Consensus Algorithms and Distributed Structure Estimation in Wireless Sensor Networks

by

Sai Zhang

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

Approved April 2017 by the
Graduate Supervisory Committee:

Cihan Tepedelenlioglu, Co-Chair
Andreas Spanias, Co-Chair
Kostas Tsakalis
Daniel Bliss

ARIZONA STATE UNIVERSITY

May 2017

ABSTRACT

Distributed wireless sensor networks (WSNs) have attracted researchers recently due to their advantages such as low power consumption, scalability and robustness to link failures. In sensor networks with no fusion center, consensus is a process where all the sensors in the network achieve global agreement using only local transmissions. In this dissertation, several consensus and consensus-based algorithms in WSNs are studied.

Firstly, a distributed consensus algorithm for estimating the maximum and minimum value of the initial measurements in a sensor network in the presence of communication noise is proposed. In the proposed algorithm, a soft-max approximation together with a non-linear average consensus algorithm is used. A design parameter controls the trade-off between the soft-max error and convergence speed. An analysis of this trade-off gives guidelines towards how to choose the design parameter for the max estimate. It is also shown that if some prior knowledge of the initial measurements is available, the consensus process can be accelerated.

Secondly, a distributed system size estimation algorithm is proposed. The proposed algorithm is based on distributed average consensus and L_2 norm estimation. Different sources of error are explicitly discussed, and the distribution of the final estimate is derived. The CRBs for system size estimator with average and max consensus strategies are also considered, and different consensus based system size estimation approaches are compared.

Then, a consensus-based network center and radius estimation algorithm is described. The center localization problem is formulated as a convex optimization problem with a summation form by using soft-max approximation with exponential functions. Distributed optimization methods such as stochastic gradient descent and diffusion adaptation are used to estimate the center. Then, max consensus is used to

compute the radius of the network area.

Finally, two average consensus based distributed estimation algorithms are introduced: distributed degree distribution estimation algorithm and algorithm for tracking the dynamics of the desired parameter. Simulation results for all proposed algorithms are provided.

To My Family.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my advisors, Dr. Cihan Tepedelenlioglu and Dr. Andreas Spanias for the continuous support of my Ph.D studies and related research, for their motivation, encouragement and constant support. Their guidance helped me in shaping this research work. Special thanks to their enthusiasm, extraordinary patience and serious attitude towards research, which proved to be an immense help to me, all the time. I could not have imagined having better advisors and mentors for my Ph.D.

Besides my advisors, I am grateful to Dr. Konstantinos Tsakalis and Dr. Daniel Bliss for their precious time in serving on my thesis committee member and for their insightful comments and valuable feedback. I would like to thank Dr. Mahesh Banavar for his advice and many discussion. Without his precious support it would not be possible to conduct this research. I would like to extend my appreciation to the School of Electrical, Computer and Energy Engineering at Arizona State University for providing me this opportunity to pursue my Ph.D degree.

I would like to thank all my friends and current and former colleagues in the SenSIP center, Sivaraman Dasarathan, Jongmin Lee, Xue Zhang, Xiaofeng Li, Ruochen Zeng, Ahmed Ewaisha, Jayaraman Jayaraman Thiagarajan, Karthikeyan Natesan Ramamurthy, Huan Song, Jie Fan, David Ramirez, Henry Braun, Abhinav Dixit, Uday Shankar and Sunil Rao for their kindness, help and support.

Most importantly , I would like to thank my parents, for their unconditional love and support, without whom, I could not have completed this work.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

CHAPTER Page

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

Chapter 1

INTRODUCTION

1.1 Wireless Sensor Networks

A wireless sensor network (WSN) is a group of specialized spatially distributed sensors used to monitor and record quantities, such as temperature, pressure, speed, chemical concentration, pollutant levels and so on [4–6]. Sensors in the wireless sensor networks are usually small, inexpensive, memory-limited, lightweight, power efficient and portable devices [4]. Therefore, wireless sensor networks usually have many advantages such as scalability and low power consumption.

The development of wireless sensor networks was motivated by military applications such as battlefield surveillance. Currently, sensor networks are used widely in many industrial and consumer applications such as environmental and habitat monitoring, disaster management, and emergency response applications [7].

1.1.1 Wireless Sensor Networks with Fusion Center

In a wireless sensor network with a fusion center, the spatially distributed sensor nodes are used to monitor physical or environmental conditions and pass their data through the network to the fusion center [5, 6, 8]. An example of the wireless sensor network with fusion center is given in Figure 1.1.

In a centralized wireless sensor network, the fusion center has all the data from sensor nodes. Therefore, functions of the data or measurements from the sensor nodes, such as the average, the maximum or the minimum of the initial measurements can be easily computed at the fusion center. However, there are also disadvantages of

using a centralized wireless sensor network. If a centralized architecture is used, the entire network will collapse if the fusion center crashes. Moreover, centralized wireless sensor networks usually require a large bandwidth since the sensor nodes in the network need to communicate with a common fusion center [9].

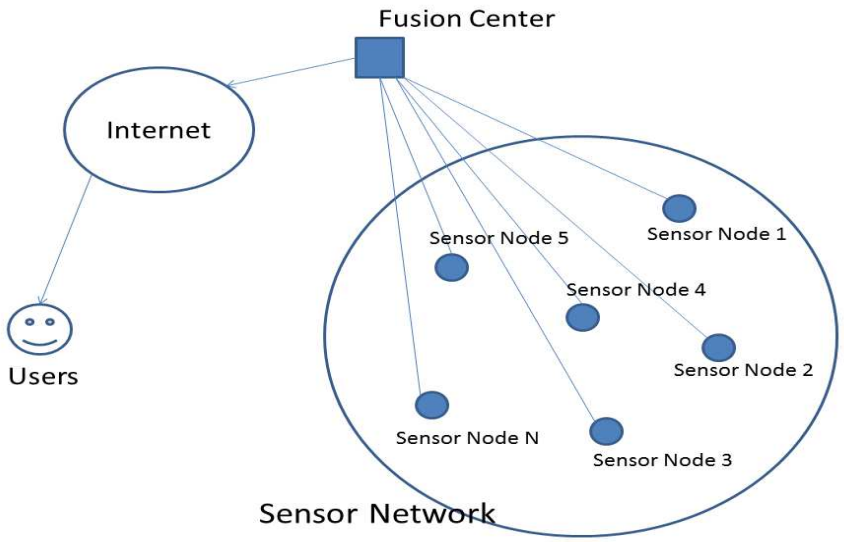

Figure 1.1: An Example of Wireless Sensor Network with A Fusion Center.

1.1.2 Wireless Sensor Network with no Fusion Center

In a distributed network without the fusion center, sensor nodes communicate and exchange data with each other. Usually it is assumed that there is a link between two nodes if their physical distance is smaller than the communication radius and that two nodes can communicate with each other if there is a link between them. An example of the distributed wireless sensor network with no fusion center is given in Figure 1.2. Wireless sensor network without a fusion center can function autonomously without a central node controlling the entire network.

Compared to the centralized network, there are many advantages of using a distributed network without a fusion center: a distributed system is more scalable than

a centralized system with a fusion center and it is more robust to link failures. Since the nodes in a decentralized network communicate only with their neighbors, the sensors require low power [10–12].

However, there are also disadvantages. Function computation in distributed wireless sensor network is usually more complicated than in centralized network. For example, system size estimation can easily be done in a centralized network by letting each node transmit a fixed constant value to the fusion center, but the problem is not straightforward in a network without a fusion center [9, 13]. Moreover, convergence of the states of nodes is slow in a distributed sensor network.

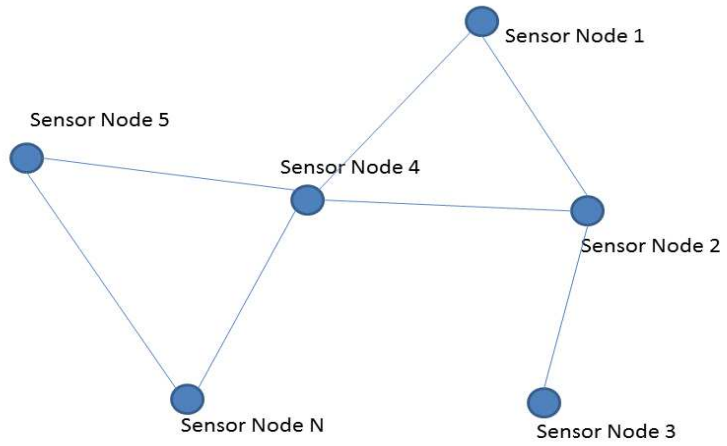

Figure 1.2: An Example of Distributed Wireless Sensor Network with No Fusion Center.

1.1.3 Applications

Wireless sensor network is widely used in both military and industrial applications [8, 14]. A comprehensive review of the wireless sensor network applications is given in [8, 15].

In military applications, wireless sensor network is mainly used for tracking enemies. In [16], based on collaborative signal process in WSNs, an approach for tracking multiple targets is presented. Improved moving vehicle target classification in battlefields using WSNs is introduced in [17], where multimodal fusion in WSN is used.

Wireless sensor network is also widely used in industrial and commercial applications. For example, distributed sensing, detection and estimation applications can be found in [18–22]. Sensors equipped with solar cells for environment protection are mentioned in [8], sensor network can be used to protect the forest without human action for months or even for years. WSN can be used in extreme environments [23–25], for example near a volcano or a flood area, and can function autonomously without manually control. WSNs can also be used in the area of health and medicine [8,26,27]. In a application named "Telemonitoring of Human Physiological", WSN is used to sense and store human physiological data, and the data is used to explore and diagnose medical and health problems. The advantage of using a WSN in health applications is that the sensors are usually small in size, therefore the sensor devises will not affect the everyday lives of patients and allow doctors to identify symptoms earlier or even in real time [28].

1.2 Consensus in Wireless Sensor Networks

The consensus problem has a long history in distributed computing and multi-agent system [29–31]. In distributed wireless sensor networks, consensus is a process where all the sensors in the network achieve global agreement using only local transmissions.

The problem of consensus in WSNs has attracted great interest among researchers in recent years since it is useful in diverse applications, especially in computer science, control and communication areas [32–35]. One of the most popular applications of

consensus is distributed sensor fusion in sensor networks [36]. Distributed average consensus is used in [36] for distributed sensor fusion and the linear least-squares estimator can be obtained at nodes in a distributed way by running average consensus. Max consensus and average consensus can also be used to estimate the environmental data, such as the average temperature or maximum pollution level, etc. In [37], max consensus is used to compensate for clock drift and is used to time-synchronize wireless sensor network nodes. A more comprehensive review of the consensus applications is given in [30], where the applications, including synchronization of coupled oscillators [38], flocking for mobile agents [39] and distributed formation control, are discussed. In the following, two most of the widely used consensus approaches in WSNs are introduced. A review of average consensus is given in Section 1.2.1 and max consensus is described in Section 1.2.2.

1.2.1 Average Consensus

Average consensus is widely used and well studied in the literature [35, 40]. By running the average consensus, the states of nodes converge to the average of the initial values. In [35], a linear average consensus algorithm in the absence of communication noise is introduced. At each iteration time, each node updates its state based on its own state at a previous time and data from its neighbors. During the iterative update step, received data is weighted with a constant weight and it is shown in [35] that the convergence rate is related to the weight and the optimal weight matrix is calculated by solving a convex optimization problem.

In [32], it is assumed that the topology of the network is changing over time and an average consensus algorithm for switching topology is proposed. Delays in the network are also considered. Convergence of algorithm is proved and it is shown in [32] that the convergence speed is related to the algebraic connectivity of the graph.

Average consensus in the presence of communication noise and link failures is considered in [34]. Two algorithms are provided: i) the first algorithm named $\mathcal{A}-\mathcal{ND}$ algorithm uses a decaying step size to control the effect of communication noise; and ii) the second algorithm named $\mathcal{A}-\mathcal{NC}$ algorithm uses the traditional constant weight as in [35]. The iterative updating algorithm only runs for a fixed number of iterations, and the algorithm is restarted and rerun for multiple times. Finally, the sample mean of the results from multiple consensus runs is obtained as the final result.

A distributed nonlinear average consensus algorithm in the presence of communication noise is proposed in [41]. A nonlinear sigmoid function is used to bound the transmit power and a decreasing step size is used to control the effect of communication noise. It is shown in [41] that the nonlinear average consensus converges slower than the linear average consensus, and there is a trade-off between the transmit power and the convergence rate: larger transmit power results in a faster convergence. In [42], average consensus with impulsive noise is considered and a receive nonlinear function is used to ensure convergence of the algorithm.

The above mentioned works all assume that the sensors first sense the environment and then average consensus is applied. In [43], it is assumed that the sensing and averaging states are simultaneous and each node has a new measurement at each iteration time. A time dependent step size is used and the average of all the initial measurements can be obtained at nodes.

In [9], the problem of computing a certain function of the sensed data is considered. The proposed algorithm is based on the average consensus algorithm and universal approximation theorem. A pre-processing function is used to map the sensed data at sensor nodes and a post-processing function is used to process the final average consensus results at nodes. It is proved in [9] that any continuous function of the initial sensed data can be approximated.

There are lots of works and applications using the results of average consensus algorithms. For example, average consensus is used for system size estimation in [2]. In [44] and [45], average consensus is used to estimate the probability mass function of the initial measurements.

1.2.2 Max Consensus

While average consensus is well studied in literature, estimating the average is not always the goal. In various applications, estimating the maximum measured value in the network is necessary [9], [46]. For example, spectrum sensing algorithms that use the OR-rule for cognitive radio applications can be implemented using max consensus [47]. Also, max consensus can be used to estimate the maximum and minimum degrees of the network graph, which are useful in optimizing consensus algorithms [35]. In [48], it is also mentioned that max consensus and min consensus have a broad range of applications in distributed decision-making for multi-agent systems. In [37], max consensus is used to compensate for clock drift and is used to time-synchronize wireless sensor network nodes.

To deal with the problem of finding a unique leader in a group of agents in a distributed way, a max consensus problem in a noise free environment is proposed in [48], where each node in the network collects data from all of its neighbors and finds the largest received data. At each iteration, after comparing its own state and the largest received data, each node updates its state with the max of the two values.

Max consensus algorithms using a similar approach as in [48] are proposed in [46, 49–52]. At each iteration time, every sensor in the network updates its state with the largest measurements it has recovered so far. Reference [46] considers both pairwise and broadcast communications, and analyzes the convergence time. A Max-plus algebra is used in [50] to analyze the max consensus algorithm in a directed

graph. Time dependent graphs are considered in [51], where it is shown that strong connectivity is required for reaching max consensus. A general class of algorithms which can be used for both average and min consensus algorithms is also mentioned in [49].

In [53], the authors extend the work of the weighted power mean algorithm originally proposed by [54] and show that this class of algorithms can also be used to calculate the maximum of the initial measurements when the design parameter is chosen to be infinity. A similar max approximation algorithm is also mentioned in [9] to compute the maximum of the initial measurements in a centralized sensor network with a fusion center. Reference [53] also describes another distributed coordination algorithm for max consensus.

Rumor spreading algorithms mentioned in [55, 56], while not designed specifically for max consensus, may be helpful in max consensus problems. In this setup, one or several nodes know that they have the maximum and can spread the rumor (max) to all the other nodes. If nodes do not know whether they have the maximum or not, a natural way to use rumor spreading for max consensus is to use the max operator. Unfortunately, such an extension of rumor spreading is susceptible to noise on the communication link.

1.3 Contributions of the Dissertation

Here we summarize the main contributions of this dissertation.

- We consider the distributed max consensus in the presence of communication noise. The contribution is in both design and analysis of a max consensus algorithm in wireless sensor networks in the presence of communication noise. Regarding design, the soft maximum, together with non-linear bounded transmissions is proposed. In the proposed max consensus algorithm, every sensor

in the network evaluates a function of its initial observation and a non-linear average consensus algorithm such as those in [41] can be used with a judicious choice of a design parameter β. Regarding analysis, sources of errors in the proposed max consensus algorithm are presented. We show that the parameter of the soft-max function that makes the soft-max approximation accurate also makes the convergence slow. The technical novelty in the analysis is the analytical study of this trade-off. By bounding the sources of error, the needed convergence time is calculated. We also introduce a shifted non-linear bounded function for faster convergence. The analyses provide guidelines for nonlinear transmission design, and algorithm parameter settings to trade-off between estimation error and faster convergence.

- We design a fully distributed node counting algorithm for any connected distributed network with communication noise. The algorithm is based on L_2 norm estimation and average consensus algorithm. A linear iterative average consensus algorithm is used with pre-processed initial values. Then, by applying average consensus and post-processing, each node reaches consensus on an estimate of number of nodes. The performance analysis in the presence of noise is provided and shows that the choice of the initial values at nodes affects performance. The sources of error between the states of nodes and the desired convergence result is quantified. The Fisher information and distribution of the estimate of N at each node is also derived. The analysis not only shows how the performance of the algorithm is affected by the number of iterations, noise variance, and structure of the graph, but also provides guidelines towards choosing the design parameters. The algorithm is fully distributed and nodes do not have to be labeled or know the structure of the graph.

- We consider system size estimation problem using different consensus algorithms such as average consensus and max consensus. We derive the Fisher information and Cramer-Rao bounds for consensus-based system size estimators considering different noise conditions. It is shown that in the absence of noise, the max consensus approach results in a lower Cramer-Rao bound than the average consensus approach. In the presence of communication noise, we demonstrate how the signal-to-noise ratio affects the Fisher information and Cramer-Rao bounds. The results not only present the best estimation variance the algorithms can achieve, but also provide guidelines on how to choose consensus algorithms and initial values for system size estimation.

- We describe the design of a fully distributed network area estimation algorithm. In the proposed algorithm, we assume that nodes only know their own locations, and the network center and radius are estimated. The main contribution is that we formulate the network center estimation problem as an optimization problem. By rewriting the objective function using soft-max approximation, the problem can be turned into a convex optimization problem with a summation form. Therefore, distributed optimization methods such as stochastic gradient descent and diffusion adaptation method can be used to solve the convex optimization problem in a distributed manner. It can be shown that the algorithm converges to an estimate of the center of the network. Then max consensus is used to estimate the radius and the network area is obtained at all nodes . The proposed algorithm is fully distributed and hence nodes do not need to be labeled; two nodes communicate with each other only if they are neighbors.

- We describe the design of a fully distributed degree distribution estimation algorithm in wireless sensor networks. We formulate the degree distribution

estimation problem as an empirical PMF estimation using consensus in the presence of communication noise. The proposed algorithm is fully distributed: sensor nodes do not need to be labeled and each node in the network only needs to know its own degree. How the communication noise affects the performance is also discussed. Finally, we show that the properties of degree distribution can be used to improve the proposed algorithm.

- We design a running consensus algorithm for tracking the dynamic of a desired estimator in a distributed wireless sensor network. A design parameter is used to control the sensitivity of the algorithm, and there is a trade-off between the sensitivity to the dynamic of the estimator and the convergence of the states at nodes. We also compare the proposed algorithm with the existing diffusion method.

1.4 Outline of the Dissertation

The rest of this dissertation is organized as follows. In Chapter 2, a brief review of the graph theory is provided. Later in the chapter, we describe the max consensus using the soft-max approach. The estimation error and convergence speed of the algorithm are also analyzed in Chapter 2. In Chapter 3, we focus on distributed node counting to estimate the system size of the network (number of active nodes in the network) in the presence of communication noise. Performance analysis of the algorithm is given, and different sources of error are explicitly discussed. The overall performance of the system size estimator is given at the end of Chapter 3, where the distribution and the Fisher information of the estimator are calculated, and simulations collaborating the analysis are given. In Chapter 4, a distributed network center and radius estimation algorithm is introduced. Discussion on performance of the al-

gorithm and simulation results are given. In Chapter 5, two distributed estimation algorithms based on consensus algorithms are presented. We first introduce a network degree distribution estimation algorithm based on average consensus and probability mass function estimation. Then, a running consensus algorithm for tracking the dynamics of a desired estimator is described. Finally, future work and conclusions are given in Chapter 6 and 7.

Chapter 2

MAX CONSENSUS USING SOFT-MAX

In this chapter, a distributed consensus algorithm for estimating the maximum value of the initial measurements in a sensor network with communication noise is described. In the absence of communication noise, max estimation can be done by updating the state value with the largest received measurements in every iteration at each sensor. In the presence of communication noise, however, the maximum estimate will incorrectly drift and the estimate at each sensor will diverge. As a result, a soft-max approximation together with a non-linear consensus algorithm is used in our work. Note that part of the works in this chapter can be found in our published pepers in [57, 58].

The following of this chapter is organized as follows. First, a brief review of the graph theory and assumptions on the system model is given in Section 2.1. A brief review of the existing average consensus algorithms is given in Section 2.2. Then, in Section 2.3, the proposed max consensus algorithm is described. The performance of the algorithm is given in Section 2.4. Different sources of error are explicitly discussed and we show that there is a trade-off between the soft-max error and convergence speed. We also show that if some prior knowledge of the initial measurements is available, the convergence speed can be made faster by using an optimal step size in the iterative algorithm. In Section 2.5, a shifted non-linear bounded transmit function is introduced for faster convergence when sensor nodes have some prior knowledge of the initial measurements. Finally, simulation results corroborating the theory are also provided in Section 2.6.

2.1 System Model

2.1.1 Graph Representation

The structure of a distributed wireless sensor network is modeled as 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 set of neighbors of node i is denoted by $\mathbb{N}_i$, i.e., $\mathbb{N}_i = \{j | \{i, j\} \in \mathbb{E}\}$. Two nodes can communicate with each other only if they are neighbors. The number of neighbors of node i is d_i. We use a degree matrix, $\mathbf{D} = \text{diag}[d_1,\ d_2,\ \ldots,\ d_N]$, which is a diagonal matrix containing the degrees of each node. The connectivity structure of the graph is characterized by the adjacency matrix $\mathbf{A} = \{a_{ij}\}$ such that $a_{ij} = 1$ if $\{i, j\} \in \mathbb{E}$ and $a_{ij} = 0$ otherwise. The graph Laplacian of the network $\mathbf{L}$ is defined as $\mathbf{L} = \mathbf{D} - \mathbf{A}$. The Laplacian matrix is basically a matrix representation of a special case of the discrete Laplacian operator, and many properties of graph can be inferred from the Laplacian matrix, for example calculate the number of spanning trees for a graph [59]. For a connected graph, the smallest eigenvalue of the graph Laplacian is always zero, i.e., $\lambda_1(\mathbf{L}) = 0$ and $\lambda_i(\mathbf{L}) > 0, i = 2, \cdots, N$. The zero eigenvalue $\lambda_1(\mathbf{L}) = 0$ corresponds to the eigenvector with all entries one, i.e. $\mathbf{L1} = \mathbf{0}$. The performance of consensus algorithms often depends on $\lambda_2(\mathbf{L})$, which is also known as the algebraic connectivity [32]. Algebraic connectivity of simple and weighted graphs are discussed in [60], where several upper and lower bounds to $\lambda_2(\mathbf{L})$ are also given.

2.1.2 Assumptions on Wireless Sensor Network Model

In distributed sensor network applications and algorithms, the two most commonly used ways of dissemination of information are i) pairwise communications and ii) broadcast communications. In pairwise communications, every node chooses a random neighbor in each iteration and the two nodes exchange information [46, 55, 56].

Broadcast communication model is more commonly used when wireless channel is considered [34, 35, 41]. In this dissertation, we assume broadcast communications, where each node broadcasts its state to its neighbors at each iteration time.

Sensors may use either analog or digital methods to transmit information between neighbors. Digital methods quantize the information, and use digital modulation [61–64]. The bandwidth for the inter-sensor communication channel is directly related to the number of quantization levels. The bandwidth is large when the number of quantization levels is large. The analog transmission methods convey information using amplitude or phase modulation. Analog modulation is also widely considered in consensus algorithms and sensor network applications [31, 35, 65]. We assume analog transmissions in this dissertation.

Noisy communications between nodes is considered in this manuscript. In wireless sensor networks, noisy communication models are widely used in average consensus problems, such as [34, 41, 66], and detection and estimation problem over multiple access channel in the presence of communication noise is considered in [3]. Therefore it is standard practice to adopt noisy communication models between sensor nodes.

To conclude, we have the following assumptions on the system model: i) nodes in the distributed sensor network have their own initial measurements, and the nodes do not know if they have the maximum; ii) the communications in the network are synchronized, and at each iteration, nodes are broadcasting their state values to their neighbors; iii) communications between nodes is analog following [31, 35, 65] and is subject to additive noise; and iv) each node updates its state based on the received data.

2.2 Review of Average Consensus

Distributed average consensus is well studied in literature. In [35], distributed linear average consensus is considered. It is assumed that the communications between nodes is perfect without noise. To compute the average of initial state $\mathbf{x}(0) = [x_1(0) \ \cdots x_N(0)]^T$, the iterative updating algorithm can be expressed as,

$$x_i(t+1) = W_{ii}x_i(t) + \sum_{j\in\mathcal{N}_i} W_{ij}x_j(t), \tag{2.1}$$

where $i = 1, \cdots, N$ is the node index and $t = 0, 1, 2, \cdots$ is the discrete time index. $W \in \mathbb{R}_{N\times N}$ is the weight matrix and W_{ij} is its element in the ith row and jth column. In the algorithm as in equation (2.1), node i is updating its state at time $t+1$ based on its state in the previous time and data received from its neighbors, $j \in \mathcal{N}_i$.

It is shown in [35] that convergence of the algorithm is guaranteed is the following conditions are satisfied,

$$\mathbf{1}^{\mathrm{T}}W = \mathbf{1}^{\mathrm{T}}, \ W\mathbf{1} = \mathbf{1}, \tag{2.2}$$

$$\rho\left(W - \mathbf{1}\mathbf{1}^{\mathrm{T}}\right) < 1, \tag{2.3}$$

where $\rho(\cdot)$ is the spectral radius of a matrix. The choice of the weight matrix W affects the convergence speed of the algorithm and an optimal W for fastest distributed linear averaging is calculated in [35] by solving an optimization problem.

In real world applications of wireless sensor networks, communications between nodes is usually noisy. In [34], a linear iterative averaging algorithm in the presence of communication noise is introduced. To compute the average of initial state $\mathbf{x}(0) = [x_1(0) \ \cdots x_N(0)]^T$, the average consensus algorithm can be expressed as,

$$x_i(t+1) = \left[1 - \alpha(t)d_i\right] x_i(t) + \alpha(t) \sum_{j\in\mathbb{N}_i} \left[x_j(t) + n_{ij}(t)\right], \tag{2.4}$$

where $i = 1, 2, \ldots, N$, and $t = 0, 1, 2, \ldots$, is the time index. The value $x_i(t+1)$ is the state update of node i at time $t+1$ and $n_{ij}(t)$ is the noise associated with the

reception of $x_j(t)$. We assume $n_{ij}(t)$ is Gaussian distributed, $n_{ij}(t) \sim \mathcal{N}(0, \sigma_n^2)$ and is independent across time and space. $\alpha(t)$ is a positive weight factor to bound the variance of communication noise, and is a decreasing function of t.

To ensure convergence, we make the following assumptions on the system model:

Assumptions:

A1) Connected Graph: The graph is connect, i.e. $\lambda_1(\mathbf{L}) = 0$ and $\lambda_i(\mathbf{L}) > 0, i = 2, \cdots, N$.

A2) Independent Noise Sequence: The reception noise is an independent sequence and we assume the noise is Gaussian distributed, i.e.

$$n_{ij}(t) \sim \mathcal{N}\left(0, \sigma_n^2\right), \ \sigma_n^2 \leq \infty. \tag{2.5}$$

A3) Persistence Condition: The positive weight step $\alpha(t)$ is a decreasing function of t, and satisfies the conditions:

$$\alpha(t) > 0, \ \sum_{t=0}^{\infty} \alpha(t) = \infty, \ \sum_{t=0}^{\infty} \alpha^2(t) < \infty. \tag{2.6}$$

The following theorem characterizes the convergence result of the average consensus algorithm in the presence of ciommunciation noise:

Theorem 1. *Assume assumptions **A1)**, **A2)** and **A3)** hold. Let* $\mathbf{x}(t) = [x_1(t) \ \cdots \ x_N(t)]^T$ *be the vector containing the states of nodes at time* t*. Then by running the iterative algorithm as in equation* (2.4)*, there exists a real random variable* θ *such that,*

$$\Pr\left[\lim_{t\to\infty} \mathbf{x}(t) = \theta \mathbf{1}\right] = 1. \tag{2.7}$$

Let $\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i(0)$ *be the average of the initial measurements. Define* $\xi = E[(\theta - \bar{x})^2]$ *be the mean square error. As* $t \to \infty$*, we have*

$$\xi = \left(\frac{\sum_{i=1}^{N} d_i}{N^2}\right) \sigma_n^2 \sum_{t=0}^{\infty} \alpha^2(t). \tag{2.8}$$

As a result, for finite t, ξ is bounded as,

$$\xi \leq \frac{(N-1)\sigma_n^2}{N} \sum_{t=0}^{\infty} \alpha^2(t). \tag{2.9}$$

Proof. The proof is similar to the proof of Theorem 4 and Lemma 5 in [34]. Equation (2.8) can be obtained by assuming the initial measurements are 0 and the nodes in the network is converging to the average of scaled noise samples received at nodes. Equation (2.9) holds since $d_i \leq N-1$. □

In wireless sensor networks, sensors are usually low cost and low power consumption. Therefore, in [41,67,68], nonlinear distributed average consensus are considered and a nonlinear function is used to bound the transmit power. The nonlinear average consensus algorithm will be used in our max consensus algorithm and will be more detailed described in the following of this chapter.

2.3 Max Consensus using the Soft-max

2.3.1 Problem Statement

Consider a wireless sensor network with N sensor nodes, each with a real-valued initial measurement, $x_i, i = 1, 2, \cdots, N$. It is desired that the nodes reach consensus on the maximum value of the initial measurements, $x_{\max} := \max_i x_i$, under the assumption that the sensors have a single state that they update based on local received measurements. Max consensus in the absence of noise is straight forward: the nodes update their states with the largest received measurement thus far in each iteration. Consider the following algorithm at each node:

$$x_i(t+1) = \max\left\{x_i(t), \max_{j\in\mathbb{N}_i} x_j(t)\right\},\ \hat{x}_{\max,i}(t+1) = x_i(t+1). \tag{2.10}$$

However, in the presence of noise, such algorithms will diverge due to positive noise samples. An intuitive explanation is that any positive noise sample will always make the maximum larger if the max operator is used in the max consensus algorithm.

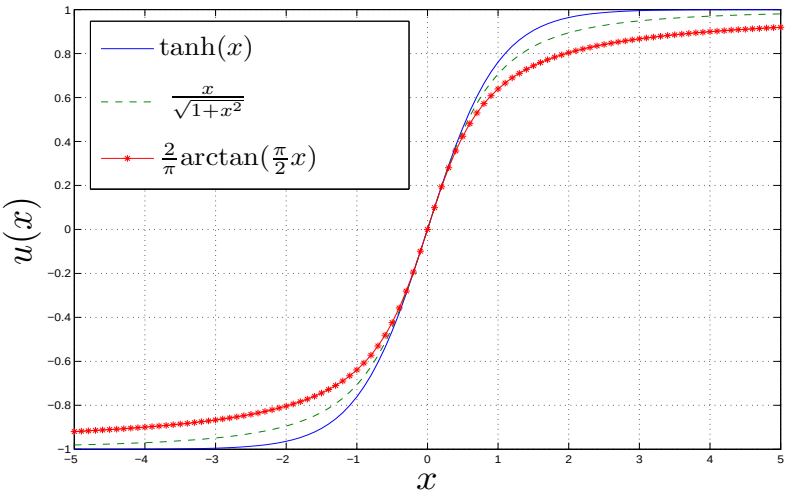

Figure 2.1: Bounded Transmission Functions.

Average consensus is well studied in literature. Existing average consensus algorithms converge to the sample mean of the initial measurements. As a result, the soft-max can be used to calculate the maximum. To relate the soft-max to the sample mean of $\{e^{\beta x_i}\}$, we have,

$$\bar{y} = \frac{1}{N}\sum_{i=1}^{N} e^{\beta x_i} = \frac{1}{N}\sum_{i=1}^{N} y_i(0), \tag{2.11}$$

where $\bar{y}$ is the sample mean of the mapped initial measurements and $y_i(0) := e^{\beta x_i}$. The quantity $\bar{y}$ is computed using an iterative distributed algorithm, in which each sensor communicates only with its neighbors. If the states of all the sensor nodes converge to $\bar{y}$, then the network is said to have reached consensus on the sample average of the mapped initial measurements. The relation between $\bar{y}$ and the soft-max value is given by

$$\mathsf{smax}(\mathbf{x}) = \frac{1}{\beta}\log\sum_{i=1}^{N} e^{\beta x_i} \;= \frac{1}{\beta}(\log N + \log \bar{y}). \tag{2.12}$$

The average consensus algorithms like in [34,41] can be used to achieve consensus in the sensor network. Sensors may adopt either a digital or analog method for transmitting their information to their neighbors. One such method is the linear amplify-and-forward (AF) scheme in which sensors transmit scaled versions of their

measurements to their neighbors where the iterative algorithm may be chosen as the linear consensus algorithm of [34]. However, using the AF technique is not a viable option for consensus on the soft-max. The reason is that accurate approximation of the max value using the soft-max method requires the parameter β to be large, which can result in a large dynamic range of the mapped initial measurements and large transmit power. Moreover, using a linear transmit amplifier is power-inefficient. As a result, a non-linear consensus (NLC) algorithm can be implemented [41]. The consensus on the soft maximum is achieved by letting each sensor map its state value at time t through a bounded function $h(\cdot)$ before transmission to ensure bounded transmit power. To describe the communications between nodes, we use the standard Gaussian MAC so that each node receives a noisy version of the superposition of the transmitted signal from its neighbors. This is because the step sizes are the same across different network links and there is no need to recover the transmitted data separately. Consider the following algorithm with additive noise at the receiver:

$$y_i(t+1) = y_i(t) - \alpha(t)\Big[d_i h(y_i(t)) - \sum_{j\in\mathbb{N}_i} h(y_{ij}(t)) + n_i(t)\Big], \tag{2.13}$$

where $i = 1, 2, \ldots, N$, and $t = 0, 1, 2, \ldots$, is the time index. The value $y_i(t+1)$ is the state update of node i at time $t+1$, $y_{ij}(t)$ is the state value of the j^{th} neighbor of node i at time t, and $n_i(t)$ is the additive noise at node i, which is assumed to be independent across time and space with zero mean and variance σ_n^2. $\alpha(t)$ is a positive step size which satisfies $\sum_{t=0}^{\infty}\alpha^2(t) < \infty$ and $\sum_{t=0}^{\infty}\alpha(t) = \infty$. The node j transmits its information $y_{ij}(t)$ by mapping it through the non-linear function $h(\cdot)$ to constrain the transmitted power. We assume that

$$h(x) = \sqrt{\gamma}\, u(\omega x), \tag{2.14}$$

where $u(x)$ is a normalized non-linear bounded function as in Figure 2.1 and we make the following assumption on $u(x)$:

Assumptions

(A1): $u(0) = 0$. $u(x) = -u(-x)$.

(A2): $\max(u(x)) = 1$.

(A3): The function $u(\cdot)$ is differentiable and invertible, $u'(0) = 1$ and $0 < \frac{du(x)}{dx} \leq 1$.

The parameter γ controls the maximum transmit power and ω is a scale parameter that controls how fast $h(\cdot)$ reaches the maximum. The values of γ and ω affect the performance of the algorithm, for example larger γ value results in faster convergence. Note that invertibility of $h(\cdot)$ is needed for convergence, however there is no need to apply the inverse of $h(\cdot)$ in equation (2.13).

Node i receives a noisy version of the superposition $\sum_{j\in\mathbb{N}_i} h(y_{ij}(t))$. The recursion in equation (2.13) can be expressed in vector form as,

$$\mathbf{y}(t+1) = \mathbf{y}(t) - \alpha(t)[\mathbf{L}h(\mathbf{y}(t)) + \mathbf{n}(t)], \tag{2.15}$$

where $\mathbf{y}(t) = [y_1(t)\ y_2(t)\ \cdots\ y_N(t)]^{\mathrm{T}}$ and $h(\mathbf{y}(t)) = [h(y_1(t))\ h(y_2(t))\ \cdots\ h(y_N(t))]^{\mathrm{T}}$. $\mathbf{L}$ is the Laplacian matrix of the graph and $\mathbf{n}(t)$ is the vector containing the additive reception noise at nodes. Since the noise is i.i.d. with variance σ_n^2, the covariance of $\mathbf{n}(t)$ is $\sigma_n^2\mathbf{I}$. Since (2.13) converges to a value that approximates (2.11), the consensus estimate of the maximum at node i can be written using (2.12) as

$$\hat{x}_{\max_i}(t^*) = \frac{1}{\beta}\left(\log N + \log y_i(t^*)\right), \tag{2.16}$$

where t^* is the iteration at which the algorithm is stopped.

2.3.2 Proof of Convergence

Since the non-linear average consensus approach is used in the max consensus algorithm, the convergence proof will follow the proof in [41] which uses a discrete time Markov process approach [69] (also see Theorem 5 in [41]). Therefore, there exists a finite real random variable θ^* such that,

$$\Pr\left[\lim_{t\to\infty} \mathbf{y}(t) = \theta^*\mathbf{1}\right] = 1, \tag{2.17}$$

where **1** is a column vector with all ones. Equation (2.17) shows that convergence is reached when $t \to \infty$. The following theorem characterizes the random variable θ^*.

Theorem 2. *θ^* in (2.17) is an unbiased estimate of $\bar{y}$, $\mathrm{E}[\theta^*] = \bar{y}$. Its mean square error $\xi_N = \mathrm{E}[(\theta^* - \overline{y})^2]$, and is finite which can be expressed as,*

$$\xi_N = \frac{\sigma_n^2}{N} \sum_{t=0}^{\infty} \alpha^2(t). \tag{2.18}$$

Proof: The proof is a straightforward adaptation of Theorem 3 in [41].

The nodes in the sensor network reach consensus on the random variable θ^* which is an unbiased estimator of the average of the mapped initial measurements, $\mathrm{E}[\theta^*] = \bar{y}$. Then the soft-max of the initial measurements can be obtained using equation (2.12).

2.4 Analysis of the Max Consensus Algorithm

2.4.1 Sources of Error

Let θ_0 be a realization of θ^*. From (2.17) we have that the states of nodes in the sensor network are converging to θ_0 as $t \to \infty$. However, in practice, we need to stop the algorithm at a finite iteration time t^*. There are three sources of error between the true maximum $x_{\max}$ and $\hat{x}_{\max_i}(t^*)$ in (2.16): i) $(\mathsf{smax}(\mathbf{x}) - x_{\max}) = \frac{1}{\beta}(\log N + \log \bar{y}) - x_{\max}$, due to the fact that soft-max approximation will always be larger than the true max; ii) $(\theta_0 - \bar{y})$ caused by communication noise and iii) $(y_i(t^*) - \theta_0)$ cause by finite number of iterations.

In the following subsection, we are going to characterize and analyze these errors.

2.4.1.1 Soft-max error

This is a deterministic error which depends on β, N, and the value of $\mathbf{x}$. We have:

$$x_{\max} \leq \mathsf{smax}(\mathbf{x}) \leq x_{\max} + \frac{1}{\beta} \log N, \tag{2.19}$$

Both inequalities are clearly tight for large β.

2.4.1.2 MSE of the algorithm

The second term $(\theta_0 - \bar{y})$ is due to the presence of communication noise: the state of the sensors does not converge to the sample mean of the mapped initial measurements, instead it converges to a random variable θ^* whose expectation is the sample mean of the mapped initial measurements, $\bar{y}$ from (2.11). This occurs also in linear average consensus in the presence of noise. The mean square error of θ^* is defined as $\xi_N = E[(\theta^* - \overline{y})^2]$ and is characterized as (2.18) in Theorem 2. From (2.18), we see that the mean square error is finite and is small when $\sum_{t=0}^{\infty} \alpha^2(t)$ or σ_n^2 small.

2.4.1.3 Convergence speed

The third cause of error is due to a finite number of iterations: even though $\lim_{t\to\infty} y(t) = \theta_0$, $y(t^*) \neq \theta_0$. However, with a judicious choice of non-linear function $h(\cdot)$ and step size $\alpha(t)$, one can reduce the convergence time. In the rest of this chapter, we will assume that $\alpha(t) = \frac{a}{t+1}, a > 0$, which satisfies $\sum_{t=0}^{\infty} \alpha^2(t) \leq \infty$, $\sum_{t=0}^{\infty} \alpha(t) = \infty$. The convergence speed is analyzed by establishing $\sqrt{t}(\mathbf{y}(t) - \theta_0 \mathbf{1})$ is asymptotically normal with zero mean and some covariance matrix $\mathbf{C}$. The next theorem further quantifies the convergence speed.

Theorem 3. *Let* $2a\lambda_2(\mathbf{L})h'(\theta_0) > 1$ *so that the matrix* $[ah'(\theta_0)\mathbf{B} + \mathbf{I}/2]$ *is stable (every eigenvalue of the square matrix has strictly negative real part) and* $\mathbf{I}$ *is the identity matrix, and* $\mathbf{B}$ *is a diagonal matrix containing all the non-zero eigenvalues of* $-\mathbf{L}$. *Define* $\mathbf{U} = [N^{-1/2}\mathbf{1}\ \mathbf{\Phi}]$ *which is a unitary matrix whose columns are the eigenvectors of* $\mathbf{L}$. *Let* $[\tilde{n}(t)\ \tilde{\mathbf{n}}(t)] = N^{-1}\mathbf{U}^{\mathrm{T}}\mathbf{n}(t)$ *and* $\mathbf{C}_{\tilde{\mathbf{n}}} = \mathrm{E}[\tilde{\mathbf{n}}\tilde{\mathbf{n}}^{\mathrm{T}}]$ *is a diagonal*

matrix, $\mathbf{C}_{\tilde{\mathbf{n}}} \in \mathbb{R}^{(N-1)\times(N-1)}$. *Then as* $t \to \infty$,

$$\sqrt{t}(\mathbf{y}(t) - \theta_0 \mathbf{1}) \sim \mathcal{N}(\mathbf{0}, \mathbf{C}), \tag{2.20}$$

where the asymptotic covariance matrix $\mathbf{C} = N^{-1}a^2\sigma_n^2 \mathbf{1}\mathbf{1}^{\mathrm{T}} + N^{-1}\mathbf{\Phi}\mathbf{S}^{\theta_0}\mathbf{\Phi}^{\mathrm{T}}$,
and $\mathbf{S}^{\theta_0} = a^2 \int_0^\infty e^{(ah'(\theta_0)\mathbf{B}+\mathbf{I}/2)t}\mathbf{C}_{\tilde{\mathbf{n}}}e^{(ah'(\theta_0)\mathbf{B}+\mathbf{I}/2)t}dt$.

The proof is the same as given in Theorem 5 in [41].

The convergence speed is quantified by $\|\mathbf{C}\|$, which is defined to be the largest eigenvalue of the covariance matrix. We show in Appendix that the l_2 norm of the covariance matrix can be expressed as

$$\begin{aligned}\|\mathbf{C}\| &= \max_{\|\mathbf{x}\|\leq 1} \mathbf{x}^{\mathrm{T}}\mathbf{C}\mathbf{x} \\ &= \max\left\{a^2\sigma_n^2,\ \frac{1}{N}\frac{a^2\sigma_n^2}{2ah'(\theta_0)\lambda_2(\mathbf{L}) - 1}\right\}.\end{aligned} \tag{2.21}$$

This norm, $\|\mathbf{C}\|$, can be optimized with respect to a, and the value that minimizes $\|\mathbf{C}\|$ is $a^* = (N+1)/[2N\lambda_2(\mathbf{L})h'(\theta_0)]$. The optimal value for the l_2 norm of the covariance matrix denoted as $\|\mathbf{C}^*\|$ can be represented as

$$\begin{aligned}\|\mathbf{C}^*\| &= \left(\frac{N+1}{2N}\right)^2 \left(\frac{\sigma_n^2}{\lambda_2^2(\mathbf{L})}\right)\left(\frac{1}{h'(\theta_0)}\right)^2 \\ &= \left(\frac{N+1}{2N}\right)^2 \left(\frac{\sigma_n^2}{\lambda_2^2(\mathbf{L})\gamma}\right)\left(\frac{1}{\omega u'(\omega\theta_0)}\right)^2,\end{aligned} \tag{2.22}$$

which is proved in Appendix. The interpretation is that convergence is slower when $\|\mathbf{C}^*\|$ is larger.

It is clear from equation (2.22) that convergence will be fast if $\lambda_2(\mathbf{L})$ large, which implies faster convergence in a more connected graph. Also the value of $\|\mathbf{C}^*\|$ decreases as $h'(\theta_0)$ increases, which shows that the convergence speed depends on the non-linear function and the convergence point. We see from equation (2.22) that larger maximum transmit power γ results in faster convergence. Also note that larger $\omega u'(\omega\theta_0)$ value results in faster convergence as shown in equation (2.22). Therefore, if

θ_0 is approximately known from prior runs of the algorithm and γ is fixed, the value of ω can be set to the solution of the optimization problem: maximize$_\omega$ $\omega u'(\omega\theta_0)$.

By observing the three sources of error mentioned above, we find there is a trade-off between the convergence speed and the soft-max error. To see this, recall that the convergence speed is quantified by $\|\mathbf{C}^*\|$. From the analysis of sources of error, choosing a larger β would reduce the deterministic bias caused by the soft-max mapping (2.19), but degrades the variance term in (2.22). The reason is that $h(\cdot)$ is chosen to be an odd bounded transmission function as in Figure 2.1 with a zero-crossing and steepest slope at the origin, with $h'(x)$ decreasing for $x \geq 0$. Since $\theta_0 \geq 0$, $h^{'}(\theta_0)$ will be small when θ_0 gets larger which increases the value of $\|\mathbf{C}^*\|$ and makes the convergence slower. The convergence point θ_0 will be large when β is chosen large since $\theta_0 \approx \frac{1}{N}\sum_i e^{\beta x_i}$. Therefore a trade-off between the convergence speed and the soft-max error exists: a more accurate soft-max can be obtained by choosing a large β, but this degrades the convergence speed.

2.4.2 Bound on Convergence Time

The convergence speed of the max-consensus algorithm is quantified by the asymptotic covariance matrix. If some prior knowledge about the distribution of the initial measurements is known, the step size can be set based on the expression of $a^* = (N+1)/[2N\lambda_2(\mathbf{L})h^{'}(\theta_0)]$ and $\alpha(t) = a^*/(t+1)$ to make the convergence fast. In this section, we assume that the step size is set to be a^* as mentioned. The trade-off controlled by β balances soft-max error and convergence speed. How much time t^* is needed for the nodes to reach consensus is always an important problem. In this subsection, we will show that by upper bounding the three sources of error in Section 2.4.1, an approximation on the iteration time for reaching consensus can be calculated.

The estimate of the max at iteration time t^* is expressed as (2.16), where $y_i(t^*)$ is the state at node i at time t^*. Of the three errors in Section 2.4.1, note that the error $(\theta_0 - \bar{y})$ can be ignored when the noise variance σ_n^2 is small, or can be reduced by running the consensus several times and taking the average of the results. In the following, we ignore the error $(\theta_0 - \bar{y})$ and calculate the iteration time by bounding the soft-max error denoted by ε_2 and error caused by a finite stopping time t^* denoted by ε_1. When $a = a^*$, the norm of the asymptotic covariance matrix of $(\mathbf{y}(t^*) - \theta_0\mathbf{1})$ can be bounded by ε_1 if

$$\frac{\|\mathbf{C}^*\|}{t^*} \leq \varepsilon_1 \Rightarrow t^* \geq \frac{\|\mathbf{C}^*\|}{\varepsilon_1}. \tag{2.23}$$

The soft-max error is bounded by bounding the upper bound in equation (2.19), which can be expressed as:

$$\frac{\log N}{\beta} \leq \varepsilon_2 \Rightarrow \beta \geq \frac{\log N}{\varepsilon_2}. \tag{2.24}$$

By substituting (2.24) into $\|\mathbf{C}^*\|$, a lower bound of the iteration time needed for reaching consensus can be calculated using (2.23):

$$\begin{aligned} t^* \geq \frac{\|\mathbf{C}^*\|}{\varepsilon_1} &= \left(\frac{N+1}{2N}\right)^2 \left(\frac{\sigma_n^2}{\lambda_2^2(\mathbf{L})}\right) \frac{\left(\frac{1}{h'(\theta_0)}\right)^2}{\varepsilon_1} \\ &= \left(\frac{N+1}{2N}\right)^2 \left(\frac{\sigma_n^2}{\lambda_2^2(\mathbf{L})}\right) \frac{\left(\frac{1}{h'(\frac{1}{N}\sum_i e^{\beta x_i})}\right)^2}{\varepsilon_1} \\ &\geq \left(\frac{N+1}{2N}\right)^2 \left(\frac{\sigma_n^2}{\lambda_2^2(\mathbf{L})}\right) \left(\frac{1}{\varepsilon_1}\right) \left(\frac{1}{h'\left(\frac{1}{N}\sum_i e^{\frac{\log N}{\varepsilon_2} x_i}\right)}\right)^2. \end{aligned} \tag{2.25}$$

The last inequality holds because of (2.24) and using that $h'(x)$ is decreasing function when $x > 0$. We now study how the final lower bound depends on the convergence error, ε_1, and soft-max error, ε_2. It is clear that the bound is inversely related to ε_1. How ε_2 affects the bound depends on the choice of $h(\cdot)$.

In the following, we provide two examples of $h(\cdot)$ and show that how equation (2.25) is affected by ε_2. We will consider two cases. In the first case, $h(x)$ converges to

its maximum value polynomially fast and in the second case it converges exponentially. First consider the polynomial case and let $h(x) \approx \sqrt{\gamma}\left(1 - \frac{1}{x^p+1}\right)$ for large $x > 0$ for $p > 0$. Note that p controls the value of ω in the definition of $h(x)$ in (2.14). Then,

$$h'\left(\frac{1}{N}\sum_i e^{\frac{\log N}{\varepsilon_2}x_i}\right) \approx h'\left(\frac{1}{N}e^{\frac{\log N}{\varepsilon_2}x_{\max}}\right) \tag{2.26}$$

$$= -p\sqrt{\gamma}\,(\log N)\,x_{\max}\frac{\varepsilon_2^{-2}N^{p(\frac{x_{\max}}{\varepsilon_2}-1)}}{\left(N^{p(\frac{x_{\max}}{\varepsilon_2}-1)}+1\right)^2} \tag{2.27}$$

$$\approx -\frac{p\sqrt{\gamma}\,(\log N)\,x_{\max}}{\varepsilon_2^2 N^{p(\frac{x_{\max}}{\varepsilon_2}-1)}}. \tag{2.28}$$

Equation (2.26) holds since when ε_2 is small, the term $x_i = x_{\max}$ dominates. Equation (2.28) shows how ε_2 affects the convergence time when $h(x) \approx \sqrt{\gamma}\left(1 - \frac{1}{x^p+1}\right)$ for large $x > 0$ for $p > 0$. The asymptotically optimal p that minimizes (2.25) is $p^* = 1/\left((\log N)\left(\frac{x_{\max}}{\varepsilon_2} - 1\right)\right)$, and the lower bound of the iteration time needed can be calculated as,

$$t^* \geq \left(\frac{N+1}{2N}\right)^2\left(\frac{\sigma_n^2}{\lambda_2^2(\mathbf{L})}\right)\left(\frac{1}{\varepsilon_1}\right)\left(\frac{\varepsilon_2^4 e^2\left(\frac{x_{\max}}{\varepsilon_2} - 1\right)^2}{\gamma x_{\max}^2}\right). \tag{2.29}$$

On the other hand, if $h(x)$ converges to its final value, $\sqrt{\gamma}$, exponentially fast, we have $h(x) \approx \sqrt{\gamma}\,(1 - e^{-qx})$ for large $x > 0$, with $q > 0$ which controls the value of ω in the definition of $h(x)$ in (2.14). Then,

$$h'\left(\frac{1}{N}\sum_i e^{\frac{\log N}{\varepsilon_2}x_i}\right) \approx h'\left(\frac{1}{N}e^{\frac{\log N}{\varepsilon_2}x_{\max}}\right) \tag{2.30}$$

$$= -q\sqrt{\gamma}\frac{\log N}{N}x_{\max}\left(\varepsilon_2^{-2}N^{\frac{x_{\max}}{\varepsilon_2}}\right)\left(e^{-qN^{\left(\frac{x_{\max}}{\varepsilon_2}-1\right)}}\right) \tag{2.31}$$

$$= -\frac{q\sqrt{\gamma}\,(\log N)\,x_{\max}N^{\left(\frac{x_{\max}}{\varepsilon_2}-1\right)}}{\varepsilon_2 e^{qN^{\left(\frac{x_{\max}}{\varepsilon_2}-1\right)}}}. \tag{2.32}$$

The asymptotically optimal q that minimizes (2.25) is $q^* = N^{\left(1-\frac{x_{\max}}{\varepsilon_2}\right)}$, and the lower bound of the iteration time needed can be calculated as,

$$t^* \geq \left(\frac{N+1}{2N}\right)^2\left(\frac{\sigma_n^2}{\lambda_2^2(\mathbf{L})}\right)\left(\frac{1}{\varepsilon_1}\right)\left(\frac{\varepsilon_2^4 e^2}{\gamma x_{\max}^2\,(\log N)^2}\right). \tag{2.33}$$

When choosing the non-linear bounded function as mentioned above, we have the following observations based on equation (2.25), (2.28) and (2.32): (i) the required convergence time will be longer when the error requirements ε_1 and ε_2 are smaller; (ii) the soft-max error term ε_2 dominates the convergence time in equation (2.25) in both examples; (iii) the required convergence time will be longer when $x_{\max}$ or the system size N is larger; (iv) by comparing equation (2.29) and (2.33), the convergence will be faster when choosing $h(\cdot)$ that converges to its maximum value exponentially fast is appropriate if

$$\left(\frac{x_{\max}}{\varepsilon_2} - 1\right)^2 > (\log N)^{-2}, \tag{2.34}$$

and $h(\cdot)$ that converges to its maximum value polynomially should be chosen otherwise.

Finally, note that estimating the minimum value of the local measurements is also sometimes necessary. The min-consensus can be achieved using the similar initial mapping but choosing $\beta < 0$.

2.5 Shifted Non-linear Bounded Function Used in Max Consensus

An accurate max estimation using the soft-max approach requires the design parameter β to be large. As a result, the exponential function used for initial measurements mapping expands the dynamic range of the initial measurements and the convergence speed is slow. We now give a modified non-linear distributed average consensus method using a shifted non-linear bounded function that can make the convergence process faster if some prior knowledge of the initial measurements is available.

The method is based on the fact that the convergence is faster when the value of $h^{'}(\theta_0)$ is large from equation (2.22). $h(\cdot)$ in the iterative algorithm is replaced by a shifted non-linear function $g(\cdot)$ defined as $g(x) = h(x-T)$, where T is a shift constant.

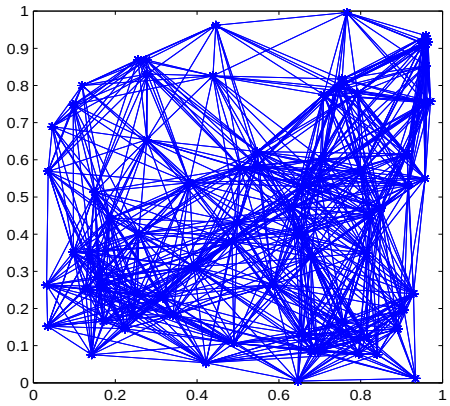

Figure 2.2: Graph Representation Of The Sensor Network, $N = 75$.

In this case, the optimal step size $a_{\mathrm{s}}^* = \left(\frac{N+1}{2N}\right)\left(\frac{1}{\lambda_2(\mathbf{L})g'(\theta_0)}\right) = \left(\frac{N+1}{2N}\right)\left(\frac{1}{\lambda_2(\mathbf{L})h'(\theta_0-T)}\right)$. The convergence speed is quantified by the norm of the asymptotic covariance matrix $\|\mathbf{C}_{\mathrm{s}}^*\|$ and can be expressed as,

$$\|\mathbf{C}_{\mathrm{s}}^*\| = \left(\frac{N+1}{2N}\right)^2 \left(\frac{\sigma_n^2}{\lambda_2^2(\mathbf{L})}\right)\left(\frac{1}{h'(\theta_0 - T)}\right)^2. \tag{2.35}$$

From (2.35), convergence will be faster when $h^{'}(\theta_0 - T)$ is larger. $h^{'}(\theta_0 - T)$ reaches its largest value when $T = \theta_0$ if $h(\cdot)$ is chosen as a sigmoid function with steepest slope at origin. Note that θ_0 is unknown in practice, but one can use prior information on the initial measurements to choose T. If the distribution of the initial measurements is known at the sensor nodes, a reasonable choice of T is to choose it as the expected value of the mapped initial measurements: $T = E[e^{\beta x_i}] \approx \frac{1}{N}\sum_{i=1}^{N} e^{\beta x_i}$.

2.6 Simulations

In this section, simulation results for max consensus algorithms are presented. Different β values are used to trade-off between convergence speed and error between the proposed approach and the true max of the initial measurements.

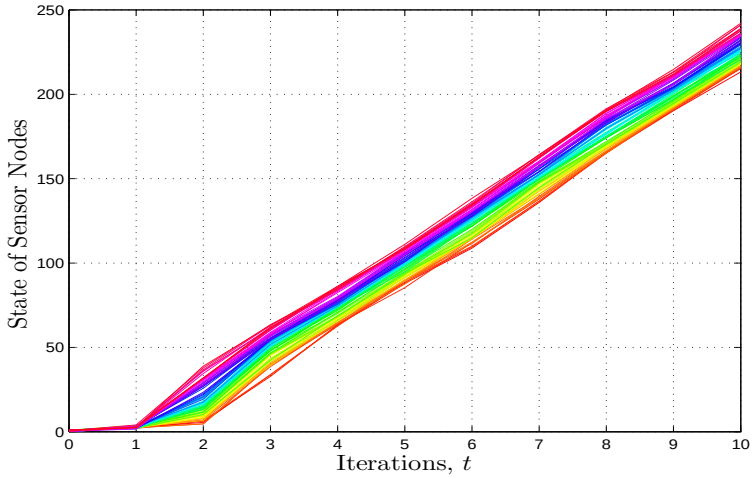

Figure 2.3: Entries of Traditional Max Consensus Result Versus Iterations t (Keep the Largest Measurement at Each Iteration).

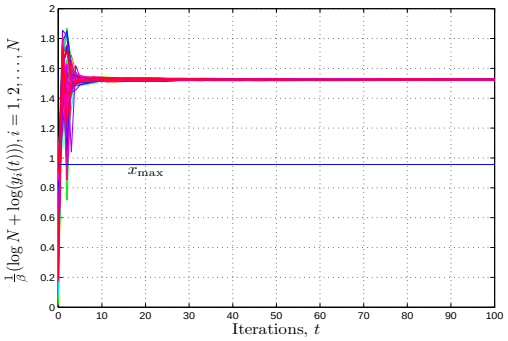

Figure 2.4: Entries of the Consensus Soft Max Result Versus Iterations t, $\beta = 5$, $\omega = 0.015$, $h(x) = \sqrt{\gamma}\tanh(\omega x)$, $\alpha(t) = \frac{4.4473}{t+1}$, $a^* \approx 4.4473$.

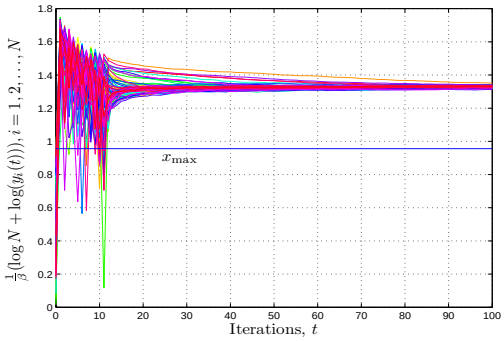

Figure 2.5: Entries of the Consensus Soft Max Result Versus Iterations t, $\beta = 7$, $\omega = 0.015$, $h(x) = \sqrt{\gamma}\tanh(\omega x)$, $\alpha(t) = \frac{61.7513}{t+1}$, $a^* \approx 61.7513$.

2.6.1 Performance of Max Consensus

In the max consensus simulations, the initial measurements $\{x_i\}$ are chosen to be uniformly distributed over $(0, 1)$. Gaussian noise with zero mean and unit variance is added to receiver nodes. The graph of the sensor network is fixed for all simulations with $N = 75$ sensors. as shown in Figure 2.2. The nodes are uniformly located in the $[0, 1] \times [0, 1]$ square. We assume that there is a link between two nodes if their distance

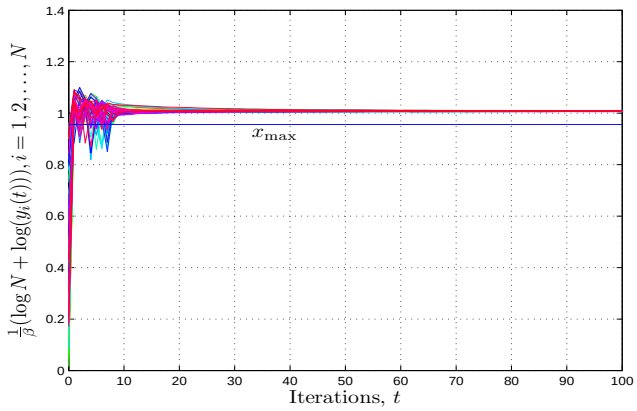

Figure 2.6: Entries of the Consensus Soft Max Result Versus Iterations t, $\beta = 30$, $\omega = 10^{-11}$, $h(x) = \sqrt{\gamma}\tanh(\omega x)$, $\alpha(t) = \frac{5.03\times10^{10}}{t+1}$, $a^* \approx 5.03 \times 10^{10}$.

is smaller than 0.4, also called communication radius . The value of communication radius controls the topology of the distributed network [70], and affects the algebraic connectivity of the network.

In Figure 2.3, the traditional max consensus algorithm is used and each node always keeps the largest measurement from its neighbors. We can see from Figure 2.3 that in the presence of noise, the states of nodes will diverge.

In Figure 2.4 and 2.5, $\hat{x}_{\max_i}(t^*)$ from equation (2.16) for all nodes are plotted to illustrate the convergence of the soft-max result for different β and a values. Note that the actual maximum value is 0.9561 in the simulations. In each of figures, $h(x) = \sqrt{\gamma}\tanh(\omega x)$, with $\omega = 0.015$ and $\gamma = 7.5$dB, note that γ controls the peak transmit power and ω controls the shape of $h(x)$; β is 5 in Figure 2.4 and 7 in Figure 2.5. a is chosen as $(N+1)/[2N\lambda_2(\mathbf{L})h'(\bar{y})] \approx a^*$ and the following observations can be made by comparing the two figures: (i) As β increases, the estimates of the soft-max of $\mathbf{x}$ are closer to the actual value of the maximum value of $\mathbf{x}$ and (ii) As β increases, the convergence is slower, which matches the result in equation (2.22). In Figure 2.6, an accurate max estimate is obtained by setting $\beta = 30$. It is shown that by properly choosing the non-linear function $h(\cdot)$ and step size a, an accurate max consensus

can be attained within a few iterations. From Figure 2.6, we can see that the error between the convergence result and the true max is around 0.06, therefore, Figure 2.6 can be a recommended solution for max estimation in sensor networks.

2.6.2 Performance of Max Consensus with Shifted Non-linear Bounded Function

In the simulation of max consensus using a shifted non-linear bounded function, the initial measurements $\{x_i\}$ are chosen to be uniformly distributed over $(0, 1)$. Gaussian noise with zero mean and unit variance is added to the receiver nodes, $\gamma = 7.5$dB in all simulations. The graph is the same as the max consensus simulation with $N = 75$ sensors. In Figure 2.7 and 2.8, $\hat{x}_{\max_i}(t^*)$ for all the nodes are plotted. In each of figures, $a = 12$, $\omega = 0.01$ and β is 7. In Figure 2.8, shifted non-linear bounded functions are used in transmission, and T is chosen to be the sample mean of the mapped initial measurements. We see that consensus is reached in about 50 iterations in Figure 2.7. In Figure 2.8, with the shifted nonlinear bounded function of Section 2.5, consensus is reached in about 15 iterations. By comparing Figure 2.7 and 2.8, it is shown that using the shifted non-linear bounded function can improve the convergence speed.

Note that overshoots in the simulations may be undesirable in real-world applications. Two solutions to the problem can be to use: i) smaller β value to make the dynamic range of the states smaller; and ii) smaller step size $\alpha(t)$ to make the convergence more smooth with less oscillations. However smaller β value will make the soft-max error larger, and smaller $\alpha(t)$ may results in slower convergence. These are also trade-offs in the proposed algorithm.

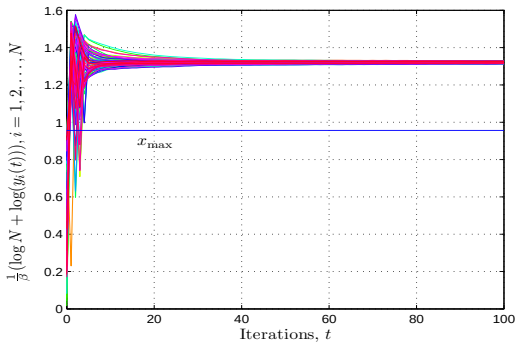

Figure 2.7: Entries of the Consensus Soft Max Result Versus Iterations t, $\beta = 7$, $\omega = 0.01$, $N = 75$, $h(x) = \sqrt{\gamma}\tanh(\omega x)$, $\alpha(t) = 12/(t{+}1)$.

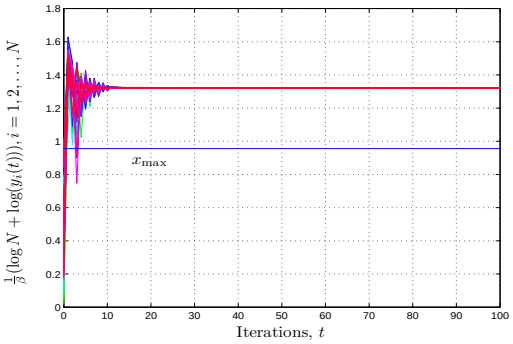

Figure 2.8: Entries of the Consensus Soft Max Result Versus Iterations t, $\beta = 7$, $\omega = 0.01$, $N = 75$, $h(x) = \sqrt{\gamma}\tanh(\omega(x{-}T))$, $T = 138.1045$, $\alpha(t) = 12/(t{+}1)$.

Chapter 3

DISTRIBUTED NODE COUNTING IN WIRELESS SENSOR NETWORKS

In this chapter, a distributed consensus algorithm for estimating the number of nodes in a wireless sensor network in the presence of communication noise is introduced. Note that part of the works in this section is presented in our published papers in [13, 71, 72].

Counting the number of nodes in a decentralized network is essential in several applications. For example, some overlay maintenance protocols require the system size to incorporate a newly joined node in the system [73]. In [57] the soft-max based max consensus method requires the system size (number of nodes). In [74], the sum of the initial values is calculated using a consensus approach by using the system size. In a centralized network with a fusion center, counting the number of nodes in the network is straightforward: each node transmits a fixed constant value to the fusion center; the estimate of number of nodes can be obtained from the aggregate at the fusion center. However, node counting is more challenging in a decentralized system where sensors only have local information.

The idea of the proposed algorithm is based on estimation of the norm of available samples at nodes. Each node generates its own random initial measurements and updates its state by only communicating with its neighbors: the algorithm is a fully distributed algorithm that nodes require no information about the structure of the network. Average consensus algorithm is used to ensure that every node in the network are able to converge to an estimate of the the network size. Different sources of error are explicitly discussed, the Fisher information and the distribution of the final estimate are derived.

3.1 System Model

The structure of the network is modeled as an undirected graph as in Section 2.1. We consider a connected network with N nodes. Nodes have no knowledge about the structure of the graph. We assume that the sensors maintain a state vector and each node broadcasts its state to its neighbors at each iteration. Nodes update the states based on local received states from their neighbors, which is described in Section 2.2. We also assume analog transmissions between nodes [34, 35, 41] and the communication between nodes is imperfect with communication noise, which is i.i.d with 0 mean and variance σ_n^2.

3.2 Node Counting using Average Consensus

3.2.1 Problem Statement

Consider a connected network with N nodes. We assume that the sensors always keep a state vector and they update it based on local received states from their neighbors. We also assume that the communication between nodes is imperfect with communication noise. It is desired that the nodes reach consensus on the number of nodes in the network.

Average consensus is well studied in literature, in which the states of the nodes converge to the sample mean of the initial states. The key to relating the network size N to average consensus is to observe that

$$N = \frac{||\mathbf{x}||_2^2/N}{||\mathbf{x}||_2^2/N^2}. \tag{3.1}$$

In the following of this chapter, we show that the value of the denominator and numerator in (3.1) can be estimated using the average consensus algorithm. Therefore, an estimate of the number of nodes in the network can be obtained.

3.2.2 Node Counting Algorithm

By running the average consensus algorithm as mentioned in Section 2.2, nodes in the sensor network converge to the sample mean of the initial values. As a result, the L_2 estimation method can be used to relate the average of the initial states and the number of nodes in the network. The node counting algorithm can be described in three phases: an estimate of the value of the denominator and numerator in (3.1) can be obtained using the average consensus algorithm in phase I and phase II respectively, and $\hat{N}$ is calculated in phase III by using the consensus results of phases I and II to compute the ratio in (3.1). In the following, details of the three phases of the algorithm are provided.

3.2.2.1 Phase I - L_2 Norm Estimation Consensus

In Phase I of the node counting algorithm, an estimate of the denominator in equation (3.1) is calculated based on L_2 norm estimation and average consensus algorithm. Assume the initial values are $\mathbf{x} = [x_1 \ \cdots \ x_i \ \cdots x_N]$, where x_i is the initial value at node i. Each node in the network generates K initial state values. The initial state values at node i is denoted as $\mathbf{y}_i(0) = [y_{i1}(0) \ \cdots \ y_{iK}(0)]$, where $y_{ik}(0) = r_{ik}x_i, 1 \leq k \leq K$, and r_{ik} are i.i.d. random variables with zero mean and variance one.

By running average consensus algorithm, each node updates the kth element in the state vector of node i at time $t+1$ with

$$y_i^{(k)}(t+1) = [1-\alpha(t)d_i]\, y_i^{(k)}(t) + \alpha(t) \sum_{j \in \mathbb{N}_i} \left[y_{ij}^{(k)}(t) + n_{ij}^{(k)}(t) \right], \tag{3.2}$$

where $n_{ij}^{(k)}$ is the noise associated with the reception of $y_{ij}^{(k)}(t)$ and $\alpha(t)$ satisfies equation (2.6). When t is large, the kth element of node i is converging to a noisy version of the average $\frac{1}{N}\sum_{i=1}^{N} r_i^{(k)} x_i$.

A post processing function $f(\cdot)$ is applied at each node by squaring each element in the state vector and take the average of the result. For node i, the post processed result can be expressed as,

$$f\left(\mathbf{y}_i(t)\right) = \frac{1}{K}||\mathbf{y}_i(t)||^2 = \frac{1}{K}\sum_{k=1}^{K}\left(y_i^{(k)}(t)\right)^2, \tag{3.3}$$

where $\mathbf{y}_i(t) = [y_i^{(1)}(t) \ \cdots \ y_i^{(K)}(t)]$ is the state vector of node i at time t.

Assume the consensus stops at iteration time t^*. To relate the post processed result $f\left(\mathbf{y}_i(t^*)\right)$ at time t^* to the L_2 norm of the initial values $\mathbf{x}$, we have,

$$\begin{aligned}
f\left(\mathbf{y}_i(t^*)\right) &= \frac{1}{K}\sum_{k=1}^{K}\left(y_i^{(k)}(t^*)\right)^2 && (3.4)\\
&\approx \frac{1}{K}\sum_{k=1}^{K}\left(\frac{1}{N}\sum_{i=1}^{N} r_i^{(k)}x_i\right)^2 && (3.5)\\
&= \frac{1}{N^2}\frac{1}{K}\sum_{k=1}^{K}\left(\sum_{i=1}^{N} r_i^{(k)}x_i\right)^2 && (3.6)\\
&\approx \frac{1}{N^2}\mathrm{E}\left[\left(\sum_{i=1}^{N} r_i^{(k)}x_i\right)^2\right] && (3.7)\\
&= \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\left(x_i x_j \mathrm{E}\left[r_i^{(k)}r_j^{(k)}\right]\right) && (3.8)\\
&= \frac{1}{N^2}||\mathbf{x}||_2^2, && (3.9)
\end{aligned}$$

where (3.5) is accurate if the average consensus algorithm in the presence of noise computes the average of the initial state values $r_i^{(k)}x_i$ as will be discussed in Section 3.3. Equation (3.7) holds when K large due to law of large numbers. Equation (3.8) holds since x_i are fixed values and the expectation taken with respect to random variables $r_i^{(k)}$, and (3.9) holds since $r_i^{(k)}$ are i.i.d. random variables with zero mean and variance 1.

Note that in Phase I, K consensus runs are required in (3.2) before the computation of (3.3).

3.2.2.2 Phase II - L_2 Norm Consensus

In Phase II of the node counting algorithm, the numerator in equation (3.1) is calculated based on the definition of L_2 norm and average consensus algorithm. Each node i sets its initial state value to $z_i(0) = x_i^2$. Nodes in the network run:

$$z_i(t+1) = \left[1 - \alpha(t)d_i\right] z_i(t) + \alpha(t) \sum_{j \in \mathbb{N}_i} \left[z_{ij}(t) + n_{ij}(t)\right]. \tag{3.10}$$

After t^* iterations we have,

$$z_i(t^*) \approx \frac{1}{N} ||\mathbf{x}||_2^2, \tag{3.11}$$

where the error in (3.11) is discussed in Section 3.3 similar to (3.5). Note that Phase II requires a single consensus run.

3.2.2.3 Phase III - Node Counting

By comparing the results from equations (3.9) and (3.11), the estimate of number of nodes in the network $\hat{N}_i(t^*)$ at node i at time t^* can be obtained as

$$\hat{N}_i(t^*) = \frac{z_i(t^*)}{f\left(\mathbf{y}_i(t^*)\right)}. \tag{3.12}$$

Note that Phase I and Phase II can be done at the same time by using a $K+1$ state vector containing both the $K \times 1$ process $\mathbf{y}_i(t)$ and the scalar $z_i(t)$.

From (3.12), the algorithm works for any $\mathbf{x} \neq \mathbf{0}$. In the algorithm, x_i can be sensor measurements or values designed for improved performance (see Section 3.3). Random variables $r_i^{(k)}$ are i.i.d. and generated at nodes with mean 0 and variance 1. Two simple ways to choose $r_i^{(k)}$ can be: i) Normally distributed, $r_i^{(k)} \sim \mathcal{N}(0,1)$; ii) Bernoulli distributed with ± 1, i.e. $\Pr[r_i^{(k)} = 1] = \Pr[r_i^{(k)} = -1] = 1/2$. How the value of x_i and $r_i^{(k)}$ affect the performance is detailed analyzed in Section 3.3. In the following subsection, a special case where initial values are chosen to be all equal, $x_i = a$ is considered.

3.2.3 Special Case: Equal x_i

Recall that the node counting algorithm is based on the estimation of $||\mathbf{x}||_2^2$ using average consensus algorithm. The algorithm works for any $\mathbf{x} \neq \mathbf{0}$ value. Therefore, a special case of the node counting algorithm is to let all the initial values to be a fixed constant value a and the initial state vector of the node i is $\mathbf{y}_i(0) = [y_i^{(1)}(0) \ \cdots \ y_i^{(K)}(0)]$ and $y_i^{(k)}(0) = ar_i^{(k)}$.

Then by running the average consensus algorithm as in equation (3.2) and applying the same post processing function after consensus is reached, the final result of Phase I at node i at time t^* can be expressed as,

$$\begin{aligned} f\left(\mathbf{y}_i(t^*)\right) &= \frac{1}{K}\sum_{k=1}^{K}\left(y_i^{(k)}(t^*)\right)^2 \\ &\approx \frac{1}{K}\sum_{k=1}^{K}\left(\frac{1}{N}\sum_{i=1}^{N} ar_i^{(k)}\right)^2 = \frac{a^2}{N}. \end{aligned} \tag{3.13}$$

Since a is a constant known to all nodes, Phase II is not needed and the estimate of N can be obtained as,

$$\hat{N}_i(t^*) = \frac{a^2}{f\left(\mathbf{y}_i(t^*)\right)}. \tag{3.14}$$

Note that the value of a controls the power of transmitted signal from each node, which makes no difference in the absence of communication noise and is chosen in [1] as $a = 1$ along with the choice $r_i^{(k)} \sim \mathcal{N}(0, 1)$. However, in the presence of noise considered herein, a offers a trade off between transmit power and SNR.

3.3 Performance Analysis

In this section, we analyze the performance of the proposed algorithm which starts by identifying different sources of error of the algorithm explicitly in Section 3.3.1. The overall performance analysis is given in Section 3.3.2: the distribution of $\hat{N}_i$ is derived, and how the design parameters affect the performance is given. We also

analyze the Fisher information in Section 3.3.3.

3.3.1 Sources of Error

In this subsection, different sources of error are explicitly discussed. The transient analysis of the consensus result is given in Section 3.3.1.1. We show that the bias of the convergence result is going to zero with number of iterations, and the graph structure and initial states affect the convergence speed of the bias.

The sources of steady state error are also analyzed. The error caused by communication noise is considered in Section 3.3.1.2, where the MSE is shown to depend on the noise variance and the step size. In Section 3.3.1.3, the error in the L_2 norm estimation is described where how the values $r_i^{(k)}$, x_i and K affect the error is studied.

3.3.1.1 Transient of Bias

We now quantify the transient of the bias in the algorithm to see how it decays with the number of iterations. The states of nodes ideally converge to the average state vector $\bar{\mathbf{y}} = [\bar{y}^{(1)} \ \cdots \ \bar{y}^{(k)} \ \cdots \ \bar{y}^{(K)}]^T$, where $\bar{y}^{(k)} = \frac{1}{N}\sum_{i=1}^{N} r_i^{(k)} x_i$. Let $\mathbf{y}^{(k)}(t) = [y_1^{(k)}(t) \ y_2^{(k)}(t) \ \cdots \ y_N^{(k)}(t)]^T$ contain the kth element in the state vector from all N nodes at time t. The convergence rate of the mean of $\mathbf{y}^{(k)}(t)$ is quantified by [34, eqn (61)] as

$$\left|\left|\mathrm{E}\left[\mathbf{y}^{(k)}(t)\right] - \bar{y}^{(k)}\mathbf{1}\right|\right|_2 \leq \left(e^{-\lambda_2(\mathbf{L})\sum_{\tau=0}^{t}\alpha(\tau)}\right)||\mathbf{y}^{(k)}(0) - \bar{y}^{(k)}\mathbf{1}||_2. \tag{3.15}$$

It is clear from equation (3.15) that the convergence is fast if the algebraic connectivity $\lambda_2(\mathbf{L})$ is large, which implies a faster convergence of bias in a more connected graph. To see this more clearly, we further simplify the dependence of (3.15) on the iteration index t by assuming $\alpha(t) = \frac{1}{t+1}$ as in [34] and [41]. We have the following

approximation,

$$\sum_{\tau=0}^{t} \alpha(\tau) = \sum_{\tau=0}^{t} \frac{1}{\tau+1}$$
$$= \ln(t+1) + \gamma + \varepsilon_{t+1} \tag{3.16}$$

where γ is the Euler constant and $\varepsilon_{t+1} \sim \frac{1}{2(t+1)}$ which approaches 0 as t goes to infinity. Therefore, the convergence rate expression in (3.15) can be expressed for large t as,

$$\left|\left|\mathrm{E}\left[\mathbf{y}^{(k)}(t)\right] - \bar{y}^{(k)}\mathbf{1}\right|\right|_2 \leq (t+1)^{-\lambda_2(\mathbf{L})}||\mathbf{y}^{(k)}(0) - \bar{y}^{(k)}\mathbf{1}||_2. \tag{3.17}$$

Equation (3.17) shows that the error $||\mathrm{E}\left[\mathbf{y}^{(k)}(t)\right] - \bar{y}^{(k)}\mathbf{1}||_2$ is polynomial decreasing with t with an exponent given by the algebraic connectivity of the graph.

3.3.1.2 Mean Square Error

Even though $\mathrm{E}[\mathbf{y}^{(k)}(t)] \to \bar{y}^{(k)}$ as $t \to \infty$, the elements in the state vectors do not converge to the true average of the initial states due to the fact that communications between nodes is noisy. Instead, the kth elements of the state vectors for all nodes converge a.s. to a finite random variable $\theta^{(k)}$ as in Theorem 1, which is an unbiased estimator of the average and satisfies the following properties,

$$\mathrm{E}[\theta^{(k)}] = \frac{1}{N}\sum_{i=1}^{N} r_i^{(k)} x_i = \bar{y}^{(k)}, \tag{3.18}$$

$$\zeta^{(k)} = \mathrm{E}\left[\left(\theta^{(k)} - \bar{y}^{(k)}\right)^2\right] \leq \frac{(N-1)\sigma_n^2\beta}{N}, \tag{3.19}$$

where σ_n^2 is the noise variance and $\beta := \sum_{t=0}^{\infty} \alpha^2(t) < \infty$.

It is seen in (3.19) that the error is proportional to σ_n^2 and is bounded if $\alpha(t)$ satisfies equation (2.6).

3.3.1.3 L_2 Norm Estimation Error

It is seen from (3.6) and (3.7) that the L_2 estimation result in (3.9) can be obtained due to the law of large numbers (large K). We now study the effect of K on variance of the L_2 estimation result. The analysis on how the design parameters $r_i^{(k)}$ and x_i will affect the variance is also given. Let

$$Y = \frac{1}{K}\sum_{k=1}^{K}\left(\frac{1}{N}\sum_{i=1}^{N} r_i^{(k)} x_i\right)^2. \tag{3.20}$$

Then it can be shown that $\mathrm{E}[Y] = ||\mathbf{x}||_2^2/N^2$ since $r_i^{(k)}$ are i.i.d. random variables. In the following, the relationship between the value of K and the variance of Y, denoted as σ_Y^2 will be shown.

Let $Z^2 = \left(\frac{1}{N}\sum_{i=1}^{N} r_i^{(k)} x_i\right)^2$ be an unbiased estimator of $||\mathbf{x}||_2^2$ because of (3.9). The variance of Z^2 can be calculated as,

$$\mathrm{Var}[Z^2] = \mathrm{E}[Z^4] - \left(\mathrm{E}[Z^2]\right)^2 \tag{3.21}$$

$$= \frac{\left(\mathrm{E}\left[\left(r_i^{(k)}\right)^4\right] - 1\right)}{N^4}\sum_i x_i^4 + \frac{4}{N^4}\sum_{i<j} x_i^2 x_j^2. \tag{3.22}$$

Then, the variance of Y can be expressed as,

$$\sigma_Y^2 = \mathrm{Var}[Y] = \frac{1}{K}\mathrm{Var}[Z^2] \tag{3.23}$$

Equation (3.22) and (3.23) shows that the variance will be small when K chosen to be large, and it is also related to $r_i^{(k)}$ and $x_i(0)$. Therefore there is a trade-off between the accuracy of the algorithm and the storage at sensor nodes: a more accurate estimate of number of nodes can be obtained if K large, but the nodes need to keep a larger state vector and therefore increase the required storage at nodes.

For deterministic x_i and K values, the distribution of $r_i^{(k)}$ also affects variance of the L_2 norm estimation result. $r_i^{(k)}$ needs to be chosen to be 0 mean and variance 1

as mentioned. In the following, we calculate the L_2 norm estimation variance for two common $r_i^{(k)}$ distributions: (i) For $r_i^{(k)} \sim \mathcal{N}(0,1)$, we have,

$$\mathrm{Var}[Y] = \frac{1}{KN^4}\left\{2\sum_i x_i^4 + 4\sum_{i<j} x_i^2 x_j^2\right\}. \tag{3.24}$$

If initial values are chosen to be equal, $x_i = a$, we have,

$$\mathrm{Var}[Y] = \frac{2a^4 + 2(N-1)a^4}{KN^3} = \frac{2a^4}{KN^2}. \tag{3.25}$$

(ii) If $r_i^{(k)}$ is Bernoulli distributed with $\Pr[r_i = 1] = \Pr[r_i = -1] = 1/2$, we have,

$$\mathrm{Var}[Y] = \frac{4}{KN^4}\sum_{i<j} x_i^2 x_j^2. \tag{3.26}$$

If initial values are chosen to be equal, $x_i = a$, we have,

$$\mathrm{Var}[Y] = \frac{2(N-1)a^4}{KN^3}. \tag{3.27}$$

Note that (3.24) is always larger than (3.26) by $2(\sum_i x_i^4)/KN^4$.

The following theorem characterizes the minimum L_2 norm estimation variance.

Theorem 4. *The distribution of $r_i^{(k)}$ that minimizes the L_2 norm estimation error in equation* (3.22) *and* (3.23) *is Bernoulli distribution with* ± 1 *with probability* 0.5.

Proof. For fixed initial values x_i and system size N, the L_2 estimation variance is related to $\left(\mathrm{E}\left[\left(r_i^{(k)}\right)^4\right] - 1\right)$ as shown in equation (3.22). For any $r_i^{(k)}$ distribution with mean 0 and variance 1, the fourth moment satisfies,

$$\mathrm{E}\left[\left(r_i^{(k)}\right)^4\right] \geq \left(\mathrm{E}\left[\left(r_i^{(k)}\right)^3\right]\right)^2 + 1 \geq 1. \tag{3.28}$$

Equation (3.28) can be obtained from a lower bound on Kurtosis in [75] by setting the variance to be 1. Bernoulli distribution with ± 1 achieves the lower bound 1 and Theorem 4 is proved. □

Based on the above analysis, we have the following observations and conclusions on the L_2 norm estimation error. The L_2 norm estimation variance is inversely proportional to K. By comparing equation (3.24) and (3.26), it is seen that for fixed x_i values, the L_2 norm estimation variance will be smaller if r_{ik} are chosen to be Bernoulli distributed. The difference between equation (3.24) and (3.26) will be small when N is large; When $r_i^{(k)}$ is Bernoulli distributed, the L_2 norm estimation variance achieves 0 if the initial values are chosen as follows:

$$x_1 \neq 0; \quad x_i = 0, i \neq 1. \tag{3.29}$$

However, this choice if not fully distributed since setting $x_1 \neq 0$ requires labeling at least one node.

To sum up, we quantified three sources of errors: lack of convergence in finite iterations, error caused by communication noise and L_2 norm estimation error. In the following subsections, the distribution of $\hat{N}_i$ and Fisher information are calculated, followed by the analysis on how the design parameters will affect the overall performance.

3.3.2 Distribution of $\hat{N}_i$

In this subsection, the distribution of the estimator $\hat{N}_i$ is calculated under a framework where each consensus run is assumed to converge to the sample mean of the initial states potentially with some additive Gaussian noise. We also show how the design parameters affect the bias and the variance of the estimator.

3.3.2.1 Steady state distribution of $\hat{N}_i$

According to Theorem 1, when consensus is reached, nodes in the network converge to a noisy version of the sample mean of the initial states. If this noise is approximated

as Gaussian, the estimate of number of nodes can be calculated from the convergence result. The estimate at node i can be expressed as,

$$\hat{N}_i = \frac{n' + \frac{1}{N}\sum_{i=1}^{N} x_i^2}{\frac{1}{K}\sum_{i=1}^{K}\left(n_i^{(k)} + \frac{1}{N}\sum_{i=1}^{N} r_i^{(k)} x_i\right)^2}, \tag{3.30}$$

where $n_i^{(k)}$ is the kth accumulated noise at node i during phase I of the node counting algorithm mentioned in Section 3.2.2.1, and n' is the accumulated noise during phase II of the node counting algorithm as mentioned in Section 3.2.2.2. When $t \to \infty$, we have $n_i^{(k)} \sim \mathcal{N}\left(0, \frac{1}{N^2}\left(\sum_{i=1}^{N} d_i\right)\sigma_n^2\beta\right)$ and $n' \sim \mathcal{N}\left(0, \frac{1}{N^2}\left(\sum_{i=1}^{N} d_i\right)\sigma_n^2\beta\right)$.

The following theorem characterizes the distribution of $\hat{N}_i$ when N and K are large.

Theorem 5. *For large N and K, the probability density function of $\hat{N}_i$, denoted as $p_{\hat{N}_i}(z)$ is,*

$$\begin{aligned} p_{\hat{N}_i}(z) = {} & \frac{b(z)d(z)}{\sqrt{2\pi}a^3(z)\sigma_1\sigma_2}\left[\Phi\left(\frac{b(z)}{a(z)}\right) - \Phi\left(-\frac{b(z)}{a(z)}\right)\right] \\ & + \frac{1}{\pi a^2(z)\sigma_1\sigma_2} e^{-\frac{c}{2}}, \end{aligned} \tag{3.31}$$

where

$$\begin{aligned} a(z) &= \sqrt{\frac{1}{\sigma_1^2}z^2 + \frac{1}{\sigma_2^2}} \\ b(z) &= \frac{\mu_1}{\sigma_1^2}z + \frac{\mu_2}{\sigma_2^2} \\ c &= \frac{\mu_1^2}{\sigma_1^2} + \frac{\mu_2^2}{\sigma_2^2} \\ d(z) &= e^{\frac{b^2(z) - ca^2(z)}{2a^2(z)}} \\ \Phi(t) &= \int_{-\infty}^{t} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}u^2} du \end{aligned}$$

and

$$\mu_1 = \frac{1}{N}\sum_{i=1}^{N} x_i^2$$
$$\sigma_1^2 = \left(\frac{\sum_{i=1}^{N} d_i}{N^2}\right)\sigma_n^2\beta$$
$$\mu_2 = \frac{\sum_{i=1}^{N} x_i^2}{N^2} + \left(\frac{\sum_{i=1}^{N} d_i}{N^2}\right)\sigma_n^2\beta$$
$$\sigma_2^2 = \left(\frac{2}{K}\right)\left[\frac{\sum_{i=1}^{N} x_i^2}{N^2} + \left(\frac{\sum_{i=1}^{N} d_i}{N^2}\right)\sigma_n^2\beta\right]^2.$$

Proof. The proof relies on the ratio of two independent Gaussian random variables and details of the proof are given in Appendix. □

The distribution of $\hat{N}_i$ is given in Theorem 5. However, how the bias and variance of the estimator will be affected is not clear. In the following, we simplify the results of Theorem 5 by assuming equal initial values $x_i = a$ as in Section 3.2.3. The distribution of $\hat{N}_i$ together with the bias and variance of the estimator is calculated.

3.3.2.2 Steady state distribution with $x_i = a$

When $x_i = a$, the estimate of number of nodes at node i in (3.30) becomes:

$$\begin{aligned}\hat{N}_i &= \frac{\frac{1}{N}\sum_{i=1}^{N} x_i^2}{\frac{1}{K}\sum_{i=1}^{K}\left(n_i^{(k)} + \frac{1}{N}\sum_{i=1}^{N} r_i^{(k)} x_i\right)^2} \\ &= \frac{a^2}{\frac{1}{K}\sum_{k=1}^{K}\left(n_i^{(k)} + \frac{1}{N}\sum_{i=1}^{N} r_i^{(k)} a\right)^2}. \end{aligned} \tag{3.32}$$

When N is large, from the non-identical central limit theorem (Lyapunov central limit theorem), we have,

$$\frac{1}{N}\sum_{i=1}^{N} r_i^{(k)} a \sim \mathcal{N}\left(0, \frac{a^2}{N}\right), \tag{3.33}$$

and $\hat{N}_i$ is scaled inverse-chi-square distributed, and the probability density function can be expressed as,

$$p_{\hat{N}_i}(z) = \left(\frac{(\tau^2\nu/2)^{\nu/2}}{\Gamma(\nu/2)}\right)\left(\frac{e^{\frac{-\nu\tau^2}{2x}}}{z^{1+\nu/2}}\right) \tag{3.34}$$

$$\nu = K, \;\; \tau^2 = \frac{1}{\frac{1}{N} + \frac{\sigma_n^2 \sum_{i=1}^N d_i}{N^2 a^2}\beta}. \tag{3.35}$$

The mean of $\hat{N}_i$ is,

$$\mathrm{E}[\hat{N}_i] = \frac{\nu\tau^2}{\nu - 2} \tag{3.36}$$

$$= N + \frac{2N}{K-2} - \left(\frac{K}{K-2}\right)\left(\frac{N\sigma_n^2\beta\sum_{i=1}^N d_i}{Na^2 + \sigma_n^2\beta\sum_{i=1}^N d_i}\right) \tag{3.37}$$

$$= N + \frac{2N}{K-2} - \left(\frac{K}{K-2}\right)\left(\frac{N\beta\sum_{i=1}^N d_i}{N(\mathrm{SNR}) + \beta\sum_{i=1}^N d_i}\right), \tag{3.38}$$

where $\mathrm{SNR} := \frac{\frac{1}{N}\sum_{i=1}^N x_i^2}{\sigma_n^2} = \frac{a^2}{\sigma_n^2}$. From the steady state mean of $\hat{N}_i$ as in equation (3.37) and (3.38), we have the following conclusions: i) The bias will be small for large K and SNR; ii) $\mathrm{E}[\hat{N}_i] = N$ when we have,

$$a^2 = \frac{(K-2)\sigma_n^2\beta\sum_i d_i}{2N}. \tag{3.39}$$

However, equation (3.39) depends on N and $\sum_i d_i$ which are usually unknown at nodes in practice; and iii) When K and N are large and $K \gg N$, we have the approximation,

$$\mathrm{E}[\hat{N}_i] = N - \frac{\sigma_n^2\beta\left(\sum_i d_i\right)}{a^2} = N - \frac{\beta\sum_i d_i}{\mathrm{SNR}}. \tag{3.40}$$

Equation (3.40) indicates that for large K, the bias in $\hat{N}_i$ given by $\mathrm{E}[\hat{N}_i - N]$ is always negative and can be made small at large SNR.

The variance of $\hat{N}_i$ is,

$$\mathrm{Var}[\hat{N}_i] = \frac{2\nu^2\tau^4}{(\nu-2)^2(\nu-4)} \tag{3.41}$$

$$= \frac{2K^2}{(K-2)^2(K-4)} \left(\frac{N^2a^2}{Na^2 + \sigma_n^2\beta\sum_{i=1}^{N} d_i} \right)^2 \tag{3.42}$$

$$= \frac{2K^2}{(K-2)^2(K-4)} \left(\frac{N^2(\mathrm{SNR})}{N(\mathrm{SNR}) + \beta\sum_{i=1}^{N} d_i} \right)^2 . \tag{3.43}$$

Note that the variance is inversely proportional to K and $K\mathrm{Var}[\hat{N}_i] \sim 2\left(\frac{N^2(\mathrm{SNR})}{N(\mathrm{SNR})+\beta\sum_{i=1}^{N} d_i}\right)^2$ for large K.

The probability density function of $\hat{N}_i$ provides guideline towards how to choose the design parameters. For example, from equation (3.37) and (3.40) we can see that when the SNR is small, we should choose the sum of squares of the step size to be small (β to be small) to make the bias small.

3.3.3 Fisher Information

Fisher information in the absence of communication noise is discussed in Section 3.3.3.1 and the Fisher information in the presence of communication noise is calculated in Section 3.3.3.2 to pinpoint the effect of noise. How the design parameters affect the Fisher information and Cramer-Rao bounds is also presented. Finally, the conditions under which $\hat{N}_i$ is asymptotically efficient (achieves the CRB) is discussed.

3.3.3.1 Absence of communication noise

First we discuss the Fisher information in the absence of communication noise. For nodes in the network, the kth element in the state vector converge by the central

limit theorem to

$$y_i^{(k)}(t) = \frac{1}{N}\sum_{i=1}^{N} r_{ik}x_i \sim \mathcal{N}\left(0, \frac{\sigma_x^2}{N}\right), \tag{3.44}$$

where $\sigma_x^2 = \frac{1}{N}\sum_{i=1}^{N} x_i^2$ and we assume that σ_x^2 does not depend on N. After a straightforward calculation, the Fisher information is given by

$$\mathcal{I}(N) = \mathrm{E}\left[\left(\frac{\partial}{\partial N}\ln g(X;N)\right)^2 \middle| N\right] = \frac{1}{2N^2}, \tag{3.45}$$

where $g(X;N)$ is the probability density function of Gaussian distribution as in equation (3.44). From equation (3.45), we can conclude that when N is large, the choice of x_i will not affect the Fisher information.

Note that equation (3.45) is the Fisher information for one consensus run. For K consensus runs, the Fisher information will be $K/2N^2$. The Cramer-Rao bound is the inverse of Fisher information. Therefore, for K consensus run, a lower bound on the estimation variance for any unbiased network size estimator can be expressed as,

$$\mathrm{Var}\left[\hat{N}_i\right] \geq \frac{2N^2}{K}. \tag{3.46}$$

3.3.3.2 Presence of communication noise

In the presence of communication noise, elements in the state vectors converge to a noisy version of the sample mean of the initial states. We use central limit theorem to approximate the average consensus results with Gaussian distribution. The kth element in the state vector can be expressed as,

$$y_i^{(k)}(t) = \frac{1}{N}\sum_{i=1}^{N} r_{ik}x_i + n_i^{(k)} \sim \mathcal{N}\left(0, \frac{\sigma_x^2}{N} + \sigma_e^2\right), \tag{3.47}$$

where $n_i^{(k)}$ is the accumulated noise at node i. We assume that $n_i^{(k)}$ is Gaussian distributed, $n_i^{(k)} \sim \mathcal{N}(0, \sigma_e^2)$. Note that the noise term $n_i^{(k)}$ can be viewed as a more

general error caused by communication noise and lack of convergence due to finite number of iterations.

Let $n_i^{(k)}$ capture the error caused by communication noise and lack of convergence. If we assume σ_e^2 is not a function of N, the Fisher information can be calculated as,

$$\mathcal{I}(N) = \mathrm{E}\left[\left(\frac{\partial}{\partial N}\ln g(X;N)\right)^2 \middle| N\right] \tag{3.48}$$

$$= \left(\frac{1}{2N^4}\right)\left(\frac{\sigma_x^4}{\left(\frac{\sigma_x^2}{N}+\sigma_e^2\right)^2}\right) \tag{3.49}$$

$$\approx \left(\frac{1}{2N^4}\right)\left(\frac{\sigma_x^2}{\sigma_e^2}\right)^2 \tag{3.50}$$

$$= \left(\frac{1}{2N^4}\right)\mathrm{SNR}^2, \tag{3.51}$$

where $g(X;N)$ is the Gaussian probability density function and the distribution is given in equation (3.47). Equality in (3.50) holds when N is large. By comparing Equation (3.45) and (3.51), we can observe that in the absence of noise, the Fisher information behaves like $O(1/N^2)$, while in the presence of noise it behaves like $O(1/N^4)$ if the SNR does not depend on N. The Fisher information in equation (3.51) also shows that the SNR affects the performance.

If we assume that the consensus is reached and the error $n_i^{(k)}$ is caused by Gaussian noise, then $n_i^{(k)}$ is Gaussian distributed, $n_i^{(k)} \sim \mathcal{N}\left(0, \left(\frac{\sum_{i=1}^N d_i}{N^2}\right)\sigma_n^2\beta\right)$ and it depends on N. We have an interesting finding that if the nodes in the network have same degree $d_i = d$, then $\sigma_e^2 = \left(\frac{d\sigma_n^2}{N}\right)\beta$. We have $y_i^{(k)}(t) = \left(\frac{1}{N}\sum_{i=1}^N r_{ik}x_i + n_i^{(k)}\right) \sim \mathcal{N}\left(0, \frac{\sigma_x^2+d\sigma_n^2\beta}{N}\right)$, and the Fisher information $\mathcal{I}(N) = 1/(2N^2)$ which is same as equation (3.45). This is because after defining $\sigma_y^2 = \sigma_x^2 + d\sigma_n^2\beta$, the Fisher information calculation will be same as in equation (3.48). The above result suggests that if all nodes have the same degree and consensus is reached, the SNR will not affect the Fisher information.

Finally, we compare the estimation variance in equation (3.42) with the Cramer-Rao bound in equation (3.46). For large SNR, the estimation bias in equation (3.40) is negligible so that

$$\begin{aligned}\mathrm{Var}[\hat{N}_i] &= \frac{2K^2}{(K-2)^2(K-4)}\left(\frac{N^2(\mathrm{SNR})}{N(\mathrm{SNR})+\beta\sum_{i=1}^{N} d_i}\right)^2 \\ &\approx \frac{2K^2N^2}{(K-2)^2(K-4)}. \end{aligned} \tag{3.52}$$

The estimation variance in equation (3.52) is always larger than the CRB result in equation (3.46), and it achieves the CRB, in the sense that the ratio of the right hand side of (3.46) to (3.52) converges to 1 as $K \to \infty$. This shows that the proposed estimator is asymptotically efficient.

3.4 Discussion: Fisher Information for Consensus Based Distributed System Size Estimation

An average consensus based distributed system size estimation algorithm is introduced above. In literature, system size estimation using consensus algorithms with randomly generated initial values at the nodes are considered in various works in [1, 2, 13, 76, 77]. In [1], two algorithms for system size estimation are introduced. The first algorithm generates uniformly distributed initial measurements at nodes, and uses max consensus with an ML estimator (uniform + max consensus + ML). The second algorithm uses Gaussian + average consensus + ML. The consensus result is used to infer the system size. It is shown that result is the maximum likelihood estimator for N^{-1}, where N is the system size. In [2], a method based on average consensus with Bernoulli random initial values namely Bernoulli trail method is proposed. It is shown in [2] that the mean square error for the proposed method goes exponentially to zero as a design parameter increases. In [77], the nodes are assumed to be labeled and one node in the network holds an initial value of 1 and all other

nodes are set to 0. The system size can be obtained from the average consensus result.

In this section, we derive the Fisher information and Cramer-Rao bounds for consensus based (average consensus and max consensus) system size estimators considering different noise conditions. It is shown that in the absence of noise, the max consensus approach results in a lower CRB than the average consensus approach. In the presence of communication noise, we demonstrate how the SNR affects the Fisher information and CRBs.

3.4.1 CRB for System Size Estimation in the Absence of Noise

In this section, we assume that the communications between nodes is perfect without any random noise, and consensus is reached. The Fisher information and CRBs are calculated to compare between different approaches (max consensus and average consensus).

3.4.1.1 Max Consensus in the Absence of Noise

In this subsection, we assume that the traditional max consensus (node always keeps the maximum value) is used and consensus is perfectly reached, i.e. nodes converge to the maximum of the initial values. The following theorem characterizes the Fisher information and CRB result.

Theorem 6. *Assume the initial values at nodes x_i are i.i.d. with PDF $f(x)$ and CDF $F(x)$, and $f(x)$ is differentiable. When max consensus is used, the Fisher information for estimate of system size N is,*

$$\mathcal{I}_{\max} = \frac{1}{N^2}. \tag{3.53}$$

The CRB is the inverse of the Fisher information, and a lower bound on the estima-

tion variance can be expressed as

$$\text{Var}\left[\hat{N}\right] \geq N^2. \tag{3.54}$$

The distribution of the initial values at nodes does not affect the Fisher information and CRB.

Proof. The main idea of the proof is that the CDF of the max consensus result is the product of CDF of initial random values at nodes, and the Fisher information can be calculated based on the definition. Details of the proof is given in the appendix. □

Theorem 6 characterizes the best estimation variance that we can achieve if max consensus is used to estimate the system size. It is also shown that if max consensus is used and consensus is perfectly reached, the distribution of the initial values at nodes does not affect the Fisher information about system size estimation.

Note that the result in equation (3.54) is the best estimation variance in one consensus run. In real system size estimation applications, large sample statistics can be used to improve the performance. A more accurate $\hat{N}$ can be obtained by taking the sample mean of multiple consensus results.

3.4.1.2 Average Consensus in the Absence of Noise

The Fisher information for average consensus based system size estimation in the absence of communication noise is calculated in Section 3.3.3.1. For completeness, we use the following theorem to conclude the results:

Theorem 7. *Assume the initial values at nodes x_i are i.i.d. with mean μ and variance σ^2. Also assume that N is large. When average consensus is used, the Fisher information for estimate of system size N is*

$$\mathcal{I}_{\text{avg}} = \frac{1}{2N^2}. \tag{3.55}$$

The CRB is the inverse of the Fisher information, and an lower bound on the estimation variance can be expressed as

$$\text{Var}\left[\hat{N}\right] \geq 2N^2. \tag{3.56}$$

Proof. The proof is based on central limit theorem with N large (also holds for small N and $x_i \sim \mathcal{N}(\mu, \sigma^2)$). Calculation details are given in Section 3.3.3.1. □

From Theorem 7, we see that the distribution of the initial values does not affect the Fisher information when N large, which matches the result in [1] that scaling and translations of initial value distribution do not affect the performance of the optimal estimator. By comparing equation (3.56) with equation (3.54), we can see that system size estimation with max consensus has a lower CRB.

3.4.2 CRB for System Size Estimation in the Presence of Noise

Consensus in wireless sensor networks always suffers from different sources of error such as imperfect communication with noise and lack of convergence in finite time. In this section, error is considered and Fisher information and CRB for system size estimation are derived.

3.4.2.1 Max Consensus in the Presence of Noise

We model the final error at nodes to be Gaussian distributed $e \sim \mathcal{N}(0, \sigma_e^2)$. When max consensus is used for system size estimation, there is no closed form expression for the Fisher information and CRB. However, if we assume the distribution of the initial values has exponential tails and N is large, an upper bound on the Fisher Information can be obtained, the following statement characterizes the result.

Theorem 8. *Assume the initial values at nodes x_i have exponential tail and its tail PDF $\lambda e^{-\lambda x}$. The distribution of the max of the initial values can be approximated us-*

ing Gumbel distribution. Assume that the final error at nodes is Gaussian distributed $e \sim \mathcal{N}(\mu_e, \sigma_e^2)$. *The Fisher information for estimate of system size* N *is bounded by*

$$\mathcal{I}_{\text{nmax}} \leq \left(\frac{1}{N^2}\right)\left(\frac{\lambda^{-2}}{\sigma_e^2 + \lambda^{-2}}\right). \tag{3.57}$$

The CRB is the inverse of the Fisher information, and a lower bound on the estimation variance can be expressed as

$$\text{Var}\left[\hat{N}\right] \geq N^2\left(\sigma_e^2\lambda^2 + 1\right). \tag{3.58}$$

Proof. The proof is based on extreme value theorem and distribution of random variables. Details of the proof is given in the appendix. □

The SNR can be defined as $\text{SNR} = \frac{\lambda^{-2}}{\sigma_e^2}$. From equation (3.57) and (3.58) we see that larger SNR causes a larger Fisher information and lower CRB. From equation (3.57), we also see that in the absence of error with $\sigma_e^2 = 0$, equation (3.57) is the same as the no error case in equation (3.53).

3.4.2.2 Average Consensus in the Presence of Noise

Fisher information and CRB for average consensus based distributed system size estimation are calculated in Section 3.3.3.2. For completeness, we use the following theorem to conclude the calculation,

Theorem 9. *Assume the initial values at nodes* x_i *are i.i.d. with mean* μ *and variance* σ^2, *the final error at nodes is Gaussian distributed* $e \sim \mathcal{N}(0, \sigma_e^2)$. *Also assume that* N *is large. When average consensus is used, the Fisher information for estimate of system size* N *is*

$$\mathcal{I}_{\text{avg}} = \frac{1}{2N^4}\left(\frac{\sigma^4}{\left(\frac{\sigma^2}{N} + \sigma_e^2\right)^2}\right). \tag{3.59}$$

When N *is large, equation* (3.59) *can be approximated as*

$$\mathcal{I}_{\text{avg}} \approx \frac{1}{2N^4}\left(\frac{\sigma^2}{\sigma_e^2}\right)^2 = \frac{1}{2N^4}(\text{SNR})^2, \tag{3.60}$$

where SNR is defined as $\text{SNR} = \frac{\sigma^2}{\sigma_e^2}$. *The CRB is the inverse of the Fisher information, and an lower bound on the estimation variance can be expressed as*

$$\text{Var}\left[\hat{N}\right] \geq 2N^4 \frac{\left(\frac{\sigma^2}{N} + \sigma_e^2\right)^2}{\sigma^4}. \tag{3.61}$$

Proof. The proof is given in Section 3.3.3.2. □

From equation (3.60) and (3.61) we see that larger SNR causes a larger Fisher information and lower CRB. From equation (3.59), we also see that in the absence of error with $\sigma_e^2 = 0$, equation (3.59) is the same as the no error case in equation (3.55).

From the calculation of Fisher information and CRBs for different consensus based system size estimation algorithms, we have the following conclusions. If max consensus is used, the distribution of the initial values does not affect the Fisher information and CRB results, and the max consensus case has a lower CRB than the average consensus case in the absence of communication noise. In the presence of error (caused by lack of convergence and communication noise), it is shown large SNR results in better estimation performance, which is a trade-off in the problem. Also note that traditional max consensus algorithms diverge in the presence of communication noise and therefore max consensus may not be a good choice for system size estimation with noise.

3.5 Simulation Results

3.5.1 Convergence of the Algorithm

The graph of the sensor network is fixed with $N = 75$ as in Figure 2.2. In Figure 3.1 and 3.2, we consider noisy communications and the algorithms based on average and max consensus mentioned in [1] and [2] are implemented for comparison. It is shown in [1, 2] that nodes converge to the $\hat{N} = 75$ in the absence of communication noise.

Simulation results for the uniform + maximum + ML algorithm mentioned in [1] is considered in Figure 3.1. We observe that the algorithm is sensitive to noise and the states of nodes always converge to 0 due to the divergence of max consensus in the presence of noise. In Figure 3.2, we show the simulation results for Bernoulli trail method proposed in [2], where the estimate at node 1 is shown. We see that Bernoulli trail method is also sensitive to communication noise and the state of node 1 is not converging.

In Figure 3.3 - 3.5, we set $K = 1000$, noise variance $\sigma_n^2 = 1$ and $\alpha(t) = 0.1/(t+1)$. In Figure 3.3, the initial values x_i are fixed and generated from a Gaussian distribution with zero mean and variance 25, and $r_i^{(k)}$ are Bernoulli distributed with ± 1 so that SNR = 13.98 dB. In Figure 3.4, the initial values are set to be fixed to $x_i = 5$ and $r_i^{(k)}$ are Bernoulli distributed with ± 1, and SNR = 13.98 dB. In Figure 3.5, the initial values are set to be fixed $x_i = 5$ and $r_i^{(k)}$ are chosen as $r_i^{(k)} \sim \mathcal{N}(0, 1)$, and SNR = 13.98 dB. In Figure 3.3, node counting algorithm described in Section 3.2.2 is used, and method in Section 3.2.3 is used in the simulations in Figure 3.4 and 3.5. From Figure 3.3 - 3.5, we see that the number of nodes can be estimated using the proposed node counting algorithm in the presence of communication noise.

In Figure 3.6, the mean square error for $\hat{N}(t)$, denoted as $\mathrm{E}\left[\left(\hat{N}(t) - N\right)^2\right]$, is plotted. x_i and $r_i^{(k)}$ are chosen to be different values as shown in the figure. We assume noisy communication with $\sigma_n^2 = 1$ and $K = 1000$. From Figure 3.6, we can see that in the presence of communication noise, larger x_i values result in a better performance since the signal to noise ratio is larger; We can also see from Figure 3.6 that for the same x_i value, MSE are almost the same for different $r_i^{(k)}$ distributions. This is because that when N is large, equation (3.24) and (3.26) are almost equal.

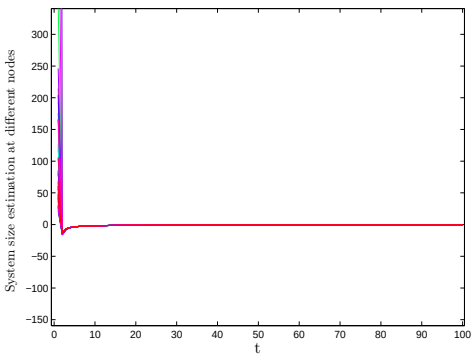

Figure 3.1: Simulation Result for Uniform + Maximum + ML Algorithm in [1]: Node Counting Result Versus Number of Iterations t. $\sigma_n^2 = 0.001$ and $K = 1000$.

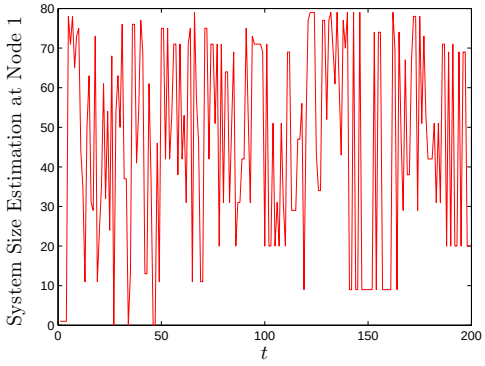

Figure 3.2: Simulation Result for Bernoulli Trail Algorithm in [2]: Node Counting Result at Node 1 Versus Number of Iterations t. $\sigma_n^2 = 1$ and $K = 1000$.

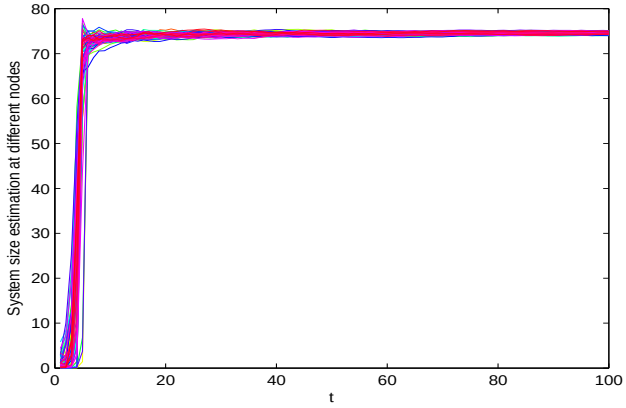

Figure 3.3: Entries of Node Counting Result Versus Number of Iterations t. $x_i(0) \sim \mathcal{N}(0, 25)$, $\sigma_n^2 = 1$ and $r_i^{(k)}$ Bernoulli Distributed with ± 1. $\alpha(t) = 0.1/(t+1)$ and $K = 1000$.

3.5.2 PDF of $\hat{N}$

In Figure 3.7 and 3.8, the probability density function of $\hat{N}$ is plotted based on equation (3.34). The network is the same as Figure 2.2. In Figure 3.7, we fix the SNR and the figure shows how the value of K affects the distribution of $\hat{N}$. While in Figure 3.8 we fix K and the figure shows how the SNR affects the distribution of $\hat{N}$.

From Figure 3.7, we can conclude the bias and the variance result in Section 3.3.2.2:

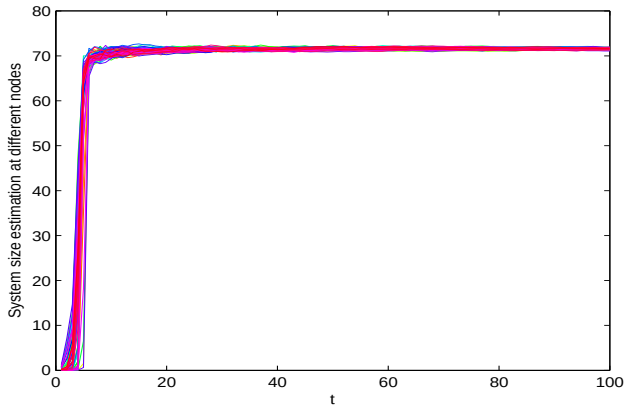

Figure 3.4: Entries of Node Counting Result Versus Number of Iterations t. $x_i(0) = a = 5$, $\sigma_n^2 = 1$ and $r_i^{(k)}$ Bernoulli Distributed with ± 1. $\alpha(t) = 0.1/(t+1)$ and $K = 1000$.

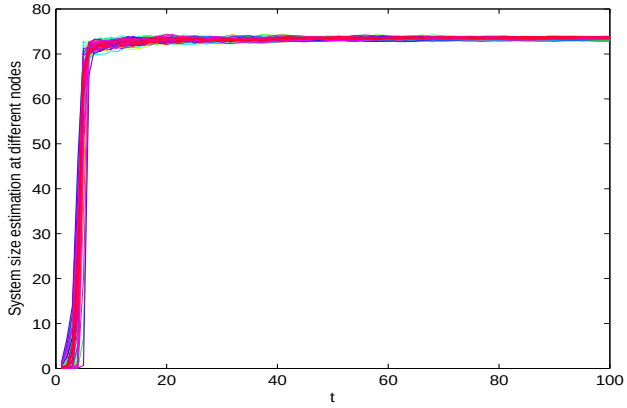

Figure 3.5: Entries of Node Counting Result Versus Number of Iterations t. $x_i(0) = a = 5$, $\sigma_n^2 = 1$ and $r_i^{(k)} \sim \mathcal{N}(0,1)$. $\alpha(t) = 0.1/(t+1)$ and $K = 1000$.

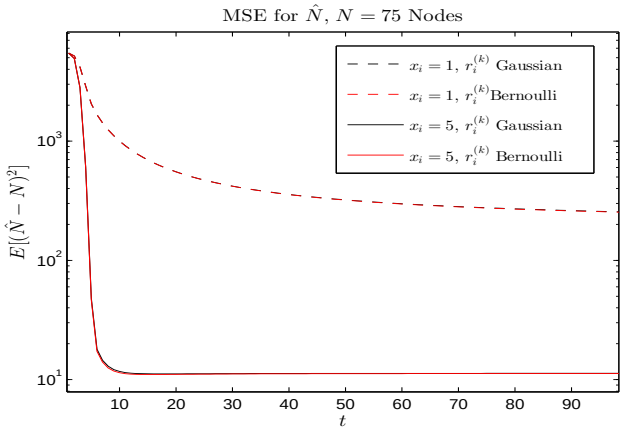

Figure 3.6: MSE Versus t, Noisy $\sigma_n^2 = 1, K = 1000$.

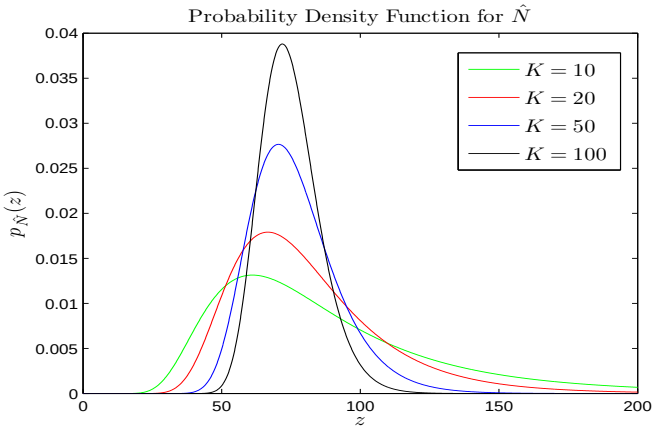

Figure 3.7: PDF for $\hat{N}$ with Different K Values, SNR = 13.98dB, $\alpha(t) = 0.1/t$.

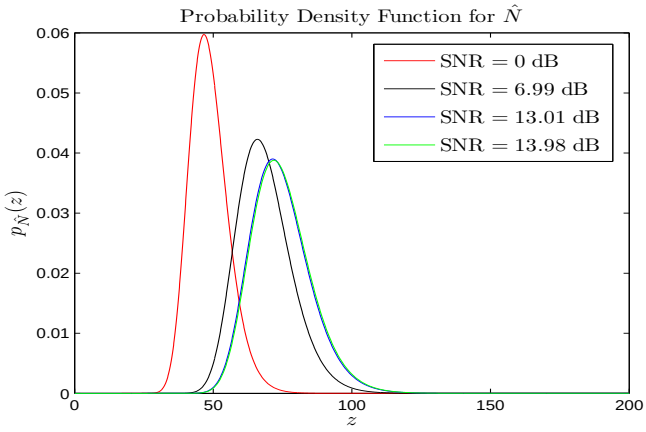

Figure 3.8: PDF for $\hat{N}$ with Different SNR Values, $K = 100$, $\alpha(t) = 0.1/t$.

When K gets larger, the bias and variance of the estimator get smaller. From Figure 3.8, we see that when the SNR is larger, the bias of the estimator gets smaller, however the variance of the estimator gets larger. We also see from Figure 3.8 that for fixed K value and large enough SNR, the distribution of $\hat{N}$ will almost be the same as SNR increases (by comparing the probability density function for SNR = 13.01dB and SNR = 13.98dB).

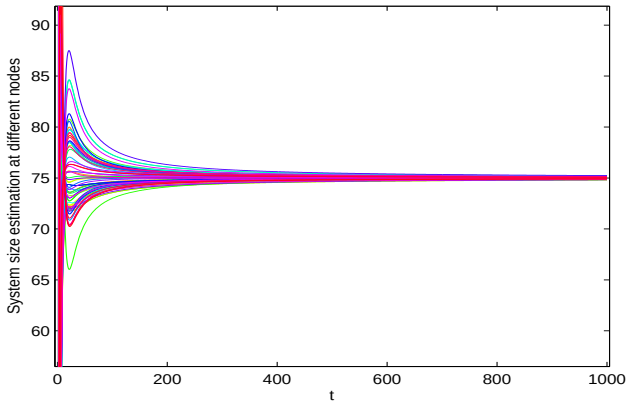

Figure 3.9: $\hat{N}(t)$ at Different Nodes, $K = 1000$, $r_i^{(k)}$ Bernoulli Distributed.

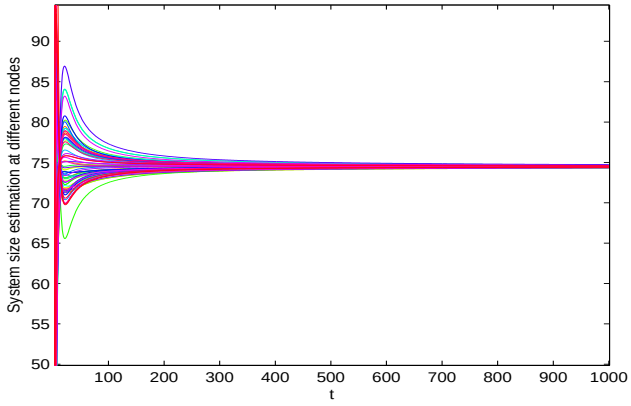

Figure 3.10: $\hat{N}(t)$ at Different Nodes, $K = 1000$, $r_i^{(k)}$ Gaussian Distributed.

3.5.3 Special Initial Values x_i as in (3.29)

In Figure 3.9 and 3.10, a special case mentioned in Section 3.3.1.3 is considered. The initial values x_i are chosen as equation (3.29) and we assume in the absence of noise, $\sigma_n^2 = 0$. The design parameter $r_i^{(k)}$ is Bernoulli distributed in Figure 3.9 and Gaussian distributed in Figure 3.10. From the simulations we can see that when $r_i^{(k)}$ be Bernoulli distributed, the states of nodes converge almost exactly to $\hat{N} = 75$ when $t \to \infty$. When choose $r_i^{(k)}$ to be Gaussian distributed, the error is small.

In Figure 3.11, initial values are chosen as equation (3.29) and the MSE versus t is plotted. From the figure we have the following observations: i) In the absence of

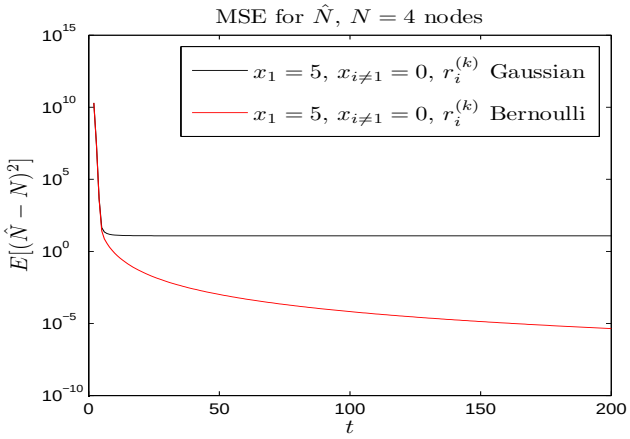

Figure 3.11: MSE Versus t (4 Nodes Network with Star Topology), $x_1 = 5, x_{i\neq 1} = 0$, $\sigma_n^2 = 0$ and $K = 1000$.

communication noise, the MSE achieves 0 as $t \to \infty$ if $r_i^{(k)}$ is Bernoulli distributed; and ii) The MSE will be small when $r_i^{(k)}$ is Gaussian distributed, but does not achieve 0.

However, choosing the initial values as in equation (3.29) is difficult in practice since this is not a distributed way to choose the initial values, as mentioned in Section 3.3.1.3.

3.5.4 Small Network with $N = 4$

In this subsection, a network of 4 nodes with star topology is considered. In Figure 3.12, we set $x_i = a = 5$, $r_i^{(k)}$ are chosen to be Gaussian and Bernoulli distributed and $\sigma_n^2 = 1$. MSE versus the number of iterations is plotted.

We see from the Figure 3.12 that when N is small, choosing $r_i^{(k)}$ to be Bernoulli yields a better performance since the variance of the L_2 norm estimation result will be smaller from equation (3.25) and (3.26).

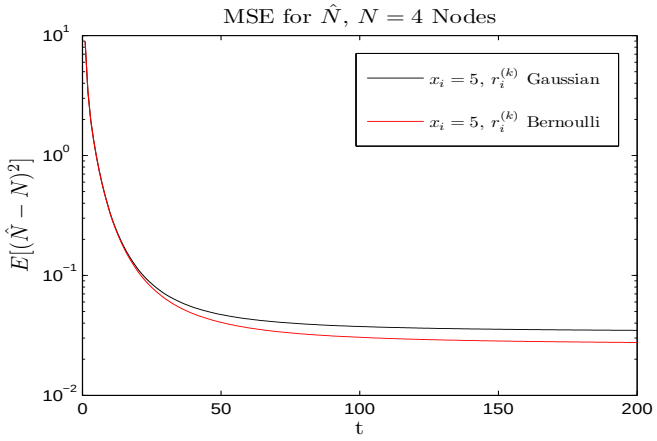

Figure 3.12: MSE versus t (4 Nodes Network with Star Topology), Noisy $\sigma_n^2 = 1, K = 1000$.

Chapter 4

DISTRIBUTED NETWORK CENTER AND RADIUS ESTIMATION

In this chapter, we have addressed the problem of locating the center and estimating the radius of a distributed wireless sensor network (WSN). The network center and radius information can be used to infer the coverage area of the WSN. Center, radius and coverage area of a WSN are useful in many applications. For example, the center and area information are helpful for locating a service center in a network [78]. It is mentioned in [79] that the knowledge of the area of the wireless sensor network and the total number of nodes in the network can be used to decide the optimal connection between sensor nodes. The required power at sensor nodes also depends on the area of the network. It is reported in [80] that energy-efficient scheduling in a WSN depends on the coverage area of the network. However, it is often hard to estimate the network center and area in a distributed WSN where sensors only have local information. In this chapter, we foucs on the problem of estimating the smallest circle or sphere that covers all sensor nodes; the center and radius of the network area are estimated in a fully distributed manner. Note that part of the works in this section will be included in our future publications.

4.1 System Model

The distributed WSN is modeled as an undirected connected graph. Two nodes can communicate with each other only if they are neighbors. It is assumed that each sensor knows its own location and the communications between sensor nodes is perfect without communication noise. We use the smallest circle (2-D) or sphere (3-D) that covers all sensor nodes to represent the network coverage area. Estimating

the smallest covering circle or sphere is basically estimating the center and radius. An example of a 2-D WSN is given in Figure 4.1.

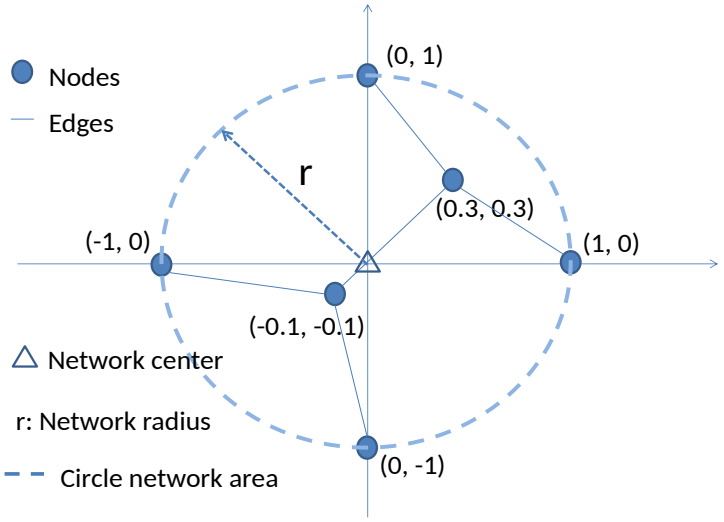

Figure 4.1: A Distributed Network (2-D) with $N = 6$ Nodes with Network Center at the Origin and Radius 1.

4.2 Review of Mathematical Background

For completeness, we briefly review the mathematical background including soft-max approximation, some basic distributed optimization methods and distributed max consensus using max operator, which will be used in the proposed algorithm.

4.2.1 Review of Soft-max Approximation

Recall that the soft-max function can be used to approximate the maximum. The soft maximum of a vector $\theta = [\theta_1 \ \theta_2 \ \cdots \ \theta_N]$ is denoted as

$$\text{smax}(\theta) = \frac{1}{\beta} \log \sum_{i=1}^{N} e^{\beta\theta_i}, \tag{4.1}$$

where $\beta > 0$ is a design parameter and the soft-max approximates the maximum for large β. The soft-max in equation (4.1) is always larger than the maximum value of

θ. The difference is small when β is large:

$$\max(\theta) \leq \text{smax}(\theta) \leq \max(\theta) + \frac{1}{\beta} \log N \tag{4.2}$$

4.2.2 Review of Distributed Optimization

In this subsection, we briefly review two distributed optimization methods: stochastic gradient descent and diffusion adaptation method.

Stochastic gradient descent is an approximation of the traditional gradient descent optimization. If the objective function can be written as a summation of differentiable functions:

$$J^{\text{global}}(\omega) = \sum_{i=1}^{N} J_i(\omega), \tag{4.3}$$

where the parameter ω that minimizes $J(\omega)$ is to be estimated. Then the gradient in the standard gradient descent, $\nabla J(\omega)$ can be approximated by a single gradient term, $\nabla J_i(\omega)$. The iterative updating rule for stochastic gradient descent can be express as

$$\omega := \omega - \eta \nabla J_i(\omega), \tag{4.4}$$

where η is the step size.

In [81], a distributed optimization algorithm based on diffusion adaptation is introduced. The diffusion adaptation allows the nodes in the network to cooperate [82], therefore makes the convergence speed faster than the stochastic gradient descent approach in equation (4.4). It consists of two steps: in the first step, an intermediate result is calculated using combination of local gradient values from all neighbors; and then in the second step, each node aggregates the intermediate results from its neighbors to calculate the the optimal ω. The algorithm and the updating rule is more detailed expressed in Section 4.3.

4.2.3 Review of Max Consensus

It is mentioned in the previous sections that there are different max consensus algorithms. In this section, we briefly review the simple max consensus algorithm with max operator. The updating rule is straightforward: the nodes update their states with the largest received measurement they receive in each iteration. Let $r_i(t)$ be the state of node i at time t, the updating rule can be expressed as

$$r_i(t+1) = \max\left\{r_i(t), \max_{j\in\mathbb{N}_i} r_j(t)\right\}. \tag{4.5}$$

It is proved in [46] and [50] that by running the iterative algorithm, states of nodes converge to the maximum of the initial states in finite time.

4.3 Estimation of Network Center and Radius

4.3.1 Problem Statement

Consider a connected distributed wireless sensor network with no fusion center. Each node in the network only knows its own location. We assume that the each sensor always keeps a single state and the sensors update their states by exchanging their states with their neighbors. It is desired that the nodes reach consensus on the center and radius of the network using only local communications.

The main idea of the algorithm is to use soft-max approximation to formulate the center estimation problem as a convex optimization problem. The object function is written as a summation of differentiable functions using soft-max function and distributed optimization methods such as stochastic gradient descent and diffusion methods can be used to estimate the center. After all nodes obtain the estimate of the center, max consensus is used for distributed radius calculation and the network area can be obtained at nodes.

The center O of the network is defined to be the point that minimizes the maximum distance between O and all nodes [78]. In 2-D case, this can be posed as the following optimization problem:

$$\text{minimize}_{x,y} \quad \max_i \left\{(a_i - x)^2 + (b_i - y)^2\right\}, \tag{4.6}$$

where (a_i, b_i) is the location of node i and the location of the center $O(x, y)$ minimizes (4.6). In 3-D case, the optimization problem can be written as

$$\text{minimize}_{x,y,z} \quad \max_i \left\{(a_i - x)^2 + (b_i - y)^2 + (c_i - z)^2\right\}, \tag{4.7}$$

where (a_i, b_i, c_i) is the location of node i and the location of the center $O(x, y, z)$ minimizes (4.7). Note that the formulation in equation (4.6) and (4.7) can be extended to higher dimension cases such as 4-D. In the following of this chapter, we will focus on calculating the network center and radius in 2-D case. The algorithms for higher dimentional cases are similar.

The min-max formulation in equation (4.6) is neither differentiable nor convex, but using the soft-max approximation in equation (4.1) in Section 4.2.1, equation (4.6) can be formulated as the following optimization problem:

$$\text{minimize}_{x,y} \quad \frac{1}{\beta} \log \left[\sum_{i=1}^{N} e^{\beta\left\{(a_i-x)^2+(b_i-y)^2\right\}}\right]. \tag{4.8}$$

Note that equation (4.6) and (4.8) will be the same when $\beta \to \infty$. Equation (4.8) can be further simplified since $\log(\cdot)$ is a monotonic increasing function and β is a constant, we have

$$\text{minimize}_{x,y} \quad \sum_{i=1}^{N} e^{\beta\left\{(a_i-x)^2+(b_i-y)^2\right\}}. \tag{4.9}$$

The objective function in (4.9) is differentiable and convex. Moreover, by using the soft-max approximation, the objective function is in the form of sum of local differentiable functions. Note that the proof of convexity of (4.9) is obtained by first

calculating the Hessian of the objective function in (4.9). Then we can show that the Hessian matrix is positive definite when $\beta > 0$. The details of the proof is given in Appendix.

In a centralized network where the fusion center has all the location information of nodes. Traditional convex optimization algorithms, for example gradient descent method can be used for center estimation. Let (x^*, y^*) be the solution of equation (4.9), which is the estimated center. Then, the radius of the network is obtained by calculating the maximum distance between the estimated center (x^*, y^*) and nodes, we have

$$\hat{r} = \max_i \sqrt{(a_i - x^*)^2 + (b_i - y^*)^2}. \tag{4.10}$$

Therefore, by knowing the center and radius, the network area is obtained.

In wireless sensor networks where nodes having limited power and storage, it is always better to adopt distributed algorithms [83]. As shown above, by applying the soft-max approximation, the center estimation problem is formulated as a sum minimization problem as in equation (4.9). Therefore, distributed iterative algorithms can be used to estimate the center and radius. In the following of this section, the distributed center estimation is firstly introduced in Section 4.3.2. Then in Section 4.3.3, the distributed radius estimation algorithm is described.

4.3.2 Distributed Center Estimation

In this section, two different distributed center estimation algorithms for solving equation (4.9) are described. The stochastic gradient descent method is introduced in Section 4.3.2.1, where there is only one active node at each iteration time and communication between nodes is pairwise. In Section 4.3.2.2, diffusion adaptation method is used for distributed optimization and all nodes are exchanging information with their neighbors at each iteration time.

4.3.2.1 Center Estimation Using Stochastic Gradient Descent

Recall that by using the soft-max approximation, the network center estimation problem is formulated as a convex optimization problem given in equation (4.9). Since the objective function in equation (4.9) is a sum of differentiable functions, stochastic gradient method mentioned in Section 4.2.2 can be used to solve the convex optimization problem in a fully distributed way. At each node, the true gradient $\nabla J^{\text{global}}(x^{(t)}, y^{(t)})$ is approximated by the local gradient $\nabla J_i(x_i^{(t)}, y_i^{(t)})$, where

$$J_i(x_i^{(t)}, y_i^{(t)}) = e^{\beta\left\{(a_i - x_i^{(t)})^2 + (b_i - y_i^{(t)})^2\right\}}. \tag{4.11}$$

Algorithm 1 presents the updating steps at nodes. Firstly, a leader is selected as the starting node with starting value $(x^{(0)}, y^{(0)})$. The leader updates its estimate using stochastic gradient descent method and randomly choose one of its neighbors and pass the estimate to the chosen node. The node that receives the data becomes an active node and the original leader turns inactive. Then the active nodes repeat doing the update: i) update the estimate of center $(x^{(t)}, y^{(t)})$ by using stochastic gradient descent and randomly choose one neighbor node to pass the estimate; and ii) after passing the data, the original source node return inactive and the node gets the data becomes active. Note that at each iteration time, there is only one active node in the network doing the update, and all the inactive nodes stay idle. Finally, when t is large, all nodes reach consensus on the estimated center.

Note that Algorithm 1 performs like a sequential stochastic gradient descent, and it is fully distributed in the sense that the update of $x_i^{(t+1)}$ and $y_i^{(t+1)}$ at node i only depends on its own location information (a_i, b_i) and received data $(x_i^{(t)}, y_i^{(t)})$. Also note that max consensus can be used for distributed leader selection (choose the starting node) at the beginning of Algorithm 1 [49].

In the stochastic gradient based center estimation algorithm, only one node is

Algorithm 1 Stochastic gradient descent for center calculation

select a leader node (active node), with starting values $(x^{(0)}, y^{(0)})$.

for active node i:

repeat

$x_i^{(t+1)} = x_i^{(t)} - \eta \frac{\partial}{\partial x_i^{(t)}} J_i(x_i^{(t)}, y_i^{(t)})$.

$y_i^{(t+1)} = y_i^{(t)} - \eta \frac{\partial}{\partial y_i^{(t)}} J_i(x_i^{(t)}, y_i^{(t)})$.

select a neighbor $j \in \mathcal{N}_i$ to pass data:

$x_j^{(t+1)} = x_i^{(t+1)}$, $y_j^{(t+1)} = y_i^{(t+1)}$.

node $i \rightarrow$ inactive, node $j \rightarrow$ active

until stopping criterion is satisfied.

active at each iteration and nodes are communicating with each other pairwisely. Therefore, the convergence speed is slow. In the following section. a faster optimization method based on diffusion adaptation strategy is introduced.

4.3.2.2 Center Estimation Using Diffusion Adaptation

Since the global cost function in equation (4.9) is in the form of summation of individual real-valued local functions, and the local functions are differentiable and convex. The diffusion adaptation strategies like in [81] can be used to achieve consensus on network center estimate.

Let $\omega_i^{(t)} = \left[x_i^{(t)} \; y_i^{(t)}\right]^{\mathrm{T}}$, and $\psi_i^{(t)} \in \mathbb{R}^2$ be an intermediate value vector. Consider the following iterative algorithm:

$$\psi_i^{(t+1)} = \omega_i^{(t)} - \mu \sum_{j \in \mathbb{N}_i} c_{j,i} \nabla_\omega J_j(\omega_i^{(t)}), \tag{4.12}$$

$$\omega_i^{(t+1)} = \sum_{j \in \mathbb{N}_i} a_{j,i} \psi_i^{(t+1)}, \tag{4.13}$$

where $\mu > 0$ is a small constant descent step size parameter. $c_{j,i}$ and $a_{j,i}$ are non-

negative coefficients that satisfy

$$\sum_{i=1}^{N} c_{j,i} = 1, c_{j,i} = 0 \text{ if } j \notin \mathbb{N}_i, j = 1, 2, \cdots, N, \tag{4.14}$$

$$\sum_{i=1}^{N} a_{j,i} = 1, a_{j,i} = 0 \text{ if } j \notin \mathbb{N}_i, j = 1, 2, \cdots, N. \tag{4.15}$$

In each iteration time, the algorithm involves two steps. In the first step in equation (4.12), node i update the intermediate vector $\psi_i^{(t+1)}$ based on its own estimate $\omega_i^{(t)}$ and the gradient vector information from its neighbors. In the second iteration step in equation (4.13), nodes exchange information with their neighbors and calculate the network center, $\omega_i^{(t+1)}$ based on the received intermediate results. Note that the first step in equation (4.12) does not require information exchange in every iteration since $J_j(\omega_i^{(t)})$ only depends on the location information of node j, (a_j, b_j). Therefore, node i can save the location information of all of its neighbors at the first iteration time and use it during update steps in equation (4.12).

The diffusion algorithm in equation (4.12) and (4.13) is usually faster than the stochastic gradient descent approach in Section 4.3.2.1 since all nodes are active and exchanging information with their neighbors at each iteration time.

Note that since the objective function is of a summation form, the famous alternating direction method of multipliers (ADMM) can also be used [84]. However, traditional ADMM requires global average calculation in each ADMM iteration. Therefore, it is usually slower than the diffusion adaptation method in a distributed wireless sensor network.

4.3.3 Distributed Radius Estimation

When all nodes reach consensus on the estimated center (x^*, y^*), max consensus can be used for radius calculation. Each node first computes its distances to the

estimated center,

$$l_i = \sqrt{(a_i - x^*)^2 + (b_i - y^*)^2}. \tag{4.16}$$

Then by setting l_i as the initial states at nodes, the radius, which is the maximum distance $\hat{r} = \max_i\{l_i\}$, can be calculated using distributed max consensus as in Section 4.2.3. As shown in equation (4.5), each node exchange information with its neighbors and always keeps the maximum received data. In finite iteration time, nodes converge to the maximum of the initial measurements.

Therefore, by knowing the center and radius, all nodes in the network obtain an estimate of the network area.

4.4 Discussion

The performance of the proposed algorithm is affected by the design parameters. The accuracy of the center estimation result is affected by β and the nodes locations (a_i, b_i). The convergence speed of the distributed center estimation algorithm depends on β, η, $(x^{(0)}, y^{(0)})$, $c_{j,i}$ and $a_{j,i}$.

4.4.1 Steady State Error for Center Estimation

By using the proposed algorithms in Section 4.3.2, when consensus is reached, nodes in the network converge to an estimate of the location of the network center. Regarding the soft-max approximation parameter β, we see from equation (4.2) that larger β value results in more accurate max approximation from (4.6) to (4.9). Therefore, more accurate network center estimates can be obtained at nodes when β is chosen to be large. However, large β value also make the value of the objective function in equation (4.9) very large since exponential function is used. This may cause problems such as slow convergence rate and large transmit power.

The accuracy of the center estimation result also depends on the locations of nodes,

(a_i, b_i). In the following of this subsection, the error between the network center and the steady state estimation result using the proposed algorithm is discussed. If we consider the simple $1-D$ case, with nodes locations at a_i. The center of all nodes depends on the maximum and minimum nodes locations, and is at location $(a_{\max} + a_{\min})/2$. By using the soft-max approximation formulation as in Section 4.3, the steady state center location is calculated as

$$\text{argmin}_x \sum_{i=1}^{N} e^{\beta(a_i - x)^2}. \tag{4.17}$$

The optimal location x^* can be found using derivative of equation (4.17), and x^* satisfies that

$$\sum_{i=1}^{N} (x^* - a_i) e^{\beta(x^* - a_i)^2} = 0. \tag{4.18}$$

We have the following observations based on equation (4.18): i) The estimated center x^* is related to the initial nodes locations a_i; and ii) When the nodes locations are symmetric around the center, the solution of equation (4.18) will be the same as the true network center, $x^* = (a_{\max} + a_{\min})/2$. This is because when locations are symmetric, we can assume without loss of generality that $x_O - a_i = a_{N-i+1} - x_O$, where x_O is the network center. Assume N is even, we have

$$\sum_{i=1}^{N} (x_O - a_i) e^{\beta(x_O - a_i)^2} \tag{4.19}$$

$$= \sum_{i=1}^{N/2} (x_O - a_i) e^{\beta(x_O - a_i)^2} + \sum_{i=\frac{N}{2}+1}^{N} (x_O - a_i) e^{\beta(x_O - a_i)^2} \tag{4.20}$$

$$= \sum_{i=1}^{N/2} (x_O - a_i) e^{\beta(x_O - a_i)^2} - \sum_{i=1}^{N/2} (x_O - a_i) e^{\beta(x_O - a_i)^2} = 0. \tag{4.21}$$

As shown in equation (4.19)-(4.21), x_O satisfies equation (4.18). Therefore, symmetric nodes locations leads to accurate steady state center estimation, regardless of the

value of β. Note that an example of symmetric nodes locations can be uniformly distributed nodes locations.

Similar results can be obtained in 2-D network. The steady state center location estimate, $O(x^*, y^*)$ satisfies

$$\sum_{i=1}^{N}(x^* - a_i)e^{\beta\left[(x^*-a_i)^2+(y^*-b_i)^2\right]} = 0 \tag{4.22}$$

$$\sum_{i=1}^{N}(y^* - b_i)e^{\beta\left[(x^*-a_i)^2+(y^*-b_i)^2\right]} = 0 \tag{4.23}$$

When the locations (a_i, b_i) are symmetric around the center, the estimated center will be same as the true network center.

We have the following conclusions based on the above analysis: i) More accurate center estimation when β is large; and ii) When β is small, accurate center estimation can also be obtained if the sensor locations are symmetric. The center estimation accuracy depends on the locations of nodes, (a_i, b_i).

4.4.2 Convergence Speed for Center and Radius Estimation

The convergence speed of the proposed center estimation algorithm is affected by the design parameters and the initial starting value at nodes. The initial values $(x^{(0)}, y^{(0)})$ or $\omega_i^{(0)}$ affect the convergence speed of both Algorithm 1 in Section 4.3.2.1 and diffusion adaptation strategy in Section 4.3.2.2. In wireless sensor networks, average consensus such as those in [32, 34, 35] can be used to choose the staring value. Nodes first run average consensus as in [34, 35, 41] on their x and y coordinate values and the average consensus results can be used as the initial starting point of the proposed network center estimation algorithm. This is because in many network structures such as random graph or uniformly distributed network, the average location is close to the center.

The convergence speed is also affected by the design parameters. Algorithm 1 uses stochastic gradient descent and the step size η affect the convergence speed. In the diffusion adaptation based center estimation algorithm in Section 4.3.2.2, the convergence speed is affected by μ, $c_{j,i}$ and $a_{j,i}$. Convergence of stochastic gradient descent and diffusion adaptation have been studied extensively in the literature. The asymptotic convergence speed of stochastic gradient descent and how to choose the step size η for faster convergence can be found in [85, Chapter 18] and [86]. Performance analysis of diffusion adaptation strategy is given in [81] and [82].

Radius estimation algorithm in Section 4.3.3 always converge in finite iterations since max consensus with the max operator is used. The convergence speed of max consensus depends on the minimum number of edges needed to connect any two nodes in the network graph. Details of the convergence speed for max consensus can be found in [46].

4.5 Simulations

In this section, simulation results for the proposed algorithm are presented. A 2-D connected graph with $N = 6$ nodes is generated as shown in Figure 4.2. The locations of the nodes are shown in the figure. The center of the network is at $O(0, 0)$ and the radius $r = 1$.

In Figure 4.3, 4.4 and 4.5, stochastic gradient descent based Algorithm 1 mentioned in Section 4.3.2 is performed for distributed center estimation. We set node 1 (location at $(1, 0)$) to be the starting node with starting values $(0.3, 0.8)$. We set the stochastic gradient descent step size $\eta = 10^{-4}$ and the soft-max parameter $\beta = 1$. In the figure, the estimated coordinates of the center O at nodes are plotted. The estimate of x coordinate of O at all nodes is given in Figure 4.3 and the y coordinate estimate is shown in Figure 4.4. From the results, we can see that the

estimate at nodes converge towards the center of the network when t is large. In Figure 4.5, the error between the center O and the estimated center at node 1, denoted as $\sqrt{(x_1^{(t)} - x_O)^2 + (y_1^{(t)} - y_O)^2}$ is plotted, where $(x_O, y_O) = (0, 0)$ is the center. We can see from the figure that the error decreases as t increases and the estimate converge towards the center.

In Figure 4.6, 4.7 and 4.8, diffusion adaptation in in Section 4.3.2.2 for center estimation. We set the initial states at nodes $\omega_i^{(0)} = [x_i^{(0)} \ y_i^{(0)}]^{\mathrm{T}}$ to be uniformly distributed, $\mathcal{U}(-0.5, 0.5)$. The descent step size $\mu = 10^{-4}$ and the soft-max parameter $\beta = 1$. The coefficients $c_{j,i}$ and $a_{j,i}$ are setted based on degree of nodes:

$$c_{j,i} = a_{j,i} = \begin{cases} \frac{1}{d_i+1}, & \text{if } j \in \mathbb{N}_i \\ 0, & \text{otherwise.} \end{cases} \tag{4.24}$$

In Figure 4.6 and 4.6, the x and y coordinate estimates at different iteration time t are plotted. We can see from the figure that the estimates converge towards the center. In Figure 4.8, the average estimation error versus iteration time t is plotted. By comparing Figure 4.8 with Figure 4.5, we can see that the method of diffusion adaptation converges faster than the stochastic gradient descent.

In Figure 4.9, simulations for Phase II of the proposed algorithm (max consensus for radius estimation) described in Section 4.3.3 is performed. We assume that stochastic gradient descent method is used and the iterative algorithm stops at $t^* = 5000$. Distances from nodes to the estimated centers are calculated at nodes and set as initial values at nodes for max consensus. Figure 4.9 shows the max consensus radius estimation process. We can see from the figure that consensus is reached in 3 iterations and the estimated radius $\hat{r} = 1.063$.

Finally based on the estimated center and radius, estimate of the network area is obtained at nodes. The estimated network area of node 1 at time $t^* = 5000$ is shown

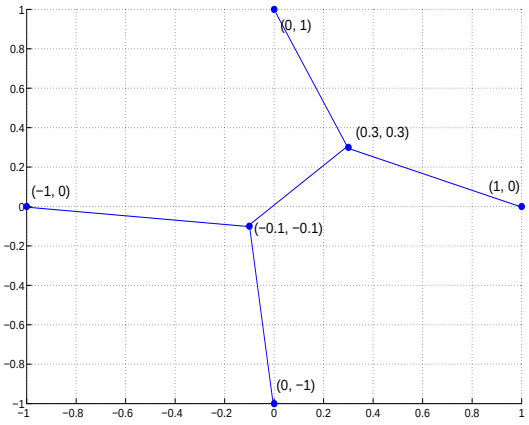

Figure 4.2: Graph Representation of the Sensor Network, $N = 6$.

in Figure 4.10.

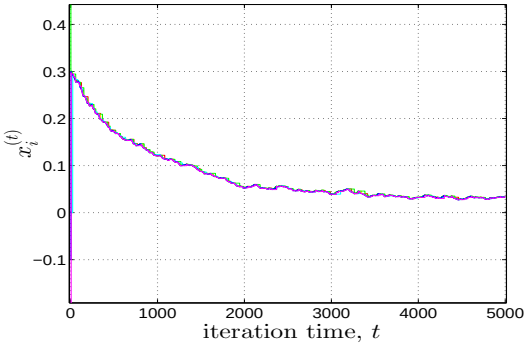

Figure 4.3: Estimate of the x Coordinate Value of the Center, $x_i^{(t)}$ Versus Iteration t Using Algorithm 1, $\eta = 10^{-4}$ and Starting Point $x^{(0)} = 0.3$.

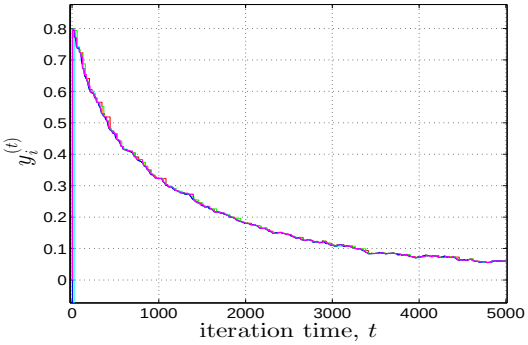

Figure 4.4: Estimate of the y Coordinate Value of the Center, $y_i^{(t)}$ Versus Iteration t Using Algorithm 1, $\eta = 10^{-4}$ and Starting Point $y^{(0)} = 0.8$.

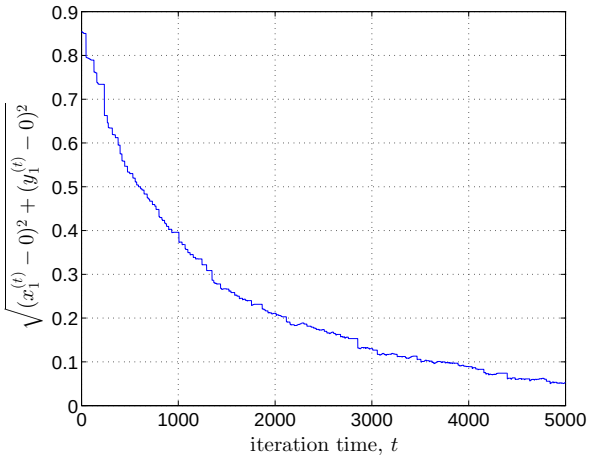

Figure 4.5: Error Versus t at Node 1 with the Algorithm 1, Where $O(x_O, y_O)$ is the True Center and $x_O = 0, y_O = 0$.

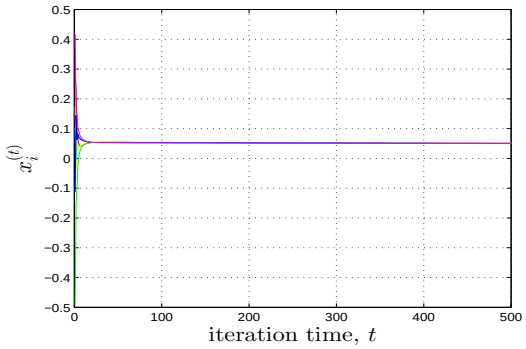

Figure 4.6: Estimate of the x Coordinate Value of the Center, $x_i^{(t)}$ Versus Iteration t Using Diffusion Adaptation, $\eta = 10^{-4}$ and Starting Point to be Uniformly Distributed $\mathcal{U}(-0.5, 0.5)$.

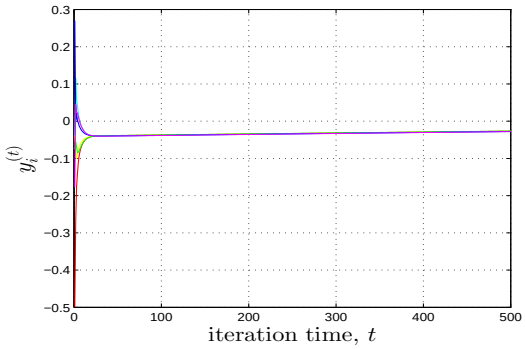

Figure 4.7: Estimate of the y Coordinate Value of the Center, $y_i^{(t)}$ Versus Iteration t Using Diffusion Adaptation, $\eta = 10^{-4}$ and Starting Point to be Uniformly Distributed $\mathcal{U}(-0.5, 0.5)$.

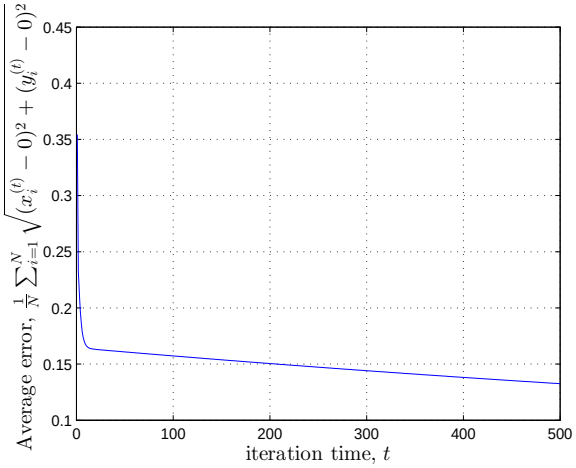

Figure 4.8: Average Error Versus t Using Diffusion Adaptation, Where $O(x_O, y_O)$ is the True Center and $x_O = 0, y_O = 0$.

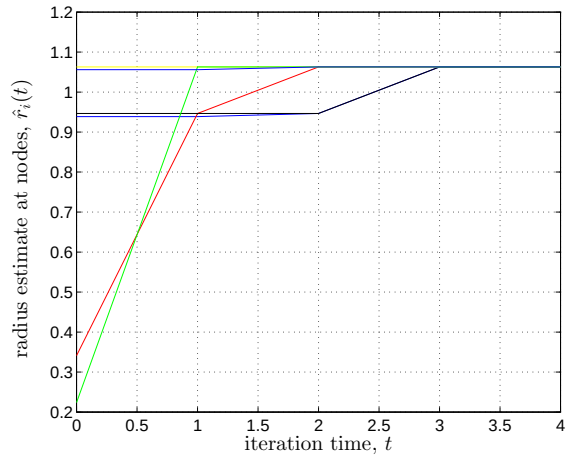

Figure 4.9: Radius Estimate Versus t Using Max Consensus in Section 4.3.3. The Initial Value at Node i is Set to be the Distance Between the Estimated Center and Its Own Location.

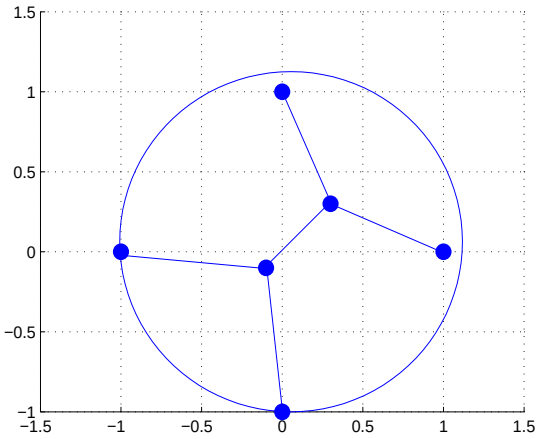

Figure 4.10: Estimated Network Area at Node 1 at $t = 5000$.

Chapter 5

CONSENSUS BASED DISTRIBUTED ESTIMATION ALGORITHMS

In this chapter, we present two average consensus based distributed estimation algorithms in wireless sensor networks. A distributed algorithm for estimating the degree distribution and degree matrix of network is described in Section 5.1 and running consensus algorithm for estimating the dynamics of a desired estimator is introduced in Section 5.2. Note that part of the degree distribution algorithm is presented in our published work in [87]. The running consensus algorithm for tracking the dynamics of an estimator may be included in our future publications.

5.1 Distributed Estimation of the Degree Distribution in Wireless Sensor Networks

Consider a connected wireless sensor network with no fusion center. Each node in the network generates a real-valued initial state vector. We assume that the each sensor always keeps a single state vector and the sensors update the state vectors based on local received measurements from their neighbors. It is desired that the nodes reach consensus on the degree distribution and degree matrix of the network.

The proposed algorithm is based on the fact that degrees are discrete values. The idea of estimation of empirical mass functions with average consensus algorithm is used. We also show that if the number of nodes in the network N is given, the degree matrix of the network can be obtained at nodes.

In the following, details of the proposed algorithm is provided. The proposed degree distribution estimation algorithm is introduced in Section 5.1.1. In Section 5.1.2, we assume that the value of N is available and we show that the degree matrix of the network can be calculated from the degree distribution. The analysis of the algorithm

is given in Section 5.1.3.

5.1.1 Estimation of Degree Distribution

The algorithm can be described as three phases: initial measurement vectors are generated at nodes in Phase I, and average consensus algorithm is performed in phase II to let nodes reach consensus on the same state vector. Finally the degree distribution is calculated at phase III by post processing the convergence result. In the following of this subsection, details of three phases of the algorithm are introduced.

5.1.1.1 Phase I - Generate Initial Values

In Phase I of the algorithm, initial measurement vectors are generated at nodes. Each node in the network generates an initial measurement vector with length K, and $K > d_{\max}$, where $d_{\max}$ is the maximum degree. Assume that node i only knows its own degree d_i, the initial measurement vector at node i, denoted as $\mathbf{x}_i(0) \in \mathcal{R}^K$ can be expressed as,

$$\mathbf{x}_i^{(k)}(0) = \begin{cases} 1, & \text{if } d_i = k \\ 0, & \text{otherwise,} \end{cases} \tag{5.1}$$

where $\mathbf{x}_i^{(k)}(0)$ is the kth element of $\mathbf{x}_i(0)$, and $k = 1, 2, \cdots, K$. Note that max consensus as mentioned in [46, 57] can be used to estimate $d_{\max}$. The max estimate will drift larger in the presence of noise [57]. Therefore K can be set as the max consensus result since $K > d_{\max}$ is required.

5.1.1.2 Phase II - Average Consensus

In Phase II of the degree distribution estimation algorithm, average consensus algorithm is used for each element in the state vector. Each node i set the initial state value $x_i(0)$ as in equation (5.1). Nodes in the network run the iterative algorithm as

expressed in equation (2.4), and the kth element in the state of node i at time $t+1$ can be expressed as,

$$\mathbf{x}_i^{(k)}(t+1) = [1-\alpha(t)d_i]\,\mathbf{x}_i^{(k)}(t) + \alpha(t)\sum_{j\in\mathbb{N}_i}\left[\mathbf{x}_j^{(k)}(t) + v_{ij}^{(k)}(t)\right], \tag{5.2}$$

where $v_{ij}^{(k)}(t)$ is the noise associated with the reception of $\mathbf{x}_j^{(k)}(t)$ at node i.

When t is large, the state vectors for all nodes are converging to a noisy version of the average of the initial measurement vectors. Assume the iteration stops at time t^*, we have,

$$\mathrm{E}\left[\mathbf{x}_i^{(k)}(t^*)\right] = \frac{1}{N}\sum_{j=1}^{N}\mathbf{x}_j^{(k)}(0) = \left(\frac{n_k}{N}\right), \tag{5.3}$$

where n_k is the number of nodes with degree k.

Equation (5.3) shows that by running the distributed averaging, the expected value of the kth element in the state vector converges to the degree distribution $p^{(k)}$. The mean square error of the algorithm is small when the noise variance σ_n^2 is small, from Theorem 1.

Assuming consensus is perfectly reached and there is no communication noise, then the convergence result will be an accurate estimate of the degree distribution.

$$\hat{p}_i^{(k)}(t^*) = \frac{n_k}{N}. \tag{5.4}$$

However, consensus in wireless sensor networks always suffers from lack of convergence and communication noise. As a result, the following post processing is needed.

5.1.1.3 Phase III - Post Processing

In this section, a threshold based on the network size is used to decide whether the convergence result is noisy averaging of the initial measurements or purely noise. For node i, the estimate of the degree distribution at time t^* can be expressed as,

$$\hat{p}_i^{(k)}(t^*) = \begin{cases} \mathbf{x}_i^{(k)}(t^*)\,, & \text{if } \mathbf{x}_i^{(k)}(t^*) \geq \frac{1}{2\hat{N}} \\ 0\,, & \text{otherwise,} \end{cases} \tag{5.5}$$

where $\hat{N}$ is an estimate of number of nodes in the network. Note that the number of nodes in the network can be estimated in a distributed way [73] and $\hat{N}$ can also be used as K since $d_{\max} \leq N$. Also note that the thresholding post processing step is used to set the purely noisy convergence result to 0. For example, by applying the thresholding, the negative consensus results caused by communication noise are set to 0. Therefore the estimate of degree distribution will be non-negative. Here, the threshold is set to be $\frac{1}{2\hat{N}}$ as shown in equation (5.5).

More ways to post process the convergence result based on graph degree structure and ways to choose state vector size, K are introduced in Section 5.1.4.

5.1.2 Estimation of Degree Matrix

In the previous subsection, it is shown that the degree distribution of the network can be obtained in a distributed way. From the degree distribution, the degree matrix can be calculated if the number of nodes in the network N is known. The idea is that with $\hat{p}_i^{(k)}$ and N, the estimate of number of nodes with degree k can be calculated as,

$$\hat{n}_k = \left\lfloor N\hat{p}_i^{(k)} \right\rceil,$$

where $\lfloor x \rceil$ rounds x to the nearest integer. Therefore, we know there are $\hat{n}_k$ nodes with degree k, $k = 1, 2, \cdots K$ and an estimate of the degree matrix can be obtained. Note that since nodes are not labeled, the obtained degree matrix is actually a permutation of the labeled degree matrix, which can be used to infer $d_{\min}$, $d_{\max}$ and the trace of the Laplacian matrix.

5.1.3 Performance Analysis

In this section, different sources of error are discussed: i) The error caused by lack of convergence in finite time is analyzed in Section 5.1.3.1, and ii) The effect of

communication noise is considered in Section 5.1.3.2.

5.1.3.1 Transient of Bias

The convergence rate analysis for each element in the state vector will be the same as in [34] since average consensus is used. Let $\mathbf{x}^{(k)}(t) = \left[x_1^{(k)}(t) \ \cdots \ x_N^{(k)}(t)\right]^T$ contains all the kth element in the state vector for all nodes at time t, then the convergence rate for $\mathbf{x}^{(k)}(t)$ can be expressed as,

$$||\mathrm{E}[\mathbf{x}^{(k)}(t)] - \frac{n_k}{N}\mathbf{1}||_2 \leq \left(\prod_{0<\tau\leq t}(1-\alpha(\tau)\lambda_2(\mathbf{L}))\right)||\mathbf{x}^{(k)}(0) - \frac{n_k}{N}\mathbf{1}||_2. \tag{5.6}$$

Equation (5.6) shows that the convergence rate depends on $\lambda_2(\mathbf{L})$ and $\alpha(t)$.

5.1.3.2 Steady State Error Analysis

In this section, we assume that the convergence is reached and state vectors of nodes converge to a noisy version of the initial state vectors as mentioned in Theorem 3. It is shown in Theorem 3 that the MSE of the convergence result is bounded, and σ_n^2 and $\alpha(t)$ affect the performance. In the following of this section, the distribution of the convergence result is calculated, and how the algorithm parameters and noise affect the convergence result is studied.

After convergence is reached as in Section 5.1.1.2, the kth element in the state vector of node i can be expressed as,

$$\mathbf{x}_i^{(k)}(t) = \frac{n_k}{N} + v', \tag{5.7}$$

where v' is a random variable caused by accumulated communication noise, $v' \sim \mathcal{N}\left(0, \left(\frac{\sum_{i=1}^N d_i}{N^2}\right)\sigma_n^2 \sum_{t=0}^{\infty}\alpha^2(t)\right)$ from Theorem 3. Therefore, the convergence result is also Gaussian distributed, we have,

$$\mathbf{x}_i^{(k)}(t) \sim \mathcal{N}\left(\frac{n_k}{N}, \left(\frac{\sum_{i=1}^N d_i}{N^2}\right)\left(\frac{\sigma_n^2}{1}\right)\sum_{t=0}^{\infty}\alpha^2(t)\right). \tag{5.8}$$

From equation (5.8) we have the following conclusions: i) The convergence result is an unbiased estimator of the degree distribution; ii) The variance of the estimator is related to $\alpha(t)$ and $\frac{1}{\sigma_n^2}$ (can be viewed as SNR since initial $\mathbf{x}_1^{(k)}$ is chosen as equation (5.1)); and iii) The variance will be small when SNR is large and the steady state performance will not be affect by SNR if there is no communication between nodes. Note that if amplify and forward is used for transmission, the value of $\mathbf{x}_1^{(k)}$ in equation (5.1) can be chosen to be large to increase the SNR, therefore make the variance of the estimate in equation (5.8) smaller.

5.1.4 Discussions

The degree distribution have some special properties, such as the degrees are integers and the degree distribution usually follows power-law in real-world networks [88]. In this section, we show that these properties can be used to obtained a more accurate estimate of the degree distribution or reduce the required storage at nodes.

5.1.4.1 Post Processing Based on Integer Degree Structure

In this subsection, a post processing step based on the fact that the degrees are integers is introduced. We assume that the number of nodes in the network, N is given. Assume the algorithm stops at time t^*, the additional post processing step after Phase III mentioned in Section 5.1.1.3 can be expressed as,

$$\hat{p}_i^{(k)}(t^*) = \frac{\left\lfloor N\hat{p}_i^{(k)}(t^*)\right\rceil}{N}, \tag{5.9}$$

where $\lfloor x\rceil$ rounds x to the nearest integer. We show in the simulations that by post processing as in equation (5.9), a better estimate can be obtained when at high SNR.

5.1.4.2 Reducing the Size of the State Vector

In the proposed algorithm, the size of state vector is chosen based on the maximum degree, $K > d_{\max}$. If we have some prior knowledge of the degree distribution, the size of the state vector can be reduced, therefore reduce the required storage at nodes. A simple way to reduce the size of the state vector can be let each bin cover same degree ranges, for example 1 to 2, 3 to 4, 5 to 6 and so on. The final estimate can be obtained by dividing the result of each element in the state vector by the width of the bin to normalize the measurement. In practice, some of the networks have power-law degree distribution, therefore one way to reduce the size of the state vector is to let bins cover an increasing range of degrees, for example 1, 2 to 3, 4 to 7, 8 to 15, and so on [88].

5.1.5 Simulations

In this section, simulation results for the proposed algorithm are presented. A connected graph with $N = 75$ nodes is generated. The true degree distribution is given in Figure 5.1, where the x-axis values are possible values of degree and y-axis is the probability $\Pr[X = k]$. In Figure 5.2 - 5.4, the proposed algorithm is performed. It is assumed that the algorithm stops at $t^* = 100$, and parameters $\alpha(t) = 0.1/t$, $K = 80$. In Figure 5.2, we assume perfect communication with no communication noise. From the results, we can see that an accurate estimate of the degree distribution can be obtained. Noisy communication is considered in Figure 5.3 and 5.4, and the noise variance are set to be different with $\sigma_n^2 = 0.1$ in Figure 5.3 and $\sigma_n^2 = 0.01$ in Figure 5.4. By comparing Figure 5.3 and Figure 5.4, we see that the estimate of the degree distribution has better performance when the SNR is larger.

In Figure 5.5, the error norm of the consensus result at node 1, $||\mathbf{x}_1(t) - \mathbf{p}||$ is

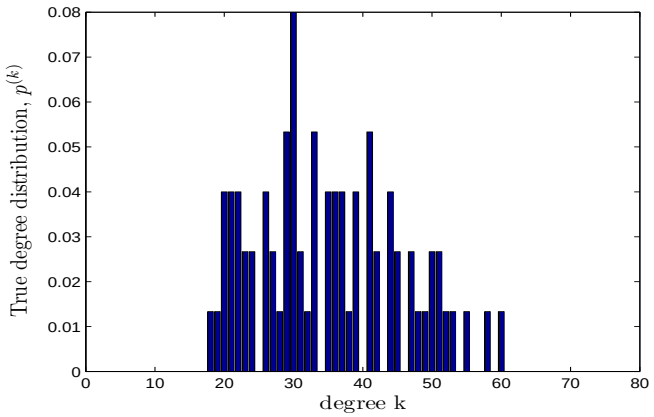

Figure 5.1: True Degree Distribution.

shown, where $\mathbf{p}$ is the true degree distribution and $\mathbf{x}_1(t) = \left[\mathbf{x}_1^{(1)}(t), \cdots, \mathbf{x}_1^{(K)}(t)\right]$ is the degree distribution estimate at node 1 at time t. Noise variance σ_n^2 are chosen to be different values as shown. From Figure 5.5 we have the following observations: i) The error decreases as t gets larger; ii) When there is no communication noise, the consensus result converges toward the desire true degree distribution; and iii) In the presence of communication noise, there will always be an error, and the error will be small when the SNR is large.

In Figure 5.6, the post processing based on equation (5.9) is applied with the assumption that the number of nodes $N = 75$ is given. When the SNR is large, we see that the circle line terminates after 6 iterations, which indicates that the error becomes exactly 0 after 6 iterations if post process the convergence result as in equation (5.9). However, when the SNR is small, the post processing step as in equation (5.9) does not improve the performance.

In Figure 5.7, simulation result for size reduced state vector mentioned in Section 5.1.4.2 is presented. We set the size of the state vector $K = 40$, and each element in the state vector covers two degrees and the final estimate is obtained by dividing the consensus result of each element by 2. Noise communications is consid-

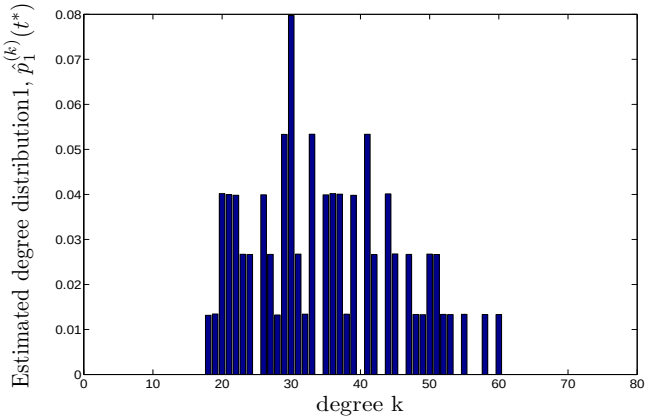

Figure 5.2: Estimate of Degree Distribution at Time $t^* = 100$ at Node 1 in the Absence of Noise, $\sigma_n^2 = 0$ and $\alpha(t) = 0.1/t$.

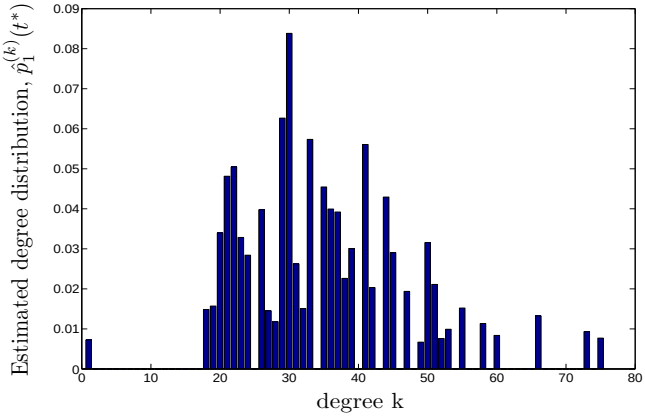

Figure 5.3: Estimate of Degree Distribution at Time $t^* = 100$ at Node 1 in the Presence of Noise, $\sigma_n^2 = 0.1$ and $\alpha(t) = 0.1/t$.

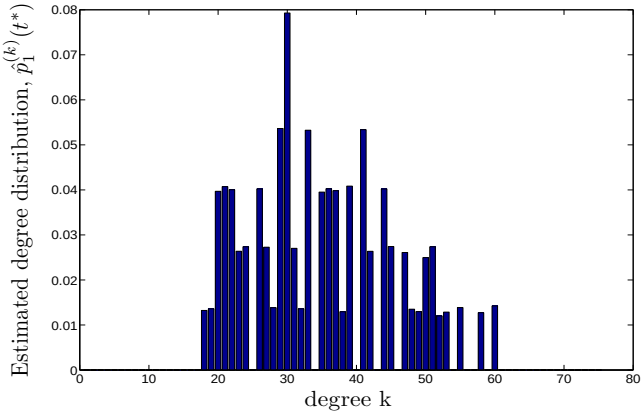

Figure 5.4: Estimate of Degree Distribution at Time $t^* = 100$ at Node 1 in the Presence of Noise, $\sigma_n^2 = 0.01$ and $\alpha(t) = 0.1/t$.

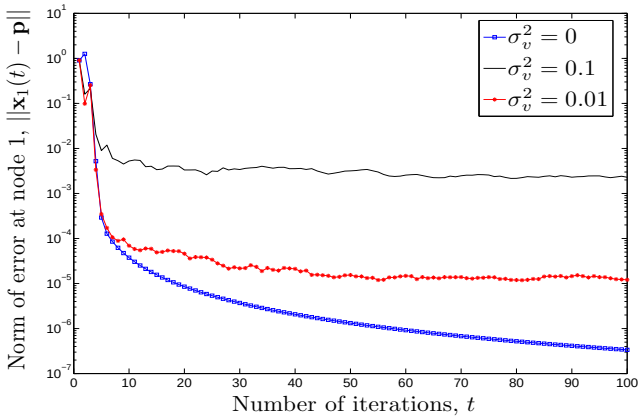

Figure 5.5: Error Versus t.

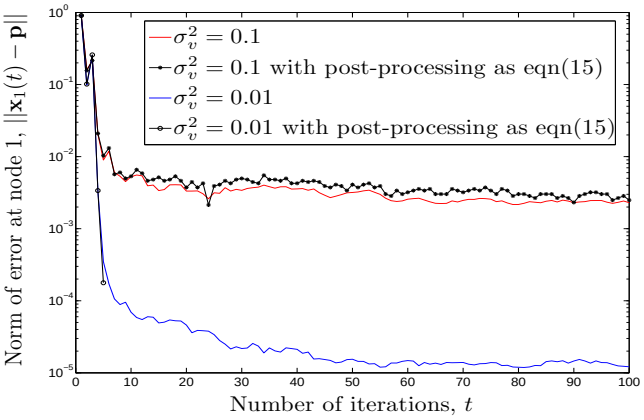

Figure 5.6: Simulation Results for Post Processing as in Equation (5.9): Error Versus t.

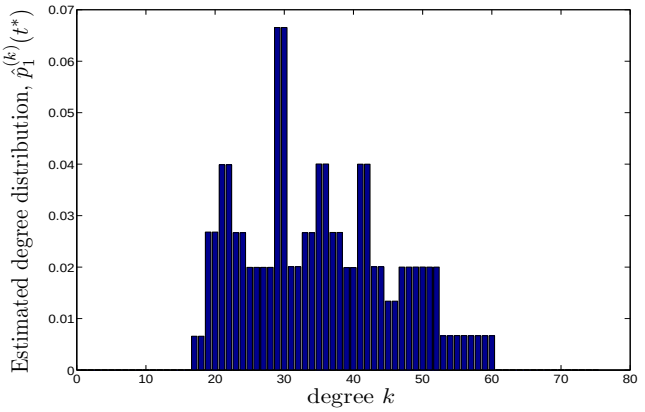

Figure 5.7: Degree Distribution Estimation at Node 1 (in the Presence of Noise and $K = 40$

.

ered with $\sigma_v^2 = 0.01$ and $\alpha(t) = 0.1/t$. By comparing Figure 5.7 with Figure 5.4 and Figure 5.1, we see that we can reduce the size of the state vectors at nodes and a reasonable estimate of the degree distribution can still be obtained.

5.2 Running Consensus Over Distributed Networks: Non-Stationary Data and Tracking Ability

In traditional consensus algorithms, it is usually assumed that sensing at nodes and communication between sensors are separate steps. Sensors first sense the environment, then the sensors run the consensus algorithms. However, sometimes in real applications, there is no meaningful way to decide when the sensing stage should be terminated to start consensus, for example the environment is changing rapidly and the temperature measurements sensed at nodes is changing during the consensus process. Therefore, a running consensus algorithm to track the dynamic of the estimator is proposed herein. We assume that the sensing and consensus stages are simultaneous and sensor nodes continuous collecting data while computing on-the-fly the desire estimator.

5.2.1 System Model

We use a undirected graph to model the distributed network as in Section 2.1.1. We assume that each sensor in the network is measuring some environmental parameters such as temperature and pressure independently. New measurements are available simultaneously at nodes and the desire parameter is non-stationary and changing over time.

5.2.2 Running Consensus with Non-Stationary Data

Assume that at each time t, sensors in the network collect noisy measurements from the environment. The measurement at node i at time t is

$$x_{i,t} = \theta_t + n_{i,t}, \tag{5.10}$$

where θ_t is the desired parameter and it is changing over time and $n_{i,t}$ is the sensing noise. Let $\mathbf{x}(t) = [x_1(t)\ x_2(t)\ \cdots\ x_N(t)]$ be the state vector at time t, where $x_i(t)$ is the state of node i at time t, and N is the total number nodes in the network. Let $\mathbf{x}_t = [x_{1,t} \cdots x_{N,t}]$ contains the measurements available at nodes at time t. To track the dynamic of θ_t, we use average consensus with a moving average method, the iterative updating rule can be expressed as

$$\mathbf{x}(t+1) = \frac{k}{k+1} W\mathbf{x}(t) + \frac{1}{k}\mathbf{x}_{t+1}, \tag{5.11}$$

where k is a design parameter that controls the sensitivity of the algorithm to the dynamics.

By running the iterative algorithm in equation (5.11), the average of state vector at time t, $\bar{x}(t+1) = \frac{1}{N}\left(x_1(t) + x_2(t) + \cdots + x_N(t)\right)$ behaves like

$$\bar{x}(t+1) = \frac{1}{k+1}\left[\bar{x}_{t+1} + \left(\frac{k}{k+1}\right)\bar{x}_t + \cdots + \left(\frac{k}{k+1}\right)^{t-1}\bar{x}_2\right] + \left(\frac{k}{k+1}\right)^{t}\bar{x}_1, \tag{5.12}$$

where $\bar{x}_t = \frac{1}{N}\left(x_{1,t} + x_{2,t} + \cdots + x_{N,t}\right)$ is the average of all the new sensed measurements at time t. Note that if we set $k = t$, the algorithm will be the same as the running consensus algorithm proposed in [43], where the states of nodes is converging to the average of all the initial measurements, which will be shown in the simulations. Also note that when N is large, we have $\theta_t \approx \bar{x}_t$ if the sensing noise has 0 mean and bounded variance. From equation (5.12), we see that the average of the states, $\bar{x}(t+1)$ is related to all the measurement across sensors and time. The more recent

measurements have a larger effect on $\bar{x}(t+1)$ and the sensitivity of the algorithm depends on the value of k: the algorithm will be more sensitive to the dynamic of the desired estimator for smaller k.

5.2.3 Simulations

In this section, simulation results for the proposed running consensus are presented. We also compare the proposed algorithm with the diffusion strategy in the literature [3]. The graph structure is fixed and is the same as the graph in Figure 2.2. In Figure 5.8 - Figure 5.12, we assume that $\theta_t = 0.05t$ and the noise is Gaussian distributed as, $n_{i,t} \sim \mathcal{N}(0, 100)$. In Figure 5.8 - Figure 5.10, the proposed algorithm is used. By comparing Figure 5.8 with Figure 5.9 with see that with smaller k value, the algorithm is more sensitive to the dynamics of the estimator and the estimate is more accurate but the convergence of the algorithm is worse, means that the states of nodes are more different than each other. In Figure 5.10, we set $k = t$ as in [43] and we see from the figure that the states of nodes are converging the global average of all the initial measurements. In Figure 5.11 - Figure 5.12, the diffusion algorithm in [3] is used. By comparing the proposed algorithm with the diffusion strategy, we see that both algorithms can track the dynamic of the desired estimator.

In Figure 5.13 and Figure 5.14, we assume that the desired estimator is changing over time like a sine wave, $\theta_t = 10\sin(0.01t)$. The proposed algorithm is used and by comparing the two figures, we have the same conclusions that there is trade-off between sensitivity to the dynamic of the desired estimator and the convergence of the states of nodes.

In Figure 5.15, we set $k = t$ and the states of nodes are converging to the global average of all the initial measurements 0 since $\theta_t = 10\sin(0.01t)$ and the global average is 0.

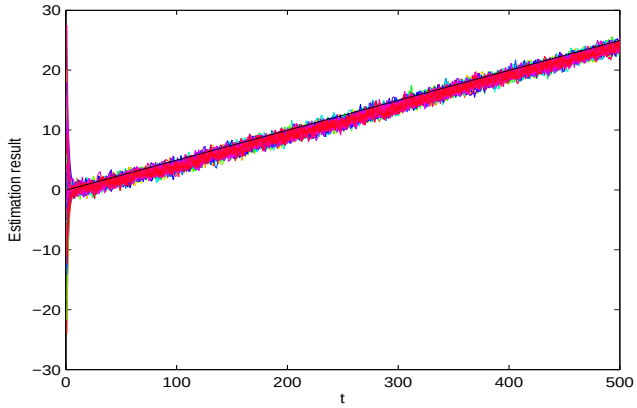

Figure 5.8: Entries of Estimation Result Versus Iteration Time t (Using Running Consensus with $k = 19$)

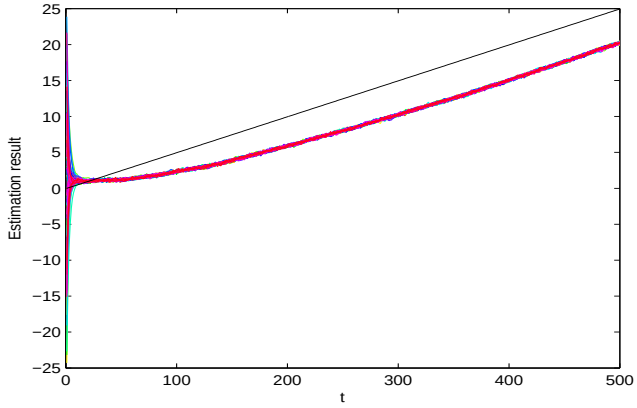

Figure 5.9: Entries of Estimation Result Versus Iteration Time t (Using Running Consensus with $k = 99$)

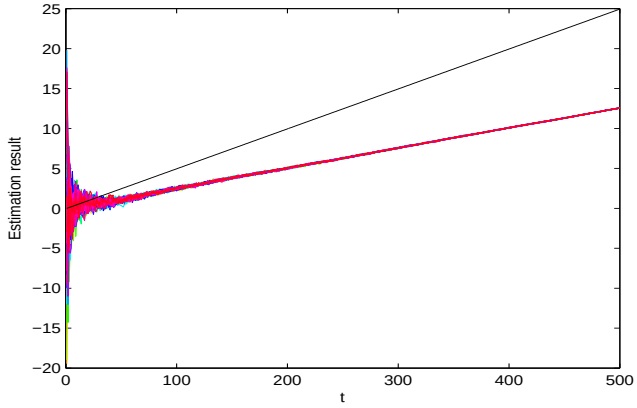

Figure 5.10: Entries of Estimation Result Versus Iteration Time t (Using Running Consensus with $k = t$)

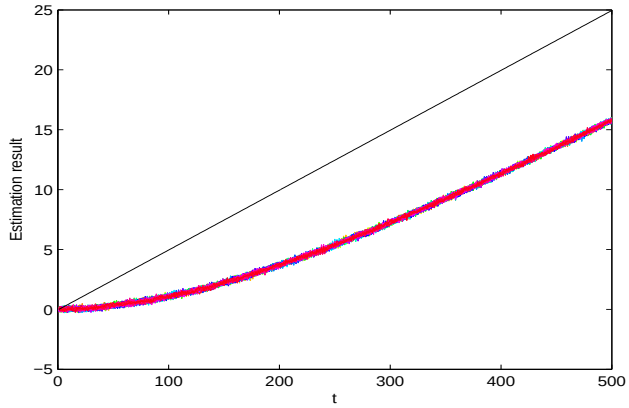

Figure 5.11: Entries of Estimation Result Versus Iteration Time t (Using Diffusion LMS in [3] with $\mu = 0.01$ and $u_{k,t} = 1$).

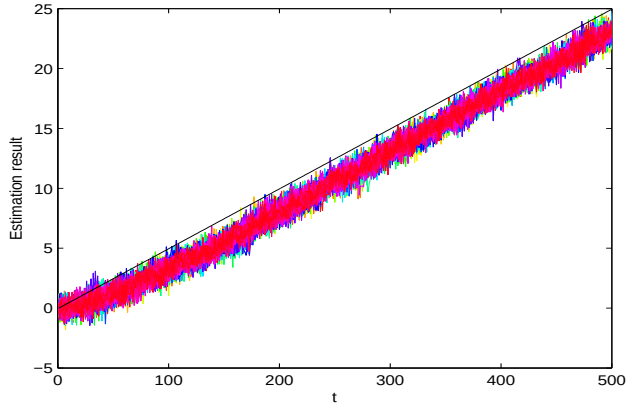

Figure 5.12: Entries of Estimation Result Versus Iteration Time t (Using Diffusion LMS in [3] with $\mu = 0.05$ and $u_{k,t} = 1$).

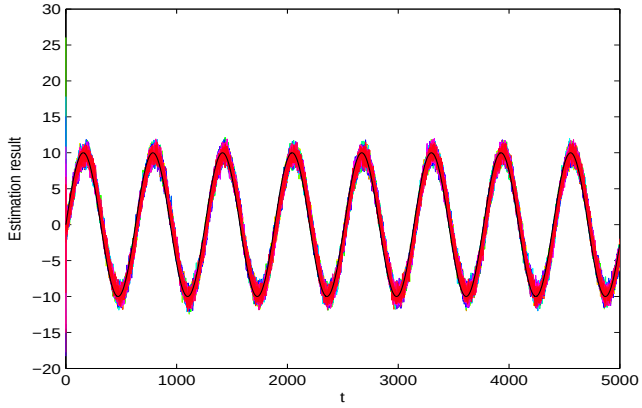

Figure 5.13: Entries of Estimation Result Versus Iteration Time t (Using Running Consensus with $k = 19$)

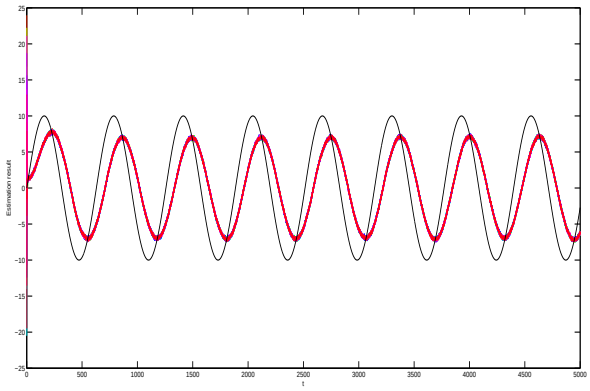

Figure 5.14: Entries of Estimation Result Versus Iteration Time t (Using Running Consensus with $k = 99$)

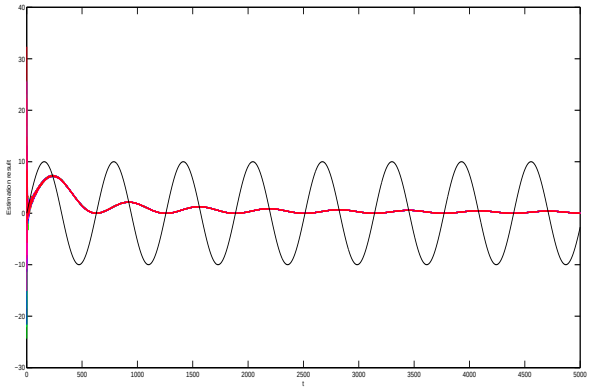

Figure 5.15: Entries of Estimation Result Versus Iteration Time t (Using Running Consensus with $k = t$)

Chapter 6

FUTURE WORK

The future work will mainly focus on function computation in wireless sensor networks and distributed network structure estimation. Function computation problem is discussed in Section 6.1, a brief literature review is first given, then several iterative updating rules that ensure convergence are discussed. In Section 6.2, possible future work on distributed network structure estimation are proposed

6.1 Distributed Function Computation in WSNs

In wireless sensor networks, it is usually desired that the nodes in the network compute some functions of the initial measurements at nodes such as the average of the initial measurements or the maximum of the initial measurements. This motive us to consider the problem: what kinds of iterative updating rules ensure convergence and what kinds of functions can be computed at nodes.

In [89], the problem of function computation and approximation in wireless sensor network is studied. It is state in [89] that average consensus algorithm can be used to approximate any continuous function of the initial measurements. The proposed work is based on the Kolmogorov theorem which states that any continuous function of $\mathbf{x} = [x_1, x_2, \cdots, x_N]$, $f(x_1, \cdots, x_N)$ can be approximated as a superposition form:

$$f(x_1, \cdots, x_N) = \sum_{j=1}^{2N+1} \psi_j \left(\sum_{i=1}^{N} \phi_{ij}(x_1) \right), \tag{6.1}$$

where N is the total number of elements in $\mathbf{x}$, and $\psi_j(\cdot)$ and $\phi_{ij}(\cdot)$ are nonconstant, bounded, and monotonically-increasing continuous functions. The updating rule in equation (6.1) can be computed using average consensus. However, it is often hard

to choose $\psi_j(\cdot)$ and $\phi_{ij}(\cdot)$ properly since $\psi_j(\cdot)$ and $\phi_{ij}(\cdot)$ depends the desire function $f(x_1, \cdots, x_N)$ and all measurements at nodes $x_1, x_2, \cdots, x_N$ [89].

As mentioned in the previous sections, the max updating rule by using the max operator in equation (2.10) lead lead to convergence in the presence of communication noise. The iterative updating rule for nonlinear average consensus in equation (2.13) enables nodes converge to the average of the initial measurements in the presence of communication noise. We also consider an average consensus with fixed transmission power, the iterative updating rule at node 1 at time $t+1$ can be express as

$$x_i(t+1) = x_i(t) - \alpha(t) \sum_{j \in \mathcal{N}_i} \left\{ \mathrm{Im}\left(e^{j\omega x_i(t)}\right) - \mathrm{Im}\left(e^{j\omega x_j(t)} + n_{ij}(t)\right) \right\}, \tag{6.2}$$

where $\omega < \frac{\pi}{2 \max_{i,t} x_i(t)}$ is a design parameter and $\mathrm{Im}(x)$ takes the imaginary part of x. In each iteration in equation (6.2), node i receives noisy data from its neighbors $j \in \mathcal{N}_i$ and the transmitted signal from node j to node i is $e^{j\omega x_i(t)}$ and the transmitted power is always 1. By running the iterative algorithm in (6.2), states of nodes in the network converge to the sample mean of the initial measurements

All the above mentioned iterative algorithms are basically Markov processes. The updating rule can be written in a general form:

$$x_i(t+1) = g\left(x_i(t), \mathbf{x}^{(i)}(t)\right), \tag{6.3}$$

where the vector $\mathbf{x}^{(i)}(t) \in \mathbb{R}^{d_i}$ contains the measurements from all the neighbors of node i at time t. $g(\cdot)$ is the updating function, for example $g(\cdot)$ is the max operator in the traditional max consensus algorithms in [46]. In our future work, the theorem of convergence of Markov process and sample functions are going to be used to analyze what kinds of updating rule in (6.3) lead to convergence and what kinds of functions can be computed using consensus algorithms.

6.2 Distributed Network Structure Estimation

In Chapter 4, we consider the network region to be a circle or sphere. Therefore, estimating the network region is basically estimating the network center and radius. However, in some cases, using the smallest sphere to represent the network region may not be a good choice. In this section, we propose future work on network region estimation with convex hull or ellipsoidal network coverage area. In the following, the centralized formulations are presented and possible distributed algorithms are discussed.

One reasonable way to represent the network region can be the convex hull of the distributed sensors. Research on computing the convex hull of a finite set of points can be found in [90, 91]. In the centralized formulation, it is assumed that the unordered set of points is given. The smallest convex hull is a convex polygon and the vertices are some of the points in the set. In the literature, methods such as gift wrapping, Graham scan, Quick hull and Monotone chain are considered and the complexity is related to the sorting problem [92].

Fully distributed algorithm for estimating the convex hull of a distributed network has not been considered in the literature. If we consider the simple 2-D case. From the definition, we can decide a node is a vertices of a convex hull if there exist another node in the network, and all the other nodes are on one side of the unique line decided by the two distinct nodes(on one side or on the line). How to formulate the problem in a distributed manner is one of our future work.

Another way to represent the network region is to use the smallest covering ellipsoid. Note that sphere is a special case of ellipsoid. In the following, we discuss two ways to estimate the smallest ellipsoid. The first method is to estimate the four ellipsoid parameters: the center O, semi-major axis a, semi-minor axis b and rotation

angle θ. It is shown in [93] that if the location of the center is $O(x_O, y_O)$, all other parameters can be determined if all the nodes locations are given. The semi-major axis can be determined as:

$$a = \max_i \sqrt{(x_i - x_O)^2 + (y_i - y_O)}. \tag{6.4}$$

Assume that the node (x_I, y_I) maximizes the above equation (6.4), then the rotation angle can be calculated as

$$\theta = \arctan\left(\frac{y_I - y_O}{x_I - x_O}\right). \tag{6.5}$$

To calculate the semi-minor axis, we need to calculate the linear equation for the minor and major axis. The linear equation for the major axis can be determined:

$$y = \frac{y_I - y_O}{x_I - x_O}(x - x_O) + y_O. \tag{6.6}$$

The minor axis is perpendicular to the major axis, and its linear equation can be determined as

$$y = -\frac{x_I - x_O}{y_I - y_O}(x - x_O) + y_O. \tag{6.7}$$

Then, the distances from node i to the major and minor axis can be calculated respectively as d_{ai} and d_{bi}. The semi-minor axis can be calculated

$$b = \max_i \frac{d_{ai}^2}{1 - (d_{bi}^2/a^2)}. \tag{6.8}$$

To minimize the smallest covering ellipsoid, we need to minimize the product of a and b, since the area of a ellipse is (πab). The above calculation requires location information of all nodes. How to approximate and solve the problem in a distributed manner is one of our future work.

The second method for smallest ellipsoid estimation is based on convex optimization [90]. A ellipsoid can be defined in the matrix form [90]:

$$||A\mathbf{x} + \mathbf{b}||_2 \leq 1. \tag{6.9}$$

where $\mathbf{x} \in \mathbb{R}^n$ contains the coordinates in n dimensional space. Finding the minimum volume covering ellipsoid can be formulated as the following optimization problem:

$$\text{minimize} \quad \log\det A^{-1} \tag{6.10}$$

$$\text{subject to} \quad ||A\mathbf{x}_i + \mathbf{b}||_2 \leq 1, \tag{6.11}$$

where $\mathbf{x}_i$ is the location of node i. To solve the above optimization problem, we need to know all sensor nodes locations. How to approximate the optimization problem and use distributed optimization methods to solve the problem is also a possible future work.

Chapter 7

CONCLUSIONS

We study several consensus algorithms in distributed wireless sensor networks. First, a practical approach for reliable computation of the maximum value of local measurements over autonomous sensor networks with no fusion center is proposed. The main idea of the scheme is to use the soft-max function before transmission. The trade-off between estimation accuracy and convergence time is quantified. It is proved that the sensor network will reach consensus. That is, the state values converge to a random variable whose expectation is the sample mean of the mapped function, and the soft-max can be calculated using the consensus result. The shifted non-linear function used to adjust the transmit nonlinearity is also introduced to make the convergence speed faster. The results provide guidelines towards nonlinear transmission design, and algorithm parameter settings to trade-off between estimation error and faster convergence.

Secondly, an algorithm for reliable estimation of the number of nodes over autonomous distributed sensor networks in the presence of communication noise is studied. L_2 norm estimation is used, together with the average consensus algorithm. Different sources of error are described, and we show there is a trade-off between the estimation accuracy and the storage at sensor nodes. The Fisher information about the estimate of number of nodes in the network is calculated. How the noise and initial values at nodes affect the Cramer-Rao bound is shown. The distribution of the final estimator is also calculated to show how the design parameters affect the estimation performance.

Then, a practical approach for reliable estimation of the center and radius of a

distributed network is proposed. Soft-max approximation is used and center estimation is formulated as a convex optimization problem. By using the soft-max function, the objective function is written as a sum of differentiable functions and stochastic gradient descent and diffusion adaptation are used for distributed optimization to estimate the center. Based on the estimated center and nodes' own location information, distributed max consensus is used to estimate the radius of the network area. It is shown that the proposed algorithm works in any connected network and an accurate estimate of the network area can be obtained. A discussion on how the design parameters affect the performance and how to choose the design parameters is also given.

Finally, we propose two average consensus based distributed estimation algorithms in wireless sensor networks. The first algorithm is for reliable estimation of the degree distribution in a distributed network in the presence of noise. The main idea of the scheme is that a state vector can be used to contain the degree distribution since the degrees are discrete values. How the noise affects the performance is analyzed. We also use the structure of the degree distribution to post process the consensus results to get more accurate estimates. It is shown that when SNR is high, accurate degree distribution can be obtained in the presence of noise by post processing the convergence result based on integer degree structure. We also show that the number of bins to represent the degree distribution can be reduced in practice, thereby saving storage at sensor nodes. The second algorithm is for estimating the dynamic of a desired parameter in wireless sensor network. We show that there is a trade-off between the sensitivity to the change of the parameter and the convergence of the states of nodes. Simulations for all proposed algorithms are provided.

REFERENCES

[1] D. Varagnolo, G. Pillonetto, and L. Schenato, "Distributed statistical estimation of number of nodes in networks," in *49th IEEE Conference on Decision and Control*, Dec. 2010, pp. 1498 – 1503.

[2] ——, "Distributed cardinality estimation in anonymous networks," *IEEE Transaction on Automatic Control*, vol. 59, no. 3, pp. 645 – 659, March 2014.

[3] S. Dasarathan and C. Tepedelenlioglu, "Distributed estimation and detection with bounded transmissions over gaussian multiple access channels," *IEEE Transaction on Signal Processing*, vol. 62, no. 13, pp. 3454 – 3463, May 2014.

[4] D. Culler, D. Estrin, and M. Srivastava, "Overview of sensor networks," *IEEE Computer: Special Issue on Sensor Networks*, vol. 37, no. 8, pp. 41 – 49, Aug. 2004.

[5] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, 2002.

[6] Wikipedia, "Wireless sensor network — Wikipedia, the free encyclopedia," 2014". [Online]. Available: https://en.wikipedia.org/wiki/Wireless_sensor_network

[7] K. Martinez, J. K. Hart, and R. Ong, "Sensor network applications," *IEEE Computer*, vol. 37, 2004.

[8] J. Yick, B. Mukherjee, and G. Dipak, "Wireless sensor network survey," *Computer Networks*, vol. 52, p. 22922330, Aug. 2008.

[9] M. Goldenbaum, S. Stanczak, and M. Kaliszan, "On function computation via wireless sensor multiple-access channels," in *Wireless Communications and Networking Conference*, April 2009, pp. 1 – 6.

[10] R. W. Santucci, M. K. Banavar, C. Tepedelenlioglu, and A. Spanias, "Energy-efficient distributed estimation by utilizing a nonlinear amplifier," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, pp. 302 – 311, Jan. 2014.

[11] R. W. Santucci, M. K. Banavar, A. Spanias, and C. Tepedelenlioglu, "Design of limiting amplifier models for nonlinear amplify-and-forward distributed estimation," in *18th International Conference on Digital Signal Processing (DSP)*, July 2013, pp. 1 – 6.

[12] ——, "Nonlinear amplify and forward distributed estimation over non-identical channels," *IEEE Transactions on Vehicular Technologies*, vol. 64, no. 11, pp. 5390–5395, Nov. 2015.

[13] S. Zhang, C. Tepedelenlioglu, M. Banavar, and A. Spanias, "Distributed node counting in wireless sensor networks," in *49th Asilomar Conference on Signals Systems and Computers*, Nov. 2015.

[14] S. H. Lee, S. Lee, H. Song, and H. S. Lee, "Wireless sensor network design for tactical military applications : Remote large-scale environments," in *2009 IEEE Military Communications Conference*, Oct. 2009, pp. 1 – 7.

[15] T. Arampatzis, J. Lygeros, and S. Manesis, "A survey of applications of wireless sensors and wireless sensor networks," in *Proceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation Intelligent Control*, June 2005, pp. 719 – 724.

[16] D. Li, K. Wong, Y. Hu, and S. A. M., "Detection, classification, and tracking of targets," *IEEE Signal Processing Magazine*, vol. 19, 2002.

[17] C. Meesookho, S. Narayanan, and C. S. Raghavendra, "Collaborative classification applications in sensor networks," in *Sensor Array and Multichannel Signal Processing Workshop Proceedings*, Aug. 2002, pp. 370 – 374.

[18] C. Tepedelenlioglu, M. K. Banavar, and A. Spanias, "On the asymptotic efficiency of distributed estimation systems with constant modulus signals over multiple-access channels," *IEEE Transactions on Information Theory*, vol. 57, no. 10, pp. 7125–7130, Oct 2011.

[19] X. Zhang, M. K. Banavar, M. Willerton, A. Manikas, C. Tepedelenlioglu, A. Spanias, T. Thornton, E. Yeatman, and A. G. Constantinides, "Performance comparison of localization techniques for sequential wsn discovery," in *Sensor Signal Processing for Defence (SSPD 2012)*, Sept 2012, pp. 1–5.

[20] M. K. Banavar, C. Tepedelenlioglu, and A. Spanias, "Distributed snr estimation with power constrained signaling over gaussian multiple-access channels," *IEEE Transactions on Signal Processing*, vol. 60, pp. 3289 – 3294, Feb. 2012.

[21] X. Zhang, C. Tepedelenliolu, M. K. Banavar, and A. Spanias, "Distributed location detection in wireless sensor networks," in *2013 Asilomar Conference on Signals, Systems and Computers*, Nov 2013, pp. 428–432.

[22] M. K. Banavar, J. J. Zhang, B. Chakraborty, H. Kwone, Y. Li, H. Jiang, A. Spanias, C. Tepedelenlioglu, C. Chakrabartie, and A. Papandreou-Suppappola, "An overview of recent advances on distributed and agile sensing algorithms and implementation," *Digital Signal Processing*, vol. 39, p. 114, april 2015.

[23] M. Castillo-Effer, D. H. Quintela, W. Moreno, R. Jordan, and W. Westhoff, "Wireless sensor networks for flash-flood alerting," in *Proceedings of the Fifth IEEE International Caracas Conference on Devices, Circuits and Systems, 2004.*, vol. 1, Nov 2004, pp. 142–146.

[24] G. Werner-Allen, K. Lorincz, M. Ruiz, O. Marcillo, J. Johnson, J. Lees, and M. Welsh, "Deploying a wireless sensor network on an active volcano," *IEEE Internet Computing*, vol. 10, no. 2, pp. 18–25, March 2006.

[25] K. Lorincz, D. J. Malan, T. R. F. Fulford-Jones, A. Nawoj, A. Clavel, V. Shnayder, G. Mainland, M. Welsh, and S. Moulton, "Sensor networks for emergency response: challenges and opportunities," *IEEE Pervasive Computing*, vol. 3, no. 4, pp. 16–23, Oct 2004.

[26] T. Gao, D. Greenspan, M. Welsh, R. R. Juang, and A. Alm, "Vital signs monitoring and patient tracking over a wireless network," in *2005 IEEE Engineering in Medicine and Biology 27th Annual Conference*, Jan 2005, pp. 102–105.

[27] C. R. Baker, K. Armijo, S. Belka, M. Benhabib, V. Bhargava, N. Burkhart, A. D. Minassians, G. Dervisoglu, L. Gutnik, M. B. Haick, C. Ho, M. Koplow, J. Mangold, S. Robinson, M. Rosa, M. Schwartz, C. Sims, H. Stoffregen, A. Waterbury, E. S. Leland, T. Pering, and P. K. Wright, "Wireless sensor networks for home health care," in *Advanced Information Networking and Applications Workshops, 2007, AINAW '07. 21st International Conference on*, vol. 2, May 2007, pp. 832–837.

[28] Y. H. Nam, Z. Halm, Y. J. Chee, and K. S. Park, "Development of remote diagnosis system integrating digital telemetry for medicine," in *Proceedings of the 20th Annual International Conference of the IEEE*, Oct. 1998, pp. 1170 – 1173.

[29] N. A. Lynch, *Distributed Algorithms*. CA: Morgan Kaufmann, 1997.

[30] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *IEEE Signal Processing Magazine*, vol. 95, no. 1, 2007.

[31] R. Olfati-Saber, J. Fax, and R. Murray, "Consensus and cooperation in networked multi-agent systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, Jan. 2007.

[32] R. Olfati-Saber and R. 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, Sept. 2004.

[33] 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: Regular Papers*, pp. 302 – 311, Jan. 2014.

[34] S. Kar and J. 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.

[35] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," in *Proc. 42nd IEEE Conference on Decision and Control*, vol. 5, Dec. 2003, pp. 4997 – 5002.

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

[37] D. Zhao, Z. An, and Y. Xu, "Time synchronization in wireless sensor networks using max and average consensus protocol," *International Journal of Distributed Sensor Networks*, Feb. 2013.

[38] A. Papachristodoulou, A. Jadbabaie, and U. Munz, "Effects of delay in multi-agent consensus and oscillator synchronization," *IEEE Transactions on Automatic Control*, vol. 55, no. 6, pp. 1471–1477, 2010.

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

[40] L. Xiao, S. Boyd, and S. Kim, "Distributed average consensus with least-mean-square deviation," *Journal of Parallel and Distributed Computing*, vol. 67, pp. 33 – 46, 2007.

[41] S. Dasarathan, C. Tepedelenlioglu, M. Banavar, and A. Spanias, "Non-linear distributed average consensus using bounded transmissions," *IEEE Transactions on Signal Processing*, vol. 61, pp. 6000 – 6009, Dec. 2013.

[42] ——, "Robust consensus in the presence of impulsive channel noise," *IEEE Transactions on Signal Processing*, vol. 63, pp. 2118 – 2129, March 2015.

[43] P. Braca, S. Marano, and V. Matta, "Running consensus in wireless sensor networks," in *2008 11th International Conference on Information Fusion*, June 2008, pp. 1 – 6.

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

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

[46] F. Iutzeler, P. Ciblat, and J. Jakubowicz, "Analysis of max-consensus algorithms in wireless channels," *IEEE Transactions on Signal Processing*, vol. 60, pp. 6103 – 6107, Nov. 2012.

[47] Z. Li, R. Yu, and M. Huang, "A distributed consensus-based cooperative spectrum-sensing scheme in cognitive radios," *IEEE Transactions on Vehicular Technology*, vol. 59, no. 1, pp. 383 – 393, Jan. 2010.

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

[49] A. Tahbaz-Salehi and A. Jadbabaie, "A one-parameter family of distributed consensus algorithms with boundary: From shortest paths to mean hitting times," in *45th IEEE Conference on Decision and Control*, Dec. 2006, pp. 4664 – 4669.

[50] B. Nejad, S. Attia, and J. Raisch, "Max-consensus in a max-plus algebraic setting: The case of fixed communication topologies," in *International Symposium on Information, Communication and Automation Technologies*, Oct. 2009, pp. 1–7.

[51] G. Shi and K. H. Johansson, "Convergence of distributed averaging and maximizing algorithms part ii: State-dependent graphs," in *2013 American Control Conference*, June 2013, pp. 6859 – 6864.

[52] S. Giannini, A. Petitti, D. D. Paola, and A. Rizzo, "Asynchronous max-consensus protocol with time delays: Convergence results and applications," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 63, pp. 256 – 264, Jan. 2016.

[53] J. Cortes, "Distributed algorithms for reaching consensus on general functions," *Automatica*, vol. 44, no. 3, pp. 401 – 420, 2008.

[54] D. Bauso, L. Giarre, and R. Pesenti, "Nonlinear protocols for optimal distributed consensus in networks of dynamic agents," *System and Control Letters*, vol. 55, no. 11, pp. 918 – 928, June 2006.

[55] G. Giakkoupis and T. Sauerwald, "Rumor spreading and vertex expansion," in *SODA*, 2012, pp. 1623 – 1641.

[56] U. Feige, D. Peleg, R. Raghavan, and E. Upfal, "Randomized broadcast in networks," *Random Structures and Algorithms*, vol. 1, no. 4, pp. 447 – 460, 1990.

[57] S. Zhang, C. Tepedelenlioglu, M. Banavar, and A. Spanias, "Max-consensus using the soft maximum," in *2013 Asilomar Conference on Signals, Systems and Computers*, Nov. 2013, pp. 433 – 437.

[58] ——, "Max consensus in sensor networks: Non-linear bounded transmission and additive noise," *IEEE Sensors Journal*, vol. 16, pp. 9089 – 9098, 2016.

[59] Wikipedia, "Laplacian matrix." [Online]. Available: https://en.wikipedia.org/wiki/Laplacian_matrix

[60] N. M. M. de Abreu, "Old and new results on algebraic connectivity of graphs," *Linear Algebra and its Applications*, vol. 423, pp. 53–73, May. 2007.

[61] A. Kashyap and T. B. R. Srikant, "Quantized consensus," *Automatica*, vol. 43, pp. 1192 – 1203, Jan. 2007.

[62] S. Kar and J. M. Moura, "Distributed consensus algorithms in sensor networks: Quantized data and random link failures," *IEEE Transactions on Signal Processing*, vol. 50, no. 3, pp. 1383 – 1400, March 2010.

[63] ——, "Distributed consensus algorithms in sensor networks: Quantized data and random link failures," *IEEE Transactions on Signal Processing*, vol. 58, pp. 1383 – 1400, Nov. 2009.

[64] T. C. Aysal, M. J. Coates, and M. G. Rabbat, "Distributed average consensus with dithered quantization," *IEEE Transactions on Signal Processing*, vol. 56, no. 10, pp. 4905 – 4918, Oct. 2008.

[65] R. Olfati-Saber, "Ultrafast consensus in small-world networks," in *Proceeding of the 2005 American Control Conference*, June 2005, pp. 2371 – 2378.

[66] M. Huang and J. Manton, "Stochastic consensus seeking with noisy and directed inter-agent communication: Fixed and randomly varying topologies," *IEEE Transaction on Automatic Control*, vol. 55, no. 1, pp. 235 – 241, Jan. 2010.

[67] S. Dasarathan, C. Tepedelenlioglu, M. K. Banavar, and A. Spanias, "Robust consensus in the presence of impulsive channel noise," *IEEE TRANSACTIONS ON SIGNAL PROCESSING*, vol. 63, no. 8, pp. 2118 – 2129, April 2015.

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

[69] M. Nevelson and R. Khasminskil, *Stochastic Approximation and Recursive Estimation*. Amer. Math Soc., 1973.

[70] M. Korman, "Minimizing interference in ad hoc networks with bounded communication radius," *Information Processing Letters*, vol. 112, pp. 748 – 752, Oct. 2012.

[71] S. Zhang, C. Tepedelenlioglu, J. Lee, H. Braun, and A. Spanias, "Cramer-rao bounds for distributed system size estimation using consensus algorithms," in *Sensor Signal Processing for Defence*, Sept. 2016, pp. 1 – 5.

[72] S. Zhang, C. Tepedelenlioglu, M. Banavar, and A. Spanias, "Distributed node counting in wireless sensor networks in the presence of communication nois," *IEEE Sensors Journal*, vol. 17, pp. 1175 – 1186, Feb. 2017.

[73] A. Ganesh, A. Kermarrec, E. Merrer, and L. Massoulie, "Peer counting and sampling in overlay networks based on random walks," *Distributed Computing*, vol. 20, pp. 267 – 278, Jan. 2007.

[74] M. Bawa, H. Garcia-Molina, A. Gionis, and R. Motwani, "Estimating aggregates on a peer-to-peer network," Stanford InfoLab, Technical Report 2003-24, April 2003. [Online]. Available: http://ilpubs.stanford.edu:8090/586/

[75] Wikipedia, "Kurtosis — Wikipedia, the free encyclopedia." [Online]. Available: https://en.wikipedia.org/wiki/Kurtosis

[76] S. Kamath, D. Manjunath, and R. Mazumdar, "On distributed function computation in structure-free random wireless networks," *IEEE Transactions on Information Theory*, vol. 60, no. 1, pp. 432–442, Jan. 2014.

[77] M. Jelasity and A. Montresor, "Epidemic-style proactive aggregation in large overlay networks," in *2004. Proceedings. 24th International Conference on Distributed Computing Systems*, 2004, pp. 102 – 109.

[78] D. W. Hearn and J. Vijay, "Efficient algorithms for the (weighted) minimum circle problem," *Operations Research*, vol. 30, pp. 777 – 795, July 1982.

[79] Z. Yu, J. Teng, X. Li, and D. Xuan, "On wireless network coverage in bounded areas," in *INFOCOM, 2013 Proceedings IEEE*, July 2013, pp. 1195 – 1203.

[80] S. Vu, Chinh T.and Gao, W. P. Deshmukh, and L. Yingshu, "Distributed energy-efficient scheduling approach for k-coverage in wireless sensor networks," in *IEEE Military Communications Conference, 2006*, Oct. 2006, pp. 1 – 7.

[81] J. Chen and A. H. Sayed, "Diffusion adaptation strategies for distributed optimization and learning over networks," *IEEE Transactions on Signal Processing*, vol. 60, pp. 4289 – 4305, Aug. 2012.

[82] F. Cattivelli and A. Sayed, "Diffusion lms strategies for distributed estimation," *IEEE Transactions on Signal Processing*, vol. 58, pp. 1035–1048, 2010.

[83] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, pp. 393 – 422, 2002.

[84] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, pp. 1 – 122, Jan. 2011.

[85] G. Montavon, G. B. Orr, and K. R. Muller, *Neural Networks: Tricks of the Trade*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2012.

[86] C. Darken and J. Moody, "Towards faster stochastic gradient search," in *NIPS'91 Proceedings of the 4th International Conference on Neural*, Dec. 1991, pp. 1009 – 1016.

[87] 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.

[88] M. Newman, "The structure and function of complex networks," *SIAM Review*, vol. 45, no. 2, pp. 167 – 256, Jan. 2003.

[89] M. Goldenbaum, H. Boche, and S. Stanczak, "Nomographic functions: Efficient computation in clustered gaussian sensor networks," *IEEE Transaction on Wireless Communications*, vol. 14, no. 4, pp. 343 – 356, April 2015.

[90] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2007.

[91] L. Devroye and G. Toussaint, "A note on linear expected time algorithms for finding convex hulls," *Computing*, vol. 26, pp. 361 – 366, 1981.

[92] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. MIT Press and McGraw-Hill, 2001.

[93] S. Li, H. Fan, and Y. Wang, "Finding the smallest ellipse containing a point set based on genetic algorithms," in *IEEE International Symposium on Knowledge Acquisition and Modeling Workshop*, Dec. 2008, pp. 693 – 696.

[94] G. Marsaglia, "Ratios of normal variables and ratios of sums of uniform variables," *Journal of the American Statistical Association*, vol. 60, no. 309, pp. 193 – 204, March 1965.

[95] D. Hinkley, "On the ratio of two correlated normal random variables," *Oxford University Press on behalf of Biometrika Trust*, vol. 56, no. 3, pp. 635–639, Dec. 1969.

[96] H. A. David and H. N. Nagaraja, *Order Statistics*. WILEY, 2003.

[97] R. Zamir, "A proof of the fisher information inequality via a data processing argument," *IEEE Transactions on Information Theory*, vol. 44, no. 1, pp. 1246 – 1250, May 1998.

APPENDIX A

PROOF OF OPTIMAL ASYMPTOTIC COVARIANCE MATRIX FOR MAX CONSENSUS IN CHAPTER 2

The Convergence will be slow when $\|\mathbf{C}\|$ is large, where $\|\mathbf{C}\|$ is the max eigenvalue of $\mathbf{C}$. The problem can be formulated as,

$$\|\mathbf{C}\| = \max_{\{x|x\in\mathbb{R}^N\ \|\mathbf{x}\|\leq 1\}} \mathbf{x}^{\mathrm{T}}\mathbf{C}\mathbf{x}. \tag{A.1}$$

Let $\mathbf{U} = [\frac{\mathbf{1}}{\sqrt{N}}\ \mathbf{\Phi}]$, the columns of $\mathbf{U}$ are the eigenvectors of $\mathbf{L}$. Since $\mathbf{L}$ is an Hermitian matrix, the columns of $\mathbf{U}$ form an orthonormal basis of $\mathbb{R}^N$. Let $\mathbf{x} = \mathbf{U}\mathbf{z}$ with $\|\mathbf{z}\| \leq 1$, we have

$$\mathbf{x}^{\mathrm{T}}\mathbf{C}\mathbf{x} = (\mathbf{U}\mathbf{z})^{\mathrm{T}}\mathbf{C}(\mathbf{U}\mathbf{z}) \tag{A.2}$$

$$= \mathbf{z}^{\mathrm{T}}\left(\frac{a^2\sigma_n^2\mathbf{U}^{\mathrm{T}}\mathbf{1}\mathbf{1}^{\mathrm{T}}\mathbf{U}}{N} + \frac{\mathbf{U}^{\mathrm{T}}\mathbf{\Phi}\mathbf{S}^{\theta_0}\mathbf{\Phi}^{\mathrm{T}}\mathbf{U}}{N}\right)\mathbf{z} \tag{A.3}$$

$$= \mathbf{z}^{\mathrm{T}}\left\{\mathbf{A}_1 + \mathbf{A}_2\right\}\mathbf{z}^{\mathrm{T}} = \mathbf{z}^{\mathrm{T}}\mathbf{A}_3\mathbf{z}^{\mathrm{T}}, \tag{A.4}$$

where $\mathbf{A}_1 = \mathrm{diag}\left[a^2\sigma_n^2,\ 0,\ \cdots,0\right]_{N\times N}$, $\mathbf{A}_2 = \mathrm{diag}\left[0,\ \frac{1}{N}\mathbf{S}_{1,1},\ \cdots,\frac{1}{N}\mathbf{S}_{n-1,n-1}\right]_{N\times N}$ and $\mathbf{S}_{i,i} = \frac{a^2\sigma_n^2}{2ah'(\theta_0)\lambda_{i+1}(\mathbf{L})-1}$, $i = 1,2,\cdots,N-1$. $\mathbf{A}_3 = \mathrm{diag}\left[a^2\sigma_v^2,\ \frac{1}{N}\mathbf{S}_{1,1},\ \cdots,\frac{1}{N}\mathbf{S}_{n-1,n-1}\right]_{N\times N}$. Equality in (A.4) holds since the columns of $\mathbf{\Phi}$ are orthogonal to $\mathbf{1}$ and $\mathbf{S}^{\theta_0}$ is a diagonal matrix which can be calculated as,

$$\mathbf{S}^{\theta_0} = a^2\int_0^\infty e^{[ah'(\theta_0)\mathbf{B}+\mathbf{I}/2]t}\mathbf{C}e^{[ah'(\theta_0)\mathbf{B}+\mathbf{I}/2]t}\,dt \tag{A.5}$$

$$= a^2\sigma_n^2\int_0^\infty e^{\mathbf{H}t}dt \tag{A.6}$$

$$= \mathrm{diag}\left[\frac{a^2\sigma_n^2}{2ah'(\theta_0)\lambda_2(\mathbf{L})-1},\cdots,\frac{a^2\sigma_n^2}{2ah'(\theta_0)\lambda_N(\mathbf{L})-1}\right], \tag{A.7}$$

where $\mathbf{H}$ is an $(N-1)\times(N-1)$ diagonal matrix and $\mathbf{H}_{i,i} = 2ah'(\theta_0)\lambda_{i+1}(\mathbf{L})-1$. Note that (A.7) holds under the assumption that $2ah'(\theta_0)\lambda_i(\mathbf{L}) - 1 > 0$ for all i, which is same as the requirement in Theorem 5 in [41] that $[ah'(\theta_0)\mathbf{B} + \mathbf{I}/2]$ is stable.

Since $\lambda_2(\mathbf{L})$ is the smallest non-zero eigenvalue, we have

$$\frac{1}{N}\frac{a^2\sigma_n^2}{2ah'(\theta_0)\lambda_2(\mathbf{L})-1} \geq \frac{1}{N}\frac{a^2\sigma_n^2}{2ah'(\theta_0)\lambda_i(\mathbf{L})-1},\ \text{for } i > 2. \tag{A.8}$$

Therefore, from equation (A.4), (A.7) and (A.8), we get,

$$\|\mathbf{C}\| = \max_{\{x|x\in\mathbb{R}^N\ \|\mathbf{x}\|\leq 1\}} \mathbf{x}^{\mathrm{T}}\mathbf{C}\mathbf{x}$$

$$= \max\left\{a^2\sigma_n^2,\ \frac{1}{N}\frac{a^2\sigma_n^2}{2ah'(\theta_0)\lambda_2(\mathbf{L})-1}\right\}. \tag{A.9}$$

In the following, the optimal $\|\mathbf{C}^*\|$ is calculated together with the corresponding optimal $a = a^*$.

$$\|\mathbf{C}^*\| = \min_{\{a|2ah'(\theta_0)\lambda_2(\mathbf{L})>1\}}\ \max_{\{x|x\in\mathbb{R}^N\ \|\mathbf{x}\|\leq 1\}} \mathbf{x}^{\mathrm{T}}\mathbf{C}\mathbf{x}$$

$$= \min_{\{a|2ah'(\theta_0)\lambda_2(\mathbf{L})>1\}} \max\left\{a^2\sigma_n^2, \frac{1}{N}\frac{a^2\sigma_n^2}{2ah'(\theta_0)\lambda_2(\mathbf{L})-1}\right\}. \tag{A.10}$$

We noticed that the first term $a^2\sigma_n^2$ in equation (A.10) is a monotonic increasing function of a. The monotonicity for the second term can be check by taking the derivative respect to a, it is easy to check that the term is decreasing if $\frac{1}{2h'(\theta_0)\lambda_2(\mathbf{L})} < a \leq \frac{1}{h'(\theta_0)\lambda_2(\mathbf{L})}$, and the term is increasing if $a > \frac{1}{h'(\theta_0)\lambda_2(\mathbf{L})}$.

By checking the value of $\|\mathbf{C}^*\|$ for marginal a, we find that the problem in equation (A.10) is solved by letting,

$$a^2\sigma_n^2 = \left(\frac{1}{N}\right)\left(\frac{a^2\sigma_n^2}{2ah'(\theta_0)\lambda_2(\mathbf{L}) - 1}\right). \tag{A.11}$$

By solving equation (A.11), we get,

$$a = a^* = \left(\frac{N+1}{2N}\right)\left(\frac{1}{\lambda_2(\mathbf{L})h'(\theta_0)}\right). \tag{A.12}$$

It is easy to check that $\frac{1}{2h'(\theta_0)\lambda_2(\mathbf{L})} < a^* \leq \frac{1}{h'(\theta_0)\lambda_2(\mathbf{L})}$. Plug the optimal a^* into the expression of $\|\mathbf{C}\|$, the corresponding optimal value, $\|\mathbf{C}^*\|$ is given by,

$$\|\mathbf{C}^*\| = \left(\frac{N+1}{2N}\right)^2\left(\frac{\sigma_n^2}{\lambda_2^2(\mathbf{L})}\right)\left(\frac{1}{h'(\theta_0)}\right)^2. \tag{A.13}$$

APPENDIX B

PROOF OF THEOREM 5

In this proof, we will first use central limit theorem to approximate the distribution of the denominator and numerator of equation (3.30) using Gaussian distribution. Then, the distribution of $\hat{N}_i$ is obtained by using the ratio distribution results in [94] and [95].

First, let the numerator in equation (3.30) $\left(n' + \frac{1}{N}\sum_{i=1}^{N} x_i^2\right) = A$. Since x_i are constants and $n' \sim \mathcal{N}\left(0, \frac{1}{N^2}\left(\sum_{i=1}^{N} d_i\right)\sigma_n^2\beta\right)$, the numerator is Gaussian distributed, we have,

$$A \sim \mathcal{N}\left(\frac{1}{N}\sum_{i=1}^{N} x_i^2,\ \frac{1}{N^2}\left(\sum_{i=1}^{N} d_i\right)\sigma_n^2\beta\right). \tag{B.1}$$

We use central limit theorem to calculate the distribution of the denominator of (3.30). Since $r_i^{(k)}$ are i.i.d. random variables with mean 0 and variance 1, we can use central limit theorem to approximate the term:

$$\frac{1}{N}\sum_{i=1}^{N} r_i^{(k)} x_i \sim \mathcal{N}\left(0, \frac{1}{N^2}\sum_{i=1}^{N} x_i^2\right). \tag{B.2}$$

Also, we know that $n_i^{(k)} \sim \mathcal{N}\left(0, \frac{1}{N^2}\left(\sum_{i=1}^{N} d_i\right)\sigma_n^2\beta\right)$, as a result, we have,

$$\left(n_i^{(k)} + \frac{1}{N}\sum_{i=1}^{N} r_i^{(k)} x_i\right) \sim \mathcal{N}\left(0, \frac{\sum_{i=1}^{N} x_i^2}{N^2} + \frac{\left(\sum_{i=1}^{N} d_i\right)\sigma_n^2\beta}{N^2}\right). \tag{B.3}$$

Therefore, the denominator of (3.30) is the sample mean of the square of K i.i.d. Gaussian random variables. Note that square of Gaussian distribution is scaled chi-squared distribution with degrees of freedom equals to 1, and its mean equals to $\left(\frac{\sum_{i=1}^{N} x_i^2}{N^2} + \frac{\left(\sum_{i=1}^{N} d_i\right)\sigma_n^2\beta}{N^2}\right)$ and variance equals to $2\left(\frac{\sum_{i=1}^{N} x_i^2}{N^2} + \frac{\left(\sum_{i=1}^{N} d_i\right)\sigma_n^2\beta}{N^2}\right)^2$. By using central limit theorem, we can approximate the denominator of (3.30) with Gaussian distribution, let $\frac{1}{K}\sum_{i=1}^{K}\left(n_i^{(k)} + \frac{1}{N}\sum_{i=1}^{N} r_i^{(k)} x_i\right)^2 = B$, we have,

$$\begin{aligned} B &\sim \mathcal{N}\left(\mu_B, \sigma_B^2\right) \\ \mu_B &= \frac{\sum_{i=1}^{N} x_i^2}{N^2} + \frac{\left(\sum_{i=1}^{N} d_i\right)\sigma_n^2\beta}{N^2} \\ \sigma_B^2 &= \frac{2}{K}\left(\frac{\sum_{i=1}^{N} x_i^2}{N^2} + \frac{\left(\sum_{i=1}^{N} d_i\right)\sigma_n^2\beta}{N^2}\right)^2. \end{aligned} \tag{B.4}$$

Finally, Since both numerator and denominator are Gaussian distributed as shown in equation (B.1) and (B.4). The results for Gaussian ratio distribution proposed in [95, eqn(1)] can be used to calculate the distribution for $\hat{N}_i$ in equation (3.30), and Theorem 5 is proved. Note that the numerator and denominator are independent since the noise are i.i.d. and x_i are constants.

APPENDIX C

PROOF OF THEOREM 6

Assume max consensus is used and convergence is perfectly reached at time t. Let $y_i = x_i(t) = \max\{x_1, x_2, \cdots, x_N\}$ be the state at node i after convergence. Since x_i are i.i.d., the distribution of y_i can be calculated

$$\text{CDF}: \{F(y)\}^N, \tag{C.1}$$

$$\text{PDF}: N\{F(y)\}^{N-1} f(y). \tag{C.2}$$

Therefore, the Fisher information can be calculated from the definition, we have

$$\mathcal{I}_{\max} = \mathrm{E}\left[\left(\frac{\partial}{\partial N}\ln\left[N\{F(y)\}^{N-1} f(y)\right]\right)^2\right] \tag{C.3}$$

$$= \frac{1}{N^2} + \frac{2}{N}\mathrm{E}\left[\ln F(y)\right] + \mathrm{E}\left[(\ln F(y))^2\right]. \tag{C.4}$$

Note that the system size estimation problem is formulated as a conventional parameter estimation problem, and N is estimated based on random values generated at nodes. Therefore, the estimate of the system size $\hat{N}$ can be non-integers and the differentiating respect to N in equation (C.3) make sense.

The term $\mathrm{E}\left[\ln F(y)\right]$ in equation (C.4) can be calculated by definition,

$$\mathrm{E}\left[\ln F(y)\right] = \int_{-\infty}^{\infty} \{\ln F(y)\} N\{F(y)\}^{N-1} f(y)dx = -\frac{1}{N}. \tag{C.5}$$

Similarly, term $\mathrm{E}\left[(\ln F(y))^2\right] = \frac{2}{N^2}$ can be calculated Therefore, by substituting the calculated values into equation (C.4), the result in (3.53) can be obtained.

APPENDIX D

PROOF OF THEOREM 8

With the error term, node i in the network will converge to $z_i = \max\{x_1, x_2, \cdots, x_N\} + e$. Assume the initial values at nodes x_i have exponential tail and its tail PDF $\lambda e^{-\lambda x}$. The distribution of the maximum $y_i = \max\{x_1, x_2, \cdots, x_N\}$ can be approximated using Gumbel distribution [96],

$$\text{CDF} : e^{-e^{-(y-\mu)/\beta}}, \;\; \text{PDF} : \frac{1}{\beta} e^{-\left(\frac{y-\mu}{\beta} + e^{-\frac{y-\mu}{\beta}}\right)}, \tag{D.1}$$

where $\mu = (\ln N)/\lambda$ and $\beta = 1/\lambda$. Therefore we can write z_i as

$$z_i = \frac{\ln N}{\lambda} + \tilde{y}_i + e, \tag{D.2}$$

where $\tilde{y}_i$ has PDF: $\lambda e^{-\left(\lambda x + e^{-\lambda x}\right)}$. Note that the closed form expression for the distribution of $\tilde{y}_i + e$ (sum of Gumbel random variable and Gaussian random variable) cannot be obtained, therefore an upper bound on the Fisher information is derived herein.

The upper bound calculation can be expressed as three phases: i) Fisher information for random variable $\tilde{y}$ and e are calculated; ii) Fisher information inequality is applied; and iii) parameter transformation lemma in [97] is used. Details of the calculation are presented in the following.

Firstly, we introduce the Fisher information of a real random variable X whose density function $f(x)$ is independent of the estimated quantity N [97],

$$\mathcal{I}_X := \int \frac{1}{f(x)} \left(\frac{\partial f(x)}{\partial x}\right)^2 dx. \tag{D.3}$$

From the definition of Fisher information of a random variable, the Fisher information for $\tilde{y}_i$ and e can be calculated

$$\mathcal{I}_{\tilde{y}} = \lambda^2, \;\; \mathcal{I}_e = \frac{1}{\sigma_e^2}. \tag{D.4}$$

Secondly, Fisher information inequality is used to calculate the upper bound on Fisher information of $\tilde{y}_i + e$, we have

$$\mathcal{I}_{\tilde{y}+e} \leq \frac{\mathcal{I}_{\tilde{y}} \mathcal{I}_e}{\mathcal{I}_{\tilde{y}} + \mathcal{I}_e} = \frac{\lambda^2}{\sigma_e^2 \lambda^2 + 1}. \tag{D.5}$$

Note that equality in equation (D.5) is achieved iff $\tilde{y}_i$ and e are both Gaussian distributed. Finally, the parameter transformation lemma (Lemma 4 in [97]) is used and the Fisher information about N can be calculated,

$$\mathcal{I}_{nmax} = \left(\frac{\partial \frac{\ln N}{\lambda}}{\partial N}\right)^2 \mathcal{I}_{\tilde{y}+e} \leq \frac{1}{N^2} \left(\frac{\lambda^{-2}}{\sigma_e^2 + \lambda^{-2}}\right). \tag{D.6}$$

Therefore, equation (3.57) and (3.58) is obtained.

Note that here we assume that x_i to be exponentially distributed for simplicity purpose. Similar proof can be obtained if the distribution of the initial values has exponential tail, for example Gaussian distribution. Also note that there is no close form expression for the true distribution of the final error in this case. If the final error is not Gaussian distributed, the upper bound on the Fisher information can be expressed as: $\mathcal{I}_{nmax} \leq \left(\frac{1}{N^2}\right) \left(\frac{\lambda^{-2}}{1/\mathcal{I}_e + \lambda^{-2}}\right)$ from equation (D.5) and (D.6).

APPENDIX E

PROOF OF CONVEXITY FOR OBJECTIVE FUNCTION IN DISTRIBUTED CENTER ESTIMATION IN CHAPTER 4

To proof the convexity of the objective function in equation (4.9), we first calculate the Hessian of the objective function. Then we show that the Hessian matrix is positive definite when $\beta > 0$.

For 2-D case, the Hessian matrix $\mathbf{H}$ is a 2×2 matrix. Let the objective function $J(x, y) = \sum_{i=1}^{N} e^{\beta\left\{(a_i-x)^2+(b_i-y)^2\right\}}$. The Hessian matrix can be calculated as:

$$\mathbf{H} = \begin{bmatrix} \frac{\partial^2 J}{\partial x^2} & \frac{\partial^2 J}{\partial x \partial y} \\ \frac{\partial^2 J}{\partial y \partial x} & \frac{\partial^2 J}{\partial y^2} \end{bmatrix}, \tag{E.1}$$

where

$$\begin{aligned} \frac{\partial^2 J}{\partial x^2} =& \sum_{i=1}^{N} 2\beta e^{\beta\left\{(a_i-x)^2+(b_i-y)^2\right\}} \\ &+ 4\beta^2 (a_i - x)^2 e^{\beta\left\{(a_i-x)^2+(b_i-y)^2\right\}}, \end{aligned} \tag{E.2}$$

$$\begin{aligned} \frac{\partial^2 J}{\partial y^2} =& \sum_{i=1}^{N} 2\beta e^{\beta\left\{(a_i-x)^2+(b_i-y)^2\right\}} \\ &+ 4\beta^2 (a_i - x)^2 e^{\beta\left\{(b_i-y)^2+(b_i-y)^2\right\}}, \end{aligned} \tag{E.3}$$

$$\frac{\partial^2 J}{\partial x \partial y} = \frac{\partial^2 J}{\partial y \partial x} = \sum_{i=1}^{N} 4\beta^2 (a_i - x)(b_i - y) e^{\beta\left\{(a_i-x)^2+(b_i-y)^2\right\}}. \tag{E.4}$$

In the following, we show that H is positive definite when $\beta > 0$. Define $\mathbf{z} = [p\ q]^{\mathrm{T}} \in \mathbb{R}^2$ be any non-zero column vector. We have

$$\begin{aligned} \mathbf{z}^{\mathrm{T}}\mathbf{H}\mathbf{z} &= \begin{bmatrix} p & q \end{bmatrix} \begin{bmatrix} \frac{\partial^2 J}{\partial x^2} & \frac{\partial^2 J}{\partial x \partial y} \\ \frac{\partial^2 J}{\partial y \partial x} & \frac{\partial^2 J}{\partial y^2} \end{bmatrix} \begin{bmatrix} p \\ q \end{bmatrix} \\ &= p^2 \sum_{i=1}^{N} \left[2\beta J_i + 4\beta^2 (a_i - x)^2 J_i\right] \\ &+ 2pq \sum_{i=1}^{N} 4\beta^2 (a_i - x)(b_i - y) J_i \\ &+ q^2 \sum_{i=1}^{N} \left[2\beta J_i + 4\beta^2 (b_i - y)^2 J_i\right] \qquad \text{(E.5)} \\ &= 2\beta p^2 \sum_{i}^{N} J_i + 2\beta q^2 \sum_{i=1}^{N} J_i \\ &+ \sum_{i=1}^{N} 4\beta^2 J_i \left\{(a_i - x)p + (b_i - y)q\right\}^2. \qquad \text{(E.6)} \end{aligned}$$

where $J_i = e^{\beta\left\{(a_i-x)^2+(b_i-y)^2\right\}}$. Equation (E.6) is positive for any p, q value since $J_i > 0$ and $\beta > 0$. Therefore, H is positive definite and the objective function in equation (4.9) is convex.