounded peak power is studied in [27]. Every sensor maps its observed data through a bounded nonlinear function before transmission to constrain the peak transmit power. Therefore the magnitude of the transmitted signal at every node in every iteration is always bounded, making it ideal for resource-constrained wireless network topologies. Reference [27] considered the fixed topology and noisy link condition of network.

1.3.3 Distributed Parameter Estimation

In this dissertation we focus on distributed parameter estimation problems where the distributed sensor nodes have their own measurement data in order to estimate a global parameter from the measurement. The parameter estimation is performed with distributed average consensus algorithms. These problems have been intensively solved in terms of many deterministic and stochastic iterative algorithms in a distributed parallel implementation. Multiple nodes perform *local update*, while *exchanging* the states with a certain common goal. Early works for this problems in terms of distributed asynchronous iterations in parallel stochastic application are referred to [31–34]. A series of extensive works are referred to *diffusion adaptation* that is summarized in the following subsection. Depending on data types and applications, there are other works that provide distributed parameter estimation based on consensus approach. Reference [35] showed convergence analysis comparing vanishing and non-vanishing step-sizes for seeking consensus in distributed network, based on stochastic approximation theory. Reference [36] also showed distributed parameter estimation problems seeking consensus with convergence analysis and two steps - consensus and innovation where the innovation step is from observation of measurement data. Reference [37] provides convergence analysis of the distributed parameter estimation by consensus based stochastic approximation.

1.3.3.1 Diffusion Adaptation

When the common goal at each sensor node is to find an unknown parameter vector from measurements collected over a fully distributed network, algorithms where every node conducts local computation based on their own measurement while exchanging the states with neighboring nodes can provide better estimation performance. The measurement data types can lead to different algorithms and convergence results. Reference [6] introduced diffusion least-means-square (LMS) algorithm where the local measurement data is generated from a linear system (which is identical to every node) and real-time. As illustrated in Fig. 1.3, each node performs LMS algorithm [6–9] to estimate the parameter vector in linear regression model. Since the model is identical to every node over the network, exchanging the intermediate estimates with neighboring nodes can improve the estimation performance in terms of mean-square error (MSE) and adaptation speed. This algorithm is referred to *diffusion adaptation* because the adaptation is from LMS adaptive filtering and the diffusion is from exchanging the estimates.

As the measurements data are real-time and the parameter can be diffused over the network, it can be applied for controlling the network itself or a flock of multi-agent [38,39], detection of abnormal sensor nodes [40,41], and many other applications (see, e.g. [42, 43] and the references therein). The diffusion adaptation algorithm was introduced in various scenarios such as noisy wireless communication environments for the diffusion step [44–46] and nonlinearity of the algorithm [41, 47]. Comparison with consensus algorithms, performance analysis, and optimization formulation are addressed in [48–54]. The diffusion adaptation algorithm is different with traditional consensus algorithms such as [15, 31] in that the measurement data is real-time which generates measurement random noise. Since the application is more focused

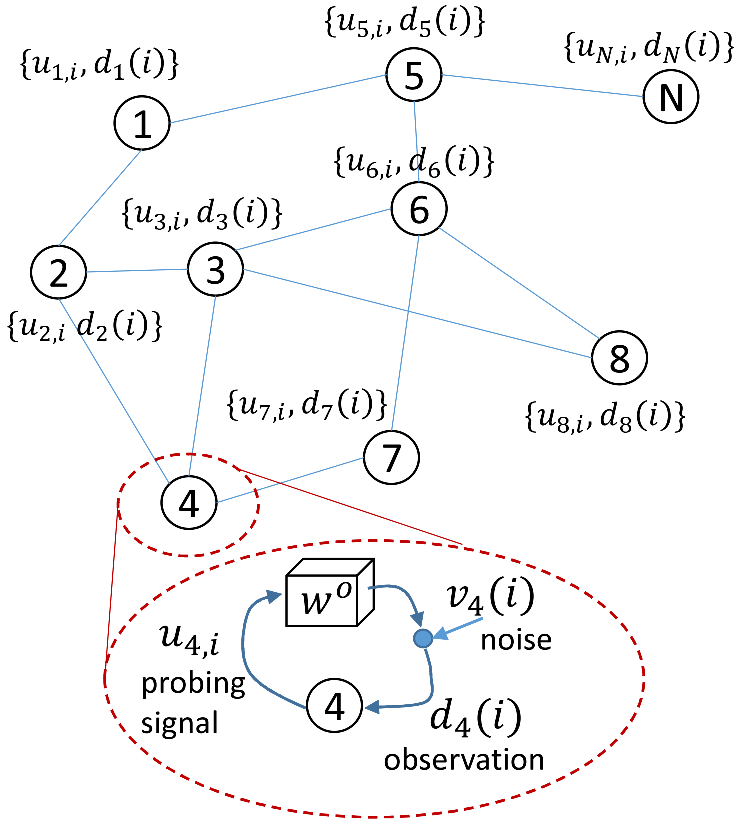
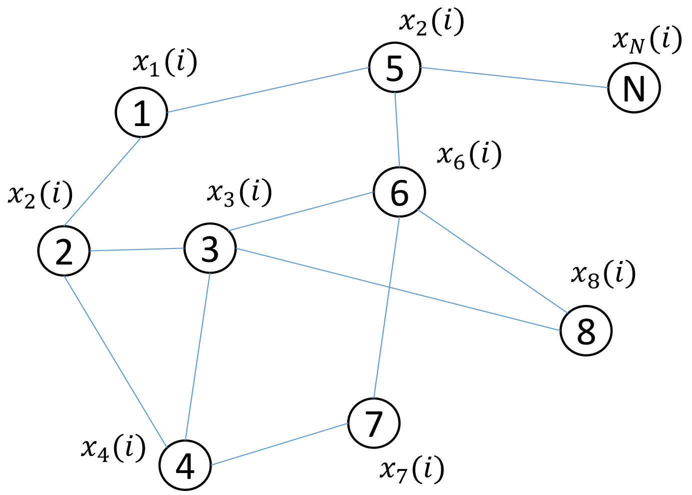


Figure 1.3: Consensus to a Global Parameter over the Network.

on online learning and adaptability from the measurement data, the diffusion adaptation algorithm uses constant step-sizes in order to have capability of adaptation. This does not guarantee convergence to consensus, although the algorithm seeks consensus.

1.3.3.2 Quantile and Mode Estimation

Distributed average (i.e., arithmetic mean) consensus of sensor data can be used in monitoring applications. One example would be to monitor average temperature (or, some other statistical quantities) over a sensor network in remote area. Generally speaking, one might estimate the arithmetic mean of temperature because the mean can represent central tendency of the data. However, arithmetic mean can be



Quantile consensus for p after i iterations:

$$x_n(i) = \inf_{\omega}\left\{\omega : \frac{1}{N}\sum_{n=1}^{N} u(\omega - x_n(0)) \geq p\right\}, \forall n,$$

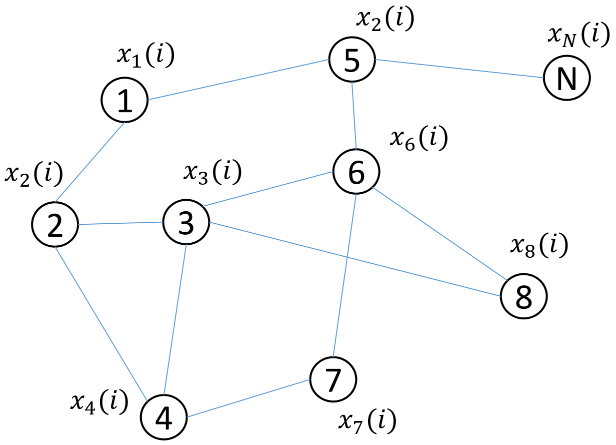
where $0 < p < 1$ and $u(\cdot)$ is the step function.

Figure 1.4: Consensus to a Global Quantile Value from Initial Measurement Data over the Network.

vulnerable to skewness of the distribution. If there are outliers in the measurement data, the mean can be highly biased.

Alternative statistics with less bias are the median and mode. The median represents the mid-point which divides the data into an equal number on either side. The mode is the value that represents the peak of the given distribution. As a generalized metric of median, *quantiles* divide the ranked measurement dataset into subsets of nearly equal sizes. Quantiles are used in various applications. One straightforward example is outliers detection. From a set of measurement data, one may want to eliminate the values higher (or lower) than a certain percentage. With the empirical cumulative distribution function (ECDF), a quantile consensus corresponding to the given ratio p can be achieved. Fig. 1.4 illustrates a quantile consensus estimation. From the initial measurement at each node, after a number iterations i, every node achieves a common value of the quantile. The median can be considered as a special case of quantiles when the sought value is 50% of the measurement data. Trimmed mean is an average of a set of data within a certain range of percentage. Maximum and minimum values can be another examples of quantiles in general point of view. Quantile regression estimates the conditional quantiles, like the conditional mean, of measurement data distribution where the statistics such as mean and variance can change over time. This method has been used in a variety of machine learning [55] as well as statistical applications [56] (and references therein).

The mode can be close to the intuitive understanding of a centrality measure in that it represents the maximum probability of data, in other words, the most probable value. For the past several decades the mode as a measure of central tendency has been extensively used for data analysis in many applications such as bioinformatics [57–60] because it is less biased than other metrics for outliers and contaminations [61, 62]. There are several practical methods to estimate the mode for continuous and discrete data. The basic idea to estimate the mode for the continuous data is to find the *densest* region of data distribution, which can be estimated by either non-parametric [60] or parametric methods [63, 64] whose methods are evaluated in [65]. For the discrete data, one can obtain the histogram first and then find the bin that contains the most frequent value. But the estimated mode depends on the bin size. The existing methods for the mode are based on centralized estimation where all the measurement data need to be collected in a central location. However, the measured sensor data are often unavailable to be transmitted to the central data center. Rather, it



Mode consensus after i iterations:

$$x_n(i) = \underset{\omega}{\text{argmax}} \frac{1}{N} \sum_{n=1}^{N} K_h(\omega, x_n(0)), \forall n,$$

where $K_h(\cdot,\cdot)$ is the Gaussian kernel with bandwidth h.

Figure 1.5: Consensus to a Global Mode of Initial Measurement Data over the Network.

may require distributed calculation to estimate the global mode. Fig. 1.5 illustrates the distributed mode estimation that finds the maximum value of probability density function (PDF) where the PDF is estimated by kernel density estimator (KDE) with Gaussian kernel function. As iteration i increases, every node n achieves the mode, which is the maximum of KDE. For example, consider sensors are deployed in large remote area to monitor natural environment and we wish to estimate the most probable value of precipitation in that area over a certain period of time. Since the communication resources are limited, each sensor shares its own measurement of precipitation with neighborhood. Then after a number of iterative in-network communications, each sensor node converges to the same estimate for the most probable value of precipitation in that area. Since the distribution of precipitation data are not necessarily symmetric, the mode can be considered as a robust measure of central tendency.

1.4 Applications

Many applications for WSNs have been developed in military target tracking and surveillance, natural disaster relief, biomedical health monitoring, and hazardous environment exploration, and seismic sensing [1, 2]. An intensive research has contributed to the problems of reaching consensus among the sensor nodes or control systems [5, 12, 66–75]. Graph network structure metrics such as degree distribution, degree matrix, and network size are useful tools in network analyses and applications. Often it is difficult to learn the graph network structure especially in fully distributed networks [76]. Refs [77–79] studied degree distribution estimation methods for distributed network environments.

A monitoring application example using distributed consensus algorithm is photovoltaic (PV) array monitoring system that can be deployed in remote locations and requires continuous monitoring to secure fault detection and efficient performance. Sensor nodes are capable of monitoring solar arrays in real time to track several parameters including individual module voltage, current, temperature, and irradiance. Continuous monitoring requires sensor connectivity at PV sites, and coordination with the entire network [80–86]. Deployment cost can be reduced by using wireless networked connectivity. PV array monitoring is achieved by sensors mounted on each module, and the data collected from the array have to be aggregated before any decisions are made. Due to the large areas occupied by the PV array, however, transmitting all the data to a fusion center is impractical. In this case, a fully distributed consensus based system can be a viable solution.

1.5 Contributions

This dissertation explores robust distributed parameter estimation based on consensus approach. From sensor measurement, we estimate a certain parameter that is associated with the data. First, we study nonlinear diffusion adaptation scheme with bounded transmission which allows a distributed sensor network to save energy resource. In this work, we consider real-time sensor data and an adaptive learning algorithm. At each iterative time instance, every node processes real-time data by LMS algorithm while the processed data is diffused to its neighboring nodes. Secondly, we study distributed quantile estimation which can be used as a measure of central tendency of sensed data such as trimmed mean, outliers removal, and median as a special case. The measurement data is not real-time in this case but a set of them is given over the network. This estimation method can be used for robust measure of central tendency of data, as the estimated quantile is robust again outliers. Lastly, we study distributed mode estimation which is also used as a metric for central tendency since it can represent the most probable value of data. Two approaches are introduced for the mode estimation. One is mean-shift mode estimation scheme that iteratively updates conditional mean of measurement data where the conditional mean is determined by the measurement data and states of the mode estimates. The other approach is based on kernel density estimator with iteratively updating a grid vector until it converges to the mode. Contributions in more details are summarized below:

- *Nonlinear diffusion adaptation*: we consider nonlinearity of diffusion adaptation scheme in distributed sensor network with bounded transmission. We have shown the nonlinear scheme with performance analysis. It shows that mean-square-deviation (MSD), which represents estimation error from desired a parameter, can be close to the linear case by controlling the nonlinear mapping function parameter with power savings. Convergence in the mean as well as stability for the nonlinear scheme are also provided.

- *Distributed quantile estimation*: we provide a distributed quantile estimation algorithm which only uses local measurement data at each node. Without the knowledge of empirical cumulative distribution function (ECDF) of measurement data, a global quantile is estimated in a distributed way. The algorithm recursively updates the states of estimation, consisting of two-steps at each iteration: one is *local update* based on the measurement data and the current state, and the other is *averaging* the updated states with neighboring nodes. We consider the realistic case of communication links between nodes being corrupted by independent random noise. It is shown that the estimated state sequence is asymptotically unbiased and converges toward the true (sample) quantile in mean-square sense. The two step-size sequences corresponding to the averaging and local update steps result in a mixed-time scale algorithm with proper conditions in order to achieve convergence. We also illustrate potential applications, including distributed estimation of trimmed mean and computation of median, maximum, or minimum values as well as identification of outliers through simulations.

- *Distributed mode estimation I*: we propose a distributed mode estimation method, where each node communicates the states of conditional mean of measurement data with neighboring nodes. The proposed method achieves the global mode by recursively shifting the conditional mean until it converges to a stationary point of the global density function. Simulation results show the robustness and scalability of our approach. The distributed mode estimation is also compared with the results obtained by the centralized methods. As an application we apply the mode estimation algorithm for finding the densest region of the network. Given the geographic location information at each node, the algorithm finds the densest region of sensor deployment using only local exchanges with neighbors.

- *Distributed mode estimation II*: we also study another approach for distributed mode estimation, which is based on kernel density estimator and a grid vector. At each point on the grid vector, kernel densities are estimated after a certain number of iterations. The grid points are updated by reducing the distance between the points, and are converged to a consensus after a number of iterations. This method is useful even when the data distribution is spatially correlated in the network. Numerical experiments are provided to demonstrate the method.

1.6 Outline of Dissertation

The outline of dissertation is following. In chapter 2, we propose a nonlinear diffusion adaptation scheme using bounded transmission function that can lead to energy efficient diffusion adaptation algorithm for wireless distributed sensor network. Chapter 3 describes the distributed quantile estimation which shows convergence toward a quantile value of empirical cumulative distribution function. In chapter 4 and 5, we propose distributed mode estimation methods based on mean-shift algorithm and grid vector scheme respectively. Finally we conclude this dissertation in chapter 6.

Chapter 2

NONLINEAR DIFFUSION ADAPTATION WITH BOUNDED TRANSMISSION

Fully distributed networks, where nodes are connected only with their neighbors, perform decentralized processing to achieve a global objective by relying on local information. One such objective is the identification of model parameters of physical phenomena observable over the entire network. All nodes in the network have access to their own observed data and wish to estimate the model common parameters, as described in Fig. 2.1. When real-time data are observed at each node, it is possible for individual nodes to locally learn and estimate the parameters using adaptive learning algorithms [7–11]. Since the nodes have the common objective of parameter estimation, cooperation with neighbors during the adaptive learning process is expected to provide some benefits in fully distributed estimation.

Distributed adaptive estimation algorithms with such cooperative processing have been developed by strategies that generally consist of two steps: diffusion and adaptation [6, 42, 46, 51]. Individual nodes communicate with neighbors to learn common parameters by combining their local estimates with those of neighbors, while each node performs least-mean-square (LMS) based adaptive learning. The diffusion step is conducted by combining the local estimates. Linear diffusion LMS is formulated and its performance is analyzed in [6]. The diffusion LMS of [6] is generalized in [51] where measurement data are exchanged as well as parameter estimates. When the linear diffusion is performed with wireless communications, noisy links are considered in [46, 52, 53], and fading channels in [44]. Further variations of linear diffusion strategies in different situations are introduced in [42] and references therein.

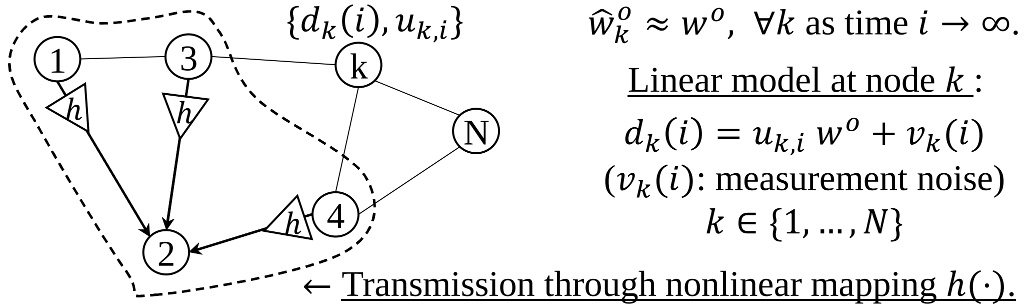

$\{d_k(i), u_{k,i}\}$
1
3
k
N
4
2
h
h
h
$\widehat{w}_k^o \approx w^o, \ \forall k$ as time $i \to \infty$.
Linear model at node k :
$d_k(i) = u_{k,i}\, w^o + v_k(i)$
($v_k(i)$: measurement noise)
$k \in \{1, \ldots, N\}$
← Transmission through nonlinear mapping $h(\cdot)$.

Figure 2.1: Nonlinear Diffusion Update at Node 2 with Its Neighbors.

In this chapter, we consider *nonlinearity* in diffusion adaptation strategies. In [41], an error nonlinearity in each node's adaptation was applied due to impulsive measurement noise. However, we are interested in nonlinearity of diffusion updates among the nodes, which occurs when local intermediate estimates are diffused with those of neighboring nodes through bounded transmissions. Fig. 2.1 illustrates a situation where a node combines local estimates from its neighbors through *nonlinear* mapping functions. In practice, nonlinearities arise at transmit power amplifiers (PAs) when the amplitude in the transmitter is relatively high [30, 87]. Moreover, if every node maps its intermediate estimate through a bounded function to constrain the peak transmit power, the magnitude of the transmitted signal is always bounded and transmit power can be saved. We propose *nonlinear diffusion* strategies using sigmoidal functions to model such bounded transmissions. We study the convergence properties of nonlinear diffusion adaptation. Numerical results show that the proposed algorithm can be close to the linear case in terms of the mean-square-deviation (MSD), with the added benefits of power savings.

The rest of this chapter is organized as follows. Section 2.1 introduces the system model and linear diffusion adaptation. In Section 2.2 nonlinear diffusion with bounded transmission is proposed. Performance analysis is provided in Section 2.3.

Numerical results in Section 2.4 support the contributions. Finally, concluding remarks are provided in Section 2.5.

2.1 System Model and Existing Solutions

Consider a distributed network that consists of a set of nodes $\mathbb{N} = \{1, 2, \ldots, N\}$. We denote the set of neighbors of node n by $\mathbb{N}_n$ where $k = 1, 2, \ldots, N$. Each node n observes the temporal realization of zero-mean random processes $\{y_n(i), \mathbf{x}_n(i)\}$ where $y_n(i)$ and $\mathbf{x}_n(i) = [x_{n1}(i), \ldots, x_{nm}(i), \ldots, x_{nM}(i)]$ are a real-valued scalar and a row regression vector of size $1 \times M$, respectively, at time instants $i \geq 0$. The measurement data sequence $\{y_n(i), \mathbf{x}_n(i)\}_{i \geq 0}$ are related linearly to an unknown real-valued $M \times 1$ vector $\boldsymbol{w}^o$:

$$y_n(i) = \mathbf{x}_n(i)\boldsymbol{w}^o + v_n(i) \tag{2.1}$$

where $v_n(i)$ denotes measurement noise which is a real-valued zero-mean Gaussian random process with variance $\sigma^2_{v,n}$. It is assumed the random variable $v_n(i)$ and $v_l(j)$ are temporally and spatially independent for $i \neq j$ and $n \neq l$.

The objective is to estimate the model parameter vector $\boldsymbol{w}^o$, which is unknown initially, by distributed adaptive estimation. Since $\boldsymbol{w}^o$ is assumed the same at every node, local cooperation with neighbors becomes beneficial when there is no centralized control. Every node n continuously exchanges its intermediate estimate of $\boldsymbol{w}^o$ with neighboring nodes, while conducting an adaptive learning algorithm. Thus the algorithm consists of two steps: *diffusion* and *adaptation*. The order of the steps can be reversed. We focus on the *diffusion* step where we propose *nonlinearity* with bounded transmission, while adopting LMS based algorithm with exchanging intermediate estimate of $\boldsymbol{w}^o$ for the adaptation step.

We first review the linear case [42] to contrast it with our nonlinear approach. Diffusion process is conducted by combining each node's intermediate estimate with its neighbor estimates. Let $\boldsymbol{\omega}_n(i)$ be the intermediate estimate of size $M \times 1$ at node n at time i. Let $\boldsymbol{\psi}_n(i)$ be the diffused estimate of $\boldsymbol{\omega}_n(i)$ and $\{\boldsymbol{\omega}_l(i)\}_{l\in\mathbb{N}_n}$ at i with coefficients $\{a_{nl}\}_{n\neq l, l\in\mathbb{N}_n}$ where a_{nl} denotes a combining weight from l to n. The linear diffusion can be defined by

$$\boldsymbol{\psi}_n(i) = a_{nn}\boldsymbol{\omega}_n(i) + \sum_{l\in\mathbb{N}_n} a_{nl}\boldsymbol{\omega}_l(i) \tag{2.2}$$

where a_{nn}, a_{nl} satisfy the following conditions [6,42]:

$$a_{nl} \geq 0, \forall n, l, \quad a_{nn} + \sum_{l\in\mathbb{N}_n} a_{nl} = 1, \quad \text{and} \quad a_{nl} = 0 \;\; \text{if} \;\; l \notin \mathbb{N}_n. \tag{2.3}$$

There are many well-known combining rules such as Laplacian [4,88], nearest neighbor [68], Metropolis [4,89,90], and maximum-degree [15]. Finding the combining weights is out of scope of this chapter. Instead, we follow the form derived for diffusion adaptation strategies in [42,49,51]:

$$a_{nl} = \begin{cases} 1 - \eta_n \sum_{l\in\mathbb{N}_n} b_{nl}, & \text{if } l = n \\ \eta_n b_{nl}, & \text{if } l \in \mathbb{N}_n \\ 0, & \text{if } l \notin \mathbb{N}_n \end{cases}, \tag{2.4}$$

where η_n is a parameter selected to satisfy the conditions in (2.3). By properly selecting η_n and b_{nl}, one can find the equivalent combining rules introduced above. By substituting (2.4) into (2.2), the linear diffusion adaptation is expressed as

$$\boldsymbol{\psi}_n(i) = \boldsymbol{\omega}_n(i) - \eta_n \sum_{l\in\mathbb{N}_n} b_{nl}\big(\boldsymbol{\omega}_n(i) - \boldsymbol{\omega}_l(i)\big), \tag{2.5}$$

$$\boldsymbol{\omega}_n(i+1) = \boldsymbol{\psi}_n(i) + \mu_n \mathbf{x}_n^T(i)\big(y_n(i) - \mathbf{x}_n(i)\boldsymbol{\psi}_n(i)\big), \tag{2.6}$$

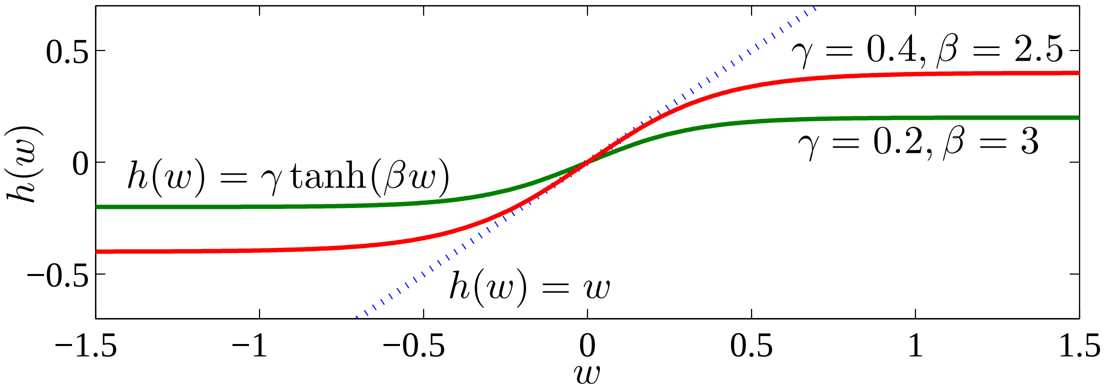


Figure 2.2: Linear (Unbounded) and Nonlinear (Bounded) Functions.

where μ_n is a step-size parameter at node n, $\boldsymbol{\omega}_n(0) = \mathbf{0}$, and $i \geq 0$. Note that $y_n(i)$ and $\mathbf{x}_n(i)$ are defined in (2.1). The two-step updates of (2.5) and (2.6) are called the combine-then-adapt (CTA) algorithm [6]. The order of combine step in (2.5) and adapt step in (2.6) can be reversed, which is given by

$$\boldsymbol{\psi}_n(i) = \boldsymbol{\omega}_n(i) + \mu_n \mathbf{x}_n^T(i)\big(y_n(i) - \mathbf{x}_n(i)\boldsymbol{\omega}_n(i)\big), \tag{2.7}$$

$$\boldsymbol{\omega}_n(i+1) = \boldsymbol{\psi}_n(i) - \eta_n \sum_{l \in \mathbb{N}_n} b_{nl}\big(\boldsymbol{\psi}_n(i) - \boldsymbol{\psi}_l(i)\big), \tag{2.8}$$

where $i \geq 0$. We primarily deal with CTA approach to propose the nonlinear diffusion adaptation. However, the adapt-then-combine (ATC) approach [42,46,51] will be also presented in numerical results to show that the nonlinearity does not affect the order of the two-step updates shown in (2.5) and (2.6).

2.2 Nonlinear Diffusion with Bounded Transmission

Let each node l transmit its intermediate estimate $\boldsymbol{\omega}_l(i)$ by mapping it through a function $h(\cdot)$ which can either be linear or nonlinear. The diffusion in (2.5) can be modified as

$$\boldsymbol{\psi}_n(i) = \boldsymbol{\omega}_n(i) - \eta_n \sum_{l \in \mathbb{N}_n} b_{nl}\Big[h\big(\boldsymbol{\omega}_n(i)\big) - h\big(\boldsymbol{\omega}_l(i)\big)\Big] \tag{2.9}$$

where the element-wise function $h(\cdot)$ can be *nonlinear* as in the following possible scenarios: when there is nonlinearity in power amplifier (PA) at transmitter of wireless sensor nodes [30, 87]; when a maximum value of $\boldsymbol{\omega}_l(i)$ is enforced for energy-efficient amplify-and-forward (AF) wireless sensor networks [27]. Sigmoidal functions, which are smooth one-to-one mapping functions with finite upper and lower bounds, are suitable for modeling the nonlinearities [27]. Typically, input-output power characteristics in PA show that output power is saturated to an upper bound as input power increases, whereas the amplifier performs linearly when input power is relatively low [30, 87]. Moreover, such sigmoidal functions make the magnitude of each node's transmission bounded, expecting energy-efficiency of resource-constrained networks. Fig. 2.2 illustrates both of linear and nonlinear transmit functions. We see that the nonlinear functions are bounded, whereas the linear function can be unbounded. The following assumption is made on the nonlinear function $h(\cdot)$:

Assumption 1 *$h(\cdot)$ is a strictly increasing odd function satisfying $|h(x)| \leq \kappa|x|$ for some $\kappa > 0$, and all $x \in \mathbb{R}$.*

When the transmission links are noisy, the transmitted values experience additive random noise. We assume zero-mean Gaussian random noise is added onto the links. Then the nonlinear diffusion in (2.9) can be extended for the noisy links:

$$\boldsymbol{\psi}_n(i) = \boldsymbol{\omega}_n(i) - \eta_n \sum_{l \in \mathbb{N}_n} b_{nl} \Big[h\big(\boldsymbol{\omega}_n(i)\big) - h\big(\boldsymbol{\omega}_l(i)\big) - \boldsymbol{\xi}_{nl}(i) \Big] \tag{2.10}$$

where $\boldsymbol{\xi}_{nl}(i)$ is a Gaussian random vector of size $M \times 1$.

The *nonlinear diffusion* adaptation algorithm with the bounded transmissions is described in Algorithm 1 where LMS based adaptive learning algorithm is considered for the adaptation step. Algorithm 2 describes the ATC approach that was implemented by reversing the order of *nonlinear diffusion* and adaptation in CTA approach.

Algorithm 1 Nonlinear Diffusion Adaptation with Bounded Transmission over a Network (Nonlinear CTA)

```
Initialization: $\boldsymbol{\omega}_n(0)$, $\gamma_n$, $\beta_n$, $\{a_{nl}\}$, $\mu_n$, $\forall n$ and $l \in \mathbb{N}_n$.
for $i \geq 0$ do
    Bounded transmission: for every transmitting node $l$,
    $h(\boldsymbol{\omega}_l(i)) = \gamma_l \tanh\big(\beta_l \boldsymbol{\omega}_l(i)\big)$
    Nonlinear diffusion: with $\{a_{nl}\}$ of (2.4), repeat $\forall n$
    $\boldsymbol{\psi}_n(i) = \boldsymbol{\omega}_n(i) - \eta_n \sum_{l \in \mathbb{N}_n} b_{nl} \Big[ h\big(\boldsymbol{\omega}_n(i)\big) - h\big(\boldsymbol{\omega}_l(i)\big) - \boldsymbol{\xi}_{nl}(i) \Big]$
    Adaptation: repeat $\forall n$
    $\boldsymbol{\omega}_n(i+1) = \boldsymbol{\psi}_n(i) + \mu_n \mathbf{x}_n^T(i) \Big( y_n(i) - \mathbf{x}_n(i) \boldsymbol{\psi}_n(i) \Big)$
    Time instant update: $i \rightarrow i + 1$.
end for
When $\boldsymbol{\xi}_{nl}(i) = \mathbf{0}$, the link $l$ to $n$ is noise-free at time $i$.
```

Algorithm 2 Nonlinear Diffusion Adaptation with Bounded Transmission over a Network (Nonlinear ATC)

```
Initialization: $\boldsymbol{\omega}_n(0)$, $\gamma_n$, $\beta_n$, $\{a_{nl}\}$, $\mu_n$, $\forall n$ and $l \in \mathbb{N}_n$.
for $i \geq 0$ do
    Adaptation: repeat $\forall n$
    $\boldsymbol{\psi}_n(i) = \boldsymbol{\omega}_n(i) + \mu_n \mathbf{x}_n^T(i) \Big( y_n(i) - \mathbf{x}_n(i) \boldsymbol{\omega}_n(i) \Big)$
    Nonlinear diffusion: with $\{a_{nl}\}$ of (2.4), repeat $\forall n$
    $\boldsymbol{\omega}_n(i+1) = \boldsymbol{\psi}_n(i) - \eta_n \sum_{l \in \mathbb{N}_n} b_{nl} \Big[ h\big(\boldsymbol{\psi}_n(i)\big) - h\big(\boldsymbol{\psi}_l(i)\big) - \boldsymbol{\xi}_{nl}(i) \Big]$
    Bounded transmission: for every transmitting node $l$,
    $h(\boldsymbol{\psi}_l(i)) = \gamma_l \tanh\big(\beta_l \boldsymbol{\psi}_l(i)\big)$
    Time instant update: $i \rightarrow i + 1$.
end for
When $\boldsymbol{\xi}_{nl}(i) = \mathbf{0}$, the link $l$ to $n$ is noise-free at time $i$.
```

2.3 Performance Analysis

In this section we study convergence properties of the nonlinear diffusion adaptation comparing to the linear case. The conventional analysis of error recursion [6,8,9,42] provides a powerful tool for evaluating the convergence behavior of the estimate. We show how the estimate $\boldsymbol{\omega}_n(i)$ approaches the common parameter $\boldsymbol{\omega}^o$ by evaluating the evolution of error $\widetilde{\boldsymbol{\omega}}_n(i) \triangleq \boldsymbol{\omega}^o - \boldsymbol{\omega}_n(i)$ over time i over the network. Note that $\widetilde{\boldsymbol{\omega}}_n(i) \approx \mathbf{0}$ means $\boldsymbol{\omega}_n(i) \approx \boldsymbol{\omega}^o$. First, we derive an error recursion for the nonlinear diffusion adaptation. Then we study the convergence properties of it in the mean and the mean-square sense. We primarily deal with the case of noisy transmission links in Algorithm 1, but one can easily see the performance of the noise-free case by setting $\boldsymbol{\xi}_{nl}(i) = \mathbf{0}$ for all l, n, and i.

Assuming the network has N nodes and each node estimates vector $\boldsymbol{\omega}_n(i)$ of size $M \times 1$, the updates for Algorithm 1 can be expressed in block vectors and matrices simultaneously for all nodes:

$$\boldsymbol{\psi}(i) = \boldsymbol{\omega}(i) - \eta \mathcal{L}^T h\big(\boldsymbol{\omega}(i)\big) - \eta\, \boldsymbol{\xi}(i), \tag{2.11}$$

$$\boldsymbol{\omega}(i+1) = \boldsymbol{\psi}(i) + \mu\, \mathbf{x}^T(i)\big(\mathbf{y}(i) - \mathbf{x}(i)\boldsymbol{\psi}(i)\big), \tag{2.12}$$

where η and μ are assumed constants ($\eta = \eta_n$, $\mu = \mu_n$, $\forall n$), and

$$
\begin{aligned}
\boldsymbol{\psi}(i) &\triangleq \left[\boldsymbol{\psi}_1^T(i), \ldots, \boldsymbol{\psi}_N^T(i)\right]^T, \\
\boldsymbol{\omega}(i) &\triangleq \left[\boldsymbol{\omega}_1^T(i), \ldots, \boldsymbol{\omega}_N^T(i)\right]^T, \\
h\big(\boldsymbol{\omega}(i)\big) &\triangleq \left[h^T\big(\boldsymbol{\omega}_1(i)\big), \ldots, h^T\big(\boldsymbol{\omega}_N(i)\big)\right]^T, \\
\boldsymbol{\xi}(i) &= \left[\sum_{l\in\mathbb{N}_1} b_{l1}\boldsymbol{\xi}_{l1}^T(i), .., \sum_{l\in\mathbb{N}_N} b_{lN}\boldsymbol{\xi}_{lN}^T(i)\right]^T, \\
\boldsymbol{\mathcal{L}} &\triangleq \mathbf{L} \otimes \mathbf{I}_M = (\mathbf{D} - \mathbf{B}) \otimes \mathbf{I}_M, \\
\mathbf{D} &\triangleq \operatorname{diag}\left\{\sum_{l\in\mathbb{N}_1} b_{l1}, \ldots, \sum_{l\in\mathbb{N}_N} b_{lN}\right\}, \\
\mathbf{x}(i) &\triangleq \operatorname{diag}\left\{\mathbf{x}_1(i), \ldots, \mathbf{x}_N(i)\right\}, \\
\mathbf{y}(i) &\triangleq \left[y_1(i), \ldots, y_N(i)\right]^T.
\end{aligned}
$$

Note that $\otimes$ denotes Kronecker product, $h(\cdot)$ is an element-wise function, $\mathbf{I}_M$ is an identity matrix of size $M \times M$, and b_{nl} is an element of $N \times N$ matrix $\mathbf{B}$. The dimensions of block vectors and matrices are summarized in Table 2.1.

Table 2.1: Dimensions of Block Vectors and Matrices.

vectors and matrices	dimensions
$\boldsymbol{\psi}(i)$, $\boldsymbol{\omega}(i)$, $h\big(\boldsymbol{\omega}(i)\big)$, $\boldsymbol{\xi}(i)$, $\boldsymbol{\omega}^{(o)}$	$MN \times 1$
$\boldsymbol{\mathcal{L}}$, $\boldsymbol{\mathcal{I}}$, $\boldsymbol{\mathcal{R}}(i)$	$MN \times MN$
$\mathbf{D}$	$N \times N$
$\mathbf{x}(i)$	$N \times MN$
$\mathbf{y}(i)$	$N \times 1$

Let $\boldsymbol{\omega}^{(o)}$ be a block vector of $M \times 1$ vector $\boldsymbol{\omega}^o$, defined as

$$\boldsymbol{\omega}^{(o)} \triangleq \left[\boldsymbol{\omega}^{oT}, \ldots, \boldsymbol{\omega}^{oT}\right]^T. \tag{2.13}$$

By substituting $\boldsymbol{\psi}(i)$ of (2.11) into (2.12) and subtracting $\boldsymbol{\omega}^{(o)}$ from both sides of (2.12), we obtain the error recursion for the nonlinear diffusion strategies:

$$\widetilde{\boldsymbol{\omega}}(i+1) = \big(\boldsymbol{\mathcal{I}} - \mu\boldsymbol{\mathcal{R}}(i)\big)\Big(\widetilde{\boldsymbol{\omega}}(i) - \eta\boldsymbol{\mathcal{L}}^T\widetilde{h}\big(\boldsymbol{\omega}(i)\big)\Big) + \eta\big(\boldsymbol{\mathcal{I}} - \mu\boldsymbol{\mathcal{R}}(i)\big)\boldsymbol{\xi}(i) - \mu\,\mathbf{s}(i) \quad (2.14)$$

where

$$\begin{aligned}
\boldsymbol{\mathcal{R}}(i) &\triangleq \mathbf{x}^T(i)\mathbf{x}(i),\\
\widetilde{\boldsymbol{\omega}}(i) &\triangleq \boldsymbol{\omega}^{(o)} - \boldsymbol{\omega}(i),\\
\widetilde{h}\big(\boldsymbol{\omega}(i)\big) &\triangleq \boldsymbol{\omega}^{(o)} - h\big(\boldsymbol{\omega}(i)\big),\\
\mathbf{s}(i) &\triangleq \Big[\big(\mathbf{x}_1^T(i)v_1(i)\big)^T, \ldots, \big(\mathbf{x}_N^T(i)v_N(i)\big)^T\Big]^T.
\end{aligned}$$

Based on the error recursion of (2.14) with fixed μ, η, and $\boldsymbol{\mathcal{L}}$, we utilize $h(\cdot)$ to compare the convergence properties of the linear and the nonlinear diffusion adaptation strategies. Linear approximation of $h(\cdot)$ makes the error recursion simple so that we can easily compare the linear and the nonlinear cases in the same metric. For the linear diffusion strategies, we assume that node n transmits $\boldsymbol{\omega}_n(i)$ through $h\big(\boldsymbol{\omega}_n(i)\big) = \kappa\,\boldsymbol{\omega}_n(i)$ at time i where κ is a positive constant. Let $\boldsymbol{\mathcal{Q}} \triangleq \big(\boldsymbol{\mathcal{I}} - \mu\boldsymbol{\mathcal{R}}(i)\big)\big(\boldsymbol{\mathcal{I}} - \eta\kappa\boldsymbol{\mathcal{L}}^T\big)$. Noting that $(\boldsymbol{\mathcal{I}} - \eta\boldsymbol{\mathcal{L}}^T)\boldsymbol{\omega}^{(o)} = (\boldsymbol{\mathcal{I}} - \eta\kappa\boldsymbol{\mathcal{L}}^T)\boldsymbol{\omega}^{(o)} = \boldsymbol{\omega}^{(o)}$, we simplify the error recursion of (2.14) to the linear case:

$$\widetilde{\boldsymbol{\omega}}(i+1) = \boldsymbol{\mathcal{Q}}\widetilde{\boldsymbol{\omega}}(i) + \eta\big(\boldsymbol{\mathcal{I}} - \mu\boldsymbol{\mathcal{R}}(i)\big)\boldsymbol{\xi}(i) - \mu\,\mathbf{s}(i). \quad (2.15)$$

The nonlinear element-wise bounded function can be expressed as

$$h\big(\boldsymbol{\omega}_n(i)\big) = \gamma\tanh\big(\beta\boldsymbol{\omega}_n(i)\big) \triangleq \mathbf{E}_n(i)\boldsymbol{\omega}_n(i) \quad (2.16)$$

with an $M \times M$ diagonal matrix $\mathbf{E}_n(i) \triangleq \mathrm{diag}\{\epsilon_{n1}(i), ..., \epsilon_{nM}(i)\}$ such that $\epsilon_{nm}(i) \leq \kappa$ for all n, i, and $m \in \{1, \ldots, M\}$, according to Assumption 1. Define

$$\boldsymbol{\mathcal{P}}(i) \triangleq (\boldsymbol{\mathcal{I}} - \mu\boldsymbol{\mathcal{R}}(i))\big(\boldsymbol{\mathcal{I}} - \eta\boldsymbol{\mathcal{L}}^T\boldsymbol{\mathcal{E}}(i)\big) \quad (2.17)$$

where $\boldsymbol{\mathcal{E}}(i) = \mathbf{E}_n(i) \otimes \mathbf{I}_N$. Similar to (2.15), the error recursion of (2.14) for the nonlinear case can be simplified as

$$\widetilde{\boldsymbol{\omega}}(i+1) = \boldsymbol{\mathcal{P}}(i)\widetilde{\boldsymbol{\omega}}(i) + \eta\big(\boldsymbol{\mathcal{I}} - \mu\boldsymbol{\mathcal{R}}(i)\big)\boldsymbol{\xi}(i) - \mu\,\mathbf{s}(i) \tag{2.18}$$

for $i \geq 0$.

2.3.1 Convergence in the Mean

For the error estimate $\widetilde{\boldsymbol{\omega}}(i)$ to converge to zero, the error recursions of (2.15) and (2.18) should be stable as $i \to \infty$. We compare the stabilities of the two diffusion adaptations in the mean sense. Taking expectation on both sides of (2.15), we have

$$E\big[\widetilde{\boldsymbol{\omega}}(i+1)\big] = \boldsymbol{\mathcal{Q}}E\big[\widetilde{\boldsymbol{\omega}}(i)\big] \tag{2.19}$$

where $\boldsymbol{\mathcal{Q}} \triangleq \big(\boldsymbol{\mathcal{I}} - \mu\boldsymbol{\mathcal{R}}(i)\big)\left(\boldsymbol{\mathcal{I}} - \eta\kappa\boldsymbol{\mathcal{L}}^T\right)$. We assumed spatial and temporal independence $\boldsymbol{\mathcal{R}}(i)$ and $\widetilde{\boldsymbol{\omega}}(i)$, as is common in traditional adaptive learning algorithms [6,8,9,42]. For the nonlinear case, however, such an assumption is not valid because $\boldsymbol{\mathcal{E}}(i)$ is dependent on $\widetilde{\boldsymbol{\omega}}(i)$. Instead, we utilize Assumption 1 of the bounded function $h(\cdot)$. For the purpose of evaluation, we can select a positive constant κ' such that

$$0 < \epsilon_{nm}(i) \leq \kappa' \leq \kappa, \tag{2.20}$$

where $n \in \{1, \ldots, N\}$, $m \in \{1, \ldots, M\}$, and $i \geq 0$. When κ' from (2.20) is substituted for $\epsilon_{nm}(i)$ in $\boldsymbol{\mathcal{P}}(i)$, taking expectation of both sides of (2.18), we have

$$E\big[\widetilde{\boldsymbol{\omega}}(i+1)\big] = \boldsymbol{\mathcal{P}}'(i)E\big[\widetilde{\boldsymbol{\omega}}(i)\big] \tag{2.21}$$

where $\boldsymbol{\mathcal{P}}'(i) = \Big(\boldsymbol{\mathcal{I}} - \mu E\big[\boldsymbol{\mathcal{R}}(i)\big]\Big)\left(\boldsymbol{\mathcal{I}} - \eta\kappa'\boldsymbol{\mathcal{L}}^T\right)$. To be stable in the mean, all eigenvalues $\lambda(\cdot)$ of $\boldsymbol{\mathcal{Q}}$ and $\boldsymbol{\mathcal{P}}'(i)$ must satisfy $\big|\lambda(\boldsymbol{\mathcal{Q}})\big| < 1$ and $\big|\lambda\big(\boldsymbol{\mathcal{P}}'(i)\big)\big| < 1$. When the step-size μ is sufficiently small so that $\Big|\lambda_{\max}\Big(\boldsymbol{\mathcal{I}} - \mu E\big[\boldsymbol{\mathcal{R}}(i)\big]\Big)\Big| < 1$, and for the right stochastic matrices $\boldsymbol{\mathcal{I}} - \eta\kappa\boldsymbol{\mathcal{L}}^T$ and $\boldsymbol{\mathcal{I}} - \eta\kappa'\boldsymbol{\mathcal{L}}^T$, the recursions of (2.19) and (2.21) are stable as $i \to \infty$ because

$$\left|\lambda_{\max}(\boldsymbol{\mathcal{Q}})\right| \leq \left|\lambda_{\max}\big(\boldsymbol{\mathcal{P}}'(i)\big)\right| \leq \left|\lambda_{\max}\Big(\boldsymbol{\mathcal{I}} - \mu E\big[\boldsymbol{\mathcal{R}}(i)\big]\Big)\right| < 1 \tag{2.22}$$

for all $i \geq 0$. Moreover, for $0 < \epsilon_{nm}(i) \leq \kappa'$, the spectral radius of $\boldsymbol{\mathcal{P}}(i)$ such that

$$\left|\lambda_{\max}\big(\boldsymbol{\mathcal{P}}'(i)\big)\right| \leq \left|\lambda_{\max}\big(\boldsymbol{\mathcal{P}}(i)\big)\right| < \left|\lambda_{\max}\Big(\boldsymbol{\mathcal{I}} - \mu E\big[\boldsymbol{\mathcal{R}}(i)\big]\Big)\right| \tag{2.23}$$

ensures the stability of (2.18) in the mean. Therefore, when the error recursion for the linear diffusion adaptation is stable with μ and κ, the *nonlinear diffusion* adaptation using bounded transmissions is also stable and its estimate $\boldsymbol{\omega}_n(i)$ converges to $\boldsymbol{\omega}^o$ in the mean sense, as $i \to \infty$.

2.3.2 Mean-square Stability

To ensure the stability of the proposed algorithm, we need to further investigate the mean-square stability because converging to $\boldsymbol{\omega}^o$ in the mean may cause large variations around $\boldsymbol{\omega}^o$. The error variance recursion can be derived from (2.18):

$$E\Big[\big|\big|\widetilde{\boldsymbol{\omega}}(i+1)\big|\big|^2\Big] = E\Big[\big|\big|\boldsymbol{\mathcal{P}}(i)\widetilde{\boldsymbol{\omega}}(i)\big|\big|^2\Big] + \eta^2 E\Big[\big|\big|\left(\boldsymbol{\mathcal{I}} - \mu\boldsymbol{\mathcal{R}}(i)\right)\boldsymbol{\xi}(i)\big|\big|^2\Big] + \mu^2 E\Big[\big|\big|\mathbf{s}(i)\big|\big|^2\Big] \tag{2.24}$$

where the cross-terms were canceled because of the independence assumption and zero-mean Gaussian random noises. By substituting κ' for $\epsilon_{nm}(i)$ in $\boldsymbol{\mathcal{P}}(i)$ again, the mean-square analysis is simplified. The sufficiently small step-size parameter μ ensures the mean-square stability of (2.18) because of the stability that is given by

$$\lambda_{\max}(\boldsymbol{\mathcal{Q}})^2 \leq \lambda_{\max}\big(\boldsymbol{\mathcal{P}}'(i)\big)^2 \leq \lambda_{\max}\Big(\boldsymbol{\mathcal{I}} - \mu E\big[\boldsymbol{\mathcal{R}}(i)\big]\Big)^2 < 1. \tag{2.25}$$

Since $0 < \epsilon_{nm}(i) \leq \kappa'$ for all n, i, and m, the spectral radius of $\boldsymbol{\mathcal{P}}(i)^T\boldsymbol{\mathcal{P}}(i)$ such that

$$\lambda_{\max}\big(\boldsymbol{\mathcal{P}}'(i)\big)^2 \leq \lambda_{\max}\big(\boldsymbol{\mathcal{P}}(i)\big)^2 < \lambda_{\max}\Big(\boldsymbol{\mathcal{I}} - \mu E\big[\boldsymbol{\mathcal{R}}(i)\big]\Big)^2 \tag{2.26}$$

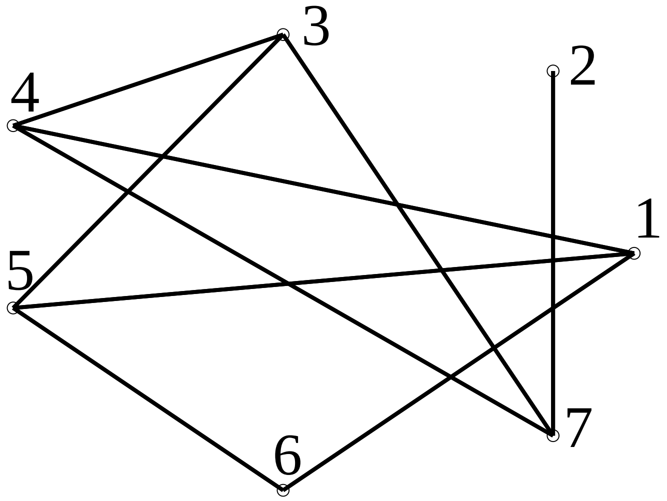


Figure 2.3: Network Topology.

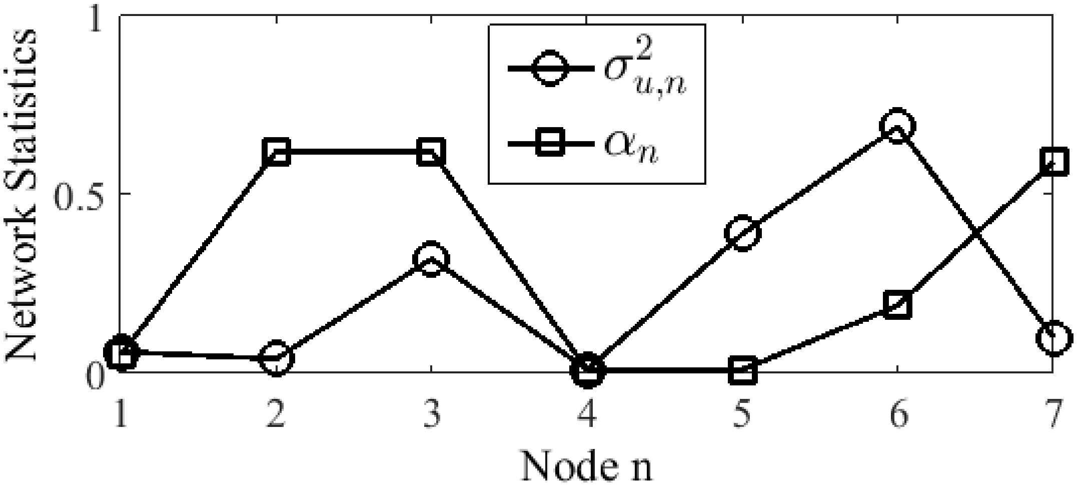


Figure 2.4: Generated Regression Data $\boldsymbol{u}_{n,i}$ Statistics: $\sigma^2_{u,n}$ and α^2_n Denote the Variance of $\boldsymbol{u}_{n,i}$ and Correlation Index at Node n Respectively; $N = 7$.

ensures the mean-square stability of (2.18).

However, the convergence rate of $E\left[||\widetilde{\boldsymbol{\omega}}(i)||^2\right]$ towards its steady-state estimate in the nonlinear case can be slower than (or equal to) the linear diffusion adaptation strategies because the eigenmodes in the nonlinear error variance recursion of (2.24) can be larger than (or equal to) those of the linear case when $\epsilon_{nm}(i) \leq \kappa' \leq \kappa$ for all n, i, and m.

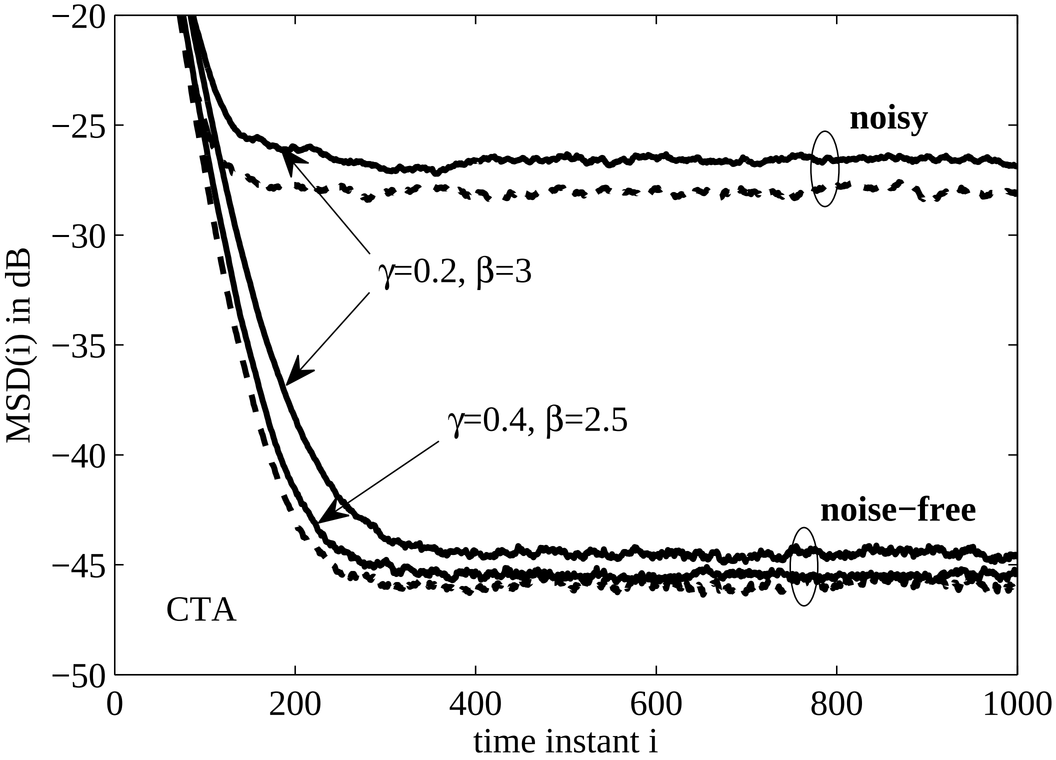


Figure 2.5: Mean-square-deviation (MSD) for CTA. Nonlinear Diffusion Adaptation with Bounded Function Can Be Close to the Linear Case by Balancing the Parameter γ and β, While the Peak Transmit Power γ^2 is Always Bounded.

2.4 Simulation Results

For the simulation results, $N = 7$ nodes are considered. Each node estimates an unknown vector $\boldsymbol{\omega}^o = [1,\, 1,\, 1,\, 1,\, 1]^T/\sqrt{M}$ where $M = 5$. The regression data $\mathbf{x}_n(i)$ in (2.1) is generated by Gaussian 1-Markov process with its correlation function $r_n(m) = \sigma^2_{\mathbf{x},n}\alpha_n^{|m|}$, $m = 0, \ldots, M-1$ where α_n is the correlation index at node n. Each node n obtains $\{y_n(i),\, \mathbf{x}_n(i)\}$ at time i where $y_n(i)$ is the output of linear model in (2.1). Fig. 2.3 illustrates the network topology and the data statistics. The measurement noise variances are set to $\sigma^2_v = 10^{-4} \times [1,\, 3,\, 8,\, 2,\, 1,\, 7,\, 5]^T$. When noisy links are considered, the noise variance σ^2_ξ is set to 1×10^{-4} for every link. The Laplacian rule for (2.4) is used by setting the combining weights $\eta_n = 1/(\text{max. degree})$

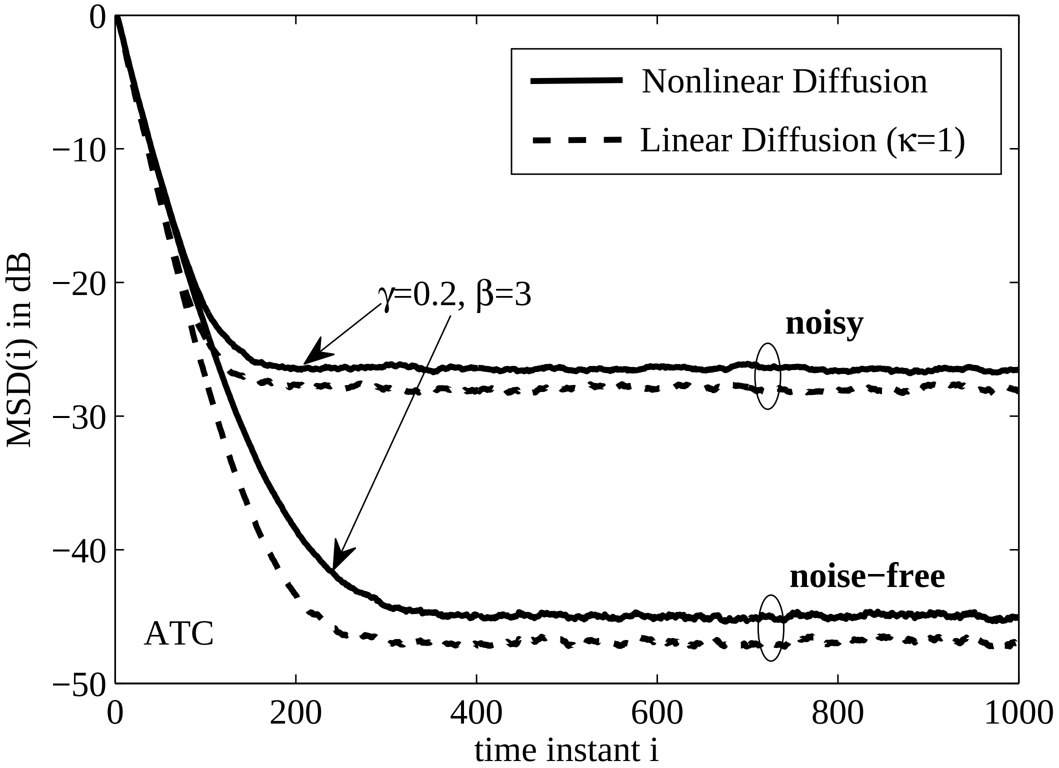


Figure 2.6: Mean-square-deviation (MSD) for ATC. Nonlinear Diffusion Adaptation with Bounded Function Can Be Close to the Linear Case by Balancing the Parameter γ and β, While the Peak Transmit Power γ^2 Is Always Bounded.

and $b_{nl} = 1$ if l and n are linked, $b_{nl} = 0$ otherwise. We use the nonlinear function $h(x) = \gamma_n \tanh(\beta_n x)$ with the parameters $(\gamma_n, \beta_n) = (0.4, 2.5)$ or $(0.2, 3)$, setting $\gamma_n = \gamma, \beta_n = \beta$ for all n. The step-size μ is set to 0.1 in the adaptation step. The learning curves in Fig. 2.5 are obtained by ensemble average of independent 200 trials where each trial has 1000 iterations.

Fig. 2.5 shows both of the linear and the nonlinear algorithms converging to $\boldsymbol{\omega}^o$ with certain amount of error variances. Mean-square-deviation (MSD) is used for the performance metric, defined as $\text{MSD}(i) \triangleq \frac{1}{N} E\left[||\widetilde{\boldsymbol{\omega}}(i)||^2 \right]$. The convergence rate of $E\left[||\widetilde{\boldsymbol{\omega}}(i)||^2 \right]$ for the nonlinear algorithm with $h(x) = 0.2 \tanh(3x)$ is slower than the linear case (i.e. $h(x) = x$) because of $\epsilon_{nm}(i) << \kappa = 1$ for all n, i, and m, as we analyzed in Section 2.3. However, it can be close to the linear case by controlling γ and β in the nonlinear $h(w_n)$.

Bounding the peak transmissions can save transmit power, while it is close to the linear case as shown in Fig. 2.5. Note that the peak power γ^2 is always bounded (i.e., $\gamma < 1/\sqrt{M}$ for all n) in transmissions. When the transmission links are noisy, it is obvious that MSDs are bigger than noise-free links as $\boldsymbol{\xi}(i)$ is in (2.24). However, the MSDs can be reduced by controlling $\{a_{nl}\}$ of (2.4): setting smaller η; or properly selecting $\{a_{nl}\}$ if noise variances are different at every node.

2.5 Conclusion

We have proposed nonlinear diffusion adaptation with bounded transmission. Convergence properties of the proposed algorithms are studied in the mean and mean square sense. For sufficiently small step-sizes and the combining weights of (2.4), the nonlinear diffusion adaptation strategies are stable. However, the convergence rate can be slower than (or equal to) the linear cases because of the bounded transmission. The numerical results show that performance of the nonlinear diffusion adaptation can be close to the linear case by balancing the peak transmission for power savings.

Chapter 3

DISTRIBUTED QUANTILE ESTIMATION IN SENSOR NETWORKS

Distributed sensors measure physical phenomena observable over a certain region and fuse the sensed information by communicating locally. This type of network is scalable and energy efficient because each node shares its data only with its neighbors. A traditional problem in this domain is to estimate an average of measurements by iteratively averaging the states with neighboring ones, and achieve a *consensus* on the global average of the initial measurements [4, 5, 15]. This has influenced many distributed estimation applications due to the broad use of the arithmetic mean in signal processing techniques.

Distributed average consensus of sensor measurement data can be used in monitoring applications. One example would be to monitor average temperature (or, other statistical metrics) over a sensor network in remote areas. The arithmetic mean of temperature data represents the *central tendency* of temperature. However, the mean can be vulnerable, as a measure of central tendency, to the skewness of the distribution. Outliers can also cause bias to the sample mean. An alternative metric is the median that represents the midpoint which divides the dataset into two subsets of equal size. More generally *quantiles* are cutpoints below which random draws from CDF fall with certain probabilities that correspond to the cutpoints. Beyond estimating the median, quantiles can be used in various applications such as outlier removal and computation of robust statistics from a set of measurement data by eliminating the values higher (or lower) than a certain cutpoint. One such roust statistics is the trimmed mean which is an average of the data excluding outliers. Maximum and minimum values can be viewed as extreme examples of quantiles. Quantile regression estimates the conditional quantiles of measurement data distribution where the statistics such as mean and variance may change over time. This method has been used in a variety of machine learning [55] as well as statistical applications [56].

In this chapter, we consider the quantiles estimation in a distributed way which is necessary, if nodes in a network have local measurement data only but want to know the quantile value without the global ECDF. The sensor network is assumed to be fully distributed where there is no fusion center. Sensor measurement data are unlabeled. Each node maintains its own data and state of estimate, and communicates the information only with neighboring nodes via noisy communication links between nodes. Any knowledge of the network graph structure is inaccessible to every node. The states of quantile estimates are recursively updated with two steps at each iteration. The *local update* step is based on the individual measurement data and the current state of quantile estimate. The updates are transfered to the neighboring nodes by *averaging* the estimates. We analyze convergence behavior of the distributed quantile estimation algorithm. We show that the estimated state sequence is asymptotically unbiased and converges toward the true quantile in mean-square sense. The proposed algorithm is applicable for finding an ordered measurement in network such as max-consensus [91], node selection [92], median, and trimmed mean.

This chapter consists of the following. In Section 3.1, we describe existing literature and define the notation used in the chapter. Section 3.2 and 3.3 describe the system model and problem statements followed by the proposed algorithm in Section 3.4. Convergence analysis is provided in Section 3.5. We demonstrate the proposed algorithm and convergence analysis with simulations in Section 3.6, discussing potential applications. Finally we describe conclusions in Section 3.7.

3.1 Comparison with Related Works

3.1.1 Distributed Parameter Estimations Based on Consensus

There are significant number of works related to such a consensus-based distributed parameter estimation. See [31–33, 93] for the early works, which inspired numerous applications. Distributed least-mean-square (LMS) algorithm is introduced to estimate a linear system parameter in various scenarios [6, 35, 42, 47, 94]. In these works, sensors observe random data at every iteration, generated by a linear system with a parameter vector. In [36], the authors proposed the consensus plus innovation scheme for distributed parameter estimation with single- and mixed-time scales. They consider nonlinear as well as linear system models and show convergence analysis. They assume that the sensors observe random data at every iteration and the observation model is continuous and invertible. In contrast, our model uses the ECDF which is discontinuous and non-invertible. Our work can be considered as a root finding problem which is similar with Robbins-Monro stochastic approximation algorithm [95], but we consider a distributed graph network setup. Reference [37] shows a performance analysis for Robbins-Monro algorithm in a distributed framework, where they considered the asynchronous random gossip algorithms [96] with random data observation of a continuous function at every iteration. However, our work assumes that the size of measurement data is finite, utilizing ECDF which is nonlinear, discontinuous, and non-invertible.

3.1.2 Distributed Selection Problem

For a limited size of measurement data in distributed networks, the work in this dissertation can be considered as distributed node selection problem, which is to find the n-th smallest measurement out of N data samples which can be related to n-th quantile. The references [92,97–99] are a few most closely related to this dissertation. They are similar with this dissertation's work in that each node maintains a piece of the entire data set with quantile state information, wishing to identify n-th smallest data. However, their main contributions are fundamentally different than this dissertation. Their focus is mainly on minimizing communication iterations, regardless of mixed-time scale consideration. Moreover, their algorithms are based on the shout-echo protocol [100] which consists of one broadcast and responses from all the other nodes. In [92], a leader node is chosen to maintain candidates of quantiles at each round of the gossip protocol, reducing the candidates until only a single candidate is left. In [97], the authors provide a lower bound of communication time complexity to reach the n-th smallest element on a connected graph network, assuming that every node knows the network's diameter which is defined as the length of the longest shortest path between any two sensor nodes. However, their algorithm needs to maintain a set of candidates for the n-th selection steadily reducing the set until it reaches the desired element under a certain criterion. Reference [98] depends on guessing and selection strategy to find the k-th smallest element. Their algorithm maintains a set of control messages such as start, small, large, and stop where the messages are transmitted to the entire network at every communication iteration. A distributed selection algorithm in [99] performs on a graph network, but their algorithm is also based on the shout-echo model and increases the number of message exchanges as the network size becomes larger. In contrast, our algorithm is a fully distributed method without any type of leading nodes or candidate sets. The algorithm is also scalable because any control messages are not transmitted to every node. Furthermore, our work considers more realistic case of communication links between nodes being corrupted by independent random noise, whereas the references above are based on noiseless communication links.

3.2 System Model

Consider N sensor nodes over a connected and undirected distributed network $\mathbb{G} = (\mathbb{N}, \mathbb{E})$ where there is no fusion center. Due to the connectedness, the eigenvalue λ_2 of the Laplacian matrix $\mathbf{L}$ is positive. Each node n has a scalar measurement denoted by $x_n \in \mathbb{R}$, where $n = 1, \ldots, N$, and $\{x_n\}_{n=1}^{N}$ constructs ECDF $\widehat{F}$. Without loss of generality, it can be assumed that the measurement set is sorted in ascending order. Let $\mathbf{x} = [x_1, \ldots, x_N]^T$ where $x_1 \leq \cdots \leq x_N$. Each node maintains a real-valued scalar state to be updated for quantile estimation. Let $\omega_n(i)$ denote the state of node n at time i. The state is transferred to neighboring nodes via wireless links in the presence of random communication noise $\xi_{nl}(i)$ from node l to n. Random noise on the link from l to n is assumed independent and identically distributed (i.i.d.) random process $\{\xi_{nl}(i)\}_{i\geq 0}$ with zero mean and variance $E\left[\xi_{nl}^2(i)\right]$ where $\sup_{n,l,i} E\left[\xi_{nl}^2(i)\right] < \infty$. As the communication iteratively continues, node n updates its own state $\omega_n(i)$ based on its own measurement x_n and neighbors' states $\{\omega_l(i)\}_{l\in\mathbb{N}_n}$ where $\mathbb{N}_n$ denotes the set of neighboring nodes of n.

Let p denote the ratio that corresponds to a quantile θ_p where $0 < p < 1$. When $p = 0.5$, the corresponding quantile $\theta_{0.5}$ is the median of $\mathbf{x}$. When $p = 0.75$, the corresponding $\theta_{0.75}$ indicates that 75% of measurement data is less than or equal to $\theta_{0.75}$. Define the ECDF from measurement data $\mathbf{x}$ as

$$\widehat{F}(\omega; \mathbf{x}) = \frac{1}{N} \sum_{n=1}^{N} u(\omega - x_n) \tag{3.1}$$

where the step function $u(\cdot)$ is given by

$$u(\omega - x_n) = \begin{cases} 1, & \text{if } \omega \geq x_n \\ 0, & \text{otherwise} \end{cases} . \tag{3.2}$$

Note that the ECDF $\widehat{F}(\omega; \mathbf{x})$ in (3.1) is a stair-case function, and θ_p is the inverse of the ECDF in some sense. More formally, for the ECDF $\widehat{F}(\omega, \mathbf{x})$, the relation between

p and θ_p can be defined [101] as

$$\theta_p = \inf_{\omega} \left\{ \omega : \widehat{F}(\omega; \mathbf{x}) \geq p \right\}. \tag{3.3}$$

The use of p in (3.3) results in two cases. One is $p \notin \{\frac{1}{N}, \frac{2}{N} \ldots, 1\}$. If N is known to each node, a quantile θ_p corresponds to a measurement x_n that is given by

$$\theta_p \triangleq x_n, \quad \text{for } \frac{n-1}{N} < p < \frac{n}{N} \tag{3.4}$$

where $n = 1, \ldots, N$. The other case is $p = \frac{n}{N}$ where a quantile could be found as $\theta_p \in [x_n, x_{n+1})$. However, we consider the first case (3.4) for the quantile definition that is more strict than the second case and can be useful in various applications because ω at every node converges toward a single value x_n.

Quantiles may be centrally obtained by using the ECDF of (3.1) after collecting all the measurement data $\mathbf{x}$. Practically in distributed wireless sensor networks, however, the centralized method is not directly applicable.

3.3 Problem Statement

Since each node has only the limited size of measurement data where we assume a fixed real-valued scalar x_n is given to node n, it may be impossible to know the global ECDF $\widehat{F}(\omega; \mathbf{x})$ in large-scale networks. In addition, it is difficult to synchronize the local states of all nodes (i.e., having $\{\omega_n(i)\}_{n=1}^N$ to be $\omega(i)$ for all n) at every iterative update over the network. The centralized method may require transmission of the measurement data $\mathbf{x}$ and the states $\{\omega_n(i)\}_{n=1}^N$ from all nodes to a fusion center with undesirable transmission power consumption. Also, all the information exchange is corrupted by communication random noise. Despite the constraints mentioned above, we want every node n to estimate the quantile θ_p for a given p as $i \to \infty$.

Suppose the ECDF of (3.1) in a fully distributed network $\mathbb{G} = (\mathbb{N}, \mathbb{E})$. There is no fusion center to collect the measurement data. Each node n communicates within neighborhood $\mathbb{N}_n$ via wireless communication channel corrupted by random noise, as described in Section 3.2. Given x_n and $p \notin \{\frac{1}{N}, \frac{2}{N} \ldots, 1\}$ at node n, where N is known at every node n, we want a distributed quantile estimation algorithm that generates the state $\omega_n(i)$ such that, as $i \to \infty$,

$$\omega_n(i) \to \theta_p, \quad \forall n \tag{3.5}$$

with the definition of θ_p in (3.4). If $p = \frac{n}{N}$ where $n = 1, \ldots, N$, as described in Section 3.2, an estimated quantile could be any value within an interval $[x_n, x_{n+1})$. In this paper we consider $p \notin \{\frac{1}{N}, \frac{2}{N} \ldots, 1\}$.

3.4 Distributed Quantile Estimation

A consensus-based distributed algorithm is proposed where each node n locally updates $\omega_n(i)$ with approximation error due to the lack of full measurement $\mathbf{x}$ and communication random noise, while combining $\omega_n(i)$ for each n with its neighboring $\omega_l(i)$ where $l \in \mathbb{N}_n$ for a given p, as iteration i increases. Let $\omega_n(i)$ and $\psi_n(i)$ denote respectively the state of quantile estimate and its intermediate state after locally updating $\omega_n(i)$ at node n at iteration i. Node n updates its state $\omega_n(i)$ based on the local measurement data x_n for the given constant p. The *local update* step is given by

$$\psi_n(i) = \omega_n(i) - \alpha(i)\Big[u\big(\omega_n(i) - x_n\big) - p\Big], \quad \forall n, \ i \geq 0, \tag{3.6}$$

where $\{\alpha(i)\}_{i\geq 0}$ is a deterministic step-size sequence that will be explained later in detail. Instead of synchronizing $\{\omega_n(i)\}_{n=1}^N$ at every iteration i, however, we consider that the intermediate state $\psi_n(i)$ at each n is averaged with its own neighboring states $\psi_l(i)$ where $l \in \mathbb{N}_n$. The *averaging* step at node n is then performed by

$$\omega_n(i+1) = \psi_n(i) - \eta(i) \sum_{l \in \mathbb{N}_n} \Big[\psi_n(i) - \big(\psi_l(i) + \xi_{nl}(i)\big)\Big], \ \forall n, \ i \geq 0, \tag{3.7}$$

where $\psi_l(i)$ denotes the state transmitted from node l with being perturbed at node n by communication random noise $\xi_{nl}(i)$, $\mathbb{N}_n$ denotes the set of neighboring nodes, and $\eta(i)$ is the step-size that controls exchange rate of node n with neighboring nodes at time i. We consider a deterministic sequence $\{\eta(i)\}_{i\geq 0}$ that will be explained later in more detail.

Let $\boldsymbol{\omega}(i) = [\omega_1(i), \dots, \omega_N(i)]^T$ and $\boldsymbol{\psi}(i) = [\psi_1(i), \dots, \psi_N(i)]^T$. Laplacian matrix $\mathbf{L}$ is described in Section II. The vector forms of (3.6) and (3.7) respectively can be described as

$$\boldsymbol{\psi}(i) = \boldsymbol{\omega}(i) - \alpha(i)\mathbf{y}(i), \tag{3.8}$$

$$\boldsymbol{\omega}(i+1) = \big(\mathbf{I} - \eta(i)\mathbf{L}\big)\boldsymbol{\psi}(i) - \eta(i)\boldsymbol{\xi}(i), \tag{3.9}$$

where

$$\mathbf{y}(i) = \big[y_1(i), \dots, y_N(i)\big]^T, \tag{3.10}$$

$$y_n(i) \triangleq u\big(\omega_n(i) - x_n\big) - p, \quad \forall n, \tag{3.11}$$

$$\boldsymbol{\xi}(i) = -\left[\sum_{l\in\mathbb{N}_1} \xi_{1l}(i), \dots, \sum_{l\in\mathbb{N}_N} \xi_{Nl}(i)\right]^T. \tag{3.12}$$

Combining (3.8) and (3.9), we can express the distributed quantile estimation algorithm as, for $i \geq 0$,

$$\begin{aligned} \boldsymbol{\omega}(0) &= \mathbf{x}, \\ \boldsymbol{\omega}(i+1) &= \big(\mathbf{I} - \eta(i)\mathbf{L}\big)\big(\boldsymbol{\omega}(i) - \alpha(i)\mathbf{y}(i)\big) - \eta(i)\boldsymbol{\xi}(i). \end{aligned} \tag{3.13}$$

The step-sizes $\alpha(i)$ and $\eta(i)$ satisfy the persistence condition:

$$\alpha(i) > 0,\ \sum_{i=0}^{\infty} \alpha(i) = \infty,\ \sum_{i=0}^{\infty} \alpha^2(i) < \infty, \tag{3.14}$$

$$\eta(i) > 0,\ \sum_{i=0}^{\infty} \eta(i) = \infty,\ \sum_{i=0}^{\infty} \eta^2(i) < \infty. \tag{3.15}$$

The conditions of (3.14) and (3.15) imply that decaying rates of the step-sizes are fast but not too fast. This condition has been commonly used for convergence analysis, based on conventional stochastic approximation theory [102–104]. However, the distributed quantile estimation algorithm (3.13) is a combined vector form of (3.8) and (3.9), and results in a mixed-time scale for the iterative updates in (3.13). For convergence, the step-size $\alpha(i)$ in (3.8) needs to decrease faster than $\eta(i)$ in (3.9). Rewriting the algorithm of (3.13) in the standard stochastic approximation form, we have

$$\boldsymbol{\omega}(i+1) = \boldsymbol{\omega}(i) - \eta(i)\bigg(\mathbf{L}\big(\boldsymbol{\omega}(i) - \alpha(i)\mathbf{y}(i)\big) + \frac{\alpha(i)}{\eta(i)}\mathbf{y}(i) + \boldsymbol{\xi}(i)\bigg). \tag{3.16}$$

If $\frac{\alpha(i)}{\eta(i)} \to \infty$ as $i \to \infty$, the states in (3.16) never converge. Thus, we need condition that $\alpha(i)$ decreases faster than $\eta(i)$. Moreover, the larger the decaying rate of $\alpha(i)$ is than $\eta(i)$, the faster $\frac{\alpha(i)}{\eta(i)}$ approaches zero. A convergence analysis for such a mixed-time scale approach with appropriate choices of step-sizes was also used in [36].

We summarize the assumption for step-sizes that will be used for the convergence behavior in Section 3.5.

Assumption 2 *(Decreasing step-sizes) The step-size $\alpha(i)$ in (3.8) decreases faster than $\eta(i)$ in (3.9) with the forms:*

$$\alpha(i) = \frac{\alpha_0}{(i+1)^{\tau_1}} \quad \textit{and} \quad \eta(i) = \frac{\eta_0}{(i+1)^{\tau_2}}, \quad \textit{for } i = 0, 1, \ldots, \tag{3.17}$$

where τ_1 and τ_2 denote constant decaying rates of $\alpha(i)$ and $\eta(i)$, respectively, α_0 and η_0 are positive initial step-sizes, and $1 \geq \tau_1 > \tau_2 > 0.5$. Moreover, $\tau_1 - \tau_2$ is close to 0.5 but less than 0.5.

One example of the step-size choice that satisfies Assumption 2 is $\tau_1 = 1$ and $\tau_2 = 0.505$. As $\tau_1 - \tau_2$ decreases, difference between the decaying rates of $\alpha(i)$ and $\eta(i)$ also decreases, resulting in more slowly decreasing sequence $\frac{\alpha(i)}{\eta(i)}$ in (3.16).

3.5 Convergence Analysis

In this section we analyze convergence behavior of the distributed quantile estimation algorithm (3.13). It is shown that the state sequence $\{\omega_n(i)\}_{i\geq 0}$ at node n is asymptotically unbiased in Theorem 3.5.3 and the estimated sequence converges to the true mode in mean-square sense in Theorem 3.5.4. To achieve the above results, we use some properties of real number sequences described in Lemma 3.5.1.

Lemma 3.5.1 *Consider the sequences $\{r_1(i)\}_{i\geq 0}$ and $\{r_2(i)\}_{i\geq 0}$, with non-negative constants a_1 and a_2, which are given by*

$$r_1(i) = \frac{a_1}{(i+1)^{\delta_1}}, \quad r_2(i) = \frac{a_2}{(i+1)^{\delta_2}} \tag{3.18}$$

where $0 \leq \delta_1 \leq 1$ and $\delta_2 \geq 0$. If $\delta_1 < \delta_2$, then, for arbitrary fixed i_0,

$$\lim_{i\to\infty} \sum_{k=i_0}^{i-1} \left[\prod_{l=k+1}^{i-1} \big(1 - r_1(l)\big) \right] r_2(k) = 0. \tag{3.19}$$

Proof See Appendix A.

Lemma 3.5.2 *(Boundedness) Define $\omega_{avg}(i) \triangleq \frac{1}{N}\mathbf{1}^T\boldsymbol{\omega}(i)$ that is the average of $\boldsymbol{\omega}(i)$ at i. Given the measurement data $\mathbf{x}$ and ratio p, there is a decreasing sequence $\{\eta(i)\}_{i\geq 0}$ of* (3.17) *and we have*

$$\limsup_{i\to\infty} \eta(i) E\Big[\omega_{avg}(i) - \theta_p\Big] = 0, \tag{3.20}$$

$$\limsup_{i\to\infty} \eta(i) E\Big[\big|\omega_{avg}(i) - \theta_p\big|^2\Big] = 0. \tag{3.21}$$

Proof See Appendix B.

We show in Theorem 1 that the quantile estimation is asymptotically unbiased, as $i \to \infty$.

Theorem 3.5.3 *(Asymptotic Unbiasedness) Consider that a constant ratio p is given for estimating a certain quantile θ_p. Suppose Lemma 3.5.1 and 3.5.2 satisfy under Assumptions 2. The state sequence $\{\omega_n(i)\}_{i\geq 0}$ at node n is asymptotically unbiased:*

$$\lim_{i\to\infty} E\big[\omega_n(i)\big] = \theta_p \quad \textit{for } 1 \leq n \leq N. \tag{3.22}$$

Proof It is shown that $\|E[\boldsymbol{\omega}(i)] - \theta_p\mathbf{1}\|$ converges to 0, as $i \to \infty$. Recall that $\mathbf{L} \cdot \mathbf{1} = \mathbf{0}$. By subtracting $\theta_p\mathbf{1}$ on both sides of (3.13), it can be rewritten as

$$\boldsymbol{\omega}(i+1) - \theta_p\mathbf{1} = \big(\mathbf{I} - \eta(i)\mathbf{L}\big)\big(\boldsymbol{\omega}(i) - \theta_p\mathbf{1}\big) - \alpha(i)\big(\mathbf{I} - \eta(i)\mathbf{L}\big)\mathbf{y}(i) - \eta(i)\boldsymbol{\xi}(i). \tag{3.23}$$

Define a rank-1 matrix

$$\mathbf{G} \triangleq \frac{1}{N}\mathbf{1}\mathbf{1}^T. \tag{3.24}$$

The average of $\boldsymbol{\omega}(i)$ at i is expressed as

$$\mathbf{z}(i) \triangleq \mathbf{G}\boldsymbol{\omega}(i) = \omega_{\text{avg}}(i)\mathbf{1}. \tag{3.25}$$

With $\mathbf{z}(i) - \theta_p\mathbf{1} = \mathbf{G}\big(\boldsymbol{\omega}(i) - \theta_p\mathbf{1}\big)$, (3.23) can be rewritten as

$$\begin{aligned}\boldsymbol{\omega}(i+1) - \theta_p\mathbf{1} = \Big(\mathbf{I} - \eta(i)\mathbf{L} - \eta(i)\mathbf{G}\Big)\big(\boldsymbol{\omega}(i) - \theta_p\mathbf{1}\big) - \alpha(i)\Big(\mathbf{I} - \eta(i)\mathbf{L}\Big)\mathbf{y}(i)\\ -\eta(i)\boldsymbol{\xi}(i) + \eta(i)\big(\mathbf{z}(i) - \theta_p\mathbf{1}\big).\end{aligned} \tag{3.26}$$

Let $\mathbf{R} \triangleq \mathbf{L} + \mathbf{G}$. Taking expectations on both sides of (3.26) leads to

$$\begin{aligned}E\big[\boldsymbol{\omega}(i+1)\big] - \theta_p\mathbf{1} = \Big(\mathbf{I} - \eta(i)\mathbf{R}\Big)\Big(E\big[\boldsymbol{\omega}(i)\big] - \theta_p\mathbf{1}\Big) - \alpha(i)\Big(\mathbf{I} - \eta(i)\mathbf{L}\Big)E\big[\mathbf{y}(i)\big]\\ +\eta(i)\Big(E\big[\mathbf{z}(i)\big] - \theta_p\mathbf{1}\Big)\end{aligned} \tag{3.27}$$

where the zero-mean random noise vector $\boldsymbol{\xi}(i)$ was canceled. The smallest eigenvalue of Laplacian matrix $\mathbf{L}$ is equal to zero, and $\|\mathbf{I}-\eta(i)\mathbf{L}\|=1$ for all i. Taking $\|\cdot\|$ of both sides of (3.27), by triangle inequality, we have

$$\begin{aligned}\Big\|E\big[\boldsymbol{\omega}(i+1)\big]-\theta_p\mathbf{1}\Big\| \le \Big\|\mathbf{I}-\eta(i)\mathbf{R}\Big\|\Big\|E\big[\boldsymbol{\omega}(i)\big]-\theta_p\mathbf{1}\Big\| + \alpha(i)\Big\|E\big[\mathbf{y}(i)\big]\Big\| \\ +\eta(i)\Big\|E\big[\mathbf{z}(i)\big]-\theta_p\mathbf{1}\Big\|. \qquad (3.28)\end{aligned}$$

There is a constant maximum eigenvalue $\lambda_{\max}$ of matrix $\mathbf{R}$. With the step-size $\eta(i)<\frac{1}{\lambda_{\max}(\mathbf{R})}$ for sufficiently large i, we can obtain

$$\Big\|\mathbf{I}-\eta(i)\mathbf{R}\Big\| \le 1-\eta(i)\lambda_{\max}(\mathbf{R}) < 1. \qquad (3.29)$$

We simplify the notation and define $\lambda \triangleq \lambda_{\max}(\mathbf{R})$. By substituting (3.29) into (3.28) and recursions from 0 up to $i-1$, we obtain

$$\begin{aligned}\Big\|E\big[\boldsymbol{\omega}(i)\big]-\theta_p\mathbf{1}\Big\| \le \prod_{k=0}^{i-1}\Big(1-\eta(k)\lambda\Big)\Big\|\boldsymbol{\omega}(0)-\theta_p\mathbf{1}\Big\| \\ +\sum_{k=0}^{i-1}\left[\prod_{l=k+1}^{i-1}\Big(1-\eta(l)\lambda\Big)\right]\alpha(k)\Big\|E\big[\mathbf{y}(k)\big]\Big\| \\ +\sum_{k=0}^{i-1}\left[\prod_{l=k+1}^{i-1}\Big(1-\eta(l)\lambda\Big)\right]\eta(k)\Big\|E\big[\mathbf{z}(k)\big]-\theta_p\mathbf{1}\Big\|. \qquad (3.30)\end{aligned}$$

We use the property $1-z\le e^{-z}$ for $0\le z\le 1$. For sufficiently large $k\ge i_0$ there exists positive $\eta(k)\lambda\le 1$. Under Assumption 2, the first term of RHS in (3.30) goes to zero as $i\to\infty$ because

$$\lim_{i\to\infty}\prod_{k=i_0}^{i-1}\Big(1-\eta(k)\lambda\Big) \le \lim_{i\to\infty} e^{-\lambda\sum_{k=i_0}^{i-1}\eta(k)} = 0. \qquad (3.31)$$

Since $\mathbf{y}(k)$ in (3.30) is bounded (i.e., $-1\le\mathbf{y}(k)\le 1$) for all k with the step function $u(\cdot)$ and p defined in (3.11), we have $\Big\|E\big[\mathbf{y}(k)\big]\Big\|\le\sqrt{N}$. The second term of RHS in

(3.30) has the following inequality:

$$\sum_{k=0}^{i-1}\left[\prod_{l=k+1}^{i-1}\Big(1-\eta(l)\lambda\Big)\right]\alpha(k)\Big\|E\big[\mathbf{y}(k)\big]\Big\| \le \sum_{k=0}^{i-1}\left[\prod_{l=k+1}^{i-1}\Big(1-\eta(l)\lambda\Big)\right]\alpha(k)\sqrt{N}. \tag{3.32}$$

Thus, we find that the second term of RHS in (3.30) falls onto the case of (3.19) in Lemma 3.5.1 where the numerators $a_1 = \eta(1)\lambda$ and $a_2 = \alpha(0)\sqrt{N}$ of $r_1(i)$ and $r_2(i)$ in (3.18), respectively. Since $\tau_2 < \tau_1$, the second term goes to zero as $i \to \infty$.

By Lemma 3.5.2 the third term is bounded: $\big\|E\big[\mathbf{z}(k)\big] - \theta_p\mathbf{1}\big\| = \big\|E\big[\omega_{\text{avg}}(k)\big] - \theta_p\mathbf{1}\big\| < \infty$. Then, the third term of RHS in (3.30) goes to zero as $i \to \infty$:

$$\lim_{i\to\infty}\sum_{k=0}^{i-1}\left[\prod_{l=k+1}^{i-1}\Big(1-\eta(l)\lambda\Big)\right]\eta(k)\Big\|E\big[\mathbf{z}(k)\big] - \theta_p\mathbf{1}\Big\| = 0, \tag{3.33}$$

because, for small k, by Lemma 3.5.1,

$$\lim_{i\to\infty}\prod_{l=k+1}^{i-1}\Big(1-\eta(l)\lambda\Big) = 0, \tag{3.34}$$

whereas for large k, by Lemma 3.5.2,

$$\lim_{k\to\infty}\eta(k)\Big\|E\big[\mathbf{z}(k)\big] - \theta_p\mathbf{1}\Big\| = 0. \tag{3.35}$$

The theorem follows because $\lim_{i\to\infty}\Big\|E\big[\boldsymbol{\omega}(i)\big] - \theta_p\mathbf{1}\Big\| = 0$ in (3.30).

We show in Theorem 3.5.4 that the proposed algorithm converges toward the true quantile θ_p in mean-square sense in the presence of communication noise variance and goes to zero in the absence of noise. Note that if an estimator converges to the true parameter in mean-square sense, it also converges in probability. The mean-square convergence indicates stronger consistency than convergence in probability.

Theorem 3.5.4 *(Mean-Square Convergence) Suppose Lemma 3.5.1 and Lemma 3.5.2 satisfy under Assumptions 2. The sequence generated by the distributed quantile estimation algorithm* (3.13)*, for a given ratio p, converges to θ_p in mean-square sense:*

$$\lim_{i\to\infty}E\Big[\big\|\boldsymbol{\omega}(i) - \theta_p\mathbf{1}\big\|^2\Big] = 0. \tag{3.36}$$

Proof Subtracting $\theta_p\mathbf{1}$ from both sides of (3.13), we have

$$\boldsymbol{\omega}(i+1)-\theta_p\mathbf{1}=\Big(\mathbf{I}-\eta(i)\mathbf{L}\Big)\big(\boldsymbol{\omega}(i)-\theta_p\mathbf{1}\big)-\alpha(i)\Big(\mathbf{I}-\eta(i)\mathbf{L}\Big)\mathbf{y}(i)-\eta(i)\boldsymbol{\xi}(i). \quad (3.37)$$

Recall (3.24) and (3.25). We have the following relation:

$$\mathbf{z}(i)-\theta_p\mathbf{1}\triangleq\mathbf{G}\big(\boldsymbol{\omega}(i)-\theta_p\mathbf{1}\big)=\mathbf{G}\boldsymbol{\omega}(i)-\theta_p\mathbf{1}. \quad (3.38)$$

Noting $\mathbf{R}=\mathbf{L}+\mathbf{G}$, we can rewrite (3.37) as

$$\begin{aligned}\boldsymbol{\omega}(i+1)-\theta_p\mathbf{1}=\Big(\mathbf{I}-\eta(i)\mathbf{R}\Big)\big(\boldsymbol{\omega}(i)-\theta_p\mathbf{1}\big)-\alpha(i)\Big(\mathbf{I}-\eta(i)\mathbf{L}\Big)\mathbf{y}(i)-\eta(i)\boldsymbol{\xi}(i)&\\ +\eta(i)\big(\mathbf{z}(i)-\theta_p\mathbf{1}\big).& \quad (3.39)\end{aligned}$$

We use the property $\Big\|\mathbf{I}-\eta(i)\mathbf{L}\Big\|=1$ for Laplacian matrix $\mathbf{L}$ and $\|\mathbf{Ab}\|\leq\|\mathbf{A}\|\|\mathbf{b}\|$ where $\mathbf{A}\in\mathbb{R}^{N\times N}$ and $\mathbf{b}\in\mathbb{R}^{N\times 1}$. From (3.39), we have

$$\begin{aligned}\Big\|\boldsymbol{\omega}(i+1)-\theta_p\mathbf{1}\Big\|^2\leq\Big\|\mathbf{I}-\eta(i)\mathbf{R}\Big\|^2\Big\|\boldsymbol{\omega}(i)-\theta_p\mathbf{1}\Big\|^2+\alpha^2(i)\big\|\mathbf{y}(i)\big\|^2+\eta^2(i)\big\|\boldsymbol{\xi}(i)\big\|^2&\\ +\eta^2(i)\Big\|\mathbf{z}(i)-\theta_p\mathbf{1}\Big\|^2-2\alpha(i)\Big[\Big(\mathbf{I}-\eta(i)\mathbf{R}\Big)\big(\boldsymbol{\omega}(i)-\theta_p\mathbf{1}\big)\Big]^T\Big(\mathbf{I}-\eta(i)\mathbf{L}\Big)\mathbf{y}(i)&\\ +2\eta(i)\Big[\Big(\mathbf{I}-\eta(i)\mathbf{R}\Big)\big(\boldsymbol{\omega}(i)-\theta_p\mathbf{1}\big)\Big]^T\Big(\mathbf{z}(i)-\theta_p\mathbf{1}\Big)&\\ -2\alpha(i)\eta(i)\Big[\Big(\mathbf{I}-\eta(i)\mathbf{L}\Big)\mathbf{y}(i)\Big]^T\Big(\mathbf{z}(i)-\theta_p\mathbf{1}\Big)&\\ -2\eta(i)\Big[\Big(\mathbf{I}-\eta(i)\mathbf{R}\Big)\big(\boldsymbol{\omega}(i)-\theta_p\mathbf{1}\big)\Big]^T\boldsymbol{\xi}(i)&\\ +2\alpha(i)\eta(i)\Big[\Big(\mathbf{I}-\eta(i)\mathbf{L}\Big)\mathbf{y}(i)\Big]^T\boldsymbol{\xi}(i)&\\ -2\eta^2(i)\Big(\mathbf{z}(i)-\theta_p\mathbf{1}\Big)^T\boldsymbol{\xi}(i).&\\ & \quad (3.40)\end{aligned}$$

Due to Cauchy-Schwarz inequality and $x \leq 1 + x^2$ for any $x \in \mathbb{R}$, the fifth term of (3.40) can be rewritten as

$$\begin{aligned}
-2\alpha(i)\Big[\Big(\mathbf{I} - \eta(i)\mathbf{R}\Big)\big(\boldsymbol{\omega}(i) - \theta_p\mathbf{1}\big)\Big]^T\Big(\mathbf{I} - \eta(i)\mathbf{L}\Big)\mathbf{y}(i)& \\
\leq 2\alpha(i)\Big\|\mathbf{I} - \eta(i)\mathbf{R}\Big\|\Big\|\boldsymbol{\omega}(i) - \theta_p\mathbf{1}\Big\|\Big\|\mathbf{y}(i)\Big\|& \\
\leq 2\alpha(i)\Big[1 + \Big\|\mathbf{I} - \eta(i)\mathbf{R}\Big\|^2\Big\|\boldsymbol{\omega}(i) - \theta_p\mathbf{1}\Big\|^2\Big]\Big\|\mathbf{y}(i)\Big\|.& \qquad (3.41)
\end{aligned}$$

Similarly the sixth and seventh terms, respectively, can be rewritten as

$$\begin{aligned}
2\eta(i)\Big[\Big(\mathbf{I} - \eta(i)\mathbf{R}\Big)\big(\boldsymbol{\omega}(i) - \theta_p\mathbf{1}\big)\Big]^T\Big(\mathbf{z}(i) - \theta_p\mathbf{1}\Big)& \\
\leq 2\eta(i)\Big\|\mathbf{I} - \eta(i)\mathbf{R}\Big\|\Big\|\boldsymbol{\omega}(i) - \theta_p\mathbf{1}\Big\|\Big\|\mathbf{z}(i) - \theta_p\mathbf{1}\Big\|& \\
\leq 2\eta(i)\Big[1 + \Big\|\mathbf{I} - \eta(i)\mathbf{R}\Big\|^2\Big\|\boldsymbol{\omega}(i) - \theta_p\mathbf{1}\Big\|^2\Big]\Big\|\mathbf{z}(i) - \theta_p\mathbf{1}\Big\|& \qquad (3.42)
\end{aligned}$$

and

$$-2\alpha(i)\eta(i)\Big[\Big(\mathbf{I} - \eta(i)\mathbf{L}\Big)\mathbf{y}(i)\Big]^T\Big(\mathbf{z}(i) - \theta_p\mathbf{1}\Big) \leq 2\alpha(i)\eta(i)\big\|\mathbf{y}(i)\big\|\big\|\mathbf{z}(i) - \theta_p\mathbf{1}\big\|. \quad (3.43)$$

Substituting (3.41), (3.42), and (3.43) into (3.40) and taking $E[\cdot]$ on both sides of

(3.40), we obtain

$$
\begin{aligned}
E\Big[\Big\|\boldsymbol{\omega}(i+1)-\theta_p\mathbf{1}\Big\|^2\Big] \leq{}& \Big\|\mathbf{I}-\eta(i)\mathbf{R}\Big\|^2 E\Big[\big\|\boldsymbol{\omega}(i)-\theta_p\mathbf{1}\big\|^2\Big] + \alpha^2(i)E\Big[\big\|\mathbf{y}(i)\big\|^2\Big] \\
&+\eta^2(i)E\Big[\big\|\boldsymbol{\xi}(i)\big\|^2\Big] + \eta^2(i)E\Big[\big\|\mathbf{z}(i)-\theta_p\mathbf{1}\big\|^2\Big] + 2\alpha(i)E\Big[\big\|\mathbf{y}(i)\big\|\Big] \\
&+2\eta(i)E\Big[\big\|\mathbf{z}(i)-\theta_p\mathbf{1}\big\|\Big] + 2\alpha(i)\Big\|\mathbf{I}-\eta(i)\mathbf{R}\Big\|^2 E\Big[\big\|\boldsymbol{\omega}(i)-\theta_p\mathbf{1}\big\|^2\big\|\mathbf{y}(i)\big\|\Big] \\
&+2\eta(i)\Big\|\mathbf{I}-\eta(i)\mathbf{R}\Big\|^2 E\Big[\big\|\boldsymbol{\omega}(i)-\theta_p\mathbf{1}\big\|^2\big\|\mathbf{z}(i)-\theta_p\mathbf{1}\big\|\Big] \\
&+2\alpha(i)\eta(i)E\Big[\big\|\mathbf{y}(i)\big\|\big\|\mathbf{z}(i)-\theta_p\mathbf{1}\big\|\Big] \\
\leq{}& \bigg(1+2\alpha(i)E\Big[\big\|\mathbf{y}(i)\big\|\Big] + 2\eta(i)E\Big[\big\|\mathbf{z}(i)-\theta_p\mathbf{1}\big\|\Big]\bigg)\Big\|\mathbf{I}-\eta(i)\mathbf{R}\Big\|^2 E\Big[\big\|\boldsymbol{\omega}(i)-\theta_p\mathbf{1}\big\|^2\Big] \\
&+\alpha^2(i)E\Big[\big\|\mathbf{y}(i)\big\|^2\Big] + \eta^2(i)E\Big[\big\|\boldsymbol{\xi}(i)\big\|^2\Big] + \eta^2(i)E\Big[\big\|\mathbf{z}(i)-\theta_p\mathbf{1}\big\|^2\Big] \\
&+2\alpha(i)E\Big[\big\|\mathbf{y}(i)\big\|\Big] + 2\eta(i)E\Big[\big\|\mathbf{z}(i)-\theta_p\mathbf{1}\big\|\Big] \\
&+2\alpha(i)\eta(i)E\Big[\big\|\mathbf{y}(i)\big\|\big\|\mathbf{z}(i)-\theta_p\mathbf{1}\big\|\Big] \\
\leq{}& \bigg(1+2\alpha(i)\sqrt{N} + 2\eta(i)E\Big[\big\|\mathbf{z}(i)-\theta_p\mathbf{1}\big\|\Big]\bigg)\Big\|\mathbf{I}-\eta(i)\mathbf{R}\Big\|^2 E\Big[\big\|\boldsymbol{\omega}(i)-\theta_p\mathbf{1}\big\|^2\Big] \\
&+\alpha(i)\Big(\alpha(i)N+2\sqrt{N}\Big) + \eta^2(i)N\sigma_\xi^2 + \eta^2(i)E\Big[\big\|\mathbf{z}(i)-\theta_p\mathbf{1}\big\|^2\Big] + 2\eta(i)E\Big[\big\|\mathbf{z}(i)-\theta_p\mathbf{1}\big\|\Big] \\
&+2\alpha(i)\eta(i)\sqrt{N}E\Big[\big\|\mathbf{z}(i)-\theta_p\mathbf{1}\big\|\Big]
\end{aligned}
\tag{3.44}
$$

where σ_ξ^2 denotes variance of $\xi_{nl}(i)$ for all n, l, i and the last inequality is due to $\big\|\mathbf{y}(i)\big\| \leq \sqrt{N}$ for all i (because each element of $\mathbf{y}(i)$ is bounded, i.e., $-1 \leq y_n(i) \leq 1,\ \forall n, i$). Recall (3.29) and $\big(1-\eta(k)\lambda\big)^2 \leq 1-\eta(k)\lambda$. Let $\gamma(i) \triangleq 2\alpha(i)\sqrt{N} +$

$2\eta(i)E\Big[\big\|\mathbf{z}(i)-\theta_p\mathbf{1}\big\|\Big]$ in (3.44). After recursions of (3.44) from 0 up to $i-1$, we have

$$
\begin{aligned}
E\Big[\Big\|\boldsymbol{\omega}(i)-\theta_p\mathbf{1}\Big\|^2\Big] \le {} & \prod_{k=0}^{i-1}\Big(1+\gamma(k)\Big)\Big(1-\eta(k)\lambda\Big)\big\|\boldsymbol{\omega}(0)-\theta_p\mathbf{1}\big\|^2 \\
& + \sum_{k=0}^{i-1}\left[\prod_{l=k+1}^{i-1}\Big(1+\gamma(l)\Big)\Big(1-\eta(l)\lambda\Big)\right]\alpha(k)\Big(\alpha(k)N+2\sqrt{N}\Big) \\
& + \sum_{k=0}^{i-1}\left[\prod_{l=k+1}^{i-1}\Big(1+\gamma(l)\Big)\Big(1-\eta(l)\lambda\Big)\right]\eta^2(k)N\sigma_\xi^2 \\
& + \sum_{k=0}^{i-1}\left[\prod_{l=k+1}^{i-1}\Big(1+\gamma(l)\Big)\Big(1-\eta(l)\lambda\Big)\right]\eta^2(k)E\Big[\big\|\mathbf{z}(k)-\theta_p\mathbf{1}\big\|^2\Big] \\
& + 2\sum_{k=0}^{i-1}\left[\prod_{l=k+1}^{i-1}\Big(1+\gamma(l)\Big)\Big(1-\eta(l)\lambda\Big)\right]\eta(k)E\Big[\big\|\mathbf{z}(k)-\theta_p\mathbf{1}\big\|\Big] \\
& + 2\sum_{k=0}^{i-1}\left[\prod_{l=k+1}^{i-1}\Big(1+\gamma(l)\Big)\Big(1-\eta(l)\lambda\Big)\right]\alpha(k)\eta(k)\sqrt{N}E\Big[\big\|\mathbf{z}(k)-\theta_p\mathbf{1}\big\|\Big].
\end{aligned}
\tag{3.45}
$$

For sufficiently large k there exists a positive constant c_1:

$$
\Big(1+\gamma(k)\Big)\Big(1-\eta(k)\lambda\Big) \le 1-\eta(k)c_1 < 1. \tag{3.46}
$$

Substituting (3.46) into (3.45), we can rewrite (3.45) as

$$
\begin{aligned}
E\Big[\big\|\boldsymbol{\omega}(i)-\theta_p\mathbf{1}\big\|^2\Big] \leq & \prod_{k=0}^{i-1}\Big(1-\eta(k)c_1\Big)\big\|\boldsymbol{\omega}(0)-\theta_p\mathbf{1}\big\|^2 \\
&+\sum_{k=0}^{i-1}\left[\prod_{l=k+1}^{i-1}\Big(1-\eta(l)c_1\Big)\right]\alpha(k)\Big(\alpha(k)N+2\sqrt{N}\Big) \\
&+\sum_{k=0}^{i-1}\left[\prod_{l=k+1}^{i-1}\Big(1-\eta(l)c_1\Big)\right]\eta^2(k)N\sigma_\xi^2 \\
&+\sum_{k=0}^{i-1}\left[\prod_{l=k+1}^{i-1}\Big(1-\eta(l)c_1\Big)\right]\eta^2(k)E\Big[\big\|\mathbf{z}(k)-\theta_p\mathbf{1}\big\|^2\Big] \\
&+2\sum_{k=0}^{i-1}\left[\prod_{l=k+1}^{i-1}\Big(1-\eta(l)c_1\Big)\right]\eta(k)E\Big[\big\|\mathbf{z}(k)-\theta_p\mathbf{1}\big\|\Big] \\
&+2\sum_{k=0}^{i-1}\left[\prod_{l=k+1}^{i-1}\Big(1-\eta(l)c_1\Big)\right]\alpha(k)\eta(k)\sqrt{N}E\Big[\big\|\mathbf{z}(k)-\theta_p\mathbf{1}\big\|\Big].
\end{aligned}
\tag{3.47}
$$

The first term of RHS in (3.47) converges to zero as $i \to \infty$ for the same property of (3.31). The other terms of RHS in (3.47), except for the fifth term, fall onto the case of $\delta_1 < \delta_2$ in Lemma 3.5.1. Moreover, by Lemma 3.5.2, the fifth term of RHS in (3.47) goes to zero as $i \to \infty$:

$$\lim_{i\to\infty}\sum_{k=0}^{i-1}\left[\prod_{l=k+1}^{i-1}\Big(1-\eta(l)c_1\Big)\right]\eta(k)E\Big[\big\|\mathbf{z}(k)-\theta_p\mathbf{1}\big\|\Big]=0, \tag{3.48}$$

because, for small k

$$\lim_{i\to\infty}\prod_{l=k+1}^{i-1}\Big(1-\eta(l)c_1\Big)=0, \tag{3.49}$$

whereas for large k

$$\lim_{k\to\infty}\eta(k)E\Big[\big\|\mathbf{z}(k)-\theta_p\mathbf{1}\big\|\Big]=0. \tag{3.50}$$

Therefore, the theorem provides (3.36).

3.6 Simulations

In this section we demonstrate the distributed quantile estimation under various conditions. Consider a distributed sensor network, illustrated in Fig. 3.1, which is a connected graph with $N = 50$ where the graph's connectivity is characterized by $\lambda_2(\mathbf{L}) = 2.2815$. Each node n has a scalar measurement x_n taken from a realization of random variable X. Without loss of generality, we can assume that the measurement data is distributed in ascending order: $x_1 \leq x_2 \leq \cdots \leq x_N$. We consider two distributions: discrete uniform and log-normal. The discrete uniform distribution is normalized, and the actual values are ranging from 0 to $(N-1)/N$ with $1/N$ increase for each sample. The log-normal distribution is generated by $X \sim \ln\mathcal{N}(0, 0.25)$. We take N realizations of random variable X whose ECDF is illustrated in Fig. 3.2. With the set of measurement data $\{x_n\}_{n=1}^{N}$, also denoted by $\mathbf{x}$ in vector form, a quantile θ_p of $\mathbf{x}$ is estimated for a desired ratio p in a distributed way. The states $\{\omega_n(i)\}_{n=1}^{N}$ are recursively updated by the algorithm (3.13), as i increases. The initial states $\boldsymbol{\omega}(0)$ are the nodes' own measurement data $\mathbf{x}$. Consider $p = \frac{k-\varepsilon}{N}$ for $\theta_p = x_k$ defined in (3.4), where $\varepsilon = 0.5$ and N is known to each node. Due to the choice of $p = \frac{k-\epsilon}{N}$, the k-th smallest element in $\mathbf{x}$ is estimated by achieving $\omega_n(i) = \theta_p$ for all n, as $i \to \infty$. We evaluate mean-squared error for convergence of the estimation by the following metric:

$$\frac{1}{N}E\Big[\big\|\boldsymbol{\omega}(i) - \theta_p\mathbf{1}\big\|^2\Big],\ i \geq 0, \tag{3.51}$$

where $\frac{1}{N}$ is due to normalization and $E\left[\cdot\right]$ can be approximated by ensemble averaging over 200 realizations. According to Assumption 2, we can set $\tau_1 = 1$ and $\tau_2 = 0.505$. We begin with $\alpha_0 = 1$ and $\eta_0 = 0.5/d_{\max}$ for $\alpha(i)$ and $\eta(i)$ respectively, where $d_{\max}$ denotes the maximum degree in graph network.

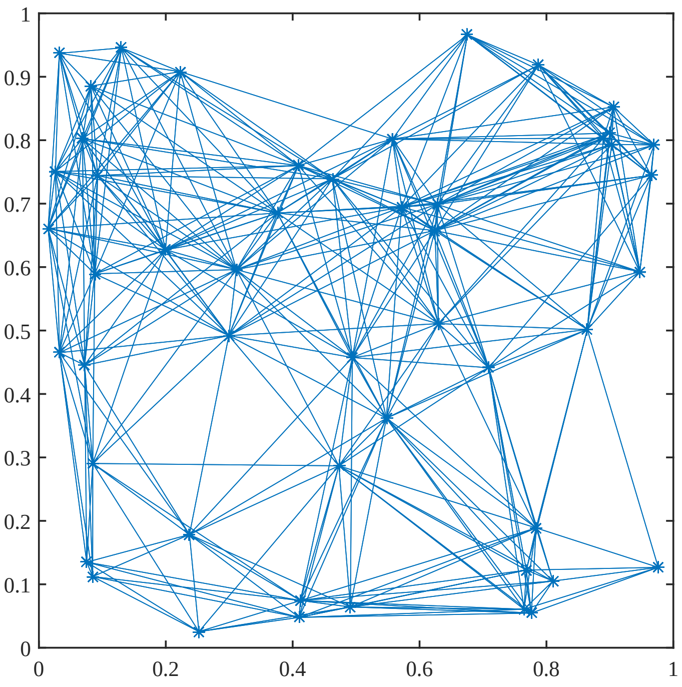


Figure 3.1: A graph for distributed sensor network ($N = 50$) where the graph's connectivity is characterized by $\lambda_2(\mathbf{L}) = 2.2815$. A node has a scalar sensor measurement, denoted as x_n where $n = 1, \ldots, N$.

3.6.1 Distributed Quantile Estimation

Given the sensor network $N = 50$ and measurement data $\mathbf{x}$, suppose that $p = 0.99$ is selected with $k = 50$ and $\varepsilon = 0.5$. Then the distributed algorithm (3.13) estimates $\max(x_1, \ldots, x_N)$, as $i \to \infty$. Fig. 3.3 shows that all the states converge toward $\theta_{p=0.99} = 0.98$, which is the maximum value of uniform $\mathbf{x}$ in the presence of communication noise. Similarly, one can estimate the minimum by setting $p = \frac{1-0.5}{50} = 0.01$. More generally, the k-th smallest element can be estimated by setting $p = \frac{k-0.5}{N}$.

The algorithm (3.13) is evaluated for different noise variances σ_ξ^2 with the metric (3.51). In Fig. 3.4, the quantile $\theta_{0.89}$ of uniform data was tested. One can see that the estimated states converge toward the true quantile. Fig. 3.5 shows the squared

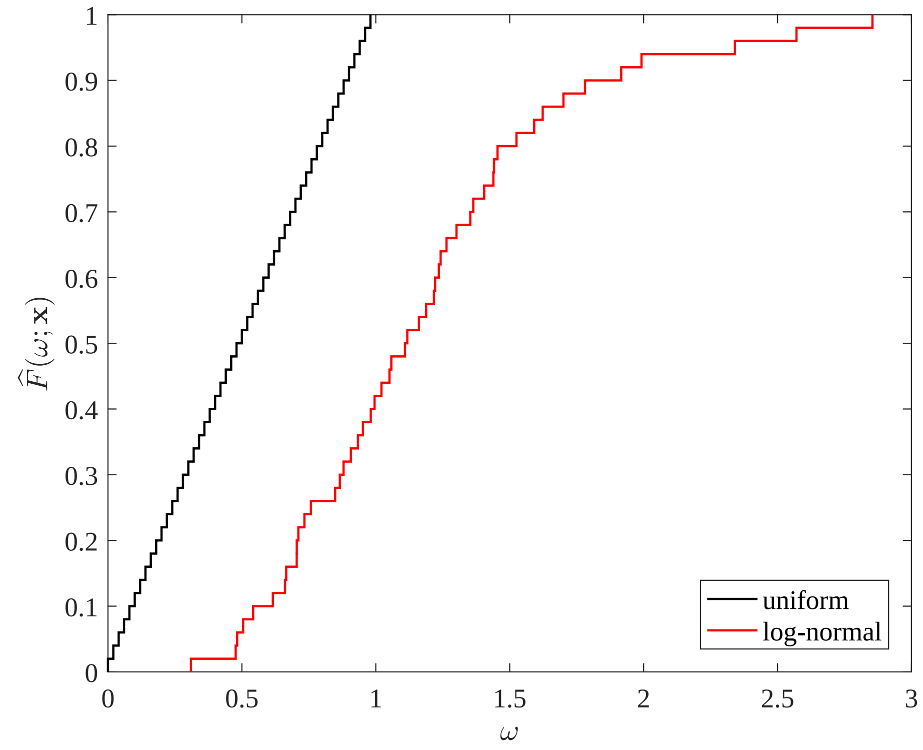


Figure 3.2: Empirical CDF generated from $\{x_n\}_{n=1}^N$ where $N = 50$. 1) uniform distribution and 2) log-normal distribution $\ln \mathcal{N}(\mu, \sigma^2)$ where $\mu = 0$ and $\sigma = 0.5$.

error convergence. We use the following metric, since there is no randomness in the absence of communication noise.

$$\frac{1}{N}\|\boldsymbol{\omega}(i) - \theta_p \mathbf{1}\|^2, \ i \geq 0. \tag{3.52}$$

One can see that the sequence converges to the true quantile where we experimented with $\theta_{0.01}$, $\theta_{0.49}$, and $\theta_{0.89}$. Note that the $\theta_{0.01}$ and $\theta_{0.49}$ are the minimum and median of $\mathbf{x}$ respectively. The initial trajectories in Fig. 3.5 depends on the sensor network structure and the measurement data contained at each node.

We experiment with the log-normal data for more practical sensor networks. The distribution of data is illustrated in Fig. 3.2. The estimated states toward the minimum value in the presence of communication noise is shown in Fig. 3.6. The desired quantile θ_p is $x_k = 0.3098$ for $p = \frac{k-\varepsilon}{N} = 0.01$ where $k = 1$, $\varepsilon = 0.5$, and $N = 50$.

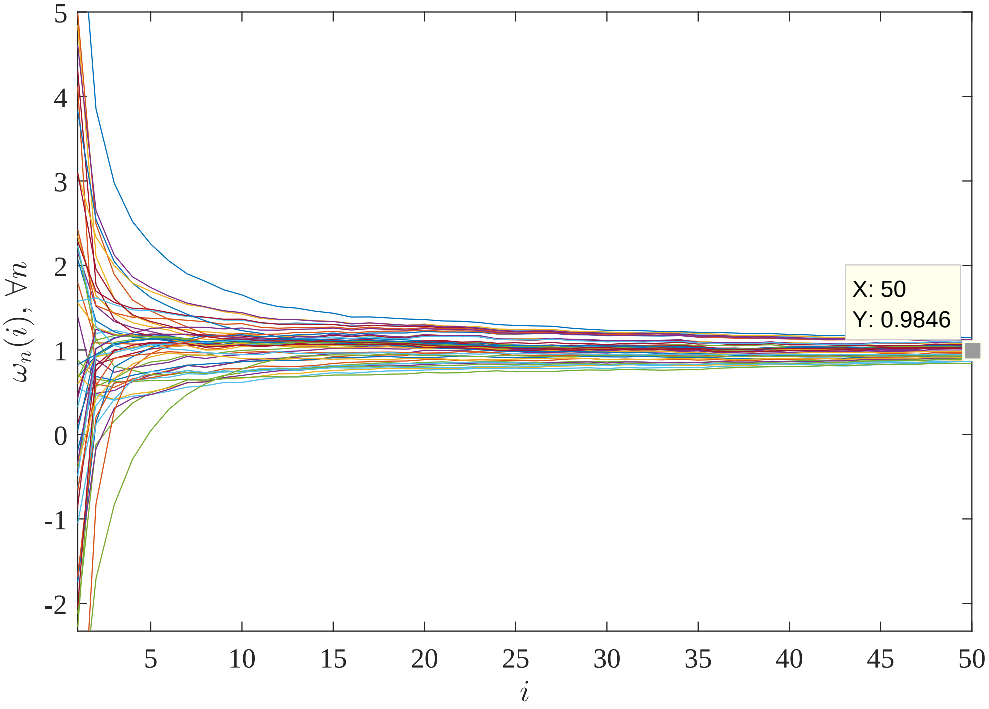


Figure 3.3: Maximum value ($\theta_{0.99} = 0.98$) estimation for the uniform data in the presence of communication noise by setting $p = 0.99$.

Fig. 3.7 shows the estimated quantile sequences for the maximum in the presence of communication noise. One can see that the states go toward the maximum value $\theta_{0.99} = 2.8544$ with the parameter $\alpha_0 = 3$.

3.6.2 Applications with Numerical Experiments

Outlier Identification and Trimmed Mean: As an application, our algorithm can be used to determine whether individual node measures an outlier value or not. One can judge that larger (or smaller) value than a quantile (e.g., 0.9 or 0.1) is assumed to be outliers. This can be used for robust average consensus, as the estimated mean is not biased by erroneous outliers. Removing outliers is often useful when there exist malicious sensor measurements. If it is identified that the measurement

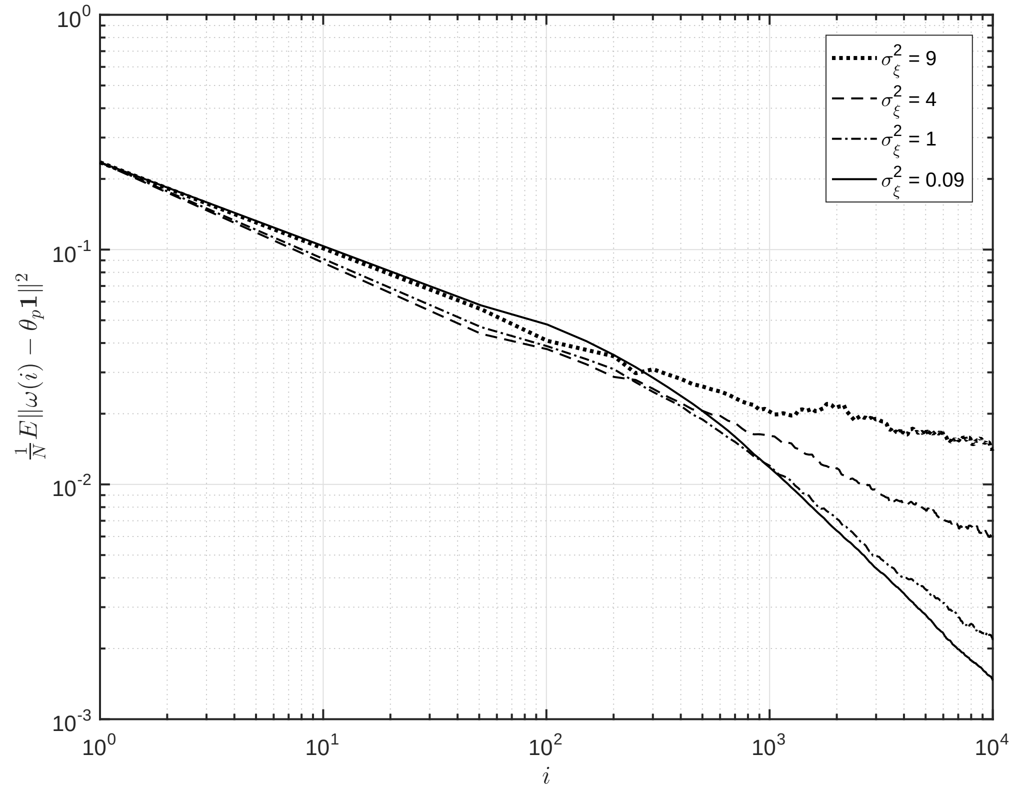


Figure 3.4: Mean-squared convergence behavior under different communication noise variances.

at node n is an outlier, then the node by itself is not averaged with neighboring nodes so that the outliers can be removed when the global average is estimated. An extended application would be the trimmed mean. We often want to average sensor measurements only within the range of $a\% \sim b\%$ where $0 < a < b < 100$. $\theta_{a/100}$ and $\theta_{b/100}$ can be estimated by our algorithm and then individual node can be identified whether they are within the range or not. Then, the trimmed mean is obtained by the average consensus [4, 5] only with the nodes in $[\theta_{a/100}, \theta_{b/100}]$.

Median Estimation: A useful metric to measure centrality of sensor measurement data is median. When there are outliers or when the data distribution is skewed, median can be used for a centrality measure of the data. When the data size $N(\geq 2)$

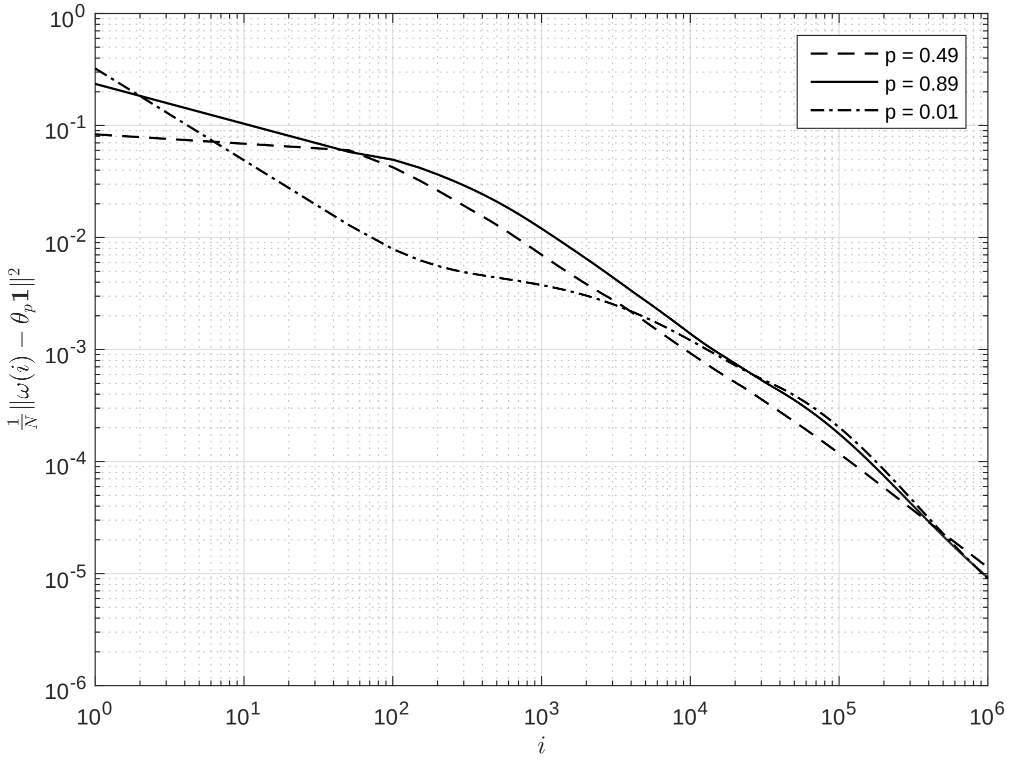


Figure 3.5: Squared error convergence behavior of various quantiles in the absence of communication noise.

is even, the median can be defined as a value between $x_{0.5N}$ and $x_{0.5N+1}$. By setting $p = 0.5 - \varepsilon/N$, the algorithm (3.13) estimates $x_{0.5N}$, and similarly $x_{0.5N+1}$. When N is odd, the median $\theta_p = x_{\lceil 0.5N \rceil}$ is estimated by setting $p = \frac{\lceil 0.5N \rceil - \varepsilon}{N}$ where $0 < \varepsilon < 1$.

Maximum and Minimum Estimation: We already showed some results of maximum and minimum value estimation for the uniform and log-normal data in Fig. 3.6 and Fig. 3.7. In the absence of communication noise, max- and min-consensus can be achieved by setting $p = \frac{N-\varepsilon}{N}$ and $p = \frac{1-\varepsilon}{N}$, respectively, where $0 < \varepsilon < 1$.

3.7 Conclusion

We have shown a consensus-based distributed quantile estimation algorithm using empirical CDF with limited size of measurement data. States of a quantile estimation

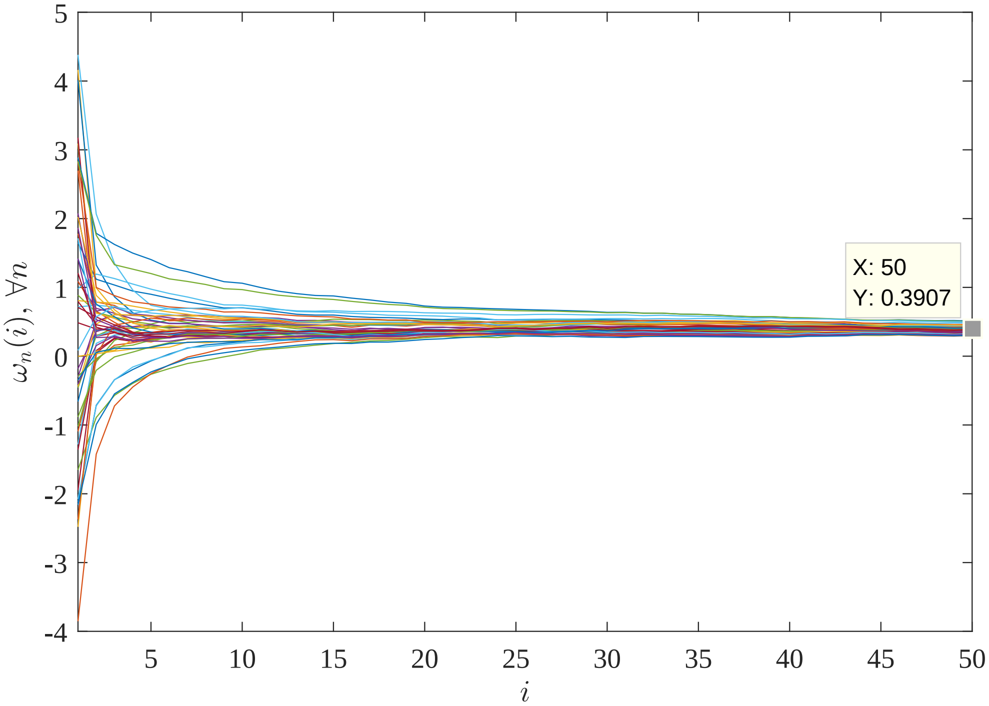


Figure 3.6: Minimum value estimation for log-normal data in the presence of communication noise.

are recursively updated by the combination of local update and averaging steps in the presence of communication noise. We analyzed convergence behaviors of the algorithm based on mixed-time scale stochastic approximation where the averaging time scale dominates the local update time scale. The estimated state sequence is asymptotically unbiased and converges toward the true quantile in mean-square sense. Also, the quantile estimation achieves a consensus in the absence of communication noise. We demonstrated the performance of algorithm with numerical experiments. Finally, potential applications by using our algorithm were discussed. Maximum, minimum, median, k-th smallest element selection out of N elements, outliers identification, and trimmed mean can be obtained in fully distributed sensor networks.

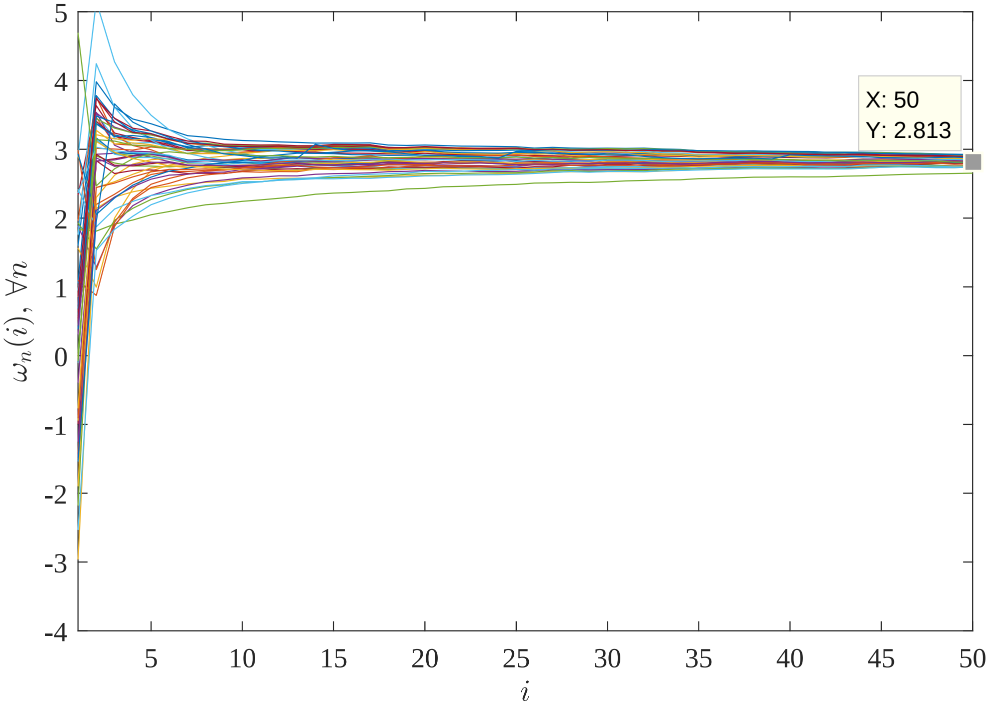


Figure 3.7: Maximum value estimation for log-normal data in the presence of communication noise ($\alpha_0 = 3$).

Chapter 4

DISTRIBUTED MEAN-SHIFT MODE ESTIMATION IN SENSOR NETWORKS

Distributed sensor networks have advantages such as scalability and energy efficient communications by allowing local individual nodes to share their data only with neighboring nodes. Achieving consensus on an arithmetic mean of sensor data is possible by distributed average consensus schemes [4, 12], which can be used in many applications. One example would be to monitor average temperature over a sensor network in a remote area. Generally one may choose an average because it can represent a measure of *central tendency* of the data. However, the mean can be highly sensitive to a small number of outliers. Also, the sample mean will not effectively locate the densest region, for data coming from a skewed distribution. Among the many statistical metrics such as median and other quantiles, the mode is arguably the closest one to the intuitive understanding of central tendency in that it represents the *most probable* value of sensor data. Moreover, when the data represents sensor location information, the mode is a useful metric for the densest region of sensor deployment.

For the past several decades, the mode as a measure of central tendency has been extensively used for data analysis [57, 59, 60] because of its robustness to outliers and other contaminations [61, 62]. The fundamental idea to estimate the mode is to find the *densest* region of the data distribution, which can be estimated by either non-parametric [60] or parametric methods [63–65]. The mode also can be obtained by estimating the density function and finding the location that maximizes the density. A well-known non-parametric method is Parzen's kernel density estimator (KDE) [105]. The kernel size may affect the accuracy of the mode estimation, and it can be selected using Silverman's rule of thumb [106]. Practical approaches to mode estimation are based on the Gaussian mean-shift (GMS) algorithm [107–111]. The method recursively updates the gradient ascent of the KDE until the updated states reach stationary points. However, the mentioned mode estimation methods presume that all the data are analyzed in a *centralized* location.

When it comes to *decentralized* methods, we can relate our work to estimating parameters of a Gaussian mixture model (GMM) in a distributed way because one of the mixture components can represent the mode. Refs. [112] and [113] proposed the distributed expectation-maximization (EM) algorithms to parametrically estimate GMM parameters where the global sufficient statistics are computed by an incremental scheme [112] and consensus filters [113]. However, both methods require large number of computations for the global statistics at every update of E- and M-steps. Diffusion strategies for distributed EM applied to GMMs were proposed in [114, 115], which does not require such multiple iterations at every E- and M-steps. Such GMM-based methods, however, are not suitable to estimate the mode when data distribution does not consist of Gaussian mixtures or when the distribution is highly skewed.

In this chapter, we propose a distributed mode estimation method to find the central tendency of measurement data in fully distributed wireless sensor networks. The measurement data is assumed to have unimodal distribution, as we are interested in finding the central tendency of the data. We take the GMS approach which is described in the EM framework. The proposed scheme iteratively updates the state of mode estimate from measurement data at each sensor node, while the intermediate states at each iteration are diffused over the network. Each node generates a mean-shift vector that is defined by current state and conditional mean of measurement data. The mode is found by making the mean-shift vector converge to zero. As the iteration continues, the states at individual nodes converge toward a consensus on the mode under the condition of decaying step sizes. Moreover, we consider wireless communication links between local nodes in a distributed network. Information exchange is corrupted by random noise. Simulation results show that the proposed algorithm estimates a global mode which is close to the centralized mode estimates.

The rest of chapter consists of the following. In Section 4.1 and 4.2, we describe the system model and problem statement for the distributed mode estimation. Standard EM algorithm to unify the GMM and GMS are summarized in Section 4.3, and the details of the distributed mode estimation method is explained in Section 4.4. Simulation results are shown in Section 4.4. Finally we provide conclusions of this section in Section 4.5.

4.1 System Model

Consider N sensor nodes over a connected and undirected graph model of a distributed network where there is no fusion center. Each node n has M vector measurements denoted by $\mathbf{x}_{nm} \in \mathbb{R}^D$, where $n = 1, \ldots, N$ and $m = 1, \ldots, M$. The vector $\mathbf{x}_{nm}$ was generated from a probability density function (PDF) that is unimodal but not necessarily symmetric. Let $f : \mathbb{R}^D \to \mathbb{R}$ be the kernel density estimator (KDE) [105] of the PDF. We use an isotropic kernel, which is the commonly used kernel type in practice. Also, we consider that the kernel function is identical for every node n and measurement m. Then, the KDE with Gaussian kernel is given by

$$f(\boldsymbol{\omega}) = \frac{1}{N}\sum_{n=1}^{N} f_n(\boldsymbol{\omega}) = \frac{1}{N}\sum_{n=1}^{N}\left[\frac{1}{M}\sum_{m=1}^{M}\frac{1}{Z}K_h\left(\boldsymbol{\omega} - \mathbf{x}_{nm}\right)\right] \tag{4.1}$$

where h is the kernel size (bandwidth), Z is a normalization term that is dependent on h that is $Z = \sqrt{2\pi}h$, and

$$\text{Gaussian: } K_h(\mathbf{x}) = \exp\left(-\frac{1}{2h^2}\|\mathbf{x}\|^2\right). \tag{4.2}$$

The choice of h can be suggested by Silverman's rule of thumb [106]:

$$h = \left(\frac{4}{D+2}\frac{1}{NM}\right)^{\frac{1}{D+4}}\sigma \tag{4.3}$$

where we used $\sigma = \sum_{d=1}^{D} \sigma_d$ and σ_d is the standard deviation of the d-th element of measurement vectors $\mathbf{x}$. The sample standard deviation for σ_d as well as σ can be obtained by the average consensus algorithms [4, 12]. The network size N can be estimated at every node n by distributed node counting algorithm [116, 117]. D and M are known to every node.

One may further investigate a kernel size matrix $\mathbf{H}$ that is not isotropic. Then, the Gaussian kernel in (4.2) can be rewritten as $K_{\mathbf{H}}(\mathbf{x}) = \exp\left(-0.5\mathbf{x}^T\mathbf{H}^{-1}\mathbf{x}\right)$. However, selection of kernel size matrix $\mathbf{H}$ is beyond the scope of this chapter. We focus on the mode estimation in distributed networks using the simplest and practical choice $\mathbf{H} = h^2\mathbf{I}$. There are other types of kernels than the Gaussian. One example is Epanechnikov kernel, which is also isotropic.

$$\text{Epanechnikov: } K_h(\mathbf{x}) = \begin{cases} 1 - \frac{\|\mathbf{x}\|^2}{h^2}, & \frac{\|\mathbf{x}\|^2}{h^2} \leq 1 \\ 0, & \text{otherwise} \end{cases}. \tag{4.4}$$

A global mode of KDE $f(\boldsymbol{\omega})$ with the Gaussian kernel in (4.2) can be found by seeking stationary points $\boldsymbol{\omega}$ such that

$$\nabla_{\boldsymbol{\omega}} f(\boldsymbol{\omega}) = \sum_{n=1}^{N}\sum_{m=1}^{M} K_h\left(\boldsymbol{\omega} - \mathbf{x}_{nm}\right)\left(\boldsymbol{\omega} - \mathbf{x}_{nm}\right) = \mathbf{0} \tag{4.5}$$

for positive h, N, M, and Z. There are several methods to find the stationary points of (4.5) such as fixed-point iteration, gradient ascent, or Newton's method that iteratively searches the maximum point of $f(\boldsymbol{\omega})$. We consider Newton's method in this paper. Let $\boldsymbol{\omega}(i)$ denote the state vector at time i. The state update rule is given by

$$\boldsymbol{\omega}(i+1) = \boldsymbol{\omega}(i) - \alpha\left[\nabla^2_{\boldsymbol{\omega}} f(\boldsymbol{\omega}(i))\right]^{-1}\nabla_{\boldsymbol{\omega}} f(\boldsymbol{\omega}(i)) \tag{4.6}$$

where α is a small step size. The next state $\boldsymbol{\omega}(i+1)$ is a function of $\boldsymbol{\omega}(i)$, and recursively updated as $i \rightarrow \infty$ with the Hessian matrix and the gradient vector at the state $\boldsymbol{\omega}(i)$.

4.2 Problem Statement in Distributed WSNs

In order for (4.6) to work in distributed networks, the state update from $\boldsymbol{\omega}(i)$ to $\boldsymbol{\omega}(i+1)$ requires the access to all local measurement data. As an another scenario, each node n maintains only the state $\boldsymbol{\omega}_n(i)$ and the measurement dataset $\{\mathbf{x}_{nm}\}_{m=1}^{M}$. Then, $\boldsymbol{\omega}_n(i), \forall n$, are synchronized to a common $\boldsymbol{\omega}(i)$ at every iteration i. However, both methods may be impossible in large-scale networks. It is difficult for every node to access all measurement data or to synchronize the local states at every iteration i of (4.6) over the network. Also, all the information exchange is corrupted by communication random noise due to wireless channel. One could consider a centralized method that requires transmission of the measurement data from all nodes to a fusion center and then executes a search algorithm to find a mode. However, this requires undesirable transmission power consumption in large-scale networks. One could also consider the traditional distributed average consensus algorithm [4] (i.e., $\widehat{\gamma} = \frac{1}{N}\sum_{n=1}^{N}\widehat{\gamma}_n$). Once every node independently estimates the local mode $\widehat{\gamma}_n$ from M measurement data. The algorithm iteratively averages the states $\{\widehat{\gamma}_n(i)\}_{n=1}^{N}$ until they converge to $\widehat{\gamma}$ as $i \rightarrow \infty$. However, this scheme may not achieve the global mode $\widehat{\theta}$ because average of modes $\widehat{\gamma}_n = \operatorname{argmax}_{\gamma} f_n(\gamma)$ for all n does not necessarily represent the global mode $\widehat{\gamma} = \operatorname{argmax}_{\gamma} \frac{1}{N}\sum_{n=1}^{N} f_n(\gamma)$. Moreover, this approach requires sufficiently large M at every node across the network, which is impractical in large scale sensor networks.

Despite the constraints mentioned above, in a distributed sensor network, we want every node n to estimate the mode $\boldsymbol{\gamma} \in \mathbb{R}^D$ as $i \to \infty$:

$$\boldsymbol{\omega}_n(i) \to \boldsymbol{\gamma}, \quad \forall n, \tag{4.7}$$

where $\boldsymbol{\gamma} = \text{argmax}_{\boldsymbol{\omega}} f(\boldsymbol{\omega})$.

4.3 Expectation Maximization (EM) Algorithm

In this section we briefly explain the standard Expectation-Maximization (EM) algorithm [118] and describe how the EM can be applied to unify Gaussian Mixture Model (GMM) and Gaussian Mean-Shift (GMS). EM algorithm is an iterative method to find maximum likelihood estimates of parameters in statistical models which generally involve *unobserved* variables as well as *observed* data and *unknown parameters*. The unobserved variables can be considered as various forms in general. They could be missing values among the data or could be memberships with which the observed variables are associated. As an example of the latter, one can consider a mixture model where each observed data sample has a series of likelihood memberships. Each membership corresponds to one of the mixture components in the statistical model. The EM algorithm iteratively updates the maximum likelihood of the model parameters with two-steps. Expectation of the (log)-likelihood function of the parameter, with respect to the posterior distribution of the unobserved variable conditioned on the observed data and the current state of parameter estimate, is evaluated in the (E)xpectation step. Then, in the (M)aximization step, the algorithm updates the state estimate of the parameter in a way that the updated state maximizes the expectation quantity which was with the previous state of the parameter.

Let $q(\mathbf{x}|\theta)$ be a statistical model with an unknown parameter θ. A set of vector observed data $\{\mathbf{x}_k\}_{k=1}^N$ was generated from the model $q(\mathbf{x}|\theta)$. Let $z_k \in \{1, \ldots, M\}$ be a discrete unobserved variable where $k = 1, \ldots, N$. The log-likelihood function of θ with the complete (or extended) data $\{\mathbf{x}_k, z_k\}$ is given by

$$\sum_{k=1}^{N} \mathcal{L}_k(\theta) = \sum_{k=1}^{N} \log q(\mathbf{x}_k, z_k|\theta) \tag{4.8}$$

where $q(\mathbf{x}_k, z_k|\theta)$ indicates the likelihood when the observed data $\mathbf{x}_k$ is associated with the unobserved variable z_k under the unknown parameter θ. The maximum likelihood of the unknown θ can be obtained by the marginal likelihood of the observed data. In the E-step, the EM algorithm computes the expected value of log-likelihood with respect to the distribution of unobserved variable z_k conditioned on the observed data $\mathbf{x}_k$ and the current state of the parameter estimate at iteration i:

$$\begin{aligned} \text{E-step: } Q(\theta|\theta(i)) &= \sum_{k=1}^{N} E_{q(z_k|\mathbf{x}_k, \theta(i))} \big[\mathcal{L}_k(\theta)\big] \\ &= \sum_{k=1}^{N} \sum_{z_k=1}^{M} q(z_k|\mathbf{x}_k, \theta(i)) \log q(\mathbf{x}_k, z_k|\theta). \end{aligned} \tag{4.9}$$

In the M-step, the algorithm finds the next state $\theta(i+1)$ that maximizes the expected value $Q(\theta|\theta(i))$ which was with the previous state $\theta(i)$ of the parameter estimate. This step can be expressed as

$$\text{M-step:} \quad \theta(i+1) = \underset{\theta}{\operatorname{argmax}}\, Q(\theta|\theta(i)) \tag{4.10}$$

where $\theta(i+1)$ can be obtained by equating the gradient of Q with respect to θ to zero: $\frac{\partial}{\partial\theta} Q(\theta|\theta(i)) = 0$.

4.3.1 Gaussian Mixture Model (GMM) with EM Algorithm

GMM parameters can be estimated by the EM algorithm. Let θ denote one of GMM parameters:

$$\theta \in \{p_m, \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m\}_{m=1}^{M} \tag{4.11}$$

where p_m is the mixing probability of m-th component, $\boldsymbol{\mu}_m$ and $\boldsymbol{\Sigma}_m$ denote $D \times 1$ mean vector and $D \times D$ covariance matrix of m-th component that we consider as a Gaussian model. Typically the statistical model can be expressed as

$$q\left(\mathbf{x}|\{p_m, \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m\}_{m=1}^{M}\right) = \sum_{m=1}^{M} p_m \, p(\mathbf{x}|m) \tag{4.12}$$

where $\mathbf{x} \in \mathbb{R}^D$, with the normalization $Z = \sqrt{(2\pi)^D \left|\boldsymbol{\Sigma}_m\right|}$,

$$p(\mathbf{x}|m) = \frac{1}{Z} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_m)^T \boldsymbol{\Sigma}_m^{-1} (\mathbf{x} - \boldsymbol{\mu}_m)\right). \tag{4.13}$$

Suppose a dataset $\{\mathbf{x}_k\}_{k=1}^{N}$ is drawn from the mixture model $q\left(\mathbf{x}|\{p_m, \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m\}_{m=1}^{M}\right)$. The unobserved variable $z_k \in \{1, \ldots, M\}$ represents membership in one of M mixture components. $q(\mathbf{x}_k, z_k|\theta)$ indicates the likelihood of a parameter θ for the data sample $\mathbf{x}_k$ and its membership z_k. The EM algorithm iteratively finds the maximum likelihood of θ with the E- and M-step of (4.9) and (4.10) respectively. As θ denotes one of GMM parameters, the EM algorithms repeats for $\{p_m, \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m\}_{m=1}^{M}$.

4.3.2 Gaussian Mean-Shift (GMS) with EM Algorithm

GMS is an iterative algorithm to find modes, which can be obtained by seeking stationary points of nonparametric kernel density estimator (KDE) $f(\mathbf{x})$. With the isotropic Gaussian kernel, the KDE can be described as

$$f(\mathbf{x}) = \frac{1}{M} \sum_{m=1}^{M} \frac{1}{Z} K_h(\mathbf{x} - \mathbf{x}_m) \tag{4.14}$$

where $Z = \sqrt{2\pi}h$ is the normalization term and $K_h(\mathbf{x} - \mathbf{x}_m)$ is the Gaussian kernel centered at $\mathbf{x}_m$ with the kernel size h:

$$K_h(\mathbf{x} - \mathbf{x}_m) = \exp\left(-\frac{1}{2h^2}\|\mathbf{x} - \mathbf{x}_m\|^2\right). \tag{4.15}$$

The stationary points of $f(\mathbf{x})$ can be obtained by equating the gradient of f with respect to $\mathbf{x}$ to zero. The gradient of f with the Gaussian kernel $K_h(\mathbf{x} - \mathbf{x}_m)$ can be given by

$$\nabla_{\mathbf{x}} f(\mathbf{x}) = \frac{1}{M}\sum_{m=1}^{M}\frac{1}{Z}K_h(\mathbf{x} - \mathbf{x}_m)\frac{1}{h^2}\big(\mathbf{x} - \mathbf{x}_m\big) = 0. \tag{4.16}$$

Solving (4.16) leads to

$$\frac{\sum_{m=1}^{M} K_h(\mathbf{x} - \mathbf{x}_m)\mathbf{x}_m}{\sum_{m'=1}^{M} K_h(\mathbf{x} - \mathbf{x}'_m)} - \mathbf{x} = 0 \tag{4.17}$$

where the first term on the LHS indicates the conditional mean of $\{\mathbf{x}_m\}_{m=1}^{M}$ given $\mathbf{x}$. Iterative methods to update $\mathbf{x}$ may be listed as fixed-point iteration scheme, gradient ascent, or Newton's method. The fixed-point iteration scheme iteratively updates $\mathbf{x}$, starting from arbitrary value of $\mathbf{x}$, until (4.16) is satisfied. $\mathbf{x}(i+1)$ can be defined as a function of $\mathbf{x}(i)$ at iteration i:

$$\mathbf{x}(i+1) = \frac{\sum_{m=1}^{M} K_h(\mathbf{x}(i) - \mathbf{x}_m)\mathbf{x}_m}{\sum_{m'=1}^{M} K_h(\mathbf{x}(i) - \mathbf{x}_{m'})}. \tag{4.18}$$

The conditional mean on the RHS in (4.18) is iteratively shifted until it converges to a mode that is a stationary point of $f(\mathbf{x})$.

The GMS can be described in the framework of EM algorithm [111] by considering $f(\mathbf{x})$ in (4.14) as a mixture model $q\big(\mathbf{x}|\frac{1}{M}, \{\mathbf{x}_m\}_{m=1}^{M}, h\big)$ with M components. In this case, the mixture model parameters $\{p_m, \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m\}_{m=1}^{M}$ in (4.11) are reduced to $\{\frac{1}{M}, \{\mathbf{x}_m\}_{m=1}^{M}, h\}$. Suppose that the mixture model parameters are *fixed*. The mean vector $\boldsymbol{\mu}_m$ (center of the m-th component) corresponds to $\mathbf{x}_m$. The mixing probability

p_m and covariance matrix $\mathbf{\Sigma}_m$ are simplified to $\frac{1}{M}$ and h respectively. The whole mixture components can be shifted by varying a displacement vector $\boldsymbol{\omega}$, when redefining the m-th component center $\mathbf{x}_m$ as $\mathbf{x}_m - \boldsymbol{\omega}$. To describe GMS in EM framework, a conditional density model with the parameter $\boldsymbol{\omega}$ is introduced:

$$g(\mathbf{y}_k|\boldsymbol{\omega}) = \frac{1}{M}\sum_{m=1}^{M}\frac{1}{Z}K_h\left(\mathbf{y}_k - \left(\mathbf{x}_m - \boldsymbol{\omega}\right)\right) \tag{4.19}$$

where $\mathbf{y}_k$ denotes an observed data among N samples in the EM algorithm. For the Gaussian kernel in (4.2), it is true that

$$K_h\left(\mathbf{y}_k - \left(\mathbf{x}_m - \boldsymbol{\omega}\right)\right) = K_h\left(\left(\boldsymbol{\omega} + \mathbf{y}_k\right) - \mathbf{x}_m\right). \tag{4.20}$$

The log-likelihood function with the complete data $\{\mathbf{y}_k, z_t\}$ becomes

$$\sum_{k=1}^{N}\mathcal{L}_k(\boldsymbol{\omega}) = \sum_{k=1}^{N}\log g(\mathbf{y}_k, z_k|\boldsymbol{\omega}) \tag{4.21}$$

where $g(\mathbf{y}_k, z_k|\boldsymbol{\omega})$ indicates the likelihood of $\boldsymbol{\omega}$ for the observed data $\mathbf{y}_k$ and its membership $z_k \in \{1, \ldots, M\}$ in the mixture model $g(\mathbf{y}_k|\boldsymbol{\omega})$. We can select the observed data as the origin, i.e. $N = 1$ and $\mathbf{y}_1 = \mathbf{0}$, and rename z_k as m. Note that if $\mathbf{y}_1 = \mathbf{0}$, then $g(\mathbf{0}|\boldsymbol{\omega}) = f(\boldsymbol{\omega})$. More generally, due to the kernel (4.20), it can be also viewed as

$$g(\mathbf{y}_k|\boldsymbol{\omega}) = f(\boldsymbol{\omega} + \mathbf{y}_k). \tag{4.22}$$

The EM algorithm finds the maximum likelihood estimate of $\boldsymbol{\omega}$ with the statistical model $g(\mathbf{y}_k|\boldsymbol{\omega})$. This is equivalent with seeking the stationary point of KDE $f(\boldsymbol{\omega}+\mathbf{y}_k)$. In the E-step, $q(z_k|\mathbf{x}_k, \theta(i))$ and $q(\mathbf{x}_k, z_k|\theta)$ in (4.9) are replaced with $g(m|\mathbf{y}_1, \boldsymbol{\omega}(i))$ and $g(\mathbf{y}_1, m|\boldsymbol{\omega})$ respectively. The M-step for (4.10) updates the state $\boldsymbol{\omega}(i)$ such that $Q(\boldsymbol{\omega}|\boldsymbol{\omega}(i+1)) \geq Q(\boldsymbol{\omega}|\boldsymbol{\omega}(i))$ at iteration i. A mean-shift vector is derived from $\nabla_{\boldsymbol{\omega}}Q(\boldsymbol{\omega}|\boldsymbol{\omega}(i)) = 0$. Updating $\boldsymbol{\omega}(i)$ in the M-step results in making the mean-shift vector zero.

4.4 Distributed Mode Estimation

The EM-based GMS algorithm for the centralized mode estimation is extended to distributed networks. Consider that every node n maintains the complete data $\{\mathbf{y}_{nt}, z_{nt}\}_{t=1}^{T}$, state vector $\boldsymbol{\omega}_n(i)$, and the fixed set of measurements $\{\mathbf{x}_{nm}\}_{m=1}^{M}$. The unobserved variable $z_{nt} \in \{1, \ldots, M\}$ denotes the mixture component at node n, considering the given data $\{\mathbf{x}_{nm}\}_{m=1}^{M}$ as centers of M mixture components. The expectation step is limited with those local measurement data, but the maximization step with Newton's method is extended to 1) *local update* and 2) *averaging* steps for in-network processing. Similar with EM-based GMS algorithm, the whole mixture components at each n are shifted by varying a displacement vector $\boldsymbol{\omega}_n$. A conditional density model with $\boldsymbol{\omega}_n$ at node n will be used, which is given by

$$g_n(\mathbf{y}_{nt}|\boldsymbol{\omega}_n) = \frac{1}{M}\sum_{m=1}^{M} g_n(\mathbf{y}_{nt}|m, \boldsymbol{\omega}_n) \tag{4.23}$$

$$= \frac{1}{M}\sum_{m=1}^{M} \frac{1}{Z} K_h\left(\mathbf{y}_{nt} - \left(\mathbf{x}_{nm} - \boldsymbol{\omega}_n\right)\right). \tag{4.24}$$

where $\mathbf{y}_{nt}$ denotes an observed data at node n.

4.4.1 Expectation

Node n calculates the expectation of the log-likelihood function with respect to the posterior distribution of z_{nt}, given the observed data $\mathbf{y}_{nt}$ and the current estimate of $\boldsymbol{\omega}_n(i)$. The expectation step at each n can be expressed as

$$Q_n\big(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i)\big) = \sum_{t=1}^{T} E_{g_n(z_{nt}|\mathbf{y}_{nt},\boldsymbol{\omega}_n(i))}\big[\mathcal{L}_{nt}(\boldsymbol{\omega})\big]$$

$$= \sum_{t=1}^{T}\sum_{z_{nt}=1}^{M} g_n\big(z_{nt}|\mathbf{y}_{nt}, \boldsymbol{\omega}_n(i)\big) \log g_n(\mathbf{y}_{nt}, z_{nt}|\boldsymbol{\omega}). \tag{4.25}$$

Since $g_n(\mathbf{y}_{nt}, z_{nt}|\boldsymbol{\omega}) = g_n(z_{nt}|\boldsymbol{\omega})g_n(\mathbf{y}_{nt}|z_{nt}, \boldsymbol{\omega})$ and $g_n(z_{nt}|\boldsymbol{\omega}) = \frac{1}{M}$, it follows that

$$Q_n\big(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i)\big) = \sum_{t=1}^{T}\sum_{z_{nt}=1}^{M}\Big[g_n\big(z_{nt}|\mathbf{y}_{nt}, \boldsymbol{\omega}_n(i)\big)\log g_n(\mathbf{y}_{nt}|z_{nt}, \boldsymbol{\omega})\Big] + C,\ \forall n, \tag{4.26}$$

where $C = -\sum_{t=1}^{T}\sum_{z_{nt}=1}^{M} g_n\big(z_{nt}|\mathbf{y}_{nt}, \boldsymbol{\omega}_n(i)\big)\log M$ is independent of $\boldsymbol{\omega}$, and does not affect the maximization of $Q_n\big(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i)\big)$.

4.4.2 Maximization - Local Update and Averaging Steps

$Q_n\big(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i)\big)$ in (4.26) is not associated with the neighboring $Q_l\big(\boldsymbol{\omega}|\boldsymbol{\omega}_l(i)\big)$, where $l \in \mathbb{N}_n$, at node n. We consider distributed optimization for collaboration of node n and $\mathbb{N}_n$. Similar method was also implemented in diffusion adaptation algorithms such as [51, 114, 119]. The following global objective function is maximized:

$$\begin{aligned} Q^{\text{glob}}(\boldsymbol{\omega}|\boldsymbol{\omega}(i)) &\triangleq \sum_{n=1}^{N} Q_n(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i)) \\ &= Q_n(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i)) + \sum_{l\neq n} Q_l(\boldsymbol{\omega}|\boldsymbol{\omega}_l(i)) \end{aligned} \tag{4.27}$$

where $Q_l(\boldsymbol{\omega}|\boldsymbol{\omega}_l(i))$ is second-order differentiable and it is assumed there exists a $\boldsymbol{\omega}_l^*$ that maximizes $Q_l(\boldsymbol{\omega}|\boldsymbol{\omega}(i))$ at node l. By a second-order Taylor series expansion around $\boldsymbol{\omega}_l^*$, $Q_l(\boldsymbol{\omega}|\boldsymbol{\omega}_l(i))$ can be approximated as

$$\begin{aligned} Q_l(\boldsymbol{\omega}|\boldsymbol{\omega}_l(i)) &\approx Q_l(\boldsymbol{\omega}_l^*|\boldsymbol{\omega}_l(i)) + \nabla_{\boldsymbol{\omega}} Q_l(\boldsymbol{\omega}_l^*|\boldsymbol{\omega}_l(i))^T(\boldsymbol{\omega} - \boldsymbol{\omega}_l^*) \\ &\quad + \frac{1}{2}(\boldsymbol{\omega} - \boldsymbol{\omega}_l^*)^T \nabla_{\boldsymbol{\omega}}^2 Q_l(\boldsymbol{\omega}_l^*|\boldsymbol{\omega}_l(i))(\boldsymbol{\omega} - \boldsymbol{\omega}_l^*) \\ &= \big\|\boldsymbol{\omega} - \boldsymbol{\omega}_l^*\big\|_{\Gamma_l}^2 + Q_l(\boldsymbol{\omega}_l^*|\boldsymbol{\omega}_l(i)) \end{aligned} \tag{4.28}$$

where $\Gamma_l = \frac{1}{2}\nabla_{\boldsymbol{\omega}}^2 Q_l(\boldsymbol{\omega}_l^*|\boldsymbol{\omega}_l(i))$. The second term was canceled because $\nabla_{\boldsymbol{\omega}} Q_l(\boldsymbol{\omega}_l^*|\boldsymbol{\omega}_l(i)) = 0$. Note that $Q_l(\boldsymbol{\omega}_l^*|\boldsymbol{\omega}_l(i))$ is independent of $\boldsymbol{\omega}$ and can be considered as a constant. The global objective function $Q^{\text{glob}}(\boldsymbol{\omega}|\boldsymbol{\omega}(i))$ in (4.27) can be rewritten as

$$Q^{\text{glob}'}(\boldsymbol{\omega}|\boldsymbol{\omega}(i)) = Q_n(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i)) + \sum_{l\neq n}\big\|\boldsymbol{\omega} - \boldsymbol{\omega}_l^*\big\|_{\Gamma_l}^2. \tag{4.29}$$

Optimization of $Q^{\text{glob}'}(\boldsymbol{\omega}|\boldsymbol{\omega}(i))$ still requires that every node n has to access the global information $\boldsymbol{\omega}_l^*$ and Γ_l for all $l \neq n$. The diffusion strategies have been admitted to approximate $Q^{\text{glob}'}(\boldsymbol{\omega}|\boldsymbol{\omega}_l(i))$ [51, 114, 119] for distributed implementation. First, the approximation confines the sum in (4.29) to be processed in neighborhood of node n. The distributed version of objective function becomes

$$Q_n^{\text{dist}}(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i)) = Q_n(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i)) + \sum_{l \in \mathbb{N}_n} \left\| \boldsymbol{\omega} - \boldsymbol{\omega}_l^* \right\|_{\Gamma_l}^2. \tag{4.30}$$

where $\mathbb{N}_n$ denotes the set of neighboring nodes of n excluding n itself. Note that the first term of RHS in (4.30) is the expectation at node n defined in (4.26) and the second term is associated with neighboring nodes $\mathbb{N}_n$. Second, the approximation replaces the unknown $\boldsymbol{\omega}_l^*$ with an intermediate estimate $\boldsymbol{\omega}_l$ at node l. Then, the distributed objective function in (4.30) can be rewritten as

$$Q_n^{\text{dist}'}(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i)) = Q_n(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i)) + \sum_{l \in \mathbb{N}_n} \left\| \boldsymbol{\omega} - \boldsymbol{\omega}_l \right\|_{\Gamma_l}^2. \tag{4.31}$$

The maximization step utilizes Newton's method with the gradient vector and the Hessian matrix of $Q_n^{\text{dist}'}(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i))$ which are given by respectively

$$\nabla_{\boldsymbol{\omega}} Q_n^{\text{dist}'}(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i)) = \nabla_{\boldsymbol{\omega}} Q_n(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i)) + \sum_{l \in \mathbb{N}_n} \nabla_{\boldsymbol{\omega}}^2 Q_l\big(\boldsymbol{\omega}_l|\boldsymbol{\omega}_l(i)\big)\big(\boldsymbol{\omega} - \boldsymbol{\omega}_l\big), \tag{4.32}$$

$$\nabla_{\boldsymbol{\omega}}^2 Q_n^{\text{dist}'}(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i)) = \nabla_{\boldsymbol{\omega}}^2 Q_n(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i)) + \sum_{l \in \mathbb{N}_n} \nabla_{\boldsymbol{\omega}}^2 Q_l\big(\boldsymbol{\omega}_l|\boldsymbol{\omega}_l(i)\big). \tag{4.33}$$

Consider the density model $g_n(\mathbf{y}_{nt}|z_{nt}, \boldsymbol{\omega})$ in (4.26) as a Gaussian function of $\boldsymbol{\omega}$ with mean vector $\mathbf{x}_{nz_{nt}} - \mathbf{y}_{nt}$ and covariance $h^2\mathbf{I}$ given the unobserved membership z_{nt} of mixture model. We can replace z_{nt} with index $m \in \{1, \ldots, M\}$ indicating m-th mixture component, and have equivalently

$$g_n(\mathbf{y}_{nt}|z_{nt}, \boldsymbol{\omega}) = g_n(\mathbf{y}_{nt}|m, \boldsymbol{\omega}). \tag{4.34}$$

With the density model (4.34) in (4.23)-(4.24) and the Gaussian kernel of (4.2), we have

$$\begin{aligned}\nabla_{\boldsymbol{\omega}} \log g_n(\mathbf{y}_{nt}|m, \boldsymbol{\omega}) &= \nabla_{\boldsymbol{\omega}} \log \left[\frac{1}{Z} K_h \left(\mathbf{x}_{nm} - \mathbf{y}_{nt} - \boldsymbol{\omega} \right) \right] \\ &= \frac{1}{h^2} \big(\mathbf{x}_{nm} - \mathbf{y}_{nt} - \boldsymbol{\omega} \big). \end{aligned} \tag{4.35}$$

From (4.26) and (4.35), the gradient vector and Hessian matrix of $Q_n(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i))$ is given by respectively

$$\nabla_{\boldsymbol{\omega}} Q_n(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i)) = \sum_{t=1}^{T} \sum_{m=1}^{M} \Big[g_n(m|\mathbf{y}_{nt}, \boldsymbol{\omega}_n(i)) \frac{1}{h^2} \big(\mathbf{x}_{nm} - \mathbf{y}_{nt} - \boldsymbol{\omega} \big) \Big], \tag{4.36}$$

$$\nabla^2_{\boldsymbol{\omega}} Q_n(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i)) = -\sum_{t=1}^{T} \sum_{m=1}^{M} g_n(m|\mathbf{y}_{nt}, \boldsymbol{\omega}_n(i)) \frac{1}{h^2}. \tag{4.37}$$

The recursive update equation for Newton's method can be written as

$$\boldsymbol{\omega}_n(i+1) = \boldsymbol{\omega}_n(i) - \alpha \big[\nabla^2_{\boldsymbol{\omega}} Q_n^{\text{dist}'}(\boldsymbol{\omega}_n(i)|\boldsymbol{\omega}_n(i)) \big]^{-1} \nabla_{\boldsymbol{\omega}} Q_n^{\text{dist}'}(\boldsymbol{\omega}_n(i)|\boldsymbol{\omega}_n(i)) \tag{4.38}$$

where α denotes a small step-size for Newton's method. We assume that the Hessian matrices $\nabla^2_{\boldsymbol{\omega}} Q_n(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i))$ for all n are not significantly different. This assumption is reasonable because at every iteration i all the nodes combine their estimate $\boldsymbol{\omega}_n(i)$ with neighboring estimates and the density model g_n was induced from the same distribution. Thus, the gradient vector and the Hessian matrix in (4.32) and (4.33) respectively can be approximated as

$$\nabla_{\boldsymbol{\omega}} Q_n^{\text{dist}'}(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i)) \approx \nabla_{\boldsymbol{\omega}} Q_n(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i)) + \nabla^2_{\boldsymbol{\omega}} Q_n\big(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i)\big) \sum_{l \in \mathbb{N}_n} \big(\boldsymbol{\omega} - \boldsymbol{\omega}_l \big), \tag{4.39}$$

$$\nabla^2_{\boldsymbol{\omega}} Q_n^{\text{dist}'}(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i)) \approx \big(|\mathbb{N}_n| + 1 \big) \nabla^2_{\boldsymbol{\omega}} Q_n(\boldsymbol{\omega}|\boldsymbol{\omega}_n(i)) \tag{4.40}$$

where $\big|\mathbb{N}_n\big|$ denotes the size of neighboring nodes at n. Substituting (4.39) and (4.40) into (4.38), we have

$$\boldsymbol{\omega}_n(i+1) = \boldsymbol{\omega}_n(i) - \frac{\alpha}{|\mathbb{N}_n+1|}\Big[\nabla^2_{\boldsymbol{\omega}} Q_n\big(\boldsymbol{\omega}_n(i)|\boldsymbol{\omega}_n(i)\big)\Big]^{-1}\nabla_{\boldsymbol{\omega}} Q_n\big(\boldsymbol{\omega}_n(i)|\boldsymbol{\omega}_n(i)\big) - \frac{\alpha}{|\mathbb{N}_n+1|}\sum_{l\in\mathbb{N}_n}\big(\boldsymbol{\omega}_n(i)-\boldsymbol{\omega}_l\big). \tag{4.41}$$

4.4.3 Distributed Mean-Shift Algorithm

Now we describe a two-steps algorithm for (4.41). Define $\mathbf{q}_n(i)$ as

$$\mathbf{q}_n(i) \triangleq \big[\nabla^2_{\omega} Q_n(\boldsymbol{\omega}_n(i)|\boldsymbol{\omega}_n(i))\big]^{-1}\nabla_{\omega} Q_n(\boldsymbol{\omega}_n(i)|\boldsymbol{\omega}_n(i)). \tag{4.42}$$

Also, replace $\boldsymbol{\omega}_l$ in (4.41) with $\boldsymbol{\psi}_l(i)$ that is an intermediate estimate available at node l at iteration i. Moreover, for convergence of the estimated state sequences $\{\boldsymbol{\omega}_n(i)\}_{i\geq 0}\forall n$, consider decreasing step sizes $\alpha(i)$ and $\eta(i)$ replacing $\frac{\alpha}{|\mathbb{N}_n+1|}$ in (4.41). The conditions for $\alpha(i)$ and $\eta(i)$ will be described later in this section. The distributed mode estimation algorithm at every node n can be expressed as

$$\boldsymbol{\psi}_n(i) = \boldsymbol{\omega}_n(i) - \alpha(i)\mathbf{q}_n(i) \tag{4.43}$$

$$\boldsymbol{\omega}_n(i+1) = \boldsymbol{\psi}_n(i) - \eta(i)\sum_{l\in\mathbb{N}_n}\big(\boldsymbol{\psi}_n(i)-\boldsymbol{\psi}_l(i)\big). \tag{4.44}$$

In wireless sensor networks, the intermediate state vector $\boldsymbol{\psi}_n(i)$ in (4.44) is perturbed by communication random noise. In the presence of wireless communication noise, the mode estimation method (4.43) - (4.44) can be rewritten as

$$\boldsymbol{\psi}_n(i) = \boldsymbol{\omega}_n(i) - \alpha(i)\mathbf{q}_n(i) \tag{4.45}$$

$$\boldsymbol{\omega}_n(i+1) = \boldsymbol{\psi}_n(i) - \eta(i)\sum_{l\in\mathbb{N}_n}\Big[\boldsymbol{\psi}_n(i)-\boldsymbol{\psi}_l(i)-\boldsymbol{\xi}_{nl}(i)\Big] \tag{4.46}$$

where $\boldsymbol{\xi}_{nl}(i)$ denotes random noise in the communication link from node l to n at time i, assuming its distribution is zero-mean Gaussian. The step-sizes $\alpha(i)$ and $\eta(i)$ decrease to zero, as the iteration i goes to infinity. However, the decaying rate should not be too fast. Moreover, $\alpha(i)$ decreases faster than $\eta(i)$ so that a consensus can be achieved. Similar methods for those step-sizes were used in [36] for a mixed-time scale stochastic approximation. Assumption 1 describes conditions for such step-sizes. The step-sizes can be in the following forms:

$$\alpha(i) = \frac{\alpha_0}{(i+1)^{\tau_1}} \quad \text{and} \quad \eta(i) = \frac{\eta_0}{(i+1)^{\tau_2}}, \tag{4.47}$$

where $0.5 < \tau_1, \tau_2 \leq 1$ and $0 < \tau_1 - \tau_2 < 0.5$ for $i = 0, 1, \ldots$ and positive constants α_0 and η_0. The step-sizes in (4.47) satisfy the persistence condition, which is given by

$$\alpha(i) > 0, \ \sum_{i=0}^{\infty} \alpha(i) = \infty, \ \sum_{i=0}^{\infty} \alpha^2(i) < \infty, \tag{4.48}$$

$$\eta(i) > 0, \ \sum_{i=0}^{\infty} \eta(i) = \infty, \ \sum_{i=0}^{\infty} \eta^2(i) < \infty. \tag{4.49}$$

4.4.4 Mean-Shift Vector $-\mathbf{q}_n(i)$

Substituting (4.36) and (4.37) into (4.42), we can describe the vector $\mathbf{q}_n(i)$ of the distributed scheme in (4.43) as

$$\mathbf{q}_n(i) = \frac{\sum_{t=1}^{T} \sum_{m=1}^{M} g_n(m|\mathbf{y}_{nt}, \boldsymbol{\omega}_n(i)) \big(\boldsymbol{\omega}_n(i) - \mathbf{x}_{nm} + \mathbf{y}_{nt}\big)}{\sum_{t'=1}^{T} \sum_{m'=1}^{M} g_n(m'|\mathbf{y}_{nt'}, \boldsymbol{\omega}_n(i))}. \tag{4.50}$$

Recall that $g_n(m|\mathbf{y}_{nt}, \boldsymbol{\omega}_n(i))$ is the posterior probability for the m-th mixture component, given $\mathbf{y}_{nt}$ and $\boldsymbol{\omega}_n(i)$. By selecting the dataset $\{\mathbf{y}_{nt}\}_{t=1}^{T}$ as the origin (i.e., $\mathbf{y}_{nt} = \mathbf{0}$ and $T = 1$), we have the equivalent form:

$$g_n(m|\mathbf{y}_{nt} = \mathbf{0}, \boldsymbol{\omega}_n(i)) = \frac{K_h(\boldsymbol{\omega}_n(i) - \mathbf{x}_{nm})}{\sum_{m'=1}^{M} K_h(\boldsymbol{\omega}_n(i) - \mathbf{x}_{nm'})} \tag{4.51}$$

where the derivation of (4.51) is available in the Appendix C. Substituting (4.51) into (4.50) with setting $\mathbf{y}_{nt} = \mathbf{0}$ and $T = 1$, we can further simplify $\mathbf{q}_n(i)$ as

$$\mathbf{q}_n(i) = \boldsymbol{\omega}_n(i) - \frac{\sum_{m=1}^{M} K_h\big(\boldsymbol{\omega}_n(i) - \mathbf{x}_{nm}\big)\mathbf{x}_{nm}}{\sum_{m'=1}^{M} K_h\big(\boldsymbol{\omega}_n(i) - \mathbf{x}_{nm'}\big)}. \tag{4.52}$$

Consider the second term of RHS in (4.52) as conditional mean of $\{\mathbf{x}_{nm}\}_{m=1}^{M}$, where the weight for $\mathbf{x}_{nm}$ is determined by distance between $\boldsymbol{\omega}_n(i)$ and $\mathbf{x}_{nm}$ at time i. We can call $-\mathbf{q}_n(i)$ of (4.52) a *mean-shift* vector at time i.

In summary, the distributed mode estimation schemes are described in Algorithm I. When $\boldsymbol{\xi}_{nl}(i) = 0$ for all n, l, i, the algorithm is in the absence of communication noise. Thus, the step size $\eta(i)$ can be set to a positive constant η_0.

Algorithm 3 Distributed Man-shift Algorithm

```
Initialization: $\boldsymbol{\omega}_n(0)$, $\mathbf{q}_n(0)$, $h$, $\alpha(0)$, $\tau_1$, $\eta(0)$, and $\tau_2$ $\forall n$.
for $i \geq 1$ do
  Mean-shift vector: each node $n$ computes $\mathbf{q}_n(i)$ in (4.52).
  Local update step: each node $n$ calculates
    $\boldsymbol{\psi}_n(i) = \boldsymbol{\omega}_n(i) - \alpha(i)\mathbf{q}_n(i)$
  Averaging step: each node $n$ calculates
    $\boldsymbol{\omega}_n(i+1) = \boldsymbol{\psi}_n(i) - \eta(i)\sum_{l\in\mathbb{N}_n}\big[\boldsymbol{\psi}_n(i) - \boldsymbol{\psi}_l(i) - \boldsymbol{\xi}_{nl}(i)\big]$
  Time instant update: $i \rightarrow i+1$.
end for
When $\boldsymbol{\xi}_{nl}(i) = 0$, the link from $l$ to $n$ is noise-free at $i$.
```

4.5 Numerical Experiments

In this section we demonstrate the distributed mode estimation algorithm. Consider a distributed sensor network, illustrated in Fig. 4.1, which is a connected graph with $N = 50$ where the graph's connectivity is characterized by $\lambda_{2,\mathbf{L}} = 2.2815$. Each node n has M measurements $\{x_{nm}\}_{m=1}^{M}$, where $D = 1$, taken from log-normal distribution $\ln \mathcal{N}(1, 0.25)$. The total number of measurements is MN from all nodes. Kernel size h was selected by (4.3). Fig. 4.2 is a realization of MN measurement from the lognormal distributed data and its centralized KDE with the properly selected kernel size h. One can observe that the mode $\gamma = 2.4992$ for the measurement data and the average as a centralized measure of data should be biased because of the right heavy-tail of the distribution. For the step sizes $\alpha(i)$ and $\eta(i)$ in (4.47), we considered $\tau_1 = 1$ and $\tau_2 = 0.505$. The initial step sizes α_0 and η_0 are respectively set to 1 and $0.5/d_{\max}$ where $d_{\max}$ denotes the maximum degree of graph network. The noise in wireless communication links is assumed to be Gaussian random variables with zero mean and different values of variance σ_ξ^2 for evaluation. Let $\boldsymbol{\omega}(i)$ be a vector of size $ND \times 1$ for the Algorithm 3, i.e. $\boldsymbol{\omega}(i) = \left[\boldsymbol{\omega}_1(i)^T, \ldots, \boldsymbol{\omega}_N(i)^T\right]^T$. If $D = 1$, $\boldsymbol{\omega}(i) = \left[\omega_1(i), \ldots, \omega_N(i)\right]^T$. The initial state $\boldsymbol{\omega}(0)$ are randomly selected at each node n. In Fig. 4.3 and Fig. 4.4, we can observe that all the states at nodes converge toward a single estimate of mode in the presence and absence of communication noise, respectively.

The kernel density estimator (KDE) of (4.1) could be used to estimate the mode if MN measurements were available in a centralized location. We compare the mean-square-deviation (MSD) between the mode estimations of Algorithm I and the centralized method. The MSD is defined as

$$\mathrm{MSD}(i) = \frac{1}{ND} E\left[\left\|\boldsymbol{\omega}(i) - \gamma \mathbf{1}\right\|^2\right] \tag{4.53}$$

where γ is obtained by the centralized KDE of (4.1) and $\mathbf{1}$ denotes a $ND \times 1$ vector with 1 of all elements. We assume the centralized method experiences noisy communication, when data are transmitted from individual nodes and the centralized location. After receiving all the measurements, the GMS algorithm [107–109] is processed at

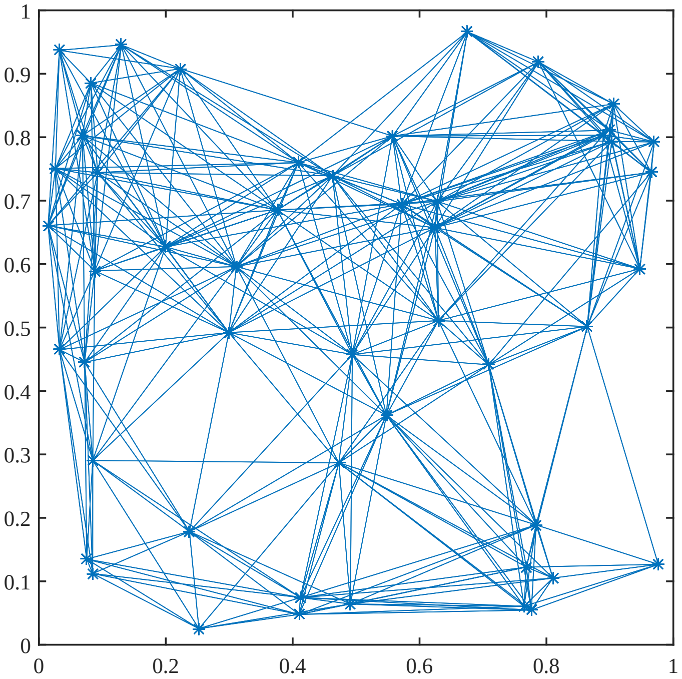


Figure 4.1: A Graph for Distributed Sensor Network ($N = 50$) Where the Graph's Connectivity Is Characterized by $\lambda_2(\mathbf{L}) = 2.2815$.

the center. The expectation is evaluated over 200 realizations of the measurement and communication noise (but the PDFs are fixed). The mode γ was obtained by the KDE of (4.1) with the kernel size h by (4.3).

In the absence of communication noise (i.e. $\boldsymbol{\xi}_{nl}(i) = \mathbf{0}$ for all n, l, and i), we can adopt a constant step size $\eta(i) = \eta_0$ for all i. Fig. 4.5 compares the MSDs of different values for the step-size $\alpha(i)$, when the other step size $\eta(i)$ is fixed as $\frac{1}{2d_{\max}}$ for all i. When we adopt the decaying step size $\alpha(i)$ with $\tau_1 = 0.505$, it is expected that the MSD converges toward zero as $i \rightarrow \infty$. We can also observe that if the constant step size $\alpha(i)$ becomes smaller (from 0.1 to 0.05), the MSD also decreases at the cost of slower convergence rate rate.

Figure 4.6 compares the MSD performance between the distributed mean-shift algorithm and centralized GMM fitted by EM algorithm in the absence of commu-

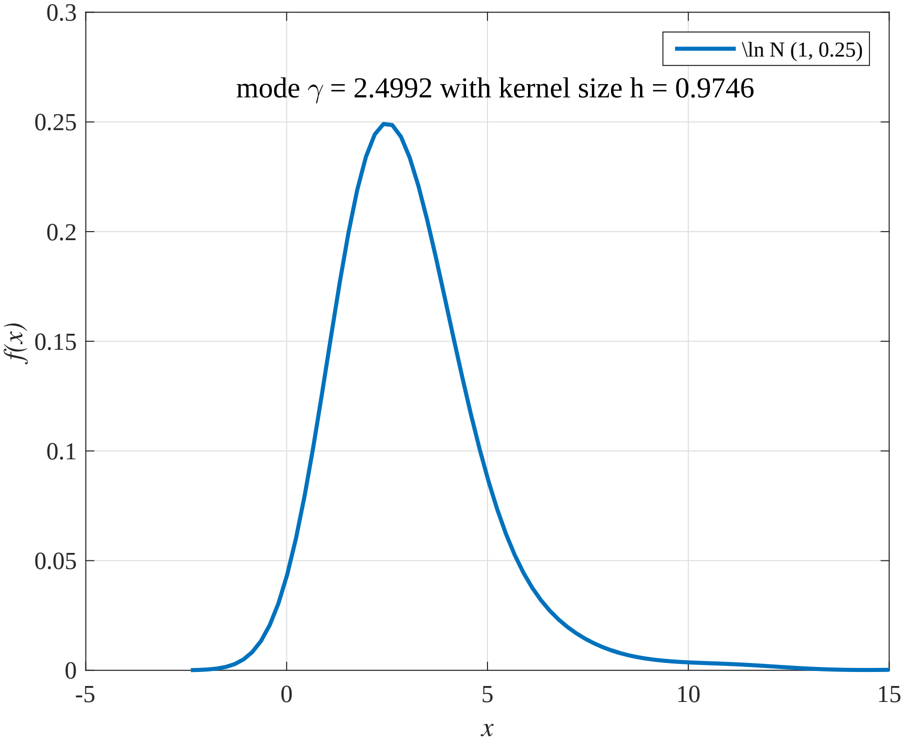


Figure 4.2: An Example of KDE from Log-normal Distribution with $\mu = 1$ and $\sigma = 0.5$. The Kernel Size $h = 0.9746$ by Silverman's Rule of Thumb (4.3). $N = 50$ and $M = 20$.

nication noise. The data distribution is lognormal with mean $\mu = 1$ and variance $\sigma^2 = 0.25$. After the centralized GMM is estimated by EM algorithm with specified number of Gaussian mixture components, the closest component to the mode was used to compute MSD metric. Among four scenarios - from 1 to 4 components, GMM with 1 component was the worst to estimate the mode because of skewed distribution, but GMM with 2 components show the best MSD performance for the given lognormal distributed data. The distributed mean-shift algorithm also converges to the best MSD, as iteration i increases. Not that the proposed method only needs to estimate a single parameter, whereas the centralized GMM requires three parameters for each component - mixing probability, mean vector, and covariance matrix.

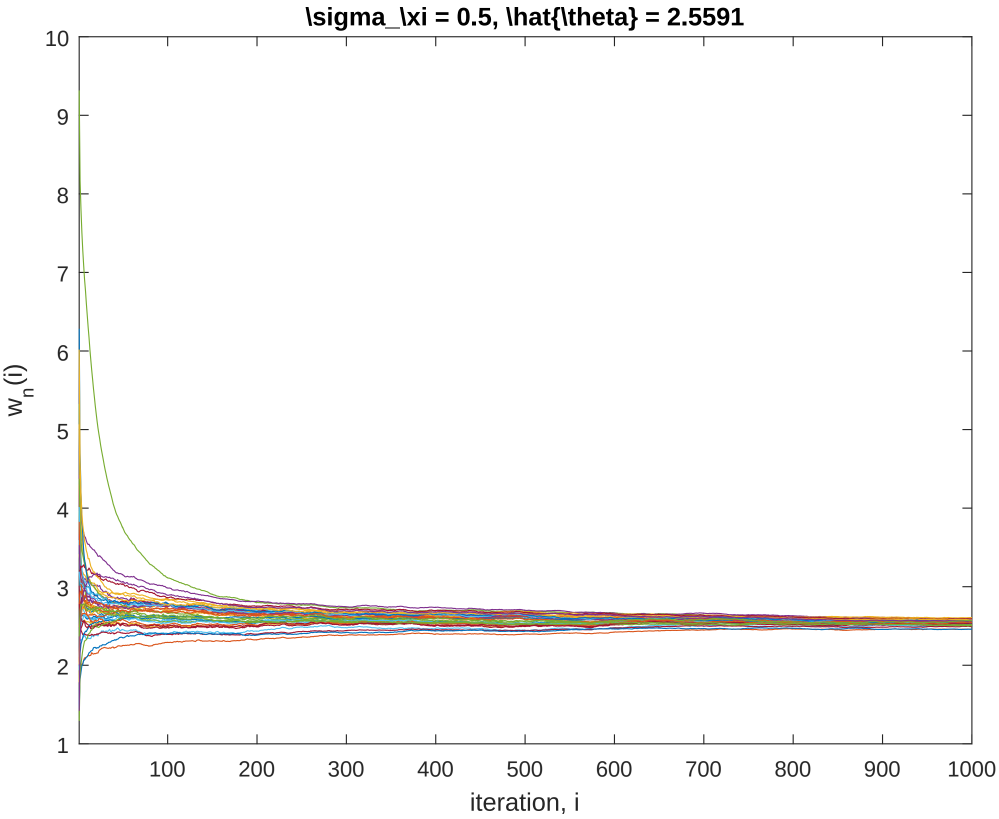


Figure 4.3: Distributed Mode Estimation in the Presence of Communication Noise ($\sigma_\xi = 0.5$). The Mode Is $\gamma = 2.5591$ Obtained by the KDE (4.1).

In Fig. 4.8, we evaluate the MSD for various values of measurement size M in the presence communication noise where $\sigma_\xi = 0.5$. For each case of M, the kernel size h is computed by (4.3). As M increases, h is reduced. With larger number of measurement data, MSD can be reduced due to statistical sufficiency for building the KDE f. However, too large M results in too small h. The latter case may not be desirable because KDE itself has large variance for the mode estimation with too small h. On contrary, too large h is responsible for large bias of KDE.

Figure 4.7 compares the MSD between Algorithm 3 (distributed) and the centralized method. It shows that MSDs of the centralized method have different levels at $\sigma_\xi = 1$ and $\sigma_\xi = 0.1$ due to the noisy transmission. For the larger noise vari-

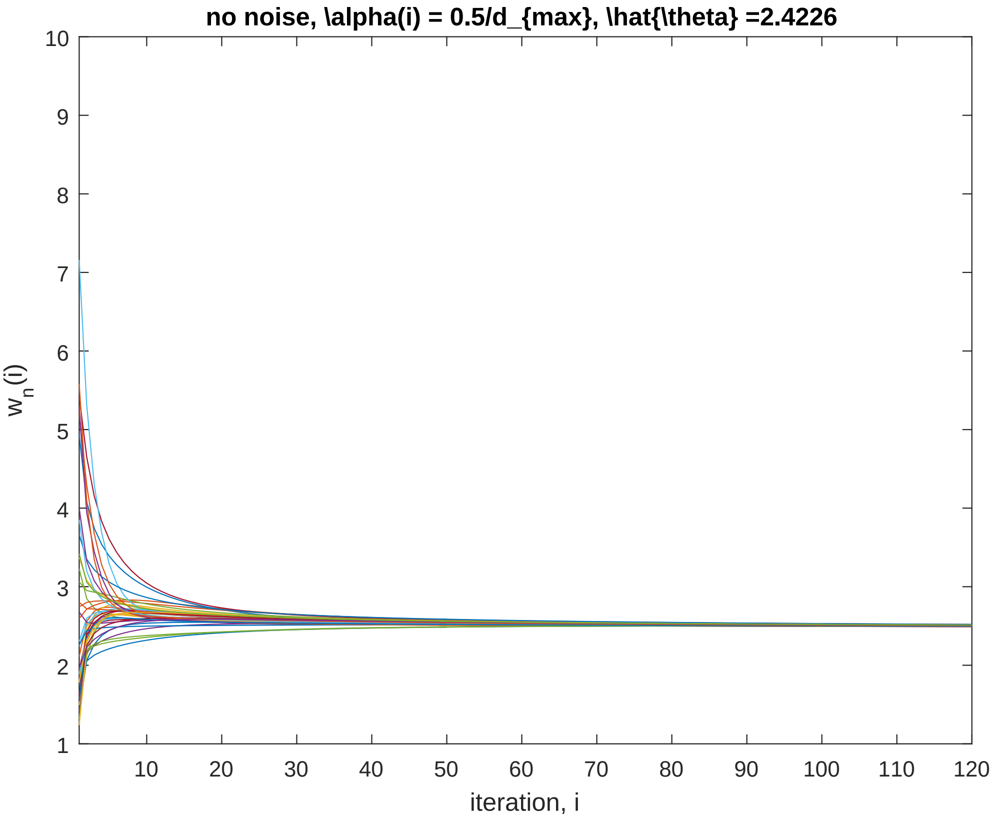


Figure 4.4: Distributed Mode Estimation in the Absence of Communication Noise. $\gamma = 2.4226$ Is the Mode Obtained by the KDE (4.1).

ance in communication links, the centralized GMS also results in larger MSDs when i is sufficiently large. Note that the received measurement data at the center were already corrupted by communication noise. Thus the noise variance is responsible for the different levels of MSDs in Fig. 4.7. We can also observe that MSDs of the distributed Algorithm 3 decrease, as $i \to \infty$. The decaying step size (i.e. $\alpha(i) \to 0$) is responsible for mitigating the communication noise, as iteration i increases.

4.6 Application to Finding Densest Deployment

We apply the Algorithm 3 to the problem of the estimation of densest sensor deployment. One may have a question: ”Where is the densest location over a sensor

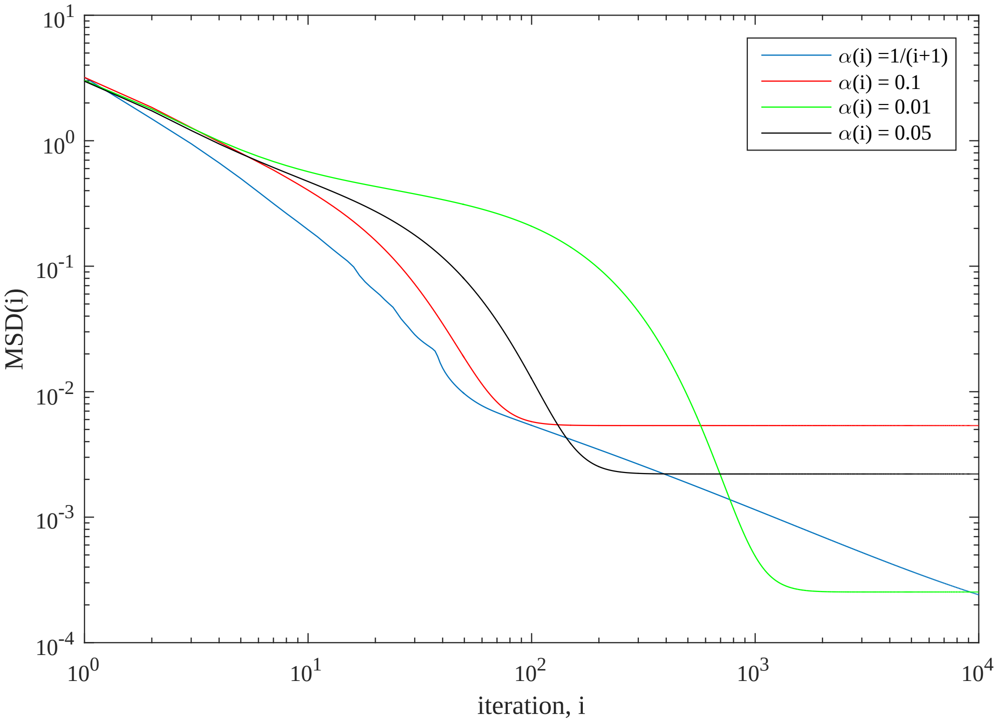


Figure 4.5: MSD for Comparison Between (Distributed) Algorithm I and Centralized GMS Algorithm with Various Step-sizes $\alpha(I)$ in the Absence of Communication Noise.

network?" The mode estimation algorithm can answer this question. Suppose there are N sensor nodes deployed in arbitrary location. Each node maintains its own location information by GPS. Let $\mathbf{x}_n$ be the vector of node n's 2D location information (it can be easily extended to 3D):

$$\mathbf{x}_n = \left[x_n^{(1)}, x_n^{(2)}\right]^T \tag{4.54}$$

where $M = 1$ is assumed because each node only maintains its own location measurement. In Fig. 4.9, there are $N = 50$ sensors in 2D space. The location for $x^{(1)}$ and $x^{(2)}$ axes were generated by lognormal distributions $\ln \mathcal{N}(0, 0.49)$ and $\ln \mathcal{N}(0, 0.25)$ respectively, and then normalized by $\max(x^{(1)})$ and $\max(x^{(2)})$ for each axis. The contour in Fig. 4.9 was generated by KDE of (4.1) with setting $M = 1$, in order to visualize the density location. Each node n randomly initializes its state $\boldsymbol{\omega}_n(0)$. Fig. 4.9 shows

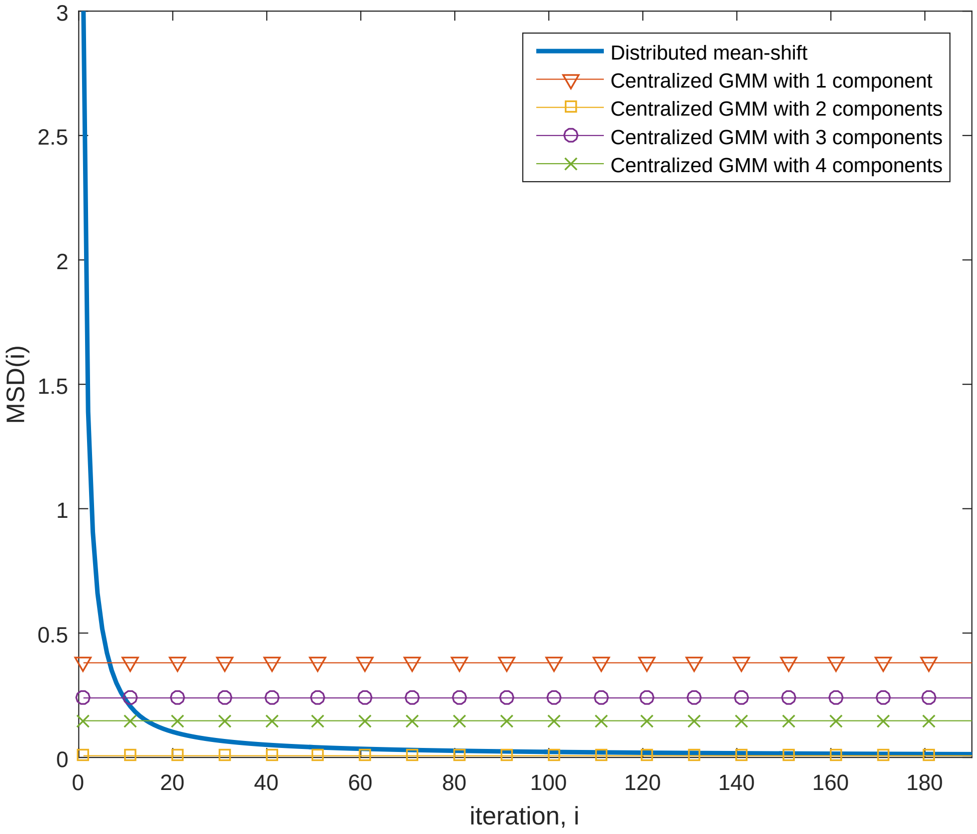


Figure 4.6: MSD for Comparison Between (distributed) Algorithm 3 and Centralized GMM Fitted by EM Algorithm in the Absence of Communication Noise.

that five different trajectories (i.e. $\boldsymbol{\omega}_n(i)$, $n = 2, 14, 26, 38, 50$) approach the densest location of sensor deployment, as iteration i increases. The kernel size $h = 0.1125$ was selected by (4.3). For the step sizes $\alpha(i)$ and $\eta(i)$ in (4.47), we used $\alpha(0) = 0.05$, $\tau_1 = 1$, $\eta(0) = 0.5/d_{\max}$, and $\tau_2 = 0.505$. In the presence of communication noise, $\sigma_\xi = 0.5$ was considered in the simulation.

4.7 Conclusion

We have shown a distributed mode estimation scheme based on algorithms of Gaussian mean-shift (GMS) and distributed extension of Expectation-Maximization

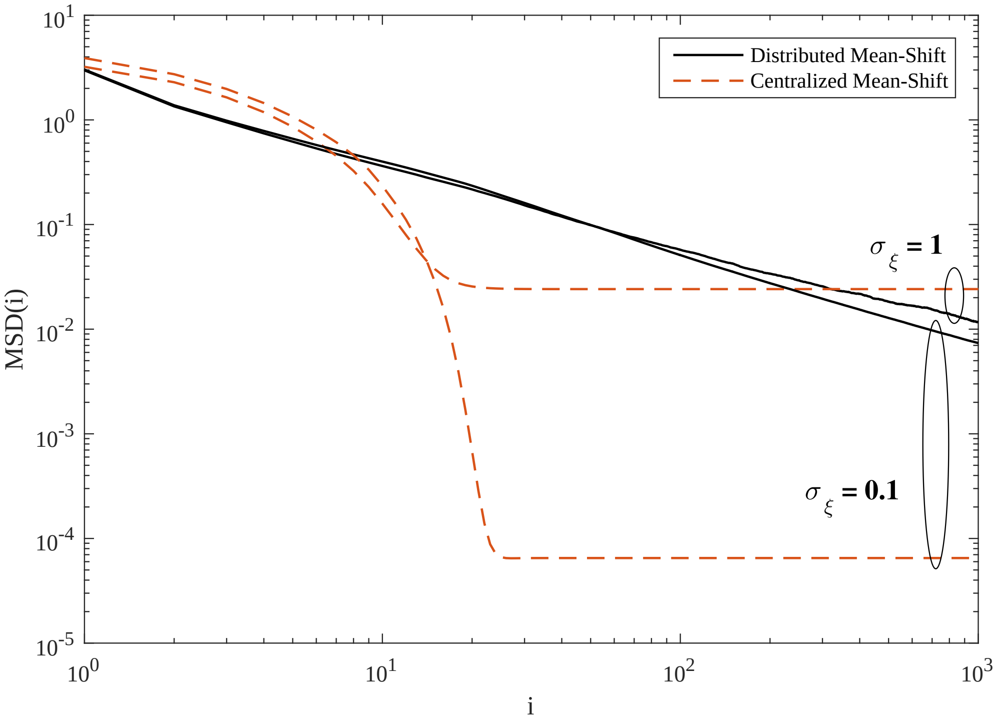


Figure 4.7: MSD for Comparison Between (Distributed) Algorithm I and Centralized GMS Algorithm in the Presence of Communication Noise.

(EM). This scheme estimates a mode, as a *central tendency* of unimodal data distribution regardless of skewness, in distributed wireless sensor networks. While each node maintains the expectation step based on local information, distributed cooperation is performed over the network, which results in updating mean-shift vector. The mean-shift mode estimation method consists of two steps - local update and averaging steps, as described in Algorithm 3. Simulation results demonstrate that the estimated states converge toward a consensus of the mode estimation in the presence and absence of communication noise. We compared performance between the distributed and centralized scenarios. The proposed algorithm was also compared with the centralized GMM to estimate a mode of skewed data distribution. Different conditions such as step-sizes and measurement size M were also evaluated in the presence and absence of communication noise.

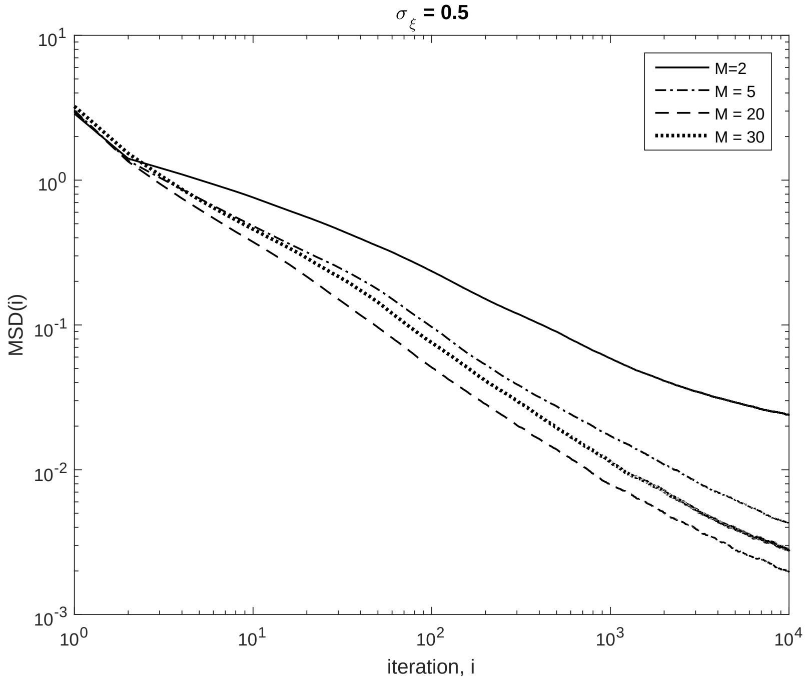


Figure 4.8: MSD for Various Values of m at Each Node n in (Distributed) Algorithm I in the Presence of Communication Noise ($\sigma_\xi = 0.5$).

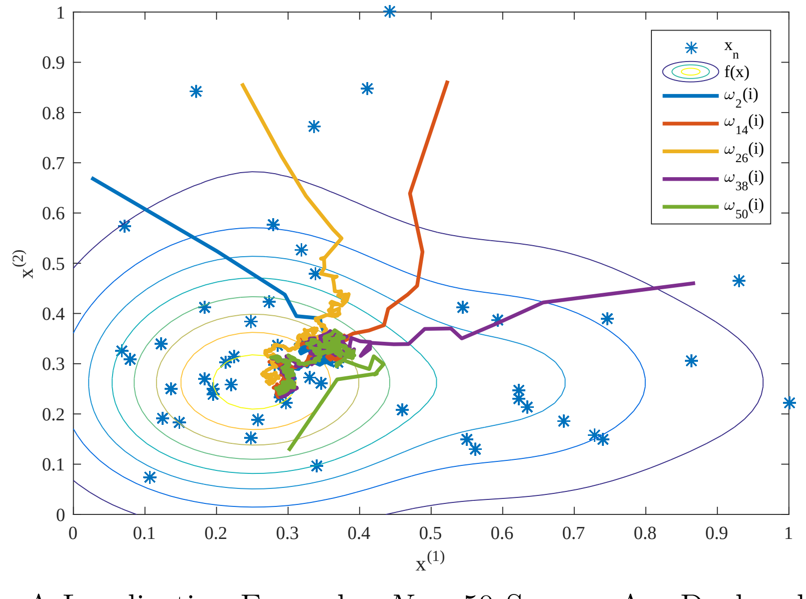


Figure 4.9: A Localization Example. $N = 50$ Sensors Are Deployed, Maintaining the Location Information $\mathbf{x}_n \in \mathbb{R}^2$ at Each Node n. Algorithm 3 Finds the *Densest Location* of the Deployment.

Chapter 5

DISTRIBUTED DENSITY ESTIMATION FOR MODE CONSENSUS

In this chapter, we present distributed density estimation methods to achieve mode consensus in wireless sensor networks. This chapter is different with Chapter 4 in that here we estimate densities of a set of grid points, whereas in Chapter 4 the mean-shift algorithm was considered.

5.1 Mode as a Measure of Central Tendency

There are several mode estimation methods presuming that all the data are analyzed in a centralized location. Half-sample mode (HSM) method iteratively finds the shortest interval that contains the half number of samples from the data. As the number of iterations increases, the interval is reduced. And finally only two samples are remained representing the mean of them is the estimated mode. This method may provide the estimate close to the mode because minimizing the interval widths with a constant number of data samples is equivalent with maximizing the empirical density [65]. Similar method is the half-range mode (HRM) method that iteratively finds modal intervals selecting the half-interval that contains the larger number of samples [65]. Several other methods are proposed in [59, 60, 64, 65]. The fundamental idea of these estimators is to find the *densest* region of data distribution. Of course the mode can be estimated by obtaining the density function and finding the location that maximizes the density. The well-known nonparametric method is parzen's kernel density estimator [105]. Since the kernel size may affect the accuracy of the mode estimation, it can be decided by Silverman's rule [106]. These methods for the mode estimate have been widely applied in modern science for data analysis [57–59]. Asymptotic convergence is analyzed with the kernel density estimator [105, 120, 121] where the mode estimator is shown as a consistent estimator. The more general and practical approach for multiple modes is the mean-shift algorithms [107–111]. These methods seek multiple modes, although every sample converges to one of the multiple modes, by recursively updating the gradient ascent of the kernel density estimator which is required to be smooth to ascend. The mean-shift is useful for clustering algorithms because each mode represents different clusters. But it is difficult to analyze whether the estimated modes are consistent with the true modes or not.

When it comes to decentralized methods, there are not many works for directly related to the distributed estimation of the mode. Some of the related works are to estimate the density function from data in a distributed way. Reference [112] proposed the distributed expectation-maximization (EM) algorithm to parametrically estimate the density function with Gaussian mixture model (GMM). Reference [113] used the average consensus filter to estimate the parametric density function. However, the parametric methods may not be suitable to estimate the mode when data distribution does not consist of Gaussian mixtures or when the distribution is highly skewed.

5.2 System Model

Consider a distributed sensor network with N nodes each with M measurements $\{x_{nm}\}_{m=1}^{M}$, where x_{nm} is a scalar and $n = 1, 2, \ldots, N$, $M \geq 1$, drawn from a distribution. The graph nodes are sensors and the undirected edges are communication links. It is assumed that the M measurements data for each n are drawn from independent and identical distribution. Every sensor node wishes to estimate the central tendency of data over the entire sensor network, although each node maintains only limited size of measurements.

The nonparametric *Parzen*'s kernel density estimator f can be described as

$$\begin{aligned} f(z) &= \frac{1}{N}\sum_{n=1}^{N}\left[\frac{1}{M}\sum_{m=1}^{M}\frac{1}{Z}K_h\left(z - x_{nm}\right)\right] \\ &\triangleq \frac{1}{N}\sum_{n=1}^{N} y_{nz} \end{aligned} \tag{5.1}$$

where y_{nz} denotes the locally estimated density that depends on the distance between z and $\{x_{nm}\}_{m=1}^{M}$ at node n and the kernel $K_h(\cdot)$ is assumed the commonly used Gaussian kernel:

$$K_h(x) = \exp\left(-\frac{1}{2h^2}|x|^2\right) \tag{5.2}$$

with the normalization term $Z = \sqrt{2\pi}h$ where h denotes the size (a.k.a. bandwidth) of the kernel. We assume that the network size N is fixed and known to every node. The link connections between nodes are static and noiseless. It is also assumed that the density function f is bounded and twice continuously differentiable. Also note that the Gaussian kernel function in (5.2) is bounded, symmetric and translation invariant (a.k.a. shift invariant) satisfying

$$\int \kappa_h(u)du = 1 \quad \text{and} \quad \int u^2\kappa_h(u)du > 0. \tag{5.3}$$

5.3 Problem Statement

We wish to find the location of continuous variable z (denoting as θ) that maximizes the global density estimator $f(z)$, which can be defined by

$$\theta = \underset{-\infty<z<\infty}{\operatorname{argmax}} f(z). \tag{5.4}$$

To obtain θ, one can simply collect all the sets of measurements $\{x_{nm}\}_{m=1}^{M}$ at a fusion center and repeatedly examine $-\infty < z < \infty$ whether $f(z)$ is the maximum or not. However, such a centralized search requires all the sensor nodes to transmit their local measurements to a fusion center, which may be impractical because of transmit power consumption and scalability. Instead, we consider distributed estimation method where each node communicates only with its neighbors in the network. More specifically, every node iteratively averages its intermediate estimate with neighbor's estimates for a certain period of time. We need to define a set of grid points at which each sensor combines the density estimates with the neighbor's. The grid points are updated until all the points approach to the mode.

5.4 Distributed Density Estimation and Mode Consensus

We average the locally estimated density y_{nz} in a distributed way, updating the grid points at which y_{nz} is combined with neighborhood. Since the global mode θ in (5.4) can be found by examining $f(z)$ with z, the distributed averaging process is repeated with a grid (row) vector at each node n

$$\mathbf{z}_n = \left[z_{n1}, \ldots, z_{nq}, \ldots, z_{nQ}\right], \tag{5.5}$$

updating $\mathbf{z}_n$ under a certain rule. We assume the size of $\mathbf{z}_n$ is fixed but the elements of the vector are updated to find the maximum $f(z_{nq})$ in (5.1). The range of $\omega_{nQ} - \omega_{n1}$ is repeatedly reduced, and $\boldsymbol{\omega}_n$ finally converges to the mode θ.

We define the local density vector $\mathbf{y}_n$ that is obtained by substituting the grid points $\{z_{nq}\}_{q=1}^{Q}$:

$$\mathbf{y}_n = [y_{n1}, \ldots, y_{nQ}] \tag{5.6}$$

where each element y_{qn} is defined by

$$y_{nq} \triangleq \frac{1}{M} \sum_{m=1}^{M} \frac{1}{Z} K_h \left(z_{nq} - x_{nm}\right). \tag{5.7}$$

The local density vector $\mathbf{y}_n$ is averaged up to $\widehat{\imath}$ iterations at grid vector $\mathbf{z}_n$ for all n. Then, the averaging procedure is repeated $\widehat{p}$ times, updating $\mathbf{z}_n$ for all n. Let $y_{nq}^{p}(0)$

denote the initial state (i.e. $i = 0$) of the local density in the p-th repeat at node n's grid point q. The distributed average consensus algorithm [4] of the p-th repeat is described as

$$y^p_{nq}(i+1) = y^p_{nq}(i) - \alpha \sum_{l \in \mathbb{N}_n} \left(y^p_{nq}(i) - y^p_{lq}(i) \right) \tag{5.8}$$

where $\mathbb{N}_n$ denotes the neighborhood of node n (excluding n) and α is the step-size that can be determined by $0 < \alpha < 2/\lambda_N(L)$ [4], $n = 1, 2, \ldots, N$, and $i = 0, 1, \ldots, \widehat{\imath}$. Let $\mathbf{Y} = \left[\mathbf{y}_1^T, \ldots, \mathbf{y}_N^T\right]^T$. The matrix form of (5.8) becomes

$$\mathbf{Y}^p(i+1) = W \cdot \mathbf{Y}^p(i) \tag{5.9}$$

where W is a doubly stochastic matrix satisfying

$$W = W^T, \; W\mathbf{1} = \mathbf{1}, \text{ and } W = I - \alpha L \tag{5.10}$$

where $L = D - A$ is Laplacian matrix and defined by degree and adjacent matrices D and A. With a proper selection of α [4], the spectral norm satisfies

$$\rho\left(W - \frac{1}{N}\mathbf{1}\mathbf{1}^T\right) < 1 \tag{5.11}$$

where $\rho(\cdot)$ is the spectral radius.

Since the grid points in $\mathbf{z}_n$ at every node should be consistent in order to average the local densities of elements in the vector $\mathbf{y}_n$ at the same grid point q, we need to specify update rules for the grid points. Essentially the grid points are updated by searching the location z_q that maximizes the estimated density $f(z_q)$. The updates are based on the *divide-and-conquer* technique where the range of grid vector $\mathbf{z}_n$ is reduced by a constant factor (e.g. half) at each repeat.

5.4.1 Minimum Point Update Rule (mPUR)

The mode can be found by making the minimum grid point z_{n1} in $\mathbf{z}_n$ converge to θ for all n. At the p-th repeat, z_{n1}^p can be updated as below:

$$z_{n1}^{p+1} = z_{n1}^{p} + \frac{V}{2^p} u \left(\omega_n^p - \frac{Q}{2} \right), \quad p \geq 1, \tag{5.12}$$

where V denotes a range of interest assuming the desired mode is within the range, $u\left(\cdot\right)$ is defined by

$$u(a) = \begin{cases} 0 & \text{if} \quad a < 0 \\ 1 & \text{if} \quad a \geq 0 \end{cases}, \tag{5.13}$$

and the index ω_n^p is obtained by

$$\omega_n^p = \underset{q}{\operatorname{argmax}}\, y_{nq}^p(\widehat{\imath} + 1) \tag{5.14}$$

where $y_{nq}^p(\widehat{\imath} + 1)$ is obtained by (5.8) after $\widehat{\imath}$ iterations. Then, every node updates the grid points at each p-th repeat under the following rule:

$$z_{nq}^p = z_{n1}^p + \frac{V}{2^{p-1}} \frac{q-1}{Q}, \; p \geq 1, \; q = 1, 2, \ldots, Q, \; \forall n, \tag{5.15}$$

where Q denotes the size of grid points between z_{n1}^p and $z_{n1}^p + V/2^{p-1}$. The grid points at different nodes can be unmatched, if $\omega_n^p \neq \omega_l^p$ for $n \neq l$ in (5.14). We will analyze the required number of iterations $\widehat{\imath}$ for a certain level of error probability. The distributed mode consensus updating z_{n1}^p is summarized in Algorithm 4.

5.4.2 Maximum Point Update Rule (MPUR)

An alternative method is to update the maximum grid point z_{nQ} in $\mathbf{z}_n$ for converging to θ for all n. This can be applied when all the nodes know the maximum

Algorithm 4 Distributed Mode Search by mPUR

```
Initialization: set $V$, $Q$, $\alpha$, $z_{n1}^{p} = z_{n1}^{p} + \frac{V}{2^{p-1}}\frac{q-1}{Q}$ for $p = 1$, $q = 1, \ldots, Q$, and all $n$.
for $p \le \widehat{p}$ do
    for $i \le \widehat{\imath}$ do
        run (5.8) with $y_{nq}^{p}$, $\forall n$. For $i = 0$, defined in (5.7).
        $i \leftarrow i + 1$.
    end for
    update $z_{n1}^{p}$ by (5.12).
    update $\mathbf{z}_{n}^{p}$ by (5.15).
    $p \leftarrow p + 1$.
end for
```

point and the range of interest V, assuming that the desired mode is located between $z_{nQ} - V$ and z_{nQ}. At the p-th repeat, z_{nQ}^{p} is updated as below:

$$z_{nQ}^{p+1} = z_{nQ}^{p} - \frac{V}{2^{p}}\, u\left(\frac{Q}{2} - \omega_{n}^{p}\right), \quad p \ge 1, \tag{5.16}$$

where $u\left(Q/2 - \omega_{n}^{p}\right)$ is defined by (5.13) and ω_{n}^{p} in (5.14). Then, every node updates the grid point vector at each p-th repeat under the following rule:

$$z_{nq}^{p} = z_{nQ}^{p} - \frac{V}{2^{p-1}}\left(1 - \frac{q-1}{Q}\right), \; p \ge 1, \, q = 1, \ldots, Q, \, \forall n, \tag{5.17}$$

where Q is the size of grid points between $z_{nQ}^{p} - V/2^{p-1}$ and z_{nQ}^{p}. The distributed mode search method updating z_{nQ}^{p} is summarized in Algorithm 5.

5.4.3 Central Point Update Rule (CPUR)

We may not know whether the maximum (or minimum) grid point is larger (or smaller) than the potential mode location. The central grid point z_{nc} in $\mathbf{z}_{n}$ is updated

Algorithm 5 Distributed Mode Search by MPUR

```
Initialization: set $V$, $Q$, $\alpha$, $z^p_{nq} = z^p_{nQ} - \frac{V}{2^{p-1}}\left(1 - \frac{q-1}{Q}\right)$ for $p = 1$, $q = 1, \ldots, Q$, and all $n$.
for $p \leq \widehat{p}$ do
    for $i \leq \widehat{\imath}$ do
        run (5.8) with $y^p_{nq}$, $\forall n$. For $i = 0$, defined in (5.7).
        $i \leftarrow i + 1$.
    end for
    update $z^p_{nQ}$ by (5.16).
    update $\mathbf{z}^p_n$ by (5.17).
    $p \leftarrow p + 1$.
end for
```

in order to make it converge to θ for all n without such an initial guess. At p-th repeat, z^p_{nc} is updated as below:

$$z^{p+1}_{nc} = z^p_{n,\omega^p_n}, \quad p \geq 1, \tag{5.18}$$

where ω^p_n is defined by (5.14). Then, every node updates the grid point vector at each p-th repeat under the following rule:

$$z^p_{nq} = z^p_{nc} + \frac{V}{2^{p-1}}\left(\frac{q}{Q} - 0.5\right), \ p \geq 1, \, q = 1, \ldots, Q, \, \forall n, \tag{5.19}$$

where Q is the size of grid vector between $z_{nc} - V/2^{p-2}$ and $z_{nc} + V/2^{p-2}$. The distributed mode search method updating z^p_{nc} is summarized in Algorithm 6.

5.5 Numerical Experiments

There are $N = 70$ sensors in a fully distributed network. Each sensor node has $M = 100$ samples. 70% of the nodes, $m = 1, 2, \ldots, 49$, observe the data distributed

Algorithm 6 Distributed Mode Search by CPUR

```
Initialization: set $V$, $Q$, $\alpha$, $z_{nq}^p = z_{nc}^p + \frac{V}{2^{p-1}}\left(\frac{q}{Q} - 0.5\right)$ for $p = 1$, $q = 1, \ldots, Q$, and all $n$.
for $p \le \widehat{p}$ do
    for $i \le \widehat{\imath}$ do
        run (5.8) with $y_{nq}^p$, $\forall n$. For $i = 0$, defined in (5.7).
        $i \leftarrow i + 1$.
    end for
    update $z_{nc}^p$ by (5.18).
    update $\mathbf{z}_n^p$ by (5.19).
    $p \leftarrow p + 1$.
end for
```

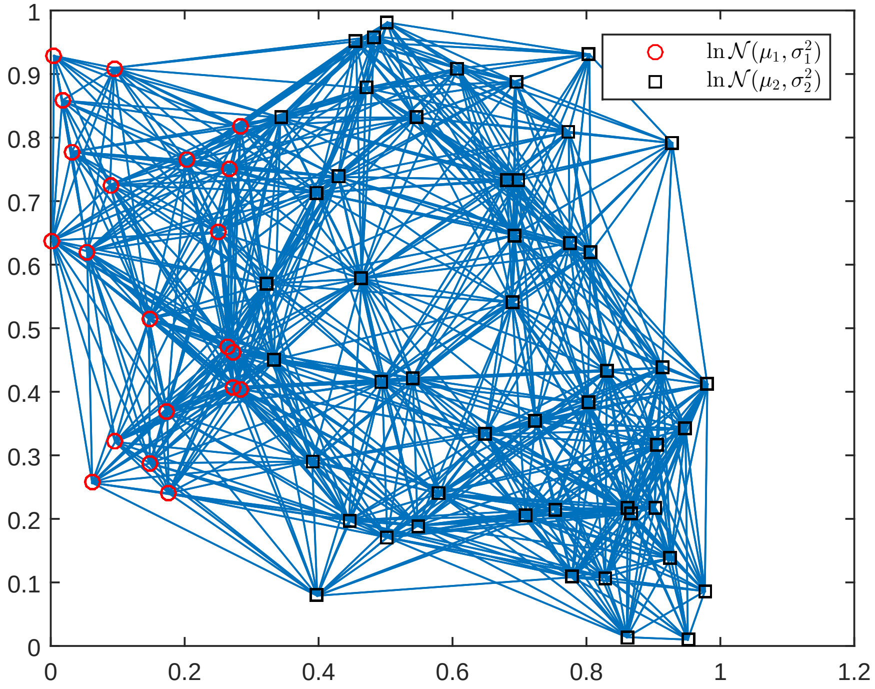


Figure 5.1: Network Graph Model. Two Different Distributions Were Generated.

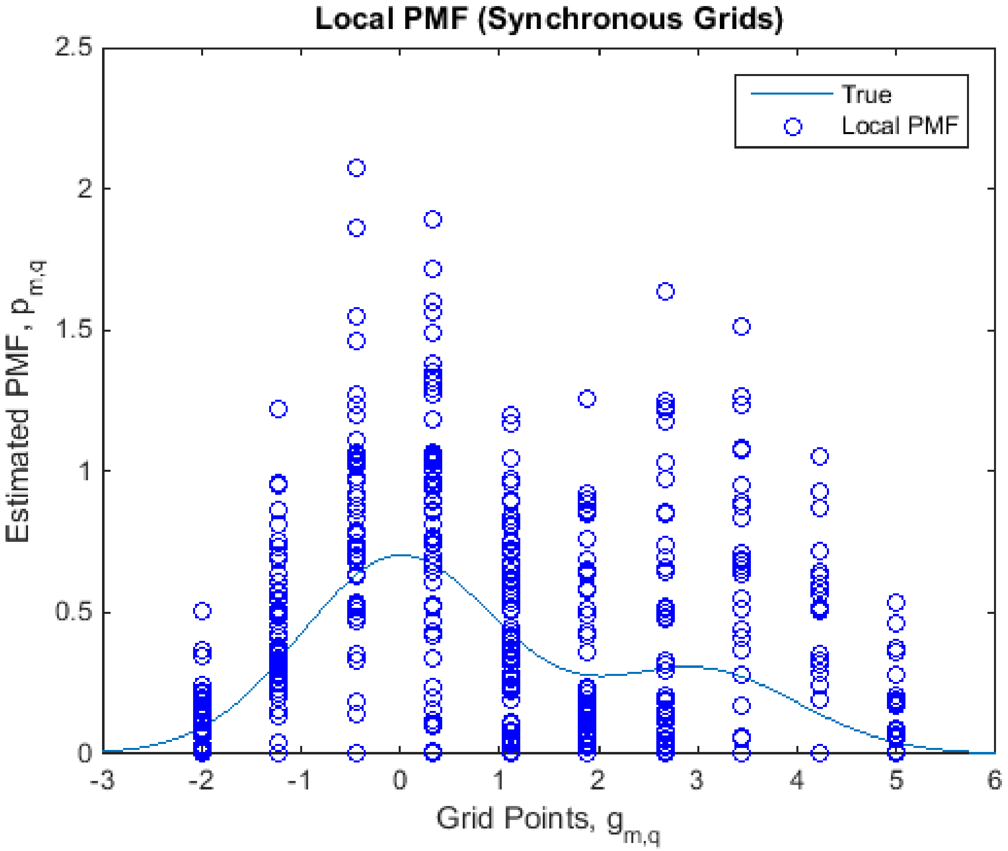


Figure 5.2: Local Density Estimates (PMFs) with Initial Grid.

as $\mathcal{N}(0,1)$, and 30% of the nodes, $m = 49, \ldots, 70$, observe $\mathcal{N}(3,1)$, as illustrated in Fig. 5.1. Each group of nodes has relatively dense connectivity, whereas inter-group is sparsely connected. The grid size at every node is $Q = 10$. The range of the grid points are $\min_{z_{nq}} = -2$ and $\max_{z_{nq}} = 5$ for all n. The kernel size $h = 0.1$, but it could be tested with another sizes.

Figure 5.2 shows local density estimates at each node with the initial grid points. Each node observes limited number of data samples M and estimates the local density at $Q = 10$ grid points. Fig. 5.3 shows the estimated density at the grid points after running the distributed average consensus algorithm (5.8). With the finite $\widehat{\imath}$, the densities are approximation of true PDF. However, Fig. 5.4 shows that all the density estimates converge to the true PDF, as $\widehat{\imath} \rightarrow \infty$. Although the estimated densities are

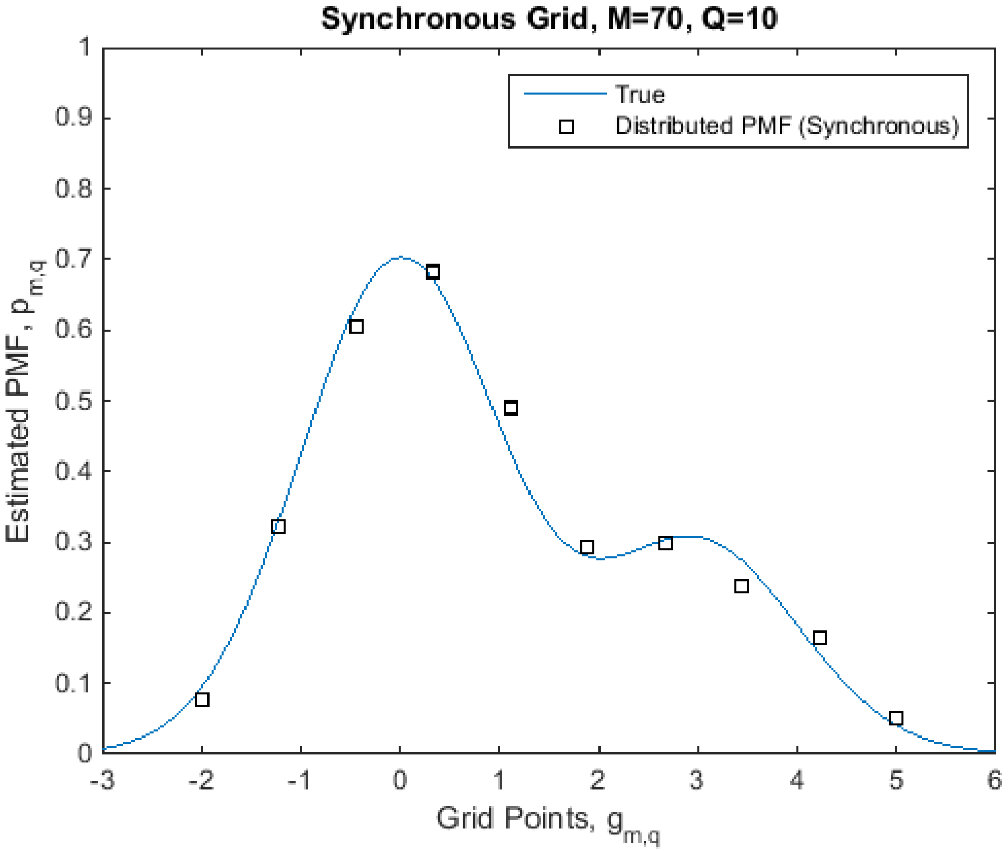


Figure 5.3: Density Estimates with Initial Grid after Running (5.8).

not true PDF but approximation of it, Fig. 5.5 shows that updating grid points can provide the mode consensus.

5.6 Concluding Remarks and Future Works

In this chapter, we have presented distributed density estimation algorithms with updating grid points, in order to achieve mode consensus. The grid points estimate densities of data by distributed average consensus algorithm with finite number of iterations $\widehat{\imath}$. Since the mode represents the most density regions, updating the grid points and running the distributed average consensus algorithm between the updates of grid points can achieve the mode consensus. We have described three update rules for the grid points, and then shown numerical experiments to evaluate the

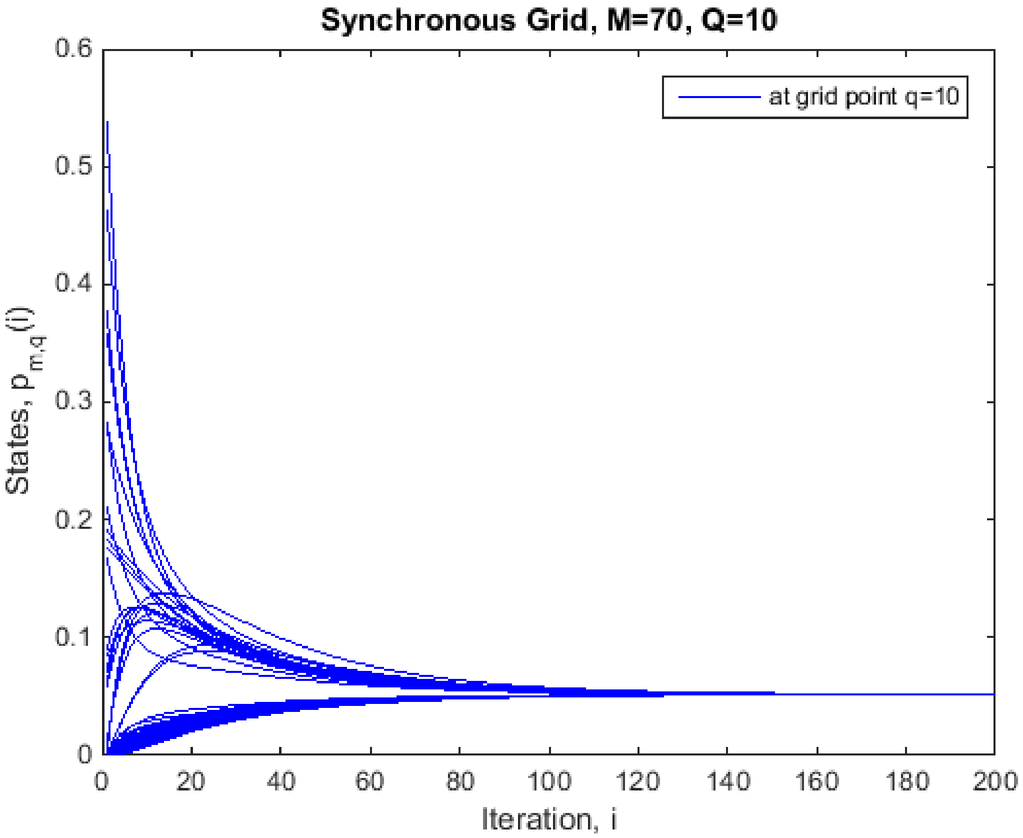


Figure 5.4: Consensus Can Be Achieved, As $\widehat{\imath} \to \infty$. The Grid Point $q = 10$.

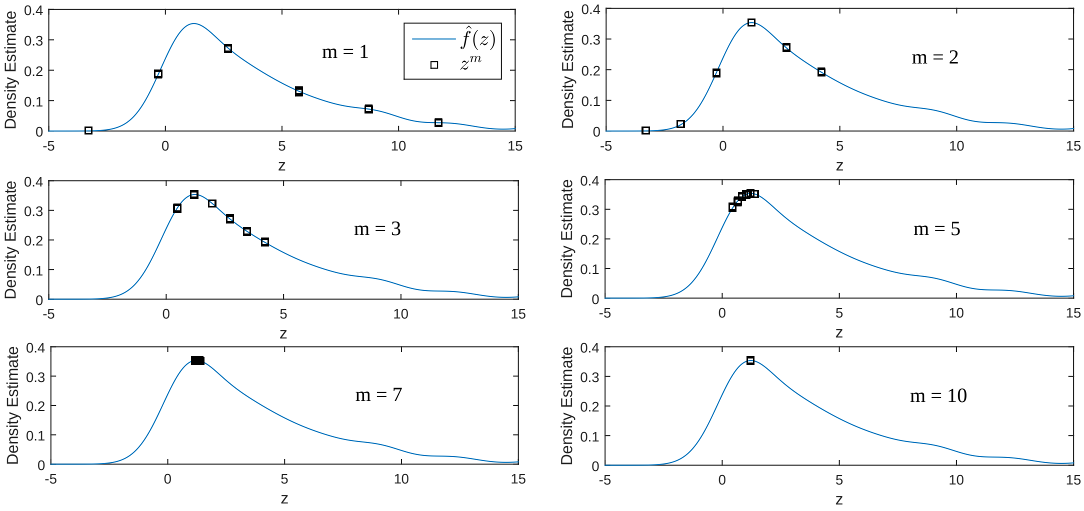


Figure 5.5: Distributed Mode Consensus, Compared to the Centralized KDE of (5.1). Squares Denote $\mathbf{z}^m$ with a Vector Size of $Q = 6$. As m Increases, $\mathbf{z}^m$ Converges to the Mode $\hat{\theta}$. $\widehat{\imath} = 40$.

mode consensus by the proposed methods. The mode estimation algorithm utilizes parzen's kernel density estimator where there is a free parameter - kernel size h. Thus, the estimated mode is not necessarily the true mode. As a future work we may design the kernel size which can be varied with the total number of measurement data. Moreover, we assumed the finite number of iterations of distributed average consensus algorithm between the updates of grid points. Another future work is to obtain the trade-off between the iterations $\widehat{\imath}$ and the measurement data size M.

5.6.1 Kernel Size h Design

Recall that the local density estimates y_{nz} for all n at arbitrary location $z \in \mathbb{R}$ depend on the kernel size h as shown in (5.1). Generally h is a free-parameter that is empirically determined by the size of samples (i.e., N and M) [106]. with a certain criterion such as mean-squared error (MSE), we may obtain a guideline how to select h as a function of $\widehat{\imath}$ as well as NM.

5.6.2 Trade-off between $\widehat{\imath}$ and M

As [4] described, the step-size α in the weight matrix $\mathbf{W} = \mathbf{I} - \alpha\mathbf{L}$ in (5.9) affects the convergence rate of averaging. Thus, we can obtain an upper bound of the averaging time, which is a function of the spectral radius defined in (5.11), and then find the achievable averaging time by selection of α. It may be shown that the upper bound on the averaging time of $\widehat{\imath}$ depends on the number of local measurements M.

Chapter 6

CONCLUSIONS

We have studied consensus-based schemes in distributed networks where there is no fusion center but every node achieves a sensor fusion by in-network processing. We proposed nonlinear diffusion adaptation scheme with bounded transmission for distributed sensor networks. The nonlinear scheme was motivated by transmit power savings and nonlinearity of power amplifiers. We have shown the proposed nonlinear diffusion adaptation algorithm can estimate a global parameter that is associated with real-time measurement data. Every sensor node can estimate the parameter with convergence in the mean and stability. The mean-square-deviation (MSD) and mean-square-error (MSE) were evaluated for performance of the proposed scheme. We have also shown the nonlinear algorithm can be close to the linear case with enhanced power savings in distributed sensor networks.

Next, we have proposed an algorithm for distributed quantile estimation where quantiles of measurement sensor data are obtained in a distributed way. Intuitively, given a set of measurement data, a quantile can be obtained by using empirical cumulative distribution function (ECDF) after collecting all the measurement data available from every node. In a fully distributed network, however, each node has only limited information of the global measurement data. The proposed algorithm estimates a quantile by in-network processing. States of a quantile estimation are recursively updated by the combination of local update and averaging steps in the presence of communication noise. We analyzed convergence behaviors of the algorithm based on mixed-time scale stochastic approximation where the averaging time scale dominates the local update time scale. The estimated state sequence is asymptotically unbiased and converges toward the true quantile in mean-square sense. By using the proposed algorithm, various statistical quantities can be estimated such as maximum, minimum, median, n-th smallest element selection out of N elements, outliers identification, and trimmed mean in fully distributed sensor networks.

Distributed mean-shift mode finding scheme was described. As a measure of central tendency of sensor data, we can use a mode that represents the most probable value of the data. Parzen's kernel density estimator can be used for centralized case after all the measurement data are collected from every node. However, in a fully distributed network, it is difficult to share the local measurement data over the network and hard to synchronize the states of mode estimates. We have proposed a mode estimation algorithm based on algorithms of Gaussian mean-shift and distributed extension of expectation-maximization. While each node maintains the expectation step based on local information, distributed cooperation is performed over the network, which results in updating mean-shift vector. Simulation results demonstrate that the estimated states converge toward a consensus of the mode estimation in the presence and absence of communication noise. We compared performance between the distributed and centralized scenarios. The proposed method was also compared with the centralized GMM to estimate a mode of skewed data distribution. Different conditions such as step-sizes and measurement size M were also evaluated in the presence and absence of communication noise.

Finally, this dissertation has described a distributed density estimation method with updating grid vectors to find a global mode of data. At each update of the grid, distributed average consensus algorithm is run with a finite number of iterations. The estimated density at each grid approaches the global mode, as the grid vector is updated. We have presented three update rules for the grid vector. As the kernel size is a free parameter of kernel density estimator, as a future work, we can design the kernel size parameter to estimate the true mode of distribution. The proposed methods are based on finite number of iterations between updates of grid vector. This leads us to investigate relation between the number of iterations for the average consensus algorithm and the measurement data size. As either the iteration number or the local measurement data size increases, we can have more accurate estimate results. It was shown that the parameters affect communication cost and memory usage at local sensor nodes.

REFERENCES

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “Wireless sensor networks: A survey,” *Computer Networks*, vol. 38, no. 4, pp. 393–422, Mar. 2002.

[2] J. Yick, B. Mukherjee, and D. Ghosal, “Wireless sensor network survey,” *Computer Networks*, vol. 52, no. 12, pp. 2292–2330, Aug. 2008.

[3] A. Bharathidasan, V. An, and S. Ponduru, “Sensor networks: An overview,” Department of Computer Science, University of California, Davis, Tech. Rep., 2002.

[4] L. Xiao and S. Boyd, “Fast linear iterations for distributed averaging,” *Systems & Control Letters*, vol. 53, no. 1, pp. 65–78, Sep. 2004.

[5] R. Olfati-Saber, J. A. Fax, and R. M. Murray, “Consensus and Cooperation in Networked Multi-Agent Systems,” *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, Jan. 2007.

[6] C. G. Lopes and A. H. Sayed, “Diffusion Least-Mean Squares Over Adaptive Networks: Formulation and Performance Analysis,” *IEEE Transactions on Signal Processing*, vol. 56, no. 7, pp. 3122–3136, Jul. 2008.

[7] B. Widrow and S. D. Stearns, *Adaptive signal processing.* Englewood Cliffs, N.J: Prentice-Hall, 1985.

[8] S. O. Haykin, *Adaptive Filter Theory (4th Edition)*, 4th ed. Prentice Hall, Sep. 2001.

[9] A. Sayed, *Fundamentals of Adaptive Filtering.* Wiley, 2003.

[10] A. Spanias, *Digital Signal Processing; An Interactive Approach - 2nd Edition.* Morrisville, NC: Lulu Press, 2014.

[11] J. Foutz, A. Spanias, and M. Banavar, *Narrowband Direction of Arrival Estimation for Antenna Arrays*, ser. Synthesis lectures on antennas. Morgan & Claypool Publishers, 2008.

[12] R. Olfati-Saber and R. M. Murray, “Consensus problems in networks of agents with switching topology and time-delays,” *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1520–1533, Sep. 2004.

[13] F. R. K. Chung, *Spectral Graph Theory*, ser. CBMS Regional Conference Series. American Mathematical Society, 1997, no. 92.

[14] S. Kar and J. M. F. Moura, “Distributed Average Consensus in Sensor Networks with Random Link Failures,” vol. 2, Apr. 2007, pp. 1013–1016.

[15] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *Proceedings of Fourth International Symposium on Information Processing in Sensor Networks*, Apr. 2005, pp. 63–70.

[16] L. Xiao, S. Boyd, and S.-J. Kim, "Distributed average consensus with least-mean-square deviation," *Journal of Parallel and Distributed Computing*, vol. 67, no. 1, pp. 33–46, Jan. 2007.

[17] M. Huang and J. Manton, "Stochastic lyapunov analysis for consensus algorithms with noisy measurements," in *Proceedings of American Control Conference*, Jul. 2007, pp. 1419–1424.

[18] ——, "Stochastic approximation for consensus seeking: Mean square and almost sure convergence," in *Proceedings of 46th IEEE Conference on Decision and Control*, Dec. 2007, pp. 306–311.

[19] I. Schizas, A. Ribeiro, and G. Giannakis, "Consensus-based distributed parameter estimation in ad hoc wireless sensor networks with noisy links," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 2, Apr. 2007, pp. 849–852.

[20] S. Kar and J. Moura, "Sensor networks with random links: Topology design for distributed consensus," *IEEE Transactions on Signal Processing*, vol. 56, no. 7, pp. 3315–3326, Jul. 2008.

[21] Y. Hatano, A. Das, and M. Mesbahi, "Agreement in presence of noise: pseudo-gradients on random geometric networks," in *Proceedings of 44th IEEE Conference on Decision and Control*, Dec. 2005, pp. 6382–6387.

[22] M. Rabbat, R. Nowak, and J. Bucklew, "Generalized consensus computation in networked systems with erasure links," in *Proceedings of IEEE 6th Workshop on Signal Processing Advances in Wireless Communications*, Jun. 2005, pp. 1088–1092.

[23] C. Wu, "Synchronization and convergence of linear dynamics in random directed networks," *IEEE Transactions on Automatic Control*, vol. 51, no. 7, pp. 1207–1210, Jul. 2006.

[24] A. Tahbaz-Salehi and A. Jadbabaie, "On consensus over random networks," *proceedings of 44th Annual Allerton Conference*, 2006.

[25] M. Porfiri and D. Stilwell, "Stochastic consensus over weighted directed networks," in *Proceedings of American Control Conference*, Jul. 2007, pp. 1425–1430.

[26] S. Kar and J. M. F. Moura, "Distributed Consensus Algorithms in Sensor Networks With Imperfect Communication: Link Failures and Channel Noise," *IEEE Transactions on Signal Processing*, vol. 57, no. 1, pp. 355–369, Jan. 2009.

[27] S. Dasarathan, C. Tepedelenliolu, M. Banavar, and A. Spanias, "Non-linear distributed average consensus using bounded transmissions," *IEEE Transactions on Signal Processing*, vol. 61, no. 23, pp. 6000–6009, Dec. 2013.

[28] Y. Hatano and M. Mesbahi, "Agreement over random networks," *IEEE Transactions on Automatic Control*, vol. 50, no. 11, pp. 1867–1872, Nov. 2005.

[29] R. Santucci, M. Banavar, C. Tepedelenlioglu, and A. Spanias, "Energy-efficient distributed estimation by utilizing a nonlinear amplifier," *IEEE Transactions on Circuits and Systems I*, vol. 61, no. 1, pp. 302–311, Jan. 2014.

[30] S. Cripps, *Advanced Techniques in RF Power Amplifier Design*, ser. Artech House microwave library. Artech House, 2002.

[31] J. N. Tsitsiklis, "Problems in decentralized decision making and computation," Ph.D. dissertation, Department of EECS, MIT, Nov. 1984.

[32] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods.* Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1989.

[33] J. Tsitsiklis, D. Bertsekas, and M. Athans, "Distributed asynchronous deterministic and stochastic gradient optimization algorithms," *IEEE Transactions on Automatic Control*, vol. 31, no. 9, pp. 803–812, Sep. 1986.

[34] H. Kushner and G. Yin, *Stochastic Approximation and Recursive Algorithms and Applications*, ser. Stochastic Modelling and Applied Probability. Springer New York, 2003.

[35] S. S. Stankovic, M. S. Stankovic, and D. M. Stipanovic, "Decentralized parameter estimation by consensus based stochastic approximation," *IEEE Transactions on Automatic Control*, vol. 56, no. 3, pp. 531–543, Mar. 2011.

[36] S. Kar, J. M. F. Moura, and K. Ramanan, "Distributed parameter estimation in sensor networks: Nonlinear observation models and imperfect communication," *IEEE Transactions on Information Theory*, vol. 58, no. 6, pp. 3575–3605, Jun. 2012.

[37] P. Bianchi, G. Fort, and W. Hachem, "Performance of a distributed stochastic approximation algorithm," *IEEE Transactions on Information Theory*, vol. 59, no. 11, pp. 7405–7418, Nov. 2013.

[38] S.-Y. Tu and A. H. Sayed, "Foraging behavior of fish schools via diffusion adaptation," in *Proceedings of 2nd International Workshop on Cognitive Information Processing*, Jun. 2010, pp. 63–68.

[39] F. S. Cattivelli and A. H. Sayed, "Modeling bird flight formations using diffusion adaptation," *IEEE Transactions on Signal Processing*, vol. 59, no. 5, pp. 2038–2051, May 2011.

[40] ——, "Distributed detection over adaptive networks using diffusion adaptation," *IEEE Transactions on Signal Processing*, vol. 59, no. 5, pp. 1917–1932, May 2011.

[41] S. Al-Sayed, A. M. Zoubir, and A. H. Sayed, "Robust distributed detection over adaptive diffusion networks," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, May 2014, pp. 7233–7237.

[42] A. H. Sayed, "Diffusion Adaptation over Networks," in *Academic Press Library in Signal Processing*, vol. 3. Academic Press, Elsevier, 2014, pp. 323–454.

[43] ——, "Diffusion Adaptation over Networks," May 2013.

[44] R. Abdolee and B. Champagne, "Diffusion LMS algorithms for sensor networks over non-ideal inter-sensor wireless channels," in *Proceedings of International Conference on Distributed Computing in Sensor Systems and Workshops*, Jun. 2011.

[45] S.-Y. Tu and A. Sayed, "Adaptive networks with noisy links," in *Proceedings of IEEE Global Telecommunications Conference*, Dec. 2011.

[46] X. Zhao, S.-Y. Tu, and A. H. Sayed, "Diffusion Adaptation Over Networks Under Imperfect Information Exchange and Non-Stationary Data," *IEEE Transactions on Signal Processing*, vol. 60, no. 7, pp. 3460–3475, Jul. 2012.

[47] J. Lee, C. Tepedelenlioglu, M. K. Banavar, and A. Spanias, "Nonlinear diffusion adaptation with bounded transmission over distributed networks," in *Proceedings of IEEE International Conference on Communications*, Jun. 2015, pp. 6707–6711.

[48] S.-Y. Tu and A. H. Sayed, "Diffusion Strategies Outperform Consensus Strategies for Distributed Estimation over Adaptive Networks," Aug. 2012.

[49] J. Chen and A. H. Sayed, "Diffusion adaptation strategies for distributed optimization and learning over networks," *IEEE Transactions on Signal Processing*, vol. 60, no. 8, pp. 4289–4305, Aug. 2012.

[50] A. Sayed, *Adaptive Filters.* Wiley, 2011.

[51] F. S. Cattivelli and A. H. Sayed, "Diffusion LMS Strategies for Distributed Estimation," *IEEE Transactions on Signal Processing*, vol. 58, no. 3, pp. 1035–1048, Mar. 2010.

[52] A. Khalili, M. A. Tinati, A. Rastegarnia, and J. A. Chambers, "Steady-State Analysis of Diffusion LMS Adaptive Networks With Noisy Links," *IEEE Transactions on Signal Processing*, vol. 60, no. 2, pp. 974–979, Feb. 2012.

[53] ——, "Transient analysis of diffusion least-mean squares adaptive networks with noisy channels," *International Journal of Adaptive Control and Signal Processing*, vol. 26, no. 2, pp. 171–180, Feb. 2012.

[54] A. H. Sayed, S.-Y. Tu, J. Chen, X. Zhao, and Z. J. Towfic, "Diffusion strategies for adaptation and learning over networks: an examination of distributed strategies and network behavior," *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 155–171, May 2013.

[55] N. Meinshausen, "Quantile regression forests," *Journal of Machine Learning Research*, vol. 7, pp. 983–999, 2006.

[56] R. Koenker, *Quantile Regression.* Cambridge University Press, 2005.

[57] S. Kumar and S. B. Hedges, "A molecular timescale for vertebrate evolution," *Nature*, vol. 392, no. 6679, pp. 917–920, Apr. 1998.

[58] D. S. Heckman, D. M. Geiser, B. R. Eidell, R. L. Stauffer, N. L. Kardos, and S. B. Hedges, "Molecular evidence for the early colonization of land by fungi and plants," *Science*, vol. 293, no. 5532, pp. 1129–1133, 2001.

[59] S. B. Hedges and P. Shah, "Comparison of mode estimation methods and application in molecular clock analysis," *BMC Bioinformatics*, vol. 4, no. 1, pp. 31+, Jul. 2003.

[60] D. R. Bickel, "Robust estimators of the mode and skewness of continuous data," *Computational Statistics & Data Analysis*, vol. 39, no. 2, pp. 153 – 163, 2002.

[61] R. R. Sokal and F. J. Rohlf, *Biometry: the Principles and Practice of Statistics in Biological Research, 4th ed.* New York, W.H. Freeman and Co., 2012.

[62] T. Dalenius, "The mode–a neglected statistical parameter," *Journal of the Royal Statistical Society. Series A (General)*, vol. 128, no. 1, pp. 110–117, 1965.

[63] U. Grenander, "Some direct estimates of the mode," *The Annals of Mathematical Statistics*, vol. 36, no. 1, pp. 131–138, 1965.

[64] D. R. Bickel, "Robust and efficient estimation of the mode of continuous data: the mode as a viable measure of central tendency," *Journal of Statistical Computation and Simulation*, vol. 73, no. 12, pp. 899–912, Dec. 2003.

[65] D. R. Bickel and R. Fruhwirth, "On a fast, robust estimator of the mode: Comparisons to other robust estimators with applications," *Computational Statistics & Data Analysis*, vol. 50, no. 12, pp. 3500 – 3530, 2006.

[66] J. A. Fax and R. M. Murray, "Information flow and cooperative control of vehicle formations," *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1465–1476, Sep. 2004.

[67] P. Ferguson and J. P. How, "Decentralized estimation algorithms for formation flying spacecraft," in *AIAA Guidance, Navigation, and Control Conference (GNC)*, Aug. 2003.

[68] A. Jadbabaie, L. Lin, and A. S. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *IEEE Transactions on Automatic Control*, vol. 48, no. 6, pp. 988–1001, Jun. 2003.

[69] Z. Lin, B. Francis, and M. Maggiore, "Necessary and sufficient graphical conditions for formation control of unicycles," *IEEE Transactions on Automatic Control*, vol. 50, no. 1, pp. 121–127, Jan. 2005.

[70] L. Moreau, "Stability of multiagent systems with time-dependent communication links," *IEEE Transactions on Automatic Control*, vol. 50, no. 2, pp. 169–182, Feb. 2005.

[71] R. Olfati-Saber, "Flocking for multi-agent dynamic systems: algorithms and theory," *IEEE Transactions on Automatic Control*, vol. 51, no. 3, pp. 401–420, Mar. 2006.

[72] W. Ren and R. W. Beard, "Consensus seeking in multiagent systems under dynamically changing interaction topologies," *IEEE Transactions on Automatic Control*, vol. 50, no. 5, pp. 655–661, May 2005.

[73] W. Ren, R. W. Beard, and D. B. Kingston, "Multi-agent kalman consensus with relative uncertainty," in *Proceedings of 2005 American Control Conference*, Jun. 2005, pp. 1865–1870.

[74] D. P. Spanos, R. Olfati-Saber, and R. M. Murray, "Distributed sensor fusion using dynamic consensus," in *IFAC World Congress*. Prague Czech Republic, 2005.

[75] A. Speranzon, C. Fischione, and K. H. Johansson, "Distributed and collaborative estimation over wireless sensor networks," in *Proceedings of 45th IEEE Conference on Decision and Control*, Dec. 2006, pp. 1025–1030.

[76] J. A. Deri and J. M. F. Moura, "Graph sampling: Estimation of degree distributions," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, May 2013, pp. 6501–6505.

[77] H. Terelius, D. Varagnolo, C. Baquero, and K. H. Johansson, "Fast distributed estimation of empirical mass functions over anonymous networks," in *52nd IEEE Conference on Decision and Control*, Dec 2013, pp. 6771–6777.

[78] J. Sacha, J. Napper, C. Stratan, and G. Pierre, "Adam2: Reliable distribution estimation in decentralised environments," in *2010 IEEE 30th International Conference on Distributed Computing Systems*, June 2010, pp. 697–707.

[79] S. Zhang, J. Lee, C. Tepedelenlioglu, and A. Spanias, "Distributed estimation of the degree distribution in wireless sensor networks," in *2016 IEEE Global Communications Conference (GLOBECOM)*, Dec 2016, pp. 1–6.

[80] S. Buddha, H. Braun, V. Krishnan, C. Tepedelenlioglu, A. Spanias, T. Yeider, and T. Takehara, "Signal processing for photovoltaic applications," in *Proceedings of IEEE International Conference on Emerging Signal Processing Applications*, Jan. 2012, pp. 115–118.

[81] R. Viswanathan and P. Varshney, "Distributed detection with multiple sensors i. fundamentals," *Proceedings of the IEEE*, vol. 85, no. 1, pp. 54–63, Jan. 1997.

[82] R. Blum, S. Kassam, and H. Poor, "Distributed detection with multiple sensors i. advanced topics," *Proceedings of the IEEE*, vol. 85, no. 1, pp. 64–79, Jan. 1997.

[83] R. Tenney and N. R. Sandell, "Detection with distributed sensors," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. AES-17, no. 4, pp. 501–510, Jul. 1981.

[84] S. Peshin, D. Ramirez, J. Lee, H. Braun, C. Tepedelenlioglu, A. Spanias, M. Banavar, and D. Srinivasan, "A photovoltaic (pv) array monitoring simulator," in *Modeling, Identification, and control, int. conf. on, Innsbruck, Austria*, 2015.

[85] S. Rao, D. Ramirez, H. Braun, J. Lee, C. Tepedelenlioglu, E. Kyriakides, D. Srinivasan, J. Frye, S. Koizumi, Y. Morimoto, and A. Spanias, "An 18 kw solar array research facility for fault detection experiments," in *Proceedings of IEEE the 18th Mediterranean Electrotechnical Conference (MELECON)*, 2016.

[86] S. Rao, S. Katoch, P. Turaga, A. Spanias, C. Tepedelenlioglu, R. Ayyanar, H. Braun, J. Lee, M. Banavar, and D. Srinivasan, "A cyber-physical system approach for photovoltaic array monitoring and control," in *Proceedings of IEEE The 8th International Conference on Information, Intelligence, Systems and Applications (IISA)*, 2017.

[87] R. Santucci, M. Banavar, C. Tepedelenlioğlu, and A. Spanias, "Energy-efficient distributed estimation by utilizing a nonlinear amplifier," *IEEE Transactions on Circuits and Systems I*, vol. 61, no. 1, pp. 302–311, Jan. 2014.

[88] D. Scherber and H. Papadopoulos, "Locally constructed algorithms for distributed computations in ad-hoc networks," in *Proceedings of Third International Symposium on Information Processing in Sensor Networks*, Apr. 2004, pp. 11–19.

[89] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of State Calculations by Fast Computing Machines," *The Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, Dec. 1953.

[90] W. K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, vol. 57, no. 1, pp. 97–109, Apr. 1970.

[91] S. Zhang, C. Tepedelenlioglu, M. K. Banavar, and A. Spanias, "Max consensus in sensor networks: Non-linear bounded transmission and additive noise," *IEEE Sensors Journal*, vol. 16, no. 24, pp. 9089–9098, Dec. 2016.

[92] D. Kempe, A. Dobra, and J. Gehrke, "Gossip-based computation of aggregate information," in *Proceedings of 44th Annual IEEE Symposium on Foundations of Computer Science*, Oct. 2003, pp. 482–491.

[93] G. Yin, *Recent progress in parallel stochastic approximations.* Berlin, Heidelberg: Springer Berlin Heidelberg, 1991, pp. 159–184.

[94] G. Mateos, I. D. Schizas, and G. B. Giannakis, “Performance analysis of the consensus-based distributed lms algorithm,” *EURASIP Journal on Advanced Signal Processing*, Jan. 2009.

[95] H. Robbins and S. Monro, “A stochastic approximation method,” *The Annals of Mathematical Statistics*, vol. 22, no. 3, pp. 400–407, 1951.

[96] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, “Randomized gossip algorithms,” *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2508–2530, Jun. 2006.

[97] F. Kuhn, T. Locher, and R. Wattenhofer, “Tight bounds for distributed selection,” in *Proceedings of the Nineteenth Annual ACM Symposium on Parallel Algorithms and Architectures*, 2007, pp. 145–153.

[98] A. Negro, N. Santoro, and J. Urrutia, “Efficient distributed selection with bounded messages,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 8, no. 4, pp. 397–401, Apr. 1997.

[99] N. Santoro, J. B. Sidney, and S. J. Sidney, “A distributed selection algorithm and its expected communication complexity,” *Theoretical Computer Science*, vol. 100, no. 1, pp. 185 – 204, 1992.

[100] J. M. Marberg and E. Gafni, “An optimal shout-echo algorithm for selection in distributed sets,” in *Proceedings of 23rd Allerton Conference on Communication, Control and Computing*, 1985.

[101] R. J. Hyndman and Y. Fan, “Sample quantiles in statistical packages,” *The American Statistician*, vol. 50, no. 4, pp. 361–365, 1996.

[102] M. B. Nevel’son and R. Z. Has’minskii, *Stochastic Approximation and Recursive Estimation.* Providence, Rhode Island: American Mathematical Society, 1973.

[103] J. C. Spall, *Introduction to Stochastic Search and Optimization.* New York, NY, USA: John Wiley & Sons, Inc., 2003.

[104] H. Chen, *Stochastic Approximation and Its Applications*, ser. Nonconvex Optimization and Its Applications. Springer US, 2006.

[105] E. Parzen, “On estimation of a probability density function and mode,” *The Annals of Mathematical Statistics*, vol. 33, no. 3, pp. 1065–1076, 1962.

[106] B. Silverman, *Density Estimation for Statistics and Data Analysis*, ser. Chapman & Hall/CRC Monographs on Statistics & Applied Probability. London: Chapman & Hall, 1986.

[107] K. Fukunaga and L. Hostetler, “The estimation of the gradient of a density function, with applications in pattern recognition,” *IEEE Transactions on Information Theory*, vol. 21, no. 1, pp. 32–40, Jan. 1975.

[108] Y. Cheng, “Mean shift, mode seeking, and clustering,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 8, pp. 790–799, Aug. 1995.

[109] D. Comaniciu and P. Meer, “Mean shift: a robust approach toward feature space analysis,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, May 2002.

[110] J. Li, S. Ray, and B. G. Lindsay, “A nonparametric statistical approach to clustering via mode identification,” *Journal of Machine Learning Research*, vol. 8, pp. 1687–1723, Dec. 2007.

[111] M. A. Carreira-Perpinan, “Gaussian mean-shift is an EM algorithm,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 5, pp. 767–776, May 2007.

[112] R. D. Nowak, “Distributed EM algorithms for density estimation and clustering in sensor networks,” *IEEE Transactions on Signal Processing*, vol. 51, no. 8, pp. 2245–2253, Aug. 2003.

[113] D. Gu, “Distributed EM algorithm for gaussian mixtures in sensor networks,” *IEEE Transactions on Neural Networks*, vol. 19, no. 7, pp. 1154–1166, Jul. 2008.

[114] Z. J. Towfic, J. Chen, and A. H. Sayed, “Collaborative learning of mixture models using diffusion adaptation,” in *Proceedings IEEE International Workshop on Machine Learning for Signal Processing*, Sep. 2011.

[115] J. Yu and J. Thompson, “Diffusion-based EM gradient algorithm for density estimation in sensor networks,” in *Proceedings of IEEE 17th International Workshop on Signal Processing Advances in Wireless Communications*, Jul. 2016.

[116] S. Zhang, C. Tepedelenlioglu, M. K. Banavar, and A. Spanias, “Distributed node counting in wireless sensor networks in the presence of communication noise,” *IEEE Sensors Journal*, vol. 17, no. 4, pp. 1175–1186, Feb. 2017.

[117] S. Zhang, C. Tepedelenlioglu, J. Lee, H. Braun, and A. Spanias, “Cramer-rao bounds for distributed system size estimation using consensus algorithms,” in *Proceedings of IEEE Sensor Signal Processing for Defence (SSPD)*, 2016.

[118] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the EM algorithm,” *Journal of the Royal Statistical Society, Series B*, vol. 39, no. 1, pp. 1–38, 1977.

[119] A. H. Sayed, “Diffusion Adaptation over Networks,” in *Academic Press Library in Signal Processing*, vol. 3. Academic Press, Elsevier, 2014, pp. 323–454.

[120] J. P. Romano, “On weak convergence and optimality of kernel density estimates of the mode,” *The Annals of Statistics*, vol. 16, no. 2, pp. 629–647, 1988.

[121] B.-G. C. B. Abraham, Christophe, “On the asymptotic properties of a simple estimate of the mode,” *ESAIM: Probability and Statistics*, vol. 8, pp. 1–11, 2004.

APPENDIX A

PROOF FOR LEMMA 3.5.1

The proof refers to Lemma 25 in [36]. Consider large enough i_0 such that

$$r_1(i) \leq 1, \ \forall i \geq i_0. \tag{A.1}$$

Since $1 - z \leq e^{-z}$ for $0 \leq z \leq 1$, and from (3.18), we have

$$\begin{aligned}
\prod_{l=k+1}^{i-1} \big(1 - r_1(l)\big) &\leq e^{-\sum_{l=k+1}^{i-1} r_1(l)} \\
&= e^{-\sum_{l=k+1}^{i-1} \frac{a_1}{(i+1)^{\delta_1}}} \\
&\leq e^{-\int_{k+2}^{i+1} \frac{a_1}{t^{\delta_1}} dt} \\
&= e^{-\frac{a_1}{1-\delta_1}\left[(i+1)^{1-\delta_1} - (k+2)^{1-\delta_1}\right]}
\end{aligned} \tag{A.2}$$

where the inequality of third line is because of the properties of Riemann integral. We thus have

$$\begin{aligned}
\sum_{k=i_0}^{i-1} \left[\left(\prod_{l=k+1}^{i-1} \big(1 - r_1(l)\big) \right) r_2(k) \right] &\leq \sum_{k=i_0}^{i-1} \left[e^{-\frac{a_1}{1-\delta_1}\left[(i+1)^{1-\delta_1} - (k+2)^{1-\delta_1}\right]} \frac{a_2}{(k+1)^{\delta_2}} \right] \\
&= a_2 e^{-\frac{a_1}{1-\delta_1}(i+1)^{1-\delta_1}} \sum_{k=i_0}^{i-1} \left[e^{\frac{a_1}{1-\delta_1}(k+2)^{1-\delta_1}} \frac{1}{(k+1)^{\delta_2}} \right] \\
&= a_2 A_1^{-1} \sum_{k=i_0}^{i-1} \left[e^{\frac{a_1}{1-\delta_1}(k+2)^{1-\delta_1}} \frac{1}{(k+1)^{\delta_2}} \right]
\end{aligned} \tag{A.3}$$

where $A_1 \triangleq e^{\frac{a_1}{1-\delta_1}(i+1)^{1-\delta_1}}$. For sufficiently large i_0, by Riemann integration properties,

$$\begin{aligned}
\sum_{k=i_0}^{i-1} \left[e^{\frac{a_1}{1-\delta_1}(k+2)^{1-\delta_1}} \frac{1}{(k+1)^{\delta_2}} \right] &\leq \sum_{k=i_0}^{i-1} \left[e^{\frac{a_1}{1-\delta_1}(k+2)^{1-\delta_1}} \frac{1}{(\frac{k}{2}+1)^{\delta_2}} \right] \\
&= 2^{\delta_2} \sum_{k=i_0+2}^{i+1} \left[e^{\frac{a_1}{1-\delta_1}k^{1-\delta_1}} \frac{1}{k^{\delta_2}} \right] \\
&= 2^{\delta_2} A_1 \frac{1}{(i+1)^{\delta_2}} + 2^{\delta_2} \sum_{k=i_0+2}^{i} \left[e^{\frac{a_1}{1-\delta_1}k^{1-\delta_1}} \frac{1}{k^{\delta_2}} \right] \\
&\leq \frac{2^{\delta_2} A_1}{(i+1)^{\delta_2}} + 2^{\delta_2} \int_{i_0+2}^{i+1} \left[e^{\frac{a_1}{1-\delta_1}t^{1-\delta_1}} \frac{1}{t^{\delta_2}} \right] dt.
\end{aligned} \tag{A.4}$$

From (A.3) and (A.4), we have

$$\sum_{k=i_0}^{i-1} \left[\left(\prod_{l=k+1}^{i-1} \big(1 - r_1(l)\big) \right) r_2(k) \right] \leq \frac{2^{\delta_2} a_2}{(i+1)^{\delta_2}} + 2^{\delta_2} a_2 A_1^{-1} \int_{i_0+2}^{i+1} \left[e^{\frac{a_1}{1-\delta_1}t^{1-\delta_1}} \frac{1}{t^{\delta_2}} \right] dt. \tag{A.5}$$

By the partial integration theorem from calculus,

$$\begin{aligned}A_1 &\triangleq e^{\frac{a_1}{1-\delta_1}(i+1)^{1-\delta_1}} \\ &= a_1 \int_{i_0+2}^{i+1} \left[e^{\frac{a_1}{1-\delta_1}t^{1-\delta_1}} \frac{1}{t^{\delta_1}} \right] dt + e^{\frac{a_1}{1-\delta_1}(i_0+2)^{1-\delta_1}} \\ &= a_1 \int_{i_0+2}^{i+1} \left[e^{\frac{a_1}{1-\delta_1}t^{1-\delta_1}} \frac{1}{t^{\delta_2}} t^{\delta_2-\delta_1} \right] dt + A_2. \end{aligned} \tag{A.6}$$

Substituting (A.6) into (A.5), we have

$$\sum_{k=i_0}^{i-1} \left[\left(\prod_{l=k+1}^{i-1} \big(1 - r_1(l)\big) \right) r_2(k) \right] \leq \frac{2^{\delta_2} a_2}{(i+1)^{\delta_2}} + \frac{2^{\delta_2} a_2 \int_{i_0+2}^{i+1} \left[e^{\frac{a_1}{1-\delta_1}t^{1-\delta_1}} \frac{1}{t^{\delta_2}} \right] dt}{a_1 \int_{i_0+2}^{i+1} \left[e^{\frac{a_1}{1-\delta_1}t^{1-\delta_1}} \frac{1}{t^{\delta_2}} t^{\delta_2-\delta_1} \right] dt + A_2}. \tag{A.7}$$

Now we have the claims. 1) If $\delta_1 = \delta_2$, then,

$$\begin{aligned}\sum_{k=i_0}^{i-1} \left[\left(\prod_{l=k+1}^{i-1} \big(1 - r_1(l)\big) \right) r_2(k) \right] &\leq \frac{2^{\delta_2} a_2}{(i+1)^{\delta_2}} + \frac{2^{\delta_2} a_2}{a_1 + A_2 \left(\int_{i_0+2}^{i+1} \left[e^{\frac{a_1}{1-\delta_1}t^{1-\delta_1}} \frac{1}{t^{\delta_2}} \right] dt \right)^{-1}} \\ &\leq \frac{2^{\delta_2} a_2}{(i+1)^{\delta_2}} + \frac{2^{\delta_2} a_2}{a_1}. \end{aligned} \tag{A.8}$$

As $i \to \infty$, we have

$$\lim_{i \to \infty} \sum_{k=i_0}^{i-1} \left[\left(\prod_{l=k+1}^{i-1} \big(1 - r_1(l)\big) \right) r_2(k) \right] \leq 2^{\delta_2} \frac{a_2}{a_1}. \tag{A.9}$$

On the other hand, 2) if $\delta_1 < \delta_2$, as $i \to \infty$, the second term of RHS in (A.7) goes to zero because the denominator increases faster than the numerator. Thus,

$$\lim_{i \to \infty} \sum_{k=i_0}^{i-1} \left[\left(\prod_{l=k+1}^{i-1} \big(1 - r_1(l)\big) \right) r_2(k) \right] = 0. \tag{A.10}$$

APPENDIX B

PROOF FOR LEMMA 3.5.2

Recall the distributed quantile estimation algorithm (3.13). Multiplying both sides of (3.13) by $\frac{1}{N}\mathbf{1}^T$ results in

$$\omega_{\text{avg}}(i+1) = \omega_{\text{avg}}(i) - \alpha(i)y_{\text{avg}}(i) - \eta(i)\xi_{\text{avg}}(i) \tag{B.1}$$

where $\xi_{\text{avg}}(i) \triangleq \frac{1}{N}\mathbf{1}^T\boldsymbol{\xi}(i)$ and from (3.11)

$$\begin{aligned} |y_{\text{avg}}(i)| &\triangleq \left|\frac{1}{N}\mathbf{1}^T\mathbf{y}(i)\right| \\ &= \left|\frac{1}{N}\sum_{n=1}^{N} u(\omega_n(i) - x_n) - p\right| \leq 1. \end{aligned} \tag{B.2}$$

After iteration i, we have the following stochastic difference equation:

$$\omega_{\text{avg}}(i+1) = \omega_{\text{avg}}(0) - \sum_{j=0}^{i}\alpha(j)y_{\text{avg}}(j) - \sum_{j=0}^{i}\eta(j)\xi_{\text{avg}}(j). \tag{B.3}$$

Since the sequence $\{\xi_{\text{avg}}\}$ is time independent and $E\big[\xi_{\text{avg}}(i)\big] = 0$ for all i, we can obtain

$$E\big[\omega_{\text{avg}}(i) - \theta_p\big] \leq \omega_{\text{avg}}(0) - \theta_p + \sum_{j=0}^{i-1}\alpha(j), \tag{B.4}$$

$$\begin{aligned} &E\Big[\big|\omega_{\text{avg}}(i) - \theta_p\big|^2\Big] \\ &\quad\leq \big(\omega_{\text{avg}}(0) - \theta_p\big)^2 + 3\sum_{j=0}^{i-1}\alpha^2(j) + 2\big(\omega_{\text{avg}}(0) - \theta_p\big)\sum_{j=0}^{i-1}\alpha(j) + \sigma^2_{\xi_{\text{avg}}}\sum_{j=0}^{i-1}\eta^2(j) \end{aligned} \tag{B.5}$$

where $\sigma^2_{\xi_{\text{avg}}} \triangleq E\big[\xi^2_{\text{avg}}(i)\big]$ for all i and we use the inequality of (B.2).

Due to (3.14) and (3.15), $\sum_{j=0}^{i-1}\alpha^2(j)$ and $\sum_{j=0}^{i-1}\eta^2(j)$ are bounded, as $i \to \infty$. Also, there exists a decreasing sequence $\eta(i)$ in the form of (3.17) such that $\limsup_{i\to\infty}$ $\eta(i)\sum_{j=0}^{i-1}\alpha(j) = 0$. For example, when $\eta(i) = \frac{1}{(i+1)^{\tau_2}}$ and $\alpha(i) = \frac{1}{(i+1)}$ for $\tau_1 = 1$ and $0.5 < \tau_2 < 1$, there exists $\frac{1}{(i+1)^{\tau_2}}\sum_{j=0}^{i-1}\frac{1}{j+1} < \frac{1}{(i+1)^{\tau_2}}\sum_{j=0}^{i-1}\frac{1}{(j+1)}\frac{(i+1)^{\epsilon}}{(j+1)^{\epsilon}} = \frac{1}{(i+1)^{\tau_2-\epsilon}}\sum_{j=0}^{i-1}$ $\frac{1}{(j+1)^{1+\epsilon}}$ for all $i > 1$ and $0 < \epsilon < 1 - \tau_2 < 0.5$. From Assumption 2, we have

$$\lim_{i\to\infty}\frac{1}{(i+1)^{\tau_2-\epsilon}} = 0, \;\; \lim_{i\to\infty}\sum_{j=0}^{i-1}\frac{1}{(j+1)^{1+\epsilon}} < \infty. \tag{B.6}$$

Then we can obtain $\limsup_{i\to\infty}\eta(i)\sum_{j=0}^{i-1}\alpha(j) = 0$ for $\tau_1 = 1$ and $0 < \epsilon < 1-\tau_2 < 0.5$. Therefore,

$$\limsup_{i\to\infty}\eta(i)E\big[\omega_{\text{avg}}(i) - \theta_p\big] = 0, \;\; \limsup_{i\to\infty}\eta(i)E\Big[\big|\omega_{\text{avg}}(i) - \theta_p\big|^2\Big] = 0. \tag{B.7}$$

APPENDIX C

DERIVATION OF (4.51)

The conditional density model $g_n(m|\mathbf{y}_{nt}, \boldsymbol{\omega}_n(i))$ in (4.51) can be written as

$$\begin{aligned} g_n\big(m|\mathbf{y}_{nt}, \boldsymbol{\omega}_n(i)\big) &= \frac{g_n\big(m, \mathbf{y}_{nt}|\boldsymbol{\omega}_n(i)\big)}{g_n\big(\mathbf{y}_{nt}|\boldsymbol{\omega}_n(i)\big)} \\ &= \frac{g_n\big(m|\boldsymbol{\omega}_n(i)\big)g_n\big(\mathbf{y}_{nt}|m, \boldsymbol{\omega}_n(i)\big)}{g_n\big(\mathbf{y}_{nt}|\boldsymbol{\omega}_n(i)\big)} \end{aligned} \tag{C.1}$$

where $g_n\big(m|\boldsymbol{\omega}_n(i)\big) = \frac{1}{M}$, $g_n\big(\mathbf{y}_{nt}|m, \boldsymbol{\omega}_n(i)\big) = \frac{1}{Z}K_h\big(\mathbf{y}_{nt} - (\mathbf{x}_{nm} - \boldsymbol{\omega}_n(i))\big)$, and $g_n\big(\mathbf{y}_{nt}|\boldsymbol{\omega}_n(i)\big)$ is defined in (4.24). Then, it follows that

$$g_n\big(m|\mathbf{y}_{nt}, \boldsymbol{\omega}_n(i)\big) = \frac{K_h\Big(\mathbf{y}_{nt} - \big(\mathbf{x}_{nm} - \boldsymbol{\omega}_n(i)\big)\Big)}{\sum_{m'=1}^{M} K_h\Big(\mathbf{y}_{nt} - \big(\mathbf{x}_{nm'} - \boldsymbol{\omega}_n(i)\big)\Big)} \tag{C.2}$$

When we set $\mathbf{y}_{nt} = \mathbf{0}$ for all t, we obtain

$$g_n\big(m|\mathbf{y}_{nt} = \mathbf{0}, \boldsymbol{\omega}_n(i)\big) = \frac{K_h\big(\boldsymbol{\omega}_n(i) - \mathbf{x}_{nm}\big)}{\sum_{m'=1}^{M} K_h\big(\boldsymbol{\omega}_n(i) - \mathbf{x}_{nm'}\big)}. \tag{C.3}$$

Eq. (C.3) is the posterior probability of component m, given the current state $\boldsymbol{\omega}_n(i)$.