THE UNIVERSITY
of EDINBURGH

Distributed Parameter and State Estimation for Wireless Sensor Networks

Jia Yu

A thesis submitted for the degree of Doctor of Philosophy.
The University of Edinburgh.
June 19, 2017

Abstract

The research in distributed algorithms is linked with the developments of statistical inference in wireless sensor networks (WSNs) applications. Typically, distributed approaches process the collected signals from networked sensor nodes. That is to say, the sensors receive local observations and transmit information between each other. Each sensor is capable of combining the collected information with its own observations to improve performance. In this thesis, we propose novel distributed methods for the inference applications using wireless sensor networks. In particular, the efficient algorithms which are not computationally intensive are investigated. Moreover, we present a number of novel algorithms for processing asynchronous network events and robust state estimation.

In the first part of the thesis, a distributed adaptive algorithm based on the component-wise EM method for decentralized sensor networks is investigated. The distributed component-wise Expectation-Maximization (EM) algorithm has been designed for application in a Gaussian density estimation. The proposed algorithm operates a component-wise EM procedure for local parameter estimation and exploit an incremental strategy for network updating, which can provide an improved convergence rate. Numerical simulation results have illustrated the advantages of the proposed distributed component-wise EM algorithm for both well-separated and overlapped mixture densities. The distributed component-wise EM algorithm can outperform other EM-based distributed algorithms in estimating overlapping Gaussian mixtures.

In the second part of the thesis, a diffusion based EM gradient algorithm for density estimation in asynchronous wireless sensor networks has been proposed. Specifically, based on the asynchronous adapt-then-combine diffusion strategy, a distributed EM gradient algorithm that can deal with asynchronous network events has been considered. The Bernoulli model has been exploited to approximate the asynchronous behaviour of the network. Compared with existing distributed EM based estimation methods using a consensus strategy, the proposed algorithm can provide more accurate estimates in the presence of asynchronous networks uncertainties, such as random link failures, random data arrival times, and turning on or off sensor nodes for energy conservation. Simulation experiments have been demonstrated that the proposed

algorithm significantly outperforms the consensus based strategies in terms of Mean-Square-Deviation (MSD) performance in an asynchronous network setting.

Finally, the challenge of distributed state estimation in power systems which requires low complexity and high stability in the presence of bad data for a large scale network is addressed. A gossip based quasi-Newton algorithm has been proposed for solving the power system state estimation problem. In particular, we have applied the quasi-Newton method for distributed state estimation under the gossip protocol. The proposed algorithm exploits the Broyden-Fletcher-Goldfarb-Shanno (BFGS) formula to approximate the Hessian matrix, thus avoiding the computation of inverse Hessian matrices for each control area. The simulation results for IEEE 14 bus system and a large scale 4200 bus system have shown that the distributed quasi-Newton scheme outperforms existing algorithms in terms of Mean-Square-Error (MSE) performance with bad data.

Declaration of Originality

I hereby declare that the research recorded in this thesis and the thesis itself was composed and originated entirely by myself in the School of Engineering at The University of Edinburgh.

Jia Yu

Acknowledgements

My sincere gratitude first goes to Prof. John Thompson, my supervisor, for his constructive suggestions, valuable ideas, great patience and encouragement. Without his help, the completion of this thesis would have been impossible. My sincere thanks also goes to Dr. Pei-Jung Chung, who provided me the opportunity to study at the university of Edinburgh. Without her precious support, it would not be possible to conduct this research.

Moreover, I thank all the academic and research staff in the Institute for Digital Communication, for the sleepless nights we were working together before deadlines. Their lectures and instructions have been of great help to my thesis. I would also like to express my thanks to my friends for their constant concern, generous help, and meaningful comments on the study during the last four years.

Finally, I must say I owe gratitude to my parents and my wife. Their love and encouragement have supported me to overcome difficulties and achieve the final success.

Contents

Declaration of Originality . . . iii

Acknowledgments . . . iv

List of Figures . . . viii

List of Tables . . . x

Acronyms and Abbreviations . . . xi

Nomenclature . . . xiii

1 Introduction **1**

1.1 Overview . . . 1

1.2 Motivation . . . 2

1.3 Contribution . . . 3

1.4 Thesis Outline . . . 4

2 Background **5**

2.1 Wireless Sensor Networks . . . 5

2.1.1 Centralized and Decentralized approaches in Sensor Networks . . . 7

2.2 Cooperation Strategies for Exchange of Information . . . 8

2.2.1 Incremental Strategy . . . 9

2.2.2 Diffusion Strategy . . . 9

2.2.3 Consensus strategy . . . 13

2.2.4 Comparison of Diffusion and Consensus Strategies . . . 14

2.2.5 Asynchronous Diffusion Strategy . . . 15

2.3 Optimization Algorithms . . . 16

2.3.1 Expectation-Maximization Algorithm . . . 16

2.3.2 Component-Wise EM Algorithm for Mixtures . . . 18

2.3.3 EM Gradient Algorithm . . . 19

2.3.4 Least Mean Square Method . . . 20

2.4 Application in Power System State Estimation . . . 21

2.5 Chapter Summary 22

3 Distributed Component-Wise EM Algorithm for Mixture Models in Sensor Networks 23

3.1 Introduction 23

3.2 Problem Formulation 26

3.3 Standard EM and Distributed EM Algorithms (DEM) 27

3.3.1 Distributed EM Algorithm based on Incremental Strategy 29

3.3.2 Consensus based EM Algorithm 30

3.4 Distributed Component-wise EM Algorithm 30

3.5 Convergence Analysis 33

3.6 Simulation Results 40

3.6.1 Well-separated Mixtures Model 41

3.6.2 Overlapping Mixtures Model 42

3.7 Chapter Summary 44

4 Diffusion-Based EM Gradient Algorithm for Density Estimation in Sensor Networks 47

4.1 Introduction 47

4.2 Problem Formulation 49

4.3 EM Gradient Algorithms 50

4.4 Distributed Optimization via Adaptive Diffusion Strategy 52

4.5 Diffusion EM Gradient Algorithm 56

4.6 Simulation Results 59

4.6.1 1-Dimensional Data 61

4.6.2 2-Dimensional Data 63

4.7 Chapter Summary 69

5 Distributed quasi-Newton Method for Power System State Estimation 72

5.1 Introduction 72

5.2 Problem Formulation 75

5.3 Centralized quasi-Newton Algorithm 76

5.4 Distributed quasi-Newton Process 77

5.4.1 Network Exchange Model 78

5.4.2 Local Update Process 80

5.5 Convergence Analysis 82
5.5.1 Gossip Errors Analysis 82
5.5.2 Local convergence analysis 83
5.6 Simulation Results 85
5.6.1 Case A: Comparison with GGN without Bad Data 86
5.6.2 Case B: Comparison with GGN in presence of Bad Data 86
5.6.3 Case C: Comparison with ADMM Method in a Large-scale Power Network 89
5.7 Conclusion 89

6 Conclusions and Future Work **91**
6.1 Summary of the Work 91
6.2 Future Work 92

Appendix A Proof of Theorem 1 **94**

Appendix B Proof of Theorem 2 **97**

Appendix C Proof of Lemma 3 **100**

Appendix D Proof of Lemma 4 **102**

Appendix E Proof of Theorem 2 **106**

Appendix F List of Publications **110**
F.1 Conference Papers 110
F.2 Journal Papers 110

List of Figures

2.1 Network Topology 8

2.2 Incremental Strategy 10

2.3 Diffusion Strategies 12

2.4 Consensus Strategy 13

3.1 Communication/iteration cycle in a sensor network 31

3.2 Data distribution for well-separated mixture case 40

3.3 Estimates for mean values by the DCEM algorithm for well-separated mixture case. 42

3.4 Comparison of log-likelihood versus iterations for the DCEM, DEM with single EM step at each node [1] and DEM with multiple EM steps at each node (DEMM) in the well-separated mixture case. 43

3.5 Data distribution for overlapping mixture case 44

3.6 Estimates for mean values by the DCEM algorithm for overlapping mixture case. 45

3.7 Comparison of log-likelihood versus iterations for the DCEM, DEM with single EM step at each node [1] and DEM with multiple EM steps at each node (DEMM) in the overlapping mixture case. 46

4.1 A network of integrator nodes in which node m receives the state $\boldsymbol{\theta}_n$ of its neighbor, node n 52

4.2 100 randomly distributed sensors with different radio ranges 60

4.3 Estimated mean and variance for two Gaussian components with different schemes 62

4.4 Comparison of network MSD vs iteration index for asynchronous diffusion, synchronous diffusion, consensus based EM [2] and DDEM [3] 64

4.5 Estimated mean and variance with different schemes for 1000 nodes 65

4.6 Large scale network MSD (1000 nodes) comparison vs iteration index for asynchronous diffusion, synchronous diffusion, consensus based EM [2] and DDEM [3] 66

4.7 Data distribution for 2D Gaussian mixtures with 3 components 67

4.8 Evolution of the first estimated mean value with different schemes for 10 sensor nodes using 2D Gaussian mixture model 68
4.9 Estimation performance versus radio range (in meters) with the diffusion EM gradient algorithm ($p_m = q_m = 0.8$) 69
4.10 The estimated mean values versus radio ranges 70

5.1 IEEE 14 bus system partitioned into $I = 4$ control areas 74
5.2 Comparison with GGN and distributed quasi-Newton using $t = 10$ exchange for each update 87
5.3 Comparison distributed quasi-Newton against GGN with bad data 88
5.4 Comparison distributed quasi-Newton against ADMM in a large-scale power network 90

List of Tables

2.1 [4] For the node m, the numbers of the complex multiplications, additions, and exchanged $N \times 1$ vectors within each iteration are compared. n_m is the degree of node k which indicates the size of its neighborhood $\mathcal{N}_m$. 15

5.1 Execution Time and Iterations in Case A 86

Acronyms and Abbreviations

EM	Expectation Maximization
DCEM	Distributed Component-wise EM
WSN	Wireless Sensor Network
DQN	Distributed quasi-Newton
MSD	Mean Square Deviation
BFGS	Broyden-Flethcher-Goldfarb-Shanno
MSE	Mean square Error
LMS	Least Mean Square
ATC	Adapt-Then-Combine
CTA	Combine-Then-Adapt
IEEE	Institute of Electric and Electronic Engineers
ML	Maximum Likelihood
E-step	Expectation Step
M-step	Maximization Step
CEMM	Component-wise EM for Mixture
PSSE	Power System State Estimation
M-CSE	Multi-Coordinate State Estimation
GMM	Gaussian Mixture Model
RLS	Recursive Least Square
DEM	Distributed EM
DEMM	Distributed EM for Multiple Iteration
DCEM	Distributed Component-wise EM
DDEM	Diffusion-based distributed EM
WAMS	Wide-Area Measurement System
PMU	Phasor Measurement Unit
ADMM	Alternating Direction Method of Multipliers

LS	Least Square

Nomenclature

M	Number of sensors
d	scalar measurement
t	time scale
ω	Distributed manner
σ^2	variance
N	Length of data
$\mathcal{N}_m$	Set of neighborhood of node m
$b_{m,n}$	Combination coefficient for $\{m, n\}$
$\sum$	Summation
$\mathbf{R}$	Input signals correlation matrix
$\mathbf{I}$	Identity Matrix
$\mathbf{B}$	Matrix of combination coefficients
diag$(\cdot)$	Diagonal of matrix
$\mathbf{A}$	Network adjacent matrix
$\mathbf{D}$	Degree matrix
$\mathcal{P}(\mu, \Sigma)$	Gaussian density with mean μ and covariance Σ
exp	Exponential
$(\cdot)^T$	Transpose
$(\cdot)^*$	Complex conjugate
$\log(\cdot)$	Logarithm
$argmax$	Argument of maximum
$\boldsymbol{\theta}$	Unknown parameters of Gaussian mixture model
$\boldsymbol{\theta}_j$	jth parameter of Gaussian mixture model
$\boldsymbol{E}$	Expectation
$L(\cdot)$	Log likelihood
$\mathcal{L}(\cdot)$	Lagrangian function

∂	Partial derivative
$\bigotimes$	Kronecker product
$\mathbf{M}$	Rate matrix
$\rho(\mathbf{M})$	Spectral radius of $\mathbf{M}$
$\|\|\mathbf{M}\|\|_{\mathbf{N}}$	Generalized matrix spectral norm with respect to a positive definite matrix $\mathbf{N}$
∇^{ij}	ith order partial derivatives with respect to the first argument and jth order partial derivatives with respect to the second argument
β	Set of eigenvalues
$vec[\cdot]$	Vector from stacking the column vectors of matrix $\cdot$
$\mathbf{1}_N$	N$\times$1 all-one vector
$Q(\cdot)$	Conditional expectation
r	Connectivity radio range of network
$\|\cdot\|$	Cardinality
$\|\|\cdot\|\|_2$	Euclidean norm
ε	The edge of network
$\mathcal{G}(\cdot)$	Undirected Graph
v	Step size
c	Nonnegative scalar
d	Newton descent
$\mathbf{h}_I(\cdot)$	Power injection vector
$\mathbf{h}_F(\cdot)$	Power flow vector
$\mathbf{H}$	Approximation of Inverse of Hessian matrix in BFGS
τ	Time snapshot
$\mathbf{\Phi}$	Weight matrix for information exchange
max	Maximum
min	Minimum
$\mathbf{e}$	Measurement error

Chapter 1

Introduction

1.1 Overview

State-of-the-art statistical inference methods in wireless sensor networks (WSNs) and power system networks are driven by distributed approaches [5–7]. Given that the distributed signal processing algorithms process the gathered information from distributed sensor nodes, each sensor generates local observation and transmits the information within the network, and thus estimation can be improved by combining all collected signals. It is well known that centralized solutions require a central processor to process all collected data and then give feedbacks to other nodes. Therefore, the processors have to be robust enough to support the whole system. Distributed solutions release the computational burdens by processing the data locally. The computation is thus significantly reduced, and lower communication bandwidth can be applied.

The major strategies for distributed adaptive processing include the incremental, diffusion, and consensus strategies, within which the diffusion one had been shown to have the best efficiency [4]. We will present the details of the strategies in Chapter 2. There are still a number of challenges for state-of-the-art distributed diffusion processing. For example, the coefficients for combing the signals from the neighbours need to be computed after the network starts to work properly, and this process will be affected by poor connections. Furthermore, one local node would need a wide enough communication bandwidth to support large number of neighbours. Also it is not easy to transmit the estimations if the unknown parameters associated with latent data sets. This issue occurs for the applications considering large scale data sets as the convergence rate would be limited by the data dimension. It is worth mentioning that the unknowns can be sparse in particular scenarios. If we still feed in all the data in the processing, it would be challenging to deal with the

increased computational burden, the slower convergence and the corrupted mean square deviation (MSD) performance.

1.2 Motivation

Energy efficiency, reliability, high estimation accuracy, and fast convergence are significant factors to evaluate the performance of estimation algorithm for WSNs. However, most exist algorithms are used in a centralized way for WSNs, which require the central process unit has the strong ability to process all the data. On the other hand, some parameters of interest can be estimated upon the network using the distributed estimation to leverage the local estimations and the links between nodes. The Expectation-Maximization (EM) algorithm are widely used in WSNs for solving the mixture model estimation problem [1]. However, the most documented problem associated with EM is its possibility of slow convergence. All existing algorithms [8] in WSNs require to update parameters simultaneously during the iteration procedure. They are only effective when the mixtures are well-separated. If the mixtures become complex or overlapping, they suffer from a slow convergence. Therefore, decomposition of the mixture parameter into component parameter and updating only one component at one iteration could be a better solution for this scenario. As a result, the decoupling of parameter updates implies the use of the smallest admissible missing data space and leads to faster convergence.

It is a common practice that the estimation in WSNs are interfered by asynchronous network behaviors . The conventional estimation algorithms like the EM, which are based on consensus and incremental strategies, can not continuous evolve in presence of asynchronous network events. But in the literature few adaptive algorithms have been reported based on asynchronous network [9–11]. Thus, there is a need to develop distributed version of EM estimation algorithm which will provide improved performance to deal with asynchronous network behaviours.

Distributed state estimation is important for power system and smart grid [6]. But it is not practicable in presence of bad data, which results in large residual in power system and impacts on the accuracy of state estimation. The conventional methods are to test the large residual in power system and remove it before estimating the state of system. The second processing is to suppress

the effects of bad data on state estimation in stead of removing it [12]. In addition, existing works for distributed state estimation are effective for small scale power system networks, which are difficult to apply into large scale networks for the reason of high computation complexity and slow convergence. To reduce the requirement of computation and speed up convergence, second order optimization are considered in this research.

1.3 Contribution

The contributions presented in this thesis are summarised as follows:

- Based on the fact that the component-wise EM algorithm has a faster convergence rate [8] than the standard EM algorithms, a component-wise EM for Gaussian mixture model based on distributed incremental solution is reported. In detail, we develop a distributed component-wise EM algorithm (DCEM) and analyse the convergence properties in both local processing and network updating. This algorithm can be used for defence, such as battlefield intelligence, movement estimation and detection, and distributed target tracking.

- The diffusion-based EM gradient method for asynchronous network problems is proposed and studied. Specifically, we develop an EM gradient algorithm that can exploit the asynchronous adapt-then-combine diffusion strategy among the sensor nodes. The proposed algorithm applies the Bernoulli model to describe the asynchronous behaviour of wireless sensor network. The proposed algorithm results in improved estimation performance in terms of the mean square deviation (MSD) associated with the estimates. In contrast to previously reported techniques, a key feature of the proposed algorithms is that they involve only EM procedure associated with the perfect synchronous network condition.

- The design of an approach, namely distributed quasi-Newton (DQN) scheme, that exploits Broyden-Flethcher-Goldfarb-Shanno (BFGS) formula [13] to update the estimates under a gossip protocol, improves the mean square error (MSE) performance in the presence of bad data and large scale network setting is proposed. We also present a design procedure and develop an algorithm to optimize the line search method, which can find a suitable step size to coordinate the whole network. In addition, we have demonstrated its convergence

properties under the network gossiping strategy.

1.4 Thesis Outline

This thesis is organized as follows:

- Chapter 2 presents an overview of the theory relevant to this thesis and introduces the system models that are used to present the work in this thesis. The topics of distributed signal processing, incremental and diffusion strategies, optimization methods and power system state estimation are covered with an outline of previous work in these fields and important applications .

- Chapter 3 presents the design of distributed algorithm for Gaussian mixture model in a wireless sensor networks based on component-wise EM procedure. The incremental version is proposed, alongside a convergence analysis and the application to mixture density estimation.

- In Chapter 4, EM gradient algorithm for density estimation and its application to asynchronous WSNs is proposed. The derivation of the proposed algorithm is presented in terms of adapt-then-combine diffusion strategy and asynchronous network behaviour.

- Chapter 5 presents a novel distributed quasi-Newton scheme for distributed state estimation in power system network. A distributed BFGS algorithm joint synchronous gossip protocol is developed and compared with existing techniques.

- Chapter 6 presents the conclusions of this thesis, and suggests directions in which further research could be carried out.

Chapter 2

Background

In this chapter, an introduction of fundamental algorithms and techniques related to the research carried out during the preparation of this thesis, including the properties of wireless sensor networks, strategies for cooperation, optimization algorithms, state estimation in power system are presented.

2.1 Wireless Sensor Networks

A sensor network is a group of sensor nodes which have communication and sensing capabilities. The sensors function together as a cooperative network for the purpose of monitoring the environment. Practically, sensor nodes are often formed in different modalities, such as radar, acoustic and thermal, based on specific sensing applications. Common features for these sensors include low-power, memory-constrained and communications. The development of WSNs [14–16] was driven by the battlefield applications such as area monitoring and military reconnaissance. WSNs appears to be developed into a powerful tool to observe and understand the regional phenomena. The sensors in a typical WSN share local observations via wireless links, and cooperatively pass the data to a main site to analyse and understand the state of the environment. The sensing systems are often integrated with signal processing techniques, such as the environmental parameter estimation and target classification, to extract the high-level information for further applications. In particular, the estimation of environmental parameters offers us further insight into describing the environment, and the classification of moving targets is necessary for general battlefield monitoring. State-of-the-art wireless sensor networks have been widely used in a variety of fields, e.g. battlefield surveillance, environmental monitoring, health care. The followings are the main characteristics of WSNs [17]:

- Low expenditure: Numerous sensor nodes are distributed in the environment to set up the WSN. The cost of sensors have to be low to support such a large network.

- Energy consumption: The computation, communication and storage are the main sources of consuming the energy. Since in general there is not route to charge the sensors, we should take account of the energy consumption factors in the algorithms.

- Computational load: All the sensor nodes are constrained by their computational power and energy need to be considered.

- Communication abilities: The wireless communications in WSNs are usually limited by the short range and narrow bandwidth. Also it is hard for the WSNs to work properly within unattended areas. The reliability, security and resiliency need to be considered in design.

- Security and privacy: In a number of scenarios, the sensors are required to be able to prevent from unauthorised access and intentional attacks. Privacy policies also need to be considered.

- Distributed sampling and processing: WSNs often consist of thousands of deployed sensors and each of them is designed to observe, communicate and process information. Such a system can benefit from the distributed processing.

- Dynamic topology: Typical WSNs are not static. Sensors can be eliminated and added which result in the changes on the topology. Consider this context, the nodes should be equipped with the reconfiguration, self-adjustment capabilities.

- Self-organization: Especially in hostile environments, the sensors are required to organize them selves to set up the network. They should be able to cooperatively adjust themselves and work automatically.

- Robustness: Since the sensors should be able to work in tough environments, they need to be error tolerant. Ideal nodes should be equipped with the self-calibrate ability.

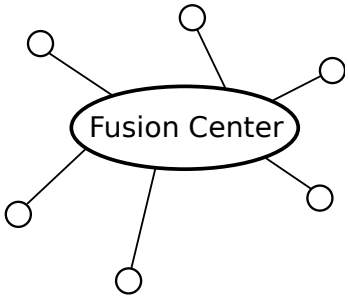
- Tiny outlook: Most sensor nodes are required to be small in the physical sizes. Also the energy consumption and communication power will be limited due to the small sizes.

2.1.1 Centralized and Decentralized approaches in Sensor Networks

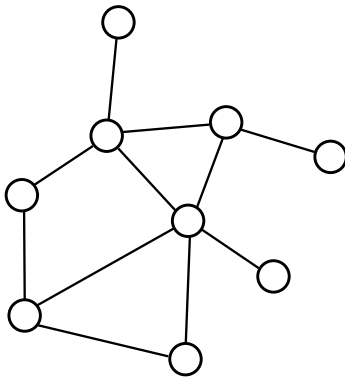
Collaborative parameter estimation in a sensor network can be carried out in two ways - centralized or decentralized [18–20], as shown in Figure 2.1. In the centralized approaches, the sensor nodes transmit their data or local statistics to a centralized unit, named a fusion center, which has the ability of processing the data centrally. The fusion center then sends the results back to the each nodes [20, 21]. In this scheme, centralized unit has a powerful processing capacity and the local sensor does not need the computational capability. However there exist some limitations in this kind of scheme. In real-time scenarios where each nodes collect data continuously, the exchange of information between the sensor nodes and the centralized fusion center require a large communication bandwidth. In addition, when the central processor fails to process the data, it will render the whole network impossible to use.

With the decentralized methods, the sensor nodes exchange summary statistics among neighbors to evaluate the global objective functions under a distributed manner. The continuous diffusion of summary statistics across the network enables nodes to adapt their performance in terms of network conditions. Distributed methods are of interest in scenarios in which a centralized unit is either unavailable or prohibitively costly; such examples include military or agriculture monitoring applications in which the nodes are deployed over a wide region and also energy constrained [20, 22].

In the distributed networks, any node can connect to other nodes directly, which increase reliability [23–25]. Different from central network, there are some advantages in distributed sensor networks. First, if the center crashes, the whole network is still working, as there is no a central processor. Second, nodes are connected to each other in a distributed network so that multiple paths can be selected for data transmission in the network. Also, influenced by the network topology, each node collects the information of the target, and shares this information with other nodes, to give estimations on the parameters of interest. A number of methods have been developed for distributed wireless networks, e.g. the EM algorithm, EM variant algorithms, and Least-Mean-Square method.



(a) A centralized network



(b) A decentralized network

Figure 2.1: Network Topology

2.2 Cooperation Strategies for Exchange of Information

As introduced previously, there are three cooperation strategies: incremental, diffusion and consensus [26]. In this section, each cooperation strategy is analysed for information exchange between sensor nodes. For simplicity, we illustrate these strategies based on linear regression model [27]. Considering a network with M nodes over a spatial domain for the incremental strategy, each node takes a scalar measurement d_m^t at each time scale t, according to linear regression model:

$$d_m^t = \boldsymbol{x}_m^t \boldsymbol{\omega}_0 + e_m^t \tag{2.1}$$

where $\boldsymbol{x}_m^t$ represents the $1 \times N$ input data, N denotes the length of data and e_m^t is the zero mean noise sample at each node with the variance σ_m^2. In order to approximate $\boldsymbol{\omega}_0$ in a distributed manner, each nodes is required to minimize the local cost function [26]

$$J_{\boldsymbol{\omega}}(\boldsymbol{\omega}_m^t) = E|d_m^t - \boldsymbol{x}_m^t \boldsymbol{\omega}_0|^2 \tag{2.2}$$

where E is the expectation and $\boldsymbol{\omega}_m^t$ stands for the estimated vector at node m with time scale t. Then, the global cost function for the entire network can be described as:

$$J_{\boldsymbol{\omega}}(\boldsymbol{\omega}) = \sum_{m=1}^{M} E|d_m^t - \boldsymbol{x}_m^t \boldsymbol{\omega}_0|^2 \tag{2.3}$$

In the following, this global cost function can be minimized by the least mean square method (LMS) in different cooperation strategies.

2.2.1 Incremental Strategy

The simplest cooperation is incremental strategy [5, 28], following a Hamiltonian cycle. These nodes receive the information from adjacent node, and re-transmit the information to next node in a pre-determined direction. In the incremental scheme, the scalar measurement d_m^t at node m, the input signal vector $\boldsymbol{x}_m^t$, and the local estimate $\boldsymbol{\psi}_{m-1}^t$ from adjacent nodes are used to construct the local estimate $\boldsymbol{\psi}_m^t$ of the network through a distributed estimation strategy [5]. Then, the local estimate of $\boldsymbol{\psi}_m^t$ is passed to the next node $m+1$ in one direction. The final node's estimate is equal to the final estimation of the network. Based on the traditional LMS algorithm, the incremental LMS algorithm updates the estimate at node m as [5]:

$$\boldsymbol{\psi}_m^t = \boldsymbol{\psi}_{m-1}^t + a_m (\boldsymbol{x}_m^t)^* [d_m^t - \boldsymbol{x}_m^t \boldsymbol{\psi}_{m-1}^t] \tag{2.4}$$

where a_m is a constant of step size. The incremental strategy is briefly illustrated in Figure 2.2.

2.2.2 Diffusion Strategy

Different from incremental strategy which obtains information from neighbor nodes, each node in the diffusion network has some linked neighbors. Two diffusion estimation strategies are presented: Adapt-then-Combine (ATC) strategy and Combine-then-Adapt (CTA) strategy [29]. In the ATC strategy, optimization algorithms are used in each node to obtain a local estimate of

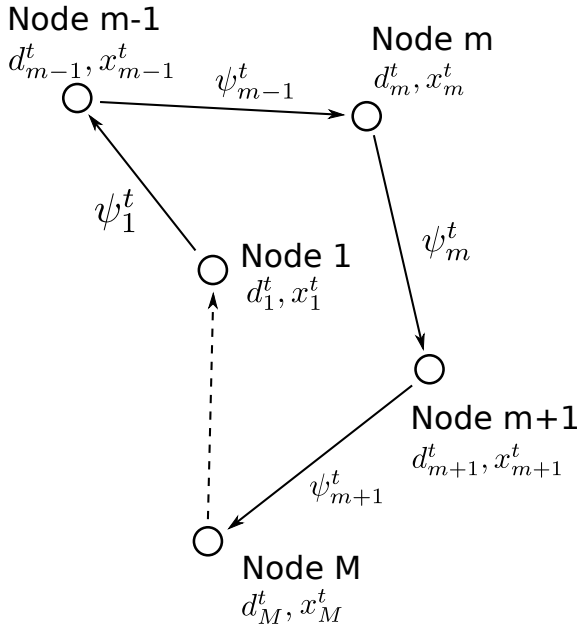


Figure 2.2: Incremental Strategy

$\boldsymbol{\psi}^t_m$. Then, each nodes collect the estimates from neighbor nodes and then combine them through

$$\boldsymbol{\omega}^{t+1}_m = \sum_{n \in \mathcal{N}_m} b^{t+1}_{n,m} \boldsymbol{\psi}^{t+1}_n, \tag{2.5}$$

where $b_{n,m}$ is the combination coefficients and calculated through the Metropolis rules, the Laplacian or the nearest neighbor rules [30], n is the neighbour node n linked to node m and $\mathcal{N}_m$ is the set of neighbors node m. The Metropolis rule can be implemented as [31]

$$b_{m,n} = \begin{cases} 1/(\max\{|\mathcal{N}_m|, |\mathcal{N}_n|\}), & n \in \mathcal{N}_m \quad and \;\; n \neq m \\ 1 - \sum_{k \in \mathcal{N}_m \setminus \{m\}} b_{n,m}, & m = n \\ 0, & \text{otherwise.} \end{cases} \tag{2.6}$$

where$|\mathcal{N}_m|$ is the cardinality of $\mathcal{N}_m$. The weight matrix by Laplacian rule is given by [30]

$$\mathbf{B} = \mathbf{I}_M - \beta \mathbf{L} \tag{2.7}$$

where $\mathbf{I}_M$ denotes a $M \times M$ identity matrix and $\mathbf{B}$ is the $M \times M$ matrix of combining coefficient, with entries $\{b_{m,n}\}$ and $\beta = 1/|\mathcal{N}_{max}|$. The Laplacian matrix is defined as

$$\mathbf{L} = \mathbf{D} - \mathbf{A} \tag{2.8}$$

with $\mathbf{D} = \text{diag}(\mathbf{A}\mathbf{1}_M)$ denoting the degree matrix, and $\mathbf{A}$ being the $M \times M$ network adjacent matrix as

$$\mathbf{A}_{m,n} = \begin{cases} 1, & \{m,n\} \in \epsilon \\ 0, & \text{otherwise} \end{cases} \tag{2.9}$$

where ϵ is the edge of network. The combining coefficients $b_{m,n}$ satisfies

$$\sum_{n \in \mathcal{N}_m} b_{m,n} = 1 \tag{2.10}$$

The ATC diffusion strategy is described in Figure 2.3(a). Based on the LMS algorithm, the diffusion LMS ATC strategy is performed as follows:

$$\boldsymbol{\psi}_m^{t+1} = \boldsymbol{\omega}_m^t + a_m(\boldsymbol{x}_m^t)^*[d_m^t - \boldsymbol{x}_m^t \boldsymbol{\omega}_m^t], \tag{2.11}$$

$$\boldsymbol{\omega}_m^{t+1} = \sum_{n \in \mathcal{N}_m} b_{n,m} \boldsymbol{\psi}_n^{t+1}, \tag{2.12}$$

The CTA strategy can operate in a reverse way. By collecting the estimates of their neighbors in previous time slot, and combining them together through

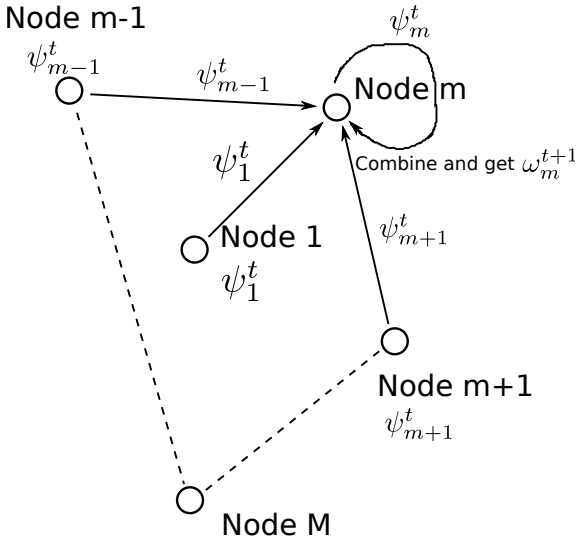
$$\boldsymbol{\psi}_m^t = \sum_{n \in \mathcal{N}_m} b_{n,m} \boldsymbol{\omega}_n^t, \tag{2.13}$$

After $\boldsymbol{\psi}_m^t$ is generated, $\boldsymbol{\psi}_m^t$ is employed at node m together with d_m^t and $(\boldsymbol{x}_m^t)$, to generate $\boldsymbol{\omega}_m^{t+1}$. Based on the LMS algorithm, the diffusion LMS CTA strategy is given by:

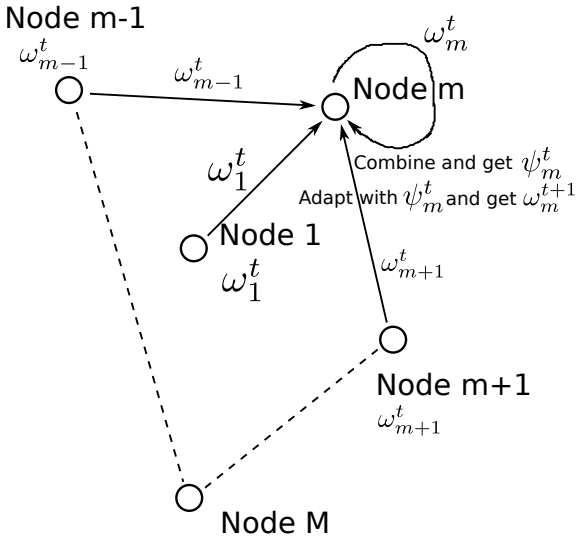
$$\boldsymbol{\psi}_m^t = \sum_{n \in \mathcal{N}_m} b_{n,m} \boldsymbol{\omega}_n^t, \tag{2.14}$$

$$\boldsymbol{\omega}_m^{t+1} = \boldsymbol{\psi}_m^t + a_m(\boldsymbol{x}_m^t)^*[d_m^t - \boldsymbol{x}_m^t \boldsymbol{\psi}_m^t], \tag{2.15}$$

The CTA process is described in Figure 2.3(b).



(a) ATC Strategy



(b) CTA Strategy

Figure 2.3: Diffusion Strategies

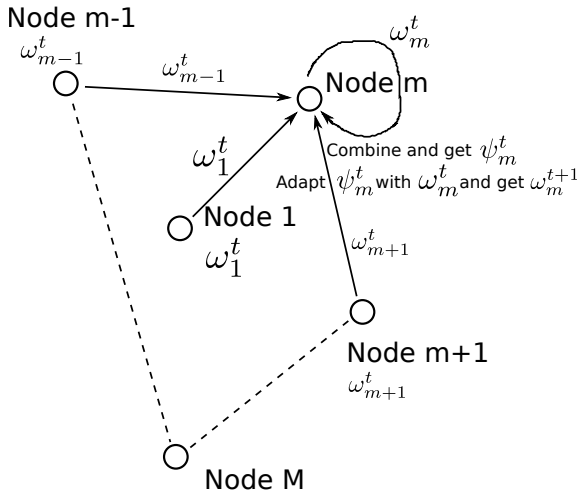


Figure 2.4: Consensus Strategy

2.2.3 Consensus strategy

As shown in Figure 2.4, each node in the consensus strategy collects the previous estimate from its neighbours, and combines them together through the Metropolis rules and the Laplacian matrix to generate $\boldsymbol{\psi}_m^t$. Each node can update the local estimate of $\boldsymbol{\omega}_m^{t+1}$ through adaptive algorithms with the estimate of $\boldsymbol{\psi}_m^t$ and its local estimate of $\boldsymbol{\omega}_m^t$.

According to the traditional LMS algorithm, the LMS consensus strategy [32] is given as:

$$\boldsymbol{\psi}_m^t = \sum_{n \in \mathcal{N}_m} b_{n,m} \boldsymbol{\omega}_n^t, \tag{2.16}$$

$$\boldsymbol{\omega}_m^{t+1} = \boldsymbol{\psi}_m^t + a_m (\boldsymbol{x}_m^t)^* [d_m^t - \boldsymbol{x}_m^t \boldsymbol{\omega}_m^t], \tag{2.17}$$

Gossip algorithm has been well investigated for network processing as a solution to consensus strategies [33–35]. A consistent agreement can be achieved among sensor nodes by exchanging information locally. Gossip algorithms can be classified as synchronous and asynchronous. In synchronous gossip algorithms [31], node m collects information from all of its neighbours, and combines with its own information to update the estimates at each iteration. In a randomized network setting, asynchronous gossip algorithms are assumed as the Poisson random process in [35,36]. In this model, each node has its own Poisson clock. When node ms clock ticks, it activates

and randomly selects one neighbour node to pair with, and then performs an averaging between the pairwised nodes [34]. Gossip algorithms for consensus problem have also been extended to the power systems [12], we will discuss an application of synchronous gossip algorithms for electric power system in Chapter 5.

2.2.4 Comparison of Diffusion and Consensus Strategies

To simplify the comparison of consensus, ATC diffusion, and CTA diffusion strategies, the recursions for each strategies can be rewritten as [4]:

Consensus:

$$\boldsymbol{\omega}_m^{t+1} = \sum_{n \in \mathcal{N}_m} b_{n,m} \boldsymbol{\omega}_n^t + a_m (\boldsymbol{x}_m^t)^* [d_m^t - \boldsymbol{x}_m^t \boldsymbol{\omega}_m^t], \tag{2.18}$$

ATC Diffusion:

$$\boldsymbol{\omega}_m^{t+1} = \sum_{n \in \mathcal{N}_m} b_{n,m} \Big(\boldsymbol{\omega}_n^t + a_n (\boldsymbol{x}_n^t)^* [d_n^t - x_n^t \boldsymbol{\omega}_n^t] \Big), \tag{2.19}$$

CTA Diffusion:

$$\boldsymbol{\omega}_m^{t+1} = \sum_{n \in \mathcal{N}_m} b_{n,m} \boldsymbol{\omega}_n^t + a_m (\boldsymbol{x}_m^t)^* \Big[d_m^t - \boldsymbol{x}_m^t \Big(\sum_{n \in \mathcal{N}_m} b_{n,m} \boldsymbol{\omega}_n^t \Big) \Big], \tag{2.20}$$

The weight-error vectors in consensus and diffusion networks are affected by the different orders of computations. Also, extra information can be included into the processing chain by the diffusion strategies introducing no extra computational complexity, compared to the consensus strategy. As proven in [4, 26], the diffusion strategy can provide a better performance than the consensus strategy, with the same computational load and data throughput, as shown in Table 2.1. The diffusion strategies introduce one intermediate variable used in subsequent computations. As shown in [4, 26], the computing orders have impact on the steady state performance of different strategies.

Table 2.1: [4] For the node m, the numbers of the complex multiplications, additions, and exchanged $N \times 1$ vectors within each iteration are compared. n_m is the degree of node k which indicates the size of its neighborhood $\mathcal{N}_m$.

	ATC diffusion	**CTA diffusion**	**Consensus**
Multiplications	$(n_m+2)M$	$(n_m+2)M$	$(n_m+2)M$
Additions	$(n_m+1)M$	$(n_m+1)M$	$(n_m+1)M$
Vectorexchanges	n_m	n_m	n_m

2.2.5 Asynchronous Diffusion Strategy

A perfect synchronous manner is assumed in the discussion of all the strategies in the WSNs. However, this is not always practical, as the measurement data might not arrive timely. Also, sensor nodes might randomly turn on and off to save energy. Since there may be failure of communication links between nodes, distributed solutions are not allowed to work properly. Therefore, the diffusion strategy is considered under the imperfections.

There have been a number of investigations on the consensus and gossip strategies under the asynchronous scheme [34,35,37]. Some methods focus on changing topologies [38–41]. However, only few studies exist for diffusion strategies [9–11,42,43]. Compared with diffusion scheme, the early studies focused on the averaging algorithms which did not deal with streaming data. These can lead to issues when data is flowing in as the noise always exists and the adaptation will be shut down by the use of diminish step-size. In this thesis, we only investigate the asynchronous ATC strategy [9–11]. To present the ATC procedure in an asynchronous network, the asynchronous ATC strategy can be modified as follows:

$$\boldsymbol{\psi}_m^{t+1} = \boldsymbol{\omega}_m^t + a_m^{t+1}(\boldsymbol{x}_m^t)^*[d_m^t - \boldsymbol{x}_m^t\boldsymbol{\omega}_m^t], \tag{2.21}$$

$$\boldsymbol{\omega}_m^{t+1} = \sum_{n\in\mathcal{N}_m^{t+1}} b_{n,m}^{t+1}\boldsymbol{\psi}_n^{t+1}, \tag{2.22}$$

where the $a_m^{t+1}, b_{n,m}^{t+1}$ are random step-sizes and combination coefficients with time, and $\mathcal{N}_m^{t+1}$ stands for the random neighbours of node m at time $t+1$. The a_m^{t+1} and $b_{n,m}^{t+1}$ are non-negative and random to control the step-size and combination coefficients respectively. In particular, $b_{n,m}^{t+1}$

needs to follow:

$$b_{m,n}^{t+1} = \begin{cases} > 0, & n \in \mathcal{N}_m \\ 0, & \text{otherwise.} \end{cases} \tag{2.23}$$

The asynchronous strategy (2.21) and (2.22) is capable of dealing with most scenarios in practice.

2.3 Optimization Algorithms

So far, a number of optimization methods have been developed for distributed networks. In this section, a few of related optimization algorithms are introduced.

2.3.1 Expectation-Maximization Algorithm

The expectation-maximization (EM) algorithm was introduced in 1977 by Dempster in [44], and is a well-developed method to provide the solution to problems of maximum likelihood (ML) estimation. An important aspect of the EM algorithm is the cost function with optimized likelihood, comprising of observed $\boldsymbol{y}$ data and unobserved $\boldsymbol{z}$ data. The unobserved data can be included, as missing data might be in the practical application or they are required for the likelihood computation. There a number of two steps for The EM procedure: Expectation (E-) and Maximization (M-) step. In the E-step, the likelihood estimation is calculated by using the observed data and the ML estimates, while the likelihood function data is maximized to refine the estimate of parameters in the M-step. In this thesis, we assume that each sensor in a WSN senses an environment that can be described as a mixture of components, these measurements of each sensor can be modeled with a mixture of Gaussian components, thus, we can present the procedure of EM algorithm for Gaussian mixture model as follows.

Let $\mathcal{P}(\boldsymbol{y}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denote the evaluation of a Gaussian density with at the data sample $\boldsymbol{y} = \{\boldsymbol{y}_1, \cdots, \boldsymbol{y}_N\}$, $\boldsymbol{y}$ is a N-dimensional continuous-valued data vector. The measurements are

assumed to obey a Gaussian mixture distribution with J components

$$\mathcal{P}(\boldsymbol{y}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{2\pi^{d/2}|\Sigma|^{1/2}}\exp\Big\{-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{y}-\boldsymbol{\mu})\Big\} \tag{2.24}$$

and

$$\boldsymbol{y}_i \sim \sum_{j=1}^{J}\alpha_j\mathcal{P}(\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j),\ i=1,\cdots,N \tag{2.25}$$

where α_j is mixing probability and $\mathcal{P}(\boldsymbol{\mu},\boldsymbol{\Sigma})$ denotes the Gaussian density function with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. The formulation of the mixture problem in the EM framework is achieved by augmenting the observed data vector $\boldsymbol{y}$ with the associated component-label vectors $\boldsymbol{z} = \{\boldsymbol{z}_i\}_{i=1}^N$. Each z_i takes on a value from the set $\{1,\ldots,J\}$, where $z_i = j$ indicates that y_i is generated by the jth mixture component

$$\boldsymbol{y}_i \sim \mathcal{P}(\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j). \tag{2.26}$$

The complete data log-likelihood $L_c(\boldsymbol{\theta})$ is then given by

$$\begin{aligned} L_c(\boldsymbol{\theta}) &= \log p(y,z|\boldsymbol{\theta}) \\ &= \sum_{i=1}^{N}\sum_{j=1}^{J} z_{i,j}(\log\alpha_j + \log\mathcal{P}(\boldsymbol{y}_i|\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)) \end{aligned} \tag{2.27}$$

where $p(y,z|\boldsymbol{\theta})$ denotes the joint density of y and z with parameter $\boldsymbol{\theta}$. Starting from an initial estimate $\boldsymbol{\theta}^0$, the standard EM algorithm alternates iteratively between the E- and M- step. In the E-step, given the current estimate $\boldsymbol{\theta}^t$, the conditional expectation of the complete data log-likelihood is computed as follows

$$\begin{aligned} Q(\boldsymbol{\theta};\boldsymbol{\theta}^t) &= E[L_c(\boldsymbol{\theta})|\boldsymbol{y},\boldsymbol{\theta}^t] \\ &= \sum_{i=1}^{N}\sum_{j=1}^{J} w_{i,j}^{t+1}(\log\alpha_j + \log\mathcal{P}(\boldsymbol{y}_i|\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)), \end{aligned} \tag{2.28}$$

where

$$w_{i,j}^{t+1} = \frac{\alpha_j^t\mathcal{P}(\boldsymbol{y}_i|\boldsymbol{\mu}_j^t,\boldsymbol{\Sigma}_j^t)}{\sum_{j=1}^{J}\alpha_j^t\mathcal{P}(\boldsymbol{y}_i|\boldsymbol{\mu}_j^t,\boldsymbol{\Sigma}_j^t)} \tag{2.29}$$

is the posterior probability that the ith sample belongs to the jth component given the observed value $\boldsymbol{y}_i$.

In the M-step, the parameters are computed by maximizing the complete data log-likelihood function (2.27)

$$\boldsymbol{\theta}^{t+1} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}; \boldsymbol{\theta}^t), \tag{2.30}$$

leading to the following update formula for $j = 1, \cdots, J$.

$$\alpha_j^{t+1} = \frac{1}{N}\sum_{i=1}^{N} w_{i,j}^{t+1}, \tag{2.31}$$

$$\boldsymbol{\mu}_j^{t+1} = \frac{\sum_{i=1}^{N} w_{i,j}^{t+1}\boldsymbol{y}_i}{\sum_{i=1}^{N} w_{i,j}^{t+1}}, \tag{2.32}$$

$$\boldsymbol{\Sigma}_j^{t+1} = \frac{\sum_{i=1}^{N} w_{i,j}^{t+1}(\boldsymbol{y}_i - \boldsymbol{\mu}_j^{t+1})(\boldsymbol{y}_i - \boldsymbol{\mu}_j^{t+1})^T}{\sum_{i=1}^{N} w_{i,j}^{t+1}} \tag{2.33}$$

The E-step and M-step repeat iteratively until converge to a local maximum likelihood.

2.3.2 Component-Wise EM Algorithm for Mixtures

Component-wise EM for Mixtures (CEMM) is one of EM-variant algorithms [8]. The mixture problem arises, when the sum of mixing proportions is equal to one. By defining an appropriate log-likelihood function, the Lagrangian dualization approach can change the initial problem into an unconstrained maximization. The CEMM algorithm is a natural coordinate-wise variant of EM algorithm. Considering mild regularity conditions, the EM algorithm can converge to a fixed point of the likelihood. The standard EM procedure updates all parameters simultaneously, which results in slow convergence in the presence of overlapping mixture densities. However, the parameters can be decoupled and updated by component-wise EM algorithm, achieving a faster convergence. For a Gaussian mixture model with J components, each iteration of CEMM consists of J cycles and the conditional expectation is computed and the parameter vector associated with jth component is updated at each cycle. The procedure of CEMM in a single iteration is presented as follow:

In the E-step, the conditional expectation is computed as:

$$w_{i,j}^{t+1} = \frac{\alpha_j^t \mathcal{P}(y_i|\boldsymbol{\mu}_j^t, \boldsymbol{\Sigma}_j^t)}{\sum_{k=1}^{j-1}\alpha_k^{t+1}\mathcal{P}(y_i|\boldsymbol{\mu}_k^{t+1}, \boldsymbol{\Sigma}_k^{t+1}) + \sum_{k=j}^{J}\alpha_k^t \mathcal{P}(y_i|\boldsymbol{\mu}_k^t, \boldsymbol{\Sigma}_k^t)}. \tag{2.34}$$

The M-step is then

$$\alpha_j^{t+1} = \frac{1}{N}\sum_{i=1}^{N} w_{i,j}^{t+1} , \tag{2.35}$$

$$\boldsymbol{\mu}_j^{t+1} = \frac{\boldsymbol{a}_j^{t+1}}{w_j^{t+1}} , \tag{2.36}$$

$$\boldsymbol{\Sigma}_j^{t+1} = \frac{\boldsymbol{b}_j^{t+1}}{w_j^{t+1}} - \boldsymbol{\mu}_j^{t+1}\boldsymbol{\mu}_j^{t+1'} , \tag{2.37}$$

As pointed out in [8], the decoupling of parameter updates means the use of the smallest admissible missing data space and provides a higher convergence rate than the standard EM algorithm. In Chapter 3, the application of CEMM will be introduced for a distributed WSN.

2.3.3 EM Gradient Algorithm

For a number of maximum likelihood based methods, it is impossible to perform the M-step within the EM algorithm . Solving the the M-step in the EM algorithm is realised with one iteration of Newton's method in the EM gradient algorithm. Thus, the EM gradient algorithm share some common local features with the EM algorithm. Similar with EM algorithm, the conditional expectation log-likelihood function with respect to complete data is calculated using (2.28) in the E-step of EM gradient algorithm. The current parameter column vector $\boldsymbol{\theta}^t$ can be updated using a single iteration of Newton's method in the M-step as

$$\begin{aligned} \boldsymbol{\theta}^{t+1} &= \boldsymbol{\theta}^t - \nabla^{20}Q(\boldsymbol{\theta}^t,\boldsymbol{\theta}^t)^{-1}\nabla^{10}Q(\boldsymbol{\theta}^t,\boldsymbol{\theta}^t), \\ &= \boldsymbol{\theta}^t - \nabla^{20}Q(\boldsymbol{\theta}^t,\boldsymbol{\theta}^t)^{-1}\nabla L(\boldsymbol{\theta}^t) \end{aligned} \tag{2.38}$$

where $\nabla^{20}Q(\boldsymbol{\theta}^t,\boldsymbol{\theta}^t)$ and $\nabla^{10}Q(\boldsymbol{\theta}^t,\boldsymbol{\theta}^t)$ denote the Hessian matrix and gradient vector of the conditional log-likelihood function $Q(\boldsymbol{\theta}^t,\boldsymbol{\theta}^t)$, respectively. When $L(\boldsymbol{\theta}) - Q(\boldsymbol{\theta},\boldsymbol{\theta}^t)$ has the minimum at $\boldsymbol{\theta} = \boldsymbol{\theta}^t$, there is the equality as $\nabla^{10}Q(\boldsymbol{\theta}^t,\boldsymbol{\theta}^t) = \nabla L(\boldsymbol{\theta}^t)$. The EM and EM gradient algorithms at the same convergence rate are attracted when using strict local maximum point of the observed likelihood. In Chapter 4, we will introduce an EM gradient based diffusion algorithm and its application to an asynchronous network.

2.3.4 Least Mean Square Method

Least-Mean-Square algorithm is a class of adaptive algorithm, developed from the MSE cost function [27, 45]:

$$J(\boldsymbol{\omega}) = E|d^t - \boldsymbol{x}^t\boldsymbol{\omega}^t|^2 \tag{2.39}$$

where d^t denotes the desired signal, $\boldsymbol{x}^t$ is the input signal and $\boldsymbol{\omega}^t$ denotes the weight vector. Then, the gradient vector of the cost function is given as

$$\frac{\partial J(\boldsymbol{\omega})}{\partial \boldsymbol{\omega}^t} = \left(\boldsymbol{\omega}^t\right)^* \mathbf{R}_x - \boldsymbol{b}_x \tag{2.40}$$

where $\boldsymbol{b}_x$ is the cross-correlation between the desired signal and the input signal and $\mathbf{R}_x$ is the input signals correlation matrix. The optimum solution to the cost function (2.39) is the Wiener solution, given by

$$\boldsymbol{\omega}^t = \mathbf{R}_x^{-1}\boldsymbol{b}_x \tag{2.41}$$

$\mathbf{R}_x$ and $\boldsymbol{b}_x$, as the statistics of the received signal, are not known in advance for the adaptive algorithms. Thus, these quantities need to be estimated. Instantaneous estimates for $\mathbf{R}_x$ and $\boldsymbol{b}_x$ [27, 45] is used in LMS algorithm, expressed as

$$\mathbf{R}_x = (\boldsymbol{x}^t)^*\boldsymbol{x}^t \tag{2.42}$$

$$\boldsymbol{b}_x = (d^t)^*\boldsymbol{x}^t \tag{2.43}$$

Consisting of (2.42) and (2.43) into (2.40) leads to

$$\frac{\partial J(\boldsymbol{\omega})}{\partial \boldsymbol{\omega}^t} = (\boldsymbol{\omega}^t)^*(\boldsymbol{x}^t)^*\boldsymbol{x}^t - (d^t)^*\boldsymbol{x}^t \tag{2.44}$$

The filter coefficient vector is updated by

$$\begin{aligned} \boldsymbol{\omega}^{t+1} &= \boldsymbol{\omega}^t - \alpha\left(\frac{\partial J(\boldsymbol{\omega})}{\partial \boldsymbol{\omega}^t}\right)^* \\ &= \boldsymbol{\omega}^t + \alpha(\boldsymbol{x}^t)^*[d^t - \boldsymbol{x}^t\boldsymbol{\omega}^t] \end{aligned} \tag{2.45}$$

where α is the step-size controlling the convergence speed.

2.4 Application in Power System State Estimation

In this section, the application of distributed estimation in power system are presented.Power system state estimation (PSSE) stands for acquiring the voltage phasors of all system buses at a fixed instant. This is realised by obtaining redundant measurements of power flows upon the network, and then implementing inferences to retrieve the values of the phasors. Previously, a centralized data processing center was used in PSSE to collect all the received data and retrieve a global solution [46, 47]. The decentralized estimation method can be developed to have higher estimation rate and better sensing ability. Large scale problems can be tackled by separating the observations and buses in distributed state estimation methods for power systems. Each separated control area senses and processes local data by itself, and communicate with other areas.

Distributed adaptive processing is now a powerful tool to perform the distributed state estimation for power systems. We consider an IEEE 14-bus system [48] which has 14 substations to demonstrate the use of distributed processing in power systems. The measurement model of the multi-agent state estimation can be expressed as:

$$\mathbf{z}_i = \mathbf{h}_i(\mathbf{x}) + \mathbf{e}_i \tag{2.46}$$

where $\mathbf{x}$ represents the state of the interconnected system, $\mathbf{h}_i(\mathbf{x})$ stands for nonlinear functions of admittance matrix, i is the bus number, $\mathbf{e}_i$ denotes the measurement error with zero means. Vector $\mathbf{x}$ denotes the state variable of the entire interconnected power system, including voltage magnitudes and voltage phase angles of all buses, which can be identified as the phase angle vector $\mathbf{x}_i$ for all buses. We can now approximate the measurement equation (2.47) with

$$\mathbf{z}_i = \mathbf{H}_i\mathbf{x}_i + \mathbf{e}_i \tag{2.47}$$

where $\mathbf{H}_i$ denotes the measurement Jacobian vector for bus i. The goal of distributed estimation method is to calculate an estimate of $\mathbf{x}_i$, which can minimize the cost function as

$$\hat{\mathbf{x}} = \min_{\mathbf{x}\in\mathbf{X}} \sum_{i=1}^{I} \|\mathbf{z}_i - \mathbf{H}_i\mathbf{x}_i\|^2 \tag{2.48}$$

To minimise the cost function (2.49), a number of distributed state estimation approaches have been proposed, such as M-CSE algorithm [6] and the alternating direction method of multipliers (ADMM) method [49], gossip based Gauss Newton (GGN) algorithm [12].

2.5 Chapter Summary

In this chapter, the characteristics and typical network topology of WSNs are introduced. By contrast with centralized processing, distributed version can reduce wireless bandwidth and energy consumption, as well as improve the robustness of network connections. Three main cooperation strategies on distributed estimation of parameters in WSNs are briefly presented. Meanwhile, the optimization algorithms aforementioned and the distributed methods for power system state estimation are presented in this chapter. In the following chapters, we will introduce these relevant optimization algorithms and their applications to distributed networks.

Chapter 3

Distributed Component-Wise EM Algorithm for Mixture Models in Sensor Networks

This chapter considers mixtures model estimation for sensor networks in a distributed manner. Based on the statistical literature, the maximum likelihood (ML) estimate of mixture distributions can be computed via a straightforward implementation of the expectation and maximization (EM) algorithm. In the sensor networks without centralized processing units, data are collected and processed locally. Modifications on standard EM-type algorithms are necessary to accommodate the characteristics of sensor networks. Existing works on the distributed EM algorithm mainly focus on the estimation performance and implementation aspects. In this chapter, we address the convergence issue by proposing a distributed EM-like algorithm that updates mixture parameters sequentially. Simulation results show that the proposed method leads to significant gain in convergence speed and considerable saving in computational overhead.

3.1 Introduction

Sensor networks are composed of enormous small devices with limited measuring, processing, and communication abilities. There has been a variety of environmental monitoring applications, e.g. the temperature sensing, automobile tracking, and cooperative information processing [50, 51]. As a powerful probabilistic modeling tool, the Gaussian Mixture Model (GMM) can be used for modeling density functions, for example, machine learning, pattern recognition and so on. The density estimation is essential especially in exploratory data analysis. For this purpose, the expectation-maximization (EM) methods are widely used [1].

The EM approach is well known to give ML approximations [52]. An expectation step (E-step) is performed in the EM method, and the likelihood expectation is calculated based on the observed latent variables. While the maximization step (M-step) is to maximize the expected likelihood to obtain the estimates of ML parameters. To start another E-step, the parameters on the M-step are required. The process is repeated a number of times until the convergence at a local maximum is reached. In the context of mixture models, it provides closed form solutions for estimating the means and covariance matrices of Gaussian components [53].

However, most EM algorithms are used in a centralized way for WSNs, which require the central process unit has the strong ability to process all the data. On the other hand, some parameters of interest can be estimated upon the network using the distributed estimation to leverage the local estimations and the links between nodes. Unlike the centralized strategy which processes all the information with a central node, the distributed estimation behaves differently and thus mitigates the computational load. Furthermore, the distributed estimation method is more robust against link failure [54, 55]. There have been a variety of distributed estimation approaches, such as diffusion Least mean squares (LMS) Strategies [56], distributed recursive least squares (RLS) method [57], distributed target source location [58], distributed power allocation and management approach [59], distributed sparse estimation [60, 61], distributed adaptive learning [62] and distributed Gaussian mixture density estimation [63].

The cooperation strategies among nodes have significant impact on WSNs within a distributed processing framework. The incremental and consensus strategies are widely used for distributed processing. The consensus strategy is discussed in [64, 65] which employs a slow time scale for sampling and a fast time scale for iterative operations. This strategy aims to derive the consistent estimates for all nodes. A distributed EM method for Gaussian mixtures using the consensus strategy is presented in [2], in which a consensus filter is introduced between the E- and M-steps. As the resources are constrained for WSNs communications [5], the application is limited for consensus-based methods with two time scales. Especially for a large scale WSN, massive computational burden will be brought in to achieve the consistency among the network nodes.

For the incremental strategy, the data flows in a pre-specific direction from one node to another node, which leads to the loop-type cooperations between nodes with minimum power and

communications. In this chapter, we first assume that individual observations over the environment can be modeled as the mixture of Gaussian distributions. In [1], this model was successfully applied to describe the data measured by sensor networks in a heterogeneous environment, such as temperature, air pressure, humidity, or light. Therein, a distributed (EM)-type algorithm was derived to identify Gaussian components common to the whole network and mixing probabilities associated with each node. Methods for improving the performance of distributed EM algorithms were suggested in [2, 66, 67].

In addition, increasing convergence speed is one of motivations for the DCEM method. The most documented problem associated with EM is its possibility of slow convergence. To speed up the convergence, various approaches have been proposed in the statistical literature [68, 69]. In [8], a component-wise EM algorithm was applied to mixture models. Instead of computing all parameters simultaneously in the M-step, the component-wise EM updates the component parameters sequentially. As the numerical results shown in [8], a better convergence rate can be achieved with this flexible approach. Another advantage of the component-wise EM is that despite the relaxation of the constraint on mixing probabilities, the sum of mixing probabilities equals to one when the algorithm converges.

To facilitate the application of the component-wise EM to sensor networks, we adopt the idea of incremental EM [53, 70] to enable local processing at sensor nodes. Note that such incremental strategies may not be suitable for large scale networks. Therefore, we assume a small enough network, typically less than 100 sensor nodes. As illustrated in the following sections, given sufficient statistics from the previous node, the E- and M-step at the current node involve only local observations. Simulation results showed that the proposed algorithm achieved a higher convergence rate than the distributed EM [1], leading to significant saving of overall computational time.

This chapter is organized as follows. The problem and data models is described in Section 3.2. Section 3.3 includes a brief description of the standard EM and distributed EM algorithms. The distributed component-wise EM algorithm for sensor networks is developed in Section 3.4. Section 3.5 presents an analysis of the convergence rate of the DCEM algorithm, Section 3.6 discusses simulation results and presents the performance of the proposed algorithm. Concluding

remarks is given in Section 3.7.

3.2 Problem Formulation

Consider a sensor network consisting of M sensor nodes. The mth node records N_m independent and identically distributed data $\boldsymbol{y}_m = \{\boldsymbol{y}_{m,1}, \cdots, \boldsymbol{y}_{m,N_m}\}$. The measurements are assumed to obey a Gaussian mixture distribution

$$\boldsymbol{y}_{m,i} \sim \sum_{j=1}^{J} \alpha_{m,j} \mathcal{P}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j),\ i = 1, \cdots, N_m \tag{3.1}$$

where $\mathcal{P}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes the Gaussian density function with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. The mixing parameters $\boldsymbol{\alpha}_j = \{\alpha_{m,j}\}_{m=1}^{M}$ are likely to be unique at each node, while the J mixing components $\mathcal{P}\left(\boldsymbol{\mu}_j, \boldsymbol{\Sigma} j\right)$ are common to all nodes. Let $\boldsymbol{\theta}_j = \{\boldsymbol{\alpha}_j, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\}_{j=1}^{J}$. Then the unknown parameter set is given as $\boldsymbol{\theta} = \{\boldsymbol{\theta}_j\}_{j=1}^{J}$. Based on the measurements $\boldsymbol{y} = \{\boldsymbol{y}_m\}_{m=1}^{M}$, the task is to compute the maximum likelihood (ML) estimate for $\boldsymbol{\theta}$ in a distributed manner.

Let $\mathcal{P}(\boldsymbol{y}_m|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denote the evaluation of a Gaussian density and the data sample $\boldsymbol{y}_m$ is given by

$$\mathcal{P}(\boldsymbol{y}_m|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{2\pi^{d/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\Big\{ -\frac{1}{2}(\boldsymbol{y}_m - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{y}_m - \boldsymbol{\mu}) \Big\} \tag{3.2}$$

It is well known that maximization of the log-likelihood for the mixture model in (3.1) [1]

$$L(\boldsymbol{\theta}) = \sum_{m=1}^{M} \sum_{i=1}^{N_m} \log \left(\sum_{j=1}^{J} \alpha_{m,j} \mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \right) \tag{3.3}$$

is greatly simplified by the EM-type algorithms [53] which are described in the following section. This data model is assumed to be statistically independent for each node. However, if the data are (spatially or temporally) correlated, this model is still valid by interpreting it as a $pseudolikelihood$ [71].

3.3 Standard EM and Distributed EM Algorithms (DEM)

The formulation of the mixture problem in the EM framework is achieved by augmenting the observed data vector $\boldsymbol{y} = \{\boldsymbol{y}_m\}_{m=1}^{M}$ with the associated component-label vectors $\boldsymbol{z} = \{\boldsymbol{z}_m\}_{m=1}^{M}$ where $\boldsymbol{z}_m = \{z_{m,i}\}_{i=1}^{N_m}$. Each $z_{m,i}$ takes on a value from the set $\{1, \ldots, J\}$, where $z_{m,i} = j$ indicates that $y_{m,i}$ is generated by the jth mixture component

$$\boldsymbol{y}_{m,i} \sim \mathcal{P}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j). \tag{3.4}$$

The complete data log-likelihood $L_c(\boldsymbol{\theta})$ is then given by

$$\begin{aligned} L_c(\boldsymbol{\theta}) &= \log p(y, z|\boldsymbol{\theta}) \\ &= \sum_{m=1}^{M}\sum_{i=1}^{N_m}\sum_{j=1}^{J} z_{m,i,j}(\log \alpha_{m,j} + \log \mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)) \end{aligned} \tag{3.5}$$

where $p(y, z|\boldsymbol{\theta})$ denotes the joint density of y and z with parameter $\boldsymbol{\theta}$. Starting from an initial estimate $\boldsymbol{\theta}^0$, the standard EM algorithm iterates between the E (expectation) and M (maximization) steps. In the E-step, given the current estimate $\boldsymbol{\theta}^t$, the conditional expectation of the complete data log-likelihood is computed as follows

$$\begin{aligned} Q(\boldsymbol{\theta}; \boldsymbol{\theta}^t) &= E[L_c(\boldsymbol{\theta})|\boldsymbol{y}, \boldsymbol{\theta}^t] \\ &= \sum_{m=1}^{M}\sum_{i=1}^{N_m}\sum_{j=1}^{J} w_{m,i,j}^{t+1}(\log \alpha_{m,j} + \log \mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)), \end{aligned} \tag{3.6}$$

where

$$w_{m,i,j}^{t+1} = \frac{\alpha_{m,j}^t \mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_j^t, \boldsymbol{\Sigma}_j^t)}{\sum_{j=1}^{J} \alpha_{m,j}^t \mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_j^t, \boldsymbol{\Sigma}_j^t)} \tag{3.7}$$

is the posterior probability that the ith sample at node m belongs to the jth component given the observed value $\boldsymbol{y}_{m,i}$.

In the M-step, the parameters are computed by maximizing the complete data log-likelihood function (3.6)

$$\boldsymbol{\theta}^{t+1} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}; \boldsymbol{\theta}^t), \tag{3.8}$$

which lead to the following update formula for $j = 1, \cdots, J$.

$$\alpha_{m,j}^{t+1} = \frac{1}{N_m} \sum_{i=1}^{N_m} w_{m,i,j}^{t+1},\ m = 1, \cdots, M \tag{3.9}$$

$$\boldsymbol{\mu}_j^{t+1} = \frac{\sum_{m=1}^{M} \sum_{i=1}^{N_m} w_{m,i,j}^{t+1} \boldsymbol{y}_{m,i}}{\sum_{m=1}^{M} \sum_{i=1}^{N_m} w_{m,i,j}^{t+1}}, \tag{3.10}$$

$$\boldsymbol{\Sigma}_j^{t+1} = \frac{\sum_{m=1}^{M} \sum_{i=1}^{N_m} w_{m,i,j}^{t+1} (\boldsymbol{y}_{m,i} - \boldsymbol{\mu}_j^{t+1})(\boldsymbol{y}_{m,i} - \boldsymbol{\mu}_j^{t+1})^T}{\sum_{m=1}^{M} \sum_{i=1}^{N_m} w_{m,i,j}^{t+1}} \tag{3.11}$$

The update formula in (3.9)-(3.11) can further be written as:

$$\boldsymbol{\mu}_j^{t+1} = \frac{\sum_{m=1}^{M} \boldsymbol{a}_{m,j}^{t+1}}{\sum_{m=1}^{M} w_{m,j}^{t+1}}, \tag{3.12}$$

$$\boldsymbol{\Sigma}_j^{t+1} = \frac{\sum_{m=1}^{M} \boldsymbol{b}_{m,j}^{t+1}}{\sum_{m=1}^{M} w_{m,j}^{t+1}} \tag{3.13}$$

in which the local summary quantities are denoted as

$$w_{m,j}^{t+1} = \sum_{i=1}^{N_m} w_{m,i,j}^{t+1}, \tag{3.14}$$

$$\boldsymbol{a}_{m,j}^{t+1} = \sum_{i=1}^{N_m} w_{m,i,j}^{t+1} \boldsymbol{y}_{m,i}, \tag{3.15}$$

$$\boldsymbol{b}_{m,j}^{t+1} = \sum_{i=1}^{N_m} w_{m,i,j}^{t+1} \boldsymbol{y}_{m,i} \boldsymbol{y}_{m,i}^T. \tag{3.16}$$

Thus, the global summary quantities are

$$w_j^{t+1} = \sum_{m=1}^{M} \sum_{i=1}^{N_m} w_{m,i,j}^{t+1}, \tag{3.17}$$

$$\boldsymbol{a}_j^{t+1} = \sum_{m=1}^{M} \sum_{i=1}^{N_m} w_{m,i,j}^{t+1} \boldsymbol{y}_{m,i}, \tag{3.18}$$

$$\boldsymbol{b}_j^{t+1} = \sum_{m=1}^{M} \sum_{i=1}^{N_m} w_{m,i,j}^{t+1} \boldsymbol{y}_{m,i} \boldsymbol{y}_{m,i}^T. \tag{3.19}$$

Notice that with these summaries defined as previous, the estimated parameters are

$$\boldsymbol{\mu}_j^{t+1} = \frac{\boldsymbol{a}_j^{t+1}}{w_j^{t+1}}, \tag{3.20}$$

$$\boldsymbol{\Sigma}_j^{t+1} = \frac{\boldsymbol{b}_j^{t+1}}{w_j^{t+1}} - \boldsymbol{\mu}_j^{t+1} (\boldsymbol{\mu}_j^{t+1})^T, \tag{3.21}$$

The E- and M-steps are alternated repeatedly until the difference between likelihoods of consecutive iterates $L(\boldsymbol{\theta}^{t+1}) - L(\boldsymbol{\theta}^t)$ is less than a pre-specified small number ϵ. Note that, given the current parameter $\boldsymbol{\theta}^t$ and local data set $\boldsymbol{y}_{m,i}$, the local summary quantities $\{w_{m,j}^{t+1}, \boldsymbol{a}_{m,j}^{t+1}, \boldsymbol{b}_{m,j}^{t+1}\}$ can be locally computed. Thus, several distributed methods [1, 2] are valid to apply the standard EM algorithm to such a WSN.

3.3.1 Distributed EM Algorithm based on Incremental Strategy

A distributed EM algorithm based on the incremental strategy for sensor network (DEM) was studied in [1]. With such a network setting, the communication path is cyclic and pre-set. Only one node update the parameter set $\boldsymbol{\theta}^{t+1}$ using its own N_m observations at each iteration, given the current parameter set $\boldsymbol{\theta}^t$. In details, node m can update the global summary quantities by using its new local summary quantities to replace the old quantities based on

$$w_j^{t+1} = w_j^t + w_{m,j}^{t+1} - w_{m,j}^t \,, \tag{3.22}$$

$$\boldsymbol{a}_j^{t+1} = \boldsymbol{a}_j^t + \boldsymbol{a}_{m,j}^{t+1} - \boldsymbol{a}_{m,j}^t \,, \tag{3.23}$$

$$\boldsymbol{b}_j^{t+1} = \boldsymbol{b}_j^t + \boldsymbol{b}_{m,j}^{t+1} - \boldsymbol{b}_{m,j}^t \,. \tag{3.24}$$

and updates the parameter set $\boldsymbol{\theta}^{t+1}$ according to (3.20) and (3.21). During this procedure, other nodes are fixed. Then, node m passes the message of updated global summary quantities $\{w_j^{t+1}, \boldsymbol{a}_j^{t+1}, \boldsymbol{b}_j^{t+1}\}$ and the estimated parameter $\boldsymbol{\theta}^{t+1}$ to the next adjacent $(m+1)$ node, and this process is repeatedly.

Note that each node only executes a single and local E- and M- step in the DEM algorithm, thus this algorithm does not require the updated means and covariances $\{\boldsymbol{\mu}_j^{t+1}, \boldsymbol{\Sigma}_j^{t+1}\}$ to reach a fixed point at each local E-step process. In order to speed up the overall convergence, DEMM algorithm refers to DEM with multiple steps at each node which was studied in [1]. Specifically, the local E- and M- steps can be repeated several times in succession until the maximization of the local log-likelihood function is found, then the updated message can be passed to next node.

3.3.2 Consensus based EM Algorithm

In [2], another distributed EM algorithm based on consensus strategy is proposed. The main idea behind this technique is the application of average consensus filter between E- step and M- step. For consensus strategies, the nodes communicate with their neighbours until network agreement is achieved. The local summary quantities of individual nodes are updated using the local N_m observations at E- step, and then the local statistics are exchanged via consensus filters. The node can access to the global summary quantities until the final consensus:

$$\alpha_j^{t+1} = \frac{1}{N_m M} \sum_{m=1}^{M} \sum_{i=1}^{N_m} w_{m,i,j}^{t+1},\ m = 1, \cdots, M \tag{3.25}$$

$$\boldsymbol{\mu}_j^{t+1} = \frac{\frac{1}{M} \sum_{m=1}^{M} \boldsymbol{a}_{m,j}^{t+1}}{\frac{1}{M} \sum_{m=1}^{M} w_{m,j}^{t+1}}, \tag{3.26}$$

$$\boldsymbol{\Sigma}_j^{t+1} = \frac{\frac{1}{M} \sum_{m=1}^{M} \boldsymbol{b}_{m,j}^{t+1}}{\frac{1}{M} \sum_{m=1}^{M} w_{m,j}^{t+1}} \tag{3.27}$$

All these algorithms require to execute the standard E- and M- step to update parameters simultaneously. They are often effective when the mixtures are well-separated. They suffered from a slow convergence when the mixtures become complex or overlapping. To speed up the convergence of the standard EM algorithm, a component-wise EM method for mixture models (CEMM) was presented in [8]. Rather than computing all parameters simultaneously, the CEMM algorithm considers the decomposition of the parameter vector $\boldsymbol{\theta}$ into component parameter vectors $\{\boldsymbol{\alpha}_j, \boldsymbol{\theta}_j\},\ j = 1, \cdots, J$ and updates only one component at a time. Specifically, each iteration consists of J cycles in which the conditional expectation (3.6) and the parameter vector associated with jth component are updated. As pointed out in [8], the decoupling of parameter updates implies the use of the smallest admissible missing data space and leads to faster convergence than the standard EM algorithm.

3.4 Distributed Component-wise EM Algorithm

Motivated by the superior convergence behavior of the component-wise EM algorithm, we propose a distributed component-wise EM algorithm for mixtures in sensor networks. In some sensor

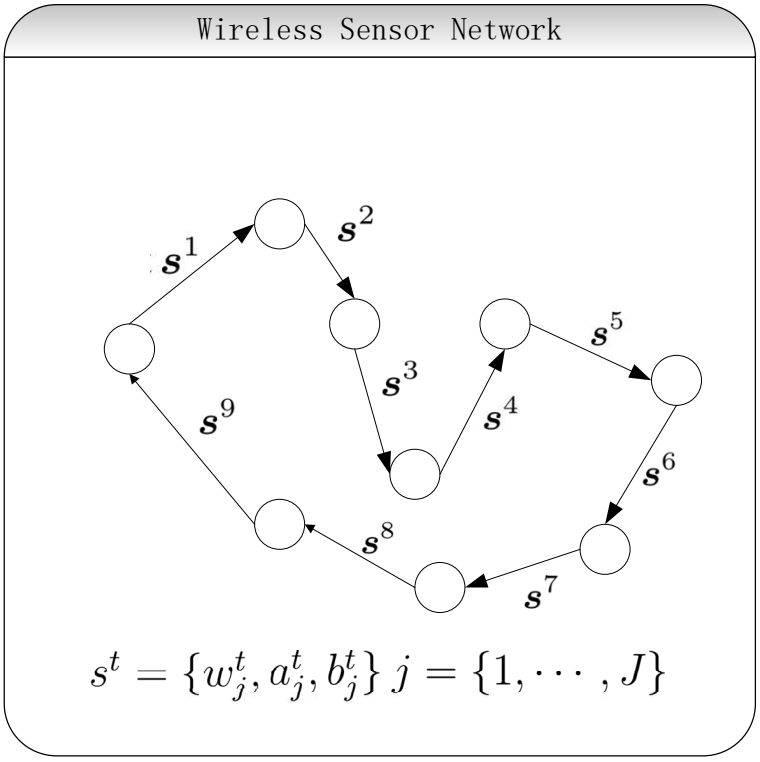


Figure 3.1: Communication/iteration cycle in a sensor network

network models, a high-performance centralized unit is involved to solve the estimation problems. But relying on the centralized unit is undesirable in scenarios in which communications between sensor nodes are much more costly than the computational cost at sensor nodes. In the following, we consider the message passage model for sensor networks as depicted in Figure 3.1 Similar to the distributed EM algorithm, our algorithm also exploits the idea of incremental EM [70] to facilitate local processing. The idea behind incremental EM is to divide the observed data into several blocks and implement the E-step for only a block of observations at a time before performing a M-step [53]. Here, the observed data at each node is considered as one data block. By applying the incremental EM, the component-wise EM can be implemented so that at node m, given the summary quantities (3.17), (3.18) and (3.19) from the previous node $(m-1)$, only local data $\boldsymbol{y}_m$ is involved.

Let $\boldsymbol{a}_j^t, \boldsymbol{b}_j^t, w_j^t$ be the received summary statistics of the $m-$th node from the previous one, and the local estimates after the $t-$th iteration be

$$\boldsymbol{\theta}_m^t = \{\boldsymbol{\theta}_{m,1}^t, \cdots, \boldsymbol{\theta}_{m,J}^t\}, \tag{3.28}$$

where $\boldsymbol{\theta}_{m,j}^t$ include the estimate for the jth component $\{\boldsymbol{\alpha}_{m,j}^t, \boldsymbol{\mu}_{m,j}^t, \boldsymbol{\Sigma}_{m,j}^t\}$. At the beginning of the $(t+1)$th iteration, the initial estimates for the mean and covariance matrix are obtained from the summary statistics as follows:

$$\boldsymbol{\mu}_{m,j}^t = \frac{\boldsymbol{a}_j^t}{w_j^t}, \quad \boldsymbol{\Sigma}_{m,j}^t = \frac{\boldsymbol{b}_j^t}{w_j^t} - \boldsymbol{\mu}_j^t \boldsymbol{\mu}_j^{t'}, \quad j = 1, \cdots, J. \tag{3.29}$$

Let $\boldsymbol{\theta}_m^{[t+1,0]} = \boldsymbol{\theta}_m^t$. The parameters associated with the jth components $\boldsymbol{\theta}_{m,j}^t$ are updated sequentially in the proposed algorithm as follows.

For $j = 1, \cdots, J$, the E-step is computed as:

$$w_{m,i,j}^{t+1} = \frac{\alpha_{m,j}^t \mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_{m,j}^t, \boldsymbol{\Sigma}_{m,j}^t)}{\sum_{k=1}^{j-1} \alpha_{m,k}^{t+1} \mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_{m,k}^{t+1}, \boldsymbol{\Sigma}_{m,k}^{t+1}) + \sum_{k=j}^{J} \alpha_{m,k}^t \mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_{m,k}^t, \boldsymbol{\Sigma}_{m,k}^t)}. \tag{3.30}$$

The M-step is then

$$\alpha_{m,j}^{t+1} = \frac{1}{N_m} \sum_{i=1}^{N_m} w_{m,i,j}^{t+1}, \tag{3.31}$$

$$\boldsymbol{\mu}_{m,j}^{t+1} = \frac{\boldsymbol{a}_{m,j}^{t+1}}{w_{m,j}^{t+1}}, \tag{3.32}$$

$$\boldsymbol{\Sigma}_{m,j}^{t+1} = \frac{\boldsymbol{b}_{m,j}^{t+1}}{w_{m,j}^{t+1}} - \boldsymbol{\mu}_{m,j}^{t+1} \boldsymbol{\mu}_{m,j}^{t+1'}, \tag{3.33}$$

where the local summary statistics $w_{m,j}, \boldsymbol{a}_{m,j}, \boldsymbol{b}_{m,j}$ are

$$w_{m,j}^{t+1} = \frac{1}{N_m} \sum_{i=1}^{N_m} w_{m,i,j}^{t+1}, \tag{3.34}$$

$$\boldsymbol{a}_{m,j}^{t+1} = \sum_{i=1}^{N_m} w_{m,i,j}^{t+1} \boldsymbol{y}_{m,i}, \tag{3.35}$$

$$\boldsymbol{b}_{m,j}^{t+1} = \sum_{i=1}^{N_m} w_{m,i,j}^{t+1} \boldsymbol{y}_{m,i} \boldsymbol{y}_{m,i}'. \tag{3.36}$$

The estimate at the jth cycle is given by

$$\boldsymbol{\theta}_m^{[t+1,j]} = \{\boldsymbol{\theta}_{m,1}^{t+1}, \cdots, \boldsymbol{\theta}_{m,j}^{t+1}, \boldsymbol{\theta}_{m,j+1}^t, \cdots, \boldsymbol{\theta}_{m,J}^t\}. \tag{3.37}$$

After J cycles, the output of the $(t+1)$th iteration is given by:

$$\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}_m^{[t+1,J]}. \tag{3.38}$$

Then the local summary statistics are computed with the new estimate $\boldsymbol{\theta}^{t+1}$ according to (3.22)-(3.24).

Note that the old values of summary statistics are replaced by the updated values at node m. In addition, the computations of the posterior probabilities (**??**) and the estimates (3.31), (3.32) and (3.33) involve only the data at node m.

The major difference of the proposed component-wise approach from the distributed EM algorithm is as follows. In the distributed EM algorithm (DEM) [1], the parameters associated with all components are updated simultaneously. The E-step is evaluated only once at the beginning of the iteration. In the proposed algorithm, every component parameter set $\boldsymbol{\theta}_j$ is computed sequentially and the posterior probability $w_{m,i,j}$ (**??**) is evaluated at each cycle. The computational time is only slightly increased by the multiple E-steps in compared to the distributed EM algorithm. Simulation results in the following sections will show that our approach leads to a much faster convergence of the log-likelihood than the distributed EM algorithm.

3.5 Convergence Analysis

In [72] and [73], the authors gave in-depth analysis on the convergence of standard EM algorithms. It is shown in [70] and [74] et al. that under standard regularity conditions, the incremental EM will give the estimates which converge with respect to the likelihood function, and the likelihood is iteratively ascending. The standard EM algorithm usually follows the linear convergence. The results in [72] and [75] help us to analyse the convergence behaviour of distributed component-wise EM in a Gaussian mixture model.

In [1], it is assumed that the $\{\boldsymbol{\theta}^t\}$ converges to $\boldsymbol{\theta}^*$ to maximize the log-likelihood $L(\boldsymbol{\theta})$. It can be shown that the estimate $\boldsymbol{\theta}^t$ near $\boldsymbol{\theta}^*$ with iterations has the following approximate relationship for

sufficiently large t

$$\boldsymbol{\theta}^{t+1} - \boldsymbol{\theta}^{*} = \mathbf{M}(\bar{\boldsymbol{\theta}}^{t} - \boldsymbol{\theta}^{*}) \tag{3.39}$$

where $\bar{\boldsymbol{\theta}}^{t}$ is described as a certain average of the past $\{\boldsymbol{\theta}^{(t-m)}\}_{m=1}^{M}$ and $\mathbf{M}$ is defined as the rate matrix of the algorithm. The convergence rate is determined by the spectral radius $\rho(\mathbf{M})$ of the rate matrix [76]. Based on the results [69], a larger $\rho(\mathbf{M})$ leads to a slower convergence speed.

Before analyzing the convergence of DCEM, we consider another analytical approach for the convergence of the DEM in [75]. In this method, we define an augmented vector including all nodes' as:

$$\boldsymbol{\Theta}^{t} = \begin{bmatrix} \boldsymbol{\theta}_1^{t} \\ \vdots \\ \boldsymbol{\theta}_M^{t} \end{bmatrix} \tag{3.40}$$

During each iteration of the DEM algorithm, only one node updates its parameters while other nodes' parameters are fixed: all parameters of $\boldsymbol{\theta}^{t}$ can be updated after a full cycle of the procedure. In addition, assuming that data sets are statistically independent at different nodes, the local objective function is calculated as:

$$L_m(\boldsymbol{\theta}_m) = \sum_{i=1}^{N_m} \log(\sum_{j=1}^{J} \alpha_{m,j} \mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_{mj}, \boldsymbol{\Sigma}_j)) \tag{3.41}$$

where $\boldsymbol{\theta}_m$ is the parameter vector for the $m-$th node. We model the conditional expectation of complete data using log-likelihood as:

$$\begin{aligned} Q_m(\boldsymbol{\theta}; \boldsymbol{\theta}_m^{t}) &= E[L_c(\boldsymbol{\theta})|\boldsymbol{y}_m, \boldsymbol{\theta}_m^{t}] \\ &= \sum_{i=1}^{N_m}\sum_{j=1}^{J} w_{m,i,j}^{t+1}(\log \alpha_{m,j} + \log \mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_{mj}, \boldsymbol{\Sigma}_j)) \end{aligned} \tag{3.42}$$

The total conditional Q function can be reformatted as [1]:

$$\begin{aligned} Q(\boldsymbol{\theta}; \boldsymbol{\theta}^{t}) &= Q(\boldsymbol{\theta}; \boldsymbol{\theta}_1^{t}, \ldots, \boldsymbol{\theta}_M^{t}) \\ &= \sum_{m=1}^{M} Q_m(\boldsymbol{\theta}; \boldsymbol{\theta}_m^{t}) \end{aligned} \tag{3.43}$$

Finally, the updated equation of the DEM algorithm for a WSN can be represented as:

$$\boldsymbol{\theta}^{t+1} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}; \boldsymbol{\theta}_1^t, \ldots, \boldsymbol{\theta}_M^t), \tag{3.44}$$

Although each iteration of the standard EM satisfies the property of $L(\boldsymbol{\theta}^{t+1}) \geq L(\boldsymbol{\theta}^t)$, this monotonicity property is not guaranteed in the DEM scheme. However, each step of the DEM method satisfies the monotonicity condition

$$Q(\boldsymbol{\theta}^{t+1}; \boldsymbol{\theta}_1^t, \ldots, \boldsymbol{\theta}_M^t) \geq Q(\boldsymbol{\theta}^t; \boldsymbol{\theta}_1^t, \ldots, \boldsymbol{\theta}_M^t) \tag{3.45}$$

This shows that the total conditional function Q is improved at each step. Using the Taylor expansion in the local Q function, it was verified in [75] that the local estimates $\boldsymbol{\theta}_m^t$ can achieve a local maximum at a fixed point $\boldsymbol{\theta}_m^*$ and satisfy the following approximate relationship for sufficiently large t

$$\boldsymbol{\theta}_m^{t+1} - \boldsymbol{\theta}_m^* = \mathbf{M}_m^{DEM}(\boldsymbol{\theta}_m^t - \boldsymbol{\theta}_m^*) \tag{3.46}$$

where $\mathbf{M}_m^{DEM}$ is the local rate matrix at node m and its expression is given by

$$\begin{aligned} \mathbf{M}_m^{DEM} &= \nabla^{11} Q_m(\boldsymbol{\theta}_m^*; \boldsymbol{\theta}_m^*)[\nabla^{20} Q_m(\boldsymbol{\theta}_m^*; \boldsymbol{\theta}_m^*)]^{-1} \\ &= \mathbf{I} - [\nabla^{20} D(\boldsymbol{\theta}_m^*; \boldsymbol{\theta}_m^*) + \nabla^2 L(\boldsymbol{\theta}_m^*)]^{-1} \nabla^2 L(\boldsymbol{\theta}_m^*) \end{aligned} \tag{3.47}$$

where ∇^{ij} denotes the ith order partial derivatives with respect to the first argument and jth order partial derivatives with respect to the second argument. $D(\boldsymbol{\theta}_m; \boldsymbol{\theta}_m^t) = E[\log p(y, z|y, \boldsymbol{\theta})|y, \boldsymbol{\theta}^t]$ is the distance between $\boldsymbol{\theta}_m$ and $\boldsymbol{\theta}_m^t$. It can be also shown that $\nabla^2 L(\boldsymbol{\theta}_m^*)$ and $D(\boldsymbol{\theta}_m; \boldsymbol{\theta}_m^t)$ are negative definite [52] and the eigenvalues of $\mathbf{M}_m^{DEM}$ all lie in $[0, 1)$. With the definition of (3.40), if $\boldsymbol{\Theta}^*$ is a fixed point of the DEM algorithm , the convergence rate of the full DEM procedure in sensor network setting can be formulated as:

$$\boldsymbol{\Theta}^{t+1} - \boldsymbol{\Theta}^* = \mathbf{M}^{DEM}(\boldsymbol{\Theta}^t - \boldsymbol{\Theta}^*) \tag{3.48}$$

where $\mathbf{M}^{DEM}$ is a block diagonal matrix defined as

$$\mathbf{M}^{DEM} = \begin{bmatrix} \mathbf{M}_1^{DEM} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \mathbf{M}_M^{DEM} \end{bmatrix} \tag{3.49}$$

Based on the definition of the spectrum radius:

$$\rho(\mathbf{M}) = \max|\boldsymbol{\beta}| \tag{3.50}$$

where $\boldsymbol{\beta}$ are the eigenvalues of $\mathbf{M}$, the convergence rate is the largest eigenvalue of $\mathbf{M}$. Therefore, if the maximum eigenvalue of $\mathbf{M}_m^{DEM}$ is denoted by $\boldsymbol{\beta}_m^{DEM}$, the convergence rate of all estimated parameters in the DEM algorithm after a full cycle will be equal to

$$\rho(\mathbf{M}^{DEM}) = \max_m |\boldsymbol{\beta}_m^{DEM}| < 1 \tag{3.51}$$

Now we consider the DCEM algorithm in a sensor network situation. In a DEM algorithm, the linear constraint for Gaussian mixtures at each node operation $\sum_{j=1}^{J} \alpha_{m,j} = 1$ is automatically satisfied during every E- and M- steps. This is obviously not satisfactory in the context of component-wise methods [8]. In [8], a Lagrangian approach is introduced to fulfill this constraint by reconstructing a modified likelihood function based on Lagrangian duality. Since the data collected at each sensor are independent of the data at other sensors, the local modified likelihood function is given by:

$$\mathcal{L}_m(\boldsymbol{\theta}_m, \lambda) = L_m(\boldsymbol{\theta}_m) - \lambda\big(\sum_{j=1}^{J} \alpha_{m,j} - 1\big) \tag{3.52}$$

From [8], we can get $\lambda = N_m$ by solving this Lagrangian function, thus, (3.52) becomes

$$\mathcal{L}_m(\boldsymbol{\theta}_m) = L_m(\boldsymbol{\theta}_m) - N_m\big(\sum_{j=1}^{J} \alpha_{m,j} - 1\big) \tag{3.53}$$

The convergence of the standard algorithm with Gaussian mixtures is investigated in [72] by linking the EM algorithm to gradient ascent methods. Motivated by this idea, we demonstrate that the E- and M- steps of the DCEM algorithm at each node can be realized by jointly using the

gradient and the projection matrices.

Theorem 1 *At the jth cycle, the local updates formulas (3.31)-(3.33) in DCEM at node m can be described as:*

$$\alpha_{m,j}^{t+1} - \alpha_{m,j}^{t} = \mathrm{P}_{\alpha_{m,j}^{t}} \frac{\partial \mathcal{L}_m(\boldsymbol{\theta}_m)}{\partial \alpha_{m,j}} |_{\mathcal{A}_m = \mathcal{A}_m^{[t+1,j-1]}}, \tag{3.54}$$

$$\boldsymbol{\mu}_{m,j}^{t+1} - \boldsymbol{\mu}_{m,j}^{t} = \mathrm{P}_{\boldsymbol{\mu}_{m,j}^{t}} \frac{\partial \mathcal{L}_m(\boldsymbol{\theta}_m)}{\partial \boldsymbol{\mu}_{m,j}} |_{\boldsymbol{\mu}_{m,j} = \boldsymbol{\mu}_{m,j}^{t}}, \tag{3.55}$$

$$vec[\boldsymbol{\Sigma}_{m,j}^{t+1}] - vec[\boldsymbol{\Sigma}_{m,j}^{t}] = \mathrm{P}_{\boldsymbol{\Sigma}_{m,j}^{t}} \frac{\partial \mathcal{L}_m(\boldsymbol{\theta}_m)}{\partial vec[\boldsymbol{\Sigma}_{m,j}]} |_{\boldsymbol{\Sigma}_{m,j} = \boldsymbol{\Sigma}_{m,j}^{t}}, \tag{3.56}$$

where vec$[C]$ denotes the vector obtained by stacking the column vectors of matrix C, $\mathcal{A}_m$ denotes the vector of mixing probabilities $[\alpha_{m,1}, \cdots, \alpha_{m,J}]^T$ at node m,

$$\mathcal{A}_m^{[t+1,j-1]} = [\alpha_{m,1}^{t+1}, \cdots, \alpha_{m,j-1}^{t+1}, \alpha_{m,j}^{t}, \cdots, \alpha_{m,J}^{t}]^T, \tag{3.57}$$

and

$$\mathrm{P}_{\alpha_{m,j}^{t}} = \frac{1}{N_m} \alpha_{m,j}^{t} \tag{3.58}$$

$$\mathrm{P}_{\boldsymbol{\mu}_{m,j}^{t}} = \frac{\boldsymbol{\Sigma}_{m,j}^{t}}{\sum_{i=1}^{N_m} w_{i,m,j}^{t+1}}, \tag{3.59}$$

$$\mathrm{P}_{\boldsymbol{\Sigma}_{m,j}^{t}} = \frac{2}{\sum_{i=1}^{N_m} w_{i,m,j}^{t+1}} \boldsymbol{\Sigma}_{m,j}^{t} \otimes \boldsymbol{\Sigma}_{m,j}^{t}, \tag{3.60}$$

where $\otimes$ is the Kronecker product.

Proof: See Appendix A

Using the notation $\boldsymbol{\theta}_{m,j} = \{\alpha_{m,j}, \boldsymbol{\mu}_{m,j}, \mathrm{vec}[\boldsymbol{\Sigma}_{m,j}]^T\}^T$ we define the local projection matrix as follow:

$$\mathbf{P}_{\boldsymbol{\theta}_{m,j}^{t}} = \begin{bmatrix} P_{\alpha_{m,j}^{t}} & & \mathbf{0} \\ & P_{\boldsymbol{\mu}_{j}^{t}} & \\ \mathbf{0} & & P_{\boldsymbol{\Sigma}_{j}^{t}} \end{bmatrix} \tag{3.61}$$

Then, the updates can be integrated into:

$$\boldsymbol{\theta}_{m,j}^{(t+1)} = \boldsymbol{\theta}_{m,j}^{t} + P_{\boldsymbol{\theta}_{m,j}}^{t} \frac{\partial \mathcal{L}_m(\boldsymbol{\theta}_m)}{\partial \boldsymbol{\theta}_{m,j}}|_{\boldsymbol{\theta}_m = \boldsymbol{\theta}_m^{[t+1,j]}} \tag{3.62}$$

Consider the tth iteration at node m and let $\boldsymbol{\theta}_m = \{\boldsymbol{\theta}_{m,j} \quad \boldsymbol{\theta}_{m,l}\}^T$ where $\boldsymbol{\theta}_{m,l}$ are the other parameters of $\boldsymbol{\theta}_m$ when $l \neq j$. We apply the Taylor formula with remainder [77] to expand this gradient at a fixed point $\boldsymbol{\theta}_m^*$. Since $\frac{\partial \mathcal{L}_m(\boldsymbol{\theta}_m)}{\partial \boldsymbol{\theta}_{m,j}}|_{\boldsymbol{\theta}_m = \boldsymbol{\theta}_m^*} = 0$, we can obtain

$$\begin{aligned} \boldsymbol{\theta}_{m,j}^{t+1} - \boldsymbol{\theta}_{m,j}^{*} &= \boldsymbol{\theta}_{m,j}^{t} - \boldsymbol{\theta}_{m,j}^{*} + \mathbf{P}_{\boldsymbol{\theta}_{m,j}^*} \frac{\partial \mathcal{L}_m(\boldsymbol{\theta}_m)}{\partial \boldsymbol{\theta}_{m,j} \partial \boldsymbol{\theta}_{m,j}}|_{\boldsymbol{\theta}_m = \boldsymbol{\theta}_m^*} (\boldsymbol{\theta}_{mj}^{t} - \boldsymbol{\theta}_{mj}^{*}) \\ &+ \mathbf{P}_{\boldsymbol{\theta}_{m,j}^*} \frac{\partial \mathcal{L}_m(\boldsymbol{\theta}_m)}{\partial \boldsymbol{\theta}_{m,j} \partial \boldsymbol{\theta}_{m,l}}|_{\boldsymbol{\theta}_m = \boldsymbol{\theta}_m^*} (\boldsymbol{\theta}_{m,l}^{t} - \boldsymbol{\theta}_{m,l}^{*}) \end{aligned} \tag{3.63}$$

Define the local Hessian of modified function at the fixed point $\boldsymbol{\theta}_m^*$ as

$$\mathbf{H}_{\boldsymbol{\theta}_m^*} = -\frac{\partial^2 \mathcal{L}_m(\boldsymbol{\theta}_m)}{\partial \boldsymbol{\theta}_m \partial \boldsymbol{\theta}_m}|_{\boldsymbol{\theta}_m = \boldsymbol{\theta}_m^*} \tag{3.64}$$

and the following submatrices of Hessian

$$\begin{aligned} \mathbf{H}_{\boldsymbol{\theta}_{m,j}^*} &= -\frac{\partial^2 \mathcal{L}_m(\boldsymbol{\theta}_m)}{\partial \boldsymbol{\theta}_{m,j} \partial \boldsymbol{\theta}_{m,j}}|_{\boldsymbol{\theta}_m = \boldsymbol{\theta}_m^*} \\ \mathbf{H}_{\boldsymbol{\theta}_{m,l}^*} &= -\frac{\partial^2 \mathcal{L}_m(\boldsymbol{\theta}_m)}{\partial \boldsymbol{\theta}_{m,j} \partial \boldsymbol{\theta}_{m,l}}|_{\boldsymbol{\theta}_m = \boldsymbol{\theta}_m^*} \end{aligned} \tag{3.65}$$

where $\mathbf{H}_{\boldsymbol{\theta}_{m,j}}$ is the curvature of the modified log-likelihood function $\mathcal{L}_m(\boldsymbol{\theta}_m)$ with respect to $\boldsymbol{\theta}_{m,j}$, and $\mathbf{H}_{\boldsymbol{\theta}_{m,l}}$ is the coupling between $\boldsymbol{\theta}_{m,j}$ and $\boldsymbol{\theta}_{m,l}$. Let $\mathbf{R}_{\boldsymbol{\theta}_{m,j}}$ denote the $J \times J$ permutation matrix that reorders the elements of $\{\boldsymbol{\theta}_{m,j}, \boldsymbol{\theta}_{m,l}\}$ into $\{1, \cdots, J\}$, and $\mathbf{R}_{\boldsymbol{\theta}_{m,j}} \mathbf{R}_{\boldsymbol{\theta}_{m,j}}^T = \mathbf{I}$. Then, we define the $J \times J$ composite local rate matrix at j cycle for DCEM algorithm

$$\mathbf{M}_{m,j^*}^{DCEM} = \mathbf{R}_{\boldsymbol{\theta}_{m,j}} \begin{bmatrix} \mathbf{I} - \mathbf{P}_{\boldsymbol{\theta}_{m,j}^*} \mathbf{H}_{\boldsymbol{\theta}_{m,j}^*} & -\mathbf{P}_{\boldsymbol{\theta}_{m,j}^*} \mathbf{H}_{\boldsymbol{\theta}_{m,l}^*} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \mathbf{R}_{\boldsymbol{\theta}_{m,j}}^T \tag{3.66}$$

The components of $\boldsymbol{\theta}_{m,l}^t$ are just copied, so after permuting $\mathbf{R}_{\boldsymbol{\theta}_{m,j}}$

$$\boldsymbol{\theta}_m^{[t+1,j]} - \boldsymbol{\theta}_m^* = \mathbf{M}_{m,j}^{DCEM} (\boldsymbol{\theta}_m^{[t+1,j-1]} - \boldsymbol{\theta}_m^*) \tag{3.67}$$

A full cycle consists of one update over each of the J index sets, therefore, after J cycle, we can obtain:

$$\boldsymbol{\theta}_m^{[t+1,J]} - \boldsymbol{\theta}_m^* = \mathbf{M}_{m,J}^{DCEM} \times \cdots \mathbf{M}_{m,1}^{DCEM} (\boldsymbol{\theta}_m^{[t,J]} - \boldsymbol{\theta}_m^*) \tag{3.68}$$

Theorem 2 *There exists $a < 1$ such that for any*

$$\rho(\mathbf{M}_m^{DCEM}) = \|\mathbf{M}_{m,J}^{DCEM} \times \cdots \mathbf{M}_{m,1}^{DCEM}\|_{\mathbf{H}_{\boldsymbol{\theta}_m^*}} \leq a \tag{3.69}$$

where $\|\mathbf{M}\|_{\mathbf{N}} = \|\mathbf{N}^{1/2}\mathbf{M}\mathbf{N}^{-1/2}\|$ denotes the generalized matrix spectral norm with respect to a positive definite matrix $\mathbf{N}$.

Proof: See Appendix B

After J cycles, the output of the $(t+1)$th iteration at node m is defined as:

$$\boldsymbol{\theta}_m^{t+1} = \boldsymbol{\theta}_m^{[t+1,J]} \tag{3.70}$$

By applying the same analytical approach of DEM algorithm to the DCEM algorithm, it is easy to obtain the similar result of convergence properties as

$$\boldsymbol{\Theta}^{t+1} - \boldsymbol{\Theta}^* = \mathbf{M}^{DCEM} (\boldsymbol{\Theta}^t - \boldsymbol{\Theta}^*) \tag{3.71}$$

where $\mathbf{M}^{DCEM}$ is a block diagonal matrix given by

$$\mathbf{M}^{DCEM} = \begin{bmatrix} \mathbf{M}_1^{DCEM} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \mathbf{M}_M^{DCEM} \end{bmatrix} \tag{3.72}$$

Given the analysis above, since $\boldsymbol{\theta}_m^*$ is a fixed point of node m, the eigenvalues corresponding to the m-th diagonal block of $\mathbf{M}^{DCEM}$ should be in the interval $[0, 1)$. For a specific sensor node, the largest eigenvalue of the submatrix corresponds to the convergence rate of the parameters. The largest eigenvalue of the rate matrix $\mathbf{M}^{DCEM}$ is related to the convergence rate of all the network parameters after a full DCEM cycle. Denote the largest eigenvalue of $\mathbf{M}_m^{DCEM}$ as $\boldsymbol{\beta}_m^{DCEM}$, the

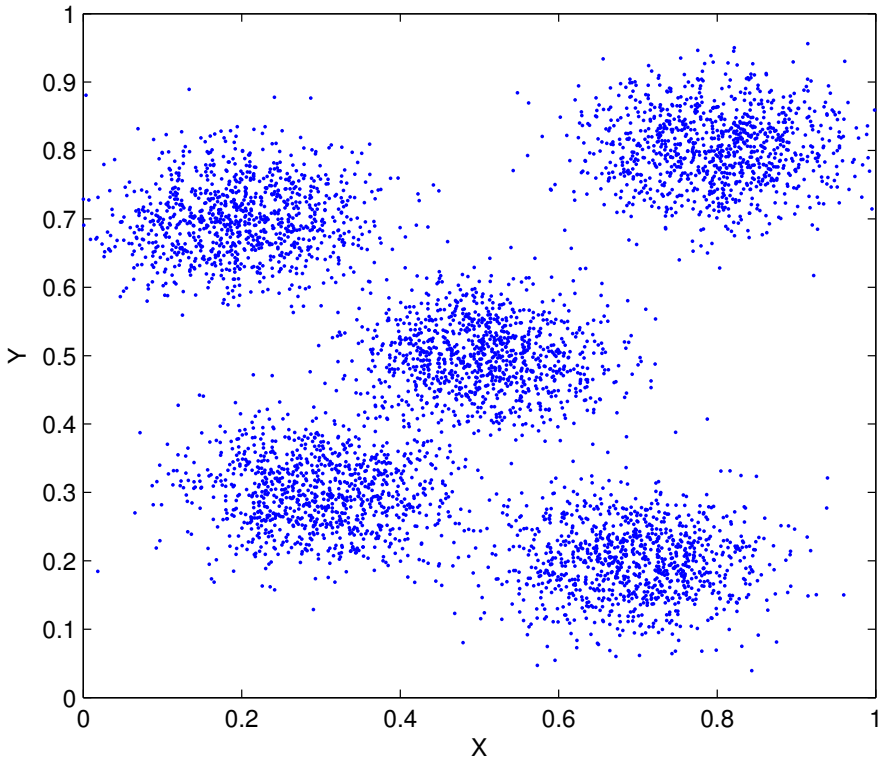


Figure 3.2: Data distribution for well-separated mixture case

convergence rate of DCEM for the whole network can be obtained as follow:

$$\rho(\mathbf{M}^{DCEM}) = \max_{m} |\boldsymbol{\beta}_m^{DCEM}| < 1. \tag{3.73}$$

3.6 Simulation Results

In this section, we demonstrate the feasibility of the proposed algorithm with two different simulated data sets, i.e. the well-separated mixture and overlapping mixture cases. In the simulations, we consider a sensor network with $M = 100$ nodes. This sensor network fulfils the communication requirements specified in [78].

3.6.1 Well-separated Mixtures Model

First, we consider a well-separated components with the observations generated from $J = 5$ Gaussian components distributed as in Figure 3.2. Each component is a $2D$ Gaussian density, the number of data samples at each node is $N_m = 1000$, which can represent environment data clusters. In the first 40 nodes, 60% observations come from the first Gaussian component and the other 40% observations evenly come from the other four Gaussian components, i.e. $\alpha_{m,1} = 60\%, \alpha_{m,2} = \alpha_{m,3} = \alpha_{m,4} = \alpha_{m,5} = 10\%$ for $m = 1, \cdots, 40$. In the next 30 nodes, 70% observations come from the second and third Gaussian components and the other 30% observations evenly come from the other three components, i.e. for $m = 41, \cdots, 70$, $\alpha_{m,1} = \alpha_{m,4} = \alpha_{m,5} = 10\%, \alpha_{m,2} = 40\%, \alpha_{m,3} = 30\%$. For $m = 71, \cdots, 100$, 70% observations come from the last two Gaussian component and other 30% observations evenly from the other three Gaussian components $\alpha_{m,1} = \alpha_{m,2} = \alpha_{m,3} = 10\%, \alpha_{m,4} = 40\%, \alpha_{m,5} = 30\%$. The component parameters (true values) are given by $\boldsymbol{\mu}_1 = [0.2, 0.7]$, $\boldsymbol{\mu}_2 = [0.7, 0.2]$, $\boldsymbol{\mu}_3 = [0.3, 0.3]$, $\boldsymbol{\mu}_4 = [0.5, 0.5]$, $\boldsymbol{\mu}_5 = [0.8, 0.8]$.

For comparison, we apply the proposed DCEM algorithm, the DEM algorithm [1] with a single EM at each node, and DEMM [1] (multiple EM steps at each node) to the same batch of data. These algorithms were randomly initialized with a guess of Gaussian mixture components. As shown in Figure 3.3, the estimates for the x- and y-components of means are close to the reference values.

In Figure 3.4, the log-likelihood values are plotted versus iterations. Convergence is reached when the norm of the difference between successive parameter estimates is less than a specified number, $\epsilon = 10^{-5}$. The proposed algorithm and DEMM algorithm require on average only 10 iterations and 11 iterations respectively to attain the maximal value of log-likelihood, while the DEM algorithm requires 16 iterations to converge. As the complexity of each iteration required using these algorithms is almost the same, this implies at least 37% saving in overall computation comparing DEM algorithm to the proposed algorithm.

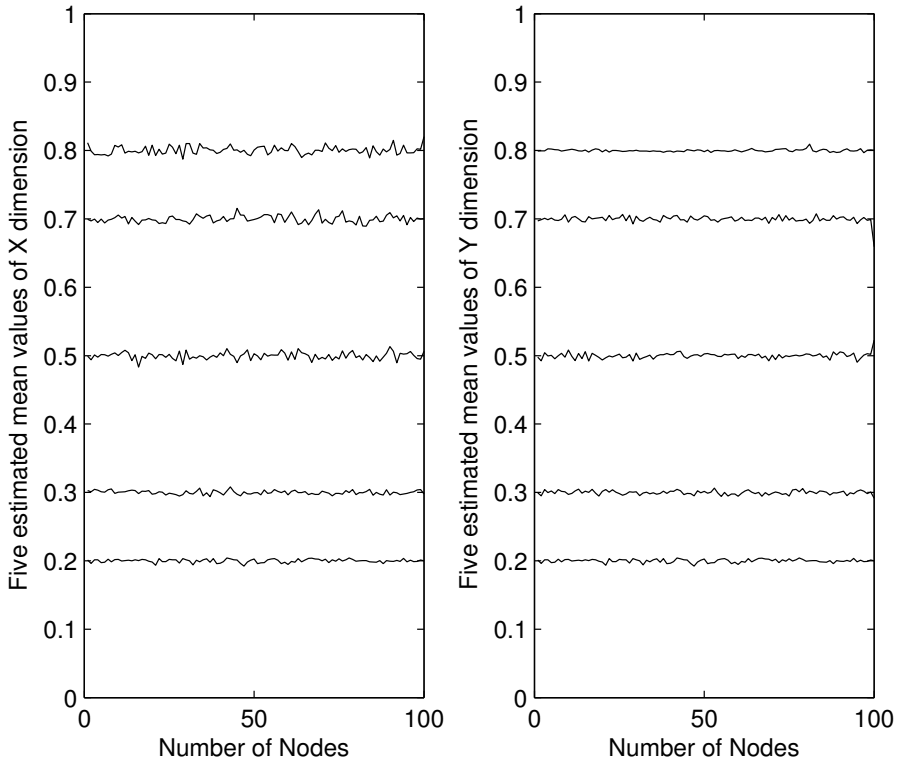


Figure 3.3: Estimates for mean values by the DCEM algorithm for well-separated mixture case.

3.6.2 Overlapping Mixtures Model

Secondly, we consider the overlapping 2D Gaussian density with the same network setting as used in the previous well-sperate mixture model. Each sensor node still takes 1000 observation samples. The observations are generated from the 2D Gaussian mixtures with 5 overlapping components distributed in Figure 3.5. The observations for each sensor node are collected as follows. In the first 30 nodes, 80% observations come from the first Gaussian component and the other 20% observations evenly come from the other four Gaussian components, i.e. $\alpha_{m,1} = 80\%, \alpha_{m,2} = \alpha_{m,3} = \alpha_{m,4} = \alpha_{m,5} = 5\%$ for $m = 1, \cdots, 30$. In the next 40 nodes, 70% observations come from the second and third Gaussian components and the other 30% observations evenly come from the other three components, i.e. for $m = 41, \cdots, 80$, $\alpha_{m,1} = \alpha_{m,4} = \alpha_{m,5} = 10\%, \alpha_{m,2} = 40\%, \alpha_{m,3} = 30\%$. For $m = 71, \cdots, 100$, 70% observations come from the last two Gaussian component and other 30% observations evenly from the other

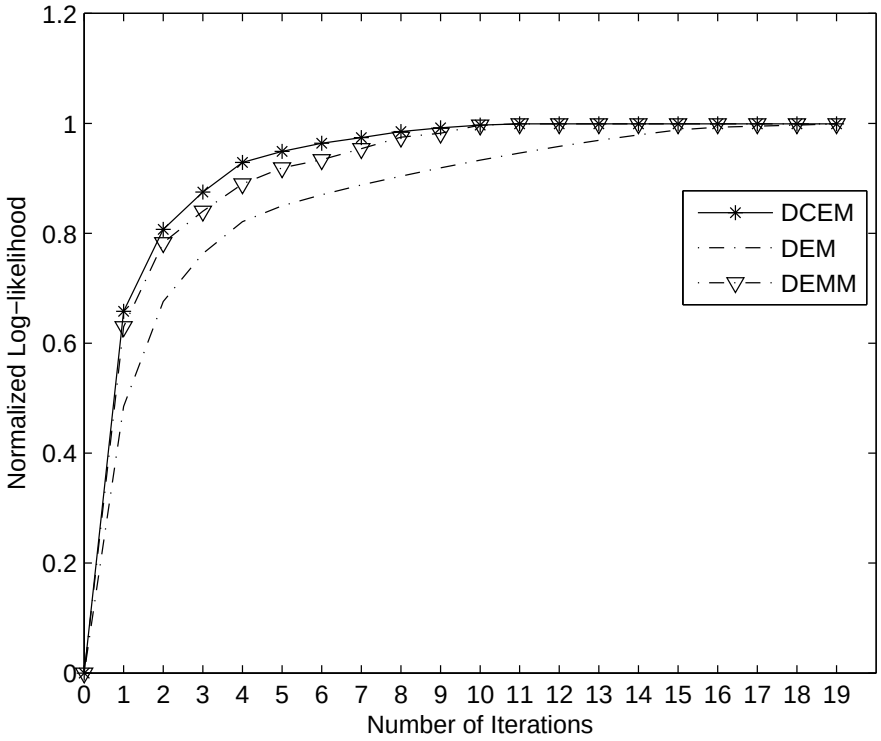


Figure 3.4: Comparison of log-likelihood versus iterations for the DCEM, DEM with single EM step at each node [1] and DEM with multiple EM steps at each node (DEMM) in the well-separated mixture case.

three Gaussian components $\alpha_{m,1} = \alpha_{m,2} = \alpha_{m,3} = 10\%, \alpha_{m,4} = 40\%, \alpha_{m,5} = 30\%$. The component parameters (true values) are given by $\boldsymbol{\mu}_1 = [0.2, 0.6]$, $\boldsymbol{\mu}_2 = [0.6, 0.2]$, $\boldsymbol{\mu}_3 = [0.3, 0.3]$, $\boldsymbol{\mu}_4 = [0.5, 0.5]$, $\boldsymbol{\mu}_5 = [0.7, 0.7]$.

It can be seen from Figure 3.6 that the estimated mean values in all nodes calculated by the DCEM algorithm approximate their true values when overlapping data exist. Figure 3.7 displays the normalized log-likelihood versus the cycle of DCEM, DEM and DEMM in presence of overlapping mixture. All three algorithms suffer from slow convergence compared to the well-separated data sets, though they converge to the same solution. More specifically, the DEM algorithm with a single EM loop at each node appears to converge slowly in 33 iterations so that the DEMM algorithm and especially DCEM algorithm show a significant improvement of convergence speed in around 16 iterations. Moreover, it appears that the implemented version of the DEMM algorithm is less beneficial than the DCEM algorithm for situations where the DEM

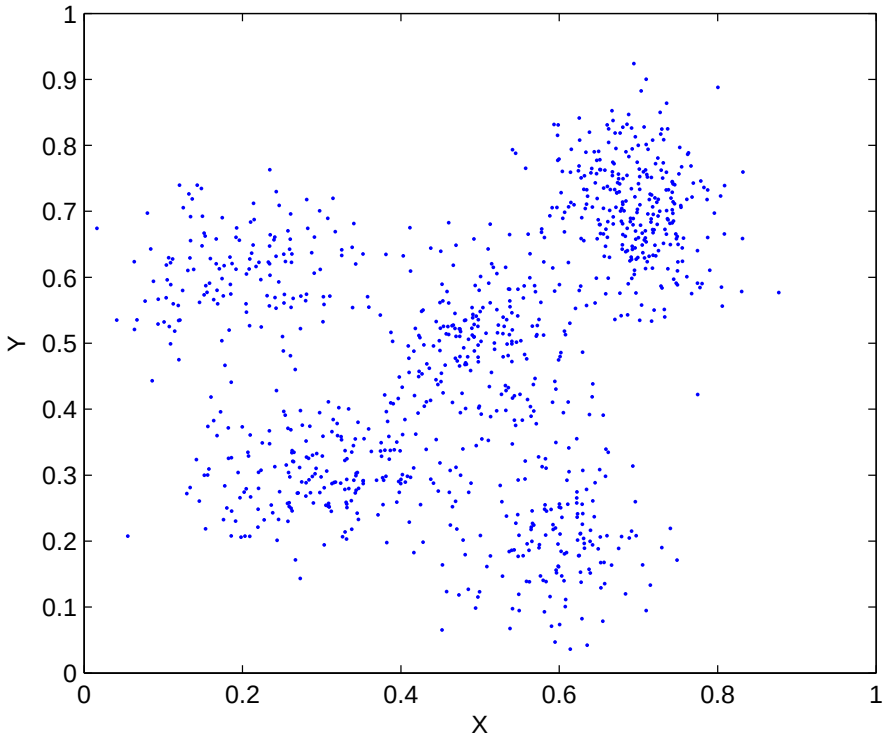


Figure 3.5: Data distribution for overlapping mixture case

algorithm converges slowly. One likely cause of this behavior is that the local procedure of DEMM at each node is still the standard EM update, which still updates the parameters simultaneously, while the local DCEM algorithm finds the estimates sequentially.

3.7 Chapter Summary

In this chapter, we proposed a distributed component-wise EM algorithm for mixture models in sensor networks. The proposed algorithm is characterized by local processing capabilities and sequential computations of component parameters. The ability to process data locally is of particular interest to sensor networks with computationally powerful nodes, and it avoids costly node-to-node communications.

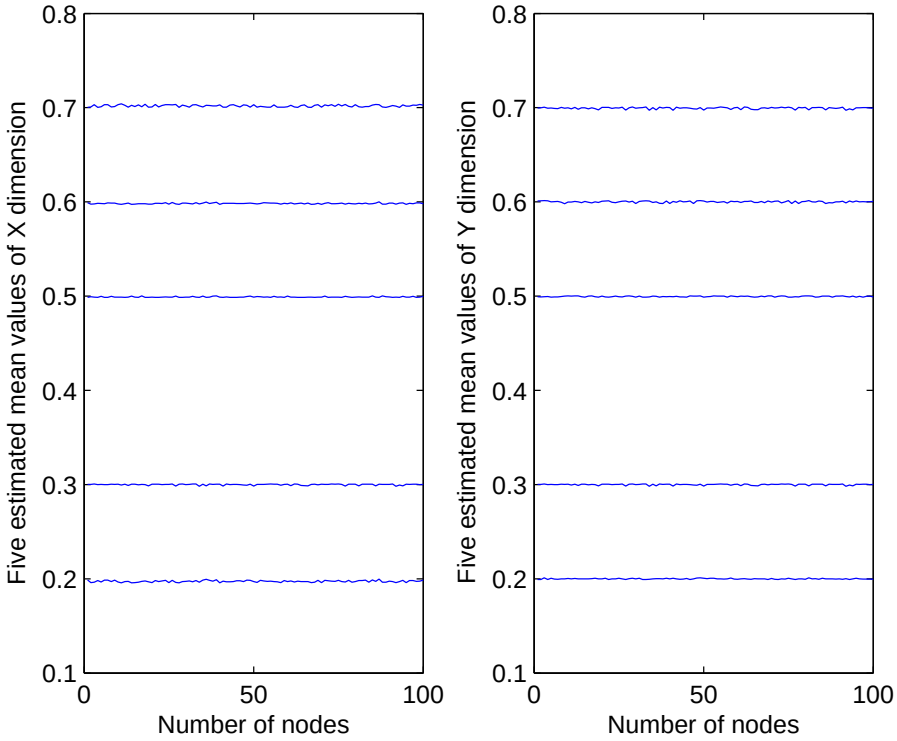


Figure 3.6: Estimates for mean values by the DCEM algorithm for overlapping mixture case.

More importantly, the component-wise update of the mixture parameters leads to significant improvement in convergence rate compared to the DEM algorithm [1]. Simulation results show that the number of iterations required by the proposed algorithm is about 37% less than that required by the distributed EM algorithm. Given the advantages of computational efficiency and simple implementation, we believe that the proposed distributed component-wise EM algorithm is a powerful tool for estimating mixture models in sensor networks. In the following, another kind of distributed EM algorithm based on diffusion strategy will be introduced in Chapter 4

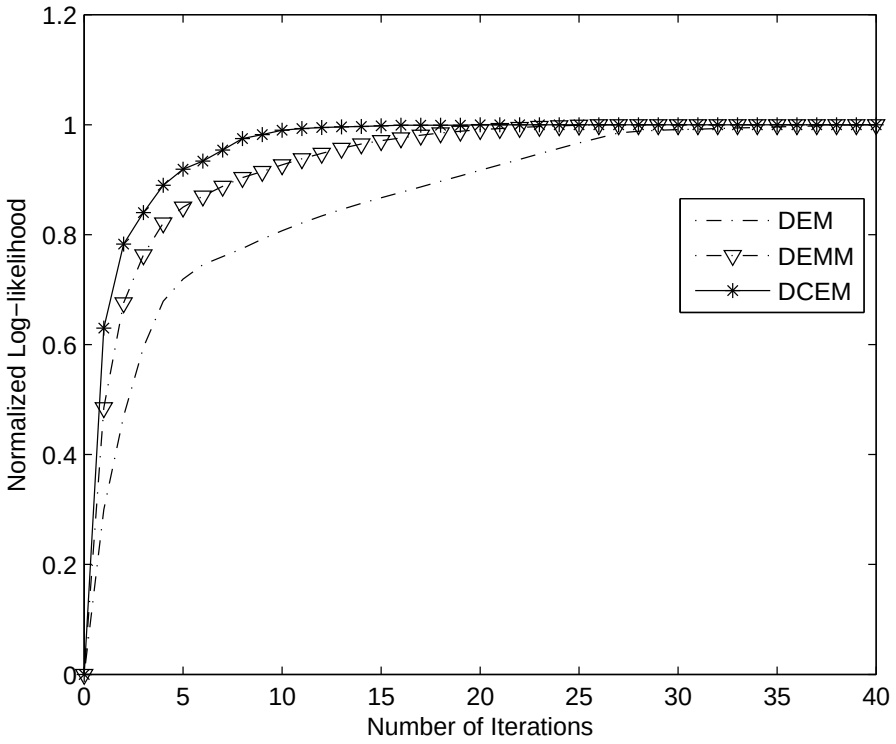


Figure 3.7: Comparison of log-likelihood versus iterations for the DCEM, DEM with single EM step at each node [1] and DEM with multiple EM steps at each node (DEMM) in the overlapping mixture case.

Chapter 4

Diffusion-Based EM Gradient Algorithm for Density Estimation in Sensor Networks

In this chapter, we focus on the mixture density estimation for an asynchronous sensor network in the distributed manner. A random sensor network requires the data samples to be collected and processed at local decentralized processing units. Reformulations of standard EM-type algorithm are necessary to accommodate the characteristics of sensor networks. Existing works on the distributed EM implementation focus mainly on synchronous networks. In this chapter, we address the issue for asynchronous networks by proposing a diffusion-based EM gradient algorithm that updates estimates by using an adapt-then-combine (ATC) diffusion strategy. Simulation results show the robustness and scalability of the proposed approach in the presence of asynchronous events.

4.1 Introduction

The mixture density estimation is used in a number of unsupervised algorithms for environmental monitoring, pattern recognition and clustering. We present the EM algorithm in Chapter 2 to retrieve the maximum likelihood (ML) using latent variables. Based on proper initialisations, the algorithm alternates between two steps, i.e. the expectation (E) step to process the expected log-likelihood function of the measurements and the maximization (M) step to update the estimates based on the conditional log-likelihood function from the E-step. The distributed EM method for a WSN thus requires an adaption to be used for the sensor nodes.

Distributed EM implementations were proposed for calculating the global sufficient statistics

under the consensus-based schemes [2], incremental schemes [1, 79] and diffusion strategies [3, 80, 81]. In [82], it is shown that searching for the global track to link all sensors using incremental strategies is a Hamiltonian circuit problem. The applications in WSNs using consensus-based approaches are hampered by the limited resources of these methods, due to their double time scales [5].

Compared to other strategies, diffusion ones are particularly of interest to us, because the parameters can be estimated locally for each node and different sensors do not have to share the same global statistics. In the literature, a number of diffusion adaptation strategies were proposed for the distributed estimation, detection and filtering, e.g. the diffusion least mean squares estimation algorithms [56, 83, 84], the diffusion recursive least-squares estimation algorithm [85], diffusion adaption for distributed detection [56], and diffusion Kalman filtering and smoothing algorithms for dynamic systems [86]. Unlike the consensus strategy, the diffusion strategy can update the estimate for each node using single time scale, which significantly helps to reduce the communication burden. Given these good features, a distributed EM with diffusion strategies was introduced in [3] to approximate the centralized EM approach using the Robbins-Monro stochastic procedures. Also a diffusion adaption algorithm was discussed in [80] in which the process was implemented in a number of steps for general mixture models. Furthermore, in [81], the authors proposed a novel diffusion-based method to integrate the information propagation into the updates of the parameters.

These algorithms are limited to the synchronous network model, where a coordinated time update is required throughout the network. The asynchronous imperfections are challenging issues for real implementations, e.g. random link failures, random data arrival times, noisy links, random topology changes, agents turning on and off randomly, and even drifting objectives. In this chapter, we present a diffusion-Based EM gradient algorithm for Gaussian mixture models in WSNs. The method is based on an EM gradient method [87] derived for Gaussian mixtures. We develop this method with an asynchronous adaptive diffusion scheme, and address here the general case of density estimation. The main idea behind the proposed algorithm is that the diffusion of the information across the network is embedded in the Expectation step to update parameters. In the Maximization step, gradient based optimization is utilized under the asynchronous ATC diffusion rule [9–11]. The advantage of the proposed method compared to the synchronous diffusion

algorithm in which individual nodes in the network may stop updating the solutions or may stop the normal communications with other nodes. This flexibility can be leveraged to save the power, which is challenging especially for large-scale networks. Although asynchronous events degrade performance as expected, numerical examples provided here still show that the performance of the proposed algorithm is robust and it outperforms the consensus based EM method [2] and diffusion-based distributed EM scheme [3].

This chapter is organized as follows. In Section 4.2 we describe the observation model and Section 4.3 derive the expressions for the centralized EM Gradient algorithm. We derive the distributed optimization of Gaussian mixtures via the synchronous diffusion strategy in Section 4.4. Section 4.5 presents the diffusion-based EM gradient method for density estimation in asynchronous WSNs under the assumption of GMM. Simulation results and summary are presented in Sections 4.6 and 4.7 respectively.

4.2 Problem Formulation

We first assume a sensor network with M nodes, and N_m i.i.d. samples for the mth node are denoted as $\boldsymbol{y}_m = \{\boldsymbol{y}_{m,1}, \cdots, \boldsymbol{y}_{m,N_m}\}$. The data samples follow the Gaussian distribution:

$$\boldsymbol{y}_{m,i} \sim \sum_{j=1}^{J} \alpha_{m,j} \mathcal{P}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j),\ i = 1, \cdots, N_m \tag{4.1}$$

where $\mathcal{P}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is the Gaussian density function. $\boldsymbol{\alpha}_j = \{\alpha_{m,j}\}_{m=1}^{M}$ represent the mixing parameters which can be distinct for different nodes, while component number J is a common parameter. Here we let $\boldsymbol{\theta}_j = \{\boldsymbol{\alpha}_j, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\}$ and the unknown parameter set is denoted as $\boldsymbol{\theta} = \{\boldsymbol{\theta}_j\}_{j=1}^{J}$. Based on the measurements $\boldsymbol{y} = \{\boldsymbol{y}_m\}_{m=1}^{M}$, we aim to estimate the maximum likelihood (ML) of $\boldsymbol{\theta}$ in a distributed manner.

Let the evaluation of the Gaussian density for $\boldsymbol{y}$ be $\mathcal{P}(\boldsymbol{y}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$, the log-likelihood for the mixture model (4.1) can be written as:

$$L(\boldsymbol{\theta}) = \sum_{m=1}^{M} \sum_{i=1}^{N_m} \log \left(\sum_{j=1}^{J} \alpha_{m,j} \mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \right) \tag{4.2}$$

The maximization of 4.2 can be significantly simplified by the EM-type algorithms [53]. We will discuss about these approaches in the following section.

4.3 EM Gradient Algorithms

The formulation of the mixture problem in the EM framework is the same as that in the previous chapter, which is achieved by augmenting the observed data vector $\boldsymbol{y} = \{\boldsymbol{y}_m\}_{m=1}^{M}$ with the associated component-label vectors $\boldsymbol{z} = \{\boldsymbol{z}_m\}_{m=1}^{M}$ where $\boldsymbol{z}_m = \{z_{m,i}\}_{i=1}^{N_m}$. Each $z_{m,i}$ takes on a value from the set $\{1, \ldots, J\}$, where $z_{m,i} = j$ indicates that $y_{m,i}$ was generated by the jth mixture component

$$\boldsymbol{y}_{m,i} \sim \mathcal{P}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j). \tag{4.3}$$

Since we assume that the data samples in this model are statistically independent for each node, the EM algorithm can be applied by each node m individually to its own data. The local data log-likelihood $L_m(\boldsymbol{\theta})$ of is then given by

$$L_m(\boldsymbol{\theta}) = \sum_{i=1}^{N_m} \sum_{j=1}^{J} z_{m,i,j} (\log \alpha_{m,j} + \log \mathcal{P}(\boldsymbol{y}_{m,i} | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)) \tag{4.4}$$

The EM procedure can be implemented at a centralized fusion center, which is assumed to collect all data from the whole network, the global complete data log-likelihood $L(\boldsymbol{\theta})$ becomes

$$L(\boldsymbol{\theta}) \quad = \quad \sum_{m=1}^{M} L_m(\boldsymbol{\theta}) \tag{4.5}$$

Let $\boldsymbol{\theta}^t$ be the parameter set at the tth iteration, the conditional expectation of the complete data log-likelihood can be approximated by

$$\begin{aligned} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^t) &= \sum_{m=1}^{M} Q_m(\boldsymbol{\theta}, \boldsymbol{\theta}_m^t) \\ &= \sum_{i=1}^{N_m} \sum_{j=1}^{J} w_{m,i,j}^{t+1} (\log \alpha_{m,j} + \log \mathcal{P}(\boldsymbol{y}_{m,i} | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)) \end{aligned} \tag{4.6}$$

where

$$w_{m,i,j}^{t+1} = \frac{\alpha_{m,j}^{t}\mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_j^t, \boldsymbol{\Sigma}_j^t)}{\sum_{j=1}^{J}\alpha_{m,j}^{t}\mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_j^t, \boldsymbol{\Sigma}_j^t)}, \tag{4.7}$$

is the posterior probability that the ith sample at node m belongs to the jth component given the observed value $\boldsymbol{y}_{m,i}$. and $Q_m(\boldsymbol{\theta}, \boldsymbol{\theta}_m^t)$ is the local conditional expectation at node m.

In the standard M-step, the parameters are computed by maximizing the complete data log-likelihood in equation (4.6)

$$\boldsymbol{\theta}^{t+1} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^t). \tag{4.8}$$

If the M-step cannot be computed in the closed form, there exists several methods which can be utilized to improve the performance of the EM algorithm in the M-step. The most common algorithm for iteratively solving the M-step is the Newton-type method, which can have the quadratic convergence compared with the linear convergence experienced by the EM algorithm. Based on this knowledge, the EM gradient algorithm was proposed in [87], to update the $\boldsymbol{\theta}^t$ by

$$\begin{aligned}
\boldsymbol{\theta}^{t+1} &= \boldsymbol{\theta}^t - \nabla^{20}Q(\boldsymbol{\theta}^t, \boldsymbol{\theta}^t)^{-1}\nabla^{10}Q(\boldsymbol{\theta}^t, \boldsymbol{\theta}^t), \\
&= \boldsymbol{\theta}^t - \left[\sum_{m=1}^{M}\nabla^{20}Q_m(\boldsymbol{\theta}_m^t, \boldsymbol{\theta}_m^t)\right]^{-1}\sum_{m=1}^{M}\nabla^{10}Q_m(\boldsymbol{\theta}_m^t, \boldsymbol{\theta}_m^t), \\
&= \boldsymbol{\theta}^t - \left[\sum_{m=1}^{M}\nabla^{20}Q_m(\boldsymbol{\theta}_m^t, \boldsymbol{\theta}_m^t)\right]^{-1}\sum_{m=1}^{M}\nabla L_m(\boldsymbol{\theta}_m^t)
\end{aligned} \tag{4.9}$$

where the operators $\nabla^{20}Q_m(\boldsymbol{\theta}_m^t, \boldsymbol{\theta}_m^t)$ and $\nabla^{10}Q_m(\boldsymbol{\theta}_m^t, \boldsymbol{\theta}_m^t)$ are the Hessian matrix and gradient vector of the local conditional log-likelihood function $Q_m(\boldsymbol{\theta}^t, \boldsymbol{\theta}_m^t)$ respectively. In addition, the equality $\nabla^{10}Q_m(\boldsymbol{\theta}_m^t, \boldsymbol{\theta}_m^t) = \nabla L_m(\boldsymbol{\theta}_m^t)$ holds, when $L_m(\boldsymbol{\theta}) - Q_m(\boldsymbol{\theta}, \boldsymbol{\theta}_m^t)$ has its minimum at $\boldsymbol{\theta} = \boldsymbol{\theta}_m^t$. After the new estimate $\boldsymbol{\theta}^{t+1}$ is calculated, it is sent back to all nodes, i.e., $\boldsymbol{\theta}_m^{t+1} = \boldsymbol{\theta}^{t+1}$. In addition, although the convergence of the EM and EM gradient algorithms are assured, the result is sensitive to the initialization. Thus, a proper initialization is crucial for the performance of the algorithm. Note that equation (4.9)is not distributed, the computation of the posteriori probabilities at the M-step require the local information only, whereas the estimates in equation (4.7) requires the global information. Thus, a distributed implementation of the EM algorithm entails local data processing and sharing of information. In the next section, we introduce a adaptive diffusion strategy [88, 89] to process the network communication issue.

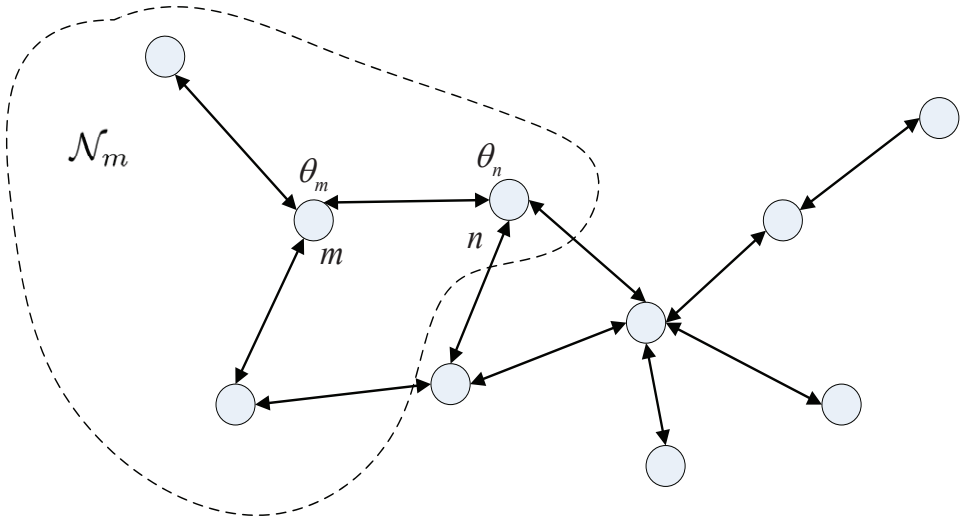


Figure 4.1: A network of integrator nodes in which node m receives the state $\boldsymbol{\theta}_n$ of its neighbor, node n

4.4 Distributed Optimization via Adaptive Diffusion Strategy

We present the details of the distributed optimization for Gaussian mixtures using synchronous ATC strategies [88, 89] in the section. The sensor nodes are assumed be interacted locally with the neighbours, and the communications are illustrated via an undirected graph $\mathcal{G}$ =$\{\mathcal{N}, \mathcal{E}\}$, where $\mathcal{N}$ denotes the node sets, $\mathcal{E}$ represents the edges in which the pair $\{m, n\} \in \mathcal{E}$ stands for the edge between node m and n. Take Figure 4.1 as an example, the neighbourhood of the node m is denoted as $\mathcal{N}_m = \{n \mid \{m, n\} \in \mathcal{E}\}$. Thus, the conditional expectation function can be reformatted as [56]:

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^t) = Q_m(\boldsymbol{\theta}, \boldsymbol{\theta}_m^t) + \sum_{n \neq m} Q_n(\boldsymbol{\theta}, \boldsymbol{\theta}_n^t) \tag{4.10}$$

where $Q_n(\boldsymbol{\theta}, \boldsymbol{\theta}_n^t)$ is second order differentiable term, and thus $Q_n(\boldsymbol{\theta}, \boldsymbol{\theta}_n^t)$ is optimized at a fixed point $\boldsymbol{\theta} = \boldsymbol{\theta}_n^*$. Now $Q_n(\boldsymbol{\theta}, \boldsymbol{\theta}_n^t)$ can be approximated via the second order Taylor expansion:

$$\begin{aligned} Q_n(\boldsymbol{\theta}, \boldsymbol{\theta}_n^t) &\approx Q_n(\boldsymbol{\theta}_n^*, \boldsymbol{\theta}_n^t) + \nabla^{10} Q_n(\boldsymbol{\theta}_n^*, \boldsymbol{\theta}_n^t) \\ &+ \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}_n^*)^T \nabla^{20} Q_n(\boldsymbol{\theta}_n^*, \boldsymbol{\theta}_n^t)(\boldsymbol{\theta} - \boldsymbol{\theta}_n^*) \\ &= \|\boldsymbol{\theta} - \boldsymbol{\theta}_n^*\|_{\Gamma_n}^2 + c \end{aligned} \tag{4.11}$$

where c is a small constant and

$$\Gamma_n = \frac{\nabla^{20} Q_n(\boldsymbol{\theta}_n^*, \boldsymbol{\theta}_n^t)}{2} \tag{4.12}$$

Since $\boldsymbol{\theta}_n^*$ is the fixed point of $Q_n(\boldsymbol{\theta}, \boldsymbol{\theta}_n^t)$, we have $\nabla^{10} Q_n(\boldsymbol{\theta}_n^*, \boldsymbol{\theta}_n^t) = 0$. We can assume that $Q_n(\boldsymbol{\theta}_n^*, \boldsymbol{\theta}_n^t)$ is a constant as it is independent of $\boldsymbol{\theta}$. Therefore, the conditional expectation function in equation (4.6) can be reformatted as:

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^t) = Q_m(\boldsymbol{\theta}, \boldsymbol{\theta}_m^t) + \sum_{n \neq m} \|\boldsymbol{\theta} - \boldsymbol{\theta}_n^*\|_{\Gamma_n}^2 \tag{4.13}$$

However, the optimization over these conditional expectation functions for each node requires that the global information, i.e. the estimate $\boldsymbol{\theta}_n^*$, and the matrices Γ_n for other nodes, are available for all sensors. The equation (4.13) here leads to a viable distributed implementation. In particular, we approximate the equation (4.13) by deriving a local cost functions for each node. Firstly, we bound the sum in equation (4.13) to the neighbourhood of node m, i.e., $n \in \mathcal{N}_m/\{m\}$. Secondly, we introduce an intermediate variable $\boldsymbol{\theta}_n$ for the $\boldsymbol{\theta}_n^*$. Therefore, the minimization can be performed upon the modified conditional expectation function for the mth node:

$$Q_m^{dist}(\boldsymbol{\theta}, \boldsymbol{\theta}_m^t) = Q_m(\boldsymbol{\theta}, \boldsymbol{\theta}_m^t) + \sum_{n \in \mathcal{N}_m/\{m\}} \|\boldsymbol{\theta} - \boldsymbol{\theta}_n^*\|_{\Gamma_n}^2 \tag{4.14}$$

A Newton-type method is applied here to minimize (4.14) in the similar way of optimising (4.9). To deal with scenarios that unknown parameter $\boldsymbol{\theta}$ is a matrix, we will present a quadratic approximation argument [90].

Given the intermediate estimate $\boldsymbol{\theta}_m^t$ for the mth node and a small perturbation δ on $\boldsymbol{\theta}_m^t$, we employ the second-order Taylor expansion of $Q_m^{dist}(\boldsymbol{\theta}_m^t + \delta, \boldsymbol{\theta}_m^t)$ and derive:

$$\begin{aligned} Q_m^{dist}(\boldsymbol{\theta}_m^t + \delta, \boldsymbol{\theta}_m^t) &\approx Q_m(\boldsymbol{\theta}_m^t, \boldsymbol{\theta}_m^t) + \nabla^{10} Q_m(\boldsymbol{\theta}_m^t, \boldsymbol{\theta}_m^t)\delta \\ &+ \frac{1}{2}\delta^T \nabla^{20} Q_m(\boldsymbol{\theta}_m^t, \boldsymbol{\theta}_m^t)\delta + \sum_{n \in \mathcal{N}_m/\{m\}} \|\boldsymbol{\theta}_m^t + \delta - \boldsymbol{\theta}_n^*\|_{\Gamma_n}^2 \end{aligned} \tag{4.15}$$

$Q_m^{dist}(\boldsymbol{\theta}_m^t + \delta, \boldsymbol{\theta}_m^t)$ can be approximated via a quadratic function around $\boldsymbol{\theta}_m^t$, and δ is selected to optimize this approximation (e.g. the gradient with respect to δ is zero). Consider the gradient of

(4.15) along δ, we have:

$$\begin{aligned} \nabla_\delta Q_m^{dist}(\boldsymbol{\theta}_m^t + \delta, \boldsymbol{\theta}_m^t) &\approx \nabla^{10} Q_m(\boldsymbol{\theta}_m^t, \boldsymbol{\theta}_m^t) + \nabla^{20} Q_m(\boldsymbol{\theta}_m^t, \boldsymbol{\theta}_m^t)\delta \\ &+ 2 \sum_{n \in \mathcal{N}_m / \{m\}} \Gamma_n(\boldsymbol{\theta}_m^t + \delta - \boldsymbol{\theta}_n^*) \end{aligned} \tag{4.16}$$

By searching for the δ to give zero gradient, the optimum update strategy is

$$\begin{aligned} \delta &\approx -H^{-1}\Big[\nabla^{10} Q_m(\boldsymbol{\theta}_m^t, \boldsymbol{\theta}_m^t) + 2 \sum_{n \in \mathcal{N}_m / \{m\}} \Gamma_n(\boldsymbol{\theta}_m^t + \delta - \boldsymbol{\theta}_n^*)\Big] \\ &= -H^{-1}\Big[\nabla^{10} Q_m(\boldsymbol{\theta}_m^t, \boldsymbol{\theta}_m^t) + 2 \sum_{n \in \mathcal{N}_m / \{m\}} \nabla^{20} Q_n(\boldsymbol{\theta}_n^*, \boldsymbol{\theta}_n^t)(\boldsymbol{\theta}_m^t + \delta - \boldsymbol{\theta}_n^*)\Big] \end{aligned} \tag{4.17}$$

where

$$\begin{aligned} H &= \nabla^{20} Q_m(\boldsymbol{\theta}_m^t, \boldsymbol{\theta}_m^t) + 2 \sum_{n \in \mathcal{N}_m / \{m\}} \Gamma_n \\ &= \nabla^{20} Q_m(\boldsymbol{\theta}_m^t, \boldsymbol{\theta}_m^t) + 2 \sum_{n \in \mathcal{N}_m / \{m\}} \nabla^{20} Q_n(\boldsymbol{\theta}_n^*, \boldsymbol{\theta}_n^t) \end{aligned} \tag{4.18}$$

Suppose that for different n, the Hessian matrices $\nabla^{20} Q_n(\boldsymbol{\theta}, \boldsymbol{\theta}_n^t)$ do not distinguish from each other significantly. This approximation is likely to be robust as the samples for different nodes follow the same distribution. Then we have

$$\nabla^{20} Q_n(\boldsymbol{\theta}_n^*, \boldsymbol{\theta}_n^t) \approx c_{m,n} H \tag{4.19}$$

$$\nabla^{20} Q_m(\boldsymbol{\theta}_m^t, \boldsymbol{\theta}_m^t) \approx c_{m,m} H \tag{4.20}$$

Compared to (4.18), we have that

$$\begin{aligned} H &\approx \sum_{n \in \mathcal{N}_m} c_{m,n} H \\ and \sum_{n \in \mathcal{N}_m} c_{m,n} &= 1 \end{aligned} \tag{4.21}$$

where $c_{m,n}$ denotes the nonnegative scalar which scales the neighborhood Hessian. Thus, equation (4.17) can be simplified as

$$\delta \approx c_{m,m}\Big[\nabla^{20} Q_m(\boldsymbol{\theta}_m^t, \boldsymbol{\theta}_m^t)\Big]^{-1} \nabla^{10} Q_m(\boldsymbol{\theta}_m^t, \boldsymbol{\theta}_m^t) \tag{4.22}$$

$$- \sum_{n \in \mathcal{N}_m / \{m\}} c_{m,n}(\boldsymbol{\theta}_m^t - \boldsymbol{\theta}_n^*)$$

and the recursive update equation can be written as

$$\begin{aligned} \boldsymbol{\theta}_m^{t+1} &= \boldsymbol{\theta}_m^t - \nu_m c_{m,m} \Big[\nabla^{20} Q_m(\boldsymbol{\theta}_m^t, \boldsymbol{\theta}_m^t)\Big]^{-1} \nabla^{10} Q_m(\boldsymbol{\theta}_m^t, \boldsymbol{\theta}_m^t) \\ &- \nu_m \sum_{n \in \mathcal{N}_m / \{m\}} c_{m,n}(\boldsymbol{\theta}_m^t - \boldsymbol{\theta}_n^*) \end{aligned} \tag{4.23}$$

where ν_m is the step-size which corresponds to the Newton step. The equation (4.23) can be realized through two stages by first introducing an intermediate estimate $\boldsymbol{\psi}_m^{t+1}$

$$\boldsymbol{\psi}_m^{t+1} = \boldsymbol{\theta}_m^t - \nu_m c_{m,m} \Big[\nabla^{20} Q_m(\boldsymbol{\theta}_m^t, \boldsymbol{\theta}_m^t)\Big]^{-1} \nabla^{10} Q_m(\boldsymbol{\theta}_m^t, \boldsymbol{\theta}_m^t) \tag{4.24}$$

$$\boldsymbol{\theta}_m^{t+1} = \boldsymbol{\psi}_m^{t+1} - \nu_m \sum_{n \in \mathcal{N}_m / \{m\}} c_{m,n}(\boldsymbol{\theta}_m^t - \boldsymbol{\theta}_n^*) \tag{4.25}$$

Then $\boldsymbol{\theta}_n^*$ in (4.34) can be replaced by the intermediate estimate $\boldsymbol{\psi}_n^t$ for node n at time t. Similarly, $\boldsymbol{\theta}_m^t$ in equation (4.25) can be replaced by $\boldsymbol{\psi}_m^t$. Based on equation (4.25), we have

$$\begin{aligned} \boldsymbol{\theta}_m^{t+1} &= \boldsymbol{\psi}_m^t - \nu_m \sum_{n \in \mathcal{N}_m / \{m\}} c_{m,n}(\boldsymbol{\psi}_m^t - \boldsymbol{\psi}_n^t) \\ &= (1 - \nu_m + \nu_m c_{m,m})\boldsymbol{\psi}_m^t + \nu_m \sum_{n \in \mathcal{N}_m / \{m\}} c_{m,n} \boldsymbol{\psi}_n^t \end{aligned} \tag{4.26}$$

According to the ATC diffusion strategy [88, 89], we use the coefficients

$$b_{m,m} = 1 - \nu_m + \nu_m c_{m,m} \tag{4.27}$$

$$b_{m,n} = \nu_m c_{m,n} \tag{4.28}$$

Let $\mathbf{B}$ be the $M \times M$ combination matrix which consists of the entries $b_{m,n}$, then $\mathbf{B}$ is a left-stochastic matrix which obeys

$$b_{m,n} = 0 \quad if \quad n \neq \mathcal{N}_m \tag{4.29}$$

$$\mathbf{B}^T \mathbf{1}_M = \mathbf{1}_M$$

where $\mathbf{1}_M$ is the $M \times 1$ all-one vector. We now express the Newton descent of the local conditional

expectation function as

$$\boldsymbol{d}_m^t = -\Big[\nabla^{20} Q_m(\boldsymbol{\theta}_m^t, \boldsymbol{\theta}_m^t)\Big]^{-1} \nabla^{10} Q_m(\boldsymbol{\theta}_m^t, \boldsymbol{\theta}_m^t) \tag{4.30}$$

Therefore, the equations (4.25) and (4.26) can be rewritten as:

$$\boldsymbol{\psi}_m^{t+1} = \boldsymbol{\theta}_m^t + a_m \boldsymbol{d}_m^t, \tag{4.31}$$

$$\boldsymbol{\theta}_m^{t+1} = \sum_{n \in \mathcal{N}_m} b_{m,n} \boldsymbol{\psi}_n^{t+1} \tag{4.32}$$

where $\boldsymbol{a}_m$ is a step-size value, and $b_{m,n}$ are the nonnegative coefficients for combining the estimates.

4.5 Diffusion EM Gradient Algorithm

Based on the gradient EM version algorithm, we propose a distributed EM algorithm scheme where the summations among all observations in equation (4.7) are computed by the asynchronous diffusion strategy [9–11]. In the following , we consider an asynchronous Bernoulli model in a WSN. The WSN is composed of M nodes, where each node adopts a random "on-off" policy to save the power. Furthermore, we employ the EM gradient method at each node, and assume that observations of different nodes are statistically independent. In the E-step, we use an intermediate estimate $\boldsymbol{\theta}_m^t$ of the unknown $\boldsymbol{\theta}$ at node m. The local conditional log-likelihood function is defined as

$$Q_m(\boldsymbol{\theta}, \boldsymbol{\theta}_m^t) = \sum_{i=1}^{N_m} \sum_{j=1}^{J} w_{m,i,j}^{t+1} (\log \alpha_{m,j} + \log \mathcal{P}(\boldsymbol{y}_{m,i} | \boldsymbol{\mu}_{m,j}, \boldsymbol{\Sigma}_{m,j})) \tag{4.33}$$

where

$$w_{m,i,j}^{t+1} = \frac{\alpha_{m,j}^t \mathcal{P}(y_{m,i} | \boldsymbol{\mu}_{m,j}^t, \boldsymbol{\Sigma}_{m,j}^t)}{\sum_{j=1}^{J} \alpha_{m,j}^t \mathcal{P}(y_{m,i} | \boldsymbol{\mu}_{m,j}^t, \boldsymbol{\Sigma}_{m,j}^t)}. \tag{4.34}$$

The main difference between $w_{m,i,j}^{t+1}$ in equations (4.7) and (4.34) is that equation (4.7) is computed using the global estimates $\boldsymbol{\mu}_j^t, \boldsymbol{\Sigma}_j^t$, whereas computing equation (4.34) only requires local estimates $\boldsymbol{\mu}_{m,j}^t, \boldsymbol{\Sigma}_{m,j}^t$ at each node m. With local periodic data exchanges, the local

information in equation (4.34) is appropriately diffused over the network. In order to reduce the computational complexity, the projection matrix [72] is utilised to replace the inverse of Hessian matrix. Then local EM procedure can be expressed as follows:

$$\mathcal{A}_m^{t+1} = \mathcal{A}_m^t + P_{\mathcal{A}_m^t} \frac{\partial L_m(\boldsymbol{\theta}_m)}{\partial \mathcal{A}_m}|_{\mathcal{A}_m=\mathcal{A}_m^t}, \tag{4.35}$$

$$\boldsymbol{\mu}_{m,j}^{t+1} = \boldsymbol{\mu}_{m,j}^t + P_{\boldsymbol{\mu}_{m,j}^t} \frac{\partial L_m(\boldsymbol{\theta_m})}{\partial \boldsymbol{\mu}_{m,j}}|_{\boldsymbol{\mu}_{m,j}=\boldsymbol{\mu}_{m,j}^t}, \tag{4.36}$$

$$\text{vec}[\boldsymbol{\Sigma}_{m,j}^{t+1}] = \text{vec}[\boldsymbol{\Sigma}_{m,j}^t] + P_{\boldsymbol{\Sigma}_{m,j}^t} \frac{\partial L_m(\boldsymbol{\theta}_m)}{\partial \text{vec}[\boldsymbol{\Sigma}_{m,j}]}|_{\boldsymbol{\Sigma}_{m,j}=\boldsymbol{\Sigma}_{m,j}^t}, \tag{4.37}$$

where vec$[\boldsymbol{C}]$ denotes the vector obtained by stacking the column vectors of matrix $\boldsymbol{C}$, $\mathcal{A}_m$ denotes the vector of mixing proportions $[\alpha_{m,1}, \cdots, \alpha_{m,J}]^T$ and

$$P_{\mathcal{A}_m^t} = \frac{\{\text{diag}[\alpha_{m,1}^t, \cdots, \alpha_{m,J}^t] - {\mathcal{A}_m}^t ({\mathcal{A}_m}^t)^T\}}{N_m}, \tag{4.38}$$

$$P_{\boldsymbol{\mu}_{m,j}^t} = \frac{\boldsymbol{\Sigma}_{m,j}^t}{\sum_{i=1}^{N_m} w_{m,i,j}^{t+1}}, \tag{4.39}$$

$$P_{\boldsymbol{\Sigma}_{m,j}^t} = \frac{2}{\sum_{i=1}^{N_m} w_{m,i,j}^{t+1}} {\boldsymbol{\Sigma}_{m,j}}^t \otimes {\boldsymbol{\Sigma}_{m,j}}^t, \tag{4.40}$$

where $\otimes$ denotes the Kronecker product. Using the notation

$$\boldsymbol{\theta} = [\mathcal{A}_m^T, \boldsymbol{\mu}_{m,1}^T, \cdots, \boldsymbol{\mu}_{m,J}^T, \text{vec}[\boldsymbol{\Sigma}_{m,1}]^T, \cdots, \text{vec}[\boldsymbol{\Sigma}_{m,J}]^T]^T \tag{4.41}$$

and

$$P(\boldsymbol{\theta}_m) = \text{diag}\{P_{\mathcal{A}_m}, P_{\boldsymbol{\mu}_{m,1}}, \cdots, P_{\boldsymbol{\mu}_{m,J}}, P_{\boldsymbol{\Sigma}_{m,1}}, \cdots, P_{\boldsymbol{\Sigma}_{m,J}}\} \tag{4.42}$$

we can obtain

$$\boldsymbol{\theta}_m^{t+1} = \boldsymbol{\theta}_m^t + P_m({\boldsymbol{\theta}_m}^t) \nabla L_m(\boldsymbol{\theta}_m^t) \tag{4.43}$$

In the M-step, the ATC diffusion-oriented optimization method is used to find the estimates, whose updates are given by

$$\boldsymbol{\psi}_m^{t+1} = \boldsymbol{\theta}_m^t + a_m^{t+1} \boldsymbol{d}_m^t, \tag{4.44}$$

$$\boldsymbol{\theta}_m^{t+1} = \sum_{n \in \mathcal{N}_m^{t+1}} b_{m,n}^{t+1} \boldsymbol{\psi}_n^{t+1}, \tag{4.45}$$

where

$$\boldsymbol{d}_m^t = P_m(\boldsymbol{\theta}_m^t) \nabla L_m(\boldsymbol{\theta}_m^t) \tag{4.46}$$

is the local gradient descent to the estimate using a Newton-like method in WSN and $\mathcal{N}_m^t$ denotes the random neighbourhood of node m at time $t+1$. Similar to the synchronous version, this scheme includes two operations as described in the previous section, the first step involves the local adaptation, in which node m update its local estimates from $\boldsymbol{\theta}_m^t$ to an intermediate value $\boldsymbol{\psi}_m^{t+1}$. The second step is a combination step, in which the combination of intermediate estimates $\{\boldsymbol{\psi}_n^{t+1}\}$ from the neighbourhood of node m is used to calculate the new estimates $\boldsymbol{\theta}_m^{t+1}$. In the adaptation step, node m enters an active mode with probability $0 < q_m < 1$ and evaluates equation (4.44), and it enters a sleep mode with probability $1 - q_m$ to save energy. The random step-sizes $\boldsymbol{a}_m^{t+1}$ used in equation (4.44) depend on the probability q_m and satisfy

$$a_m^{t+1} = \begin{cases} a_m, & \text{with probability } q_m \\ 0, & \text{with probability } 1 - q_m \end{cases} \tag{4.47}$$

where a_m is a constant step-size. The underlying topology of network is assumed to be fixed. In the combination step, each node m is allowed to randomly select its neighborhood n with probability $0 < p_m < 1$ and evaluates equation (4.45) for saving communication costs. The combination coefficients $\{b_{n,m}^{t+1}\}$ are nonnegative parameters and they are required to satisfy the following constraints

$$b_{m,n}^{t+1} = \begin{cases} b_{m,n} > 0, & \text{with probability } p_m \\ 0, & \text{with probability } 1 - p_m \end{cases} \tag{4.48}$$

For all $n \in \mathcal{N}_m \setminus \{m\}$, and node m is required to adjust its own weight $b_{m,m}^{t+1}$ at each update via

$$b_{m,m}^{t+1} = 1 - \sum_{n \in \mathcal{N}_m^t \setminus \{m\}} b_{m,n}^{t+1} \tag{4.49}$$

to guarantee $\sum_{n \in \mathcal{N}_m^t} b_{m,n}^{t+1} = 1$. Note that $\boldsymbol{a}_m^{t+1}$ and $\{b_{m,n}^{t+1}\}$ are mutually independent, and the use of these distributed control parameters enables the diffusion strategies which can process various type of asynchronous network events. The independence between $\{\boldsymbol{a}_m\}$ and the constant step-size used in synchronous diffusion networks enables us to set up a random "on-off" behavior at the mth agent to save power. Furthermore, the coefficient $\{b_{m,n}^{t+1}\}$ can be used to structure a random "on-

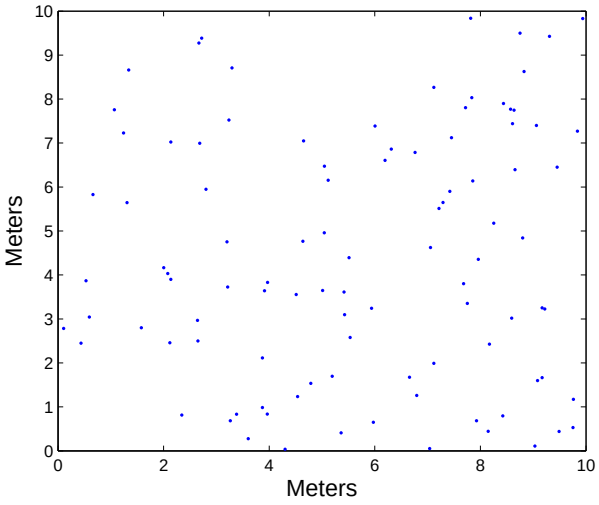
off" status for the connection from agent m to agent n at time $t+1$ to save the communication cost. If the agents select the links randomly. i.e. there is only one other neighbouring agent being communicated with at each time, then we are able to mimic the random gossip strategies failures [7, 91–94]. It is worth mentioning that the sources for the randomness of the combination coefficients are three facets. Firstly, it can come from the randomness in the topology which is often used to simulate the network dynamics. Secondly, the connections between agents can drop randomly. This can happen when we have interferences or other power saving strategies. Thirdly, some agents may hold random combination coefficients, given that $\sum_{n\in\mathcal{N}_m} b_{m,n}^{t+1} = 1$ is satisfied. We will demonstrate the convergence and reliability through numerical simulation results in the next section.

4.6 Simulation Results

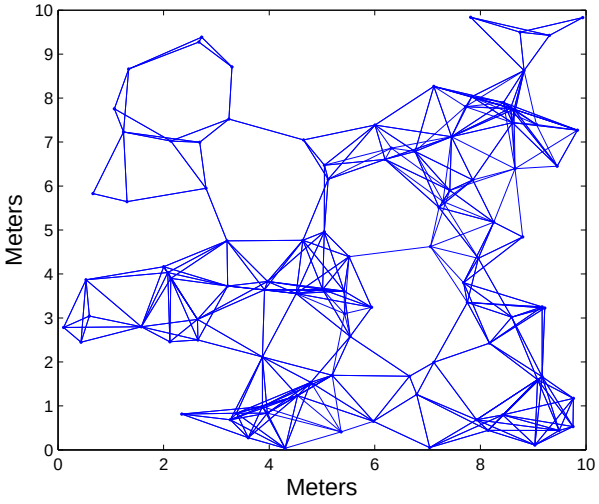
In this section, we demonstrate the feasibility of the proposed algorithm through MATLAB simulations. In the simulation, we consider a sensor network with $M = 100$ nodes in a 10×10 meter squares, and the number of data samples at each node is $N_m = 100$. It can be extended to other scenarios without loss of generality. This sensor network fulfills the communication requirement specified in [9–11] with connectivity radio range r. Figure 4.2 shows the cases of different communication radio range with $r = 0, r = 1.5$, and $r = 2.5$. the proposed algorithms use asynchronous diffusion combination matrix B under Metropolis rule [9–11] with entries defined as

$$b_{m,n} = \begin{cases} 1/(\max\{|\mathcal{N}_m|, |\mathcal{N}_n|\}), & n \in \mathcal{N}_m \\ 1 - \sum_{k\in\mathcal{N}_m\setminus\{m\}} \boldsymbol{b}_{m,n}, & m = n \\ 0, & \text{otherwise.} \end{cases} \tag{4.50}$$

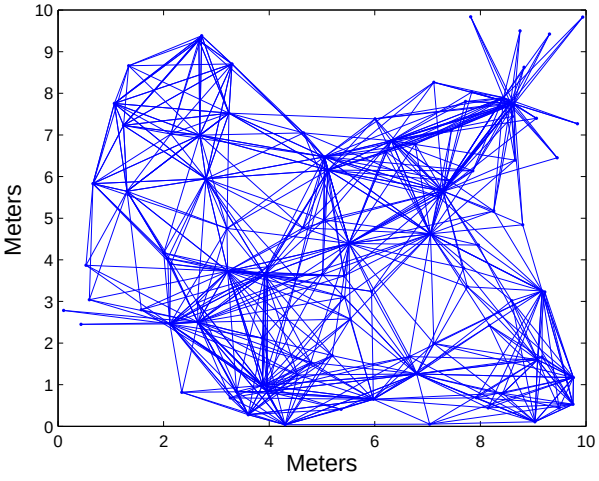
where $|\cdot|$ denotes the cardinality. For comparison, we apply the proposed asynchronous diffusion-based EM gradient algorithm, the diffusion-based distributed EM algorithm (DDEM) [3], the consensus based EM algorithm and the standard EM algorithm to the same batch of data.



(a) Unconnected nodes (Radio Range=0 m)



(b) Radio Range=1.5 m



(c) Radio Range=2.5 m

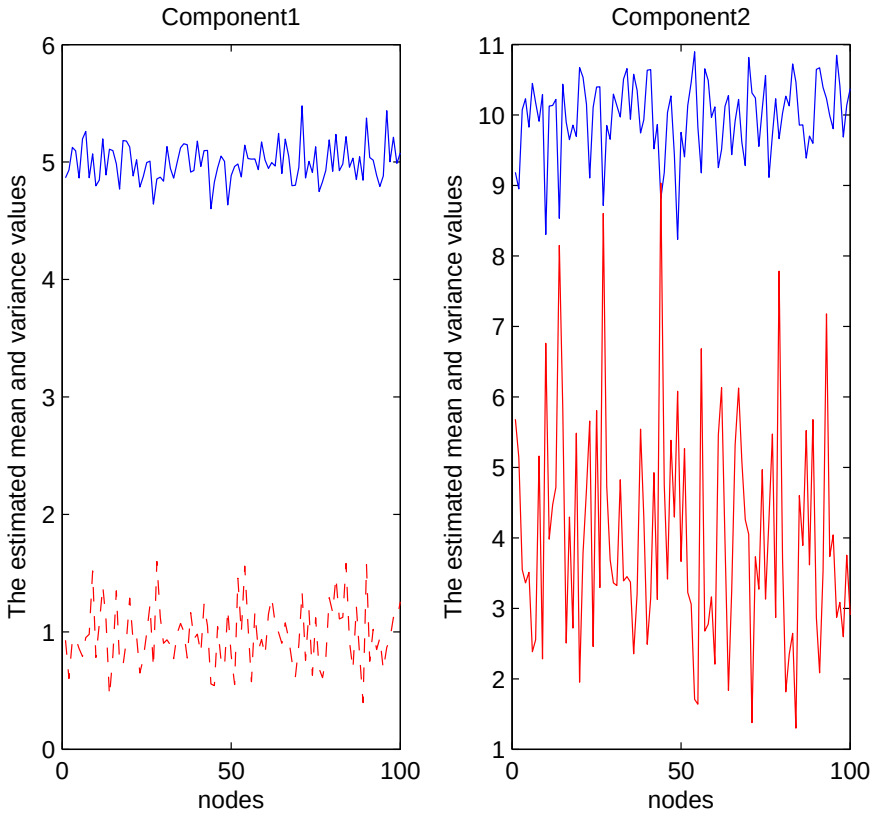
Figure 4.2: 100 randomly distributed sensors with different radio ranges

4.6.1 1-Dimensional Data

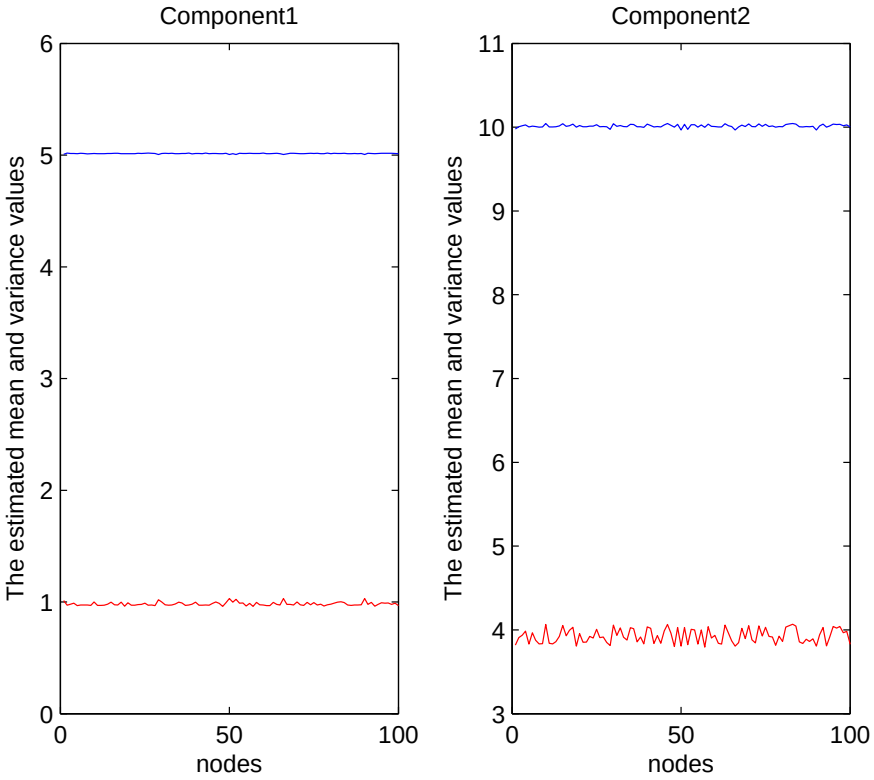
First, we consider the 1-dimensional data case, and the observations are generated from $J = 2$ distributed Gaussian components. Each component is a 1-dimensional Gaussian mixture density, which can represent environment data clusters. In the first 50 nodes, 60% of the observations come from the first Gaussian component and the other 40% observations evenly come from the second Gaussian component, i.e. $\alpha_{m,1} = 0.6, \alpha_{m,2} = 0.4$ for $m = 1, \cdots, 50$. In the last 50 nodes, 30% observations come from the first Gaussian component and the other 70% observations evenly come from the second component, i.e. for $m = 51, \cdots, 100$, $\alpha_{m,1} = 0.3, \alpha_{m,2} = 0.7$. The component means and variances are given by $\boldsymbol{\mu}_1 = 5$, $\boldsymbol{\mu}_2 = 10$, $\boldsymbol{\sigma}_1^2 = 1$, $\boldsymbol{\sigma}_2^2 = 4$. The radio range for communication is set to 1.5 meters. The step-size $\boldsymbol{a}_m = 0.05$ is uniform across the network. As shown in Figure 4.3, the EM gradient algorithm with asynchronous diffusion setting and the local standard EM without cooperation are tested. The probabilities for the Bernoulli model are set as $q_m = p_m = 0.8$. It can been seen from Figure 4.3, the mean and variance estimates are noisy for each sensor node with the standard EM algorithm using only the local information, while the estimates with the proposed algorithm are much smoother for each sensor node, even under the imperfect communication condition. In Figure 4.4, the mean-square-deviation (MSD) is used and evaluated for performance of different algorithms, which is defined as

$$\text{MSD}_{\boldsymbol{\theta}} = E\left[\|\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}\|_2^2\right] \tag{4.51}$$

where $\|\cdot\|_2$ is the Euclidean norm. We compare our algorithm with the consensus based EM and DDEM. We select 100 consensus iterations in order to satisfy the condition of the convergence of discrete consensus filter, and set $\eta = 1/M$ to satisfy the condition for the convergence of consensus filter in [2] is that $\eta = 1/d_{max}$, where d_{max} is the maximum degree. In addition, we include a single round of averaging in the D-step and M-step for the DDEM algorithm [3], respectively. Both consensus based EM and DDEM algorithm are operated in a synchronous network setting, for a fair comparison, we select the value of probability $p_m = q_m = 1$ (corresponds to traditional synchronous diffusion). It can be seen from Figure 4.4(a), a diffusion strategy provides improved mean-square-deviation in simulation compare to consensus based EM and DDEM algorithms. To illustrate the performance of the proposed algorithm in an



(a) Local standard EM algorithm without cooperation for 100 nodes



(b) Diffusion-based EM gradient algorithm for 100 nodes

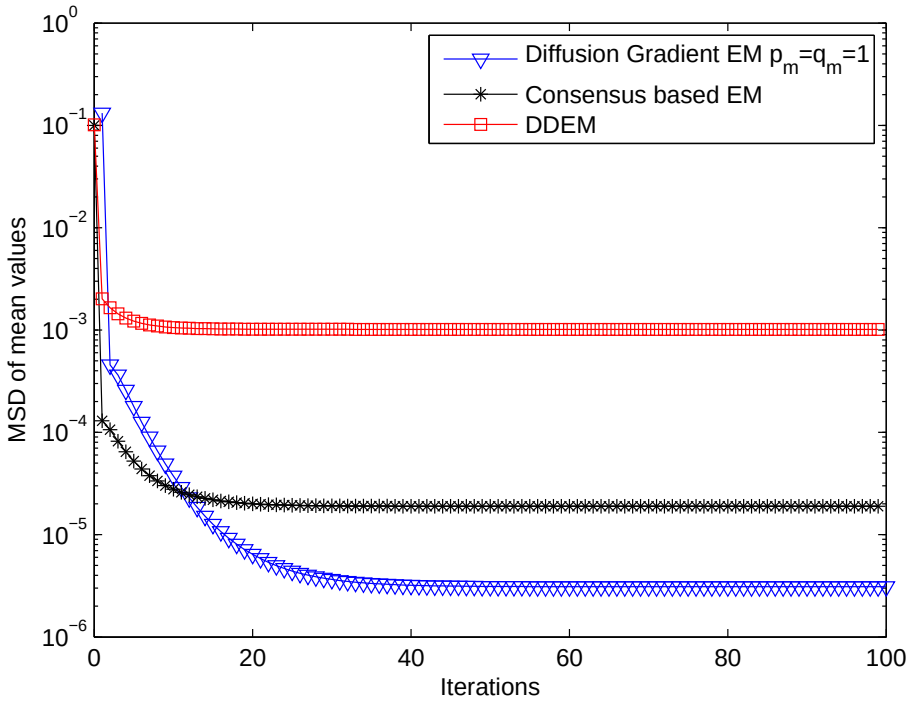
Figure 4.3: Estimated mean and variance for two Gaussian components with different schemes

asynchronous network setting, we selected the value of probabilities with different cases, $p_m = q_m = 0.3$, $p_m = q_m = 0.5$, $p_m = q_m = 0.8$ and $p_m = q_m = 1$. From Figure 4.4 (b), the proposed asynchronous diffusion algorithm converges to almost as the same rate as the synchronous version. However, due to the additional randomness in the adaption process, EM gradient method with asynchronous diffusion suffers from a slight degradation in MSD performance.

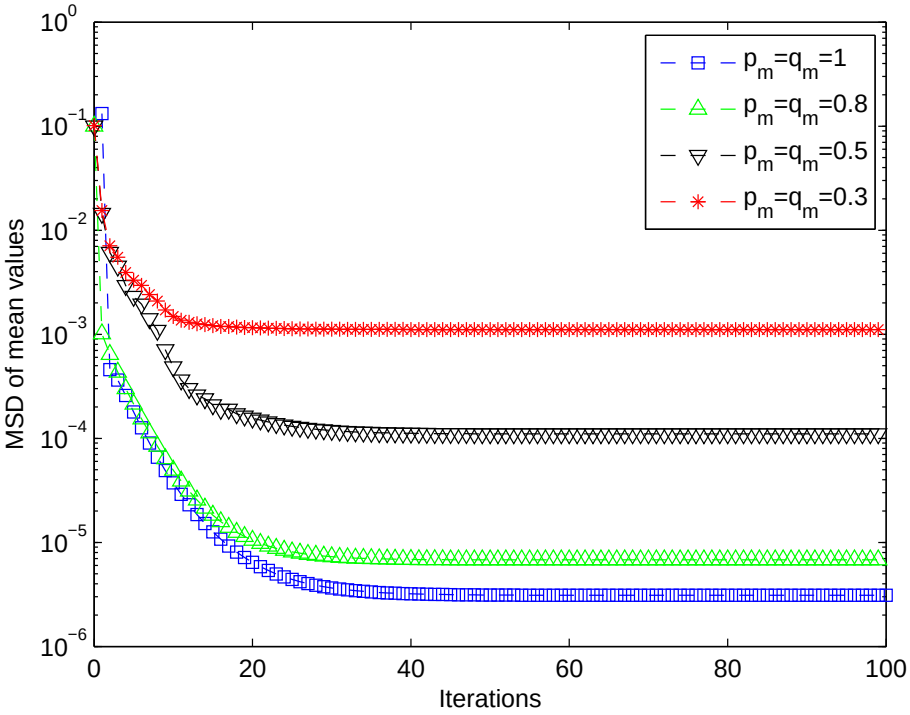
The scalability of the proposed diffusion EM gradient approach is also explored, with $M = 1000$ sensor nodes randomly deployed in the same square. Observations for each sensor node and asynchronous network setting are selected as the test of network size $M = 100$. The estimated mean and variance for 1000 nodes are shown in Figure 4.5 for both the local and the diffusion scheme. From Figure 4.6. we can see that the MSD performance of all these algorithms are improved by the increased network size in comparison with $M = 100$ test, and the diffusion strategy still outperforms consensus based EM and DDEM algorithms in MSD performance with a large scale network.

4.6.2 2-Dimensional Data

In this subsection, we investigate the 2D Gaussian mixture density with $M = 100$ sensor nodes randomly generated in the same square. The observations are generated from $J = 2$ 2D Gaussian components distributed as shown in Figure 4.7. Each component is a $2D$ Gaussian density, the number of data samples at each node is $N_m = 100$. In the first 40 nodes, 60% observations come from the first Gaussian component and the other 40% observations evenly come from the other two Gaussian components, i.e. $\alpha_{m,1} = 60\%, \alpha_{m,2} = \alpha_{m,3} = 20\%$ for $m = 1, \cdots, 40$. In the next 30 nodes, 70% observations come from the second Gaussian component and the other 30% observations evenly come from the other two components, i.e. for $m = 41, \cdots, 70$, $\alpha_{m,1} = \alpha_{m,3} = 15\%, \alpha_{m,2} = 70\%$. For $m = 71, \cdots, 100$, 60% observations come from the last Gaussian component and the other 40% observations evenly come from the other Gaussian components $\alpha_{m,1} = \alpha_{m,2} = 20\%, \alpha_{m,3} = 60\%$. The component parameters are given by $\boldsymbol{\mu}_1 = [0.7, 0.7]$, $\boldsymbol{\mu}_2 = [0.5, 0.5]$, $\boldsymbol{\mu}_3 = [0.3, 0.3]$,

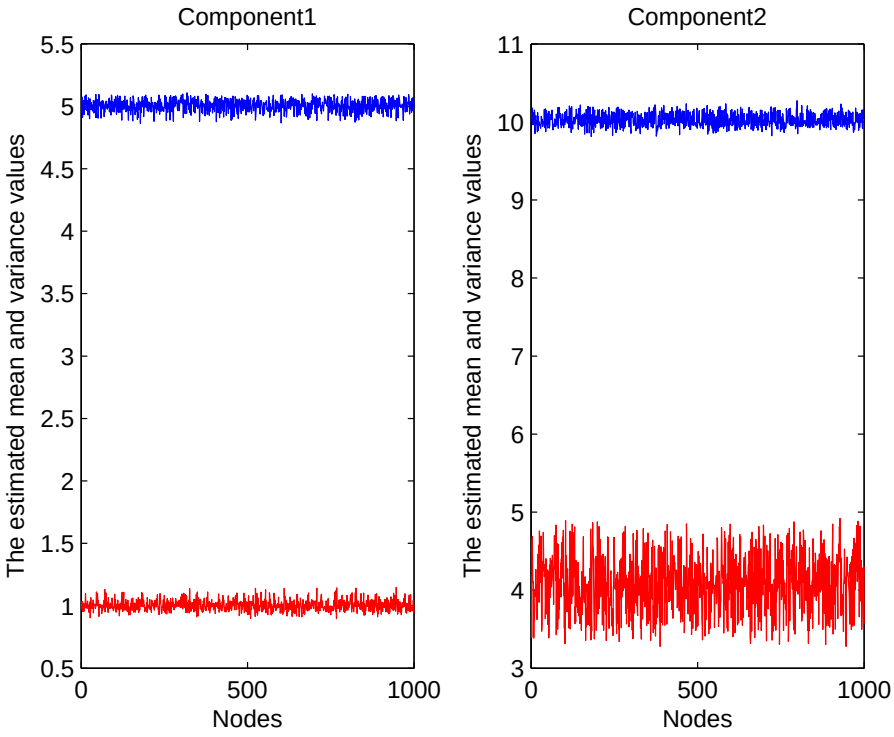


(a) Synchronous network setting for 100 nodes

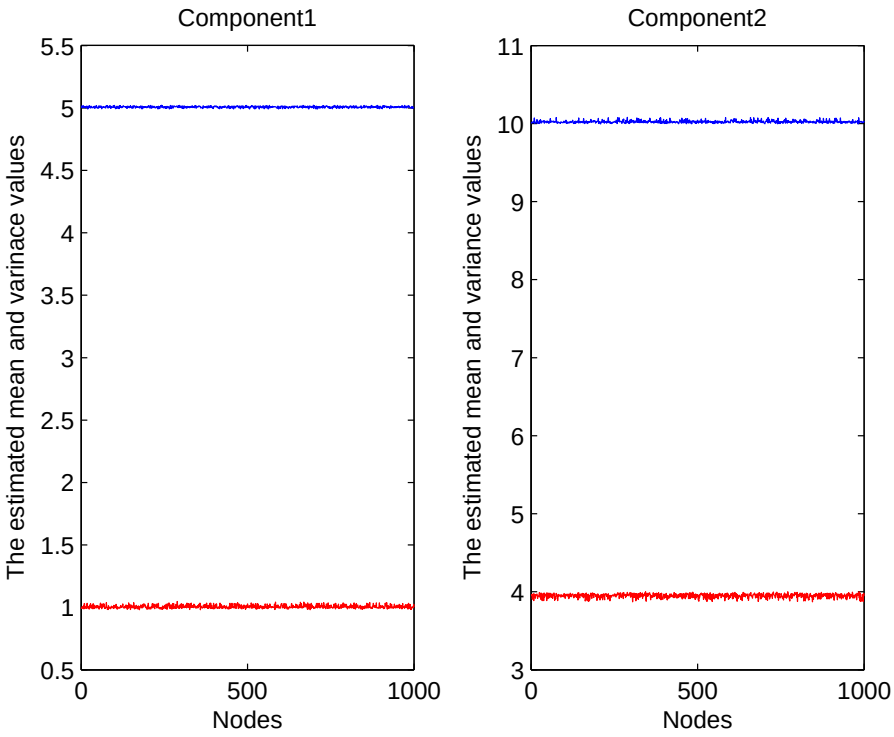


(b) Asynchronous network setting for 100 nodes

Figure 4.4: Comparison of network MSD vs iteration index for asynchronous diffusion, synchronous diffusion, consensus based EM [2] and DDEM [3]

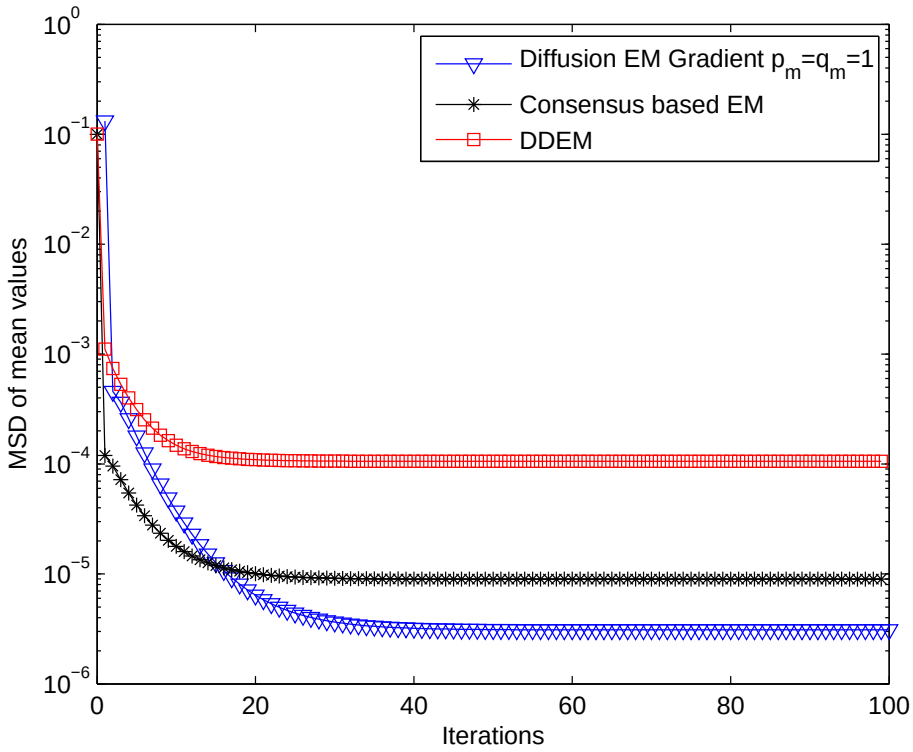


(a) Local standard EM algorithm without cooperation for 1000 nodes

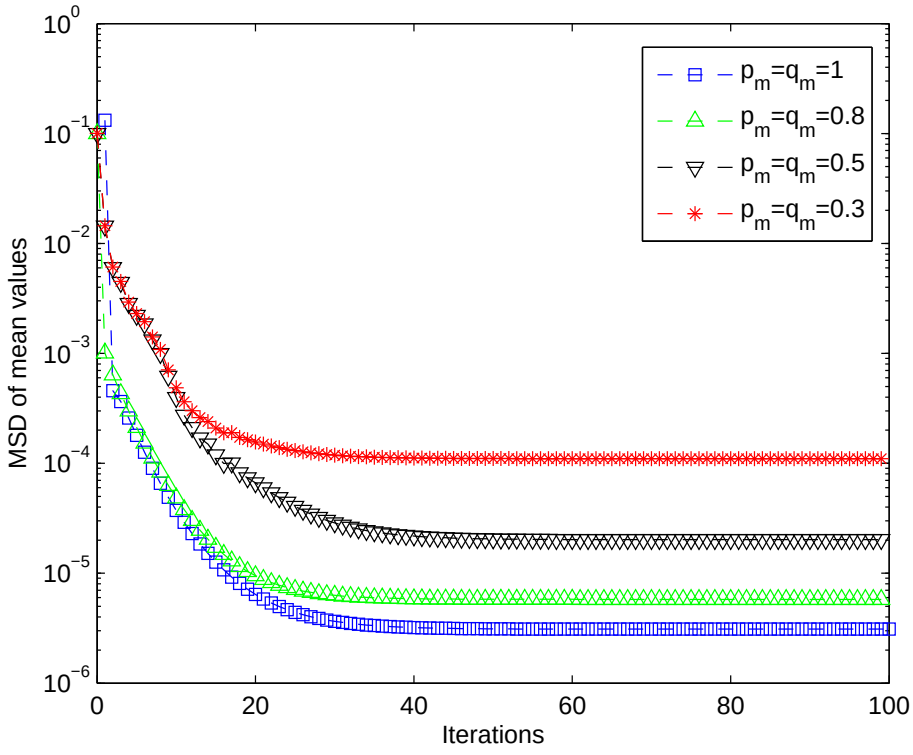


(b) Diffusion-based EM gradient algorithm for 1000 nodes

Figure 4.5: Estimated mean and variance with different schemes for 1000 nodes



(a) Synchronous network setting for 1000 nodes



(b) Asynchronous network setting for 1000 nodes

Figure 4.6: Large scale network MSD (1000 nodes) comparison vs iteration index for asynchronous diffusion, synchronous diffusion, consensus based EM [2] and DDEM [3]

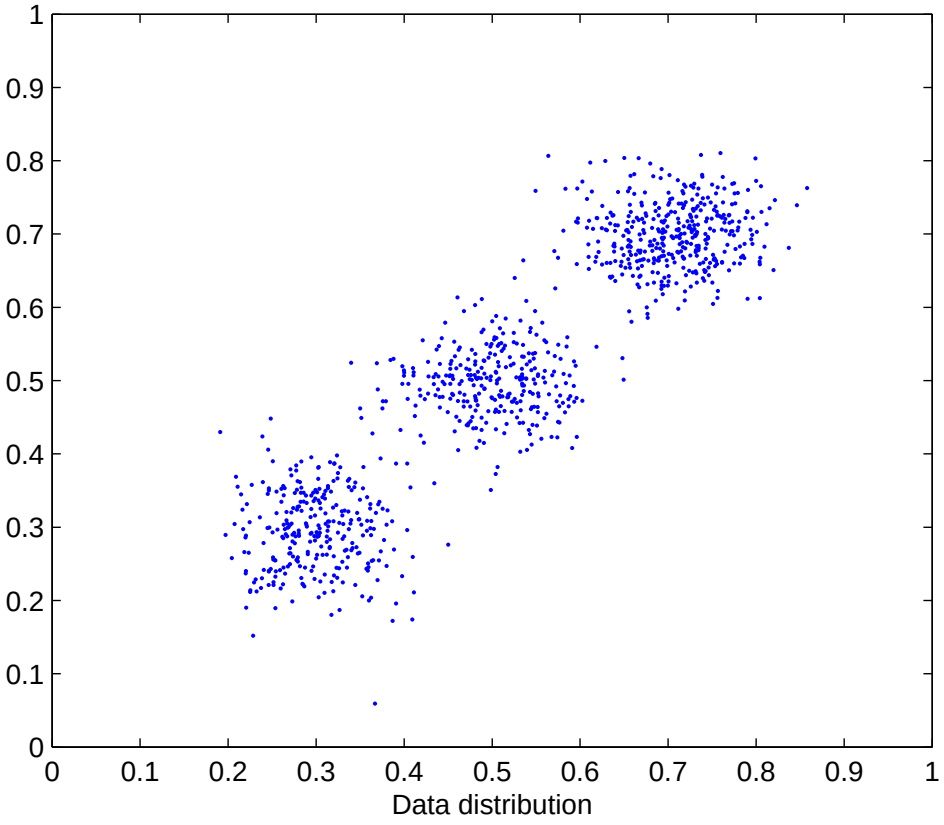
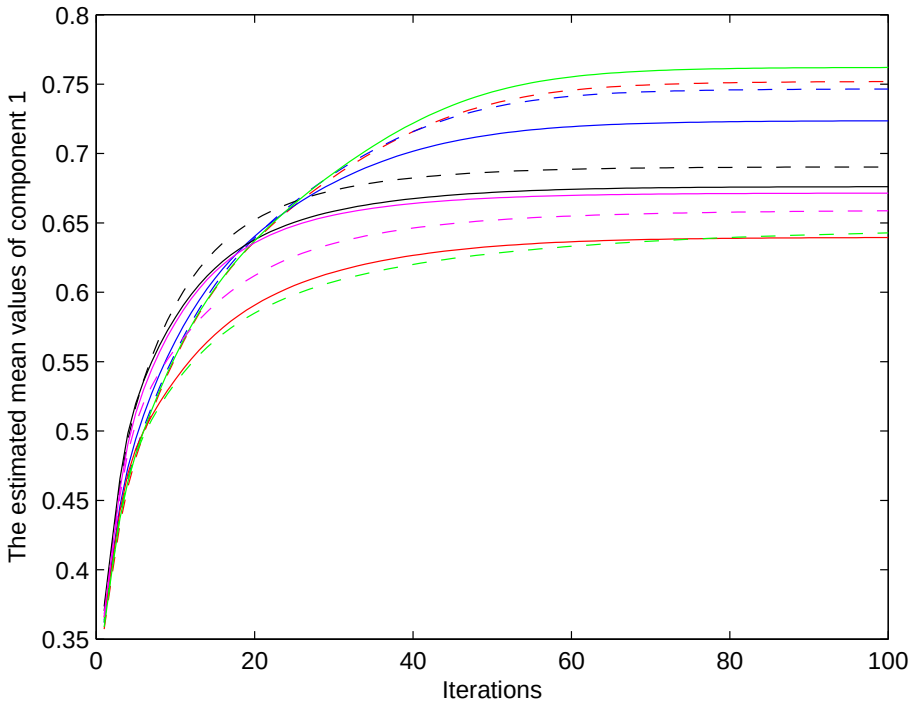
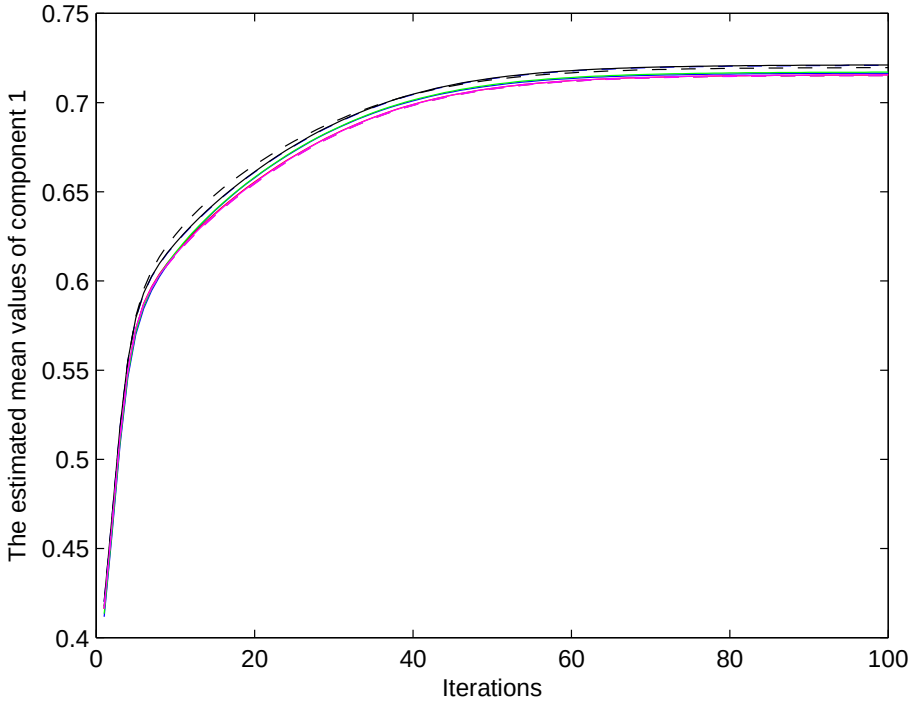


Figure 4.7: Data distribution for 2D Gaussian mixtures with 3 components

We set the probabilities for Bernoulli model as $q_m = p_m = 0.8$ and communication radio range $r = 1.5$ meters, and randomly select 10 sensor nodes to demonstrate the evolution of the mean of the first Gaussian component $\boldsymbol{\mu}_1 = [0.7, 0.7]$ during the iteration process in the Figure 4.8. As shown in Figure 4.8, the estimated mean values almost converge to the same values based on the asynchronous diffusion EM gradient method while the local standard EM converge to different values. The impact of estimation performance with different communication radio ranges is also considered here. As seen in Figure 4.9 , the performance of both estimated mean and variance are significantly improved by increasing communication radio range. In Figure.1.10, we show the performance of the estimated mean values of each nodes with different communication radio ranges under the asynchronous network setting. When radio range $r = 0$ as shown in Figure 4.10 (a) the network becomes an unconnected network, each node only can perform local standard EM without communication with other neighbour nodes, which leads to a noisy estimation result. It can be seen from Figure 4.10 (b) and (c), the estimated mean values in all nodes are very close to their true mean values (the true values are $\boldsymbol{\mu}_1 = [0.7, 0.7]'$, $\boldsymbol{\mu}_2 = [0.5, 0.5]'$, $\boldsymbol{\mu}_3 = [0.3, 0.3]'$). By



(a) Local standard EM algorithm without diffusion for 2D Gaussian mixture model



(b) Diffusion-based EM gradient algorithm for 2D Gaussian mixture model

Figure 4.8: Evolution of the first estimated mean value with different schemes for 10 sensor nodes using 2D Gaussian mixture model

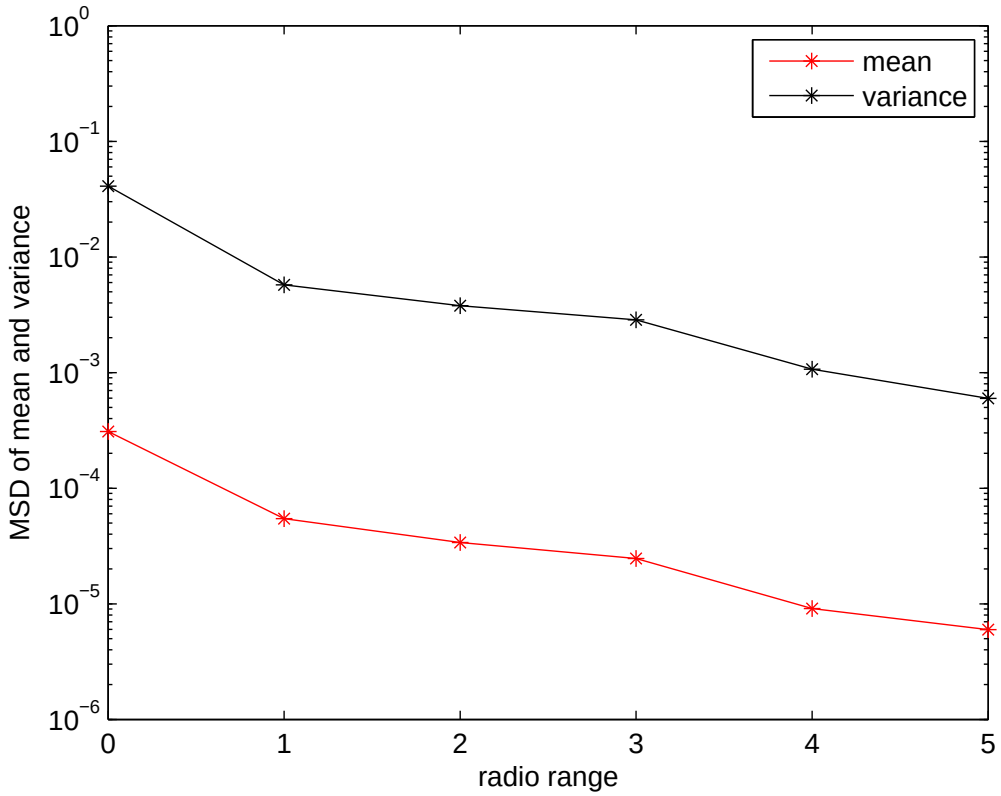
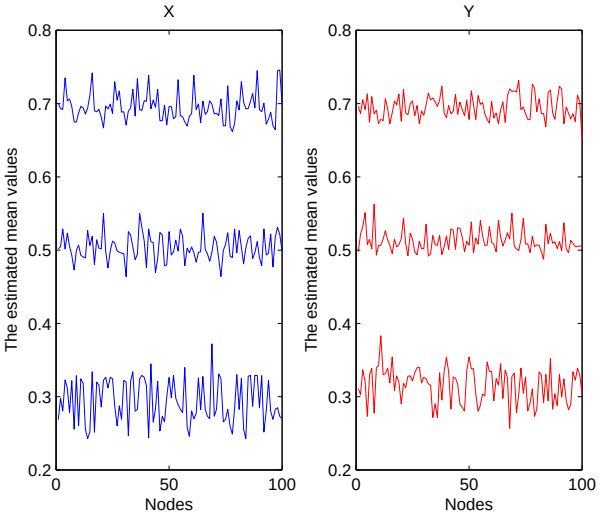


Figure 4.9: Estimation performance versus radio range (in meters) with the diffusion EM gradient algorithm ($p_m = q_m = 0.8$)

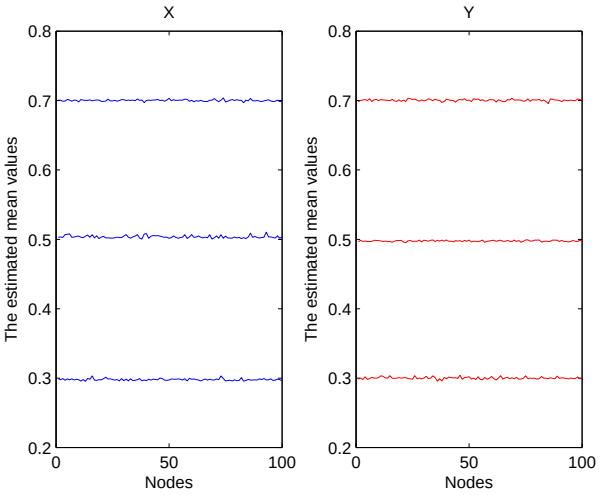
increasing communication radio ranges, the estimation performance is improved.

4.7 Chapter Summary

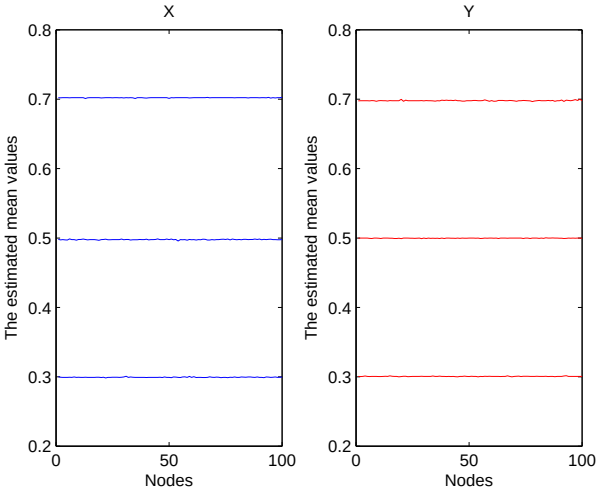
In this chapter, we proposed a diffusion based EM gradient algorithm for mixture models in asynchronous sensor networks. The proposed algorithm is characterized by local EM gradient based processing and computations of component parameters with asynchronous ATC diffusion strategies. The ability to process data locally is of particular interest to the sensor networks with computationally powerful node which require costly node-to-node communications. More importantly, with asynchronous diffusion model, each node is allowed to obtain the flexibility through their own assessment of local information without coordinated behavior over the network in comparison with synchronous strategies. Simulation results show that the proposed algorithm outperforms the local-standard EM without cooperation but as expected it degrades the MSD



(a) Unconnected nodes (Radio Range=0)



(b) Radio Range=1.5m



(c) Radio Range=2.5m

Figure 4.10: The estimated mean values versus radio ranges

performance compared to synchronous diffusion scheme. Note that EM gradient method preserves the procedure of E- step in EM algorithm and does not separate the space of components, thus distributed EM gradient algorithm cannot provide a better performance under the overlapping mixture model.

Chapter 5

Distributed quasi-Newton Method for Power System State Estimation

In this chapter, the system-wide power system state estimation (PSSE) is promising in deregulating the energy market and improving the situational awareness. In practice, the use of centralised estimator is not viable due to the high complexity, communication cost, and robustness issues. Thus, with the systematic manner, we consider the distributed PSSE approaches which are designed based on the quasi-Newton and backtracking line search. We demonstrate the effectiveness of the proposed algorithms via the IEEE 14- and 4200- buses. It is shown in the simulation results that the proposed method performs better than other algorithms when dealing with bad data and large-scale problems.

5.1 Introduction

State estimation functions as an essential part in power systems. It significantly impacts the capabilities in dispatching power, frequency management and error identifications. The system administrator can monitor the state of the power grid via state estimation methods [95]. It has become more and more important to estimate the system states with better accuracies. Researchers have made great efforts in combining new sensing techniques with the state-of-the-art state estimations. For example, in [96], the authors presented a Wide-Area Measurement System (WAMS) aided by Phasor Measurement Units (PMUs). Since the computational load is proportional to the amount of measurements, state-of-the-art systems would require the individual buses to have their own processing abilities [97]. Compared to a centralized scheme, the distributed methods have less amount of data for each estimator to process. Higher robustness

is achieved since the information of the state is stored in a distributed way. Finally, the communication overhead can be kept low by using advanced gossip based algorithms.

There have been a number of research efforts in investigating distributed state estimation approaches for power systems. The hierarchical distributed approaches estimate system states locally, exchange the information using a central processor, and combine the local estimations to give the overall estimates [98–100]. However, such methods are limited by the communication burden. In general, the distributed state estimations require the local communications rather than counting on a central processor. Recent developments in fully distributed methods include: leveraging the matrix decompositions [101, 102]; employing the alternating direction method of multipliers (ADMM) method [49]; and information filter-based techniques [103]. The matrix decomposition methods in [101, 102] give no guarantee on the convergence of the distributed state estimates. The ADMM approach in [49] guarantees the asymptotic convergence. However, the Lagrange multipliers require extra memory and asynchronous configurations can be troublesome, which limit the use of ADMM methods. The method proposed in [103] guarantees the convergence, but the required iterations scale linearly with the scale of the network. Asymptotically convergent approaches can be particularly useful to deal with large-scale networks especially when the convergence rate is independent of the scale of the network.

In [6, 7, 12], the authors proposed the gossip-based algorithms for complete distributed state estimations. In particular, the method presented in [6] is a first order approach driven by the diffusion strategy in [7]. Although the first order approaches are simple, their developments are hampered by the slow convergence rate. However, the Newton-type methods usually have quadratic convergence. A gossip-based Gauss-Newton method was developed in [12] to solve the general nonlinear least squares problem and applied to the power system state estimation. The Gauss-Newton method only exploits the presence of first-order information of Hessian, and thus requires the cost function to be zero or a small residual. However, the presence of bad data will result in a large residual in power system, which cannot be neglected during the estimation process. Such situation can no longer be handled by Gauss-Newton methods efficiently. By contrast, quasi-Newton methods are more efficient under these conditions, approximated Hessian can preserve second order information, which allows our method to reduce the impact of bad data on the state estimates.

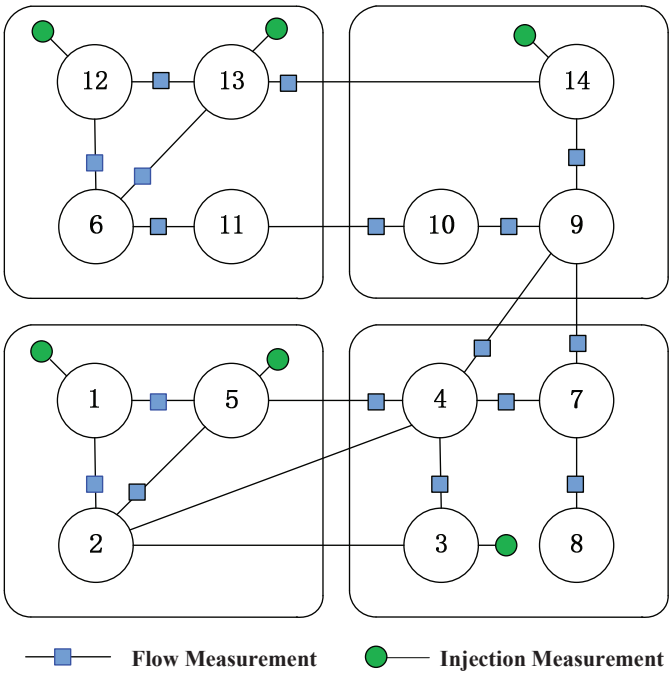


Figure 5.1: IEEE 14 bus system partitioned into $I = 4$ control areas

With this context, we reformulate the state estimation problem and propose a distributed quasi-Newton method (DQN) for wide-area PSSE. Similar with [12],and employ the multi-agent gossip-based scheme to describe the network communications. Under this scheme, the state of each agent (control area) can be estimated by using the local information and a limited information exchange with neighbour areas, for which the fusion center is not necessary. The agents can only preserve their own states. This has advantages in both communication efficiency and storages, address the large residual or bad data problem [104, 105].

In Section 5.2, we formulate the power system state estimation into a (non)linear least square (LS) problem. The details of the proposed distributed quasi-Newton method are presented in Section 5.4. In Section 5.5, we provide the convergence analysis and finally in Section 5.6, the numerical simulations are conducted to show the performance of our approach.

5.2 Problem Formulation

A multi-area power network can be conveniently expressed as an undirected graph $(\mathcal{N}, \mathcal{E})$, where the set of the vertices $\mathcal{N} \triangleq \{1, \cdots, B\}$ denotes buses and the edge set $\mathcal{E}$ represents the transmission lines that connect the buses. The power system state is normally defined as the collection of the voltages (containing both phase and magnitude information) at all buses, $\mathbf{x} = [\mathbf{\Theta}^T, \mathbf{V}^T]^T$ with $\mathbf{\Theta} \triangleq [\theta_1, \cdots, \theta_B]^T$ being the phase vector and $\mathbf{V} \triangleq [V_1, \cdots, V_B]^T$ the magnitude vector. The whole network can be divided into I non-overlapping areas, each governed by a control cite, which gathers the local measurements taken at the corresponding area and is allowed to communicate with its neighboring areas. Fig. **??** shows a concrete example where the network is partitioned into $I = 4$ regions. Apparently, the local measurements available to one control cite is insufficient for it to estimate the total system state. Therefore, in this work we study how to design the cooperation process between the multiple areas so that a distributed estimation of the global state can be efficiently implemented.

We consider the traditional measurement system, SCADA (supervisory control and data acquisition), which provides measurements on both power injections at some of the buses and on power flows along some of the transmission lines. Since in SCADA system, the measurements update rate is around once 2-6 seconds, which is relatively a long period of time compared with the communication delay between different cites, a static setting is considered in this chapter, i.e. measurement set is separated into different snapshots and each run of state estimation process is based upon the most recent one. The measurement model can therefore be represented as:

$$\mathbf{t}_i = \mathbf{h}_i(\mathbf{x}) + \mathbf{e}_i \tag{5.1}$$

where $\mathbf{e}_i$ denotes measurement noise at the i^{th} sensor as well as some other uncertainties, such as the modeling inaccuracy, and $\mathcal{I} = \{1, \cdots, I\}$ where I is the number of control sites. We further define $M = 2B$ as the dimension of the system state. In general, the observation function $\{\mathbf{h}_i(\mathbf{x})_{i=1}^I\}$ should be nonlinear. It is only in some special cases, such as when PMU measurements are considered, that the observation function can be linear. In this chapter, the general case is studied. By stacking the local measurements together, the global expression is

shown as

$$\mathbf{t} = \mathbf{h}(\mathbf{x}) + \mathbf{e}, \tag{5.2}$$

where $\mathbf{h}(\mathbf{x}) = [\mathbf{h}_1^T, \cdots, \mathbf{h}_I^T]^T, \mathbf{e} = [\mathbf{e}_1^T, \cdots, \mathbf{e}_I^T]^T$. A weighted least squares problem related to this global representation can be written as

$$\hat{\mathbf{x}} = \min_{\mathbf{x}\in\mathbf{X}} J(\mathbf{x}) = (\mathbf{t} - \mathbf{h}(\mathbf{x}))^T R^{-1} (\mathbf{t} - \mathbf{h}(\mathbf{x})) \tag{5.3}$$

where $R = cov([\mathbf{e}_1, \cdots, \mathbf{e}_N]^T)$ and $\mathbf{X} \doteq \{\theta_n \in [-\theta_{max}, \theta_{max}], V_n \in [0, V_{max}], n \in \mathcal{N}\}$, with $\theta_{\max}$ and $V_{\max}$ being the phase angle and voltage limit. According to [12], problem (5.3) is equivalent to the maximum likelihood estimation for (5.2), under the assumption that the measurement errors at different regions are gaussian and uncorrelated with each other, i.e. $R = diag(R_1, \cdots, R_I)$ with R_i being the covariance matrix for the measurement error at the i^{th} region. Since R is block diagonal matrix, problem (5.3) can be reformulated to facilitate a distributed implementation,

$$\hat{\mathbf{x}} = \min_{\mathbf{x}\in\mathbf{X}} J(\mathbf{x}) = \sum_{i=1}^{I} \|\bar{\mathbf{t}}_i - \bar{\mathbf{h}}_i(\mathbf{x})\|^2, \tag{5.4}$$

with $\bar{\mathbf{t}}_i = R_i^{-\frac{1}{2}} \mathbf{t}_i$ and $\bar{\mathbf{h}}_i = R_i^{-\frac{1}{2}} \mathbf{h}_i$. Both problem (5.3) and its distributed version (5.4) are essentially non-linear least squares problems. For centralized processing structure, Newton type algorithms are typically used to search for the stationary point because of their faster convergence rate than the first-order methods such as gradient-descent method and ADMM. In the next section, we introduce the centralized approach for solving problem (5.4), using a particular type of quasi-Newton algorithm.

5.3 Centralized quasi-Newton Algorithm

A multi-agent network previously illustrated through Fig. 5.1 is considered, where there are I distributed agents, and the i^{th} agent only knows a subset function $\mathbf{f}_i(\mathbf{x}) \doteq \bar{\mathbf{z}}_i - \bar{\mathbf{h}}_i(\mathbf{x}) : \mathbb{R}^M \to \mathbb{R}^{N_i}, i \in \mathcal{I}$,

$$\mathbf{f}_i(\mathbf{x}) = \left[f_1(\mathbf{x})^T, \cdots f_{N_i}(\mathbf{x})^T\right]^T. \tag{5.5}$$

Given the Jacobian $\mathbf{J}_i = \partial \mathbf{f}_i(\mathbf{x})/\partial \mathbf{x}$, the gradient of function $\|\mathbf{f}_i\|^2$ at $\mathbf{x}$ is denoted as $F_i(\mathbf{x}) = \mathbf{J}_i(\mathbf{x})^T \mathbf{f}_i(\mathbf{x})$. Since the global objective function can be rewritten as

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}\in\mathbf{X}} \sum_{i=1}^{I} \|\mathbf{f}_i(\mathbf{x})\|^2 \tag{5.6}$$

where individual agent only gets access to the partial information of the global cost function, the gradient of the global function can be computed as

$$F(\mathbf{x}) = \sum_{i=1}^{I} F_i(\mathbf{x}) = \sum_{i=1}^{I} \mathbf{J}_i(\mathbf{x})^T \mathbf{f}_i(\mathbf{x}). \tag{5.7}$$

In this chapter, we are interested in the BFGS (broyden-fletcher-goldfarb-shanno) quasi-Newton algorithm [106] due to its wide applications and robust performance. According to BFGS algorithm, with a properly chosen initial point $\mathbf{x}^0$ as well as a positive-definite matrix $\mathbf{H}^0$, the searching for the stationary point of problem (5.6) can be established by iteratively computing the following terms,

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \alpha \mathbf{H}^k F(\mathbf{x}^0), \tag{5.8}$$

$$\mathbf{H}^k = (I - \rho^k \mathbf{z}^k \mathbf{y}^{kT})\mathbf{H}^{k-1}(I - \rho^k \mathbf{y}^k \mathbf{z}^{kT}) + \rho^k \mathbf{z}^k \mathbf{z}^{kT}, k \geq 0, \tag{5.9}$$

where $\mathbf{y}^k = \hat{F}(\mathbf{x}^k) - \hat{F}(\mathbf{x}^{k-1}), \mathbf{z}^k = \mathbf{x}^k - \mathbf{x}^{k-1}, \rho^k = 1/(\mathbf{y}^{kT}\mathbf{z}^k)$ and α is a fixed value that controls the length of the searching step. The stopping criterion for convergence is typically set by checking the difference between the objective function values in the present and the last iterations or by simply assuming a maximum limit on the iteration number.

5.4 Distributed quasi-Newton Process

Motivated by the superior convergence behavior of the quasi-Newton method for large-scale optimization, we propose a distributed quasi-Newton method that combines quasi-Newton iterations with a network consensus process in this section.

Solving the minimization problem (5.6) in a distributed manner is challenging. The information on the global objective function is required for the computation of the searching step, as discussed in

the previous section. However, in our setting, each agent can only access a part of the global information. To obviate this problem, we augment the quasi-Newton searching with gossip process, which would disseminate local information across the network. We hope that by doing so the distributed process would behave similarly as its centralized version.

5.4.1 Network Exchange Model

Gossip process is used to disseminate local information. Essentially, an agreement among all agents is reached, to a certain degree of accuracy, via proper local information exchanges prescribed in the gossip algorithm. In this section, we first brief the data exchange model before introducing some assumptions that the gossip algorithm used in this chapter is built upon.

Since we assume that the data exchanges are synchronized among the agents, we can denote the epoch for the data exchanges between the k^{th} and the $k+1^{th}$ local iteration as $[\tau_k, \tau_{k+1})$. During the epoch, each agent is allowed to only communicate with its neighboring nodes. The network topology can be modeled as a time-varying graph $\mathcal{G}_{k,t} = (\mathcal{I}, \mathcal{E}_{k,t})$, where t is the counter for the data exchanges of the gossip process. The network topology is therefore assumed to be stationary only within a single exchange stage in the gossip process. The node set corresponds to the area set and is denoted also as $\mathcal{I} = \{1, \cdots, I\}$. The edges $\{i, j\} \in \mathcal{E}_{k,t}$ correspond to the available communication links used for data exchanges. The adjacency matrix related to the graph is denoted as $\mathbf{A}^k(t) = [A_{i,j}^{(k,t)}]_{I \times I}$

$$\mathbf{A}_{i,j}^{(k,t)} = \begin{cases} 1, & \{i,j\} \in \mathcal{E}_t^k \\ 0, & \text{otherwise} \end{cases} \tag{5.10}$$

A connected graph $\mathcal{G}_{k,\infty} = \{\mathcal{I}, \cup_{t'=t}^{\infty} \mathcal{E}_{t'}^k\}$ for all $t \geq 0$ within the k^{th} update is defined such that there exists an integer $L \geq 1$ which, for each pair of $\{i, j\}$, satisfies

$$\{i,j\} \in \bigcup_{t'=0}^{L-1} \mathcal{E}_{t'+t}^k. \tag{5.11}$$

Define the weight matrix $\mathbf{\Phi}^k(t) = [\Phi_{i,j}^k(t)]_{I \times I}$ for network, where $[\Phi_{i,j}^k(t)]$ is non-zero entry of matrix $\mathbf{\Phi}^k(t)$ if and only if $\{i, j\} \in \mathcal{E}_t^k$. To ensure that the exchanges happen between adjacent

agents, we require that $\mathbf{\Phi}^k(t)$ is symmetric and doubly stochastic for any k and t. Furthermore, with the $i, j \in \mathcal{I}$, we assume there exists a $0 < \eta < 1$ such that

1) $\Phi_{i,j}^k(t) \geq \eta$ for all $k > 0$ and $t > 0$

2) $\Phi_{i,j}^k(t) \geq \eta$ for all $k > 0$ and $t > 0$ if $\{i, j\} \in \mathcal{E}_t^k$

3) $\Phi_{i,j}^k(t) = \eta$ for all $k > 0$ and $t > 0$ if $\{i, j\} \in \mathcal{E}_t^k$

Gather the local information in one single vector $\mathcal{W}^k(t) \triangleq [\mathcal{W}_1^k(t), \cdots, \mathcal{W}_I^k(t)]$, so that we can write the network exchange explicitly as

$$\mathcal{W}^k(t) = [\mathbf{\Phi}^k(t) \otimes \mathbf{I}_{I_\mathcal{W}}]\mathcal{W}^k(t-1), 1 \leq t \leq t_k, \tag{5.12}$$

where $\mathbf{I}_{I_\mathcal{W}}$ is the identity matrix and $I_\mathcal{W}$ equals to the length of the local information exchanged at agent i, $\mathcal{W}_i^k(t)$ (in our case, $I_\mathcal{W} = M$ for both the exchanges of state variables and local gradients) and t_k is the number of exchanges during $[\tau_k, \tau_{k+1})$.

In general, the weight matrix $\mathbf{\Phi}^k(t)$ is time-varying. However, we only consider the special case of the general model, i.e. Coordinated Static Exchange [107, 108] in which each agent collects the messages from the neighbourhood, and updates parameters based on a static weight matrix Φ. This network can satisfy the fully connected condition with

$$\mathbf{A} = \mathbf{I}_I - \mathbf{1}_I\mathbf{1}_I^T \tag{5.13}$$

where $\mathbf{1}_I$ is an I-dimensional all-one vector. In most CSE protocol based gossip network, the weight matrix is constructed by Laplacian matrix

$$\mathbf{L} = \mathbf{D} - \mathbf{A} \tag{5.14}$$

where $\mathbf{D} = \text{diag}(\mathbf{A}\mathbf{1}_I)$ is the degree matrix and

$$\mathbf{\Phi} = \mathbf{I}_I - \beta\mathbf{L} \tag{5.15}$$

where $\beta = 1/max(\mathbf{A}\mathbf{1}_I)$.

Lemma 1 *[Proposition 1, [109]] Let connectivity and stochastic weights assumptions hold. The entries of the matrix product* $\prod_{t'=0}^{t} \mathbf{\Phi}^k(t')$ *converge to* $1/I$ *with a geometric rate uniformly with respect to* $i, j \in \mathcal{I}$*, and* k

$$\Big|\Big[\prod_{t'=0}^{t} \mathbf{\Phi}^k(t')\Big]_{i,j} - \frac{1}{\mathcal{I}}\Big| \leq 2\Big(\frac{1+\eta^{-L_0}}{1-\eta^{L_0}}\Big)(1-\eta^{L_0})^{t/L_0} \tag{5.16}$$

where $L_0 = (1-I)L$ *and* L *bound the intercommunication interval ensuring graph connectivity.*

The limit of the weight matrix product exists

$$\lim_{t\to\infty} = \prod_{t'=0}^{t} \mathbf{\Phi}^k(t') = \mathbf{1}_N \mathbf{1}_N^T / \mathbf{I} \tag{5.17}$$

and thus

$$\lim_{t\to\infty} \mathcal{W}_i^k(t) = \frac{1}{I}\sum_{i=1}^{I} \mathcal{W}_i^k(0), k = 1, 2, \cdots . \tag{5.18}$$

5.4.2 Local Update Process

To start with, an initial state variable $\mathbf{x}_i^0$ and an initial approximation for the inverse of Hessian matrix $\mathbf{H}_i^0$, need to be set at each agent. For reasons that will be clear later, before any local iteration k, a gossip process is implemented to compute the average state, denoted as

$$\bar{\mathbf{x}_i}^k(l_k) \approx \frac{1}{I}\sum_{i=1}^{I} \mathbf{x}_i^k, \tag{5.19}$$

where l_k is the number of gossip exchange. We assume that all the agents are synchronized so that the data exchange happens in an synchronous way. The deviation of $\bar{\mathbf{x}_i}^k(l_k)$ from the real average is related to both the states $\mathbf{x}_i^k$ and l_k, and will be discussed with more details in the next section. Now $F_i(\bar{\mathbf{x}_i}^k(l_k))$ can be computed at each agent and the average of the gradients,

$$\hat{F}(\bar{\mathbf{x}_i}^k, l_k^{'}) \approx \frac{1}{I}\sum_{i=1}^{I} F_i(\bar{\mathbf{x}_i}^k) \tag{5.20}$$

can be similarly obtained by another gossip process, where $l_k^{'}$ is the gossip exchange number. As we later show, the values of l_k and $l_k^{'}$ have varied degrees of influence on the distributed algorithm's

convergence property.

After updating (except for the first iteration, i.e. $k = 0$) the approximation for the inverse of Hessian matrix according to

$$\mathbf{H}_i^k = (I - \rho_i^k \mathbf{z}_i^k \mathbf{y}_i^{kT})\mathbf{H}_i^{k-1}(I - \rho_i^k \mathbf{y}_i^k \mathbf{z}_i^{kT}) + \rho_i^k \mathbf{z}_i^k \mathbf{z}_i^{kT}, \tag{5.21}$$

where $\mathbf{y}_i^k = \hat{F}(\bar{\mathbf{x}}_i{}^k) - \hat{F}(\bar{\mathbf{x}}_i{}^{k-1})$, $\mathbf{z}_i^k = \bar{\mathbf{x}}_i{}^k - \bar{\mathbf{x}}_i{}^{k-1}$, $\rho_i^k = 1/(\mathbf{y}_i^{kT}\mathbf{z}_i^k)$, the following local iteration is then implemented at each agent,

$$\mathbf{x}_i^{k+1} = \bar{\mathbf{x}}_i{}^k - \alpha_i \mathbf{H}_i^k \hat{F}(\bar{\mathbf{x}}_i{}^k), i \in \mathcal{I}, \tag{5.22}$$

where α_i is used to control the size of the searching step. To simplify the analysis, we fix α_i to be 1 at all agents.

From the above description, it can be seen that only state variables and first-order information are required to be exchanged between the nodes, while the second-order information is locally estimated. More importantly, no matrix inverse is required, which reduce the computational burden significantly compared with the Gossip-based Gauss Newton method in [12].

The whole procedure of DQN method is summarized in the Algorithm 1. In the next section, we will provide its convergence analysis.

Algorithm 1 Distributed BFGS Algorithms

1: **given** initial variables $\mathbf{x}_i^0$, $\mathbf{H}_i^0$ at all agents $i \in \mathcal{I}$, as well as proper weight matrix $\mathbf{\Phi}$ that satisfies Assumption 1 and 2.
2: **set** $k = 0$.
3: **repeat**
4: **network exchanges**: Agents exchange their local state variables according to (5.12) with $t_k = l_k$. After $F_i(\bar{\mathbf{x}}_i{}^k(l_k))$ is computed at each agent, these local graients are exchanged according to (5.12) with $t_k = l_k^{'}$ and the estimate of the global gradient $\hat{F}(\bar{\mathbf{x}}_i{}^k)$ is obtained at each node.
5: **local update**: If $k \geq 1$, update the approximated inverse Hessian matrix as (5.21). For each $i \in \mathcal{I}$. agent i updates its local variables as (5.22).
6: **set** $k = k + 1$
7: **until** $\|\mathbf{x}_i^{k+1} - \mathbf{x}_i^k\| \leq \epsilon$ or $k = K$.
8: **set** the local estimate as $\hat{\mathbf{x}}_i = \mathbf{x}_i^k$.

5.5 Convergence Analysis

In this section, we analyze the convergence of the DQN algorithm (summarized in Algorithm 1). Local convergence instead of global convergence property is studied here since the objective function in our problem formulation is not guaranteed to be convex and for non-convex functions, a global convergence proof is not found even for the centralized version of quasi-Newton algorithm in existing literature. We mainly develop the local convergence analysis first used in [106] and study the impact of the distributed implementations on the DQN's convergence property.

5.5.1 Gossip Errors Analysis

One of the noticeable differences of DQN from its centralized version is that the values of state variables and gradients utilized at the iteration equations (5.21),(5.22) are deviated from the real values since the gossip exchange number is finite. We denote such deviations as gossip errors. To facilitate our later analysis of the local convergence, we bound the gossip errors by making some reasonable assumptions on the objective functions.

Lemma 2 *[106] Assume the gradient of the global objective function, $F : \mathbb{R}^M \rightarrow \mathbb{R}^M$ is differentiable in the open set $D \subset \mathbb{X}$, and for some minima x^* in D, $p > 0$ and $K > 0$,*

$$\left\| F^{'}(x) - F^{'}(x^*) \right\| \leq K \|x - x^*\|^p. \tag{5.23}$$

The following inequality is satisfied for every u, v in D,

$$\left\| F(v) - F(u) - F^{'}(x^*)(v - u) \right\| \tag{5.24}$$

$$\leq K max\{\|v - x^*\|^p, \|u - x^*\|^p\} \|u - v\|. \tag{5.25}$$

If $F^{'}(x^*)$ is further invertible, there exist $\epsilon > 0$ and $\rho > 0$ such that $max\{\|u - x^*\|, \|v - x^*\|\} \leq \epsilon$ leads to $u, v \in D$ and

$$(1/\rho)\|v - u\| \leq \|F(v) - F(u)\| \leq \rho \|v - u\|. \tag{5.26}$$

Denote the gossip errors for the local state and the gradient exchanges respectively as $p_i(l_k), q_i(l_k^{'})$ at agent i and the k^{th} iteration. To ease the expression, we use $\hat{F}_i^k(t)$ in place of $\hat{F}(\bar{\mathbf{x}_i}^k, t)$ hereafter. Define $\mathcal{W}_F^k(t) = \left[\hat{F}_1^k(t), \cdots, \hat{F}_I^k(t)\right], \mathcal{W}_x^k(t) = \left[\bar{\mathbf{x}_1}^k(t), \cdots, \bar{\mathbf{x}_I}^k(t)\right], \bar{\mathcal{W}}_F^k = [\mathbf{1}_I \mathbf{1}_I^T \otimes \mathbf{I}_{I_M}]\mathcal{W}_F^k(0)/I$ and $\bar{\mathcal{W}}_x^k = [\mathbf{1}_I \mathbf{1}_I^T \otimes \mathbf{I}_{I_M}]\mathcal{W}_x^k(0)/I$. Note that $\mathcal{W}_x^k(0) = \left[\mathbf{x}_1^k, \cdots, \mathbf{x}_I^k\right], \mathcal{W}_F^k(0) = \left[F_1(\bar{\mathbf{x}_1}^k), \cdots, F_I(\bar{\mathbf{x}_I}^k)\right]$. By the above definitions, we would have

$$\mathcal{W}_F^k(t) - \bar{\mathcal{W}}_F^k = \begin{bmatrix} q_1(t) \\ \vdots \\ q_I(t) \end{bmatrix}, \mathcal{W}_x^k(t) - \bar{\mathcal{W}}_x^k = \begin{bmatrix} p_1(t) \\ \vdots \\ p_I(t) \end{bmatrix}. \tag{5.27}$$

Lemma 3 *Let the assumptions made in Lemma 2 hold and the gradients of the local functions are upper bounded in D. The following inequalities are then satisfied for $i \in \mathcal{I}$,*

$$\|p_i(t)\| = \left\| \bar{\mathbf{x}_i}^k(t) - \sum_{j=1}^{I} \mathbf{x}_i^k \right\| \leq C_1(t_1), 0 < t_1 < l_k, \tag{5.28}$$

$$\left\| \hat{F}_i^k(t) - F(\bar{\mathbf{x}_i}^k) \right\| \leq 2C_1(l_k) + C_2(t_2), 0 < t_2 < l_k^{'}, \tag{5.29}$$

where C_1 and C_2 are both positive reals that decrease exponentially with the number of the gossip exchanges.

Proof: See Appendix C

5.5.2 Local convergence analysis

One prominent feature of quasi-Newton algorithm that differentiate it from other unconstrained optimization algorithms is that its second-order information is updated recursively. To analyze the local convergence property, we first characterize this recursive process by establishing the following lemma.

Lemma 4 *Let the gradient of the global function, F, satisfy the assumptions made in lemma 3 and*

further let the hessian matrix at the minima, $F^{'}(\mathbf{x}^*)$, be symmetric and positive definite. Then there exists an neighborhood $\mathcal{N} = \mathcal{N}_1 \times \mathcal{N}_2$ of $(\mathbf{x}^*, F^{'}(\mathbf{x}^*)^{-1})$ such that for each $(\bar{\mathbf{x}}^{k-1}, \mathbf{H}^{k-1}) \in \mathcal{N}$, the updated Hessian inverse $\mathbf{H}^k$, as defined in (5.21)*, satisfies*

$$\begin{aligned}\left\|\mathbf{H}^k - F^{'}(\mathbf{x}^*)\right\|_M \leq \left[1 + \lambda_1 \max\left\{\|\bar{\mathbf{x}}^k - \mathbf{x}^*\|^p, \|\bar{\mathbf{x}}^{k-1} - \mathbf{x}^*\|^p\right\}\right] \\ \left\|\mathbf{H}^{k-1} - F^{'}(\mathbf{x}^*)\right\|_M + \lambda_2 \max\left\{\|\bar{\mathbf{x}}^k - \mathbf{x}^*\|^p, \|\bar{\mathbf{x}}^{k-1} - \mathbf{x}^*\|^p\right\},\end{aligned} \tag{5.30}$$

where λ_1, λ_2 are non-negative constants, and $\|\cdot\|_M$ is certain Matrix norm. Note that we omit i in $\bar{\mathbf{x}}_i{}^k$ and $\mathbf{H}_i^k$ for ease of expression.

Proof: See Appendix D

Now that the inverse of the hessian matrix is bounded through inequality (5.30), the local convergence result can be well established. We conclude the main result in the following theorem.

Theorem 2 *Let the assumptions made in Lemma 4 be satisfied by the gradient function, F. Then there exists a neighborhood $\mathcal{N} = \mathcal{N}_1 \times \mathcal{N}_2$ of $(\mathbf{x}^*, F^{'}(\mathbf{x}^*)^{-1})$, such that for each $r \in (0, 1)$, if the initial states satisfy the following condition*

$$\|\mathbf{x}_i^0\| < \epsilon(r)/2, \|\mathbf{H}_i^0 - F^{'}(\mathbf{x}^*)\|_M < \delta(r), \tag{5.31}$$

where $\epsilon(r), \delta(r)$ are positive constants, then the sequences of $\bar{\mathbf{x}}_i{}^k, \mathbf{H}_i^k, k > 0$ are well defined in $\mathcal{N}$ and $\bar{\mathbf{x}}_i{}^k$ converges to the local minimum $\mathbf{x}^*$ in the following manner

$$\|\bar{\mathbf{x}}_i{}^{k+1} - \mathbf{x}^*\| \leq r^{'}\|\bar{\mathbf{x}}_i{}^k - \mathbf{x}^*\|, \tag{5.32}$$

where $r^{'} \in (0, 1)$.

Proof: See Appendix E

5.6 Simulation Results

In this section, we conduct experiments to compare the existing distributed quasi-Newton algorithm performance to those of the GGN algorithm [12] and ADMM algorithm [49]. The distributed estimate in each are $\{\hat{V}_{i,n}^k\}_{n=1}^N, \{\hat{\theta}_{i,n}^k\}_{n=1}^N$ at each local update, the Mean Squared Error (MSE) with respect to the voltage magnitude and the phase at the i-th site are

$$\mathbf{MSE}_{V,i}^k = \sum_{n=1}^{N}((\hat{V}_{i,n}^k) - \bar{V}_n)^2 \tag{5.33}$$

$$\mathbf{MSE}_{\theta,i}^k = \sum_{n=1}^{N}((\hat{\theta}_{i,n}^k) - \bar{\theta}_n)^2 \tag{5.34}$$

In addition, the metric used in our comparisons are the cost function in (7),

$$\mathbf{Val}_k = \sum_{i=1}^{I} \|\mathbf{z}_i - \mathbf{H}_i \mathbf{x}_i^k\|^2 \tag{5.35}$$

which is evaluated using the decentralized estimates at each updates, and the global MSE is given by

$$\mathbf{MSE}_V^k = \frac{1}{I}\sum_{i=1}^{I} \mathbf{MSE}_{V,i}^k \tag{5.36}$$

$$\mathbf{MSE}_\theta^k = \frac{1}{I}\sum_{i=1}^{I} \mathbf{MSE}_{\theta,i}^k \tag{5.37}$$

In the simulations we used the MATPOWER 5.1 [110] test case IEEE-14 (N=14) system, and took the load form from Power Systems Test Case Archive, University of Washington [111], and scale the base load from MATPOWER upon load buses, and selected the work program as Optimal Power Flow to give the generation dispatch for that instant. The initialization for the voltage magnitudes and phases are 1 and 0, respectively.

Sensor observations are generated by introducing independent Gaussian errors $\{\mathbf{e}\} \sim \mathcal{N}(0, \sigma^2)$ where $\sigma^2 = 10^{-6}$. The IEEE 14-bus grid is partitioned into 4 areas as depicted in Figure 5.1. The control areas contain $I_1 = 3$, $I_2 = 4$, $I_3 = 4$ and $I_4 = 4$ buses, respectively.

Table 5.1: Execution Time and Iterations in Case A

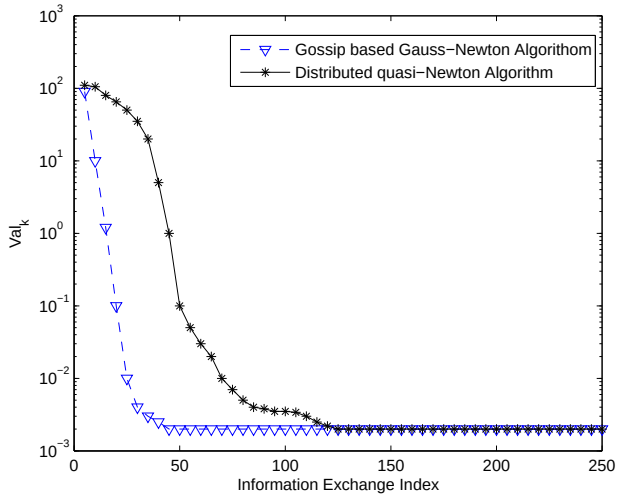
IEEE 14-bus	GGN	DQN	Centralized Estimation
Computation times (s)	0.0287	0.0236	0.0228
Iterations	48	122	Undefined

5.6.1 Case A: Comparison with GGN without Bad Data

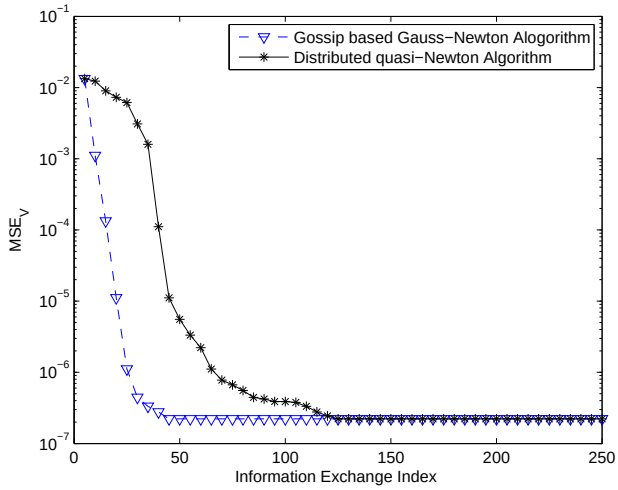
Here we present how the distributed quasi-Newton scheme performs against the Gossip based Gauss Newton algorithm for PSSE in [12]. These distributed network algorithms proceed at each tth gossip exchange, and run the with $t = 10$ gossip exchanges for each update. The comparison is made on the same time scale based on the number of exchanges. By using the $t = 10$ gossip exchanges between every two descent updates $k = 1, \cdots, 50$, thus we have the total number of 500 exchanges per snapshot. We assume that all sensors are connected, which leads to the adjacency matrix $\mathbf{A} = \mathbf{I}_I - \mathbf{1}_I\mathbf{1}_I^T$, and the weight matrix is constructed with the Laplacian $\mathbf{L} = \text{diag}(\mathbf{A}\mathbf{1}_I) - \mathbf{A}$ and $\Phi = \mathbf{I}_I - \omega\mathbf{L}$ with $\omega = \beta/\max(\mathbf{A}\mathbf{1}_I)$ where $\beta = 0.5$. We choose the step-size for Gossip based Gauss Newton algorithm as $\alpha_{GGN} = 0.5$. It can be seen from Figure 5.2(a)to(c), GGN algorithms converge faster than the proposed DQN method, because that GGN algorithm can achieve the convergence rate of centralized Gauss-Newton Algorithm, which converge quadratically when the system error or residual is very small. On the other hand, distributed quasi-Newton is a Newton-like algorithm that converges superlinearly. However, based on the comparison in Table 5.1, GGN method require to compute the inversion of Hessian matrix with complexity order of $\mathcal{O}(N^3)$, where N is the matrix size. This results in high computation complexity and requirements of the local processor to have capability to maintain such computations on time for exchange. In contrast, the proposed method requires an $\mathcal{O}(N^2)$ computation cost. It uses an iterative solution of approximation for the Hessian matrix and avoids calculating matrix inverse , which makes it more effective and realistic in a power system.

5.6.2 Case B: Comparison with GGN in presence of Bad Data

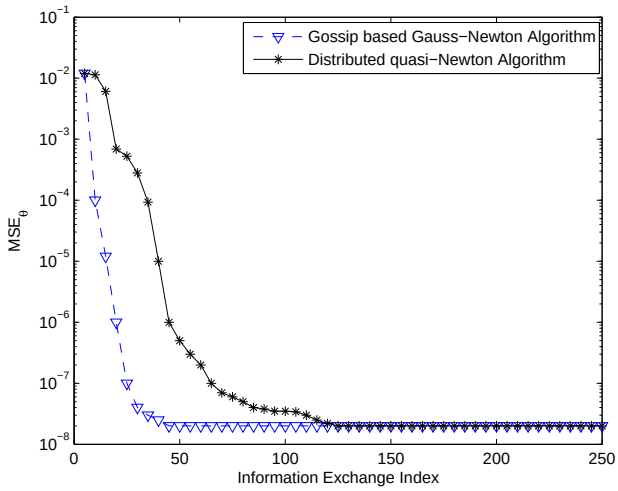
We compare our proposed method with the GGN algorithm when bad data is present. We add random Gaussian system errors $\mathbf{e}_s$ with $E(\mathbf{e}_s\mathbf{e}_s^T) = 100\sigma^2$. We examine the MSE performance of



(a) Cost Function $\mathbf{Val}_k$

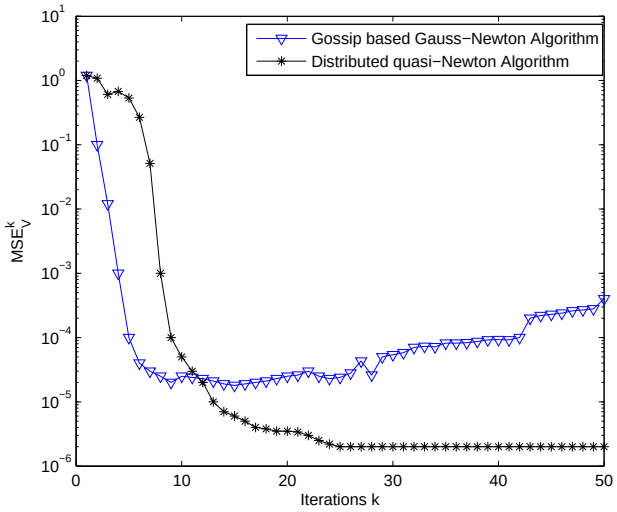


(b) $\mathbf{MSE}_V^k$

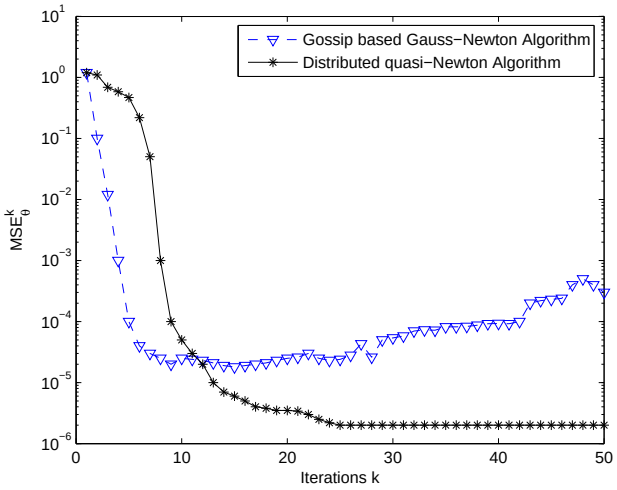


(c) $\mathbf{MSE}_\theta^k$

Figure 5.2: Comparison with GGN and distributed quasi-Newton using $t = 10$ exchange for each update



(a) $\mathbf{MSE}_V^k$



(b) $\mathbf{MSE}_\theta^k$

Figure 5.3: Comparison distributed quasi-Newton against GGN with bad data

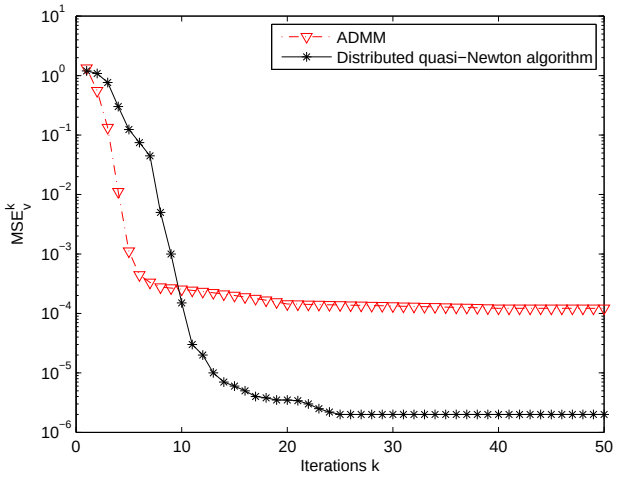
the distributed quasi-Newton method where, in each snapshot t, each agent exchange to neighbour agents 10 times on average during the interval$[\tau_k, \tau_{k+1})$ for all $k = 1, \cdots, 50$. Clearly, as shown in Figure 5.3(a) and (b), when large residual is present, caused by bad data, estimation with the GGN method fail to improve the cost function after iteration $k = 11$ in each snapshot. On the other hand this distributed quasi-Newton method degrades more gracefully. The GGN method only considers the first order term of Hessian matrix, however, for the large residual problem, second order terms cannot be neglected. By contrast, the distributed quasi-Newton method can build up the second-order derivative term for approximated Hessian with iterative process. That is the reason our method outperforms the GGN algorithm in the presence of bad data.

5.6.3 Case C: Comparison with ADMM Method in a Large-scale Power Network

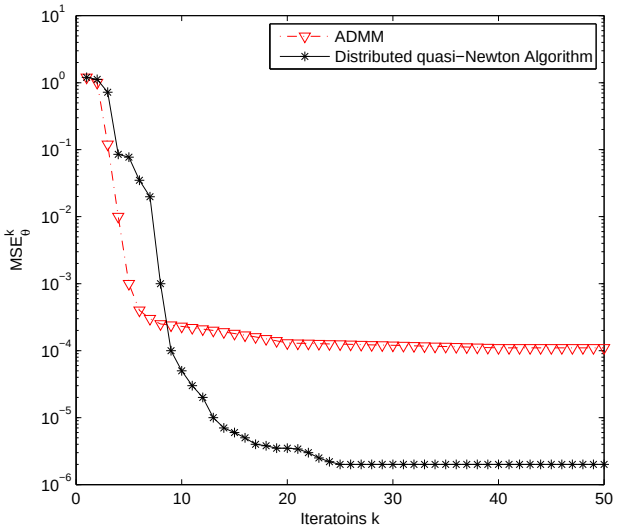
We finally compare our method to ADMM [49] using a larger power network: a 4200-bus power grid constructed using the IEEE 14- and 300-bus power grid. By assuming that 300 buses are different regions, a copy of the IEEE 14-bus grid can be used as the substitute for each of them. Moreover, we randomly choose the the terminal buses among the incident to the line areas for the IEEE 300-bus grid. Measurements and bad data are selected as the tests for IEEE 14-bus grid. The step-size for ADMM is $\alpha_{ADMM} = 0.5$. Figure 5.3(a) and (b) demonstrate the MSE plot which are averaged upon 300 areas. Observing that the distributed quasi-Newton method converges substantially faster than ADMM methods, achieving a Mean-square error of 10^{-6} less than 25 iteration, while ADMM just reaches MSE of 10^{-3} by iteration 40. Note that the IEEE 300-bus is used as the substitute of the agents in the IEEE 14-bus grid. This reserved topology of the 14 agents is also tested. It can be seen from the Figure 5.4(c) that the algorithm converged a slightly faster (around 5%) due to the looser areas coupling.

5.7 Conclusion

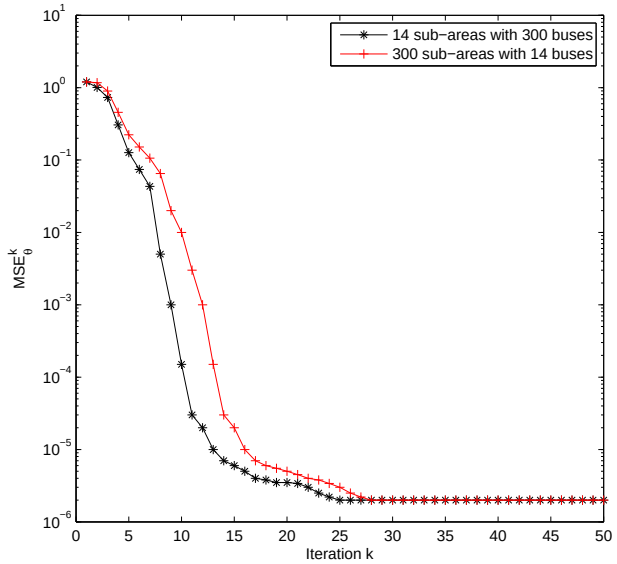
In this chapter, we proposed a distributed quasi-Newton method for hybrid power system state estimation integrating the seamlessly WAMS and SCADA measurement system, which adaptively estimated the global state vector along with a large residual. The proposed algorithm reduced the complexity of computation and maintained the property of fast convergence. In particular, only gradient information is required to disseminated over the network, which significantly lowers the communication overhead compared with other gossip-based algorithms. The numerical results proved that the proposed approach was capable of delivering accurate estimates of the entire state vector at each distributed area, even in the presence of bad data. Meanwhile, its effectiveness was demonstrated by applying this method to a large-scale power system network.



(a) $\mathbf{MSE}_V^k$



(b) $\mathbf{MSE}_\theta^k$



(c) $\mathbf{MSE}_\theta^k$

Figure 5.4: Comparison distributed quasi-Newton against ADMM in a large-scale power network

Chapter 6

Conclusions and Future Work

6.1 Summary of the Work

In this thesis, a number of innovative distributed cooperative strategies for dealing with exchange of information, asynchronous network settings and state estimation have been considered for wireless sensor and power system networks. The efficiency and convergence rate of the proposed algorithms are often the main specifications in these types of applications. The presented distributed approaches had been proven to be efficient when dealing with the statistical inference problems for multiple applications. It was also shown that the proposed algorithm can outperform existing algorithms in a variety of circumstances.

In Chapter 3, a distributed adaptive algorithm based on the component-wise EM method for Gaussian mixture model in wireless sensor networks has been presented. The distributed component-wise EM algorithm has been designed and applied into a Gaussian density estimation. In particular, the proposed algorithm operates component-wise EM procedure for local parameter estimation and exploits the incremental strategy in network updating, which can provide an improved performance in terms of convergence rate. Numerical simulation results have proved the advantages of the proposed DCEM algorithms in both well-separated and overlapped Gaussian mixture densities. The distributed DCEM algorithm is able to outperform the existing DEM and DEMM algorithms in the presence of overlap Gaussian mixtures. Note that due to the limitation of the incremental strategy, the proposed algorithm only can be implemented on a small size (less than 100 nodes) wireless sensor network.

In Chapter 4, a diffusion based EM gradient algorithm for density estimation in asynchronous wireless sensor networks has been investigated. Specifically, based on the asynchronous adapt-

then-combine diffusion strategy, a distributed EM gradient algorithm that can be applied to asynchronous wireless sensor networks has been considered. We have derived the procedure of diffusion optimization and exploited the Bernoulli model to approximate the asynchronous behaviour of the network. In comparison with existing distributed EM based estimation methods, more accurate estimates can be obtained for the proposed algorithm in the presence of asynchronous uncertainties, such as random link failures, random data arrival times and turning on or off sensor nodes for energy conservation. Simulation experiments have been conducted to illustrate that the proposed algorithms significantly outperform the consensus based strategies in terms of MSD performance under network uncertainties and imperfections.

In Chapter 5, the challenge of distributed state estimation in power system that requires low complexity and high stability in the presence of bad data and in a large scale network is addressed. A gossip based distributed quasi-Newton algorithm has been proposed for power system state estimation. In particular, we have applied the quasi-Newton method in distributed state estimation under the gossip protocol. The proposed algorithm exploits the BFGS formula to approximate the Hessian matrix to avoid calculating the inverse Hessian matrix in each control area. A distributed back track line search method has also been presented to coordinate the whole network with a suitable step-size. The simulation results for the IEEE 14 bus system and a large scale 4200 bus system have shown that the distributed quasi-Newton scheme outperforms existing algorithms in terms of MSE performance with bad data. Researchers could also implement the distributed quasi-Newton method efficiently in hardware design for a large scale power system.

6.2 Future Work

The proposed schemes in this thesis provide further potential to be implemented outside of our considered scope. Further work and in-depth analysis can be done to extend our contribution to other fields in the future.

The proposed distributed component-wise EM algorithm can be extended to a consensus based strategy. More specifically, the component-wise EM algorithm can be carried out at local sensor nodes to update estimates, and the local statistical summary can be exchanged based on a

consensus protocol among the neighbour nodes in a wireless sensor network. In addition, the important issue on stability of the DCEM method under quantization should be addressed in future.

The diffusion based EM algorithm proposed in Chapter 4 does not have analytical results in terms of steady state performance and mean square convergence. In order to demonstrate the excellent performance of the proposed algorithm, related mathematical performance analysis should be performed in the future work.

The distributed quasi-Newton method in power system state estimation only employs the consensus protocol for information exchange. However, diffusion strategies are more effective than the consensus scheme. Further research on state estimation under diffusion protocols can be carried out in the future work.

Appendix A

Proof of Theorem 1

1. Consider the EM update for the mixing proportions $\alpha_{m,j}$, from Equations (3.1), (3.3) and (3.53), it can be obtained

$$\frac{\partial \mathcal{L}_m(\boldsymbol{\theta}_m)}{\partial \alpha_{m,j}}|_{\mathcal{A}_m=\mathcal{A}_m^{[t+1,j-1]}} = \tag{A.1}$$
$$\sum_{i=1}^{N_m} \frac{\mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_{m,j}^t, \boldsymbol{\Sigma}_{m,j}^t)}{\sum_{k=1}^{j-1}\alpha_{m,k}^{t+1}\mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_{m,k}^{t+1}, \boldsymbol{\Sigma}_{m,k}^{t+1}) + \sum_{k=j}^{J}\alpha_{m,k}^{t}\mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_{m,k}^{t}, \boldsymbol{\Sigma}_{m,k}^{t})} - N_m$$

Premultiplying by $P_{\alpha_{m,j}^t}$, yields

$$P_{\alpha_{m,j}^t}\frac{\partial \mathcal{L}_m(\boldsymbol{\theta}_m)}{\partial \alpha_{m,j}}|_{\mathcal{A}_m=\mathcal{A}_m^{[t+1,j-1]}} = \tag{A.2}$$
$$\frac{1}{N_m}\sum_{i=1}^{N_m} \frac{\alpha_{m,j}^t\mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_{m,j}^t, \boldsymbol{\Sigma}_{m,j}^t)}{\sum_{k=1}^{j-1}\alpha_{m,k}^{t+1}\mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_{m,k}^{t+1}, \boldsymbol{\Sigma}_{m,k}^{t+1}) + \sum_{k=j}^{J}\alpha_{m,k}^{t}\mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_{m,k}^{t}, \boldsymbol{\Sigma}_{m,k}^{t})} - \alpha_{m,j}^t \tag{A.3}$$

The M-step formula for $\mathcal{A}$ in equation (3.31) can be rewritten as

$$\alpha_{m,j}^{t+1} = \alpha_{m,j}^t + \frac{1}{N_m}\sum_{i=1}^{N_m} w_{i,m,j}^{t+1} - \alpha_{m,j}^t \tag{A.4}$$

2. Consider update formula for the mean $\boldsymbol{\mu}_j$ which follows from (3.1) and (3.3) that

$$\frac{\partial \mathcal{L}_m(\boldsymbol{\theta}_m)}{\partial \boldsymbol{\mu}_{m,j}}|_{\boldsymbol{\mu}_{m,j}=\boldsymbol{\mu}_{m,j}^t} = \sum_{i=1}^{N_m} \frac{\alpha_{m,j}^t\mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_{m,j}^t, \boldsymbol{\Sigma}_{m,j}^t)}{\sum_{k=1}^{j-1}\alpha_{m,k}^{t+1}\mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_{m,k}^{t+1}, \boldsymbol{\Sigma}_{m,k}^{t+1}) + \sum_{k=j}^{J}\alpha_{m,k}^{t}\mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_{m,k}^{t}, \boldsymbol{\Sigma}_{m,k}^{t})} \tag{A.5}$$
$$\times (\boldsymbol{\Sigma}_{m,j}^t)^{-1}[\boldsymbol{y}_{m,i} - \boldsymbol{\mu}_{m,j}^t]$$
$$= \sum_{i=1}^{N_m} w_{i,m,j}^{t+1}(\boldsymbol{\Sigma}_{m,j}^t)^{-1}[\boldsymbol{y}_{m,i} - \boldsymbol{\mu}_{m,j}^t]$$

Premultiplying by $P_{\boldsymbol{\mu}_{m,j}^t}$, yields

$$P_{\boldsymbol{\mu}_{m,j}^t} \frac{\partial \mathcal{L}_m(\boldsymbol{\theta}_m)}{\partial \boldsymbol{\mu}_{m,j}}|_{\boldsymbol{\mu}_{m,j}=\boldsymbol{\mu}_{m,j}^t} = \frac{1}{\sum_{i=1}^{N_m} w_{i,m,j}^{t+1}} \sum_{i=1}^{N_m} w_{i,m,j}^{t+1} \boldsymbol{y}_{m,i} - \boldsymbol{\mu}_{m,j}^t \tag{A.6}$$
$$= \boldsymbol{\mu}_{m,j}^{t+1} - \boldsymbol{\mu}_{m,j}^t$$

Based on equation (3.31), it can be derived that $\sum_{i=1}^{N_m} w_{i,m,j}^{t+1} > 0$, and $\boldsymbol{\Sigma}_{m,j}^t$ is positive definite with 1-probability under the assumption of a large enough N_m (the matrix has full rank). Similarly, based on (3.32), $P_{\boldsymbol{\mu}_j^t}$ is positive definite with 1-probability. 3. The third piece of the theorem is based on the equation (3.1) and (3.3) that

$$\frac{\partial \mathcal{L}_m(\boldsymbol{\theta}_m)}{\partial \boldsymbol{\Sigma}_{m,j}}|_{\boldsymbol{\Sigma}_{m,j}=\boldsymbol{\Sigma}_{m,j}^t} = -\frac{1}{2} \sum_{i=1}^{N_m} \frac{\alpha_{m,j}^t \mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_{m,j}^t, \boldsymbol{\Sigma}_{m,j}^t)}{\sum_{k=1}^{j-1} \alpha_{m,k}^{t+1} \mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_k^{t+1}, \boldsymbol{\Sigma}_k^{t+1}) + \sum_{k=j}^{J} \alpha_{m,k}^t \mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_k^t, \boldsymbol{\Sigma}_k^t)} \tag{A.7}$$
$$\times (\boldsymbol{\Sigma}_{m,j}^t)^{-1} \{\boldsymbol{\Sigma}_{m,j}^t - [\boldsymbol{y}_{m,i} - \boldsymbol{\mu}_{m,j}^t][\boldsymbol{y}_{m,i} - \boldsymbol{\mu}_{m,j}^t]^T\} (\boldsymbol{\Sigma}_{m,j}^t)^{-1}$$
$$= -\frac{1}{2} \sum_{i=1}^{N_m} w_{i,m,j}^{t+1} (\boldsymbol{\Sigma}_{m,j}^t)^{-1} \{\boldsymbol{\Sigma}_{m,j}^t - [\boldsymbol{y}_{m,i} - \boldsymbol{\mu}_{m,j}^t][\boldsymbol{y}_{m,i} - \boldsymbol{\mu}_{m,j}^t]^T\} (\boldsymbol{\Sigma}_{m,j}^t)^{-1}$$

Based on the discussion above, the EM update formula for $\boldsymbol{\Sigma}_{m,j}^t$ can be restructured as

$$\boldsymbol{\Sigma}_{m,j}^{t+1} = \boldsymbol{\Sigma}_{m,j}^t + \frac{1}{\sum_{i=1}^{N_m} w_{i,m,j}^{t+1}} \sum_{i=1}^{N_m} w_{i,m,j}^{t+1} [\boldsymbol{y}_{m,i} - \boldsymbol{\mu}_{m,j}^t][\boldsymbol{y}_{m,i} - \boldsymbol{\mu}_{m,j}^t]^T - \boldsymbol{\Sigma}_{m,j}^t \tag{A.8}$$
$$= \boldsymbol{\Sigma}_{m,j}^t + \frac{2\boldsymbol{\Sigma}_{m,j}^t}{\sum_{i=1}^{N_m} w_{i,m,j}^{t+1}} V_{\boldsymbol{\Sigma}_{m,j}^t} \boldsymbol{\Sigma}_{m,j}^t,$$

where

$$V_{\boldsymbol{\Sigma}_{m,j}^t} = -\frac{1}{2} \sum_{i=1}^{N_m} w_{i,m,j}^{t+1} (\boldsymbol{\Sigma}_{m,j}^t)^{-1} \{\boldsymbol{\Sigma}_{m,j}^t - [\boldsymbol{y}_{m,i} - \boldsymbol{\mu}_{m,j}^t][\boldsymbol{y}_{m,i} - \boldsymbol{\mu}_{m,j}^t]^T\} (\boldsymbol{\Sigma}_{m,j}^t)^{-1} \tag{A.9}$$
$$= \frac{\partial \mathcal{L}_m(\boldsymbol{\theta})}{\partial \boldsymbol{\Sigma}_j}|_{\boldsymbol{\Sigma}_j=\boldsymbol{\Sigma}_{m,j}^t}$$

which yields

$$\boldsymbol{\Sigma}_{m,j}^{t+1} = \boldsymbol{\Sigma}_{m,j}^t + \frac{2\boldsymbol{\Sigma}_{m,j}^t}{\sum_{i=1}^{N_m} w_{i,m,j}^{t+1}} \frac{\partial \mathcal{L}_m(\boldsymbol{\theta}_m)}{\partial \boldsymbol{\Sigma}_j}|_{\boldsymbol{\Sigma}_j=\boldsymbol{\Sigma}_{m,j}^t} \tag{A.10}$$

Facilitating the definition of vec operator, $\text{vec}[\boldsymbol{ABC}] = (\boldsymbol{C}^T \otimes \boldsymbol{A})\text{vec}[\boldsymbol{B}]$, it can be derived

$$\text{vec}[\boldsymbol{\Sigma}_{m,j}^{t+1}] = \text{vec}[\boldsymbol{\Sigma}_{m,j}^{t}] + \frac{2}{\sum_{i=1}^{N_m} w_{i,m,j}^{t+1}} (\boldsymbol{\Sigma}_{m,j}^{t} \otimes \boldsymbol{\Sigma}_{m,j}^{t}) \frac{\partial \mathcal{L}_m(\boldsymbol{\theta}_m)}{\partial \boldsymbol{\Sigma}_j} |_{\boldsymbol{\Sigma}_j = \boldsymbol{\Sigma}_{m,j}^{t}} \tag{A.11}$$

Thus, $P_{\boldsymbol{\Sigma}_{m,j}^{t}}$ equal to $\frac{2}{\sum_{i=1}^{N_m} w_{i,m,j}^{t+1}} \boldsymbol{\Sigma}_{m,j}^{t} \otimes \boldsymbol{\Sigma}_{m,j}^{t}$. Furthermore, with an arbitrary matrix U, it can be derived that

$$\begin{aligned} \text{vec}[U]^T (\boldsymbol{\Sigma}_{m,j}^{t} \otimes \boldsymbol{\Sigma}_{m,j}^{t}) \text{vec}[U] &= \text{tr}(\boldsymbol{\Sigma}_{m,j}^{t} U \boldsymbol{\Sigma}_{m,j}^{t} U^T) \\ &= \text{tr}(\boldsymbol{\Sigma}_{m,j}^{t} U \boldsymbol{\Sigma}_{m,j}^{t} U^T) \\ &= \text{vec}[\boldsymbol{\Sigma}_{m,j}^{t} U]^T \text{vec}[\boldsymbol{\Sigma}_{m,j}^{t} U] \end{aligned} \tag{A.12}$$

which is valid only when $\boldsymbol{\Sigma}_{m,j}^{t} U = 0$ for all U satisfies. Given that $\boldsymbol{\Sigma}_{m,j}^{t}$ is positive definite with 1-probability and a large N_m, this condition can not be achieved. Therefore it follows from equation 3.33) and $\sum_{i=1}^{N_m} w_{i,m,j}^{t+1} > 0$.

Appendix B

Proof of Theorem 2

With (3.66), it can be obtained:

$$\begin{aligned}\mathbf{R}_{\boldsymbol{\theta}_{m,j}}^{T}\mathbf{M}_{m,j}^{DCEM}\mathbf{R}_{\boldsymbol{\theta}_{m,j}} &= \begin{bmatrix} \mathbf{I}-\mathbf{P}_{\boldsymbol{\theta}_{m,j}^{*}}\mathbf{H}_{\boldsymbol{\theta}_{m,j}^{*}} & -\mathbf{P}_{\boldsymbol{\theta}_{m,j}^{*}}\mathbf{H}_{\boldsymbol{\theta}_{l}^{*}} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \\ &= \mathbf{I}-\begin{bmatrix} \mathbf{I} \\ \mathbf{0} \end{bmatrix}\mathbf{P}_{\boldsymbol{\theta}_{m,j}^{*}}[\mathbf{H}_{\boldsymbol{\theta}_{m,j}^{*}} \;\; \mathbf{H}_{\boldsymbol{\theta}_{m,l}^{*}}] \\ &= \mathbf{I}-\begin{bmatrix} \mathbf{I} \\ \mathbf{0} \end{bmatrix}\mathbf{P}_{\boldsymbol{\theta}_{m,j}^{*}}[\mathbf{I} \;\; \mathbf{0}]\mathbf{R}_{\boldsymbol{\theta}_{m,j}}^{T}\mathbf{H}_{\boldsymbol{\theta}_{m}^{*}}\mathbf{R}_{\boldsymbol{\theta}_{m,j}}\end{aligned} \tag{B.1}$$

$$\begin{aligned}\mathbf{R}_{\boldsymbol{\theta}_{m,j}}^{T}\mathbf{M}_{m,j}^{DCEM}\mathbf{R}_{\boldsymbol{\theta}_{m,j}} &= \begin{bmatrix} \mathbf{I}-\mathbf{P}_{\boldsymbol{\theta}_{m,j}^{*}}\mathbf{H}_{\boldsymbol{\theta}_{m,j}^{*}} & -\mathbf{P}_{\boldsymbol{\theta}_{m,j}^{*}}\mathbf{H}_{\boldsymbol{\theta}_{l}^{*}} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \\ &= \mathbf{I}-\begin{bmatrix} \mathbf{I} \\ \mathbf{0} \end{bmatrix}\mathbf{P}_{\boldsymbol{\theta}_{m,j}^{*}}[\mathbf{H}_{\boldsymbol{\theta}_{m,j}^{*}} \;\; \mathbf{H}_{\boldsymbol{\theta}_{m,l}^{*}}] \\ &= \mathbf{I}-\begin{bmatrix} \mathbf{I} \\ \mathbf{0} \end{bmatrix}\mathbf{P}_{\boldsymbol{\theta}_{m,j}^{*}}[\mathbf{I} \;\; \mathbf{0}]\mathbf{R}_{\boldsymbol{\theta}_{m,j}}^{T}\mathbf{H}_{\boldsymbol{\theta}_{m}^{*}}\mathbf{R}_{\boldsymbol{\theta}_{m,j}}\end{aligned} \tag{B.2}$$

Therefore,

$$\mathbf{M}_{m,j}^{DCEM} = \mathbf{I} - \mathbf{R}_{\boldsymbol{\theta}_{m,j}} \begin{bmatrix} \mathbf{I} \\ \\ \mathbf{0} \end{bmatrix} \mathbf{P}_{\boldsymbol{\theta}^*_{m,j}} [\mathbf{I}\ \mathbf{0}] \mathbf{R}^T_{\boldsymbol{\theta}_{m,j}} \mathbf{H}_{\boldsymbol{\theta}^*_m} \tag{B.3}$$

Multiplying both sides with $\mathbf{H}^{\frac{1}{2}}_{\boldsymbol{\theta}^*_m}$ and $\mathbf{H}^{-\frac{1}{2}}_{\boldsymbol{\theta}^*_m}$, (B.3) becomes

$$\mathbf{H}^{\frac{1}{2}}_{\boldsymbol{\theta}^*_m} \mathbf{M}_{m,j}^{DCEM} \mathbf{H}^{-\frac{1}{2}}_{\boldsymbol{\theta}^*_m} = \mathbf{I} - \mathbf{H}^{\frac{1}{2}}_{\boldsymbol{\theta}^*_m} \mathbf{R}_{\boldsymbol{\theta}_{m,j}} \begin{bmatrix} \mathbf{I} \\ \\ \mathbf{0} \end{bmatrix} \mathbf{P}_{\boldsymbol{\theta}^*_{m,j}} [\mathbf{I}\ \mathbf{0}] \mathbf{R}^T_{\boldsymbol{\theta}^*_j} \mathbf{H}^{\frac{1}{2}}_{\boldsymbol{\theta}^*_m} \tag{B.4}$$

$$= \mathbf{I} - \mathbf{H}^{\frac{1}{2}}_{\boldsymbol{\theta}^*_m} \mathbf{R}_{\boldsymbol{\theta}^*_{m,j}} \begin{bmatrix} \mathbf{P}_{\boldsymbol{\theta}^*_{m,j}} & \mathbf{0} \\ \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{R}^T_{\boldsymbol{\theta}_{m,j}} \mathbf{H}^{\frac{1}{2}}_{\boldsymbol{\theta}^*_m}$$

$$\mathbf{H}^{\frac{1}{2}}_{\boldsymbol{\theta}^*_m} \mathbf{M}_{m,j}^{DCEM} \mathbf{H}^{-\frac{1}{2}}_{\boldsymbol{\theta}^*_m} = \mathbf{I} - \mathbf{H}^{\frac{1}{2}}_{\boldsymbol{\theta}^*_m} \mathbf{R}_{\boldsymbol{\theta}^*_{m,j}} \begin{bmatrix} \mathbf{P}_{\boldsymbol{\theta}^*_{m,j}} & \mathbf{0} \\ \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{R}^T_{\boldsymbol{\theta}_{m,j}} \mathbf{H}^{\frac{1}{2}}_{\boldsymbol{\theta}^*_m} \tag{B.5}$$

If parameter sets are chosen cyclically in a natural order, i.e., $\{1, \cdots, J\}$, it follows from (B.5) that

$$\mathbf{M}_{m,J}^{DCEM} \times \cdots \mathbf{M}_{m,1}^{DCEM} = \mathbf{I} - \mathbf{P}_{\boldsymbol{\theta}^*_m} \mathbf{H}_{\boldsymbol{\theta}^*_m} \tag{B.6}$$

where

$$\mathbf{P}_{\boldsymbol{\theta}^*_m} = \mathbf{D}_{\mathbf{P}} + \mathbf{L}_{\mathbf{P}} \tag{B.7}$$

where

$$\mathbf{D}_{\mathbf{P}} = \begin{bmatrix} \mathbf{P}_{\boldsymbol{\theta}^*_{m,1}} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \mathbf{P}_{\boldsymbol{\theta}^*_{m,J}} \end{bmatrix} \tag{B.8}$$

is the block diagonal of $\mathbf{P}_{\boldsymbol{\theta}^*_m}$, and $\mathbf{L}_{\mathbf{P}}$ is the corresponding strictly lower block triangular matrix of $\mathbf{P}_{\boldsymbol{\theta}^*_m}$. Then, we decompose the local Hessian matrix $\mathbf{H}_{\boldsymbol{\theta}^*_m}$ by

$$\mathbf{H}_{\boldsymbol{\theta}^*_m} = \mathbf{D}_{\mathbf{H}} + \mathbf{L}_{\mathbf{H}} + \mathbf{L}^T_{\mathbf{H}} \tag{B.9}$$

where $\mathbf{D_H}, \mathbf{L_H}$ represent the block diagonal, strictly lower triangular block parts of $\mathbf{H}_{\boldsymbol{\theta}_m^*}$. From [68, Theorem 2], it can be verified that $\mathbf{D_P} = \mathbf{D_H^{-1}}$, we rewrite the (B.7) as follow:

$$\mathbf{P}_{\boldsymbol{\theta}_m^*} = (\mathbf{D_H} + \mathbf{L_H})^{-1} \tag{B.10}$$

Thus, by letting

$$\mathbf{M}_m^{DCEM} = \prod_{j=1}^{J} \mathbf{M}_{m,j}^{DCEM} \tag{B.11}$$

the local rate matrix of DCEM at node m is given by

$$\mathbf{M}_m^{DCEM} = \mathbf{I} - (\mathbf{D_H} + \mathbf{L_H})^{-1}\mathbf{H}_{\boldsymbol{\theta}_m^*} \tag{B.12}$$

Let $\|\mathbf{M}\|_2 = \sqrt{\rho(\mathbf{M}^H\mathbf{M})}$ denotes the matrix spectral norm of $\mathbf{M}$ and define $\bar{\mathbf{M}} = \mathbf{H}_{\boldsymbol{\theta}_m^*}^{\frac{1}{2}} \mathbf{M}_m^{DCEM} \mathbf{H}_{\boldsymbol{\theta}_m^*}^{-\frac{1}{2}}$, according to (B.12),

$$\begin{aligned}
\bar{\mathbf{M}}^H\bar{\mathbf{M}} &= \bar{\mathbf{M}}^T\bar{\mathbf{M}} \qquad (B.13)\\
&= \big(\mathbf{I} - \mathbf{H}_{\boldsymbol{\theta}_m^*}^{\frac{1}{2}}(\mathbf{D_H} + \mathbf{L_H})^{-1}\mathbf{H}_{\boldsymbol{\theta}_m^*}^{\frac{1}{2}}\big)^T\big(\mathbf{I} - \mathbf{H}_{\boldsymbol{\theta}_m^*}^{\frac{1}{2}}(\mathbf{D_H} + \mathbf{L_H})^{-1}\mathbf{H}_{\boldsymbol{\theta}_m^*}^{\frac{1}{2}}\big)\\
&= \mathbf{I} - \mathbf{H}_{\boldsymbol{\theta}_m^*}^{\frac{1}{2}}(\mathbf{D_H} + \mathbf{L_H})^{-T}(\mathbf{D_H} + \mathbf{L_H} + \mathbf{D_H^T} + \mathbf{L_H^T} - \mathbf{H}_{\boldsymbol{\theta}_m^*})(\mathbf{D_H} + \mathbf{L_H})^{-1}\mathbf{H}_{\boldsymbol{\theta}_m^*}^{\frac{1}{2}}\\
&< \mathbf{I}
\end{aligned}$$

By defining

$$\|\mathbf{M}_m^{DCEM}\|_{\mathbf{H}_{\boldsymbol{\theta}_m^*}}^2 = \rho(\bar{\mathbf{M}}^H\bar{\mathbf{M}}) \tag{B.14}$$

The inequality $\rho(\mathbf{M}) \leq \|\mathbf{M}\|_{\mathbf{N}}$ leads to

$$\begin{aligned}
\rho(\mathbf{M}_m^{DCEM}) &\leq \|\mathbf{M}_m^{DCEM}\|_{\mathbf{H}_{\boldsymbol{\theta}^*}} \qquad (B.15)\\
&= \sqrt{\rho(\bar{\mathbf{M}}^H)}\\
&< \sqrt{\rho(\mathbf{I})} = 1
\end{aligned}$$

Appendix C

Proof of Lemma 3

According to Lemma 1, we can obtain

$$\mathcal{W}_x^k(t) - \bar{\mathcal{W}}_x^k \tag{C.1}$$

$$= \left[\left(\prod_{t'}^{t} \mathbf{\Phi}_k(t') - \frac{\mathbf{1}_I \mathbf{1}_I}{I} \right) \otimes \mathbf{I}_{I_M} \right] \mathcal{W}_x^k(0), \tag{C.2}$$

the norm of which is bounded as

$$\left\| \mathcal{W}_x^k(t) - \bar{\mathcal{W}}_x^k \right\| \leq \left\| \prod_{t'=0}^{t} \mathbf{\Phi}_k(t') - \frac{\mathbf{1}_I \mathbf{1}_I}{I} \right\| \left\| \mathcal{W}_x^k(0) \right\| \tag{C.3}$$

$$\leq \left\| \prod_{t'=0}^{t} \mathbf{\Phi}_k(t') - \frac{\mathbf{1}_I \mathbf{1}_I}{I} \right\|_F \left\| \mathcal{W}_x^k(0) \right\| \tag{C.4}$$

$$\leq \left[2I \left(\frac{1+\eta^{-L_0}}{1-\eta^{-L_0}} \right) \lambda_\eta^t \right] \left\| \mathcal{W}_x^k(0) \right\|, \tag{C.5}$$

where $\lambda_\eta = (1-\eta^{L_0})^{1/L_0}$ and (C.4) is due to $\|\cdot\| \leq \|\cdot\|_F$. Since x_i lies in D, there exists positive real $C_D > 0$ such that $\left\| \mathcal{W}_x^k(0) \right\|^2 = \sum_{i=1}^{I} \|x_i^k\|^2 < C_D^2$ and we can further obtain that

$$\|p_i(t)\| \leq \left\| \mathcal{W}_x^k(t) - \bar{\mathcal{W}}_x^k \right\| \leq C_1(t), \tag{C.6}$$

where $C_1(t) = \left[2I \left(\frac{1+\eta^{-L_0}}{1-\eta^{-L_0}} \right) \lambda_\eta^t \right] C_D$. It can be seen that the norm of $p_i(t)$ decreases exponentially with t. Since the gradients of F_i are upper bounded in D, there exists positive real $C_G > 0$ such that

$$\left\| \mathcal{W}_F^k(0) \right\|^2 = \sum_{i=1}^{I} \left\| F_i(\bar{x}_i^k) \right\|^2 < C_G. \tag{C.7}$$

Therefore, similarly to the derivation of (C.6), we can obtain

$$\|q_i(t)\| \leq \left\| \mathcal{W}_F^k(t) - \bar{\mathcal{W}}_F^k \right\| \leq C_2(t), \tag{C.8}$$

where $C_2(t) = \left[2I \left(\frac{1+\eta^{-L_0}}{1-\eta^{-L_0}} \right) \lambda_\eta^t \right] C_G$. It can be then obtained that

$$\begin{aligned} \left\| \hat{F}_i^k(t) - F(\bar{\mathbf{x}}_i{}^k) \right\| &= \left\| \sum_{j=1}^{I} \left[F_j(\bar{\mathbf{x}}_j{}^k) - F_j(\bar{\mathbf{x}}_i{}^k) \right] + q_i(t) \right\| && \text{(C.9)} \\ &\leq \sum_{j=1}^{I} \rho \|\bar{\mathbf{x}}_j{}^k - \bar{\mathbf{x}}_i{}^k\| + \|q_i(t)\|, && \text{(C.10)} \end{aligned}$$

where (C.10) is from Lemma 2. From (C.6), it can be derived that

$$\|\bar{\mathbf{x}}_j{}^k - \bar{\mathbf{x}}_i{}^k\| \leq \|p_i(l_k)\| + \|p_j(l_k)\| \leq 2C_1(l_k). \tag{C.11}$$

Finally, from (C.8),(C.10) and(C.11), we have (5.29).

Appendix D

Proof of Lemma 4

In the following analysis, we let ϵ, ρ be the corresponding parameters in Lemma 2, i.e. $\max\{\|u - x^*\|, \|v - x^*\|\} \leq \epsilon$ would lead to $u, v \in D$ and inequality (5.26). Define $\mathcal{N}_2$ as

$$\mathcal{N}_2 = \left\{ \mathbf{H} \in L(\mathbb{R}^n) | \|F^{'}(\mathbf{x}^*)\| \|\mathbf{H} - F^{'}(\mathbf{x}^*)^{-1}\| < 1/2 \right\}. \tag{D.1}$$

To start with, we prove that the norm of $\mathbf{y}_i^k$ is upper bounded by a constant. For any $\mathbf{H} \in \mathcal{N}_2$, we have that $\mathbf{H}$ is non-singular and there exists a positive real c, s.t. $\|\mathbf{H}\| \leq c$. If $\bar{\mathbf{x}}^{k-1} \in D$ is satisfied and further define $\|\mathbf{s}_i^k\| = \|\mathbf{x}_i^k - \bar{\mathbf{x}_i}^{k-1}\|$, then

$$\|\mathbf{s}_i^k\| \leq \|\mathbf{H}\|^{k-1} \hat{F}_i^{k-1}\| \tag{D.2}$$

$$= \|\mathbf{H}\|^{k-1} \left\| F(\bar{\mathbf{x}_i}^{k-1}) - F(\mathbf{x}^*) + \hat{F}_i^{k-1} - F(\bar{\mathbf{x}_i}^{k-1}) \right\| \tag{D.3}$$

$$\leq c \left[\rho \|\bar{\mathbf{x}_i}^{k-1} - \mathbf{x}^*\| + \left\| \hat{F}_i^{k-1} - F(\bar{\mathbf{x}_i}^{k-1}) \right\| \right], \tag{D.4}$$

where (D.4) is due to Lemma 2. Then we can bound the state (after state averaging) of the k^{th} iteration by

$$\|\bar{\mathbf{x}_i}^k - \mathbf{x}^*\| = \left\| p_i(l_k) + \mathbf{x}_i^k - \bar{\mathbf{x}_i}^{k-1} + \bar{\mathbf{x}_i}^{k-1} - \mathbf{x}^* \right\| \tag{D.5}$$

$$\leq \|p_i(l_k)\| + \|\mathbf{s}_i^k\| + \|\bar{\mathbf{x}_i}^{k-1} - \mathbf{x}^*\| \tag{D.6}$$

$$\leq (c\rho + 1)\|\bar{\mathbf{x}_i}^{k-1} - \mathbf{x}^*\| + \|p_i(l_k)\| + c \left\| \hat{F}_i^{k-1} - F(\bar{\mathbf{x}_i}^{k-1}) \right\|. \tag{D.7}$$

Now we define $\mathcal{N}_1$ as

$$\forall \mathbf{x}_i^{k-1} \in \mathcal{N}_1, \|\bar{\mathbf{x}_i}^{k-1} - \mathbf{x}^*\| \leq \min \left\{ \frac{\epsilon}{2(1 + c\rho)}, \frac{\epsilon}{2} \right\}, \tag{D.8}$$

where $\mu_2(2\rho\epsilon)^p < 1/3$. According to Lemma 3, by choosing $l_{k-1}, l_{k-1}^{'}$ such that

$$c\left(2C_1(l_{k-1}) + C_2(l_{k-1}^{'})\right) + C_1(l_{k-1}) < \epsilon/2, \tag{D.9}$$

it can derived that $\|\bar{\mathbf{x}_i}^k - \mathbf{x}^*\| < \epsilon$ or $\bar{\mathbf{x}_i}^k \in D$. Now that both $\bar{\mathbf{x}_i}^k, \bar{\mathbf{x}_i}^{k-1} \in D$, by Lemma 2, we can derive that

$$1/\rho\|\mathbf{z}_i^k\| \leq \|F(\bar{\mathbf{x}_i}^k) - F(\bar{\mathbf{x}_i}^{k-1})\| \leq \rho\|\mathbf{z}_i^k\|, \tag{D.10}$$

which is related to the term that we are trying to bound as

$$\mathbf{y}_i^k = \|F(\bar{\mathbf{x}_i}^k) - F(\bar{\mathbf{x}_i}^{k-1}) + \tag{D.11}$$

$$(\hat{F}_i^k - F(\bar{\mathbf{x}_i}^k)) - (\hat{F}_i^{k-1} - F(\bar{\mathbf{x}_i}^{k-1}))\|. \tag{D.12}$$

Again, using Lemma 3, we can bound the last two terms in (D.11) by choosing appropriate $l_k, l_{k-1}, l_k^{'}$ and $l_{k-1}^{'}$ such that

$$1/(2\rho)\|\mathbf{z}_i^k\| \leq \|\mathbf{y}_i^k\| \leq 2\rho\|\mathbf{z}_i^k\| \tag{D.13}$$

Next, we prove that $\|\mathbf{y}_i^k\|$ is also lower bounded as

$$\frac{\|\mathbf{M}\mathbf{z}_i^k - \mathbf{M}^{-1}\mathbf{y}_i^k\|}{\|\mathbf{M}^{-1}\mathbf{y}_i^k\|} \leq \mu_2\|\mathbf{y}_i^k\|^p, \tag{D.14}$$

for some constants $p > 0, \mu_2 > 0$ and symmetric, non-singular $\mathbf{M}$. To see this, first since $F^{'}(\mathbf{x}^*)$ is symmetric and positive definite, there exists a positive symmetric $\mathbf{M}$ s.t. $F^{'}(\mathbf{x}^*) = \mathbf{M}^2$. We could then write

$$\mathbf{M}^{-1}\mathbf{y}_i^k - \mathbf{M}\mathbf{z}_i^k = \mathbf{M}^{-1}[\mathbf{y} - F^{'}(\mathbf{x}^*)\mathbf{z}_i^k], \tag{D.15}$$

which by Lemma 2, is equivalent to

$$\frac{\|\mathbf{M}\mathbf{z}_i^k - \mathbf{M}^{-1}\mathbf{y}_i^k\|}{\|\mathbf{M}^{-1}\mathbf{y}_i^k\|} \leq \mu_0 \max\{\|\bar{\mathbf{x}_i}^k - \mathbf{x}^*\|^p, \|\bar{\mathbf{x}_i}^{k-1} - \mathbf{x}^*\|^p\}. \tag{D.16}$$

Since $\mathbf{H}^{k-1}$ is in a neighborhood of $F^{'}(\mathbf{x})^{-1}$, i.e. $\mathcal{N}_2$, by the Banach Perturbation Lemma we can

bound the operator norm of $\mathbf{H}^{k-1}$ as

$$\|\mathbf{H}^{k-1}\| \leq 2\|F^{'}(\mathbf{x}^*)^{-1}\|. \tag{D.17}$$

By Lemma 2, it can then be derived that

$$1/\rho\|\bar{\mathbf{x}_i}^k - \mathbf{x}^*\| \leq \|\mathbf{H}^{k-1}\|^{-1}\|\mathbf{z}_i^k\|, \tag{D.18}$$

which combined with (D.17) indicates that there exists $\lambda > 0$ s.t.

$$\max\{\|\bar{\mathbf{x}_i}^k - \mathbf{x}^*\|^p, \|\bar{\mathbf{x}_i}^{k-1} - \mathbf{x}^*\|^p\} \leq \lambda\|\mathbf{z}_i^k\|^p. \tag{D.19}$$

It is easy to see that due to Lemma 2, (D.16) combined with (D.19) is equivalent to (D.14). From (D.14) and (D.13), we can finally derive that

$$\frac{\|\mathbf{M}\mathbf{z}_i^k - \mathbf{M}^{-1}\mathbf{y}_i^k\|}{\|\mathbf{M}^{-1}\mathbf{y}_i^k\|} \leq \mu_2\|\mathbf{y}_i^k\|^p \leq \mu_2(\rho\epsilon)^p \leq 1/3, \tag{D.20}$$

which enables us to use Lemma 5.2 in [106] to derive the following inequality

$$\begin{aligned}\left\|\mathbf{H}^k - F^{'}(\mathbf{x}^*)^{-1}\right\|_M \leq \left[1 + \lambda_1^{'}\|\mathbf{y}\|^p\right]\left\|\mathbf{H}^{k-1} - F^{'}(\mathbf{x}^*)^{-1}\right\|_M \\ +\lambda_2^{'}\frac{\|\mathbf{z}_i^k - F^{'}(\mathbf{x}^*)^{-1}\mathbf{y}_i^k\|}{\|\mathbf{M}^{-1}\mathbf{y}_i^k\|},\end{aligned} \tag{D.21}$$

where λ_1, λ_2 are positive constants. Rewrite that

$$\|\mathbf{z}_i^k - F^{'}(\mathbf{x}^*)^{-1}\mathbf{y}_i^k\| = \|F^{'}(\mathbf{x}^*)^{-1}\|\|F^{'}(\mathbf{x}^*)\mathbf{z}_i^k - \mathbf{y}_i^k\| \tag{D.22}$$

Since $\bar{\mathbf{x}_i}^k, \bar{\mathbf{x}_i}^k \in D$ and according to Lemma 2, it can be derived that

$$\begin{aligned}\|F^{'}(\mathbf{x}^*)\mathbf{z}_i^k - \mathbf{y}_i^k\| &= \|F^{'}(\mathbf{x}^*)\mathbf{z}_i^k - \left(F(\bar{\mathbf{x}_i}^k) - F(\bar{\mathbf{x}_i}^{k-1})\right) \\ &\quad - \left(\hat{F}_i^k - F(\bar{\mathbf{x}_i}^k)\right) + \left(\hat{F}_i^{k-1} - F(\bar{\mathbf{x}_i}^{k-1})\right)\| \qquad \text{(D.23)} \\ &\leq K\max\left\{\|\bar{\mathbf{x}_i}^k - \mathbf{x}^*\|, \|\bar{\mathbf{x}_i}^{k-1} - \mathbf{x}^*\|\right\}\|\mathbf{z}_i^k\| \\ &\quad + \left\|\hat{F}_i^k - F(\bar{\mathbf{x}_i}^k)\right\| + \left\|\hat{F}_i^{k-1} - F(\bar{\mathbf{x}_i}^{k-1})\right\| \qquad \text{(D.24)} \\ &\leq (K + K_q)\max\left\{\|\bar{\mathbf{x}_i}^k - \mathbf{x}^*\|, \|\bar{\mathbf{x}_i}^{k-1} - \mathbf{x}^*\|\right\}\|\mathbf{z}_i^k\|, \qquad \text{(D.25)}\end{aligned}$$

where (D.25) is derived by choosing sufficiently large iteration number so that the last two terms on the right hand side of (D.24) are bounded according to Lemma 3. Moreover, by using (D.13), it can be obtained that

$$\|\mathbf{y}_i^k\| \leq 2\rho \leq 2\rho \max\left\{\|\bar{\mathbf{x}_i}^k - \mathbf{x}^*\|, \|\bar{\mathbf{x}_i}^{k-1} - \mathbf{x}^*\|\right\}\|. \tag{D.26}$$

Finally, by combining (D.13), (D.21), (D.25) and (D.26), we can finally prove that inequality (5.30) is satisfied under the aforementioned assumptions.

Appendix E

Proof of Theorem 2

We set the neighborhood $\mathcal{N}$ as the one that satisfies the requirements in Lemma 4, i.e. for each $(\bar{\mathbf{x}}^{k-1}, \mathbf{H}^{k-1}) \in \mathcal{N}$, inequality (5.30) is satisfied. Then we choose $\epsilon(r), \delta(r)$ such that $\|\mathbf{x} - \mathbf{x}^*\|$ and $\|\mathbf{H} - F^{'}(\mathbf{x}^*)^{-1}\|_M < \delta$ would imply that $(\mathbf{x}, \mathbf{H}) \in \mathcal{N}$.

First, according to Lemma 3, by choosing sufficiently large l_0 such that $\|p_i(l_0)\| \leq \epsilon$, it can be derived that

$$\|\bar{\mathbf{x}_i}^0 - \mathbf{x}^*\| = \|\mathbf{x}_i^0 + p_i(l_0) - \mathbf{x}^*\| \leq \epsilon/2 + \|p_i(l_0)\| \leq \epsilon, \tag{E.1}$$

which leads to that $(\bar{\mathbf{x}_i}^0, \mathbf{H_i^0}) \in \mathcal{N}$. Since

$$\mathbf{x}_i^1 = \bar{\mathbf{x}_i}^0 - \mathbf{H_i^0}\hat{F}_i^0, \tag{E.2}$$

we can write that

$$\begin{aligned}\mathbf{x}_i^1 - \mathbf{x}^* = -\mathbf{H}_i^0 \Big[& F(\bar{\mathbf{x}_i}^0) - F(\mathbf{x}^*) - F^{'}(\mathbf{x}^*)(\bar{\mathbf{x}_i}^0 - \mathbf{x}^*) \\ & + \hat{F}_i^0 - F(\bar{\mathbf{x}_i}^0) \Big] + \Big[\mathbf{I} - \mathbf{H}_i^0 F^{'}(\mathbf{x}^*) \Big]. \end{aligned} \tag{E.3}$$

Since $\mathcal{N}_1 \subset D$ (as shown in the proof of Lemma 4), according to Lemma 3, it can be derived that

$$\begin{aligned} & \Big\| F(\bar{\mathbf{x}_i}^0) - F(\mathbf{x}^*) - F^{'}(\mathbf{x}^*)(\bar{\mathbf{x}_i}^0 - \mathbf{x}^*) \Big\| \\ \leq K\|\bar{\mathbf{x}_i}^0 - \mathbf{x}^*\|^p \|\bar{\mathbf{x}_i}^0 - \mathbf{x}^*\| & \leq K\epsilon^p \|\bar{\mathbf{x}_i}^0 - \mathbf{x}^*\|. \end{aligned} \tag{E.4}$$

By the equivalence of all norms that deal with a finite-dimensional space, there exists a constant α, s.t. $\|\mathbf{A}\| \leq \alpha \|\mathbf{A}\|_M$. Therefore, from $\|\mathbf{H}_i^0 - F^{'}(\mathbf{x}^*)\|_M < \delta$, we derive that

$$\Big\| \mathbf{H}_i^0 - F^{'}(\mathbf{x}^*)^{-1} \Big\| < \alpha\delta. \tag{E.5}$$

Further we assume $\sigma \geq \|F^{'}(\mathbf{x}^*)\|, \gamma \geq F^{'}(\mathbf{x}^*)^{-1}$. Then we can write

$$\left\|\mathbf{I} - \mathbf{H}_i^0 F^{'}(\mathbf{x}^*)\right\| = \left\|F^{'}(\mathbf{x}^*)^{-1} - \mathbf{H}_i^0\right\| \left\|F^{'}(\mathbf{x}^*)\right\| \leq 2\alpha\delta\sigma. \tag{E.6}$$

Combining (E.3), (E.4) and (E.6), it can be derived that

$$\begin{aligned}\|\mathbf{x}_i^1 - \mathbf{x}^*\| \leq \left[\|\mathbf{H}_i^0\| K\epsilon^p + 2\alpha\delta\sigma\right] \|\bar{\mathbf{x}_i}^0 - \mathbf{x}^*\| \\ + \|\mathbf{H}_i^0\| \left\|\hat{F}_i^0 - F(\bar{\mathbf{x}_i}^0)\right\|.\end{aligned} \tag{E.7}$$

Further we bound the vector norm of $\mathbf{H}_i^0$ by

$$\|\mathbf{H}_i^0\| \leq \|\mathbf{H}_i^0 - F^{'}(\mathbf{x}^*)^{-1}\| \leq 2\alpha\delta + \gamma. \tag{E.8}$$

Let ϵ, δ be sufficiently small, such that

$$(2\alpha\delta + \gamma)K\epsilon^p + 2\sigma\delta\alpha \leq r. \tag{E.9}$$

Then we can have

$$\|\mathbf{x}_i^1 - \mathbf{x}^*\| \leq r\|\bar{\mathbf{x}_i}^0 - \mathbf{x}^*\| + (2\alpha\delta + \gamma) \left\|\hat{F}_i^0 - F(\bar{\mathbf{x}_i}^0)\right\|. \tag{E.10}$$

Choose $l_0, l_0^{'}$ such that

$$\left\|\hat{F}_i^0 - F(\bar{\mathbf{x}_i}^0)\right\| \leq \frac{1-\gamma}{\eta(2\alpha\delta + \gamma)} \|\bar{\mathbf{x}_i}^0 - \mathbf{x}^*\|, \eta > 1. \tag{E.11}$$

We can derive that

$$\|\mathbf{x}_i^1 - \mathbf{x}^*\| \leq \|\bar{\mathbf{x}_i}^0 - \mathbf{x}^*\|, \tag{E.12}$$

where $\hat{r} = (r + (1-r)/\eta) \in (0,1)$. Using Lemma 3, with sufficiently large l_1, we can further have the following bound

$$\|\bar{\mathbf{x}_i}^1 - \mathbf{x}^*\| \leq r^{'}\|\bar{\mathbf{x}_i}^0 - \mathbf{x}^*\|, r^{'} \in (0,1), \tag{E.13}$$

which indicates that $\bar{\mathbf{x}_i}^1 \in \mathcal{N}_1$.

Next, we start an induction argument. First, for $k = 0, \cdots, m-1$, we assume that

$$\left\|\mathbf{H}_i^k - F^{'}(\mathbf{x}^*)^{-1}\right\|_M \leq 2\delta, \tag{E.14}$$

$$\|\bar{\mathbf{x}}_i{}^{k+1} - \mathbf{x}^*\| \leq r^{'}\|\bar{\mathbf{x}}_i{}^k - \mathbf{x}^*\|. \tag{E.15}$$

Since $(\mathbf{x}_i^k, \mathbf{H}_i^k) \in \mathcal{N}$, by (5.30), it can be derived that

$$\left\|\mathbf{H}_i^{k+1} - F^{'}(\mathbf{x}^*)^{-1}\right\|_M - \left\|\mathbf{H}_i^k - F^{'}(\mathbf{x}^*)^{-1}\right\|_M \tag{E.16}$$

$$\leq 2\lambda_1\sigma\epsilon^p r^{'kp} + \lambda_2\epsilon^p r^{'kp}. \tag{E.17}$$

By summing the two sides of inequality (E.16) for $k = 0, \cdots, m-1$, we obtain

$$\left\|\mathbf{H}_i^m - F^{'}(\mathbf{x}^*)^{-1}\right\|_M \leq \left\|\mathbf{H}_i^0 - F^{'}(\mathbf{x}^*)^{-1}\right\|_M + (2\lambda_1\delta + \lambda_2)\frac{\epsilon^p}{1 - r^{'p}}. \tag{E.18}$$

By choosing sufficiently small ϵ, we can have

$$(2\lambda_1\delta + \lambda_2)\frac{\epsilon^p}{1 - r^{'p}} < \delta, \tag{E.19}$$

which further leads to

$$\left\|\mathbf{H}_i^m - F^{'}(\mathbf{x}^*)^{-1}\right\| \leq 2\alpha\delta. \tag{E.20}$$

Similarly to the case when $m = 1$, with the help of Lemma 2, it can be derived that

$$\|\mathbf{x}_i^{m+1} - \mathbf{x}^*\| \leq$$

$$[\|\mathbf{H}_i^m\|K\epsilon^p + 2\sigma\delta\alpha] + \|\mathbf{H}_i^m\|\|\hat{F}_i^m - F(\bar{\mathbf{x}}_i{}^m)\|. \tag{E.21}$$

Noticing that

$$\|\mathbf{H}_i^m\| \leq \|\mathbf{H}_i^m - F^{'}(\mathbf{x}^*)^{-1}\| \leq 2\alpha\delta + \gamma, \tag{E.22}$$

we can rewrite (E.21) as

$$\|\mathbf{x}_i^{m+1} - \mathbf{x}^*\| \leq r\|\bar{\mathbf{x}}_i{}^m - \mathbf{x}^*\| + (2\alpha\delta + \gamma)\|\hat{F}_i^m - F(\bar{\mathbf{x}}_i{}^m)\|. \tag{E.23}$$

Again, by Lemma 3, by choosing $l_m, l_m^{'}$ sufficiently large, we can conclude the induction argument

by showing that

$$\|\bar{\mathbf{x}}_i^{m+1} - \mathbf{x}^*\| \leq r^{'} \|\bar{\mathbf{x}}_i^{m} - \mathbf{x}^*\| \tag{E.24}$$

Appendix F

List of Publications

This Appendix contains a list of published and submitted papers.

F.1 Conference Papers

- **Jia Yu** and Pei-Jung Chung. "Distributed componentwise EM algorithm for mixture models in sensor networks." 2013 IEEE Global Communications Conference (GLOBECOM). IEEE, 2013.

- **Jia Yu** and John Thompson. "Diffusion-based EM gradient algorithm for density estimation in sensor networks." Signal Processing Advances in Wireless Communications (SPAWC), 2016 IEEE 17th International Workshop on. IEEE, 2016.

F.2 Journal Papers

- **Jia Yu** and John Thompson, "Distributed componentwise EM algorithm for density estimation in wireless sensor networks,", *IET Signal Processing*, submitted on November, 2016

- **Jia Yu** and John Thompson, "Distributed qausi-Newton method for state estimation in power systems,", *IEEE Trans. on Smart Grid*, to be submitted.

Bibliography

[1] Robert D Nowak. Distributed EM algorithms for density estimation and clustering in sensor networks. *IEEE transactions on signal processing*, 51(8):2245–2253, 2003.

[2] Dongbing Gu. Distributed EM algorithm for Gaussian mixtures in sensor networks. *IEEE Transactions on Neural Networks*, 19(7):1154–1166, 2008.

[3] Yang Weng, Lihua Xie, and Wendong Xiao. Diffusion scheme of distributed EM algorithm for Gaussian mixtures over random networks. In *2009 IEEE International Conference on Control and Automation*, pages 1529–1534. IEEE, 2009.

[4] Sheng-Yuan Tu and Ali H Sayed. Diffusion strategies outperform consensus strategies for distributed estimation over adaptive networks. *IEEE Transactions on Signal Processing*, 60(12):6217–6234, 2012.

[5] Cassio G Lopes and Ali H Sayed. Incremental adaptive strategies over distributed networks. *IEEE Transactions on Signal Processing*, 55(8):4064–4077, 2007.

[6] Le Xie, Dae-Hyun Choi, Soummya Kar, and H Vincent Poor. Fully distributed state estimation for wide-area monitoring systems. *IEEE Transactions on Smart Grid*, 3(3):1154–1169, 2012.

[7] Soummya Kar, José MF Moura, and Kavita Ramanan. Distributed parameter estimation in sensor networks: Nonlinear observation models and imperfect communication. *IEEE Transactions on Information Theory*, 58(6):3575–3605, 2012.

[8] Gilles Celeux, Stéphane Chrétien, Florence Forbes, and Abdallah Mkhadri. A component-wise EM algorithm for mixtures. *Journal of Computational and Graphical Statistics*, 2012.

[9] Xiaochuan Zhao and Ali H Sayed. Asynchronous adaptation and learning over

networksPart I: Modeling and stability analysis. *IEEE Transactions on Signal Processing*, 63(4):811–826, 2015.

[10] Xiaochuan Zhao and Ali H Sayed. Asynchronous adaptation and learning over networksPart II: Performance analysis. *IEEE Transactions on Signal Processing*, 63(4):827–842, 2015.

[11] Xiaochuan Zhao and Ali H Sayed. Asynchronous adaptation and learning over networksPart III: Comparison analysis. *IEEE Transactions on Signal Processing*, 63(4):843–858, 2015.

[12] Xiao Li and Anna Scaglione. Convergence and applications of a Gossip-based Gauss-Newton algorithm. *IEEE Transactions on Signal Processing*, 61(21):5231–5246, 2013.

[13] John E Dennis, Jr and Jorge J Moré. Quasi-Newton methods, motivation and theory. *SIAM review*, 19(1):46–89, 1977.

[14] Kazem Sohraby, Daniel Minoli, and Taieb Znati. *Wireless sensor networks: technology, protocols, and applications*. John Wiley & Sons, 2007.

[15] Chee-Yee Chong and Srikanta P Kumar. Sensor networks: evolution, opportunities, and challenges. *Proceedings of the IEEE*, 91(8):1247–1256, 2003.

[16] Ian F Akyildiz and Mehmet Can Vuran. *Wireless sensor networks*, volume 4. John Wiley & Sons, 2010.

[17] Chiara Buratti, Andrea Conti, Davide Dardari, and Roberto Verdone. An overview on wireless sensor networks technology and evolution. *Sensors*, 9(9):6869–6896, 2009.

[18] J-F Chamberland and Venugopal V Veeravalli. Decentralized detection in sensor networks. *IEEE Transactions on Signal Processing*, 51(2):407–416, 2003.

[19] Neal Patwari, Joshua N Ash, Spyros Kyperountas, Alfred O Hero, Randolph L Moses, and Neiyer S Correal. Locating the nodes: cooperative localization in wireless sensor networks. *IEEE Signal processing magazine*, 22(4):54–69, 2005.

[20] Michael Rabbat and Robert Nowak. Distributed optimization in sensor networks. In *Proceedings of the 3rd international symposium on Information processing in sensor networks*, pages 20–27. ACM, 2004.

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

[22] Zhi-Quan Luo, Michael Gastpar, Juan Liu, and Ananthram Swami. Distributed signal processing in sensor networks [from the guest editors]. *IEEE Signal processing magazine*, 23(4):14–15, 2006.

[23] Laurent Eschenauer and Virgil D Gligor. A key-management scheme for distributed sensor networks. In *Proceedings of the 9th ACM conference on Computer and communications security*, pages 41–47. ACM, 2002.

[24] Alejandro Ribeiro and Georgios B Giannakis. Bandwidth-constrained distributed estimation for wireless sensor networks-part ii: unknown probability density function. *IEEE Transactions on Signal Processing*, 54(7):2784–2796, 2006.

[25] Alejandro Ribeiro and Georgios B Giannakis. Bandwidth-constrained distributed estimation for wireless sensor networks-part i: Gaussian case. *IEEE transactions on signal processing*, 54(3):1131–1143, 2006.

[26] Ali H Sayed and Cassio G Lopes. Adaptive processing over distributed networks. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 90(8):1504–1510, 2007.

[27] S.Haykin. *Adaptive filter theory, 4th ed.* Upper Saddle River, NJ,USA: Prentice Hall, 2002.

[28] Leilei Li, Jonathon A Chambers, Cassio G Lopes, and Ali H Sayed. Distributed estimation over an adaptive incremental network based on the affine projection algorithm. *IEEE Transactions on Signal Processing*, 58(1):151–164, 2010.

[29] Federico S Cattivelli and Ali H Sayed. Diffusion lms strategies for distributed estimation.

IEEE Transactions on Signal Processing, 58(3):1035–1048, 2010.

[30] Cassio G Lopes and Ali H Sayed. Diffusion least-mean squares over adaptive networks: Formulation and performance analysis. *IEEE Transactions on Signal Processing*, 56(7):3122–3136, 2008.

[31] Lin Xiao and Stephen Boyd. Fast linear iterations for distributed averaging. *Systems & Control Letters*, 53(1):65–78, 2004.

[32] Ali H Sayed et al. Adaptation, learning, and optimization over networks. *Foundations and Trends® in Machine Learning*, 7(4-5):311–801, 2014.

[33] Alexandros G Dimakis, Soummya Kar, José MF Moura, Michael G Rabbat, and Anna Scaglione. Gossip algorithms for distributed signal processing. *Proceedings of the IEEE*, 98(11):1847–1864, 2010.

[34] Stephen Boyd, Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah. Randomized gossip algorithms. *IEEE/ACM Transactions on Networking (TON)*, 14(SI):2508–2530, 2006.

[35] John N Tsitsiklis, Dimitri P Bertsekas, and Michael Athans. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. In *1984 American Control Conference*, pages 484–489, 1984.

[36] Dimitri P Bertsekas and John N Tsitsiklis. *Parallel and distributed computation: numerical methods*, volume 23. Prentice hall Englewood Cliffs, NJ, 1989.

[37] Soummya Kar and José MF Moura. Distributed consensus algorithms in sensor networks: Link failures and channel noise. *arXiv preprint arXiv:0711.3915*, 2007.

[38] Tuncer Can Aysal, Mehmet Ercan Yildiz, Anand D Sarwate, and Anna Scaglione. Broadcast gossip algorithms for consensus. *IEEE Transactions on Signal processing*, 57(7):2748–2761, 2009.

[39] Dusan Jakovetic, João Xavier, and José MF Moura. Weight optimization for consensus algorithms with correlated switching topology. *IEEE Transactions on Signal Processing*,

58(7):3788–3801, 2010.

[40] Dusan Jakovetic, Joao Xavier, and José MF Moura. Cooperative convex optimization in networked systems: Augmented lagrangian algorithms with directed gossip communication. *IEEE Transactions on Signal Processing*, 59(8):3889–3902, 2011.

[41] Tuncer C Aysal, Anand D Sarwate, and Alexandros G Dimakis. Reaching consensus in wireless networks with probabilistic broadcast. In *Communication, Control, and Computing, 2009. Allerton 2009. 47th Annual Allerton Conference on*, pages 732–739. IEEE, 2009.

[42] Cassio G Lopes and Ali H Sayed. Diffusion adaptive networks with changing topologies. In *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2008.

[43] Noriyuki Takahashi and Isao Yamada. Link probability control for probabilistic diffusion least-mean squares over resource-constrained networks. In *ICASSP*, pages 3518–3521, 2010.

[44] Emmanuel J Candès and Michael B Wakin. An introduction to compressive sampling. *IEEE signal processing magazine*, 25(2):21–30, 2008.

[45] S.Haykin. *Adaptive filter theory, 3rd ed*. Upper Saddle River, NJ,USA: Prentice Hall, 2002.

[46] Fred C Schweppe and J Wildes. Power system static-state estimation, part i: Exact model. *IEEE Transactions on Power Apparatus and systems*, (1):120–125, 1970.

[47] Fred C Schweppe and Douglas B Rom. Power system static-state estimation, part ii: Approximate model. *IEEE Transactions on Power Apparatus and Systems*, (1):125–130, 1970.

[48] Anjan Bose. Smart transmission grid applications and their supporting infrastructure. *IEEE Transactions on Smart Grid*, 1(1):11–19, 2010.

[49] Vassilis Kekatos and Georgios B Giannakis. Distributed robust power system state estimation. *IEEE Transactions on Power Systems*, 28(2):1617–1626, 2013.

[50] Ian F Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, and Erdal Cayirci. A survey on sensor networks. *IEEE communications magazine*, 40(8):102–114, 2002.

[51] Deborah Estrin, Ramesh Govindan, John Heidemann, and Satish Kumar. Next century challenges: Scalable coordination in sensor networks. In *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 263–270. ACM, 1999.

[52] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

[53] Geoffrey McLachlan and David Peel. *Finite mixture models*. John Wiley & Sons, 2004.

[54] Ali H Sayed, Sheng-Yuan Tu, Jianshu Chen, Xiaochuan Zhao, and Zaid J Towfic. Diffusion strategies for adaptation and learning over networks: an examination of distributed strategies and network behavior. *IEEE Signal Processing Magazine*, 30(3):155–171, 2013.

[55] Jianshu Chen and Ali H Sayed. Diffusion adaptation strategies for distributed optimization and learning over networks. *IEEE Transactions on Signal Processing*, 60(8):4289–4305, 2012.

[56] Federico S Cattivelli and Ali H Sayed. Diffusion LMS strategies for distributed estimation. *IEEE Transactions on Signal Processing*, 58(3):1035–1048, 2010.

[57] Gonzalo Mateos and Georgios B Giannakis. Distributed recursive least-squares: Stability and performance analysis. *IEEE Transactions on Signal Processing*, 60(7):3740–3754, 2012.

[58] Meng-Li Cao, Qing-Hao Meng, Ming Zeng, Biao Sun, Wei Li, and Cheng-Jun Ding. Distributed least-squares estimation of a remote chemical source via convex combination in wireless sensor networks. *Sensors*, 14(7):11444–11466, 2014.

[59] Lei Cao, Chen Xu, Wei Shao, Guoan Zhang, Hui Zhou, Qiang Sun, and Yuehua Guo. Distributed power allocation for sink-centric clusters in multiple sink wireless sensor

networks. *Sensors*, 10(3):2003–2026, 2010.

[60] Paolo Di Lorenzo and Ali H Sayed. Sparse distributed learning based on diffusion adaptation. *IEEE Transactions on signal processing*, 61(6):1419–1433, 2013.

[61] Zhaoting Liu, Ying Liu, and Chunguang Li. Distributed sparse recursive least-squares over networks. *IEEE Transactions on Signal Processing*, 62(6):1386–1395, 2014.

[62] Chunguang Li, Pengcheng Shen, Ying Liu, and Zhaoyang Zhang. Diffusion information theoretic learning for distributed estimation over network. *IEEE Transactions on Signal Processing*, 61(16):4011–4024, 2013.

[63] Dongbing Gu and Huosheng Hu. Spatial gaussian process regression with mobile sensor networks. *IEEE Transactions on Neural Networks and Learning Systems*, 23(8):1279–1290, 2012.

[64] Lin Xiao, Stephen Boyd, and Sanjay Lall. A space-time diffusion scheme for peer-to-peer least-squares estimation. In *Proceedings of the 5th international conference on Information processing in sensor networks*, pages 168–176. ACM, 2006.

[65] Ioannis D Schizas, Gonzalo Mateos, and Georgios B Giannakis. Distributed LMS for consensus-based in-network adaptive processing. *IEEE Transactions on Signal Processing*, 57(6):2365–2382, 2009.

[66] Wojtek Kowalczyk and Nikos Vlassis. Newscast EM. *Advances in Neural Information Processing Systems 17*, pages 713–720, 2005.

[67] Jason Wolfe, Aria Haghighi, and Dan Klein. Fully distributed EM for very large datasets. In *Proceedings of the 25th international conference on Machine learning*, pages 1184–1191. ACM, 2008.

[68] Jeffrey A Fessler and Alfred O Hero. Space-alternating generalized Expectation-Maximization algorithm. *IEEE Transactions on Signal Processing*, 42(10):2664–2677, 1994.

[69] Xiao-Li Meng and David Van Dyk. The EM algorithman old Folk-song sung to a fast New Tune. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 59(3):511–567, 1997.

[70] Radford M Neal and Geoffrey E Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in graphical models*, pages 355–368. Springer, 1998.

[71] Julian Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 259–302, 1986.

[72] Lei Xu and Michael I Jordan. On convergence properties of the EM algorithm for Gaussian mixtures. *Neural computation*, 8(1):129–151, 1996.

[73] Jinwen Ma, Lei Xu, and Michael I Jordan. Asymptotic convergence rate of the EM algorithm for gaussian mixtures. *Neural Computation*, 12(12):2881–2907, 2000.

[74] Bo Thiesson, Christopher Meek, and David Heckerman. Accelerating EM for large databases. *Machine Learning*, 45(3):279–299, 2001.

[75] Behrooz Safarinejadian, Mohammad B Menhaj, and Mehdi Karrari. A distributed EM algorithm to estimate the parameters of a finite mixture of components. *Knowledge and information systems*, 23(3):267–292, 2010.

[76] Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 2012.

[77] Elijah Polak. *Computational methods in optimization: a unified approach*, volume 77. Academic press, 1971.

[78] Piyush Gupta and Panganmala R Kumar. The capacity of wireless networks. *IEEE Transactions on information theory*, 46(2):388–404, 2000.

[79] Jia Yu and Pei-Jung Chung. Diibuted componentwise EM algorithm or mixture models in sensor networks. In *2013 IEEE Global Communications Conference (GLOBECOM)*, pages 3418–3422. IEEE, 2013.

[80] Zaid J Towfic, Jianshu Chen, and Ali H Sayed. Collaborative learning of mixture models using diffusion adaptation. In *2011 IEEE International Workshop on Machine Learning for Signal Processing*, pages 1–6. IEEE, 2011.

[81] Silvana Silva Pereira, Sergio Barbarossa, and Alba Pages-Zamora. Consensus for distributed EM-based clustering in WSNs. In *Sensor Array and Multichannel Signal Processing Workshop (SAM), 2010 IEEE*, pages 45–48. IEEE, 2010.

[82] D. Garey, M. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. WH Freeman and Company: San Francisco, CA, USA, 1979.

[83] Cassio G Lopes and Ali H Sayed. Diffusion least-mean squares over adaptive networks: Formulation and performance analysis. *IEEE Transactions on Signal Processing*, 56(7):3122–3136, 2008.

[84] Noriyuki Takahashi, Isao Yamada, and Ali H Sayed. Diffusion least-mean squares with adaptive combiners: Formulation and performance analysis. *IEEE Transactions on Signal Processing*, 58(9):4795–4810, 2010.

[85] Federico S Cattivelli, Cassio G Lopes, and Ali H Sayed. Diffusion recursive least-squares for distributed estimation over adaptive networks. *IEEE Transactions on Signal Processing*, 56(5):1865–1877, 2008.

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

[87] Kenneth Lange. A gradient algorithm locally equivalent to the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 425–437, 1995.

[88] Ali H Sayed and Cassio G Lopes. Adaptive processing over distributed networks. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 90(8):1504–1510, 2007.

[89] CassioG Lopes and Ali H Sayed. Distributed processing over adaptive networks. In *Proc. adaptive sensor array processing workshop*, pages 1–5, 2006.

[90] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[91] Stephen Boyd, Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah. Randomized gossip algorithms. *IEEE/ACM Transactions on Networking (TON)*, 14(SI):2508–2530, 2006.

[92] Tuncer C Aysal, Anand D Sarwate, and Alexandros G Dimakis. Reaching consensus in wireless networks with probabilistic broadcast. In *Communication, Control, and Computing, 2009. Allerton 2009. 47th Annual Allerton Conference on*, pages 732–739. IEEE, 2009.

[93] Tuncer Can Aysal, Mehmet Ercan Yildiz, Anand D Sarwate, and Anna Scaglione. Broadcast gossip algorithms for consensus. *IEEE Transactions on Signal processing*, 57(7):2748–2761, 2009.

[94] Dusan Jakovetic, Joao Xavier, and José MF Moura. Cooperative convex optimization in networked systems: Augmented lagrangian algorithms with directed gossip communication. *IEEE Transactions on Signal Processing*, 59(8):3889–3902, 2011.

[95] Ali Abur and Antonio Gomez Exposito. *Power system state estimation: theory and implementation*. CRC press, 2004.

[96] Aranya Chakrabortty and Pramod P Khargonekar. Introduction to wide-area control of power systems. In *2013 American Control Conference*, pages 6758–6770. IEEE, 2013.

[97] Yih-Fang Huang, Stefan Werner, Jing Huang, Neelabh Kashyap, and Vijay Gupta. State estimation in electric power grids: Meeting new challenges presented by the requirements of the future grid. *IEEE Signal Processing Magazine*, 29(5):33–43, 2012.

[98] Reza Ebrahimian and Ross Baldick. State estimation distributed processing. *IEEE Trans. Power Syst*, 15(4):1240–1246, 2000.

[99] Liang Zhao and Ali Abur. Multi area state estimation using synchronized phasor measurements. *IEEE Transactions on Power Systems*, 20(2):611–617, 2005.

[100] Antonio Gomez-Exposito, Ali Abur, Antonio de la Villa Jaen, and Catalina Gomez-Quiles. A multilevel state estimation paradigm for smart grids. *Proceedings of the IEEE*, 99(6):952–976, 2011.

[101] Antonio J Conejo, Sebastian de la Torre, and Miguel Canas. An optimization approach to multiarea state estimation. *IEEE Transactions on Power Systems*, 1(22):213–221, 2007.

[102] Jose Beleza Carvalho and F Maciel Barbosa. Distributed processing in power system state estimation. In *Electrotechnical Conference, 2000. MELECON 2000. 10th Mediterranean*, volume 3, pages 1128–1131. IEEE, 2000.

[103] Xin Tai, Zhiyun Lin, Minyue Fu, and Yuanzhang Sun. A new distributed state estimation technique for power networks. In *2013 American Control Conference*, pages 3338–3343. IEEE, 2013.

[104] Reza Ebrahimian and Ross Baldick. State estimation distributed processing. *IEEE Trans. Power Syst*, 15(4):1240–1246, 2000.

[105] Alcir Monticelli. *State estimation in electric power systems: a generalized approach*, volume 507. Springer Science & Business Media, 1999.

[106] Dennis J. E. Mor J. J. Broyden, C. G. On the local and superlinear convergence of quasi-newton methods. *IMA Journal of Applied Mathematics*, 12(3):223–245, 1973.

[107] Alexandros G Dimakis, Soummya Kar, José MF Moura, Michael G Rabbat, and Anna Scaglione. Gossip algorithms for distributed signal processing. *Proceedings of the IEEE*, 98(11):1847–1864, 2010.

[108] Bjorn Johansson, Tamás Keviczky, Mikael Johansson, and Karl Henrik Johansson. Subgradient methods and consensus algorithms for solving convex optimization problems. In *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, pages 4185–4190. IEEE, 2008.

[109] Angelia Nedic and Asuman Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, 2009.

[110] Ray Daniel Zimmerman, Carlos Edmundo Murillo-Sánchez, and Robert John Thomas. Matpower: Steady-state operations, planning, and analysis tools for power systems research and education. *IEEE Transactions on power systems*, 26(1):12–19, 2011.

[111] University of Washington. Power systems test case archive. *Available: http://www.ee.washington.edu/research/pstca/*.

Distributed Componentwise EM Algorithm for Mixture Models in Sensor Networks

Jia Yu and Pei-Jung Chung
Institute for Digital Communications, The University of Edinburgh, UK
j.yu@ed.ac.uk, p.chung@ed.ac.uk

***Abstract*—This work considers mixture model estimation in sensor networks in a distributed manner. In the statistical literature, the maximum likelihood (ML) estimate of mixture distributions can be computed via a straightforward application of the expectation and maximization (EM) algorithm. In sensor networks without centralized processing units, data are collected and processed locally. Modifications of standard EM-type algorithms are necessary to accommodate the characteristics of sensor networks. Existing works on the distributed EM algorithm focus mainly on estimation performance and implementation aspects. Here, we address the convergence issue by proposing a distributed EM-like algorithm that updates mixture parameters sequentially. Simulation results show that the proposed approach leads to significant gain in convergence speed and considerable saving in computational time.**

***Index Terms*—sensor networks, expectation and maximization (EM) algorithm, componentwise EM algorithm, distributed processing, mixture models**

I. Introduction

Sensor networks consist of massively distributed, small devices with limited sensing, processing, and communication capabilities. They have a broad range of environmental sensing applications, including temperature monitoring, vehicle tracking, collaborative processing of information and data collection from spatially distributed sources [1], [2], [3], [6].

In this work, it is assumed that each node in the sensor networks senses an environment that can be modeled as a mixture of normal distributions. In [14], this model was successfully applied to describe data measured by sensor networks in an inhomogeneous environment. Therein, a distributed expectation and maximization (EM)-type algorithm was derived to identify Gaussian components common to the whole network and mixing probabilities associated with each node. Methods for improving the performance of distributed EM algorithm were suggested in [10], [8], [15].

The EM algorithm is a well known numerical method for finding maximum likelihood (ML) estimates [5]. In the context of mixture models, it provides closed form solutions for estimating the means and covariance matrices of Gaussian components [11]. However, the most documented problem associated with EM is its possible slow convergence. To speed up its convergence, various approaches have been proposed in the statistical literature [7], [12]. In [4], a componentwise EM algorithm was applied to mixture models. In stead of computing all parameters simultaneously in the M-step, the componentwise EM updates the component parameters sequentially. As the numerical results shown in [4], a better convergence rate can be achieved by this flexible approach. Another advantage of the componentwise EM is that despite relaxation of the constraint on mixing probabilities, it can be shown that when the algorithm converges, the sum of mixing probabilities equals one.

To facilitate the application of the componentwise EM to sensor networks, we adopt the idea of incremental EM [11], [13] to enable local processing at sensor nodes. As will be illustrated in the following sections, given sufficient statistics from the previous node, the E- and M-step at the current node involve only local observations. Simulation results show that the proposed algorithm achieves a higher convergence rate than the distributed EM [14], leading to significant saving in overall computational time.

This paper is organized as follows. The problem and data models will be defined in Section II. Section III includes a brief description of the standard EM and componentwise EM algorithms. The distributed componentwise EM algorithm for sensor networks is developed in Section IV. Section V presents and discusses simulation results. Concluding remarks are given in Section VI.

II. Problem Formulation

Consider a sensor network consisting of M sensor nodes. The mth node records N_m independent and identically distributed data $\boldsymbol{y}_m = \{\boldsymbol{y}_{m,1}, \cdots, \boldsymbol{y}_{m,N_m}\}$. The measurements are assumed to obey Gaussian mixture distributions

$$\boldsymbol{y}_{m,i} \sim \sum_{j=1}^{J} \alpha_{m,j} \mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j),\ i = 1, \cdots, N_m \qquad (1)$$

where $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes the Gaussian density function with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. The mixing parameters $\boldsymbol{\alpha}_j = \{\alpha_{m,j}\}_{m=1}^{M}$ are potentially unique at each node, but the J mixing components are common to all nodes. Define $\boldsymbol{\theta}_j = \{\boldsymbol{\alpha}_j, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\}_{j=1}^{J}$. Then the unknown parameter set is given by $\boldsymbol{\theta} = \{\boldsymbol{\theta}_j\}_{j=1}^{J}$. Based on the measurements $\boldsymbol{y} = \{\boldsymbol{y}_m\}_{m=1}^{M}$, the problem of central interest is to compute

978-1-4799-1353-4/13/$31.00 ©2013 Crown

the maximum likelihood (ML) estimate for $\boldsymbol{\theta}$ in a distributed manner.

Let $\mathcal{N}(\boldsymbol{y}|\boldsymbol{\mu},\boldsymbol{\Sigma})$ denote the evaluation of a Gaussian density at the data sample $\boldsymbol{y}$. It is well known that maximization of the log-likelihood for the mixture model (1)

$$L(\boldsymbol{\theta}) = \sum_{m=1}^{M}\sum_{i=1}^{N_m}\log(\sum_{j=1}^{J}\alpha_{m,j}\mathcal{N}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)) \quad (2)$$

is greatly simplified by the EM-type algorithms [11] which will be described in the following section. This data model is assumed to be statistically independent in each node, but if the data are (spatially or temporally) correlated, this model can still be employed by interpreting it as *pseudolikelihood*[16].

III. Standard EM and Componentwise EM Algorithms

The formulation of the mixture problem in the EM framework is achieved by augmenting the observed data vector $\boldsymbol{y} = \{\boldsymbol{y}_m\}_{m=1}^{M}$ with the associated component-label vectors $\boldsymbol{z} = \{\boldsymbol{z}_m\}_{m=1}^{M}$ where $\boldsymbol{z}_m = \{z_{m,i}\}_{i=1}^{N_m}$. Each $z_{m,i}$ takes on a value from the set $\{1,\ldots,J\}$, where $z_{m,i}=j$ indicates that $y_{m,i}$ was generated by the jth mixture component

$$\boldsymbol{y}_{m,i} \sim \mathcal{N}(\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j). \quad (3)$$

The complete data log-likelihood $L_c(\boldsymbol{\theta})$ is then given by

$$L_c(\boldsymbol{\theta}) = \sum_{m=1}^{M}\sum_{i=1}^{N_m}\sum_{j=1}^{J} z_{m,i,j}(\log\alpha_{m,j} + \log\mathcal{N}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)). \quad (4)$$

Starting from an initial estimate $\boldsymbol{\theta}^0$, the standard EM algorithm iterates between the E (expectation) and M (maximization) step. In the E-step, given the current estimate $\boldsymbol{\theta}^t$, the conditional expectation of the complete data log-likelihood is computed as follows

$$\begin{aligned} Q(\boldsymbol{\theta},\boldsymbol{\theta}^t) &= E[L_c(\boldsymbol{\theta})|\boldsymbol{y},\boldsymbol{\theta}^t] \\ &= \sum_{m=1}^{M}\sum_{i=1}^{N_m}\sum_{j=1}^{J} w_{m,i,j}^{t+1}(\log\alpha_{m,j} + \log\mathcal{N}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)), \end{aligned} \quad (5)$$

where

$$w_{m,i,j}^{t+1} = \frac{\alpha_{m,j}^t\mathcal{N}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_j^t,\boldsymbol{\Sigma}_j^t)}{\sum_{j=1}^{J}\alpha_{m,j}^t\mathcal{N}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_j^t,\boldsymbol{\Sigma}_j^t)} \quad (6)$$

is the posterior probability that the ith sample at node m belongs to the jth component given the observed value $\boldsymbol{y}_{m,i}$.

In the M-step, the parameters are computed by maximizing the complete data log-likelihood (5)

$$\boldsymbol{\theta}^{t+1} = \arg\max_{\theta} Q(\boldsymbol{\theta},\boldsymbol{\theta}^t), \quad (7)$$

leading to the following update formulae for $j = 1,\cdots,J$.

$$\alpha_{m,j}^{t+1} = \frac{1}{N_m}\sum_{i=1}^{N_m} w_{m,i,j}^{t+1}, \quad m = 1,\cdots,M \quad (8)$$

$$\boldsymbol{\mu}_j^{t+1} = \frac{\boldsymbol{a}_j^{t+1}}{w_j^{t+1}}, \quad (9)$$

$$\boldsymbol{\Sigma}_j^{t+1} = \frac{\boldsymbol{b}_j^{t+1}}{w_j^{t+1}} - \boldsymbol{\mu}_j^{t+1}\boldsymbol{\mu}_j^{t+1\prime}, \quad (10)$$

where the summary quantities are defined as follows

$$w_j^{t+1} = \sum_{m=1}^{M}\sum_{i=1}^{N_m} w_{m,i,j}^{t+1}, \quad (11)$$

$$\boldsymbol{a}_j^{t+1} = \sum_{m=1}^{M}\sum_{i=1}^{N_m} w_{m,i,j}^{t+1}\boldsymbol{y}_{m,i}, \quad (12)$$

$$\boldsymbol{b}_j^{t+1} = \sum_{m=1}^{M}\sum_{i=1}^{N_m} w_{m,i,j}^{t+1}\boldsymbol{y}_{m,i}\boldsymbol{y}'_{m,i}. \quad (13)$$

The E- and M-steps are alternated repeatedly until the difference between likelihoods of consecutive iterates $L(\boldsymbol{\theta}^{t+1}) - L(\boldsymbol{\theta}^t)$ is less than a pre-specified small number ϵ.

To improve the convergence rate of the standard EM algorithm, a componentwise EM algorithm for mixture models (CEMM) was proposed in [4]. Rather than computing all parameters simultaneously, the CEMM algorithm considers the decomposition of the parameter vector $\boldsymbol{\theta}$ into component parameter vectors $\{\boldsymbol{\alpha}_j,\boldsymbol{\theta}_j\}$, $j = 1,\cdots,J$ and updates only one component at a time. More specifically, each iteration consists of J cycles and the conditional expectation (5) is computed each time the parameter vector associated with jth component is updated. As pointed out in [4], the decoupling of parameter updates implies the use of the smallest admissible missing data space and leads to faster convergence than the standard EM.

IV. Distributed Componentwise EM Algorithm

Motivated by the superior convergence behavior of the componentwise EM algorithm, we propose a distributed componentwise EM algorithm for mixtures in sensor networks. In some sensor network models, a high-performance centralized unit is involved to solve the estimation problems. But relying on the centralized unit is undesirable in scenarios in which communications between sensor nodes are much more costly than the computational cost at sensor nodes. In the following, we consider the message passage model for sensor networks proposed in [14] (see Fig.1). Similar to the distributed EM, our algorithm also exploits the idea of incremental EM [13] to facilitate local processing. The idea behind incremental EM is to divide the observed data into several blocks and implement the E-step for only one block of observations at a time before performing a M-step

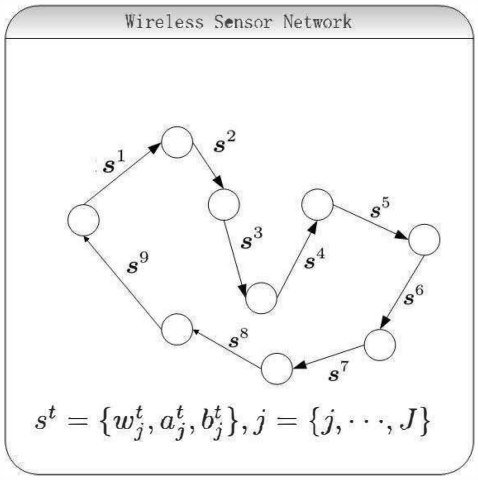


Fig. 1. Communication/iteration cycle in a sensor network

[11]. Here, the observed data at each node is considered as one data block. By applying the incremental EM, the componentwise EM can be implemented so that at node m, given the summary quantities (11), (12) and (13) from the previous node $(m-1)$, only local data $\boldsymbol{y}_m$ is involved.

Assume that at time $(t+1)$, node m receives summary statistics $\boldsymbol{a}_j^t, \boldsymbol{b}_j^t, w_j^t$ from the previous node. Define the estimate after the tth iteration as

$$\boldsymbol{\theta}^t = \{\boldsymbol{\theta}_1^t, \cdots, \boldsymbol{\theta}_J^t\}, \tag{14}$$

where $\boldsymbol{\theta}_j^t$ include the estimate for the jth component $\{\boldsymbol{\alpha}_j^t, \boldsymbol{\mu}_j^t, \boldsymbol{\Sigma}_j^t\}$. At the beginning of the $(t+1)$th iteration, the initial estimates for the mean and covariance matrix are obtained from the summary statistics as follows:

$$\boldsymbol{\mu}_j^t = \frac{\boldsymbol{a}_j^t}{w_j^t}, \quad \boldsymbol{\Sigma}_j^t = \frac{\boldsymbol{b}_j^t}{w_j^t} - \boldsymbol{\mu}_j^t \boldsymbol{\mu}_j^{t'}, \quad j = 1, \cdots, J. \tag{15}$$

Set $\boldsymbol{\theta}^{[t+1,0]} = \boldsymbol{\theta}^t$. The parameters associated with the jth components $\boldsymbol{\theta}_j^t$ are updated sequentially in the proposed algorithm as follows.

For $j = 1, \cdots, J$

E-step

$$w_{m,i,j}^{t+1} = \frac{\alpha_{m,j}^t \mathcal{N}(y_{m,i}|\boldsymbol{\mu}_j^t, \boldsymbol{\Sigma}_j^t)}{\sum_{k=1}^{j-1} \alpha_{m,k}^{t+1} \mathcal{N}(y_{m,i}|\boldsymbol{\mu}_k^{t+1}, \boldsymbol{\Sigma}_k^{t+1}) + \sum_{k=j}^{J} \alpha_{m,k}^t \mathcal{N}(y_{m,i}|\boldsymbol{\mu}_k^t, \boldsymbol{\Sigma}_k^t)}. \tag{16}$$

M-step

$$\alpha_{m,j}^{t+1} = \frac{1}{N_m} \sum_{i=1}^{N_m} w_{m,i,j}^{t+1}, \tag{17}$$

$$\boldsymbol{\mu}_{m,j}^{t+1} = \frac{\boldsymbol{a}_{m,j}^{t+1}}{w_{m,j}^{t+1}}, \tag{18}$$

$$\boldsymbol{\Sigma}_{m,j}^{t+1} = \frac{\boldsymbol{b}_{m,j}^{t+1}}{w_{m,j}^{t+1}} - \boldsymbol{\mu}_{m,j}^{t+1} \boldsymbol{\mu}_{m,j}^{t+1'}, \tag{19}$$

where the local summary statistics $w_{m,j}, \boldsymbol{a}_{m,j}, \boldsymbol{b}_{m,j}$ are

$$w_{m,j}^{t+1} = \frac{1}{N_m} \sum_{i=1}^{N_m} w_{m,i,j}^{t+1}, \tag{20}$$

$$\boldsymbol{a}_{m,j}^{t+1} = \sum_{i=1}^{N_m} w_{m,i,j}^{t+1} \boldsymbol{y}_{m,i}, \tag{21}$$

$$\boldsymbol{b}_{m,j}^{t+1} = \sum_{i=1}^{N_m} w_{m,i,j}^{t+1} \boldsymbol{y}_{m,i} \boldsymbol{y}'_{m,i}. \tag{22}$$

The estimate at the jth cycle is given by

$$\boldsymbol{\theta}^{[t+1,j]} = \{\boldsymbol{\theta}_1^{t+1}, \cdots, \boldsymbol{\theta}_j^{t+1}, \boldsymbol{\theta}_{j+1}^t, \cdots, \boldsymbol{\theta}_J^t\}. \tag{23}$$

End; $\%j$

After J cycles, the output of the $(t+1)$th iteration is given by:

$$\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^{[t+1,J]}. \tag{24}$$

Then the local summary statistics are computed with the new estimate $\boldsymbol{\theta}_m^{t+1}$ as follows:

$$w_j^{t+1} = w_j^t + w_{m,j}^{t+1} - w_{m,j}^t, \tag{25}$$

$$\boldsymbol{a}_j^{t+1} = \boldsymbol{a}_j^t + \boldsymbol{a}_{m,j}^{t+1} - \boldsymbol{a}_{m,j}^t, \tag{26}$$

$$\boldsymbol{b}_j^{t+1} = \boldsymbol{b}_j^t + \boldsymbol{b}_{m,j}^{t+1} - \boldsymbol{b}_{m,j}^t. \tag{27}$$

Note that the old values of summary statistics are replaced by updated values at node m. In addition, the computations of the posterior probabilities (16) and the estimates (17), (18) and (19) involve only data at node m.

The major difference of the proposed componentwise approach from the distributed EM algorithm is as follows. In the distributed EM algorithm (DEM) [14], the parameters associated with all components are updated simultaneously. The E-step is evaluated only once at the beginning of the iteration. In the proposed algorithm, each component parameter set $\boldsymbol{\theta}_j$ is computed sequentially and the posterior probability $w_{m,i,j}$ (16) is evaluated at each cycle. The computational time is only slightly increased by the multiple E-steps in comparison to the distributed EM algorithm. Simulation results in the next section will show that our approach leads to a much faster convergence rate of log-likelihood than the distributed EM algorithm.

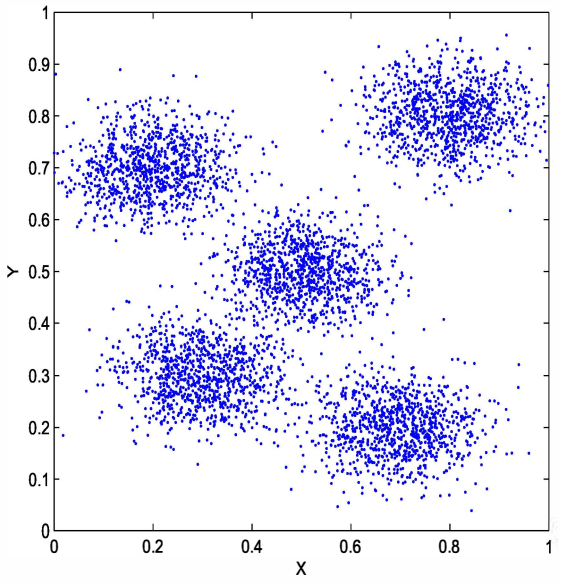


Fig. 2. Data distribution

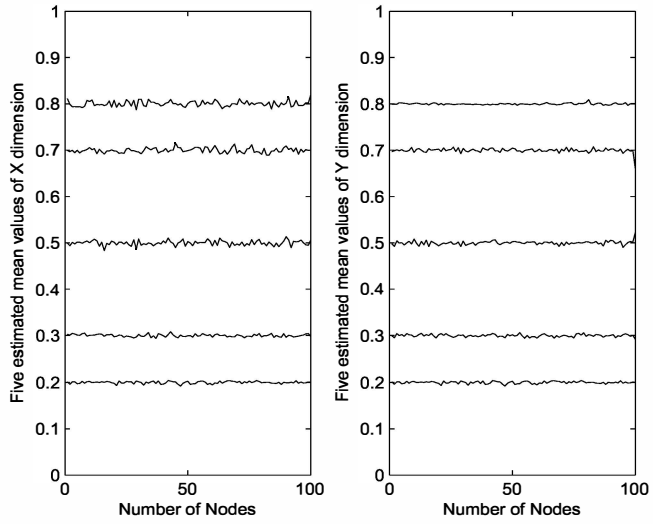


Fig. 3. Estimates for mean values by the distributed componentwise EM algorithm.

V. Simulation Results

In this section, we demonstrate the feasibility of the proposed algorithm by simulated data. In the simulation, we consider a sensor network with $M = 100$ nodes. This sensor network fulfills the communication requirement specified in [9]. The number of data samples at each node is $N_m = 100$. The observations are generated from $J = 5$ Gaussian components distributed as in Fig. 2. Each component is a $2D$ Gaussian density, which can represent environment data clusters. In the first 40 nodes, 60% observations come from the first Gaussian component and other 40% observations evenly from the other four Gaussian components, i.e. $\alpha_{m,1} = 60\%, \alpha_{m,2} = \alpha_{m,3} = \alpha_{m,4} = \alpha_{m,5} = 10\%$ for $m = 1, \cdots, 40$. In the next 30 nodes, 70% observations come from the second and third Gaussian components and other 30% observations evenly from the other three components, i.e. for $m = 41, \cdots, 70$, $\alpha_{m,1} = \alpha_{m,4} = \alpha_{m,5} = 10\%, \alpha_{m,2} = 40\%, \alpha_{m,3} = 30\%$. For $m = 71, \cdots, 100$, 70% observations come from the last two Gaussian component and other 30% observations evenly from the other three Gaussian components $\alpha_{m,1} = \alpha_{m,2} = \alpha_{m,3} = 10\%, \alpha_{m,4} = 40\%, \alpha_{m,5} = 30\%$. The component parameters are given by $\boldsymbol{\mu}_1 = [0.2, 0.7]'$, $\boldsymbol{\mu}_2 = [0.7, 0.2]'$, $\boldsymbol{\mu}_3 = [0.3, 0.3]'$, $\boldsymbol{\mu}_4 = [0.5, 0.5]'$, $\boldsymbol{\mu}_5 = [0.8, 0.8]'$.

For comparison, we apply both the proposed distributed componentwise EM algorithm and the distributed EM algorithm [14] to the same batch of data. As shown in Fig. 3, the estimates for the x- and y-components of means are close to the reference values. In Fig. 4, the log-likelihood values

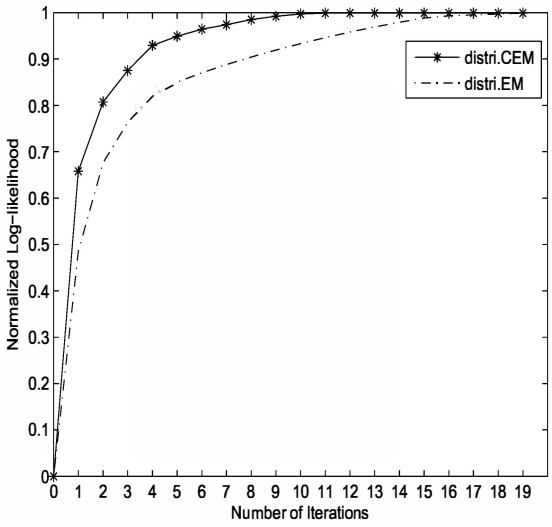


Fig. 4. Comparison of log-likelihood versus iterations for the distributed componentwise EM algorithm and the distributed EM algorithm.

are plotted versus iterations. The proposed algorithm requires on average only 10 iterations to attain the maximal value of log-likelihood, while the distributed EM algorithm requires 16 iterations to converge. As the complexity of each iteration required by both algorithms is almost the same, this implies 37% saving in overall computational time by the proposed algorithm.

VI. Conclusion

In this work, we proposed a distributed componentwise EM algorithm for mixture models in sensor networks. The proposed algorithm is characterized by local processing capabilities and sequential computations of component parameters. The ability to process data locally is of particular interest to sensor networks with computationally powerful nodes and require costly node-to-node communications. More importantly, the componentwise update of the mixture parameters leads to significant improvement in convergence rate compared to the distributed EM algorithm [14]. Simulation results show that the number of iterations required by the proposed algorithm is about 40% less than that required by the distributed EM algorithm. Given the advantages of computational efficiency and simple implementation, we believe that the proposed distributed componentwise EM algorithm is a powerful tool for estimating mixture models in sensor networks. The important issue on convergence of the proposed algorithm and other finite mixture model based on non-Gaussian distributions [11] will be addressed in future publications.

References

[1] L. F. Akyildiz and W. Su and Y. Sankarasubramaniam and Erdal Cayirci, *A survey on sensor networks,* 2002.

[2] P. K. Varshney. *Distributed Detection and Data Fusion,* Springer-verlad, 1997.

[3] J. F. Chaamberland and V. V. Veeravalli, *Decentralized detection in sensor networks,* IEEE Transactions on Signal processing, vol. 51, no. 2, pp.407-416, Feb. 2003.

[4] G. Celeux and S. Chretien and F. Forbes and A. Mkhadri, *A component-wise EM algorithm for mixtures,* Journal of Computational and Graphical Statistics, Vol 10(4), 2001.

[5] A. P. Dempster and N. Laird and D. B. Rubin, *Maximum likelihood from incomplete data via the EM algorithm*, Journal of the Royal Statistical Society, Series B, 29, pp. 1-38.

[6] D. Estrin and R. Govindan and J. Heidemann and S. Kumar, *Next century challenges: scalable coordination in sensor networks,* Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking, pp. 263-270, 1999.

[7] J. A. Fessler and A. O. Hero, *Space alternating generalized Expectation-Maximization algorithm*, IEEE Transactions on Signal Processing, vol. 42, no. 10, pp. 2664-2677, Oct. 1994.

[8] D. Gu, *Distrbuted EM algorithm for Gaussian mixtures in sensor network,* IEEE Transactions on Neural Network, vol. 19, no. 7, pp. 1154-1166, Jul. 2008

[9] P. Gupta and P. R. Kumar, *The capacity of wireless networks,* IEEE Transactions on Information Theory, vol. 46, no. 2, pp. 388-404, Nov. 2000.

[10] W. Kowalczyk and N. Vlassis, *Newcast EM,* in Proc. Adv. Neural Inf. Process. Syst., Vancouver, BC, Canada, pp. 13-18, Dec. 2005.

[11] G. McLachlan and D. Peel, *Finite Mixture Models,* Wiley Series in Probability and Statistics, Oct. 2000.

[12] X. L. Meng and D. van Dyk , *The EM Algorithm – An old folk song sung to the fast tune*, Journal of the Royal Statistical Society, Ser. B., vol. 59, pp. 511-567, 1997.

[13] R. Neal and G. E. Hinton, *A view of the EM algorithm that justifies incremental, sparse, and other variants*, in *Learning in Graphical Models*, Kluwer Academic Publishers, 1998, 355-368.

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

[15] J. Wolfe and A. Haghighi and D. Klein, *Fully distributed EM for very large datasets,* in Proc. 25th Int. Conf. Mach. Learn., Helsinki, Finland, pp. 1184-1191, Jul. 6-8, 2008.

[16] J. Besag, On the statistical analysis of dirty pictures, Journal of the Royal Statistical Society, Series B, vol.48, no.3, pp. 259-302.

Diffusion-based EM Gradient Algorithm for Density Estimation in Sensor Networks

Jia Yu and John Thompson
Institute for Digital Communications, The University of Edinburgh, UK
j.yu@ed.ac.uk, john.thompson@ed.ac.uk

***Abstract*—This paper considers mixture density estimation in an asynchronous sensor networks in a distributed manner. In the statistical literature, the maximum likelihood (ML) estimate of mixture distributions can be computed via a straightforward application of the expectation and maximization (EM) algorithm. In a random sensor networks, data are required to collected and processed at local decentralized processing units. Reformulations of standard EM-type algorithms are necessary to accommodate the characteristics of sensor networks. Existing works on the distributed EM implementation focus mainly on synchronous network. Here, we address the issue of asynchronous behaviors by proposing a diffusion-based EM gradient algorithm that updates estimates under ATC diffusion strategy. Simulation results show the robustness and scalability of the proposed approach in the presence of additional randomness of asynchronous events.**

***Index Terms*—sensor networks, expectation and maximization (EM) algorithm, asynchronous network, distributed processing, Gaussian mixtures**

I. Introduction

Mixture density estimation belongs to the general class of unsupervised learning problems and has a broad range of applications, including environmental monitoring, pattern classification and recognition for image analysis, and also for clustering. In presence of latent variable, the EM algorithm is a well known numerical method for finding maximum likelihood (ML) estimates [1]. It starts from an initial guess, the method alternates between an expectation (E) step, where the expected log-likelihood function of the observations is evaluated based on the current estimates, and a maximization (M) step, where the maximization is performed using conditional log-likelihood function of the E-step to find the new estimates. A distributed implementation of EM algorithm in a wireless sensor network (WSN) entails therefore a modification of the operations such that such that they can be executed at each local node.

Related contributions in the literature propose distributed EM implementations where the global sufficient statistics are computed using for incremental schemes [2], [3], a consensus-based scheme [4], and diffusion strategies [5], [6], [7]. Among these schemes, diffusion strategies are attractive because they do not require different nodes to converge to the same global statistics, and individual nodes are allowed update parameters through their own local information. Based on these merits, a diffusion-based distributed EM is proposed in [5] where the authors use the Robbins-Monro stochastic procedure to approximate the centralized EM approach, and a diffusion adaption algorithm is proposed for general mixture models in [6], where the adaptive diffusion process is executed in M-step rather than solving a closed form optimization. In [7], another diffusion-type estimator is developed, where the propagation of information across the network is embedded in the iterative updates of the parameters, where a faster term for information is combined with a slower term for information averaging.

The referred algorithms are limited to the synchronous network model, where a coordinated behavior is required throughout the network. In this paper we present a diffusion-Based EM gradient algorithm for Gaussian mixture models in WSNs. The method is based on a EM gradient method [8] derived for Gaussian mixtures. We develop this method with asynchronous adaptive diffusion scheme, and address here the general case of density estimation. The main idea behind the proposed algorithm is that the diffusion of the information across the network is embedded in the Expectation step to update parameters. In the Maximization step, gradient based optimization is utilized under the asynchronous ATC (adapt-then-combine) diffusion rule [9]. The advantage of the proposed with respect to the synchronous diffusion algorithm is more flexible, individual nodes in the network may stop updating their solutions or may stop sending or receiving information in a random manner and without coordination with other nodes. This flexibility can be translated into energy savings, a critical issue specially in large-scale deployments. Although asynchrony events degrade performance as expected, numerical examples provided here still show that performance of the proposed algorithm are robustness and outperforms diffusion-based distributed EM scheme [5].

978-1-5090-1749-2/16/$31.00 ©2016 IEEE

The paper is organized as follows. In section II we describe the observation model and Section III derive the expressions for the centralized EM algorithm and EM Gradient algorithm. Section IV presents the Diffusion-based EM gradient method for density estimation in asynchronous WSNs under the assumption of GMMs. Simulations results and conclusions are presented in sections V and VI respectively.

II. Problem Formulation

Consider a sensor network consisting of M sensor nodes. The mth node records N_m independent and identically distributed data sample $\boldsymbol{y}_m = \{\boldsymbol{y}_{m,1}, \cdots, \boldsymbol{y}_{m,N_m}\}$. The measurements are assumed to obey a Gaussian mixture distribution

$$\boldsymbol{y}_{m,i} \sim \sum_{j=1}^{J} \alpha_{m,j} \mathcal{P}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j),\ i = 1, \cdots, N_m \quad (1)$$

where $\mathcal{P}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes the Gaussian density function. The mixing parameters $\boldsymbol{\alpha}_j = \{\alpha_{m,j}\}_{m=1}^{M}$ are potentially unique at each node, but the number of mixing components J are common to all nodes. Define $\boldsymbol{\theta}_j = \{\boldsymbol{\alpha}_j, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\}_{j=1}^{J}$. Then the unknown parameter set is given by $\boldsymbol{\theta} = \{\boldsymbol{\theta}_j\}_{j=1}^{J}$. Based on the measurements $\boldsymbol{y} = \{\boldsymbol{y}_m\}_{m=1}^{M}$, the problem of central interest is to compute the maximum likelihood (ML) estimate for $\boldsymbol{\theta}$ in a distributed manner.

Let $\mathcal{P}(\boldsymbol{y}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denote the evaluation of a Gaussian density with at the data sample $\boldsymbol{y}$. It is well known that maximization of the log-likelihood for the mixture model in (1)

$$L(\boldsymbol{\theta}) = \sum_{m=1}^{M} \sum_{i=1}^{N_m} \log \left(\sum_{j=1}^{J} \alpha_{m,j} \mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \right) \quad (2)$$

is greatly simplified by the EM-type algorithms [11] which will be described in the following section. The data sample in this model are assumed to be statistically independent in each node, but if the data are (spatially or temporally) correlated, this model can still be employed by interpreting it as a $pseudolikelihood$ [11].

III. Standard EM and EM Gradient Algorithms

The formulation of the mixture problem in the EM framework is achieved by augmenting the observed data vector $\boldsymbol{y} = \{\boldsymbol{y}_m\}_{m=1}^{M}$ with the associated component-label vectors $\boldsymbol{z} = \{\boldsymbol{z}_m\}_{m=1}^{M}$ where $\boldsymbol{z}_m = \{z_{m,i}\}_{i=1}^{N_m}$. Each $z_{m,i}$ takes on a value from the set $\{1, \ldots, J\}$, where $z_{m,i} = j$ indicates that $y_{m,i}$ was generated by the jth mixture component

$$\boldsymbol{y}_{m,i} \sim \mathcal{P}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j). \quad (3)$$

The global complete data log-likelihood $L(\boldsymbol{\theta})$ is then given by

$$\begin{aligned} L(\boldsymbol{\theta}) &= \sum_{m=1}^{M} L_m(\boldsymbol{\theta}), \\ &= \sum_{m=1}^{M} \sum_{i=1}^{N_m} \sum_{j=1}^{J} z_{m,i,j} (\log \alpha_{m,j} + \log \mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)). \end{aligned} \quad (4)$$

where $L_m(\boldsymbol{\theta})$ is the local log-likelihood function at each node m. Starting from an initial estimate $\boldsymbol{\theta}^0$, the standard EM algorithm iterates between the E (expectation) and M (maximization) step. In the E-step, given the current estimate $\boldsymbol{\theta}^t$, the conditional expectation of the complete data log-likelihood is computed as follows

$$\begin{aligned} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^t) &= \sum_{m=1}^{M} Q_m(\boldsymbol{\theta}, \boldsymbol{\theta}^t), \\ &= \sum_{m=1}^{M} \sum_{i=1}^{N_m} \sum_{j=1}^{J} w_{m,i,j}^{t+1} (\log \alpha_{m,j} + \log \mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)). \end{aligned} \quad (5)$$

where

$$w_{m,i,j}^{t+1} = \frac{\alpha_{m,j}^t \mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_j^t, \boldsymbol{\Sigma}_j^t)}{\sum_{j=1}^{J} \alpha_{m,j}^t \mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_j^t, \boldsymbol{\Sigma}_j^t)}, \quad (6)$$

is the posterior probability that the ith sample at node m belongs to the jth component given the observed value $\boldsymbol{y}_{m,i}$.

In the M-step, the parameters are computed by maximizing the complete data log-likelihood in equation (5)

$$\boldsymbol{\theta}^{t+1} = \arg\max_{\theta} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^t). \quad (7)$$

The E- and M-steps are alternated repeatedly until the difference between likelihoods of consecutive iteration $L(\boldsymbol{\theta}^{t+1}) - L(\boldsymbol{\theta}^t)$ is less than a pre-defined small number ϵ.

Several methods can be used to improve the performance of the EM algorithm in the M-step, if the M-step cannot be computed in closed form. The most common algorithm for iteratively solving the M-step would be Newton-type method, which have quadratic convergence compared with the linear convergence experienced by the EM algorithm. Based on this knowledge, EM gradient algorithm was proposed in [8], which updates the $\boldsymbol{\theta}^t$ by

$$\begin{aligned} \boldsymbol{\theta}^{t+1} &= \boldsymbol{\theta}^t - \left[\sum_{m=1}^{M} \nabla^{20} Q_m(\boldsymbol{\theta}^t, \boldsymbol{\theta}^t) \right]^{-1} \\ &\quad \times \sum_{m=1}^{M} \nabla^{10} Q_m(\boldsymbol{\theta}^t, \boldsymbol{\theta}^t), \end{aligned} \quad (8)$$

where the operators $\nabla^{20}Q_m(\boldsymbol{\theta}^t,\boldsymbol{\theta}^t)$ and $\nabla^{10}Q_m(\boldsymbol{\theta}^t,\boldsymbol{\theta}^t)$ are the Hessian matrix and gradient vector of the local conditional log-likelihood function $Q_m(\boldsymbol{\theta}^t,\boldsymbol{\theta}^t)$. In addition, the equality $\nabla^{10}Q_m(\boldsymbol{\theta}^t,\boldsymbol{\theta}^t) = \nabla L_m(\boldsymbol{\theta}^t)$ holds, when $L_m(\boldsymbol{\theta}^t) - Q_m(\boldsymbol{\theta},\boldsymbol{\theta}^t)$ has its minimum at $\boldsymbol{\theta} = \boldsymbol{\theta}^t$. In [10], Xu and Jordan use the projection matrix $P(\boldsymbol{\theta}^t)$ to take place of the inverse of Hessian matrix as follows

$$\mathcal{A}^{t+1} = \mathcal{A}^t + \left[\sum_{m=1}^{M} P^t_{\mathcal{A}_m}\right] \sum_{m=1}^{M} \frac{\partial L_m(\boldsymbol{\theta})}{\partial \mathcal{A}}|_{\mathcal{A}=\mathcal{A}^t}, \quad (9)$$

$$\boldsymbol{\mu}_j^{t+1} = \boldsymbol{\mu}_j^t + \left[\sum_{m=1}^{M} P^t_{\boldsymbol{\mu}_{j,m}}\right] \sum_{m=1}^{M} \frac{\partial L_m(\boldsymbol{\theta})}{\partial \boldsymbol{\mu}_j}|_{\boldsymbol{\mu}_j=\boldsymbol{\mu}_j^t}, \quad (10)$$

$$\begin{aligned}\text{vec}[\boldsymbol{\Sigma}_j^{t+1}] &= \text{vec}[\boldsymbol{\Sigma}_j^t] + \left[\sum_{m=1}^{M} P^t_{\boldsymbol{\Sigma}_{j,m}}\right] \\ &\times \sum_{m=1}^{M} \frac{\partial L_m(\boldsymbol{\theta})}{\partial \text{vec}[\boldsymbol{\Sigma}_j]}|_{\boldsymbol{\Sigma}_j=\boldsymbol{\Sigma}_j^t}, \end{aligned} \quad (11)$$

where vec$[C]$ denotes the vector obtained by stacking the column vectors of matrix C, $\mathcal{A}$ denotes the vector of mixing proportions $[\alpha_1,\cdots,\alpha_J]^T$ and

$$P^t_{\mathcal{A}_m} = \frac{\{\text{diag}[\alpha^t_{1,m},\cdots,\alpha^t_{J,m}] - \mathcal{A}_m{}^t(\mathcal{A}_m{}^t)^T\}}{N_m}, \quad (12)$$

$$P^t_{\boldsymbol{\mu}_{j,m}} = \frac{\boldsymbol{\Sigma}_j^t}{\sum_{i=1}^{N_m} w^{t+1}_{m,i,j}}, \quad (13)$$

$$P^t_{\boldsymbol{\Sigma}_{j,m}} = \frac{2}{\sum_{i=1}^{N_m} w^{t+1}_{m,i,j}} \boldsymbol{\Sigma}_j{}^t \otimes \boldsymbol{\Sigma}_j{}^t, \quad (14)$$

where $\otimes$ denotes the Kronecker product. Using the notation

$$\boldsymbol{\theta} = [\mathcal{A}^T,\boldsymbol{\mu}_1^T,\cdots,\boldsymbol{\mu}_J^T,\text{vec}[\boldsymbol{\Sigma}_1]^T,\cdots,\text{vec}[\boldsymbol{\Sigma}_J]^T]^T, \quad (15)$$

$$P(\boldsymbol{\theta}) = \text{diag}[P_{\mathcal{A}},P_{\boldsymbol{\mu}_1},\cdots,P_{\boldsymbol{\mu}_J},P_{\boldsymbol{\Sigma}_1},\cdots,P_{\boldsymbol{\Sigma}_j}], \quad (16)$$

we can obtain

$$\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t + \left[\sum_{m=1}^{M} P_m(\boldsymbol{\theta}^t)\right] \sum_{m=1}^{M} \nabla L_m(\boldsymbol{\theta}^t) \quad (17)$$

Hence, iterative EM algorithm can be considered as a variant of quasi-Newton Methods. In addition, although the EM and EM gradient algorithm are guaranteed to converge to a local maximum of the likelihood function, the result is sensitive to the initialization of the parameters. Therefore, in order to start, a suitable initializer is needed. Notice that computation of the posteriori probabilities at M-step require knowledge of local information only, whereas the estimates in (6) require knowledge of global information. Therefore, a distributed implementation of the EM algorithm entails local data processing and sharing of information.

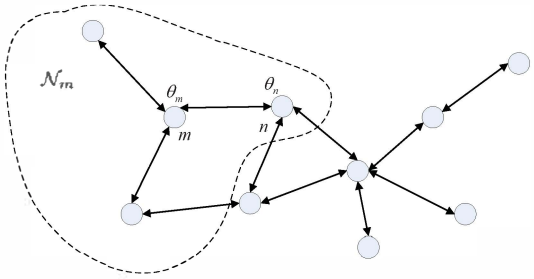


Fig. 1. A network of integrator nodes in which node m receives the state $\boldsymbol{\theta}_n$ of its neighbor, node n

IV. Diffusion EM Gradient Algorithm

Based on the gradient EM version algorithm, we propose a distributed EM algorithm scheme where the summations among all observations in (6) are computed by the asynchronous diffusion strategy [9]. In the following , we consider an asynchronous Bernoulli model in wireless sensor network, the WSN composed of node N, where each node adopt a random "on-off" policy to reduce energy consumption. The communications for each node are restricted to a closed neighborhood, and the information flow among the nodes is described by means of an undirected graph $\mathcal{G}$ ={$\mathcal{N},\mathcal{E}$}, where $\mathcal{N}$ is the sets of the nodes and $\mathcal{E}$ is the set of edges. The unordered pair $\{m,n\} \in \mathcal{E}$ if there exists an edge between node m and n. The neighborhood of node m is defined as $\mathcal{N}_m = \{n \mid \{m,n\} \in \mathcal{E}\}$ shown in Fig 1. Further, we employ the EM gradient method at each node, and assume observations of different nodes are statistically independent. In the E-step, we use an intermediate estimate $\boldsymbol{\theta}_m^t$ of the unknown $\boldsymbol{\theta}$ at node m, and the local conditional log-likelihood function is defined as

$$\begin{aligned}&Q_m(\boldsymbol{\theta},\boldsymbol{\theta}_m^t) = \\ &\sum_{i=1}^{N_m}\sum_{j=1}^{J} w^{t+1}_{m,i,j}(\log\alpha_{m,j} + \log\mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_{m,j},\boldsymbol{\Sigma}_{m,j})),\end{aligned} \quad (18)$$

where

$$w^{t+1}_{m,i,j} = \frac{\alpha^t_{m,j}\mathcal{N}(y_{m,i}|\boldsymbol{\mu}^t_{m,j},\boldsymbol{\Sigma}^t_{m,j})}{\sum_{j=1}^{J}\alpha^t_{m,j}\mathcal{N}(y_{m,i}|\boldsymbol{\mu}^t_{m,j},\boldsymbol{\Sigma}^t_{m,j})}. \quad (19)$$

The main difference between $w^{t+1}_{m,i,j}$ in (6) and (19) is that (6) is computed using the global estimates $\boldsymbol{\mu}_j^t,\boldsymbol{\Sigma}_j^t$, whereas computing (19) only requires local estimates $\boldsymbol{\mu}^t_{m,j},\boldsymbol{\Sigma}^t_{m,j}$ at each node m. By means of local periodic data exchanges, the local information in (19) is appropriately diffused over the network. In the M-step, the ATC

diffusion-oriented optimization method is introduce to find the estimates, whose updates are given by

$$\boldsymbol{\psi}_m^{t+1} = \boldsymbol{\theta}_m^t + \boldsymbol{a}_m^{t+1}\boldsymbol{d}_m^t, \tag{20}$$

$$\boldsymbol{\theta}_m^{t+1} = \sum_{n\in\mathcal{N}_m} \boldsymbol{b}_{n,m}^{t+1}\boldsymbol{\psi}_n^{t+1}, \tag{21}$$

where

$$\boldsymbol{d}_m^t = P_m(\boldsymbol{\theta}_m^t)\nabla L_m(\boldsymbol{\theta}_m^t) \tag{22}$$

is the local exact descent to the estimate using a Newton-like method in WSN. There are two operations in this scheme, the first step involves local adaption, where node m update its local estimates from $\boldsymbol{\theta}_m^t$ to an intermediate value $\boldsymbol{\psi}_m^{t+1}$. The second step is a combination step, where the combination of intermediate estimates $\{\boldsymbol{\psi}_n^{t+1}\}$ from neighborhood of node m is used to calculate the new estimates $\boldsymbol{\theta}_m^{t+1}$. In the adaption step, node m enters an active mode with probability $0 < q_m < 1$ and performs (20), and it enters a sleep mode with probability $1 - q_m$ to save energy. The random step-sizes $\boldsymbol{a}_m^{t+1}$ that are used in (20) depend on the probability q_m and are required to satisfy

$$\boldsymbol{a}_m^{t+1} = \begin{cases} \boldsymbol{a}_m, & \text{with probability } q_m \\ 0, & \text{with probability } 1 - q_m \end{cases} \tag{23}$$

where $\boldsymbol{a}_m$ is a constant step-size. The underlying topology of network is assumed to be fixed. In the combination step, each node m is allowed to randomly select its neighborhood n with probability $0 < p_m < 1$ and performs (21) for saving communication costs. The combination coefficients $\{\boldsymbol{b}_{n,m}^{t+1}\}$ are nonnegative parameters and are required to satisfy the following constraints

$$\boldsymbol{b}_{n,m}^{t+1} = \begin{cases} \boldsymbol{b}_{n,m} > 0, & \text{with probability } p_m \\ 0, & \text{with probability } 1 - p_m \end{cases} \tag{24}$$

for all $n \in \mathcal{N}_m \setminus \{m\}$, and node m is required to adjust its own weight $\boldsymbol{b}_{m,m}^{t+1}$ at each update via

$$\boldsymbol{b}_{m,m}^{t+1} = 1 - \sum_{n\in\mathcal{N}_m\setminus\{m\}} \boldsymbol{b}_{n,m}^{t+1} \tag{25}$$

to guarantee $\sum_{n\in\mathcal{N}_m} \boldsymbol{b}_{n,m}^{t+1} = 1$. Let B denote the $N\times N$ combination matrix whose $\{m,n\}$ entry is $\boldsymbol{b}_{n,m}$, and B is left-stochastic matrix which satisfy $B^T\mathbb{1}_N = \mathbb{1}_N$, where $\mathbb{1}_N$ is the $N \times 1$ all-one vector. Notice that $\boldsymbol{a}_m^{t+1}$ and $\{\boldsymbol{b}_{n,m}^{t+1}\}$ are mutually independent, and the use of these distributed control parameters enable diffusion strategies process various type of asynchronous network events. In the following, numerical results will verify its convergence and robustness.

V. Simulation Results

In this section, we demonstrate the feasibility of the proposed algorithm by simulated data. In the simulation, we consider a sensor network with $M = 100$ nodes. This sensor network fulfills the communication requirement specified in [9] with connectivity radius $r = 0.5$. The number of data samples at each node is $N_m = 100$. The observations are generated from $J = 2$ distributed Gaussian components. Each component is a 1-dimensional Gaussian mixtures density, which can represent environment data clusters. In the first 50 nodes, 60% of the observations come from the first Gaussian component and other 40% observations evenly from the second Gaussian component, i.e. $\alpha_{m,1} = 0.6, \alpha_{m,2} = 0.4$ for $m = 1, \cdots, 50$. In the last 50 nodes, 30% observations come from the first Gaussian component and other 70% observations evenly from the second component, i.e. for $m = 51, \cdots, 100$, $\alpha_{m,1} = 0.3, \alpha_{m,2} = 0.7$. The component means and variances are given by $\boldsymbol{\mu}_1 = 5$, $\boldsymbol{\mu}_2 = 10$, $\boldsymbol{\sigma}_1^2 = 1$, $\boldsymbol{\sigma}_2^2 = 4$. The step-size $\boldsymbol{a}_m = 0.05$ are uniform across the network, and the proposed algorithm is run a diffusion combination matrix B under Metropolis rule [9] with entries defined as

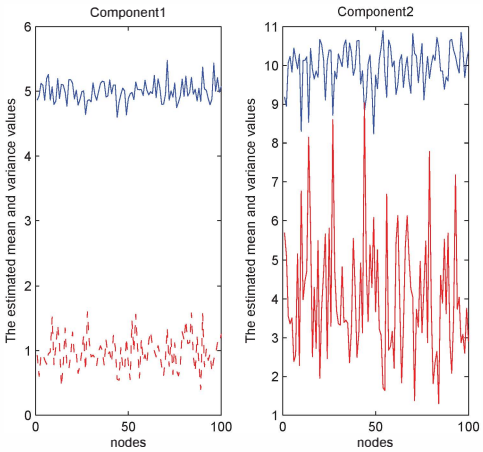
$$\boldsymbol{b}_{m,n} = \begin{cases} 1/(\max\{|\mathcal{N}_m|, |\mathcal{N}_n|\}), & n \in \mathcal{N}_m \\ 1 - \sum_{k\in\mathcal{N}_m\setminus\{m\}} \boldsymbol{b}_{n,m}, & m = n \\ 0, & \text{otherwise.} \end{cases} \tag{26}$$

where $|\cdot|$ denotes the cardinality. For comparison, we apply both the proposed asynchronous diffusion-based EM gradient algorithm, the diffusion-based distributed EM algorithm (DDEM) [5] and standard EM algorithm to the same batch of data. As shown in Fig. 2, the EM gradient algorithm with asynchronous diffusion setting and local standard EM without cooperation are tested. The probabilities for Bernoulli model are set as $q_m = p_m = 0.8$. The estimates of both mean and variance are very noisy in Fig 2(a) for each sensor node with standard EM algorithm only based on the local data, while the estimation of both mean and variance with the proposed algorithm are much smooth for each sensor node, even under the imperfect communication condition.

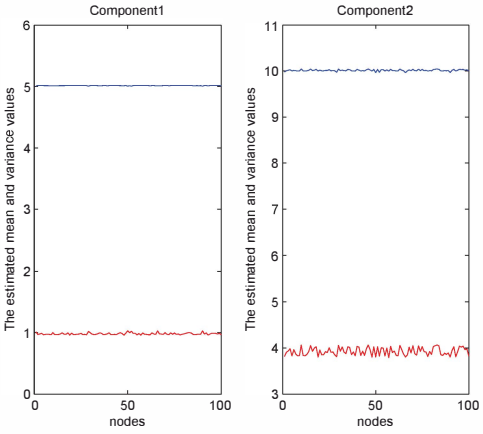
In Fig. 3, the mean-square-deviation (MSD) is used and evaluated for performance of different algorithms, which is defined as:

$$\text{MSD}_{\boldsymbol{\theta}} = E\left[\|\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}\|_2^2\right] \tag{27}$$

where $\|\cdot\|_2$ is the Euclidean norm. We selected the value of probabilities with two different cases, $p_m = q_m = 0.8$ and $p_m = q_m = 1$ (corresponds to traditional synchronous diffusion). Both adaptive diffusion algorithms provide improved mean-square-deviation in simulation



(a) Local standard EM algorithm without cooperation for 100 nodes



(b) Diffusion-based EM gradient algorithm for 100 nodes

Fig. 2. Estimated mean and variance with different schemes

compare to DDEM. The proposed asynchronous diffusion algorithm converge to almost as the same rate as the synchronous version. However, due to the additional randomness over the adaption process, EM gradient method with asynchronous diffusion suffer a slight degradation in MSD performance.

VI. Conclusion

In this work, we proposed a diffusion based EM gradient algorithm for mixture models in sensor networks. The proposed algorithm is characterized by local gradient based processing and computations of component parameters with asynchronous diffusion strategies . The ability to process data locally is of particular interest to sensor networks with computationally powerful nodes and requires costly node-to-node communications. More importantly, with asynchronous diffusion model, each node are allowed flexibility through their own assessment

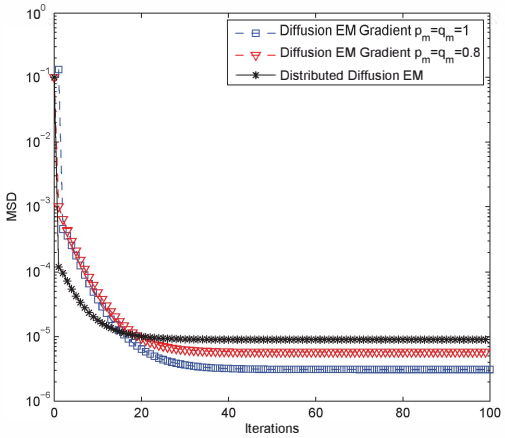


Fig. 3. Comparison of network MSD vs iteration index for asynchronous diffusion, synchronous diffusion and DDEM [5]

of local information without coordinated behavior over the network in comparison with synchronous strategies. Simulation results show the proposed algorithm outperforms local-standard EM without cooperation and degrades the MSD performance in compare to synchronous diffusion scheme. Theoretical analysis of convergence of the proposed algorithm will be addressed in future publications.

References

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

[2] R. Nowak, Distributed EM algorithms for density estimation and clustering in sensor networks, in IEEE Trans. on Signal Processing, vol. 51, no. 8, pp. 2245-2253, 2003.

[3] J. Yu and Pei-Jung Chung, Distributed componentwise EM algorithm or mixture models in sensor networks, in IEEE Global Telecommunications Conf., pp. 3418-3422, 2013

[4] D. Gu, Distributed EM algorithm for Gaussian mixtures in sensor networks, IEEE Trans. on Neural Networks, vol. 19, no. 7, pp. 1154-1166, 2008.

[5] Y. Weng, L. Xie, and W. Xiao, Diffusion scheme of distributed EM algorithm for Gaussian mixtures over random networks, in Proc. of the IEEE ICCA, pp. 1529-1534, 2009.

[6] Z. J. Towfic, J. Chen and A. Sayed, Collaborative learning of mixture models using diffusion adaptation, IEEE MLSP, pp. 1-6, Sep. 2011.

[7] S. S. Pereira, S. Barbarossa, and A. Pages-Zamora, Consensus for distributed EM-based clustering in WSNs, in Proc. of the 6th IEEE SAM, pp. 45-48, 2010.

[8] Lange, Kenneth.A gradient algorithm locally equivalent to the EM algorithm. Journal of the Royal Statistical Society. Series B (Methodological), pp. 425-437, 1995.

[9] X. Zhao and A. H. Sayed, Asynchronous adaptation and learning over networks-Part I: Modeling and stability analysis, IEEE Trans. Signal Process., vol. 63, no. 4, pp. 811-826, Feb. 2015.

[10] L. Xu and M. I. Jordan, On convergence properties of the EM algorithm for Gaussian mixtures, Neural Comput., vol. 8, pp. 129C151, 1996

[11] G. McLachlan and D. Peel, *Finite Mixture Models,* Wiley Series in Probability and Statistics, Oct. 2000.

Distributed Component-Wise EM Algorithm for Mixture Models in Sensor Networks

Jia Yu,*[1] and John Thompson[2]

[1]Institute for Digital Communications, School of Engineering, University of Edinburgh, Edinburgh EH9 3JL, UK
[2]Institute for Digital Communications, School of Engineering, University of Edinburgh, Edinburgh EH9 3JL, UK
*corresponding:j.yu@ed.ac.uk

Abstract: This paper considers mixtures model estimation for sensor networks in a distributed manner. In the statistical literature, the maximum likelihood (ML) estimate of mixture distributions can be computed via a straightforward implementation of the expectation and maximization (EM) algorithm. In the sensor networks without centralized processing units, data are collected and processed locally. Modifications of standard EM-type algorithm are necessary to accommodate the characteristics of sensor networks. Existing works on the distributed EM algorithm mainly focus on estimation performance and implementation aspects. In this paper, we address the convergence issue by proposing a distributed EM-like algorithm that updates mixture parameters sequentially. Simulation results show that the proposed method leads to significant gain in convergence speed and considerable saving in computational time.

1. Introduction

Sensor networks are composed of enormous small devices with limited measuring, processing, and communication abilities. There have been a variety of environmental monitoring applications, e.g. the temperature sensing, automobile tracking, and cooperative information processing [1, 2]. As a powerful probabilistic modeling tool, Gaussian Mixture Model (GMM) can be used for modeling density function in multiple applications, such as machine learning, pattern recognition and so on. It is an important step to estimate density in exploratory data analysis. For this purpose, the expectation-maximization (EM) method has been widely used [3].

The EM approach is well known to give ML approximations [4]. An expectation step (E-step) is performed in the EM method, and the likelihood expectation is calculated with observed latent variables included. While the maximization step (M-step) is to maximize the expected likelihood to obtain the estimates of ML parameters, this process is repeated a number of times until convergence at a local maximum. However, most EM algorithms are designed in a centralized way for sensor networks. Unlike the centralized strategy which processes all the information with a central node, the distributed estimation behaves differently and thus mitigates the computational load. Furthermore,the distributed estimation method is more robust against link failure [5].

The strategies of cooperation among nodes have significant impact on sensor networks within a distributed processing framework. The incremental and consensus strategies are widely used for distributed processing. The consensus strategy is discussed in [6, 8] which employs a slow

time scale for sampling and a fast time scale for iterative operations. This strategy aims to derive the consistent estimates for all nodes. A distributed EM method for Gaussian mixtures using the consensus strategy is presented in [10], in which a consensus filter is introduced between the E- and M- steps. As the resources are constrained for sensor networkss communications [7], the application is limited for consensus-based methods with two time scales. Especially for a large scale sensor networks, massive computational burden will be brought in to achieve the consensus among the network nodes.

For the incremental strategy, the data flows in a pre-specific direction from one node to another node, which leads to the loop-type cooperations between nodes with minimum power and communications. In [3], this model was successfully applied to describe the data measured by sensor networks in an inhomogeneous environment. Therein, a distributed (EM)-type algorithm was derived to identify Gaussian components common to the whole network and mixing probabilities associated with each node. Methods for improving the performance of distributed EM algorithm were suggested in [12, 10, 17].

In addition, the most documented problem associated with EM is its possibility of slow convergence. To speed up its convergence, various approaches have been proposed in the statistical literature [9, 15]. In [13], a component-wise EM algorithm was applied to mixture models. Instead of computing all parameters simultaneously in the M-step, the component-wise EM updates the component parameters sequentially. As the numerical results shown in [13], a better convergence rate can be achieved by this flexible approach. Another advantage of the component-wise EM is that despite relaxation of the constraint on mixing probabilities, it can be shown that when the algorithm converges, the sum of mixing probabilities equals to one.

To facilitate the application of the component-wise EM to sensor networks, we adopt the idea of incremental EM [14, 16] to present a distributed component-wise EM algorithm (DCEM) for sensor networks. Note that such incremental strategies may not be suitable for large scale networks. Therefore, we assume a small enough network, typically less than 100 sensor nodes. As illustrated in the following sections, given sufficient statistics from the previous node, the E- and M-step at the current node involve only local observations. Simulation results show that the proposed algorithm achieves a higher convergence rate than the distributed EM [3], leading to significant saving of overall computational time.

This paper is organized as follows. The problem and data models is defined in Section 2. Section 3 includes a brief description of the standard EM and distributed EM algorithms. The distributed component-wise EM algorithm for sensor networks is developed in Section 4. Section 5 presents an analysis of the convergence rate of the DCEM algorithm, Section 6 discusses simulation results and shows the performance of the proposed algorithm. Concluding remarks is given in Section 7.

2. Problem Formulation

Consider a sensor network consisting of M sensor nodes. The mth node records N_m independent and identically distributed data $\boldsymbol{y}_m = \{\boldsymbol{y}_{m,1}, \cdots, \boldsymbol{y}_{m,N_m}\}$. The measurements are assumed to obey a Gaussian mixture distribution

$$\boldsymbol{y}_{m,i} \sim \sum_{j=1}^{J} \alpha_{m,j} \mathcal{P}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j),\ i = 1, \cdots, N_m \tag{1}$$

where $\mathcal{P}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes the Gaussian density function with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. The mixing parameters $\boldsymbol{\alpha}_j = \{\alpha_{m,j}\}_{m=1}^{M}$ are potentially unique at each node, but the J mixing components $\mathcal{P}\left(\boldsymbol{\mu}_j, \boldsymbol{\Sigma_j}\right)$ are common to all nodes. Set $\boldsymbol{\theta}_j = \{\boldsymbol{\alpha}_j, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\}_{j=1}^{J}$, then the unknown parameter set is given by $\boldsymbol{\theta} = \{\boldsymbol{\theta}_j\}_{j=1}^{J}$. Based on the measurements $\boldsymbol{y} = \{\boldsymbol{y}_m\}_{m=1}^{M}$, the task is to compute the maximum likelihood (ML) estimate for $\boldsymbol{\theta}$ in a distributed manner.

It is well known that maximization of the log-likelihood for the mixture model (1)

$$L(\boldsymbol{\theta}) = \sum_{m=1}^{M} \sum_{i=1}^{N_m} \log \left(\sum_{j=1}^{J} \alpha_{m,j} \mathcal{P}(\boldsymbol{y}_{m,i} | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \right) \tag{2}$$

is greatly simplified by the EM-type algorithms [14] which will be described in the following section. This data model is assumed to be statistically independent for each node. However, if the data are (spatially or temporally) correlated, this model is still valid by interpreting it as a $pseudolikelihood$ [18].

3. Standard EM and Distributed EM Algorithms (DEM)

The formulation of the mixture problem in the EM framework is achieved by augmenting the observed data vector $\boldsymbol{y} = \{\boldsymbol{y}_m\}_{m=1}^{M}$ with the associated component-label vectors $\boldsymbol{z} = \{\boldsymbol{z}_m\}_{m=1}^{M}$ where $\boldsymbol{z}_m = \{z_{m,i}\}_{i=1}^{N_m}$. Each $z_{m,i}$ takes on a value from the set $\{1, \ldots, J\}$, where $z_{m,i} = j$ indicates that $\boldsymbol{y}_{m,i}$ is generated by the jth mixture component

$$\boldsymbol{y}_{m,i} \sim \mathcal{P}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j). \tag{3}$$

The complete data log-likelihood $L_c(\boldsymbol{\theta})$ is then given by

$$\begin{aligned} L_c(\boldsymbol{\theta}) &= \log p(\boldsymbol{y}, \boldsymbol{z} | \boldsymbol{\theta}) \\ &= \sum_{m=1}^{M} \sum_{i=1}^{N_m} \sum_{j=1}^{J} z_{m,i,j} (\log \alpha_{m,j} + \log \mathcal{P}(\boldsymbol{y}_{m,i} | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)) \end{aligned} \tag{4}$$

where $p(\boldsymbol{y}, \boldsymbol{z} | \boldsymbol{\theta})$ denotes the joint density of y and z with parameter $\boldsymbol{\theta}$. Starting from an initial estimate $\boldsymbol{\theta}^0$, the standard EM algorithm iterates between the E (expectation) and M (maximization) steps. In the E-step, given the current estimate $\boldsymbol{\theta}^t$, the conditional expectation of the complete data log-likelihood is computed as follows

$$\begin{aligned} Q(\boldsymbol{\theta}; \boldsymbol{\theta}^t) &= E[L_c(\boldsymbol{\theta}) | \boldsymbol{y}, \boldsymbol{\theta}^t] \\ &= \sum_{m=1}^{M} \sum_{i=1}^{N_m} \sum_{j=1}^{J} w_{m,i,j}^{t+1} (\log \alpha_{m,j} + \log \mathcal{P}(\boldsymbol{y}_{m,i} | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)), \end{aligned} \tag{5}$$

where

$$w_{m,i,j}^{t+1} = \frac{\alpha_{m,j}^t \mathcal{P}(\boldsymbol{y}_{m,i} | \boldsymbol{\mu}_j^t, \boldsymbol{\Sigma}_j^t)}{\sum_{j=1}^{J} \alpha_{m,j}^t \mathcal{P}(\boldsymbol{y}_{m,i} | \boldsymbol{\mu}_j^t, \boldsymbol{\Sigma}_j^t)} \tag{6}$$

is the posterior probability that the ith sample at node m belongs to the jth component given the observed value $\boldsymbol{y}_{m,i}$.

In the M-step, the parameters are computed by maximizing the complete data log-likelihood function (5)

$$\boldsymbol{\theta}^{t+1} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}; \boldsymbol{\theta}^t), \tag{7}$$

leading to the following update formula for $j = 1, \cdots, J$.

$$\alpha_{m,j}^{t+1} = \frac{1}{N_m} \sum_{i=1}^{N_m} w_{m,i,j}^{t+1}, \quad m = 1, \cdots, M \tag{8}$$

$$\boldsymbol{\mu}_j^{t+1} = \frac{\sum_{m=1}^{M} \sum_{i=1}^{N_m} w_{m,i,j}^{t+1} \boldsymbol{y}_{m,i}}{\sum_{m=1}^{M} \sum_{i=1}^{N_m} w_{m,i,j}^{t+1}}, \tag{9}$$

$$\boldsymbol{\Sigma}_j^{t+1} = \frac{\sum_{m=1}^{M} \sum_{i=1}^{N_m} w_{m,i,j}^{t+1} (\boldsymbol{y}_{m,i} - \boldsymbol{\mu}_j^{t+1})(\boldsymbol{y}_{m,i} - \boldsymbol{\mu}_j^{t+1})^T}{\sum_{m=1}^{M} \sum_{i=1}^{N_m} w_{m,i,j}^{t+1}} \tag{10}$$

Thus, the global summary quantities are

$$w_j^{t+1} = \sum_{m=1}^{M} \sum_{i=1}^{N_m} w_{m,i,j}^{t+1}, \tag{11}$$

$$\boldsymbol{a}_j^{t+1} = \sum_{m=1}^{M} \sum_{i=1}^{N_m} w_{m,i,j}^{t+1} \boldsymbol{y}_{m,i}, \tag{12}$$

$$\boldsymbol{b}_j^{t+1} = \sum_{m=1}^{M} \sum_{i=1}^{N_m} w_{m,i,j}^{t+1} \boldsymbol{y}_{m,i} \boldsymbol{y}_{m,i}^T. \tag{13}$$

in which the local summary quantities are denoted as

$$w_{m,j}^{t+1} = \sum_{i=1}^{N_m} w_{m,i,j}^{t+1}, \tag{14}$$

$$\boldsymbol{a}_{m,j}^{t+1} = \sum_{i=1}^{N_m} w_{m,i,j}^{t+1} \boldsymbol{y}_{m,i}, \tag{15}$$

$$\boldsymbol{b}_{m,j}^{t+1} = \sum_{i=1}^{N_m} w_{m,i,j}^{t+1} \boldsymbol{y}_{m,i} \boldsymbol{y}_{m,i}^T. \tag{16}$$

Notice that with these summaries defined as previous, the estimated parameters are

$$\boldsymbol{\mu}_j^{t+1} = \frac{\boldsymbol{a}_j^{t+1}}{w_j^{t+1}}, \tag{17}$$

$$\boldsymbol{\Sigma}_j^{t+1} = \frac{\boldsymbol{b}_j^{t+1}}{w_j^{t+1}} - \boldsymbol{\mu}_j^{t+1} (\boldsymbol{\mu}_j^{t+1})^T, \tag{18}$$

The E- and M-steps are alternated repeatedly until the difference between likelihoods of consecutive iterates $L(\boldsymbol{\theta}^{t+1}) - L(\boldsymbol{\theta}^t)$ is less than a pre-specified small number ϵ.

A distributed EM algorithm based on the incremental strategy for sensor network (DEM) was studied in [3]. With such a network setting, the communication path is cyclic and pre-set. Only one node updates the parameter set $\boldsymbol{\theta}^{t+1}$ using its own N_m observations at each iteration, given the current parameter set $\boldsymbol{\theta}^t$. In details, node m can update the global summary quantities by using its new local summary quantities to replace the old quantities based on

$$w_j^{t+1} = w_j^t + w_{m,j}^{t+1} - w_{m,j}^t \,, \tag{19}$$

$$\boldsymbol{a}_j^{t+1} = \boldsymbol{a}_j^t + \boldsymbol{a}_{m,j}^{t+1} - \boldsymbol{a}_{m,j}^t \,, \tag{20}$$

$$\boldsymbol{b}_j^{t+1} = \boldsymbol{b}_j^t + \boldsymbol{b}_{m,j}^{t+1} - \boldsymbol{b}_{m,j}^t \,. \tag{21}$$

and update the parameter set $\boldsymbol{\theta}^{t+1}$ according to (17) and (18). During this procedure, other nodes are fixed. Then, node m passes the message of updated global summary quantities $\{w_j^{t+1}, \boldsymbol{a}_j^{t+1}, \boldsymbol{b}_j^{t+1}\}$ and the estimated parameter $\boldsymbol{\theta}^{t+1}$ to next adjacent $(m+1)$ node, and this process is repeatedly implemented.

Note that each node only executes a single and local E- and M- step in DEM algorithm, thus this algorithm do not require the updated means and covariances $\{\boldsymbol{\mu}_j^{t+1}, \boldsymbol{\Sigma}_j^{t+1}\}$ to reach a fixed point at each local E-step process. In order to speed up the overall convergence, DEMM algorithm refers to DEM with multiple steps at each node was also studied in [3]. Specifically, the local E- and M-steps can be repeated several times in succession until the maximization of the local log-likelihood function is found, then the updated message can be passed to the next node.

All these algorithms require to execute the standard E- and M- step to update parameters simultaneously. They are often effective when the mixtures are well-separated, but suffered a slow convergence when the mixtures become complex or overlapping. To speed up the convergence of the standard EM algorithm, a component-wise EM method for mixture models (CEMM) was presented in [13]. Rather than computing all parameters simultaneously, the CEMM algorithm considers the decomposition of the parameter vector $\boldsymbol{\theta}$ into component parameter vectors $\{\boldsymbol{\alpha}_j, \boldsymbol{\theta}_j\}$, $j = 1, \cdots, J$ and updates only one component at a time. Specifically, each iteration consists of J cycles and the conditional expectation (5) is computed and the parameter vector associated with jth component is updated at each cycle. As pointed out in [13], the decoupling of parameter updates implies the use of the smallest admissible missing data space and leads to faster convergence than the standard EM algorithm.

4. Distributed Component-wise EM Algorithm

Motivated by the superior convergence behavior of the component-wise EM algorithm, we propose a distributed component-wise EM algorithm for mixtures in sensor networks. In the following, we consider the incremental strategy for information exchange between sensor networks as depicted in Fig. 1, which exploits the idea of incremental EM [16] to facilitate local processing. By applying the incremental EM, the component-wise EM can be implemented so that at node m, given the summary quantities (11), (12) and (13) from the previous node $(m-1)$, only local data $\boldsymbol{y}_m$ is involved.

Let $\boldsymbol{a}_j^t, \boldsymbol{b}_j^t, w_j^t$ be the received summary statistics of the mth node from the previous one, and the local estimates after the tth iteration be

$$\boldsymbol{\theta}_m^t = \{\boldsymbol{\theta}_{m,1}^t, \cdots, \boldsymbol{\theta}_{m,J}^t\}, \tag{22}$$

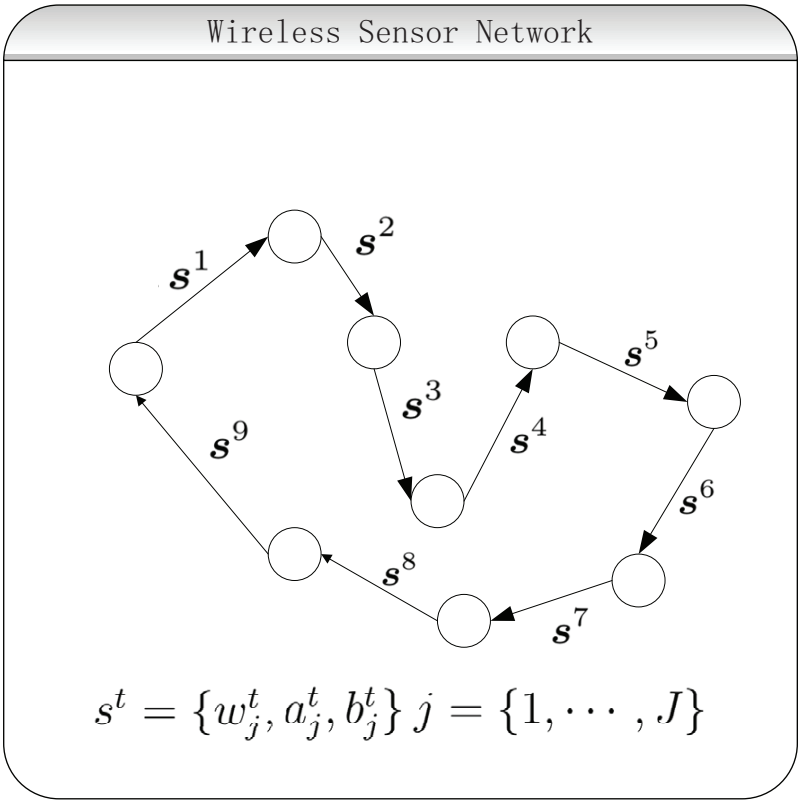


Fig. 1. *Communication/iteration cycle in a sensor network*

where $\boldsymbol{\theta}_{m,j}^t$ include the estimate for the jth component $\{\alpha_{m,j}^t, \boldsymbol{\mu}_{m,j}^t, \boldsymbol{\Sigma}_{m,j}^t\}$. At the beginning of the $(t+1)$th iteration, the initial estimates for the mean and covariance matrix are obtained from the summary statistics as follows:

$$\boldsymbol{\mu}_j^t = \frac{\boldsymbol{a}_j^t}{w_j^t}, \quad \boldsymbol{\Sigma}_j^t = \frac{\boldsymbol{b}_j^t}{w_j^t} - \boldsymbol{\mu}_j^t \boldsymbol{\mu}_j^{t'}, \quad j = 1, \cdots, J. \tag{23}$$

let $\boldsymbol{\theta}_m^{[t+1,0]} = \boldsymbol{\theta}_m^t$, the parameters associated with the jth components $\boldsymbol{\theta}_{m,j}^t$ are updated sequentially in the proposed algorithm as follows.
For $j = 1, \cdots, J$, the E-step is computed as:

$$w_{m,i,j}^{t+1} = \frac{\alpha_{m,j}^t \mathcal{P}(\boldsymbol{y}_{m,i} | \boldsymbol{\mu}_j^t, \boldsymbol{\Sigma}_j^t)}{\sum_{k=1}^{j-1} \alpha_{m,k}^{t+1} \mathcal{P}(\boldsymbol{y}_{m,i} | \boldsymbol{\mu}_{m,k}^{t+1}, \boldsymbol{\Sigma}_{m,k}^{t+1}) + \sum_{k=j}^{J} \alpha_{m,k}^t \mathcal{P}(\boldsymbol{y}_{m,i} | \boldsymbol{\mu}_{m,k}^t, \boldsymbol{\Sigma}_{m,k}^t)}. \tag{24}$$

The M-step is then

$$\alpha_{m,j}^{t+1} = \frac{1}{N_m}\sum_{i=1}^{N_m} w_{m,i,j}^{t+1} , \tag{25}$$

$$\boldsymbol{\mu}_{m,j}^{t+1} = \frac{\boldsymbol{a}_{m,j}^{t+1}}{w_{m,j}^{t+1}} , \tag{26}$$

$$\boldsymbol{\Sigma}_{m,j}^{t+1} = \frac{\boldsymbol{b}_{m,j}^{t+1}}{w_{m,j}^{t+1}} - \boldsymbol{\mu}_{m,j}^{t+1}\boldsymbol{\mu}_{m,j}^{t+1'} , \tag{27}$$

where the local summary statistics $w_{m,j}, \boldsymbol{a}_{m,j}, \boldsymbol{b}_{m,j}$ are

$$w_{m,j}^{t+1} = \frac{1}{N_m}\sum_{i=1}^{N_m} w_{m,i,j}^{t+1} , \tag{28}$$

$$\boldsymbol{a}_{m,j}^{t+1} = \sum_{i=1}^{N_m} w_{m,i,j}^{t+1}\boldsymbol{y}_{m,i} , \tag{29}$$

$$\boldsymbol{b}_{m,j}^{t+1} = \sum_{i=1}^{N_m} w_{m,i,j}^{t+1}\boldsymbol{y}_{m,i}\boldsymbol{y}_{m,i}' . \tag{30}$$

The estimate at the jth cycle is given by

$$\boldsymbol{\theta}_m^{[t+1,j]} = \{\boldsymbol{\theta}_{m,1}^{t+1}, \cdots, \boldsymbol{\theta}_{m,j}^{t+1}, \boldsymbol{\theta}_{m,j+1}^{t}, \cdots, \boldsymbol{\theta}_{m,J}^{t}\}. \tag{31}$$

After J cycles, the output of the $(t+1)$th iteration is given by:

$$\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}_m^{[t+1,J]}. \tag{32}$$

Then the local summary statistics are computed with the new estimate $\boldsymbol{\theta}^{t+1}$ according to (19)-(21). Note that the old values of summary statistics are replaced by updated values at node m. In addition, the computations of the posterior probabilities (24) and the estimates (25), (26) and (27) involve only the data at node m.

The major difference of the proposed component-wise approach from the distributed EM algorithm is as follows. In the distributed EM algorithm (DEM) [3], the parameters associated with all components are updated simultaneously. The E-step is evaluated only once at the beginning of the iteration. In the proposed algorithm, each component parameter set $\boldsymbol{\theta}_j$ is computed sequentially and the posterior probability $w_{m,i,j}$ (24) is evaluated at each cycle. The computational time is only slightly increased by the multiple E-steps in comparison to the distributed EM algorithm. Simulation results in the following sections will show that our approach leads to a much faster convergence of the log-likelihood than the distributed EM algorithm.

5. Convergence Analysis

In [19] and [20], the authors gave in-depth analysis on the convergence of standard EM algorithms and . It is shown in [16] that under standard regularity conditions, the incremental EM will give the

estimates which converge with respect to the likelihood function, and the likelihood is iteratively ascending. In [3], it is assumed that the $\{\boldsymbol{\theta}^t\}$ converges to $\boldsymbol{\theta}^*$ to maximize the log-likelihood $L(\boldsymbol{\theta})$. It can be shown that the estimate $\boldsymbol{\theta}^t$ near $\boldsymbol{\theta}^*$ with iterations has the following approximate relationship for sufficiently large t

$$\boldsymbol{\theta}^{t+1} - \boldsymbol{\theta}^* = \mathbf{M}(\bar{\boldsymbol{\theta}}^t - \boldsymbol{\theta}^*) \tag{33}$$

where $\bar{\boldsymbol{\theta}}^t$ is described as a certain average of the past $\{\boldsymbol{\theta}^{(t-m)}\}_{m=1}^{M}$ and $\mathbf{M}$ is defined as the rate matrix of the algorithm. The convergence rate is determined by the spectral radius $\rho(\mathbf{M})$ of the rate matrix [22]. Based on the results of [15], a larger $\rho(\mathbf{M})$ leads to a slower convergence speed.

Before analyzing the convergence of DCEM, we consider another analytical approach for the convergence of the DEM in [21]. In this method, we define an augmented vector including all nodes' parameters as:

$$\boldsymbol{\Theta}^t = \begin{bmatrix} \boldsymbol{\theta}_1^t \\ \vdots \\ \boldsymbol{\theta}_M^t \end{bmatrix} \tag{34}$$

During each iteration of the DEM algorithm, only one node updates its parameters while other nodes' parameters are fixed, all parameters of $\boldsymbol{\Theta}^t$ can be updated after a full cycle of the procedure. In addition, assuming that data sets are statistically independent at different nodes, the local objective function is calculated as:

$$L_m(\boldsymbol{\theta}) = \sum_{i=1}^{N_m} \log(\sum_{j=1}^{J} \alpha_{m,j} \mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)) \tag{35}$$

where $\boldsymbol{\theta}_m$ is the parameter vector for the mth node. We model the conditional expectation of complete data using log-likelihood as:

$$\begin{aligned} Q_m(\boldsymbol{\theta}; \boldsymbol{\theta}_m^t) &= E[L_c(\boldsymbol{\theta})|\boldsymbol{y}_m, \boldsymbol{\theta}_m^t] \\ &= \sum_{i=1}^{N_m} \sum_{j=1}^{J} w_{m,i,j}^{t+1} (\log \alpha_{m,j} + \log \mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)) \end{aligned} \tag{36}$$

The total conditional Q function can be reformatted as [3]:

$$\begin{aligned} Q(\boldsymbol{\theta}; \boldsymbol{\theta}^t) &= Q(\boldsymbol{\theta}; \boldsymbol{\theta}_1^t, \ldots, \boldsymbol{\theta}_M^t) \\ &= \sum_{m=1}^{M} Q_m(\boldsymbol{\theta}; \boldsymbol{\theta}_m^t) \end{aligned} \tag{37}$$

Finally, the updated equation of the DEM algorithm for a sensor networks can be represented as:

$$\boldsymbol{\theta}^{t+1} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}; \boldsymbol{\theta}_1^t, \ldots, \boldsymbol{\theta}_M^t), \tag{38}$$

Using the Taylor expansion in the local Q function, it was verified in [21] that the local estimates $\boldsymbol{\theta}_m^t$ can achieve a local maximum at a fixed point $\boldsymbol{\theta}_m^*$ and satisfy the following approximate relationship for sufficiently large t

$$\boldsymbol{\theta}_m^{t+1} - \boldsymbol{\theta}_m^* = \mathbf{M}_m^{DEM}(\boldsymbol{\theta}_m^t - \boldsymbol{\theta}_m^*) \tag{39}$$

where $\mathbf{M}_m^{DEM}$ is the local rate matrix at node m and its expression is given by

$$\begin{aligned}\mathbf{M}_m^{DEM} &= \nabla^{11}Q_m(\boldsymbol{\theta}_m^*;\boldsymbol{\theta}_m^*)[\nabla^{20}Q_m(\boldsymbol{\theta}_m^*;\boldsymbol{\theta}_m^*)]^{-1} \\ &= \mathbf{I} - [\nabla^{20}D(\boldsymbol{\theta}_m^*;\boldsymbol{\theta}_m^*) + \nabla^2 L(\boldsymbol{\theta}_m^*)]^{-1}\nabla^2 L(\boldsymbol{\theta}_m^*)\end{aligned} \tag{40}$$

where ∇^{ij} denotes the ith order partial derivatives with respect to the first argument and jth order partial derivatives with respect to the second argument. $D(\boldsymbol{\theta}_m;\boldsymbol{\theta}_m^t) = E[\log p(y,z|y,\boldsymbol{\theta})|y,\boldsymbol{\theta}^t]$ is the distance between $\boldsymbol{\theta}_m$ and $\boldsymbol{\theta}_m^t$. It can be shown that $\nabla^2 L(\boldsymbol{\theta}_m^*)$ and $D(\boldsymbol{\theta}_m;\boldsymbol{\theta}_m^t)$ are negative definite [4] and the eigenvalues of $\mathbf{M}_m^{DEM}$ all lie in $[0,1)$. With the definition (34), if $\boldsymbol{\Theta}^*$ is a fixed point of the DEM algorithm , the convergence rate of the full DEM procedure in sensor network setting can be formulated as:

$$\boldsymbol{\Theta}^{t+1} - \boldsymbol{\Theta}^* = \mathbf{M}^{DEM}(\boldsymbol{\Theta}^t - \boldsymbol{\Theta}^*) \tag{41}$$

where $\mathbf{M}^{DEM}$ is a block diagonal matrix defined as

$$\mathbf{M}^{DEM} = \begin{bmatrix} \mathbf{M}_1^{DEM} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \mathbf{M}_M^{DEM} \end{bmatrix} \tag{42}$$

Based on the definition of the spectrum radius:

$$\rho(\mathbf{M}) = \max|\boldsymbol{\beta}| \tag{43}$$

where $\boldsymbol{\beta}$ are the eigenvalues of $\mathbf{M}$, the convergence rate is the largest eigenvalue of $\mathbf{M}$. Therefore, if the maximum eigenvalue of $\mathbf{M}_m^{DEM}$ is denoted by $\boldsymbol{\beta}_m^{DEM}$, the convergence rate of all estimated parameters in the DEM algorithm after a full cycle will be equal to

$$\rho(\mathbf{M}^{DEM}) = \max_m |\boldsymbol{\beta}_m^{DEM}| < 1 \tag{44}$$

Now we consider the DCEM algorithm in a sensor network situation. In a DEM algorithm, the linear constraint for Gaussian mixtures at each node operation $\sum_{j=1}^{J}\alpha_{m,j} = 1$ is automatically satisfied during every E- and M- step. This is obviously not satisfactory in the context of component-wise methods [13]. In [13], a Lagrangian approach is introduced to fulfill this constraint by reconstructing a modified likelihood function based on Lagrangian duality. Since the data collected at each sensor are independent of the data at other sensors, the local modified likelihood function is given by:

$$\mathcal{L}_m(\boldsymbol{\theta},\lambda) = L_m(\boldsymbol{\theta}) - \lambda\Big(\sum_{j=1}^{J}\alpha_{m,j} - 1\Big) \tag{45}$$

From [13], we can get $\lambda = N_m$ by solving this Lagrangian function, thus, (45) becomes

$$\mathcal{L}_m(\boldsymbol{\theta}) = L_m(\boldsymbol{\theta}) - N_m\Big(\sum_{j=1}^{J}\alpha_{m,j} - 1\Big) \tag{46}$$

The convergence of the standard algorithm with Gaussian mixtures is investigated in [19] by linking the EM algorithm to gradient ascent methods. Motivated by this idea, we demonstrate that the E- and M- steps of the DCEM algorithm at each node can be realized by jointly using the gradient and the projection matrices.

Theorem 1. At the jth cycle, the local updates formulas (25)-(27) in DCEM at node m can be described as:

$$\alpha_{m,j}^{t+1} - \alpha_{m,j}^{t} = P_{\alpha_{m,j}^{t}} \frac{\partial \mathcal{L}_m(\boldsymbol{\theta})}{\partial \alpha_{m,j}}|_{\mathcal{A}_m = \mathcal{A}_m^{[t+1,j-1]}}, \tag{47}$$

$$\boldsymbol{\mu}_{m,j}^{t+1} - \boldsymbol{\mu}_{m,j}^{t} = P_{\boldsymbol{\mu}_{m,j}^{t}} \frac{\partial \mathcal{L}_m(\boldsymbol{\theta})}{\partial \boldsymbol{\mu}_{m,j}}|_{\boldsymbol{\mu}_{m,j} = \boldsymbol{\mu}_{m,j}^{t}}, \tag{48}$$

$$\text{vec}[\boldsymbol{\Sigma}_{m,j}^{t+1}] - \text{vec}[\boldsymbol{\Sigma}_{m,j}^{t}] = P_{\boldsymbol{\Sigma}_{m,j}^{t}} \frac{\partial \mathcal{L}_m(\boldsymbol{\theta})}{\partial \text{vec}[\boldsymbol{\Sigma}_{m,j}]}|_{\boldsymbol{\Sigma}_{m,j} = \boldsymbol{\Sigma}_{m,j}^{t}}, \tag{49}$$

where vec$[\boldsymbol{C}]$ denotes the vector obtained by stacking the column vectors of matrix $\boldsymbol{C}$, $\mathcal{A}_m$ denotes the vector of mixing probabilities $[\alpha_{m,1}, \cdots, \alpha_{m,J}]^T$ at node m,

$$\mathcal{A}_m^{[t+1,j-1]} = [\alpha_{m,1}^{t+1}, \cdots, \alpha_{m,j-1}^{t+1}, \alpha_{m,j}^{t}, \cdots, \alpha_{m,J}^{t}]^T, \tag{50}$$

and

$$P_{\alpha_{m,j}^{t}} = \frac{1}{N_m} \alpha_{m,j}^{t} \tag{51}$$

$$P_{\boldsymbol{\mu}_{m,j}^{t}} = \frac{\boldsymbol{\Sigma}_{m,j}^{t}}{\sum_{i=1}^{N_m} w_{i,m,j}^{t+1}}, \tag{52}$$

$$P_{\boldsymbol{\Sigma}_{m,j}^{t}} = \frac{2}{\sum_{i=1}^{N_m} w_{i,m,j}^{t+1}} \boldsymbol{\Sigma}_{m,j}^{t} \otimes \boldsymbol{\Sigma}_{m,j}^{t}, \tag{53}$$

where $\otimes$ is the Kronecker product.

Proof: See Appendix 8.1

Using the notation $\boldsymbol{\theta}_{m,j} = \{\alpha_{m,j}, \boldsymbol{\mu}_{m,j}, \text{vec}[\boldsymbol{\Sigma}_{m,j}]^T\}^T$, we define the local projection matrix as follow:

$$\mathbf{P}_{\boldsymbol{\theta}_{m,j}^{t}} = \begin{bmatrix} P_{\alpha_{m,j}^{t}} & & \mathbf{0} \\ & P_{\boldsymbol{\mu}_{j}^{t}} & \\ \mathbf{0} & & P_{\boldsymbol{\Sigma}_{j}^{t}} \end{bmatrix} \tag{54}$$

Then, the updates can be integrated into:

$$\boldsymbol{\theta}_{m,j}^{(t+1)} = \boldsymbol{\theta}_{m,j}^{t} + P_{\boldsymbol{\theta}_{m,j}}^{t} \frac{\partial \mathcal{L}_m(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_{m,j}}|_{\boldsymbol{\theta}_m = \boldsymbol{\theta}_m^{[t+1,j]}} \tag{55}$$

Consider the tth iteration at node m and let $\boldsymbol{\theta}_m = \{\boldsymbol{\theta}_{m,j} \ \boldsymbol{\theta}_{m,l}\}^T$ where $\boldsymbol{\theta}_{m,l}$ are the other parameters of $\boldsymbol{\theta}_m$ when $l \neq j$. We apply the Taylor formula with remainder [23] to expand this gradient at a fixed point $\boldsymbol{\theta}_m^*$. Since $\frac{\partial \mathcal{L}_m(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_{m,j}}|_{\boldsymbol{\theta}_m = \boldsymbol{\theta}_m^*} = 0$, we can obtain

$$\begin{aligned} \boldsymbol{\theta}_{m,j}^{t+1} - \boldsymbol{\theta}_{m,j}^{*} &= \boldsymbol{\theta}_{m,j}^{t} - \boldsymbol{\theta}_{m,j}^{*} + \mathbf{P}_{\boldsymbol{\theta}_{m,j}^{*}} \frac{\partial \mathcal{L}_m(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_{m,j} \partial \boldsymbol{\theta}_{m,j}}|_{\boldsymbol{\theta}_m = \boldsymbol{\theta}_m^*} (\boldsymbol{\theta}_j^t - \boldsymbol{\theta}_j^*) \\ &+ \mathbf{P}_{\boldsymbol{\theta}_{m,j}^{*}} \frac{\partial \mathcal{L}_m(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_{m,j} \partial \boldsymbol{\theta}_{m,l}}|_{\boldsymbol{\theta}_m = \boldsymbol{\theta}_m^*} (\boldsymbol{\theta}_{m,l}^t - \boldsymbol{\theta}_{m,l}^*) \end{aligned} \tag{56}$$

Define the local Hessian of modified function at the fixed point $\boldsymbol{\theta}_m^*$ by

$$\mathbf{H}_{\boldsymbol{\theta}_m^*} = -\frac{\partial \mathcal{L}_m(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_m \partial \boldsymbol{\theta}_m}|_{\boldsymbol{\theta}_m = \boldsymbol{\theta}_m^*} \tag{57}$$

and the following submatrices of Hessian

$$\begin{aligned} \mathbf{H}_{\boldsymbol{\theta}_{m,j}^*} &= -\frac{\partial \mathcal{L}_m(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_{m,j} \partial \boldsymbol{\theta}_{m,j}}|_{\boldsymbol{\theta}_m = \boldsymbol{\theta}_m^*} \\ \mathbf{H}_{\boldsymbol{\theta}_{m,l}^*} &= -\frac{\partial \mathcal{L}_m(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_{m,j} \partial \boldsymbol{\theta}_{m,l}}|_{\boldsymbol{\theta}_m = \boldsymbol{\theta}_m^*} \end{aligned} \tag{58}$$

where $\mathbf{H}_{\boldsymbol{\theta}_{m,j}}$ is the curvature of the modified log-likelihood function $\mathcal{L}_m(\boldsymbol{\theta}_m)$ with respect to $\boldsymbol{\theta}_{m,j}$, and $\mathbf{H}_{\boldsymbol{\theta}_{m,l}}$ is the coupling between $\boldsymbol{\theta}_{m,j}$ and $\boldsymbol{\theta}_{m,l}$. Let $\mathbf{R}_{\boldsymbol{\theta}_{m,j}}$ denote the $J \times J$ permutation matrix that reorders the elements of $\{\boldsymbol{\theta}_{m,j}, \boldsymbol{\theta}_{m,l}\}$ into $\{1, \cdots, J\}$, and $\mathbf{R}_{\boldsymbol{\theta}_{m,j}} \mathbf{R}_{\boldsymbol{\theta}_{m,j}}^T = \mathbf{I}$. Then, we define the $J \times J$ composite local rate matrix at j cycle for DCEM algorithm

$$\mathbf{M}_{m,j}^{DCEM} = \mathbf{R}_{\boldsymbol{\theta}_{m,j}} \begin{bmatrix} \mathbf{I} - \mathbf{P}_{\boldsymbol{\theta}_{m,j}^*} \mathbf{H}_{\boldsymbol{\theta}_{m,j}^*} & -\mathbf{P}_{\boldsymbol{\theta}_{m,j}^*} \mathbf{H}_{\boldsymbol{\theta}_{m,l}^*} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \mathbf{R}_{\boldsymbol{\theta}_{m,j}}^T \tag{59}$$

The components of $\boldsymbol{\theta}_{m,l}^t$ are just copied, so after permuting $\mathbf{R}_{\boldsymbol{\theta}_{m,j}}$

$$\boldsymbol{\theta}_m^{[t+1,j]} - \boldsymbol{\theta}_m^* = \mathbf{M}_{m,j}^{DCEM} (\boldsymbol{\theta}_m^{[t+1,j-1]} - \boldsymbol{\theta}_m^*) \tag{60}$$

A full cycle consists of one update over each of the J index sets, therefore, after J cycle, we can obtain:

$$\boldsymbol{\theta}_m^{[t+1,J]} - \boldsymbol{\theta}_m^* = \mathbf{M}_{m,J}^{DCEM} \times \cdots \mathbf{M}_{m,1}^{DCEM} (\boldsymbol{\theta}_m^{[t,J]} - \boldsymbol{\theta}_m^*) \tag{61}$$

Theorem 2. There exists $a < 1$ such that for any

$$\rho(\mathbf{M}_m^{DCEM}) = \|\mathbf{M}_{m,J}^{DCEM} \times \cdots \mathbf{M}_{m,1}^{DCEM}\|_{\mathbf{H}_{\boldsymbol{\theta}_m^*}} \leq a \tag{62}$$

where $\|\mathbf{M}\|_{\mathbf{N}} = \|\mathbf{N}^{1/2} \mathbf{M} \mathbf{N}^{-1/2}\|$ denotes the generalized matrix spectral norm with respect to a positive definite matrix $\mathbf{N}$.

Proof: See Appendix 8.2

After J cycles, the output of the $(t+1)$th iteration at node m is defined as:

$$\boldsymbol{\theta}_m^{t+1} = \boldsymbol{\theta}_m^{[t+1,J]} \tag{63}$$

By applying the same analytical approach of DEM algorithm to the DCEM algorithm, it is easy to obtain the similar result of convergence properties as

$$\boldsymbol{\Theta}^{t+1} - \boldsymbol{\Theta}^* = \mathbf{M}^{DCEM} (\boldsymbol{\Theta}^t - \boldsymbol{\Theta}^*) \tag{64}$$

where $\mathbf{M}^{DCEM}$ is a block diagonal matrix given by

$$\mathbf{M}^{DCEM} = \begin{bmatrix} \mathbf{M}_1^{DCEM} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \mathbf{M}_M^{DCEM} \end{bmatrix} \tag{65}$$

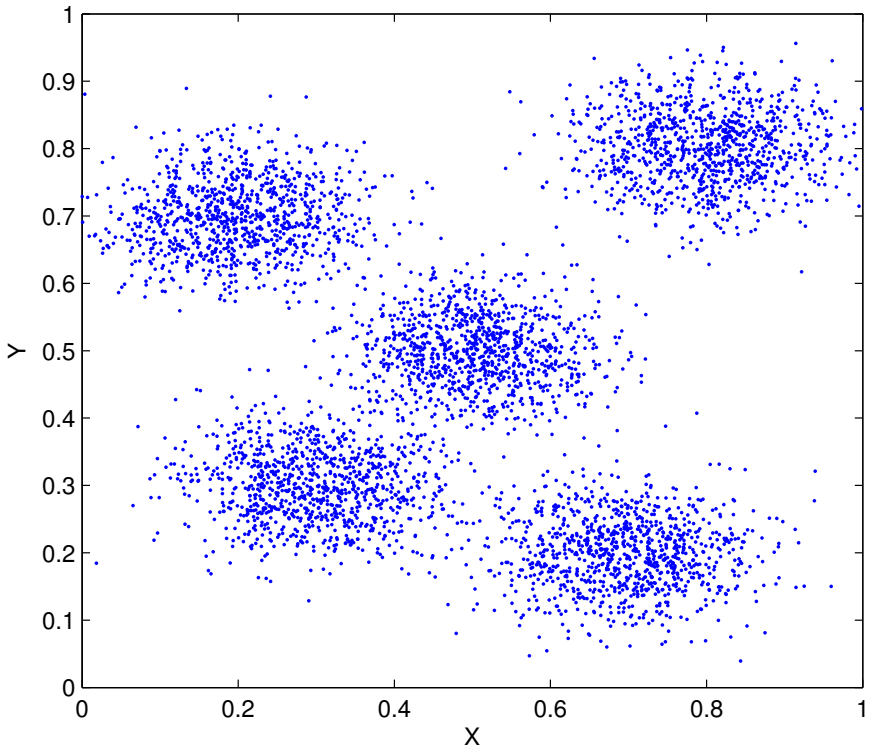


Fig. 2. *Data distribution for well-separated mixture case*

Given the analysis above, since $\boldsymbol{\theta}_m^*$ is a fixed point of node m, the eigenvalues corresponding to the mth diagonal block of $\mathbf{M}^{DCEM}$ should be in the interval $[0, 1)$. For a specific sensor node, the largest eigenvalue of the submatrix corresponds to the convergence rate of the parameters. The largest eigenvalue of the rate matrix $\mathbf{M}^{DCEM}$ is related to the convergence rate of all the network parameters after a full DCEM cycle. Denote the largest eigenvalue of $\mathbf{M}_m^{DCEM}$ as $\boldsymbol{\beta}_m^{DCEM}$, the convergence rate of DCEM for the whole network can be obtained as follow:

$$\rho(\mathbf{M}^{DCEM}) = \max_m |\boldsymbol{\beta}_m^{DCEM}| < 1. \tag{66}$$

6. Simulation Results

In this section, we demonstrate the feasibility of the proposed algorithm with two different simulated data sets,i.e, the well-separated mixture and overlapping mixture cases. In the simulations, we consider a sensor network with $M = 100$ nodes which fulfils the communication requirements specified in [11].

6.1. Well-separated Mixtures Model

First, we consider a well-separated components with the observations are generated from $J = 5$ Gaussian components distributed as in Fig. 2. Each component is a $2D$ Gaussian density, the

number of data samples at each node is $N_m = 1000$, which can represent environment data clusters. In the first 40 nodes, 60% observations come from the first Gaussian component and other 40% observations evenly from the other four Gaussian components, i.e. $\alpha_{m,1} = 60\%, \alpha_{m,2} = \alpha_{m,3} = \alpha_{m,4} = \alpha_{m,5} = 10\%$ for $m = 1, \cdots, 40$. In the next 30 nodes, 70% observations come from the second and third Gaussian components and other 30% observations evenly from the other three components, i.e. for $m = 41, \cdots, 70$, $\alpha_{m,1} = \alpha_{m,4} = \alpha_{m,5} = 10\%, \alpha_{m,2} = 40\%, \alpha_{m,3} = 30\%$. For $m = 71, \cdots, 100$, 70% observations come from the last two Gaussian component and other 30% observations evenly from the other three Gaussian components $\alpha_{m,1} = \alpha_{m,2} = \alpha_{m,3} = 10\%, \alpha_{m,4} = 40\%, \alpha_{m,5} = 30\%$. The component parameters (true values) are given by $\boldsymbol{\mu}_1 = [0.2, 0.9]'$, $\boldsymbol{\mu}_2 = [0.9, 0.2]'$, $\boldsymbol{\mu}_3 = [0.3, 0.3]'$, $\boldsymbol{\mu}_4 = [0.5, 0.5]'$, $\boldsymbol{\mu}_5 = [0.8, 0.8]'$.

For comparison, we apply the proposed DCEM algorithm, the DEM algorithm [3] with a single EM at each node, and DEMM [3] (multiple EM steps at each node) to the same batch of data. These algorithms were randomly initialized with a guess of Gaussian mixture components. As shown in Fig. 3, the estimates for the x- and y-components of means are close to the reference values.

In Fig. 4, the log-likelihood values are plotted versus iterations. Convergence is declared when the norm of the difference between successive parameter estimates is less than a specified number $\epsilon = 10^{-5}$. The proposed algorithm and DEMM algorithm require on average only 10 iterations and 11 iterations, respectively, to attain the maximal value of log-likelihood, while the DEM algorithm requires 16 iterations to converge. As the complexity of each iteration required by these algorithms is almost the same, this implies at least 37% saving in overall computation over DEM algorithm time for the proposed algorithm.

6.2. Overlapping Mixtures Model

Secondly, we consider the overlapping 2D Gaussian density with the same network setting used in previous well-sperate mixture model. Each sensor node still takes 1000 observation samples. The observations are generated from the 2D Gaussian mixtures with 5 overlapping components distributed in Fig. 5. The observations for each sensor node are collected as follows. In the first 30 nodes, 80% observations come from the first Gaussian component and the other 20% observations evenly from the other four Gaussian components, i.e. $\alpha_{m,1} = 80\%, \alpha_{m,2} = \alpha_{m,3} = \alpha_{m,4} = \alpha_{m,5} = 5\%$ for $m = 1, \cdots, 30$. In the next 40 nodes, 70% observations come from the second and third Gaussian components and other 30% observations evenly from the other three components, i.e. for $m = 41, \cdots, 80$, $\alpha_{m,1} = \alpha_{m,4} = \alpha_{m,5} = 10\%, \alpha_{m,2} = 40\%, \alpha_{m,3} = 30\%$. For $m = 71, \cdots, 100$, 70% observations come from the last two Gaussian component and other 30% observations evenly from the other three Gaussian components $\alpha_{m,1} = \alpha_{m,2} = \alpha_{m,3} = 10\%, \alpha_{m,4} = 40\%, \alpha_{m,5} = 30\%$. The component parameters (true values) are given by $\boldsymbol{\mu}_1 = [0.2, 0.9]'$, $\boldsymbol{\mu}_2 = [0.9, 0.2]'$, $\boldsymbol{\mu}_3 = [0.3, 0.3]'$, $\boldsymbol{\mu}_4 = [0.5, 0.5]'$, $\boldsymbol{\mu}_5 = [0.9, 0.9]'$.

It can be seen from Fig. 6 that the estimated mean values in all nodes calculated by the DCEM algorithm approximate their true value when overlapping data exits. Fig. 7 displays the normalized log-likelihood versus the cycle of DCEM, DEM and DEMM in presence of overlapping mixtures.

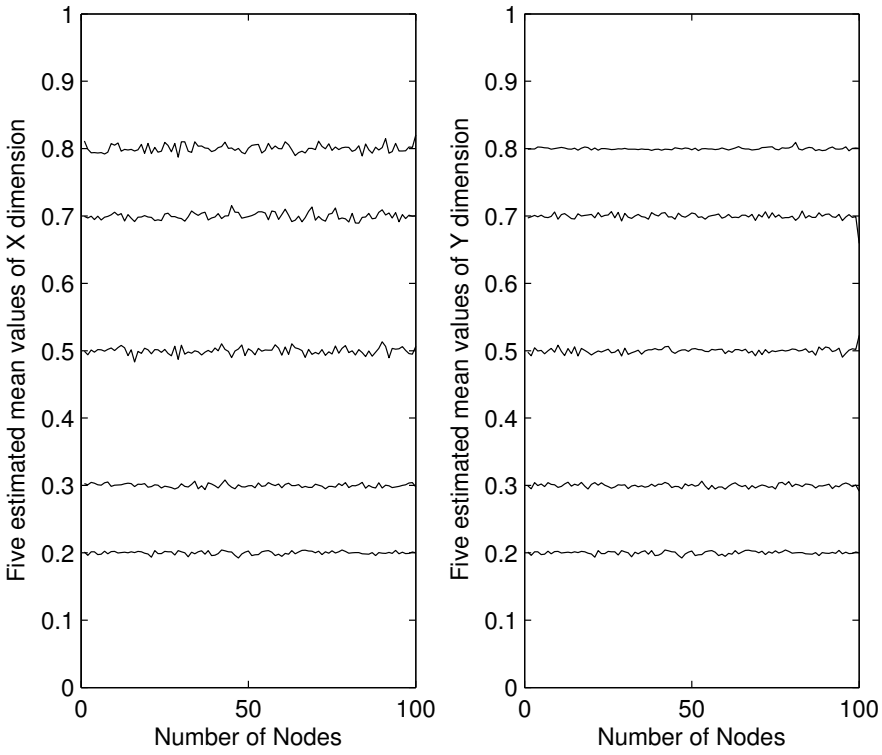


Fig. 3. *Estimates for mean values by the DCEM algorithm for well-separated mixture case.*

All three algorithms suffer from slow convergence compared to the well-separated data set, although they converge to the same solution. More specifically, the DEM algorithm with a single EM loop at each node appears to converge slowly in 33 iterations so that the DEMM algorithm and especially DCEM algorithm show a significant improvement of convergence speed around 16 iterations. Moreover, it appears that the implemented version of the DEMM algorithm is less beneficial than the DCEM algorithm for situations where the DEM algorithm converges slowly. One likely cause of this behavior is that the local procedure of DEMM at each node is still the standard EM update, which still updates the parameters simultaneously, while local DCEM algorithm find the estimates sequentially.

7. Conclusion

In this paper, we proposed a distributed componentwise EM algorithm for mixture models in sensor networks. The proposed algorithm is characterized by local processing capabilities and sequential computations of component parameters. The ability to process data locally is of particular interest to sensor networks with computationally powerful nodes, and it avoids costly node-to-node communications.

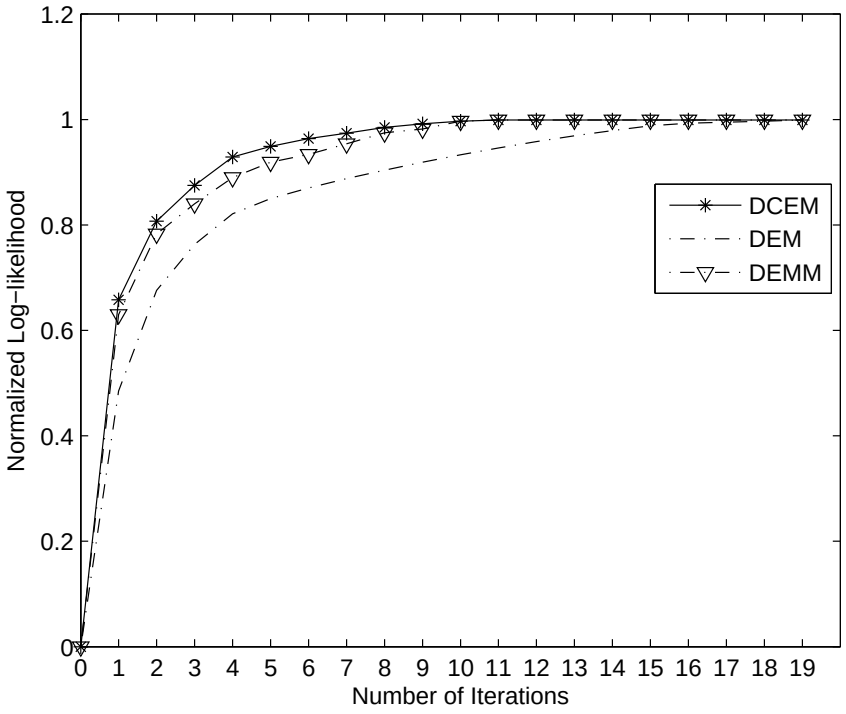


Fig. 4. *Comparison of log-likelihood versus iterations for the DCEM, DEM with single EM step at each node [3] and DEM with multiple EM steps at each node (DEMM) in the well-separated mixture case.*

More importantly, the component-wise update of the mixture parameters leads to significant improvement in convergence rate compared to the DEM algorithm [3]. Simulation results show that the number of iterations required by the proposed algorithm is about 40% less than that required by the distributed EM algorithm. Given the advantages of computational efficiency and simple implementations, the proposed distributed component-wise EM algorithm is a powerful tool for estimating mixture models in sensor networks.

8. Appendices

8.1. Proof of Theorem 1

1. Consider the EM update for the mixing proportions $\alpha_{m,j}$, from Equations (1), (2) and (46), it can be obtained

$$\frac{\partial \mathcal{L}_m(\theta)}{\partial \alpha_{m,j}}|_{\mathcal{A}_m=\mathcal{A}_m^{[t+1,j-1]}} = \tag{67}$$
$$\sum_{i=1}^{N_m} \frac{\mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_j^t, \boldsymbol{\Sigma}_j^t)}{\sum_{k=1}^{j-1}\alpha_{m,k}^{t+1}\mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_{m,k}^{t+1}, \boldsymbol{\Sigma}_{m,k}^{t+1}) + \sum_{k=j}^{J}\alpha_{m,k}^{t}\mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_{m,k}^{t}, \boldsymbol{\Sigma}_{m,k}^{t})} - N_m$$

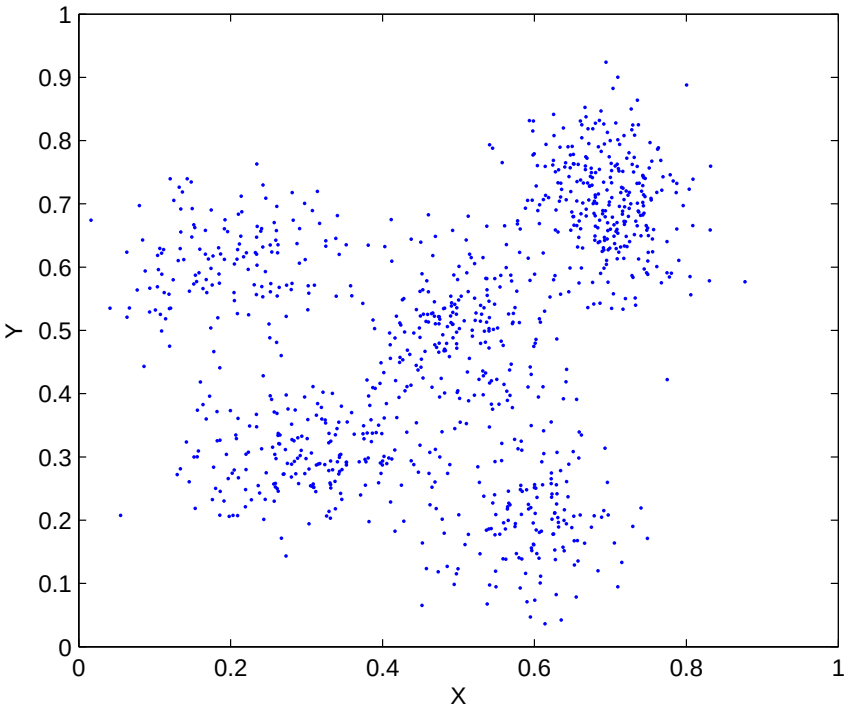


Fig. 5. *Data distribution for overlapping mixture case*

Premultiplying by $P_{\alpha_{m,j}^t}$, yields

$$P_{\alpha_{m,j}^t} \frac{\partial \mathcal{L}_m(\theta)}{\partial \alpha_{m,j}} |_{\mathcal{A}_m = \mathcal{A}_m^{[t+1,j-1]}} = \tag{68}$$

$$\frac{1}{N_m} \sum_{i=1}^{N_m} \frac{\alpha_{m,j}^t \mathcal{P}(\boldsymbol{y}_{m,i} | \boldsymbol{\mu}_j^t, \boldsymbol{\Sigma}_j^t)}{\sum_{k=1}^{j-1} \alpha_{m,k}^{t+1} \mathcal{P}(\boldsymbol{y}_{m,i} | \boldsymbol{\mu}_{m,k}^{t+1}, \boldsymbol{\Sigma}_{m,k}^{t+1}) + \sum_{k=j}^{J} \alpha_{m,k}^t \mathcal{P}(\boldsymbol{y}_{m,i} | \boldsymbol{\mu}_{m,k}^t, \boldsymbol{\Sigma}_{m,k}^t)} - \alpha_{m,j}^t \tag{69}$$

The M-step formula for $\mathcal{A}$ in equation (25) can be rewritten as

$$\alpha_{m,j}^{t+1} = \alpha_{m,j}^t + \frac{1}{N_m} \sum_{i=1}^{N_m} w_{i,m,j}^{t+1} - \alpha_{m,j}^t \tag{70}$$

2. Consider update formula for the mean $\boldsymbol{\mu}_j$ which follows from (1) and (2) that

$$\begin{aligned} \frac{\partial \mathcal{L}_m(\theta)}{\partial \boldsymbol{\mu}_{m,j}} |_{\boldsymbol{\mu}_{m,j} = \boldsymbol{\mu}_{m,j}^t} &= \sum_{i=1}^{N_m} \frac{\alpha_{m,j}^t \mathcal{P}(\boldsymbol{y}_{m,i} | \boldsymbol{\mu}_j^t, \boldsymbol{\Sigma}_j^t)}{\sum_{k=1}^{j-1} \alpha_{m,k}^{t+1} \mathcal{P}(\boldsymbol{y}_{m,i} | \boldsymbol{\mu}_{m,k}^{t+1}, \boldsymbol{\Sigma}_{m,k}^{t+1}) + \sum_{k=j}^{J} \alpha_{m,k}^t \mathcal{P}(\boldsymbol{y}_{m,i} | \boldsymbol{\mu}_{m,k}^t, \boldsymbol{\Sigma}_{m,k}^t)} \\ &\times (\boldsymbol{\Sigma}_{m,j}^t)^{-1} [\boldsymbol{y}_{m,i} - \boldsymbol{\mu}_{m,j}^t] \\ &= \sum_{i=1}^{N_m} w_{i,m,j}^{t+1} (\boldsymbol{\Sigma}_{m,j}^t)^{-1} [\boldsymbol{y}_{m,i} - \boldsymbol{\mu}_{m,j}^t] \end{aligned} \tag{71}$$

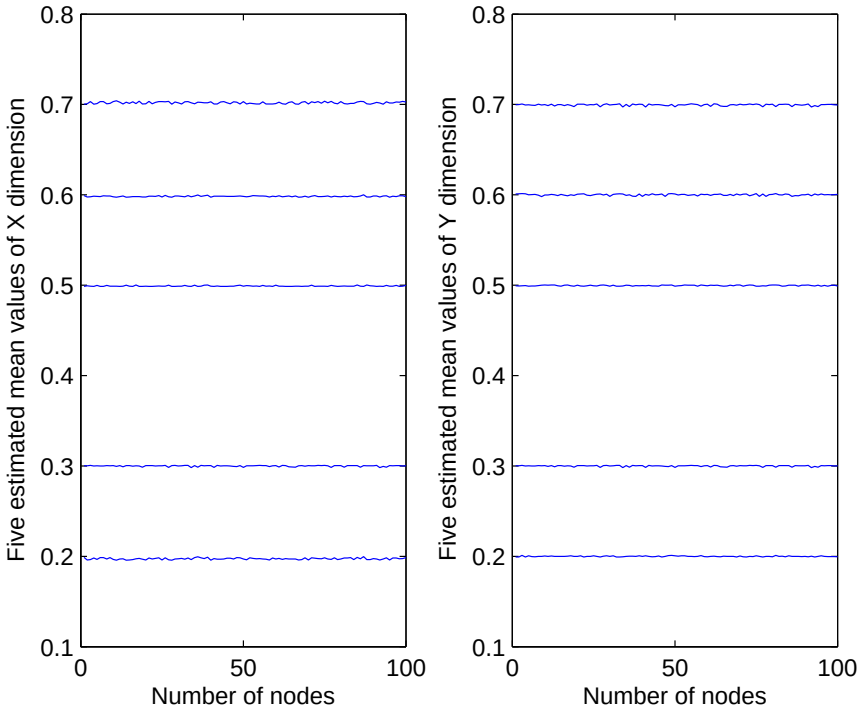


Fig. 6. *Estimates for mean values by the DCEM algorithm for overlapping mixture case.*

Premultiplying by $P_{\boldsymbol{\mu}_{m,j}^t}$, yields

$$P_{\boldsymbol{\mu}_{m,j}^t}\frac{\partial \mathcal{L}_m(\theta)}{\partial \boldsymbol{\mu}_{m,j}}|_{\boldsymbol{\mu}_{m,j}=\boldsymbol{\mu}_{m,j}^t} = \frac{1}{\sum_{i=1}^{N_m} w_{i,m,j}^{t+1}}\sum_{i=1}^{N_m} w_{i,m,j}^{t+1}\boldsymbol{y}_{m,i} - \boldsymbol{\mu}_{m,j}^t \\ = \boldsymbol{\mu}_{m,j}^{t+1} - \boldsymbol{\mu}_{m,j}^t \tag{72}$$

Based on equation (25), it can be derived that $\sum_{i=1}^{N_m} w_{i,m,j}^{t+1} > 0$, and $\boldsymbol{\Sigma}_j^t$ is positive definite with 1-probability under the assumption of a large enough N_m (the matrix has full rank). Similarly, based on (26), $P_{\boldsymbol{\mu}_j^t}$ is positive definite with 1-probability. 3. The third piece of the theorem is based on the equation (1) and (2) that

$$\frac{\partial \mathcal{L}_m(\theta)}{\partial \boldsymbol{\Sigma}_j}|_{\boldsymbol{\Sigma}_j=\boldsymbol{\Sigma}_j^t} = -\frac{1}{2}\sum_{i=1}^{N_m}\frac{\alpha_{m,j}^t \mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_j^t,\boldsymbol{\Sigma}_j^t)}{\sum_{k=1}^{j-1}\alpha_{m,k}^{t+1}\mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_k^{t+1},\boldsymbol{\Sigma}_k^{t+1}) + \sum_{k=j}^{J}\alpha_{m,k}^t\mathcal{P}(\boldsymbol{y}_{m,i}|\boldsymbol{\mu}_k^t,\boldsymbol{\Sigma}_k^t)} \\ \times (\boldsymbol{\Sigma}_j^t)^{-1}\{\boldsymbol{\Sigma}_j^t - [\boldsymbol{y}_{m,i}-\boldsymbol{\mu}_j^t][\boldsymbol{y}_{m,i}-\boldsymbol{\mu}_j^t]^T\}(\boldsymbol{\Sigma}_j^t)^{-1} \\ = -\frac{1}{2}\sum_{i=1}^{N_m} w_{i,m,j}^{t+1}(\boldsymbol{\Sigma}_j^t)^{-1}\{\boldsymbol{\Sigma}_j^t - [\boldsymbol{y}_{m,i}-\boldsymbol{\mu}_j^t][\boldsymbol{y}_{m,i}-\boldsymbol{\mu}_j^t]^T\}(\boldsymbol{\Sigma}_j^t)^{-1} \tag{73}$$

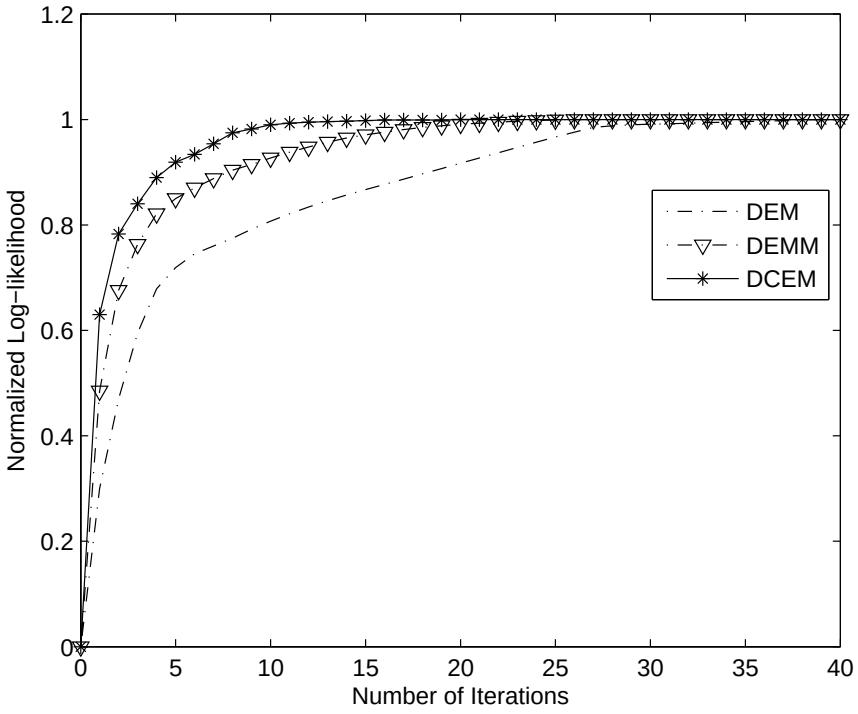


Fig. 7. *Comparison of log-likelihood versus iterations for the DCEM, DEM with single EM step at each node [3] and DEM with multiple EM steps at each node (DEMM) in the overlapping mixture case.*

Based on the discussion above, the EM update formula for $\boldsymbol{\Sigma}_j^t$ can be restructured as

$$\begin{aligned}\boldsymbol{\Sigma}_j^{t+1} &= \boldsymbol{\Sigma}_j^t + \frac{1}{\sum_{i=1}^{N_m} w_{i,m,j}^{t+1}} \sum_{i=1}^{N_m} w_{i,m,j}^{t+1} [\boldsymbol{y}_{m,i} - \boldsymbol{\mu}_j^t][\boldsymbol{y}_{m,i} - \boldsymbol{\mu}_j^t]^T - \boldsymbol{\Sigma}_j^t \\ &= \boldsymbol{\Sigma}_j^t + \frac{2\boldsymbol{\Sigma}_j^t}{\sum_{i=1}^{N_m} w_{i,m,j}^{t+1}} V_{\boldsymbol{\Sigma}_j^t} \boldsymbol{\Sigma}_j^t,\end{aligned} \tag{74}$$

where

$$\begin{aligned}V_{\boldsymbol{\Sigma}_j^t} &= -\frac{1}{2} \sum_{i=1}^{N_m} w_{i,m,j}^{t+1} (\boldsymbol{\Sigma}_j^t)^{-1} \{\boldsymbol{\Sigma}_j^t - [\boldsymbol{y}_{m,i} - \boldsymbol{\mu}_j^t][\boldsymbol{y}_{m,i} - \boldsymbol{\mu}_j^t]^T\} (\boldsymbol{\Sigma}_j^t)^{-1} \\ &= \frac{\partial \mathcal{L}_m(\theta)}{\partial \boldsymbol{\Sigma}_j} |_{\boldsymbol{\Sigma}_j = \boldsymbol{\Sigma}_j^t}\end{aligned} \tag{75}$$

which yields

$$\boldsymbol{\Sigma}_j^{t+1} = \boldsymbol{\Sigma}_j^t + \frac{2\boldsymbol{\Sigma}_j^t}{\sum_{i=1}^{N_m} w_{i,m,j}^{t+1}} \frac{\partial \mathcal{L}_m(\theta)}{\partial \boldsymbol{\Sigma}_j} |_{\boldsymbol{\Sigma}_j = \boldsymbol{\Sigma}_j^t} \tag{76}$$

Facilitating the definition of vec operator, $\text{vec}[\boldsymbol{ABC}] = (\boldsymbol{C}^T \otimes \boldsymbol{A})\text{vec}[\boldsymbol{B}]$, it can be derived

$$\text{vec}[\boldsymbol{\Sigma}_j^{t+1}] = \text{vec}[\boldsymbol{\Sigma}_j^t] + \frac{2}{\sum_{i=1}^{N_m} w_{i,m,j}^{t+1}} (\boldsymbol{\Sigma}_j^t \otimes \boldsymbol{\Sigma}_j^t) \frac{\partial \mathcal{L}_m(\theta)}{\partial \boldsymbol{\Sigma}_j} |_{\boldsymbol{\Sigma}_j = \boldsymbol{\Sigma}_j^t} \tag{77}$$

8.2. Proof of Theorem 2

With (59), it can be obtained:

$$\begin{aligned} \mathbf{R}_{\theta_{m,j}}^T \mathbf{M}_{m,j}^{DCEM} \mathbf{R}_{\theta_{m,j}} &= \begin{bmatrix} \mathbf{I} - \mathbf{P}_{\theta_{m,j}^*} \mathbf{H}_{\theta_{m,j}^*} & -\mathbf{P}_{\theta_{m,j}^*} \mathbf{H}_{\theta_l^*} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \\ &= \mathbf{I} - \begin{bmatrix} \mathbf{I} \\ \mathbf{0} \end{bmatrix} \mathbf{P}_{\theta_{m,j}^*} [\mathbf{I} \ \mathbf{0}] \mathbf{R}_{\theta_{m,j}}^T \mathbf{H}_{\theta_m^*} \mathbf{R}_{\theta_{m,j}} \end{aligned} \tag{78}$$

Therefore,

$$\mathbf{M}_{m,j}^{DCEM} = \mathbf{I} - \mathbf{R}_{\theta_{m,j}} \begin{bmatrix} \mathbf{I} \\ \mathbf{0} \end{bmatrix} \mathbf{P}_{\theta_{m,j}^*} [\mathbf{I} \ \mathbf{0}] \mathbf{R}_{\theta_{m,j}}^T \mathbf{H}_{\theta_m^*} \tag{79}$$

Multiplying both sides with $\mathbf{H}_{\theta_m^*}^{\frac{1}{2}}$ and $\mathbf{H}_{\theta_m^*}^{-\frac{1}{2}}$, (79) becomes

$$\mathbf{H}_{\theta_m^*}^{\frac{1}{2}} \mathbf{M}_{m,j}^{DCEM} \mathbf{H}_{\theta_m^*}^{-\frac{1}{2}} = \mathbf{I} - \mathbf{H}_{\theta_m^*}^{\frac{1}{2}} \mathbf{R}_{\theta_{m,j}^*} \begin{bmatrix} \mathbf{P}_{\theta_{m,j}^*} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{R}_{\theta_{m,j}}^T \mathbf{H}_{\theta_m^*}^{\frac{1}{2}} \tag{80}$$

If parameter sets are chosen cyclically in a natural order, i.e., $\{1, \cdots, J\}$, it follows from (80) that

$$\mathbf{M}_{m,J}^{DCEM} \times \cdots \mathbf{M}_{m,1}^{DCEM} = \mathbf{I} - \mathbf{P}_{\theta_m^*} \mathbf{H}_{\theta_m^*} \tag{81}$$

where

$$\mathbf{P}_{\theta_m^*} = \mathbf{D}_{\mathbf{P}} + \mathbf{L}_{\mathbf{P}} \tag{82}$$

where

$$\mathbf{D}_{\mathbf{P}} = \begin{bmatrix} \mathbf{P}_{\theta_{m,1}^*} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \mathbf{P}_{\theta_{m,J}^*} \end{bmatrix} \tag{83}$$

is the block diagonal of $\mathbf{P}_{\theta_m^*}$, and $\mathbf{L}_{\mathbf{P}}$ is the corresponding strictly lower block triangular matrix of $\mathbf{P}_{\theta_m^*}$. Then, we decompose the local Hessian matrix $\mathbf{H}_{\theta_m^*}$ by

$$\mathbf{H}_{\theta_m^*} = \mathbf{D}_{\mathbf{H}} + \mathbf{L}_{\mathbf{H}} + \mathbf{L}_{\mathbf{H}}^T \tag{84}$$

where $\mathbf{D}_{\mathbf{H}}, \mathbf{L}_{\mathbf{H}}$ represent the block diagonal, strictly lower triangular block parts of $\mathbf{H}_{\theta_m^*}$. From [9, Theorem 2], it can be verified that $\mathbf{D}_{\mathbf{P}} = \mathbf{D}_{\mathbf{H}}^{-1}$, we rewrite the (82) as follow:

$$\mathbf{P}_{\theta_m^*} = (\mathbf{D}_{\mathbf{H}} + \mathbf{L}_{\mathbf{H}})^{-1} \tag{85}$$

Thus, by letting

$$\mathbf{M}_m^{DCEM} = \prod_{j=1}^{J} \mathbf{M}_{m,j}^{DCEM} \tag{86}$$

the local rate matrix of DCEM at node m is given by

$$\mathbf{M}_m^{DCEM} = \mathbf{I} - (\mathbf{D_H} + \mathbf{L_H})^{-1}\mathbf{H}_{\theta_m^*} \tag{87}$$

Let $\|\mathbf{M}\|_2 = \sqrt{\rho(\mathbf{M}^H\mathbf{M})}$ denotes the matrix spectral norm of $\mathbf{M}$ and define $\bar{\mathbf{M}} = \mathbf{H}_{\theta_m^*}^{\frac{1}{2}}\mathbf{M}_m^{DCEM}\mathbf{H}_{\theta_m^*}^{-\frac{1}{2}}$, according to (87),

$$\begin{aligned}\bar{\mathbf{M}}^H\bar{\mathbf{M}} &= \bar{\mathbf{M}}^T\bar{\mathbf{M}} \\ &= \mathbf{I} - \mathbf{H}_{\theta_m^*}^{\frac{1}{2}}(\mathbf{D_H} + \mathbf{L_H})^{-T}(\mathbf{D_H} + \mathbf{L_H} + \mathbf{D_H}^T + \mathbf{L_H}^T - \mathbf{H}_{\theta_m^*})(\mathbf{D_H} + \mathbf{L_H})^{-1}\mathbf{H}_{\theta_m^*}^{\frac{1}{2}} \\ &< \mathbf{I}\end{aligned} \tag{88}$$

By defining

$$\|\mathbf{M}_m^{DCEM}\|_{\mathbf{H}_{\theta_m^*}}^2 = \rho(\bar{\mathbf{M}}^H\bar{\mathbf{M}}) \tag{89}$$

The inequality $\rho(\mathbf{M}) \leq \|\mathbf{M}\|_{\mathbf{N}}$ leads to

$$\begin{aligned}\rho(\mathbf{M}_m^{DCEM}) &\leq \|\mathbf{M}_m^{DCEM}\|_{\mathbf{H}_{\theta^*}} \\ &= \sqrt{\rho(\bar{\mathbf{M}}^H)} \\ &< \sqrt{\rho(\mathbf{I})} = 1\end{aligned} \tag{90}$$

9. References

[1] L. F. Akyildiz and W. Su and Y. Sankarasubramaniam and Erdal Cayirci, *A survey on sensor networks,* 2002.

[2] D. Estrin and R. Govindan and J. Heidemann and S. Kumar, *Next century challenges: scalable coordination in sensor networks,* Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking, pp. 263-270, 1999.

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

[4] A. P. Dempster and N. Laird and D. B. Rubin, *Maximum likelihood from incomplete data via the EM algorithm*, Journal of the Royal Statistical Society, Series B, 29, pp. 1-38.

[5] Sayed A.H., Tu S.Y., Chen J., Zhao X., Towfic Z. Diffusion strategies for adaptation and learning over networks: An examination of distributed strategies and network behavior. IEEE Signal Process. Mag. 2013;30:155-171

[6] Xiao L, Boyd S, Lall S. A space-time diffusion scheme for peer-to-peer least-squares estimation. Proceeding of the Information Processing in Sensor Networks (IPSN); Nashville, TN, USA. April 2006; pp. 168-176.

[7] Lopes C, Sayed AH. Incremental adaptive strategies over distributed networks. IEEE Trans. Signal Process. 2007; 55: 4064-4077.

[8] Schizas I, Mateos G, Giannakis G. Distributed LMS for consensus-based in-network adaptive processing. IEEE trans. Signal Process. 2009; 56: 2365-2381

[9] J. A. Fessler and A. O. Hero, *Space alternating generalized Expectation-Maximization algorithm*, IEEE Transactions on Signal Processing, vol. 42, no. 10, pp. 2664-2677, Oct. 1994.

[10] D. Gu, *Distrbuted EM algorithm for Gaussian mixtures in sensor network,* IEEE Transactions on Neural Network, vol. 19, no. 7, pp. 1154-1166, Jul. 2008

[11] P. Gupta and P. R. Kumar, *The capacity of wireless networks,* IEEE Transactions on Information Theory, vol. 46, no. 2, pp. 388-404, Nov. 2000.

[12] W. Kowalczyk and N. Vlassis, *Newcast EM,* in Proc. Adv. Neural Inf. Process. Syst., Vancouver, BC, Canada, pp. 13-18, Dec. 2005.

[13] G. Celeux and S. Chretien and F. Forbes and A. Mkhadri, *A componentwise EM algorithm for mixtures,* Journal of Computational and Graphical Statistics, Vol 10(4), 2001.

[14] G. McLachlan and D. Peel, *Finite Mixture Models,* Wiley Series in Probability and Statistics, Oct. 2000.

[15] X. L. Meng and D. van Dyk , *The EM Algorithm – An old folk song sung to the fast tune*, Journal of the Royal Statistical Society, Ser. B., vol. 59, pp. 511-567, 1997.

[16] R. Neal and G. E. Hinton, *A view of the EM algorithm that justifies incremental, sparse, and other variants*, in *Learning in Graphical Models*, Kluwer Academic Publishers, 1998, 355-368.

[17] J. Wolfe and A. Haghighi and D. Klein, *Fully distributed EM for very large datasets,* in Proc. 25th Int. Conf. Mach. Learn., Helsinki, Finland, pp. 1184-1191, Jul. 6-8, 2008.

[18] J. Besag, On the statistical analysis of dirty pictures, Journal of the Royal Statistical Society, Series B, vol.48, no.3, pp. 259-302.

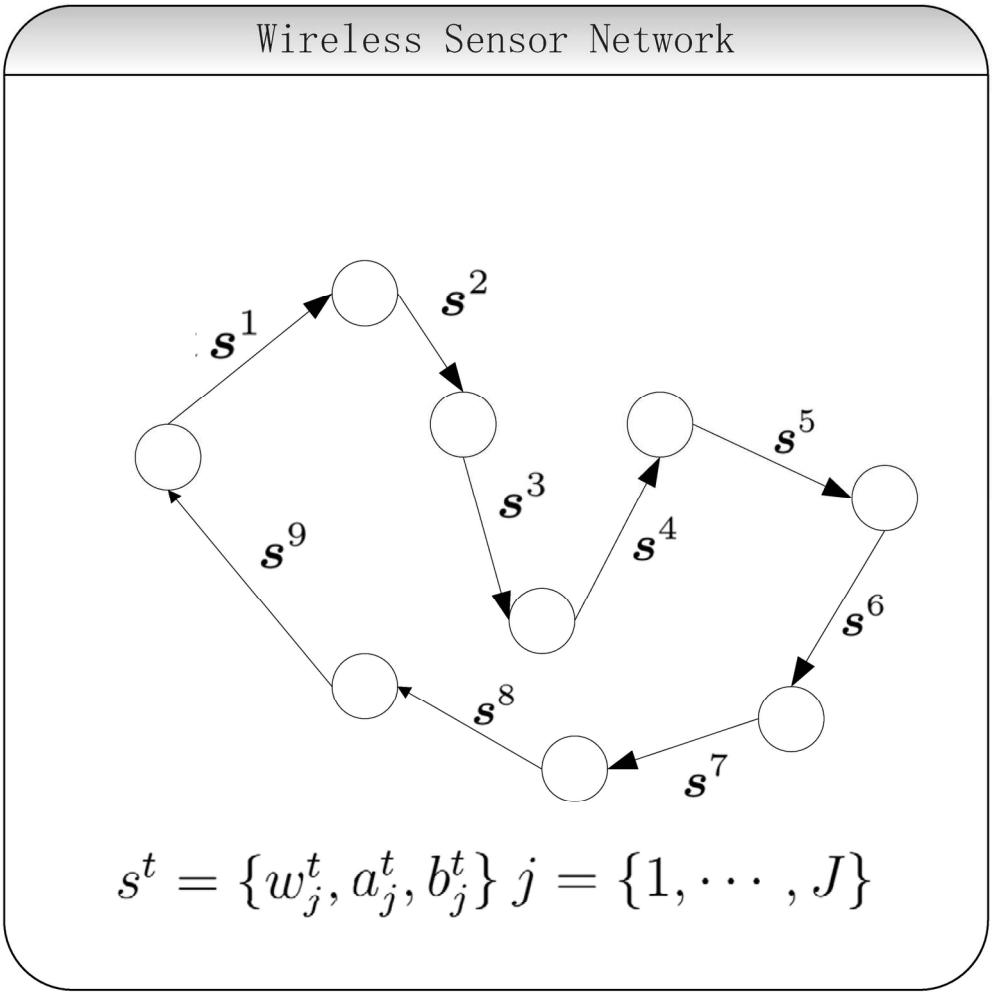
[19] Xu, Lei, and Michael I. Jordan. ”On convergence properties of the EM algorithm for Gaussian mixtures.” Neural computation 8.1 (1996): 129-151.

[20] Ma, Jinwen, Lei Xu, and Michael I. Jordan. ”Asymptotic convergence rate of the EM algorithm for Gaussian mixtures.” Neural Computation 12.12 (2000): 2881-2907.

[21] Safarinejadian, Behrooz, Mohammad B. Menhaj, and Mehdi Karrari. ”A distributed EM algorithm to estimate the parameters of a finite mixture of components.” Knowledge and information systems 23.3 (2010): 267-292.

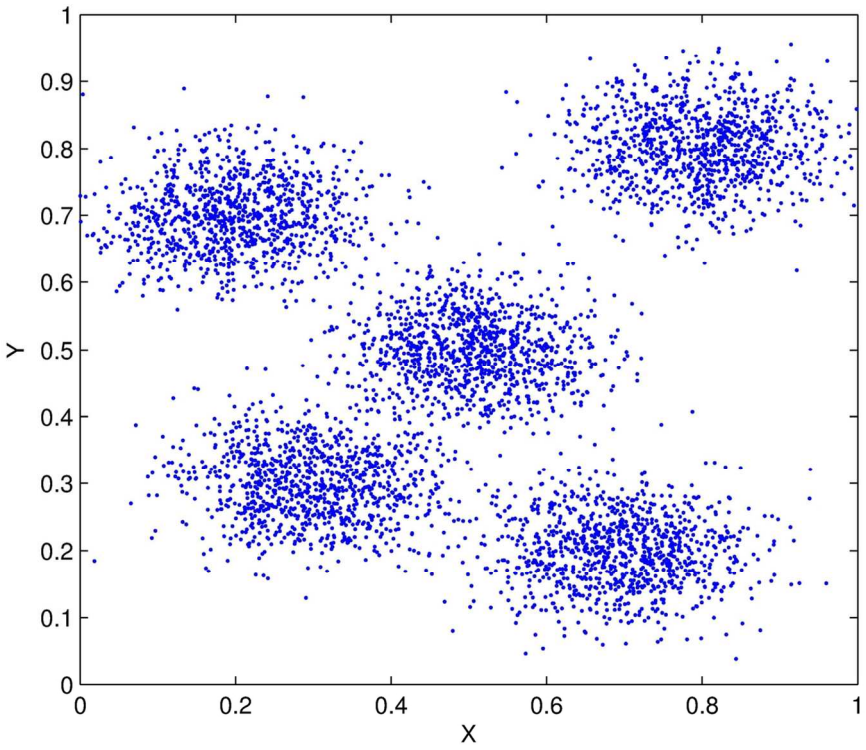
[22] Horn, Roger A., and Charles R. Johnson. Matrix analysis. Cambridge university press, 2012.

[23] Polak, Elijah. Computational methods in optimization: a unified approach. Vol. 77. Academic press, 1971.



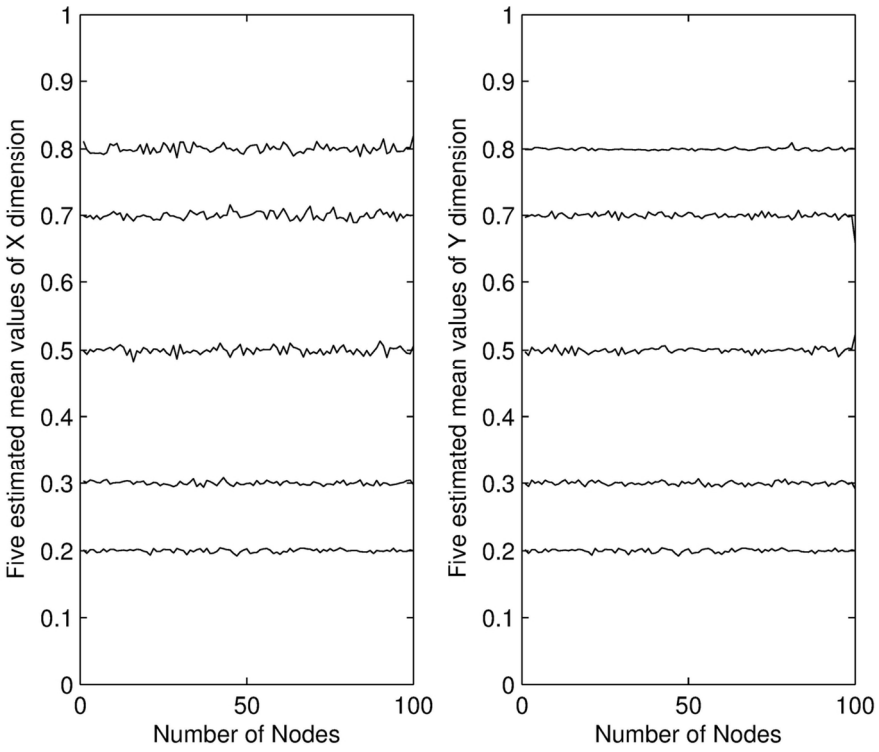
Communication/iteration cycle in a sensor network

153x154mm (300 x 300 DPI)



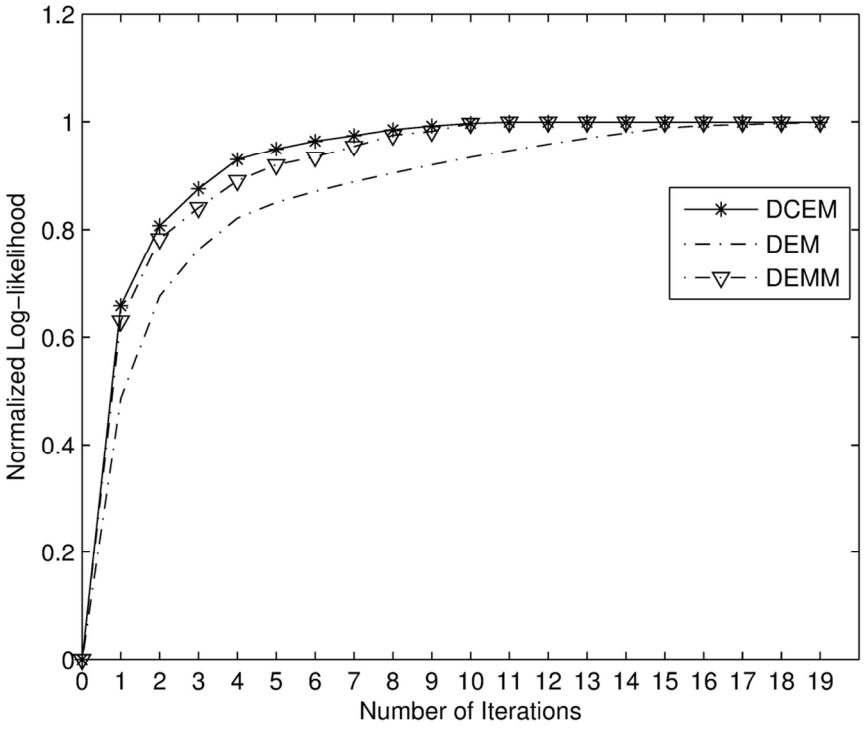
Data distribution for well-separated mixture case

121x100mm (300 x 300 DPI)



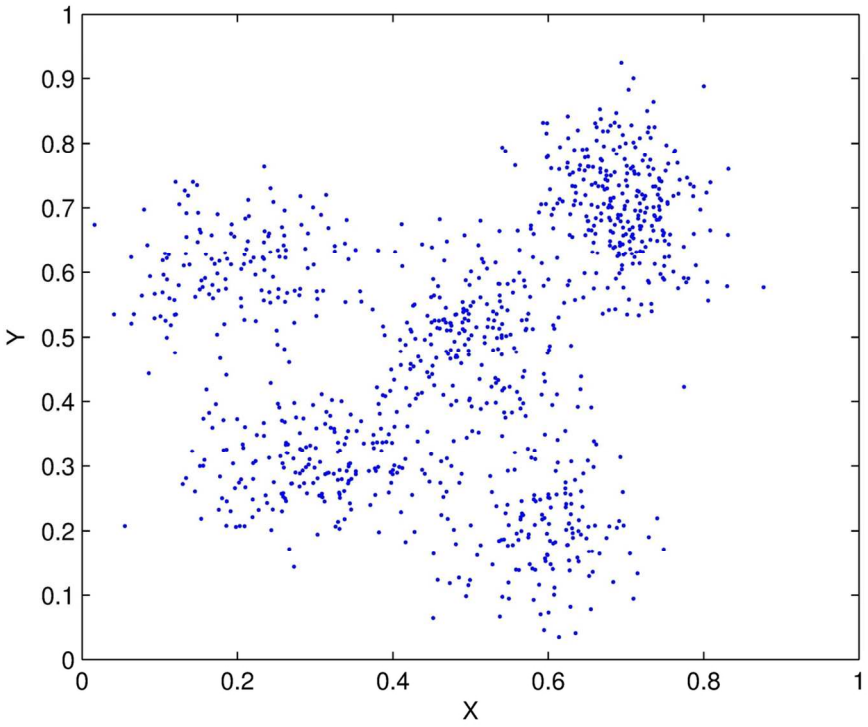
Estimates for mean values by the DCEM algorithm for well-separated mixture case

121x100mm (300 x 300 DPI)



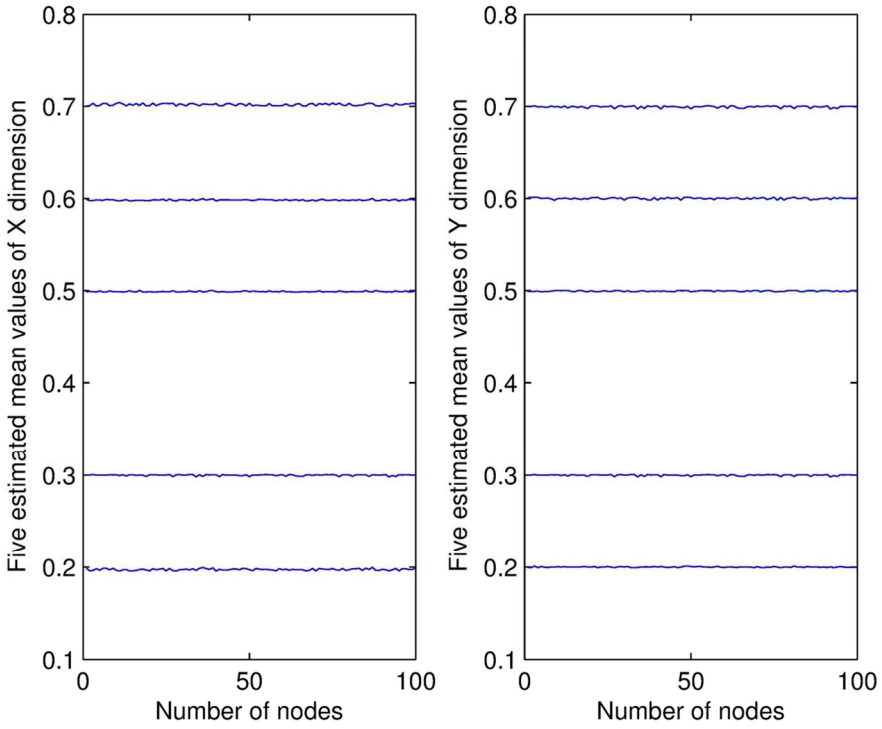
Comparison of log-likelihood versus iterations for the DCEM, DEM with single EM step at each node \cite{Nowak03} and DEM with multiple EM steps at each node (DEMM) in the well-separated mixture case.

114x90mm (300 x 300 DPI)



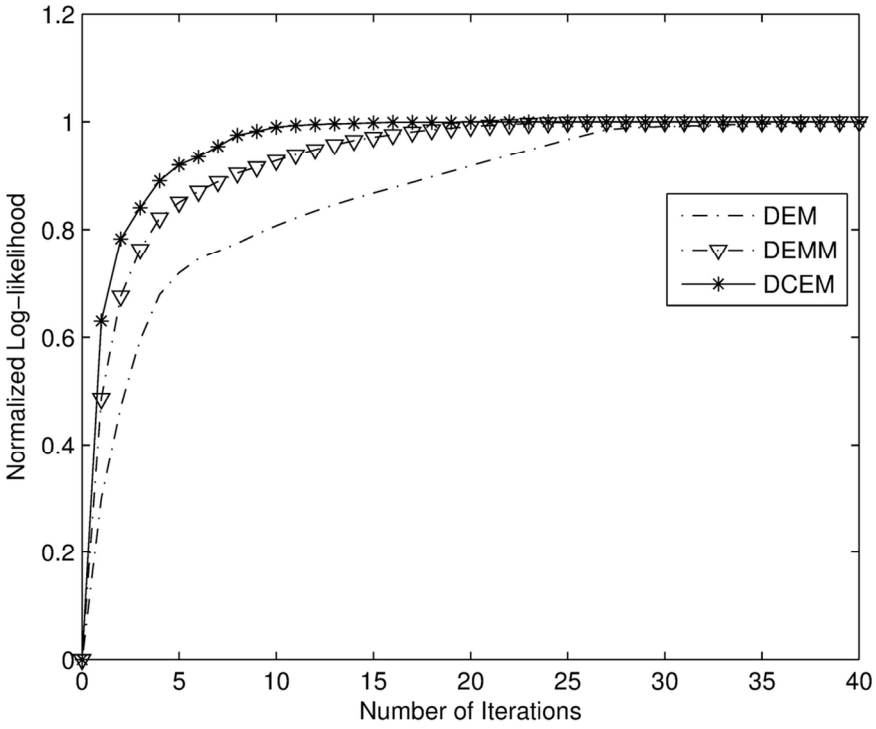
Data distribution for overlapping mixture case

114x90mm (300 x 300 DPI)



Estimates for mean values by the DCEM algorithm for overlapping mixture case

114x90mm (300 x 300 DPI)



Comparison of log-likelihood versus iterations for the DCEM, DEM with single EM step at each node \cite{Nowak03} and DEM with multiple EM steps at each node (DEMM) in the overlapping mixture case.

114x90mm (300 x 300 DPI)

Distributed quasi-Newton Method for Power System State Estimation

Jia Yu, and John Thompson, *Fellow, IEEE,*

***Abstract*—In this paper, the system-wide power system state estimation (PSSE) is promising in loosing the energy market and improving the situational awareness. In practice, the use of centralised estimator is not viable due to the high complexity, communication cost, and robustness issues. Thus, with the systematic manner, we consider the distributed PSSE approaches which are designed based on the quasi-Newton and backtracking line search. We demonstrate the effectiveness of the proposed algorithms via the IEEE 14- and 4200- buses. It is shown in the simulation results that the proposed method performs better than other algorithms when dealing with bad data and large-scale problems.**

***Index Terms*—power system state estimation, quasi-Newton, average consensus.**

I. Introduction

STATE estimation functions as an essential part in power systems. It significantly impacts the capabilities in dispatching power, frequency management and error identifications. The system administrator can monitor the state of the power grid via state estimation methods [**?**]. It has become more and more important to estimate the system states with better accuracies. Researchers have made great efforts in combing new sensing techniques with the state-of-the-art state estimations. For example, in [**?**], the authors presented a Wide-Area Measurement System (WAMS) aided by Phasor Measurement Units (PMUs). Since the computational load is proportional to the amount of measurements, state-of-the-art systems would require the individual buses to have their own processing abilities [3]. The distributed methods have benefits in reliabilities, computational efficiency, communication load, and memory storage.

There have been a number of research efforts on investigating distributed state estimation approaches for power systems. The hierarchical distributed approaches estimate system states locally, exchange the information using a central processor, and combine the local estimations to give the overall estimates [4]–[6]. However, such methods are limited by the communication burden. In general, the distributed state estimations require the local communications rather than counting on a central processor. Recent developments in fully distributed methods include: leveraging the matrix decompositions [7], [8]; employing the alternating direction method of multipliers (ADMM) method [9]; and information filter-based techniques [10]. The matrix decomposition methods in [7], [8] give no guarantee on the convergence of the distributed state estimates. The ADMM approach in [9] guarantees the asymptotic convergence. However, the use of ADMM methods is limited, since the Lagrange multipliers require extra memory and asynchronous configurations can be troublesome. The method proposed in [10] guarantees the convergence, but the required iterations scale linearly with the scale of the network. Asymptotically convergent approaches can be particularly useful to deal with large-scale networks especially when the convergence rate is independent of the scale of the network.

In [11]–[13], the authors proposed the gossip-based algorithms for complete distributed state estimations. In particular, the method presented in [11] is a first order approach driven by the diffusion strategy in [12]. Although the first order approaches are simple, their developments are hampered by the slow convergence rate. However, the Newton-type methods usually have quadratic convergence. A gossip-based Gauss-Newton method was developed in [13] to solve the general nonlinear least squares problem and applied to the power system state estimation. The Gauss-Newton method only exploits the presence of first-order information of Hessian, and thus requires the cost function to be zero or a small residual. However, the presence of bad data will result in a large residual in power system, which cannot be neglected during the estimation process. Such situation can no longer be handled by Gauss-Newton methods efficiently. By contrast, quasi-Newton methods are more efficient under these conditions, approximated Hessian can preserve second order information, which allows our method to reduce the impact of bad data on the state estimates.

With this context, we reformulate the state estimation problem and propose a distributed quasi-Newton method (DQN) for wide-area PSSE. Similar with [13],and employ the multi-agent gossip-based scheme to describe the network communications. Under this scheme, the state of each agent (control area) can be estimated by using the local information and a limited information exchange with neighbor areas, for which the fusion center is not necessary. The agents can only preserve their own states. This has advantages in both communication efficiency and storages, address the large residual or bad data problem [18], [19].

Moreover, we introduce a distributed line search method to accelerate the convergence of the presented approach in [20]. Our investigation aims to extend the commonly used Armijo rule in backtracking line searches [20]. We form a local Armijo

J. Yu was with the Institute for Digital Communications, The University of Edinburgh, UK e-mail: j.yu@ed.ac.uk.

Manuscript submmited Feb 5, 2017.

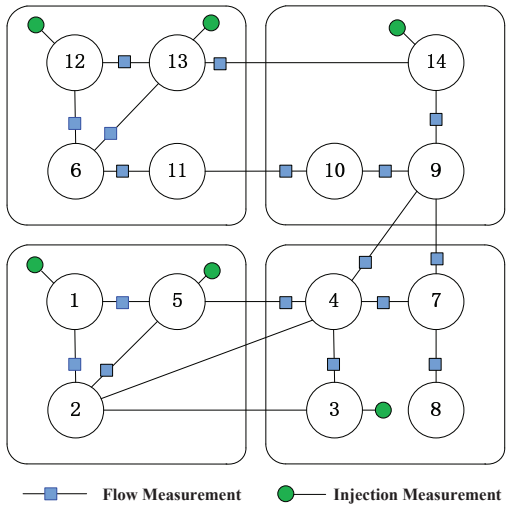


Fig. 1: IEEE 14 bus system partitioned into $I = 4$ control areas

rule for each agent by taking only the valid terms at that agent, using the exchange information from neighbours. These features make the proposed approach a viable distributed alternate to the central line search methods.

In Section II, we formulate the power system state estimation into a (non)linear least square (LS) problem. In Section III, we introduce centralized BFGS with line search for the LS problem. Furthermore, the details of the proposed distributed quasi-Newton method are presented in Section IV. Finally, in Section VI the numerical simulations are conducted to show the performance of our approach.

II. Problem Formulation

A multi-area power network can be conveniently expressed as an undirected graph $(\mathcal{N}, \mathcal{E})$, where the set of the vertices $\mathcal{N} \triangleq \{1, \cdots, B\}$ denotes buses and the edge set $\mathcal{E}$ represents the transmission lines that connect the buses. The power system state is normally defined as the collection of the voltages (containing both phase and magnitude information) at all buses, $\mathbf{x} = [\mathbf{\Theta}^T, \mathbf{V}^T]^T$ with $\mathbf{\Theta} \triangleq [\theta_1, \cdots, \theta_B]^T$ being the phase vector and $\mathbf{V} \triangleq [V_1, \cdots, V_B]^T$ the magnitude vector. The whole network can be divided into I non-overlapping areas, each governed by a control cite, which gathers the local measurements taken at the corresponding area and is allowed to communicate with its neighboring areas. Fig. 1 shows a concrete example where the network is partitioned into $I = 4$ regions. Apparently, the local measurements available to one control cite is insufficient for it to estimate the total system state. Therefore, in this work we study how to design the cooperation process between the multiple areas so that a distributed estimation of the global state can be efficiently implemented.

We consider the traditional measurement system, SCADA (supervisory control and data acquisition), which provides measurements on both power injections at some of the buses and on power flows along some of the transmission lines. Since in SCADA system, the measurements update rate is around once 2-6 seconds, which is relatively a long period of time compared with the communication delay between different cites, a static setting is considered in this paper, i.e. measurement set is separated into different snapshots and each run of state estimation process is based upon the most recent one. The measurement model can therefore be represented as:

$$\mathbf{t}_i = \mathbf{h}_i(\mathbf{x}) + \mathbf{e}_i \tag{1}$$

where $\mathbf{e}_i$ denotes measurement noise at the i^{th} sensor as well as some other uncertainties, such as the modeling inaccuracy, and $\mathcal{I} = \{1, \cdots, I\}$ where I is the number of control sites. We further define $M = 2B$ as the dimension of the system state. In general, the observation function $\{\mathbf{h}_i(\mathbf{x})_{i=1}^I\}$ should be nonlinear. It is only in some special cases, such as when PMU measurements are considered, that the observation function can be linear. In this paper, the general case is studied. By stacking the local measurements together, the global expression is shown as

$$\mathbf{t} = \mathbf{h}(\mathbf{x}) + \mathbf{e}, \tag{2}$$

where $\mathbf{h}(\mathbf{x}) = [\mathbf{h}_1^T, \cdots, \mathbf{h}_I^T]^T, \mathbf{e} = [\mathbf{e}_1^T, \cdots, \mathbf{e}_I^T]^T$. A weighted least squares problem related to this global representation can be written as

$$\hat{\mathbf{x}} = \min_{\mathbf{x} \in \mathbf{X}} J(\mathbf{x}) = (\mathbf{t} - \mathbf{h}(\mathbf{x}))^T R^{-1} (\mathbf{t} - \mathbf{h}(\mathbf{x})) \tag{3}$$

where $R = cov([\mathbf{e}_1, \cdots, \mathbf{e}_N]^T)$ and $\mathbf{X} \doteq \{\theta_n \in [-\theta_{max}, \theta_{max}], V_n \in [0, V_{max}], n \in \mathcal{N}\}$, with $\theta_{\max}$ and $V_{\max}$ being the phase angle and voltage limit. According to [14], problem (3) is equivalent to the maximum likelihood estimation for (2), under the assumption that the measurement errors at different regions are gaussian and uncorrelated with each other, i.e. $R = diag(R_1, \cdots, R_I)$ with R_i being the covariance matrix for the measurement error at the i^{th} region. Since R is block diagonal matrix, problem (3) can be reformulated to facilitate a distributed implementation,

$$\hat{\mathbf{x}} = \min_{\mathbf{x} \in \mathbf{X}} J(\mathbf{x}) = \sum_{i=1}^{I} \|\bar{\mathbf{t}}_i - \bar{\mathbf{h}}_i(\mathbf{x})\|^2, \tag{4}$$

with $\bar{\mathbf{t}}_i = R_i^{-\frac{1}{2}} \mathbf{t}_i$ and $\bar{\mathbf{h}}_i = R_i^{-\frac{1}{2}} \mathbf{h}_i$. Both problem (3) and its distributed version (4) are essentially non-linear least squares problems. For centralized processing structure, Newton type algorithms are typically used to search for the stationary point because of their faster convergence rate than the first-order methods such as gradient-descent method and ADMM. In the next section, we introduce the centralized approach for solving problem (4), using a particular type of quasi-Newton algorithm.

III. Centralized quasi-Newton Algorithm

A multi-agent network previously illustrated through Fig. 1 is considered, where there are I distributed agents, and the

i^{th} agent only knows a subset function $\mathbf{f}_i(\mathbf{x}) \doteq \bar{\mathbf{z}}_i - \bar{\mathbf{h}}_i(\mathbf{x}) : \mathbb{R}^M \to \mathbb{R}^{N_i}, i \in \mathcal{I}$,

$$\mathbf{f}_i(\mathbf{x}) = \left[f_1(\mathbf{x})^T, \cdots f_{N_i}(\mathbf{x})^T\right]^T. \tag{5}$$

Given the Jacobian $\mathbf{J}_i = \partial \mathbf{f}_i(\mathbf{x})/\partial \mathbf{x}$, the gradient of function $\|\mathbf{f}_i\|^2$ at $\mathbf{x}$ is denoted as $F_i(\mathbf{x}) = \mathbf{J}_i(\mathbf{x})^T \mathbf{f}_i(\mathbf{x})$. Since the global objective function can be rewritten as

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}\in\mathbf{X}} \sum_{i=1}^{I} \|\mathbf{f}_i(\mathbf{x})\|^2 \tag{6}$$

where individual agent only gets access to the partial information of the global cost function, the gradient of the global function can be computed as

$$F(\mathbf{x}) = \sum_{i=1}^{I} F_i(\mathbf{x}) = \sum_{i=1}^{I} \mathbf{J}_i(\mathbf{x})^T \mathbf{f}_i(\mathbf{x}). \tag{7}$$

In this paper, we are interested in the BFGS (broydenCfletcher-CgoldfarbCshanno) quasi-Newton algorithm [29] due to its wide applications and robust performance. According to BFGS algorithm, with a properly chosen initial point $\mathbf{x}^0$ as well as a positive-definite matrix $\mathbf{H}^0$, the searching for the stationary point of problem (6) can be established by iteratively computing the following terms,

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \alpha \mathbf{H}^k F(\mathbf{x}^0), \tag{8}$$

$$\mathbf{H}^k = (I - \rho^k \mathbf{z}^k \mathbf{y}^{kT}) \mathbf{H}^{k-1} (I - \rho^k \mathbf{y}^k \mathbf{z}^{kT}) + \rho^k \mathbf{z}^k \mathbf{z}^{kT}, k \geq 0, \tag{9}$$

where $\mathbf{y}^k = \hat{F}(\mathbf{x}^k) - \hat{F}(\mathbf{x}^{k-1}), \mathbf{z}^k = \mathbf{x}^k - \mathbf{x}^{k-1}, \rho^k = 1/(\mathbf{y}^{kT}\mathbf{z}^k)$ and α is a fixed value that controls the length of the searching step. The stopping criterion for convergence is typically set by checking the difference between the objective function values in the present and the last iterations or by simply assuming a maximum limit on the iteration number.

IV. Distributed quasi-Newton Process

Motivated by the superior convergence behavior of the quasi-Newton method for large-scale optimization, we propose a distributed quasi-Newton method that combines quasi-Newton iterations with a network consensus process in this section.

Solving the minimization problem (6) in a distributed manner is challenging. The information on the global objective function is required for the computation of the searching step, as discussed in the previous section. However, in our setting, each agent can only access a part of the global information. To obviate this problem, we augment the quasi-Newton searching with gossip process, which would disseminate local information across the network. We hope that by doing so the distributed process would behave similarly as its centralized version.

A. Network Exchange Model

Gossip process is used to disseminate local information. Essentially, an agreement among all agents is reached, to a certain degree of accuracy, via proper local information exchanges prescribed in the gossip algorithm. In this section, we first brief the data exchange model before introducing some assumptions that the gossip algorithm used in this paper is built upon.

Since we assume that the data exchanges are synchronized among the agents, we can denote the epoch for the data exchanges between the k^{th} and the $k+1^{th}$ local iteration as $[\tau_k, \tau_{k+1})$. During the epoch, each agent is allowed to only communicate with its neighboring nodes. The network topology can be modeled as a time-varying graph $\mathcal{G}_{k,t} = (\mathcal{I}, \mathcal{E}_{k,t})$, where t is the counter for the data exchanges of the gossip process. The network topology is therefore assumed to be stationary only within a single exchange stage in the gossip process. The node set corresponds to the area set and is denoted also as $\mathcal{I} = \{1, \cdots, I\}$. The edges $\{i,j\} \in \mathcal{E}_{k,t}$ correspond to the available communication links used for data exchanges. The adjacency matrix related to the graph is denoted as $\mathbf{A}^k(t) = [A_{i,j}^{(k,t)}]_{I\times I}$

$$\mathbf{A}_{i,j}^{(k,t)} = \begin{cases} 1, & \{i,j\} \in \mathcal{E}_t^k \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

A connected graph $\mathcal{G}_{k,\infty} = \{\mathcal{I}, \cup_{t'=t}^{\infty} \mathcal{E}_{t'}^k\}$ for all $t \geq 0$ within the k^{th} update is defined such that there exists an integer $L \geq 1$ which, for each pair of $\{i,j\}$, satisfies

$$\{i,j\} \in \bigcup_{t'=0}^{L-1} \mathcal{E}_{t'+t}^k. \tag{11}$$

Define the weight matrix $\mathbf{\Phi}^k(t) = [\Phi_{i,j}^k(t)]_{I\times I}$ for network, where $[\Phi_{i,j}^k(t)]$ is non-zero entry of matrix $\mathbf{\Phi}^k(t)$ if and only if $\{i,j\} \in \mathcal{E}_t^k$. To ensure that the exchanges happen between adjacent agents, we require that $\mathbf{\Phi}^k(t)$ is symmetric and doubly stochastic for any k and t. Furthermore, with the $i,j \in \mathcal{I}$, we assume there exists a $0 < \eta < 1$ such that

1) $\Phi_{i,j}^k(t) \geq \eta$ for all $k > 0$ and $t > 0$

2) $\Phi_{i,j}^k(t) \geq \eta$ for all $k > 0$ and $t > 0$ if $\{i,j\} \in \mathcal{E}_t^k$

3) $\Phi_{i,j}^k(t) = \eta$ for all $k > 0$ and $t > 0$ if $\{i,j\} \in \mathcal{E}_t^k$

Gather the local information in one single vector $\mathcal{W}^k(t) \triangleq [\mathcal{W}_1^k(t), \cdots, \mathcal{W}_I^k(t)]$, so that we can write the network exchange explicitly as

$$\mathcal{W}^k(t) = [\mathbf{\Phi}^k(t) \otimes \mathbf{I}_{I_\mathcal{W}}] \mathcal{W}^k(t-1), 1 \leq t \leq t_k, \tag{12}$$

where $\mathbf{I}_{I_\mathcal{W}}$ is the identity matrix and $I_\mathcal{W}$ equals to the length of the local information exchanged at agent i, $\mathcal{W}_i^k(t)$ (in our case, $I_\mathcal{W} = M$ for both the exchanges of state variables and local gradients) and t_k is the number of exchanges during $[\tau_k, \tau_{k+1})$.

In general, the weight matrix $\mathbf{\Phi}^k(t)$ is time-varying. However, we only consider the special case of the general model, i.e. Coordinated Static Exchange [27], [28] in which each agent collects the messages from the neighbourhood, and updates parameters based on a static weight matrix Φ. This network can satisfy the fully connected condition with

$$\mathbf{A} = \mathbf{I}_I - \mathbf{1}_I \mathbf{1}_I^T \tag{13}$$

where $\mathbf{1}_I$ is an I-dimensional all-one vector. In most CSE protocol based gossip network, the weight matrix is constructed by Laplacian matrix

$$\mathbf{L} = \mathbf{D} - \mathbf{A} \tag{14}$$

where $\mathbf{D} = \text{diag}(\mathbf{A}\mathbf{1}_I)$ is the degree matrix and

$$\mathbf{\Phi} = \mathbf{I}_I - \beta\mathbf{L} \tag{15}$$

where $\beta = 1/max(\mathbf{A}\mathbf{1}_I)$.

Lemma 1. *[26, Proposition 1] Let connectivity and stochastic weights assumptions hold. The entries of the matrix product $\prod_{t'=0}^{t} \mathbf{\Phi}^k(t')$ converge to $1/I$ with a geometric rate uniformly with respect to $i, j \in \mathcal{I}$, and k*

$$\Big|\Big[\prod_{t'=0}^{t} \mathbf{\Phi}^k(t')\Big]_{i,j} - \frac{1}{\mathcal{I}}\Big| \leq 2\Big(\frac{1+\eta^{-L_0}}{1-\eta^{L_0}}\Big)(1-\eta^{L_0})^{t/L_0} \tag{16}$$

where $L_0 = (1-I)L$ and L bound the intercommunication interval ensuring graph connectivity.

The limit of the weight matrix product exists

$$\lim_{t\to\infty} = \prod_{t'=0}^{t} \mathbf{\Phi}^k(t') = \mathbf{1}_N\mathbf{1}_N^T/\mathbf{I} \tag{17}$$

and thus

$$\lim_{t\to\infty} \mathcal{W}_i^k(t) = \frac{1}{I}\sum_{i=1}^{I} \mathcal{W}_i^k(0), k = 1, 2, \cdots. \tag{18}$$

B. Local Update Process

To start with, an initial state variable $\mathbf{x}_i^0$ and an initial approximation for the inverse of Hessian matrix $\mathbf{H}_i^0$, need to be set at each agent. For reasons that will be clear later, before any local iteration k, a gossip process is implemented to compute the average state, denoted as

$$\bar{\mathbf{x}_i}^k(l_k) \approx \frac{1}{I}\sum_{i=1}^{I} \mathbf{x}_i^k, \tag{19}$$

where l_k is the number of gossip exchange. We assume that all the agents are synchronized so that the data exchange happens in an synchronous way. The deviation of $\bar{\mathbf{x}_i}^k(l_k)$ from the real average is related to both the states $\mathbf{x}_i^k$ and l_k, and will be discussed with more details in the next section. Now $F_i(\bar{\mathbf{x}_i}^k(l_k))$ can be computed at each agent and the average of the gradients,

$$\hat{F}(\bar{\mathbf{x}_i}^k, l_k^{'}) \approx \frac{1}{I}\sum_{i=1}^{I} F_i(\bar{\mathbf{x}_i}^k) \tag{20}$$

can be similarly obtained by another gossip process, where $l_k^{'}$ is the gossip exchange number. As we later show, the values of l_k and $l_k^{'}$ have varied degrees of influence on the distributed algorithm's convergence property.

After updating (except for the first iteration, i.e. $k = 0$) the approximation for the inverse of Hessian matrix according to

$$\mathbf{H}_i^k = (I - \rho_i^k\mathbf{z}_i^k\mathbf{y}_i^{kT})\mathbf{H}_i^{k-1}(I - \rho_i^k\mathbf{y}_i^k\mathbf{z}_i^{kT}) + \rho_i^k\mathbf{z}_i^k\mathbf{z}_i^{kT}, \tag{21}$$

where $\mathbf{y}_i^k = \hat{F}(\bar{\mathbf{x}_i}^k) - \hat{F}(\bar{\mathbf{x}_i}^{k-1}), \mathbf{z}_i^k = \bar{\mathbf{x}_i}^k - \bar{\mathbf{x}_i}^{k-1}, \rho_i^k = 1/(\mathbf{y}_i^{kT}\mathbf{z}_i^k)$, the following local iteration is then implemented at each agent,

$$\mathbf{x}_i^{k+1} = \bar{\mathbf{x}_i}^k - \alpha_i\mathbf{H}_i^k\hat{F}(\bar{\mathbf{x}_i}^k), i \in \mathcal{I}, \tag{22}$$

where α_i is used to control the size of the searching step. To simplify the analysis, we fix α_i to be 1 at all agents.

From the above description, it can be seen that only state variables and first-order information are required to be exchanged between the nodes, while the second-order information is locally estimated. More importantly, no matrix inverse is required, which reduce the computational burden significantly compared with the Gossip-based Gauss Newton method in [13].

The whole procedure of DQN method is summarized in the Algorithm 1. In the next section, we will provide its convergence analysis.

Algorithm 1 Distributed BFGS Algorithms

1: **given** initial variables $\mathbf{x}_i^0$, $\mathbf{H}_i^0$ at all agents $i \in \mathcal{I}$, as well as proper weight matrix $\mathbf{\Phi}$ that satisfies Assumption 1 and 2.
2: **set** $k = 0$.
3: **repeat**
4: **network exchanges**: Agents exchange their local state variables according to (12) with $t_k = l_k$. After $F_i(\bar{\mathbf{x}_i}^k(l_k))$ is computed at each agent, these local graients are exchanged according to (12) with $t_k = l_k^{'}$ and the estimate of the global gradient $\hat{F}(\bar{\mathbf{x}_i}^k)$ is obtained at each node.
5: **local update**: If $k \geq 1$, update the approximated inverse Hessian matrix as (21). For each $i \in \mathcal{I}$. agent i updates its local variables as (22).
6: **set** $k = k + 1$
7: **until** $\|\mathbf{x}_i^{k+1} - \mathbf{x}_i^k\| \leq \epsilon$ or $k = K$.
8: **set** the local estimate as $\hat{\mathbf{x}}_i = \mathbf{x}_i^k$.

V. Convergence Analysis

In this section, we analyze the convergence of the DQN algorithm (summarized in Algorithm 1). Local convergence instead of global convergence property is studied here since the objective function in our problem formulation is not guaranteed to be convex and for non-convex functions, a global convergence proof is not found even for the centralized version of quasi-Newton algorithm in existing literature. We mainly develop the local convergence analysis first used in [29] and study the impact of the distributed implementations on the DQN's convergence property.

A. Gossip Errors Analysis

One of the noticeable differences of DQN from its centralized version is that the values of state variables and gradients utilized at the iteration equations (21),(22) are deviated from

the real values since the gossip exchange number is finite. We denote such deviations as gossip errors. To facilitate our later analysis of the local convergence, we bound the gossip errors by making some reasonable assumptions on the objective functions.

Lemma 2. *[29] Assume the gradient of the global objective function, $F : \mathbb{R}^M \to \mathbb{R}^M$ is differentiable in the open set $D \subset \mathbb{X}$, and for some minima x^* in D, $p > 0$ and $K > 0$,*

$$\left\| F^{'}(x) - F^{'}(x^*) \right\| \leq K \|x - x^*\|^p. \tag{23}$$

The following inequality is satisfied for every u, v in D,

$$\left\| F(v) - F(u) - F^{'}(x^*)(v-u) \right\| \tag{24}$$

$$\leq K max\{\|v - x^*\|^p, \|u - x^*\|^p\}\|u - v\|. \tag{25}$$

If $F^{'}(x^*)$ is further invertible, there exist $\epsilon > 0$ and $\rho > 0$ such that $max\{\|u - x^*\|, \|v - x^*\|\} \leq \epsilon$ leads to $u, v \in D$ and

$$(1/\rho)\|v - u\| \leq \|F(v) - F(u)\| \leq \rho\|v - u\|. \tag{26}$$

Denote the gossip errors for the local state and the gradient exchanges respectively as $p_i(l_k), q_i(l_k^{'})$ at agent i and the k^{th} iteration. To ease the expression, we use $\hat{F}_i^k(t)$ in place of $\hat{F}(\bar{\mathbf{x}_i}^k, t)$ hereafter. Define $\mathcal{W}_F^k(t) = \left[\hat{F}_1^k(t), \cdots, \hat{F}_I^k(t)\right]$, $\mathcal{W}_x^k(t) = \left[\bar{\mathbf{x}_1}^k(t), \cdots, \bar{\mathbf{x}_I}^k(t)\right], \bar{\mathcal{W}}_F^k = [\mathbf{1}_I \mathbf{1}_I^T \otimes \mathbf{I}_{I_M}]\mathcal{W}_F^k(0)/I$ and $\bar{\mathcal{W}}_x^k = [\mathbf{1}_I \mathbf{1}_I^T \otimes \mathbf{I}_{I_M}]\mathcal{W}_x^k(0)/I$. Note that $\mathcal{W}_x^k(0) = \left[\mathbf{x}_1^k, \cdots, \mathbf{x}_I^k\right], \mathcal{W}_F^k(0) = \left[F_1(\bar{\mathbf{x}_1}^k), \cdots, F_I(\bar{\mathbf{x}_I}^k)\right]$. By the above definitions, we would have

$$\mathcal{W}_F^k(t) - \bar{\mathcal{W}}_F^k = \begin{bmatrix} q_1(t) \\ \vdots \\ q_I(t) \end{bmatrix}, \mathcal{W}_x^k(t) - \bar{\mathcal{W}}_x^k = \begin{bmatrix} p_1(t) \\ \vdots \\ p_I(t) \end{bmatrix}. \tag{27}$$

Lemma 3. *Let the assumptions made in Lemma 2 hold and the gradients of the local functions are upper bounded in D. The following inequalities are then satisfied for $i \in \mathcal{I}$,*

$$\|p_i(t)\| = \left\| \bar{\mathbf{x}_i}^k(t) - \sum_{j=1}^{I} \mathbf{x}_i^k \right\| \leq C_1(t_1), 0 < t_1 < l_k, \tag{28}$$

$$\left\| \hat{F}_i^k(t) - F(\bar{\mathbf{x}_i}^k) \right\| \leq 2C_1(l_k) + C_2(t_2), 0 < t_2 < l_k^{'}, \tag{29}$$

where C_1 and C_2 are both positive reals that decrease exponentially with the number of the gossip exchanges.

Proof. Please see Appendix A. □

B. Local convergence analysis

One prominent feature of quasi-Newton algorithm that differentiate it from other unconstrained optimization algorithms is that its second-order information is updated recursively. To analyze the local convergence property, we first characterize this recursive process by establishing the following lemma.

Lemma 4. *Let the gradient of the global function, F, satisfy the assumptions made in lemma 3 and further let the hessian matrix at the minima, $F^{'}(\mathbf{x}^*)$, be symmetric and positive definite. Then there exists an neighborhood $\mathcal{N} = \mathcal{N}_1 \times \mathcal{N}_2$ of $(\mathbf{x}^*, F^{'}(\mathbf{x}^*)^{-1})$ such that for each $(\bar{\mathbf{x}}^{k-1}, \mathbf{H}^{k-1}) \in \mathcal{N}$, the updated Hessian inverse $\mathbf{H}^k$, as defined in (21), satisfies*

$$\left\| \mathbf{H}^k - F^{'}(\mathbf{x}^*) \right\|_M \leq \left[1 + \lambda_1 \max\left\{\|\bar{\mathbf{x}}^k - \mathbf{x}^*\|^p, \|\bar{\mathbf{x}}^{k-1} - \mathbf{x}^*\|^p\right\}\right]$$

$$\left\| \mathbf{H}^{k-1} - F^{'}(\mathbf{x}^*) \right\|_M + \lambda_2 \max\left\{\|\bar{\mathbf{x}}^k - \mathbf{x}^*\|^p, \|\bar{\mathbf{x}}^{k-1} - \mathbf{x}^*\|^p\right\} \tag{30}$$

where λ_1, λ_2 are non-negative constants, and $\|\cdot\|_M$ is certain Matrix norm. Note that we omit i in $\bar{\mathbf{x}_i}^k$ and $\mathbf{H}_i^k$ for ease of expression.

Proof. Please see Appendix B □

Now that the inverse of the hessian matrix is bounded through inequality (30), the local convergence result can be well established. We conclude the main result in the following theorem.

Theorem 1. *Let the assumptions made in Lemma 4 be satisfied by the gradient function, F. Then there exists a neighborhood $\mathcal{N} = \mathcal{N}_1 \times \mathcal{N}_2$ of $(\mathbf{x}^*, F^{'}(\mathbf{x}^*)^{-1})$, such that for each $r \in (0, 1)$, if the initial states satisfy the following condition*

$$\|\mathbf{x}_i^0\| < \epsilon(r)/2, \|\mathbf{H}_i^0 - F^{'}(\mathbf{x}^*)\|_M < \delta(r), \tag{31}$$

where $\epsilon(r), \delta(r)$ are positive constants, then the sequences of $\bar{\mathbf{x}_i}^k, \mathbf{H}_i^k, k > 0$ are well defined in $\mathcal{N}$ and $\bar{\mathbf{x}_i}^k$ converges to the local minimum $\mathbf{x}^*$ in the following manner

$$\|\bar{\mathbf{x}_i}^{k+1} - \mathbf{x}^*\| \leq r^{'}\|\bar{\mathbf{x}_i}^k - \mathbf{x}^*\|, \tag{32}$$

where $r^{'} \in (0, 1)$.

Proof. Please see Appendix C. □

VI. Simulation Results

In this section, we conduct experiments to compare the existing distributed quasi-Newton algorithm performance to those of the GGN algorithm [13] and ADMM algorithm [9]. The distributed estimate in each are $\{\hat{V}_{i,n}^k\}_{n=1}^N, \{\hat{\theta}_{i,n}^k\}_{n=1}^N$ at each local update, the Mean Squared Error (MSE) with respect to the voltage magnitude and the phase at the i-th site are

$$\mathbf{MSE}_{V,i}^k = \sum_{n=1}^{N}((\hat{V}_{i,n}^k) - \bar{V}_n)^2 \tag{33}$$

$$\mathbf{MSE}_{\theta,i}^k = \sum_{n=1}^{N}((\hat{\theta}_{i,n}^k) - \bar{\theta}_n)^2 \tag{34}$$

In addition, the metric used in our comparisons are the cost function in (7),

$$\mathbf{Val}_k = \sum_{i=1}^{I} \|\mathbf{z}_i - \mathbf{W}_i \mathbf{x}_i^k\|^2 \tag{35}$$

evaluated using the decentralized estimates at each updates, and the global MSE is given by

$$\mathbf{MSE}_V^k = \frac{1}{I}\sum_{i=1}^{I} \mathbf{MSE}_{V,i}^k \tag{36}$$

TABLE I: Execution Time and Iterations in Case A

IEEE 14-bus	GGN	DQN	Centralized Estim
Computation times (s)	0.0287	0.0236	0.0228
Iterations	48	122	Undefined

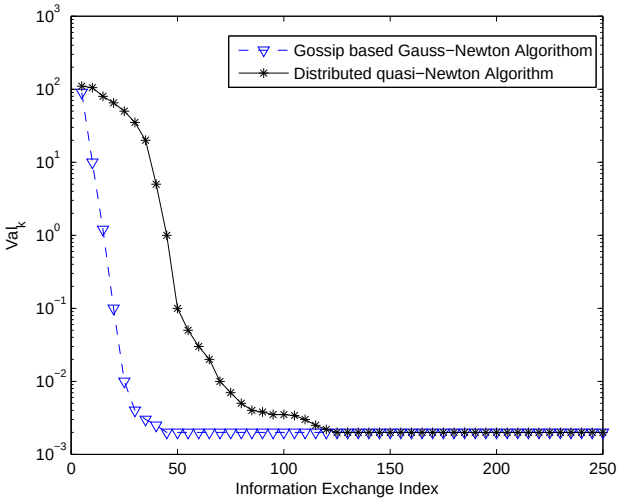
$$\mathbf{MSE}_{\theta}^{k} = \frac{1}{I}\sum_{i=1}^{I}\mathbf{MSE}_{\theta,i}^{k} \tag{37}$$

In the simulations we used MATPOWER 5.1 [22] test case IEEE-14 (N=14) system, and the load form is taken from Power Systems Test Case Archive, University of Washington [21], and scale the base load from MATPOWER upon load buses, and select the work program as Optimal Power Flow to give the generation dispatch for that instant. The initialization for the voltage magnitudes and phases are 1 and 0, respectively.

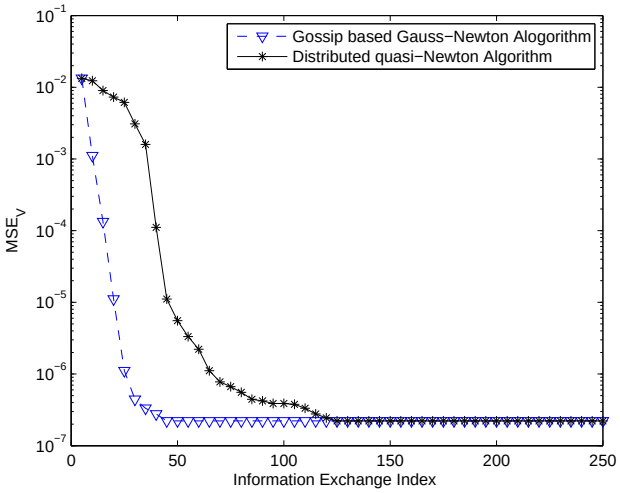
Sensor observations are generated by introducing independent Gaussian errors $\{\mathbf{e}\} \sim \mathcal{N}(0, \sigma^2)$ where $\sigma^2 = 10^{-6}$. The IEEE 14-bus grid is partitioned into 4 areas depicted in Fig. 1. The control areas contain $I_1 = 3$, $I_2 = 4$, $I_3 = 4$ and $I_4 = 4$ buses, respectively.

A. Case A: Comparison with GGN without Bad Data

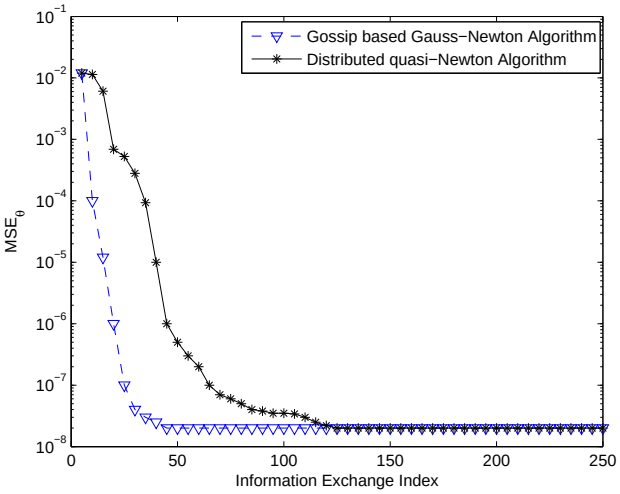
Here we present how the distributed quasi-Newton scheme performs against the existing Gossip based Gauss Newton algorithm for PSSE in [13]. These distributed network algorithms proceed at each tth gossip exchange, and run the them with $t = 10$ gossip exchanges for each update. The comparison is made on the same time scale based on the number of exchanges. By using the $t = 10$ gossip exchanges between every two descent updates $k = 1, \cdots, 50$, thus we have the total number of 500 exchanges per snapshot. We assume that all sensors are connected, which leads to the adjacency matrix $\mathbf{A} = \mathbf{I}_I - \mathbf{1}_I\mathbf{1}_I^T$, and the weight matrix is constructed with the Laplacian $\mathbf{L} = \text{diag}(\mathbf{A}\mathbf{1}_I) - \mathbf{A}$ and $\Phi = \mathbf{I}_I - \omega\mathbf{L}$ with $\omega = \beta/\max(\mathbf{A}\mathbf{1}_I)$ where $\beta = 0.5$. We choose the step-size for Gossip based Gauss Newton algorithm as $\alpha_{GGN} = 0.5$. It can be seen from Fig. 2(a-c), GGN algorithms converge faster than the proposed DQN method, because GGN algorithm can achieve the convergence rate of centralized Gauss-Newton Algorithm, which converge quadratically when the system error or residual is very small. On the other hand, distributed quasi-Newton is Newton-like algorithm that converges superlinearly. However, from the comparison in Table 1, GGN method require to compute the inversion of Hessian matrix with complexity order of $\mathcal{O}(N^3)$, where N is the matrix size. This results in high computation complexity and requirements of the local processor to have capability to maintain such computations on time for exchange. In contrast, the proposed method requires an $\mathcal{O}(N^2)$ computation cost. It uses an iterative solution of approximation for the Hessian matrix and avoids calculating matrix inverse , which makes it more effective and realistic in a power system.



(a) Cost Function $\mathbf{Val}_k$



(b) $\mathbf{MSE}_V^k$



(c) $\mathbf{MSE}_{\theta}^k$

Fig. 2: Comparison with GGN and distributed quasi-Newton using $t = 10$ exchange for each update

B. Case B: Comparison with GGN in presence of Bad Data

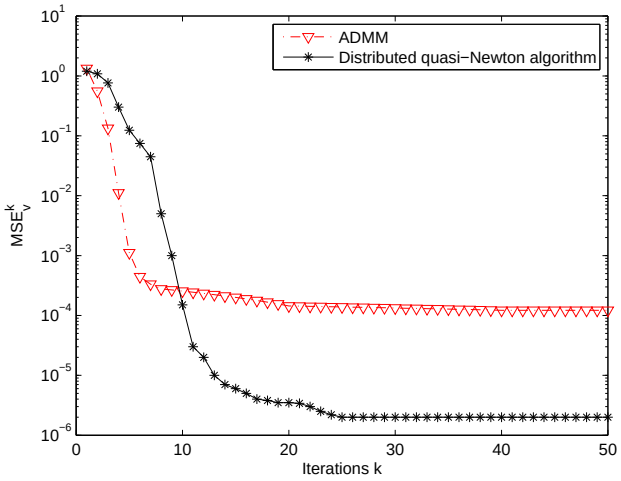
We compare our proposed method with GGN algorithm when bad data is present. We added random Gaussian system errors $\mathbf{e}_s$ with $E(\mathbf{e}_s\mathbf{e}_s^T) = 100\sigma^2$. We examine the MSE performance of the distributed quasi-Newton method where, in each snapshot t, each agent exchange to neighbour agents 10 times on average during the interval$[\tau_k, \tau_{k+1})$ for all $k = 1, \cdots, 50$. Clearly, as shown in Fig. 2(a) and (b), when large residual is present, caused by bad data, estimation with the GGN method fail to improve the cost function after iteration $k = 11$ in each snapshot. On the other hand this distributed quasi-Newton method degrades more gracefully. The GGN method only considers the first order term of Hessian matrix, however, for the large residual problem, second order terms cannot be neglected. By contrast, the distributed quasi-Newton method can build up the second-order derivative term for approximated Hessian with iterative process. That is the reason for our method which outperforms significantly the GGN algorithm in the presence of bad data.

C. Case C: Comparison with ADMM Method in a Large-scale Power Network

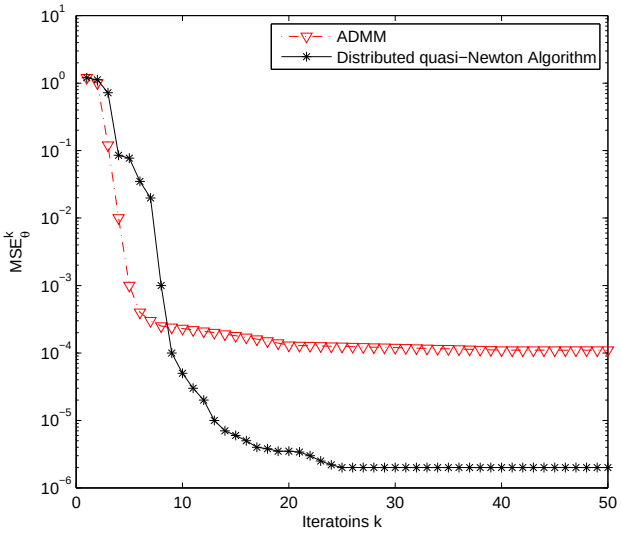
We finally compare our method to ADMM [9] using a larger power network: a 4200-bus power grid constructed using the IEEE 14- and 300-bus power grid. By assuming that 300 buses are different regions, a copy of the IEEE 14-bus grid can be used as the substitute for each of them. Moreover, we randomly choose the the terminal buses among the incident to the line areas for the IEEE 300-bus grid. Measurements and bad data are selected as the tests for IEEE 14-bus grid. The step-size for ADMM is $\alpha_{ADMM} = 0.5$. Fig. 3(a) and (b) demonstrate the MSE plot which are averaged upon 300 areas. Observing that distributed quasi-Newton method converges substantially faster than ADMM methods, achieving a Mean-square error of 10^{-6} less than 25 iteration, while ADMM just reaches MSE of 10^{-3} by iteration 40. Note that the IEEE 300-bus is used as the substitute of the agents in the IEEE 14-bus grid. This reserved topology of the 14 agents is also tested. It can be seen from the Fig. 3(c) that the algorithm converged a slightly faster (around 5%) due to the looser areas coupling.

VII. Conclusion

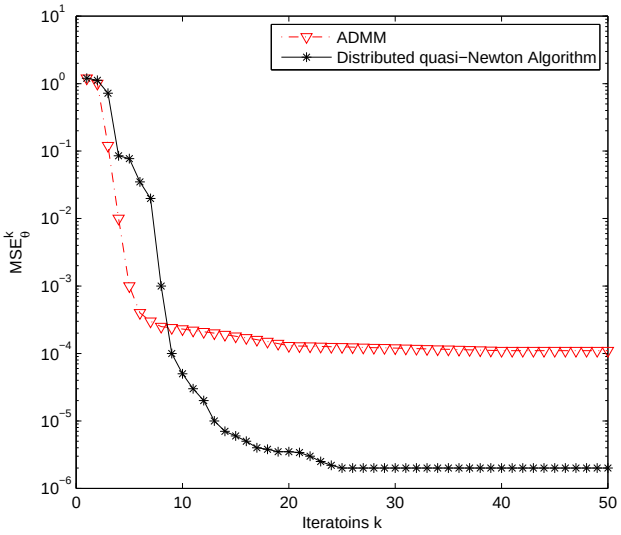
In this paper, we proposed a distributed quasi-Newton for hybrid power system state estimation integrating seamlessly WAMS and SCADA measurement system, which adaptively estimated the global state vector along with a large residual. The proposed algorithm reduced the complexity of computation and maintained the property of fast convergence. The numerical results proved that the proposed approach was capable of delivering accurate estimates of the entire state vector at each distributed area, even in the presence of bad data. Meanwhile, its effectiveness was demonstrated by applying this method to a large-scale power system network.



(a) $\mathbf{MSE}_V^k$



(b) $\mathbf{MSE}_\theta^k$



(c) $\mathbf{MSE}_\theta^k$

Fig. 3: Comparison distributed quasi-Newton against ADMM in a large-scale power network

APPENDIX A
PROOF OF LEMMA 3

According to Lemma 1, we can obtain

$$\mathcal{W}_x^k(t) - \bar{\mathcal{W}}_x^k \tag{A.1}$$

$$= \left[\left(\prod_{t'}^{t} \mathbf{\Phi}_k(t') - \frac{\mathbf{1}_I\mathbf{1}_I}{I}\right) \otimes \mathbf{I}_{I_M}\right] \mathcal{W}_x^k(0), \tag{A.2}$$

the norm of which is bounded as

$$\left\|\mathcal{W}_x^k(t) - \bar{\mathcal{W}}_x^k\right\| \leq \left\|\prod_{t'=0}^{t} \mathbf{\Phi}_k(t') - \frac{\mathbf{1}_I\mathbf{1}_I}{I}\right\| \left\|\mathcal{W}_x^k(0)\right\| \tag{A.3}$$

$$\leq \left\|\prod_{t'=0}^{t} \mathbf{\Phi}_k(t') - \frac{\mathbf{1}_I\mathbf{1}_I}{I}\right\|_F \left\|\mathcal{W}_x^k(0)\right\| \tag{A.4}$$

$$\leq \left[2I\left(\frac{1+\eta^{-L_0}}{1-\eta^{-L_0}}\right)\lambda_\eta^t\right] \left\|\mathcal{W}_x^k(0)\right\|, \tag{A.5}$$

where $\lambda_\eta = (1-\eta^{L_0})^{1/L_0}$ and (A.4) is due to $\|\cdot\| \leq \|\cdot\|_F$. Since x_i lies in D, there exists positive real $C_D > 0$ such that $\left\|\mathcal{W}_x^k(0)\right\|^2 = \sum_{i=1}^{I} \|x_i^k\|^2 < C_D^2$ and we can further obtain that

$$\|p_i(t)\| \leq \left\|\mathcal{W}_x^k(t) - \bar{\mathcal{W}}_x^k\right\| \leq C_1(t), \tag{A.6}$$

where $C_1(t) = \left[2I\left(\frac{1+\eta^{-L_0}}{1-\eta^{-L_0}}\right)\lambda_\eta^t\right] C_D$. It can be seen that the norm of $p_i(t)$ decreases exponentially with t. Since the gradients of F_i are upper bounded in D, there exists positive real $C_G > 0$ such that

$$\left\|\mathcal{W}_F^k(0)\right\|^2 = \sum_{i=1}^{I} \left\|F_i(\bar{x}_i^k)\right\|^2 < C_G. \tag{A.7}$$

Therefore, similarly to the derivation of (A.6), we can obtain

$$\|q_i(t)\| \leq \left\|\mathcal{W}_F^k(t) - \bar{\mathcal{W}}_F^k\right\| \leq C_2(t), \tag{A.8}$$

where $C_2(t) = \left[2I\left(\frac{1+\eta^{-L_0}}{1-\eta^{-L_0}}\right)\lambda_\eta^t\right] C_G$. It can be then obtained that

$$\left\|\hat{F}_i^k(t) - F(\bar{\mathbf{x}}_i{}^k)\right\| = \left\|\sum_{j=1}^{I}\left[F_j(\mathbf{x}_j{}^k) - F_j(\bar{\mathbf{x}}_i{}^k)\right] + q_i(t)\right\| \tag{A.9}$$

$$\leq \sum_{j=1}^{I} \rho\|\mathbf{x}_j{}^k - \bar{\mathbf{x}}_i{}^k\| + \|q_i(t)\|, \tag{A.10}$$

where (A.10) is from Lemma 2. From (A.6), it can be derived that

$$\|\mathbf{x}_j{}^k - \bar{\mathbf{x}}_i{}^k\| \leq \|p_i(l_k)\| + \|p_j(l_k)\| \leq 2C_1(l_k). \tag{A.11}$$

Finally, from (A.8),(A.10) and(A.11), we have (29).

APPENDIX B
PROOF OF LEMMA 4

In the following analysis, we let ϵ, ρ be the corresponding parameters in Lemma 2, i.e. $\max\{\|u - x^*\|, \|v - x^*\|\} \leq \epsilon$ would lead to $u, v \in D$ and inequality (26). Define $\mathcal{N}_2$ as

$$\mathcal{N}_2 = \left\{\mathbf{H} \in L(\mathbb{R}^n) | \|F'(\mathbf{x}^*)\| \|\mathbf{H} - F'(\mathbf{x}^*)^{-1}\| < 1/2\right\}. \tag{B.12}$$

To start with, we prove that the norm of $\mathbf{y}_i^k$ is upper bounded by a constant. For any $\mathbf{H} \in \mathcal{N}_2$, we have that $\mathbf{H}$ is non-singular and there exists a positive real c, s.t. $\|\mathbf{H}\| \leq c$. If $\bar{\mathbf{x}}^{k-1} \in D$ is satisfied and further define $\|\mathbf{s}_i^k\| = \|\mathbf{x}_i^k - \bar{\mathbf{x}}_i{}^{k-1}\|$, then

$$\|\mathbf{s}_i^k\| \leq \|\mathbf{H}\|^{k-1}\hat{F}_i^{k-1}\| \tag{B.13}$$

$$= \|\mathbf{H}\|^{k-1}\left\|F(\bar{\mathbf{x}}_i{}^{k-1}) - F(\mathbf{x}^*) + \hat{F}_i^{k-1} - F(\bar{\mathbf{x}}_i{}^{k-1})\right\| \tag{B.14}$$

$$\leq c\left[\rho\|\bar{\mathbf{x}}_i{}^{k-1} - \mathbf{x}^*\| + \left\|\hat{F}_i^{k-1} - F(\bar{\mathbf{x}}_i{}^{k-1})\right\|\right], \tag{B.15}$$

where (B.15) is due to Lemma 2. Then we can bound the state (after state averaging) of the k^{th} iteration by

$$\|\bar{\mathbf{x}}_i{}^k - \mathbf{x}^*\| = \left\|p_i(l_k) + \mathbf{x}_i^k - \bar{\mathbf{x}}_i{}^{k-1} + \bar{\mathbf{x}}_i{}^{k-1} - \mathbf{x}^*\right\| \tag{B.16}$$

$$\leq \|p_i(l_k)\| + \|\mathbf{s}_i^k\| + \|\bar{\mathbf{x}}_i{}^{k-1} - \mathbf{x}^*\| \tag{B.17}$$

$$\leq (c\rho+1)\|\bar{\mathbf{x}}_i{}^{k-1} - \mathbf{x}^*\| + \|p_i(l_k)\| + c\left\|\hat{F}_i^{k-1} - F(\bar{\mathbf{x}}_i{}^{k-1})\right\| \tag{B.18}$$

Now we define $\mathcal{N}_1$ as

$$\forall \mathbf{x}_i^{k-1} \in \mathcal{N}_1, \|\bar{\mathbf{x}}_i{}^{k-1} - \mathbf{x}^*\| \leq \min\left\{\frac{\epsilon}{2(1+c\rho)}, \frac{\epsilon}{2}\right\}, \tag{B.19}$$

where $\mu_2(2\rho\epsilon)^p < 1/3$. According to Lemma 3, by choosing $l_{k-1}, l_{k-1}^{'}$ such that

$$c\left(2C_1(l_{k-1}) + C_2(l_{k-1}^{'})\right) + C_1(l_{k-1}) < \epsilon/2, \tag{B.20}$$

it can derived that $\|\bar{\mathbf{x}}_i{}^k - \mathbf{x}^*\| < \epsilon$ or $\bar{\mathbf{x}}_i{}^k \in D$. Now that both $\bar{\mathbf{x}}_i{}^k, \bar{\mathbf{x}}_i{}^{k-1} \in D$, by Lemma 2, we can derive that

$$1/\rho\|\mathbf{z}_i^k\| \leq \|F(\bar{\mathbf{x}}_i{}^k) - F(\bar{\mathbf{x}}_i{}^{k-1})\| \leq \rho\|\mathbf{z}_i^k\|, \tag{B.21}$$

which is related to the term that we are trying to bound as

$$\mathbf{y}_i^k = \|F(\bar{\mathbf{x}}_i{}^k) - F(\bar{\mathbf{x}}_i{}^{k-1}) + \tag{B.22}$$

$$(\hat{F}_i^k - F(\bar{\mathbf{x}}_i{}^k)) - (\hat{F}_i^{k-1} - F(\bar{\mathbf{x}}_i{}^{k-1}))\|. \tag{B.23}$$

Again, using Lemma 3, we can bound the last two terms in (B.22) by choosing appropriate $l_k, l_{k-1}, l_k^{'}$ and $l_{k-1}^{'}$ such that

$$1/(2\rho)\|\mathbf{z}_i^k\| \leq \|\mathbf{y}_i^k\| \leq 2\rho\|\mathbf{z}_i^k\| \tag{B.24}$$

Next, we prove that $\|\mathbf{y}_i^k\|$ is also lower bounded as

$$\frac{\|\mathbf{M}\mathbf{z}_i^k - \mathbf{M}^{-1}\mathbf{y}_i^k\|}{\|\mathbf{M}^{-1}\mathbf{y}_i^k\|} \leq \mu_2\|\mathbf{y}_i^k\|^p, \tag{B.25}$$

for some constants $p > 0, \mu_2 > 0$ and symmetric, non-singular $\mathbf{M}$. To see this, first since $F'(\mathbf{x}^*)$ is symmetric and positive definite, there exists a positive symmetric $\mathbf{M}$ s.t. $F'(\mathbf{x}^*) = \mathbf{M}^2$. We could then write

$$\mathbf{M}^{-1}\mathbf{y}_i^k - \mathbf{M}\mathbf{z}_i^k = \mathbf{M}^{-1}[\mathbf{y} - F'(\mathbf{x}^*)\mathbf{z}_i^k], \tag{B.26}$$

which by Lemma 2, is equivalent to

$$\frac{\|\mathbf{M}\mathbf{z}_i^k - \mathbf{M}^{-1}\mathbf{y}_i^k\|}{\|\mathbf{M}^{-1}\mathbf{y}_i^k\|} \leq \mu_0 \max\{\|\bar{\mathbf{x}}_i{}^k - \mathbf{x}^*\|^p, \|\bar{\mathbf{x}}_i{}^{k-1} - \mathbf{x}^*\|^p\}. \tag{B.27}$$

Since $\mathbf{H}^{k-1}$ is in a neighborhood of $F'(\mathbf{x})^{-1}$, i.e. $\mathcal{N}_2$, by the

Banach Perturbation Lemma we can bound the operator norm of $\mathbf{H}^{k-1}$ as

$$\|\mathbf{H}^{k-1}\| \leq 2\|F^{'}(\mathbf{x}^*)^{-1}\|. \tag{B.28}$$

By Lemma 2, it can then be derived that

$$1/\rho\|\bar{\mathbf{x}_i}^k - \mathbf{x}^*\| \leq \|\mathbf{H}^{k-1}\|^{-1}\|\mathbf{z}_i^k\|, \tag{B.29}$$

which combined with (B.28) indicates that there exists $\lambda > 0$ s.t.

$$\max\{\|\bar{\mathbf{x}_i}^k - \mathbf{x}^*\|^p, \|\bar{\mathbf{x}_i}^{k-1} - \mathbf{x}^*\|^p\} \leq \lambda\|\mathbf{z}_i^k\|^p. \tag{B.30}$$

It is easy to see that due to Lemma 2, (B.27) combined with (B.30) is equivalent to (B.25). From (B.25) and (B.24), we can finally derive that

$$\frac{\|\mathbf{M}\mathbf{z}_i^k - \mathbf{M}^{-1}\mathbf{y}_i^k\|}{\|\mathbf{M}^{-1}\mathbf{y}_i^k\|} \leq \mu_2\|\mathbf{y}_i^k\|^p \leq \mu_2(\rho\epsilon)^p \leq 1/3, \tag{B.31}$$

which enables us to use Lemma 5.2 in [29] to derive the following inequality

$$\left\|\mathbf{H}^k - F^{'}(\mathbf{x}^*)^{-1}\right\|_M \leq \left[1 + \lambda_1^{'}\|\mathbf{y}\|^p\right]\left\|\mathbf{H}^{k-1} - F^{'}(\mathbf{x}^*)^{-1}\right\|_M + \lambda_2^{'}\frac{\|\mathbf{z}_i^k - F^{'}(\mathbf{x}^*)^{-1}\mathbf{y}_i^k\|}{\|\mathbf{M}^{-1}\mathbf{y}_i^k\|}, \tag{B.32}$$

where λ_1, λ_2 are positive constants. Rewrite that

$$\|\mathbf{z}_i^k - F^{'}(\mathbf{x}^*)^{-1}\mathbf{y}_i^k\| = \|F^{'}(\mathbf{x}^*)^{-1}\|\|F^{'}(\mathbf{x}^*)\mathbf{z}_i^k - \mathbf{y}_i^k\|. \tag{B.33}$$

Since $\bar{\mathbf{x}_i}^k, \bar{\mathbf{x}_i}^k \in D$ and according to Lemma 2, it can be derived that

$$\|F^{'}(\mathbf{x}^*)\mathbf{z}_i^k - \mathbf{y}_i^k\| = \|F^{'}(\mathbf{x}^*)\mathbf{z}_i^k - \left(F(\bar{\mathbf{x}_i}^k) - F(\bar{\mathbf{x}_i}^{k-1})\right) - \left(\hat{F}_i^k - F(\bar{\mathbf{x}_i}^k)\right) + \left(\hat{F}_i^{k-1} - F(\bar{\mathbf{x}_i}^{k-1})\right)\| \tag{B.34}$$

$$\leq K\max\left\{\|\bar{\mathbf{x}_i}^k - \mathbf{x}^*\|, \|\bar{\mathbf{x}_i}^{k-1} - \mathbf{x}^*\|\right\}\|\mathbf{z}_i^k\| + \left\|\hat{F}_i^k - F(\bar{\mathbf{x}_i}^k)\right\| + \left\|\hat{F}_i^{k-1} - F(\bar{\mathbf{x}_i}^{k-1})\right\| \tag{B.35}$$

$$\leq (K + K_q)\max\left\{\|\bar{\mathbf{x}_i}^k - \mathbf{x}^*\|, \|\bar{\mathbf{x}_i}^{k-1} - \mathbf{x}^*\|\right\}\|\mathbf{z}_i^k\| \tag{B.36}$$

where (B.36) is derived by choosing sufficiently large iteration number so that the last two terms on the right hand side of (B.35) are bounded according to Lemma 3. Moreover, by using (B.24), it can be obtained that

$$\|\mathbf{y}_i^k\| \leq 2\rho \leq 2\rho\max\left\{\|\bar{\mathbf{x}_i}^k - \mathbf{x}^*\|, \|\bar{\mathbf{x}_i}^{k-1} - \mathbf{x}^*\|\right\}\|. \tag{B.37}$$

Finally, by combining (B.24), (B.32), (B.36) and (B.37), we can finally prove that inequality (30) is satisfied under the aforementioned assumptions.

APPENDIX C
PROOF OF THEOREM 1

We set the neighborhood $\mathcal{N}$ as the one that satisfies the requirements in Lemma 4, i.e. for each $(\bar{\mathbf{x}}^{k-1}, \mathbf{H}^{k-1}) \in \mathcal{N}$, inequality (30) is satisfied. Then we choose $\epsilon(r), \delta(r)$ such that $\|\mathbf{x} - \mathbf{x}^*\|$ and $\|\mathbf{H} - F^{'}(\mathbf{x}^*)^{-1}\|_M < \delta$ would imply that $(\mathbf{x}, \mathbf{H}) \in \mathcal{N}$.

First, according to Lemma 3, by choosing sufficiently large l_0 such that $\|p_i(l_0)\| \leq \epsilon$, it can be derived that

$$\|\bar{\mathbf{x}_i}^0 - \mathbf{x}^*\| = \|\mathbf{x}_i^0 + p_i(l_0) - \mathbf{x}^*\| \leq \epsilon/2 + \|p_i(l_0)\| \leq \epsilon, \tag{C.38}$$

which leads to that $(\bar{\mathbf{x}_i}^0, \mathbf{H_i^0}) \in \mathcal{N}$. Since

$$\mathbf{x}_i^1 = \bar{\mathbf{x}_i}^0 - \mathbf{H_i^0}\hat{F}_i^0, \tag{C.39}$$

we can write that

$$\mathbf{x}_i^1 - \mathbf{x}^* = -\mathbf{H}_i^0\left[F(\bar{\mathbf{x}_i}^0) - F(\mathbf{x}^*) - F^{'}(\mathbf{x}^*)(\bar{\mathbf{x}_i}^0 - \mathbf{x}^*) + \hat{F}_i^0 - F(\bar{\mathbf{x}_i}^0)\right] + \left[\mathbf{I} - \mathbf{H}_i^0 F^{'}(\mathbf{x}^*)\right]. \tag{C.40}$$

Since $\mathcal{N}_1 \subset D$ (as shown in the proof of Lemma 4), according to Lemma 3, it can be derived that

$$\left\|F(\bar{\mathbf{x}_i}^0) - F(\mathbf{x}^*) - F^{'}(\mathbf{x}^*)(\bar{\mathbf{x}_i}^0 - \mathbf{x}^*)\right\| \leq K\|\bar{\mathbf{x}_i}^0 - \mathbf{x}^*\|^p\|\bar{\mathbf{x}_i}^0 - \mathbf{x}^*\| \leq K\epsilon^p\|\bar{\mathbf{x}_i}^0 - \mathbf{x}^*\|. \tag{C.41}$$

By the equivalence of all norms that deal with a finite-dimensional space, there exists a constant α, s.t. $\|\mathbf{A}\| \leq \alpha\|\mathbf{A}\|_M$. Therefore, from $\|\mathbf{H}_i^0 - F^{'}(\mathbf{x}^*)\|_M < \delta$, we derive that

$$\left\|\mathbf{H}_i^0 - F^{'}(\mathbf{x}^*)^{-1}\right\| < \alpha\delta. \tag{C.42}$$

Further we assume $\sigma \geq \|F^{'}(\mathbf{x}^*)\|, \gamma \geq F^{'}(\mathbf{x}^*)^{-1}$. Then we can write

$$\left\|\mathbf{I} - \mathbf{H}_i^0 F^{'}(\mathbf{x}^*)\right\| = \left\|F^{'}(\mathbf{x}^*)^{-1} - \mathbf{H}_i^0\right\|\left\|F^{'}(\mathbf{x}^*)\right\| \leq 2\alpha\delta\sigma. \tag{C.43}$$

Combining (C.40), (C.41) and (C.43), it can be derived that

$$\|\mathbf{x}_i^1 - \mathbf{x}^*\| \leq \left[\|\mathbf{H}_i^0\|K\epsilon^p + 2\alpha\delta\sigma\right]\|\bar{\mathbf{x}_i}^0 - \mathbf{x}^*\| + \|\mathbf{H}_i^0\|\left\|\hat{F}_i^0 - F(\bar{\mathbf{x}_i}^0)\right\|. \tag{C.44}$$

Further we bound the vector norm of $\mathbf{H}_i^0$ by

$$\|\mathbf{H}_i^0\| \leq \|\mathbf{H}_i^0 - F^{'}(\mathbf{x}^*)^{-1}\| \leq 2\alpha\delta + \gamma. \tag{C.45}$$

Let ϵ, δ be sufficiently small, such that

$$(2\alpha\delta + \gamma)K\epsilon^p + 2\sigma\delta\alpha \leq r. \tag{C.46}$$

Then we can have

$$\|\mathbf{x}_i^1 - \mathbf{x}^*\| \leq r\|\bar{\mathbf{x}_i}^0 - \mathbf{x}^*\| + (2\alpha\delta + \gamma)\left\|\hat{F}_i^0 - F(\bar{\mathbf{x}_i}^0)\right\|. \tag{C.47}$$

Choose $l_0, l_0^{'}$ such that

$$\left\|\hat{F}_i^0 - F(\bar{\mathbf{x}_i}^0)\right\| \leq \frac{1 - \gamma}{\eta(2\alpha\delta + \gamma)}\|\bar{\mathbf{x}_i}^0 - \mathbf{x}^*\|, \eta > 1. \tag{C.48}$$

We can derive that

$$\|\mathbf{x}_i^1 - \mathbf{x}^*\| \leq \|\bar{\mathbf{x}_i}^0 - \mathbf{x}^*\|, \tag{C.49}$$

where $\hat{r} = (r + (1 - r)/\eta) \in (0, 1)$. Using Lemma 3, with sufficiently large l_1, we can further have the following bound

$$\|\bar{\mathbf{x}_i}^1 - \mathbf{x}^*\| \leq r^{'}\|\bar{\mathbf{x}_i}^0 - \mathbf{x}^*\|, r^{'} \in (0, 1), \tag{C.50}$$

which indicates that $\bar{\mathbf{x}_i}^1 \in \mathcal{N}_1$.

Next, we start an induction argument. First, for $k =$

$0, \cdots, m-1$, we assume that

$$\left\| \mathbf{H}_i^k - F^{'}(\mathbf{x}^*)^{-1} \right\|_M \leq 2\delta, \tag{C.51}$$

$$\|\bar{\mathbf{x}_i}^{k+1} - \mathbf{x}^*\| \leq r^{'} \|\bar{\mathbf{x}_i}^{k} - \mathbf{x}^*\|. \tag{C.52}$$

Since $(\mathbf{x}_i^k, \mathbf{H}_i^k) \in \mathcal{N}$, by (30), it can be derived that

$$\left\| \mathbf{H}_i^{k+1} - F^{'}(\mathbf{x}^*)^{-1} \right\|_M - \left\| \mathbf{H}_i^{k} - F^{'}(\mathbf{x}^*)^{-1} \right\|_M \tag{C.53}$$

$$\leq 2\lambda_1 \sigma \epsilon^p r^{'kp} + \lambda_2 \epsilon^p r^{'kp}. \tag{C.54}$$

By summing the two sides of inequality (C.53) for $k = 0, \cdots, m-1$, we obtain

$$\left\| \mathbf{H}_i^{m} - F^{'}(\mathbf{x}^*)^{-1} \right\|_M \leq \left\| \mathbf{H}_i^{0} - F^{'}(\mathbf{x}^*)^{-1} \right\|_M + (2\lambda_1\delta + \lambda_2)\frac{\epsilon^p}{1 - r^{'p}}. \tag{C.55}$$

By choosing sufficiently small ϵ, we can have

$$(2\lambda_1\delta + \lambda_2)\frac{\epsilon^p}{1 - r^{'p}} < \delta, \tag{C.56}$$

which further leads to

$$\left\| \mathbf{H}_i^{m} - F^{'}(\mathbf{x}^*)^{-1} \right\| \leq 2\alpha\delta. \tag{C.57}$$

Similarly to the case when $m = 1$, with the help of Lemma 2, it can be derived that

$$\|\mathbf{x}_i^{m+1} - \mathbf{x}^*\| \leq$$
$$[\|\mathbf{H}_i^m\| K\epsilon^p + 2\sigma\delta\alpha] + \|\mathbf{H}_i^m\| \|\hat{F}_i^m - F(\bar{\mathbf{x}_i}^m)\|. \tag{C.58}$$

Noticing that

$$\|\mathbf{H}_i^m\| \leq \|\mathbf{H}_i^m - F^{'}(\mathbf{x}^*)^{-1}\| \leq 2\alpha\delta + \gamma, \tag{C.59}$$

we can rewrite (C.58) as

$$\|\mathbf{x}_i^{m+1} - \mathbf{x}^*\| \leq r\|\bar{\mathbf{x}_i}^m - \mathbf{x}^*\| + (2\alpha\delta + \gamma)\|\hat{F}_i^m - F(\bar{\mathbf{x}_i}^m)\|. \tag{C.60}$$

Again, by Lemma 3, by choosing $l_m, l_m^{'}$ sufficiently large, we can conclude the induction argument by showing that

$$\|\bar{\mathbf{x}_i}^{m+1} - \mathbf{x}^*\| \leq r^{'} \|\bar{\mathbf{x}_i}^m - \mathbf{x}^*\| \tag{C.61}$$

REFERENCES

[1] A. Abur and A. G. Exposito, Power System State Estimation: Theory and Implementation. CRC Press, 2004.

[2] A. Chakrabortty and P. Khargoneka, Introduction to Wide-Area Control of Power Systems, in American Control Conference, 2013, pp. 6773C 6785.

[3] Y.-F. Huang, S. Werner, J. Huang, N. Kashyap, and V. Gupta, State Estimation in Electric Power Grids: Meeting New Challenges Presented by the Requirements of the Future Grid, IEEE Signal Process. Mag., vol. 29, no. 5, pp. 33C43, 2012.

[4] R. Ebrahimian and R. Baldick, State Estimation Distributed Processing, IEEE Trans. Power Systems, vol. 15, no. 4, pp. 1240C1246, 2000.

[5] L. Zhao and A. Abur, Multiarea State Estimation Using Synchronized Phasor Measurements, IEEE Trans. Power Systems, vol. 20, no. 2, pp. 611C617, 2005.

[6] A. G. Exposito, A. Abur, A. de la Villa Jaen, and C. Gomez-Quiles, A Multilevel State Estimation Paradigm for Smart Grids, Proceedings of the IEEE, vol. 99, no. 6, pp. 952C976, 2011.

[7] A. Conejo, S. De La Torre, and M. Canas, An Optimization Approach to Multiarea State Estimation, IEEE Trans. Power Systems, vol. 22, no. 1, pp. 213C221, Feb 2007

[8] J. Carvalho and F. Barbosa, Distributed Processing in Power System State Estimation, in 10th Mediterranean Electrotechnical Conference (MELECON), vol. 3, May 2000, pp. 1128C1131.

[9] V. Kekatos and G. Giannakis, Distributed Robust Power System State Estimation, IEEE Trans. Power Systems, vol. 28, no. 2, pp. 1617C1626, May 2013.

[10] X. Tai, Z. Lin, M. Fu, and Y. Sun, A New Distributed State Estimation Technique for Power Networks, in American Control Conference, 2013, pp. 3338C3343.

[11] L. Xie, D. Choi, S. Kar, and H. Poor, Fully Distributed State Estimation for Wide-Area Monitoring Systems, Smart Grid, IEEE Transactions on, vol. 3, no. 3, pp. 1154C1169, 2012.

[12] S. Kar, J. Moura, and K. Ramanan, Distributed Parameter Estimation in Sensor Networks: Nonlinear Observation Models and Imperfect Communication, Information Theory, IEEE Transactions on, vol. 58, no. 6, pp. 3575 C3605, june 2012.

[13] X. Li and A. Scaglione, Convergence and applications of a Gossip based Gauss-Newton algorithm, IEEE Trans. Signal Process., vol. 61, no. 21, pp. 5231C5246, Nov. 2013

[14] X. Li, A. Scaglione Robust decentralized state estimation and tracking for power systems via network gossiping[J]. IEEE Journal on Selected Areas in Communications, 2013, 31(7): 1184-1194.

[15] J. E. Dennis, Jr. and J. J. More, Quasi-Newton methods, motivation and theory, SIAM Rev., 19 (1977), pp.46-89

[16] R. Fletcher and M. J. D. Powell, A rapidly convergent descent method for minimization, Comput. J., 6 (1963), pp. 163C168.

[17] D.C. Liu and J. Nocenal, On the limited memory BFGS methodfor large-scale optimization methods, Math. Programming, 5(1989), pp. 503-528.

[18] R. Ebrahimian and R. Baldick, State Estimation Distributed Processing, IEEE Trans. Power Syst., vol. 15, no. 4, pp. 1240C1246, 2000.

[19] A. Monticelli, State Estimation in Electric Power Systems: A Generalized Approach, 1999.

[20] D. G. Luenberger, Linear and nonlinear programming, Klewer Academic Publishers, Boston, 2003.

[21] Power systems test case archive. University of Washington. [Online]. Available: http://www.ee.washington.edu/research/pstca/

[22] R. D. Zimmerman, C. E. Murillo-Sanchez, and R. J. Thomas, MATPOWER:steady-state operations, planning and analysis tools for power systems research and education, IEEE Trans. Power Syst., vol. 26, no. 1, pp. 12C19, Feb. 2011.

[23] D.P.Bertsekas, Nonlinear Programming. Athena Sientific,1999

[24] F. Garin and L. Schenato, A survey on distributed estimation and control applications using linear consensus algorithms. Springer, 2011, vol. 406, ch. 3, pp. 75C107.

[25] D. G. Luenberger, Linear and nonlinear programming, Klewer Academic Publishers, Boston, 2003

[26] A. Nedic and A. Ozdaglar, Distributed Subgradient Methods for Multi-agent Optimization, Automatic Control, IEEE Transactions on, vol. 54, no. 1, pp. 48C61, 2009

[27] Dimakis, Alexandros G., et al. "Gossip algorithms for distributed signal processing." Proceedings of the IEEE 98.11 (2010): 1847-1864.

[28] Johansson, Bjorn, et al. "Subgradient methods and consensus algorithms for solving convex optimization problems." Decision and Control, 2008. CDC 2008. 47th IEEE Conference on. IEEE, 2008.

[29] Broyden, C. G., Dennis, J. E., Mor, J. J. (1973). On the local and superlinear convergence of quasi-Newton methods. IMA Journal of Applied Mathematics, 12(3), 223-245.

Jia Yu received the B.Eng. degree in electronics information engineering and the MSc degree in electronics engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2007 and 2009, respectively. From 2010 to 2012, he was a system engineer at Huawei Technologies Co., Shenzhen, China. He is currently working towards a Ph.D. degree at the University of Edinburgh. His research interesting lies in statistic signal processing and wireless sensor networks.

John Thompson (F16) is currently a Professor in signal processing and communications with the School of Engineering, University of Edinburgh, Edinburgh, U.K. His research interests include antenna array processing, co-operative communications systems, and energy efficient wireless communications. He has published in excess of 300 papers on these topics, including 100 journal paper publications. He is currently the Project Coordinator for U.K. EPSRC SERAN project on 5G technologies and the EU Marie Curie International Training Network project ADVANTAGE, which studies how communications and power engineering can provide future smart grid systems. He was an elected Member-at-Large for the Board of Governors of the IEEE Communications Society from 2012 to 2014, the second largest IEEE Society. He was also a Distinguished Lecturer on the topic of energy efficient communications and smart grid for the IEEE Communications Society from 2014 to 2015. He is an Editor for the Green Communications and Computing Series that appears regularly in the IEEE Communications Magazine