On the Development of Distributed Estimation Techniques for Wireless Sensor Networks

Trilochan Panigrahi

Department of Electronics and Communication Engineering

National Institute of Technology Rourkela
Rourkela, Odisha, 769 008, India

On the Development of Distributed Estimation Techniques for Wireless Sensor Networks

Thesis submitted in partial fulfillment
of the requirements for the degree of

Doctor of Phylosophy

in

Electronics and Communication Engineering

by

Trilochan Panigrahi
(Roll: 507EC106)

under the guidance of

Prof. Ganapati Panda
IIT Bhubaneswar

and

Prof. Bernard Mulgrew
University of Edinburgh, UK

Department of Electronics and Communication Engineering
National Institute of Technology Rourkela
Rourkela-769 008, Odisha, India
February 2012

dedicated to my parents with love...

Dept. of Electronics and Communication Engineering
National Institute of Technology Rourkela
Rourkela-769 008, Odisha, India.

February 20, 2012

Certificate

This is to certify that the work in the thesis entitled ***On the Development of Distributed Estimation Techniques for Wireless Sensor Networks*** by ***Trilochan Panigrahi*** is a record of an original research work carried out under our supervision and guidance in partial fulfillment of the requirements for the award of the degree of Doctor of Philosophy in Electronics and Communication Engineering. Neither this thesis nor any part of it has been submitted for any degree or academic award elsewhere.

Prof. Ganapati Panda
Professor
School of Electrical Sciences
Indian Institute of Technology Bhubaneswar
Odisha, India - 751 013

Prof. Bernard Mulgrew
Professor
Institute for Digital Communication
The University of Edinburgh
Scotland, UK

Acknowledgment

"The will of God will never take you where Grace of God will not protect you."
Thank you God for showing me the path...

I owe deep gratitude to the ones who have contributed greatly in completion of this thesis.

Foremost, I would like to express my sincere gratitude to my advisor, Prof. Ganapati Panda for providing me with a platform to work on challenging areas of distributed signal processing in Wireless Sensor Network. His profound insights and attention to details have been true inspirations to my research.

I am grateful to my co-supervisor, Prof. Bernard Mulgrew who has provided me with continuous encouragement and support to carry out research. His dedication and consistent notation in my writings has motivated me to work for excellence.

I am thankful to Prof. S. Meher, Prof. S. K. Patra, Prof. K. K. Mahapatra, Prof. D.P. Acharya, Prof. S. K. Behera of Electronics and Communication Engg. Department and Prof. S. K. Sahu of Civil Engg. Department for extending their valuable suggestions and help whenever I approached them.

Especially, I owe many thanks to Dr. Renato Cavalcante of University of Edinburgh, UK (2008-2009) from whom I learnt the concepts of distributed estimation on which my thesis belongs to. I am also thankful to George P. Gera, Shuang Wan, Sujit Bhattacharya, Rakesh, Kallol and their families of the University of Edinburgh for their support and guidance during my stay at Edinburgh. I am also grateful to the University of Edinburgh for providing me the resources available at the Department of IDCOM.

I am grateful to Pyari Mohan Pradhan for his useful comments which gave me moral boost to complete my dissertation work.

It is my great pleasure to show indebtedness to my friends like Upendra, Sudhansu, Sitanshu, Satyasai, Babita, Nithin and Vikas for their help during the course of this work.

The years spent in Rourkela would not have been as wonderful without my NIT friends, including Prashant, Bijaya, Jagannath, Goutam, Amitav, Prakash, Yogesh, Natrajan, Runakumari, Ratnakar, Saroj, Arunnashu Sir, Kanhu Sir, Karuppanan who contributed through their support, knowledge and friendship to this work. They made my stay in Rourkela an unforgettable and rewarding experience.

Special thanks to Prof. Ajit Kumar Sahoo and Prof. Ayash Kant Swain, for infallible motivation and moral support, whose involvement gave a new breath to my research.

I acknowledge all staff, research scholars and juniors of ECE Department, NIT Rourkela for helping me during my research work.

I am grateful to NIT Rourkela for providing me adequate infrastructure to carry out the present investigations. I am also grateful to UKIERI project for providing me financial support for traveling to Edinburgh and staying there for three months during summer (2008-2010).

I take this opportunity to express my regards and obligation to my family members whose support and encouragement I can never forget in my life.

I wish to appreciate and thank my daughter, Priyanshi, for allowing me to stay away in department for writing this thesis.

Without the constant support and encouragement of my wife, Prachi, I could hardly have completed this work. Her unending patience, encouragement and understanding had made it all possible, and meaningful. This thesis is for her.

Trilochan Panigrahi

Abstract

Wireless sensor networks (WSNs) have lately witnessed tremendous demand, as evidenced by the increasing number of day-to-day applications. The sensor nodes aim at estimating the parameters of their corresponding adaptive filters to achieve the desired response for the event of interest. Some of the burning issues related to linear parameter estimation in WSNs have been addressed in this thesis mainly focusing on reduction of communication overhead and latency, and robustness to noise. The first issue deals with the high communication overhead and latency in distributed parameter estimation techniques such as diffusion least mean squares (DLMS) and incremental least mean squares (ILMS) algorithms. Subsequently the poor performance demonstrated by these distributed techniques in presence of impulsive noise has been dealt separately. The issue of source localization i.e. estimation of source bearing in WSNs, where the existing decentralized algorithms fail to perform satisfactorily, has been resolved in this thesis. Further the same issue has been dealt separately independent of nodal connectivity in WSNs.

This thesis proposes two algorithms namely the block diffusion least mean squares (BDLMS) and block incremental least mean squares (BILMS) algorithms for reducing the communication overhead in WSNs. The theoretical and simulation studies demonstrate that BDLMS and BILMS algorithms provide the same performances as that of DLMS and ILMS, but with significant reduction in communication overheads per node. The latency also reduces by a factor as high as the block-size used in the proposed algorithms.

With an aim to develop robustness towards impulsive noise, this thesis proposes three robust distributed algorithms i.e. saturation nonlinearity incremental LMS (SNILMS), saturation nonlinearity diffusion LMS (SNDLMS) and Wilcoxon norm diffusion LMS (WNDLMS) algorithms. The steady-state analysis of SNILMS algorithm is carried out based on spatial-temporal energy conservation principle. The theoretical and simulation results show that these algorithms are robust to impul-

sive noise. The SNDLMS algorithm is found to provide better performance than SNILMS and WNDLMS algorithms.

In order to develop a distributed source localization technique, a novel diffusion maximum likelihood (ML) bearing estimation algorithm is proposed in this thesis which needs less communication overhead than the centralized algorithms. After forming a random array with its neighbours, each sensor node estimates the source bearing by optimizing the ML function locally using a diffusion particle swarm optimization algorithm. The simulation results show that the proposed algorithm performs better than the centralized multiple signal classification (MUSIC) algorithm in terms of probability of resolution and root mean square error. Further, in order to make the proposed algorithm independent of nodal connectivity, a distributed in-cluster bearing estimation technique is proposed. Each cluster of sensors estimates the source bearing by optimizing the ML function locally in cooperation with other clusters. The simulation results demonstrate improved performance of the proposed method in comparison to the centralized and decentralized MUSIC algorithms, and the distributed in-network algorithm.

***Keywords*:** Wireless Sensor Network, Distributed Estimation, Diffusion LMS, Incremental LMS, Error Saturation Nonlinearity, Wilcoxon Norm, Direction of Arrival, Maximum likelihood Estimation, Particle Swarm Optimization.

Contents

List of Figures

List of Tables

List of Abbreviations

AML	Approximated Maximum Likelihood
AP	Alternating Projection
AP-AML	Alternating Projection - Approximated Maximum Likelihood
BDLMS	Block Diffusion Least Mean Squares
BILMS	Block Incremental Least Mean Squares
BLMS	Block Least Mean Squares
BMSE	Block Mean Square Error
CRLB	Cramer-Rao Lower Bound
DLMS	Diffusion Least Mean Squares
DOA	Direction of Arrival
DPSO	Diffusion Particle Swarm Optimization
DPSO-ML	Diffusion Particle Swarm Optimization - Maximum Likelihood
EML	Exact Maximum Likelihood
EMSE	Excess Mean Square Deviation
ESPRIT	Estimation of Signal Parameters via Rotational Invariance Techniques
FC	Fusion Center
GA	Genetic Algorithm - Exact Maximum Likelihood
GA-EML	Genetic Algorithm - Exact Maximum Likelihood
GPS	Global Positioning System
iid	Independent and Identically Distributed
ILMS	Incremental Least Mean Squares
LMS	Least Mean Squares
ML	Maximum Likelihood
MLE	Maximum Likelihood Estimator
MSD	Mean Square Deviation
MSE	Mean Square Error
MUSIC	MUltiple SIgnal Classification
PR	Probability of Resolution
PSO	Particle Swarm Optimization

PSO-ML	Particle Swarm Optimization - Maximum Likelihood
PSO-EML	Particle Swarm Optimization - Exact Maximum Likelihood
RLS	Recursive Least Squares
RMSE	Root mean Square Error
SAGE	Space-Alternating Generalized Expectation-Maximization
SNDLMS	Saturation Nonlinearity Diffusion Least Mean Squares
SNILMS	Saturation Nonlinearity Incremental Least Mean Squares
SNR	Signal-to-Noise Ratio
WSNs	Wireless Sensor Networks
WNDLMS	Wilcoxon Norm Diffusion Least Mean Squares
WNILMS	Wilcoxon Norm Incremental Least Mean Squares

Chapter 1

Introduction

Chapter 1

Introduction

We interact with the physical world through our eyes, ears, nose, mouth, hands, and of course, our brain. In addition, we create instruments to augment our capabilities. With the advance in computing, communication, and microelectronic mechanical system technologies, we are getting closer to the physical world and monitoring and managing it. The wireless sensor networks (WSNs) opens a door for potential real world applications. A sensor network is a robust, distributed system, consisting of thousands of physically embedded, unattended, and often, untethered devices. Since the WSNs is distributed over a physical space, distributed signal processing algorithms are more suitable to extract information from the data collected at various nodes. If the required applications, and the sensor architecture allows more local processing, then it would be more energy-efficient, compared to communication extensive centralized processing.

1.1 Wireless Sensor Networks

WSNs comprise of a large number of small sensing self-powered sensor nodes distributed in a geographical region, which gather information or detect special events and communicate in a wireless fashion. Sensing, processing and communication are three key elements whose combination in one tiny device gives rise to a vast number of remote sensing applications [1,2]. Due to their several popular applications, efficient design and implementation of WSNs [3, 4] have become an area of current research. The nodes in a WSN operate with small and limited battery power and usually

non-renewable resource. Since communication among nodes consumes most of the energy [5], it is important to design the network with less communication among the nodes to estimate the required parameter vector. However, recent advances in low power very-large-scale integration (VLSI), embedded computing, communication hardware, and in general, the convergence of computing and communications, lead to the growth and implementation of this emerging technology [6].

There are many diverse applications of WSNs [1, 7]. In military, WSNs can be used for command, control, communication, intelligence, surveillance, reconnaissance, targeting system etc. WSNs can monitor patients and help disable patient in health care. WSNs can also improve the performance of industries in areas such as inventory management, product quality monitoring, disaster monitoring etc.

Although WSNs provide endless opportunities, but at the same time pose formidable challenges. Some of these challenges are accurate estimation of source position known as source localization, energy minimization, fault tolerant etc. To overcome these challenges, the sensor networks should have distributed processing capability. In a centralized system, some of the sensor nodes need to communicate over long distances which lead to more energy depletion. Hence, it is desirable to process locally as much information as possible in order to minimize the total number of bits transmitted [5, 8–10]. The sensor nodes are densely deployed, and are prone to failures. The topology of a sensor network changes frequently. Thus, unlike traditional networks, where the focus is on maximizing channel throughput or minimizing node deployment, the major consideration in a sensor network is to extend the system lifetime as well as the system robustness. A number of papers propose solutions to one or more of the above problems.

1.2 Distributed Estimation in WSNs

In many applications of WSNs, the objective is to solve an optimization problem in which the global cost function can be decomposed as the sum of local cost functions, each of which is known to only one sensor node in a network [11]. Applications that can be posed as such optimization problems include, motion planning in multiagent

systems [12, 13], acoustic source localization [14–16], and distributed adaptive filtering [17–20]. Typically, in these problems centralized approaches are not desirable because of physical limitations where the central processor may not have a direct access to the data of all other nodes or because of robustness issues where the system may fail if the central processor collapses. Therefore, a great deal of effort has been devoted to the development of distributed optimization algorithms [18,19,21–23]. In particular, the focus is on decentralized methods where nodes can work massively in parallel and exchange information with point-to-multipoint links [12, 17, 24]. These approaches often give rise to low-complexity iterative optimization algorithms that are suitable for large-scale systems using a simple communication model among nodes. The general mathematical formulation for the distributed estimation in WSNs is discussed in the following subsection.

This thesis focuses on the analysis of real-valued data, but can be extended to complex-valued data. Small bold letters are used to denote vectors, e.g., $\mathbf{w}$ and capital bold letter e.g. $\mathbf{W}$ denotes the matrix. T denotes the transpose operator. $\|\mathbf{w}\|^2 = \mathbf{w}^T\mathbf{w}$ and $\|\mathbf{w}\|^2_\Sigma = \mathbf{w}^T\Sigma\mathbf{w}$ denote the squared Euclidean norm and weighted-squared Euclidean norm of a vector respectively. The time index is placed as a subscript for vectors and between parentheses for scalars, e.g. $\mathbf{w}_i$ and $e(i)$ respectively.

1.2.1 Mathematical Formulation of Distributed Estimation

Consider a sensor network with N sensor nodes randomly distributed over the region of interest. The topology of a sensor network is modeled using an undirected graph. Let $\mathcal{G}$ be an undirected graph defined by a set of nodes $\mathcal{V}$ and a set of edges $\mathcal{E}$ [25]. Nodes j and k are called neighbours if the nodes are connected by an edge; that is $(j,k) \in \mathcal{E}$. A loop consists of a set of nodes $1, 2, ..., N-1$ such that the node k is $(k+1)$th node's neighbour, $k = 1, 2, ..., N$ and node 1 is Nth node's neighbour. Every node in the network $k \in \mathcal{V}$ is associated with noisy output d_k to the input data vector $\mathbf{u}_k$. It is assumed that the noise is independent of both input and output data and therefore the observations are spatially and temporally independent. The neighbourhood of node k is defined as the set of nodes connected to node k which

is defined as $\mathcal{N}_k = \{j|(j,k) \in \mathcal{E}\}$ [26].

It is assumed that at each discrete time instant i, every node k has access to the local scalar measurement $d_k(i)$ which is related to the input data vector $\mathbf{u}_{k,i}$ [27, 28] as

$$d_k(i) = \mathbf{u}_{k,i}\mathbf{w} + v_k(i), \qquad k = 1, 2, \ldots, N \tag{1.1}$$

where $\mathbf{w}$ is the $M \times 1$ vector parameter to be estimated; $\mathbf{u}_{k,i}$ is a $1 \times M$ regression vector, and $v_k(i)$ is the additive Gaussian noise with variance σ_k^2. The objective is to estimate the global parameter vector $\mathbf{w}^\circ$ using the data $\{d_k(i), \mathbf{u}_{k,i}\}$ available at every node in the network.

In order to estimate the optimum least square estimation vector $\mathbf{w}^\circ$ the following global cost function to be minimized:

$$f(\mathbf{w}) = \sum_{k=1}^{N} E\|d_k(i) - \mathbf{u}_{k,i}\mathbf{w}\|^2 \tag{1.2}$$

where $E\|\cdot\|$ denotes the expectation operator. The optimal least square solution can be obtained by solving the above equation [29] as

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \sum_{k=1}^{N} E\|d_k(i) - \mathbf{u}_{k,i}\mathbf{w}\|^2 \tag{1.3}$$

Assuming the process $d_k(i)$ and $\mathbf{u}_{k,i}$ are jointly stationary, solution of (1.3) leads to the well known Wiener filter estimate [27, 29]

$$\mathbf{w}^\circ = \left(\sum_{k=1}^{N} R_{u,k}\right)^{-1} \left(\sum_{k=1}^{N} R_{du,k}\right) \tag{1.4}$$

where $R_{u,k} = E[\mathbf{u}_{k,i}^T \mathbf{u}_{k,i}]$, $R_{du,k} = E[\mathbf{u}_k^T(i) d_k(i)]$. It may be observed that the second order moments vary from node to node. If $R_{u,k}$ and $R_{du,k}$ are known, then the steepest-descent approach is given as

$$\hat{\mathbf{w}}_i = \hat{\mathbf{w}}_{i-1} + \mu \sum_{k=1}^{N} (R_{u,k} - R_{du,k}) \tag{1.5}$$

where $\mu > 0$ is a step-size parameter and $\hat{\mathbf{w}}_i$ is an estimate of $\mathbf{w}^\circ$ at iteration i. With sufficiently small step-size, (1.5) would converge to $\mathbf{w}^\circ$ while avoiding the burden of

inverting $R_{u,k}$. In many linear regression applications involving online processing of data, this covariance information may be either unavailable or time varying, and thus impossible to update continuously [29]. An adaptive implementation can be obtained by replacing the second order moments with local instantaneous approximations as $R_{u,k} \approx \mathbf{u}_{k,i}^T \mathbf{u}_{k,i}$ and $R_{du,k} \approx d_k(i)\mathbf{u}_{k,i}^T$. Then a global centralized least mean squares (LMS) recursion is obtained as

$$\hat{\mathbf{w}}_i = \hat{\mathbf{w}}_{i-1} + \mu \sum_{k=1}^{N} \mathbf{u}_{k,i}^T (d_k(i) - \mathbf{u}_{k,i}\hat{\mathbf{w}}_{i-1}) \tag{1.6}$$

The update algorithm (1.6) is not distributed in nature. In centralized approach, every node in the network sends its data $\{d_k(i), \mathbf{u}_{k,i}\}$ either directly or by multi-hop relay, to a central fusion center (FC) for processing. The FC then solves the least square problem to find $\hat{\mathbf{w}}$. In the multihop relay case, each node must establish and maintain a routing table for the data packet to reach the FC. This is extremely challenging if the topology changes with time due to mobility or power constraints of the sensor nodes. Further this approach has the disadvantage of not being robust to failure of the FC.

However in a distributed sensor fusion scheme, there is no central FC. Each sensor only exchanges data with its neighbours and does local computation. The goal for each sensor is to have a global estimate of the unknown parameter. Therefore the objective is to estimate the global parameter in distributed way. In distributed optimization problem, the global cost function can be expressed as a sum of local cost functions and is given by,

$$f(\mathbf{w}) = \sum_{k=1}^{N} f_k(\mathbf{w}) \tag{1.7}$$

where $f(\mathbf{w})$ is the global cost function which can be expressed as a sum of N *local* cost functions $\{f_k(\mathbf{w})\}_{k=1}^N$ in which $f_k(\mathbf{w})$ depends on the data measured at sensor k. The local cost functions can be expressed as

$$f_k(\mathbf{w}) = E\|d_k(i) - \mathbf{u}_{k,i}\mathbf{w}\|^2 \tag{1.8}$$

Now we consider the problem of optimizing the sum of convex objective functions

corresponding to N nodes connected over a time varying topology. The goal of the sensor nodes is to cooperatively solve the unconstrained optimization problem of (1.3).

The performance of distributed algorithm depends on the mode of cooperation among the nodes, i.e., incremental [18, 30, 31] or diffusion [19, 32–34] types. In incremental mode of cooperation, each node transmits its local parameter estimate to the adjacent node and the information flows in a sequential manner. During this time, the nodes act like independent agents and there is a limited interaction among the nodes. This requires less amount of communication and power. On the other hand in diffusion mode of cooperation, each node transmits its local parameter estimate to all its neighbors as dictated by network topology [35,36]. But this mode of cooperation requires more amount of communication compared to incremental mode. Although the diffusion mode needs more communication, it is more robust to link and node failures [33]. To improve the robustness against the spatial variation of SNR over the network, recently an efficient adaptive combination strategy has been proposed [34]. A fully distributed network with provision of adaptive implementation enabling each node to take individual decision has been recently reported [37]. The amount of communication can be reduced by allowing each node to communicate only with a subset of its neighbors.

1.2.2 Incremental Adaptive Estimation

In a decentralized incremental approach, each iteration (1.6) is divided into N subiterations [30]. In kth subiteration, each sensor node updates its local parameter estimate$f_k(\mathbf{w})$. The algorithm can be written as:

$$\begin{aligned}
\boldsymbol{\psi}_0^{(i)} &= \hat{\mathbf{w}}_{i-1} \\
\boldsymbol{\psi}_k^{(i)} &= \boldsymbol{\psi}_{k-1}^{(i)} - \mathbf{u}_{k,i}^T(d_k(i) - \mathbf{u}_{k,i}\boldsymbol{\psi}_{k-1}^{(i)}), \quad k = 1, 2, \ldots, N \\
\hat{\mathbf{w}}_i &= \boldsymbol{\psi}_N^{(i)}
\end{aligned} \tag{1.9}$$

where $\hat{\mathbf{w}}^{(i)}$ is the estimated parameter vector obtained after i iterations and $\psi_k^{(i)}$ is the parameter estimate of kth node in ith iteration. For analyzing the rate of

convergence an arbitrary starting point is assumed.

1.2.3 Diffusion Adaptive Estimation

The main idea of the distributed diffusion algorithm is the use of consensus as a mechanism for distributing the computations among the nodes [35,36]. Each sensor node starts with an initial estimate $\mathbf{w}_k^0 \in \mathtt{R}^M$ and updates its estimate at discrete instant $i = 1, 2, \ldots, ...$ The variable $\mathbf{w}_k^i$ denotes the vector estimate maintained by node k at ith iteration. The node k updates its current estimate using the estimate received form neighbour nodes $\mathcal{N}_k$. The update equation is given as

$$\mathbf{w}_k^i = \sum_{j \in N_k} a_{jk} \mathbf{w}_j^{i-1} - \alpha g_k^{i-1}(\mathbf{w}), \quad k = 1, 2, \ldots, N \tag{1.10}$$

where the scalar $\alpha > 0$ is a step size and the vector $g_k^{i-1}(\mathbf{w})$ is the gradient of the kth node cost function $f_k(\mathbf{w})$ with respect to $\mathbf{w}_k^{i-1}$. The scalars a_{jk} are non negative weights that node k gives to the estimates of neighbour nodes $\mathcal{N}_k$. The gradient vector $g_k^{i-1}(\mathbf{w})$ for real data is given as

$$g_k^{i-1}(\mathbf{w}) = -2 * \mathbf{u}_{k,i}^T (d_k(i) - \mathbf{u}_{k,i} \hat{\mathbf{w}}_{i-1}) \tag{1.11}$$

Substituting (1.11) into (1.10), the recursion equation is obtained as

$$\mathbf{w}_k^i = \sum_{j \in \mathcal{N}_k} a_{jk} \mathbf{w}_j^{i-1} + \alpha \mathbf{u}_{k,i}^T (d_k(i) - \mathbf{u}_{k,i} \hat{\mathbf{w}}_{i-1}), \quad k = 1, 2, \ldots, N \tag{1.12}$$

Equation (1.12) can be viewed as the combination of the *consensus* $\sum_{j \in \mathcal{N}_k} a_{jk} \mathbf{w}_k^{i-1}$ and the *gradient* $-\alpha g_k^{i-1}(\mathbf{w})$ steps. The gradient step is taken by the node to minimize its own objective, while the consensus step serves to align its estimation with the estimation of the neighbours. When the network is sufficiently connected, one would expect that all nodes have the same estimate after some iteration.

1.3 Background and Scope of the Thesis

The conventional distributed estimation algorithms involve significant communication overheads. It is of great advantage to reduce the communication bandwidth and

power consumptions involved in the transmission and reception of messages across the resource-constrained nodes in WSNs. In the coming years, with continuing advances in microelectronics, enough computing resources can be accommodated in the nodes to reduce the processing delays, but the communication bandwidth and communication delay will pose major operational bottlenecks in the WSNs. In literature, a number of research papers have appeared and addressed the energy issues of sensor networks [38,39]. It is of great importance to minimize the communication among nodes by maximizing local estimation in each sensor node [8, 24, 29, 40, 41]. In distributed parameter estimation problem, during each sampling instant, a typical sensor node communicates its estimate either by the diffusion or incremental manner. If the nodes communicate after processing a block of data instead of communicating after each sample data, then substantial communication overhead can be reduced.

It is a fact that when data is contaminated with non-Gaussian noise or outliers, the conventional algorithms which are based on squared error minimization yield poor performance. Nonlinear techniques [42, 43] are employed to reduce the effect of impulsive interference on the systems. Robust LMS algorithm [44] has been reported in the literature using Wilcoxon norm which is not distributed in nature. In this dissertation a novel distributed estimation algorithm is developed using error saturation nonlinearity which is robust to impulsive noise or outliers.

The exact ML DOA estimation for source localization using evolutionary algorithm has been reported in the literature [45, 46]. These centralized algorithms can be used in WSNs for source localization. But the centralized approach possesses excess communication overhead problem. A decentralized method has been proposed by dividing the large array into sub arrays [47,48] for the DOA estimation. Attempts have also been made to estimate the source location by measuring DOA with an antenna array at each sensor node [49,50] or by taking group of sensor subarrays [51]. But these methods fail to provide the global performance. Hence distributed bearing estimation algorithm can be developed by using consensus algorithms [40] in order to achieve global performance. Further, the advantage of clustering in WSNs [52]

can also be used to improvise the algorithm performance.

1.4 Motivation Behind the Research Work

Energy efficiency, low latency, high estimation accuracy, and fast convergence are important goals in distributed estimation algorithm for WSNs. If the estimation algorithm is distributed, then it tries to minimize the amount of communication required by processing the data locally as much as possible. The conventional distributed LMS algorithms [18, 32] involve significant communication overheads because the nodes are communicating after processing each sample of data. On the other hand, if block LMS algorithm are used for weight update then there will be substantial reduction in communication overhead. Therefore the block formulation of the existing distributed LMS algorithm [18, 32] is needed to make the algorithm energy efficient. The adaptive mechanism is such that the nodes of the same neighborhood communicate with each other after processing a block of data, instead of communicating the estimates to the neighbors after every input sample. As a result, the average bandwidth for communication among the neighboring nodes decreases.

It is a common practice that the data in WSNs are corrupted by impulsive noise. The conventional estimation algorithms like the LMS, and recursive least square (RLS), which are based on squared error as the cost function, are not robust to the error caused due to nodal failure or in presence of impulsive noise. But in the literature few adaptive algorithms have been reported based on robust function which are robust to impulsive noise [42, 44, 53–55]. Thus there is a need to develop distributed version of robust estimation algorithm which will provide improved performance when data is contaminated with impulsive noise.

Accurate DOA estimation is important for source localization and tracking [15, 30, 49]. But it is not practicable by using measured data of a single node. An alternative approach is to employ the centralized processing where the whole network is considered as an arbitrary array. The DOA can be estimated using ML-PSO [46] efficiently. But centralized processing is not practicable for large network as it requires excessive communication overheads to deliver and process the data to

the central processor. It may not be possible to maintain the coherence between signals received from widely separated sensor nodes in large distributed SN. In recent past the distributed optimization methods using consensus algorithms [40] have received significant attention. A truly distributed DOA estimation algorithm based on diffusion consensus [35] is needed where each node makes an arbitrary array with the neighbouring nodes and then estimates the DOA by optimizing the local ML function in a cooperative manner using DPSO algorithm. In this formulation the overall performance depends on the node connectivity. Specifically the nodes present in the edge of a network are unable to achieve the global performance due to less degree of connectivity. To overcome this difficulty and to reduce the overall communication, the clustering based distributed DOA estimation can be proposed in this work. Instead of running the PSO at each node, now the cluster heads cooperate to achieve the global performance. Thus the computational burden and memory usage are also reduced. The communication overhead can also be reduce by using block concept in diffusion cooperation.

1.5 Objectives of the Thesis

The objectives of the proposed research work are as follows

- To develop new distributed adaptive algorithms for reducing the communication overhead and latency in WSNs.
- To develop distributed robust adaptive algorithms for estimating parameters in presence of impulsive noise.
- To develop a distributed ML bearing estimation technique for improving the accuracy of source localization in WSNs.

1.6 Structure and Chapter Wise Contributions of the Thesis

Chapter 1

Introduction to WSNs, its applications and distributed parameter estimation in WSNs is presented in this chapter. The motivation behind the energy efficient distributed estimation over centralized method is outlined. The importance of robust parameter estimation in presence of outliers is also briefed. The motivation of present research structure and the chapter wise presentation of the thesis are also dealt in this chapter.

Chapter 2

This chapter provides a comprehensive overview of the related work done by different authors in the area of WSNs. The main focuses are given to distributed estimation, robust estimation in when link/node failure and in presence of impulsive noise, and distributed source localization using WSNs.

Chapter 3

In this chapter, two new distributed algorithms namely the BDLMS and BILMS are proposed by extending the concept of block adaptive filtering technique to distributed adaptation scenario. The performance analysis of the proposed BDLMS and BILMS algorithms has been carried out, and has been shown to provide similar performance as those offered by the conventional diffusion and incremental LMS algorithms, respectively. The convergence characteristics in terms of mean-square error (MSE), mean-square deviation (MSD) and excess mean-square error (EMSE) of the proposed algorithms are obtained from the simulation study and are found to be in agreement with those obtained from the theoretical analysis. An analysis of communication cost and latency are also presented. A theoretical comparison of the performance of distributed LMS with block distributed LMS is also given. An interesting observation is noticed that the proposed block-based algorithms provide significantly lower communication overheads per node and latencies.

Chapter 4

Robust distributed estimation algorithms for WSNs based on error saturation nonlinearity and Wilcoxon norm are presented in Chapter 3. The error saturation nonlinearity strategy is first introduced in incremental cooperative distributed network to estimate the desired parameters in presence of impulsive noise. The theoretical analysis of saturation nonlinearity ILMS (SNILMS) is carried out by employing the spatial-temporal energy conservation principle. The steady-state expressions for MSD and EMSE are derived. Finally it is shown that the presence of feedback error nonlinearity has made the proposed distributed incremental algorithm robust to the impulsive noise. In a similar manner, two robust diffusion algorithms based on error saturation nonlinearity and Wilcoxon norm are also derived and their performances are studied in details.

Chapter 5

In this chapter a novel distributed ML bearing estimation strategy is developed following diffusion principle in WSNs. Each sensor node estimates the source bearing by optimizing the ML function locally after forming a random array with its neighbours. The diffusion particle swarm optimization (DPSO) is proposed to optimize the ML function. During optimization process each associated node shares its best information with other nodes so that global estimation is achieved. The performance of the proposed distributed algorithm is analyzed in terms of probability of resolution (PR) and root mean square error (RMSE). The simulation results are compared with that offered by the centralized MUSIC algorithm and the global ML-PSO algorithm. The RMSE is also compared with that of theoretical Cramer-Rao lower bound (CRLB). When the nodes estimate the source bearing in distributed manner, the PR is nearly the same as that of the global estimation. The RMSE performance is equal at lower SNR; but at higher SNR the distributed estimation is better in comparison to the small network when it is not a part of the global network. This algorithm is energy efficient because it needs less communication overhead compared to the conventional centralized method.

Chapter 6

A distributed ML bearing estimation strategy based on clustering technique is proposed in this chapter. In this approach, each sensor node groups into a cluster and forms a random array with its neighbour nodes inside the cluster. The clusters estimate the source bearing by optimizing the ML function locally with the cooperation of other clusters. The DPSO is used to optimize the ML function. During the optimization process each associated cluster shares its best information with other clusters so that global estimation is achieved. The simulation results exhibit improved performance of the proposed distributed in-cluster method compared to that offered by the centralized and decentralized MUSIC, and distributed in-network algorithm (described in Chapter 5). The new distributed in-clustering algorithm uses less communication over head compared to all the existing algorithms. Further the computational burden and memory usage are also less because in this approach only the cluster head runs the PSO for bearing estimation unlike at each node in distributed in-network algorithm described in previous chapter.

Chapter 7

Finally, Chapter 7 outlines conclusions of the work that has been described in this thesis. It also discuss the achievements and limitations of the results obtained. Research topics which can be taken up subsequently on the same area are also included.

1.7 Conclusion

This chapter provides a brief introduction to WSNs and distributed estimation problem formulation. It also systematically outlines the scope, the motivation which resulted in the investigation and the objectives of the thesis. A concise presentation of research work carried out in each chapter and the contribution made by the candidate have also been dealt. In essence this chapter provides a complete overview of the total thesis in a condensed manner.

Chapter 2

Related Work

Chapter 2

Related Work

Literature survey on the current research work has been discussed in this chapter. In the beginning, the burning issues for the design and implementation of the WSNs are introduced. Out of several challenges, the basic problems of WSNs are focused in this Ph.D. Dissertation. These are energy efficient distributed estimation, robust distributed estimation and distributed bearing estimation for source localization. In this chapter, the related work done by several researches on the aforesaid challenges are briefly discussed.

2.1 Introduction

Recent advances in wireless communications and electronics have made the deployment of small, inexpensive, low-power, distributed devices, which are capable of local processing and communicate untethered in short distance [4, 56]. Such nodes are called as sensor nodes. Each sensor node is capable of only a limited amount of processing. But when coordinated with the information from a large number of other nodes, they have the ability to measure a given physical environment in great detail. Thus, a sensor network can be described as a collection of sensor nodes which co-ordinate to perform some specific action [1, 5, 57, 58]. These sensors are deployed for carrying out specialized tasks like surveillance and security, environmental monitoring, transport, precision agriculture, manufacturing and inventory tracking and health care [1, 59]. The principal advantage is their ability to be deployed in almost all kinds of terrain with hostile environment where it might not be possible or

difficult to use conventional wired networks.

Previously, sensor networks consisted of small number of sensor nodes that were wired to a central processing station. However, nowadays, the focus is more on wireless, distributed, sensing nodes. But, why distributed, wireless sensing is necessary is a big question comes in our mind [5]. When the exact location of a particular phenomenon is unknown, distributed sensing allows for closer placement to the phenomenon than a single sensor would permit. Also, in many cases, multiple sensor nodes are required to overcome environmental obstacles like obstructions, line of sight constraints etc. In most cases, the environment to be monitored does not have an existing infrastructure for either energy or communication.

Another requirement for sensor networks would be distributed processing capability [60]. This is necessary since communication is a major consumer of energy. A centralized system would mean that some of the sensors would need to communicate over long distances that lead to even more energy depletion. Hence, it would be a good idea to process locally as much information as possible in order to minimize the total number of bits transmitted [8, 9].

A number of literatures propose solutions to one or more of the above problems. Our survey focuses on the suggested solutions in the following areas:

- **Energy Efficiency :** Energy efficiency is a dominant consideration no matter what the problem is. This is because sensor nodes only have a small and finite source of energy. Many solutions, both hardware and software related, have been proposed to optimize energy usage [39]. But this thesis focuses on minimization of communication overheads in the WSNs.

- **Robust Estimation :** The failure of node/link is a common phenomenon in WSNs. The data may be corrupted by impulsive noise. In that situation, the conventional estimation algorithm fail to achieve desired performance. Thus a robust adaptive distributed algorithm for that situation [53, 54] is required.

- **Distributed Source Localization:** In most of the cases, sensor nodes are used to find the exact source location by estimating the source bearing [30, 47,

59, 61–63]. Centralized methods are used to find the direction of arrival which required more communication overhead. Therefore this chapter focuses on the issue to develop distributed bearing estimation algorithm in WSNs.

2.2 Distributed Estimation in WSNs

The WSNs, being distributed over a physical area, requires distributed signal processing algorithms to extract information from data collected at various nodes. Each node in a network collects noisy observations related to certain desired parameters. The neighbouring nodes then interact with themselves in certain manner. However a traditional centralized solution, the node in the network collects observations and conveys them to central processor where they are fused and the parameters are estimated and broadcasted back to the individual node. This approach requires a powerful central processor and more communication between nodes and the central processor. In addition, a centralized solution limits the ability of the nodes to adapt in real time [10].

Efficient design and implementation of wireless sensor networks has become a hot area of present day research. An efficient network involves less communication among the nodes to estimate the required parameter vector [1, 5, 64] because the life of battery decides the working period of the battery-powered network. In literature a number of research papers have appeared to address the energy issue of sensor network. Investigators are also engaged to develop distributed collaborative signal processing to estimate, detect and track to reduce communication overheads between nodes and also with central processor.

In recent years a number of distributed adaptive algorithms for wireless sensor networks have been developed. However, the performance of these networks depends on the mode of cooperation, e.g., incremental [18, 30, 31, 65], diffusion [19, 32], diffusion recursive least square [21], distributed detection [37], probabilistic diffusion [33] and diffusion with adaptive combiner [34]. In an incremental mode of cooperation the data flows from one node to the neighbour node in a cyclic manner. On the other hand in diffusion mode, each node communicates with all its neighbouring

nodes as defined by the network topology [35, 36]. The amount of communications in this mode is higher than incremental cooperation. Though this mode needs more communication, it is more robust to link and node failure [33]. To improve the robustness against the spatial variation of SNR over the network recently an efficient adaptive combination strategy has been proposed [34]. If the intended application and the sensor architecture allow more local processing, then it is more energy efficient compared to communication extensive of processing. So there is need to search for new type of adaptive algorithms to reduce communication overhead as well as latency in the network. Distributed Kalman filtering [20] is used in WSNs for tracking the object.

The block least-mean square (BLMS) algorithm is proposed in [66, 67] and its performance is studied in [68–70]. In this filtering the filter coefficients are adjusted once per each block of data in contrast to once per input sample in conventional least mean square (LMS) algorithm. The study allows that the block adaptive filter permits fast implementation while maintaining equivalent performance to that of widely used LMS adaptive filter. Adaptive algorithms based on the conjugate gradient method for finite impulse response (FIR) block adaptive filters are reported in [71, 72]. Using fast convolution technique, the block conjugate algorithm (BCG) is implemented in the frequency domain to provide significant computational savings over time-domain BCG algorithm, especially for a large filter-tap order. Subsequently a new adaptive algorithm is reported [67] based on Newton transversal filtering algorithm. These block LMS algorithms may be conveniently used at local nodes so that the communication between nodes is reduced to once per block.

2.3 Robust Estimation in WSNs

It is known that when data is contaminated with non-Gaussian noise, the conventional adaptive filters minimizing least square or mean square criterion provides poor performance. This leads to a new research in modern communication systems, where the performance is limited by interference of impulsive nature. In many physical environments the additive noise is modeled as impulsive and is characterized

by long-tailed non-Gaussian distribution. The performance of the system is evaluated under the assumption that the Gaussian noise is severally degraded by the non-Gaussian or Gaussian mixture [73] noise due to deviation from normality in the tails [54, 55]. Nonlinear techniques are employed to reduce the effect of impulsive interference on the systems. The effects of saturation type of non-linearity on the least-mean square adaptation for Gaussian inputs and Gaussian noise have been studied [43, 74].

A number of algorithms have been proposed [44, 53, 54, 75–81] in the literature to reduce the effects of impulsive noise. For example, in the order statistic least mean-square algorithm and median filter are used to limit the adverse effect of impulsive noise [76]. Similarly, in the adaptive threshold nonlinear algorithm [77], nonlinear clipping function is used to limit the transient fluctuation. Recently rank-based Wilcoxon approach is used for linear regression problems to make the learning algorithm robust against outliers [82]. A robust LMS algorithm [44, 81] has been reported in the literature using Wilcoxon norm. This class of algorithms is difficult to analyze and hence alternative methods have been sought for distributed robust estimation. Bang *et. al.* [53] have proposed a proportional sign algorithm which is robust in the presence of contaminated-Gaussian noise, but this algorithm involves a fixed nonlinear function [77]. Delouille *et. al.* in [80] have proposed a method to minimize the mean square error by using an iterative algorithm. Transmission of all data to a central processor and then performing estimation using techniques such as Wiener Filtering (with complexity $O(N^2)$), requires large amount of communication.

In [42], it has been reported that the error saturation nonlinearities based LMS provides good performance in presence of impulsive noise. The effects of saturation type of non-linearity on the least-mean square adaptation for Gaussian inputs and Gaussian noise have been reported in literature [43, 74]. The robust adaptive algorithms discussed here are not directly useful in applications which are inherently distributed in nature.

The LMS algorithm is a popular adaptive algorithm because of its simplicity and stability [83, 84]. Recently several distributed type of algorithms based on LMS has

been suggested and analyzed in the literature [18, 32]. These are not robust against impulsive noise in the training signal as the squared error norm is used as the cost function in deriving the algorithm. On the other hand centralized robust class of least-mean square algorithm with error saturation nonlinearity is of special importance. The error nonlinearity analysis [85, 86] using weighted-energy conservation method for Gaussian data has been dealt in the literature. In [42] the authors provide the basis for extending to Gaussian mixture case. It also suggests how it can be applied to each independent component separately to obtain recursive relation for the nonlinear LMS. The convergence analysis of non distributed error saturation nonlinearity LMS algorithm in the presence of impulsive noise has been reported in [87].

2.4 Distributed Source Localization

In WSNs the sensor nodes are distributed arbitrarily in any geographical area for special applications like surveillance and security, environmental monitoring, manufacturing and inventory tracking, transport, precision agriculture and health care etc [1, 59]. Bearing estimation is an important problem in WSNs to estimate the source location which is a well-known problem in the fields of radar, sonar, radio-astronomy, underwater surveillance and seismology etc. In bearing estimation, the outputs from a set of sensors are analyzed to determine the bearings of signals arriving at the sensor array. Many existing array processing methods are available in literature which required a centralized processing of sensor outputs. One of the simplest versions of this problem is the estimation of the directions-of-arrival (DOA) of narrow-band sources where the sources are assumed to be located in the far field of the sensor array [88].

The maximum likelihood (ML) method is one of the solution techniques in source localization problem [89]. The ML function can formulated from signal-noise model equation, which holds its maxima when the tried angle of arrival is exactly equal to the actual incident angle. An optimal estimation of the incident angle may be obtained by minimizing a ML function. This is known as maximum likelihood solution.

Different ML algorithms have different likelihood functions, which came from different models of the signals and noise to be estimated. Standard iterative approaches have been proposed to solve the resulting nonlinear optimization problem. Due to the high computational load of the multivariate nonlinear optimization problem required in ML estimation, several high resolution suboptimal techniques have been suggested in literature which includes the minimum variance method of Capon [90], the MUSIC method of Schmidt [91]. The performances of suboptimal techniques are in general inferior to the ML technique in low signal-to-noise ratio (SNR) or small number of samples. The MUSIC provides comparatively same performance with ML if the number of snapshots are high at high SNR [92].

Several authors have proposed an alternative approaches that allow reducing the communication and computational burdens associated in centralized processing. A decentralized method has been adapted by dividing the large array into sub arrays [47] for DOA estimation. The observed data is analysed by local processors and the results are communicated to central processor. The central processor uses the sub array information to solve the problem of source detection and location. There are two decentralized array processing algorithms using MODE have been dealt in [93]. The authors have combined all the estimates form subbarray by using the theoretical bounds to achieve the global optimum. The statistical analysis of MUSIC [48] conveys that the the decentralized estimate is always poorer than the centralized method since in the later the information associated is more as it possesses all the data from the network compared to any subarray. The robustness of decentralized process is discussed in [62].

Distributed sensor signal processing in sensor network deals with the problem of extracting information from a collection of measurements obtained from sensors distributed in space. Distributed estimation algorithms in SN have been proposed in literature to achieve global optimum estimation by processing the data locally at every node so that the burden on the fusion center is reduced. Global decision or estimation is achieved totally by distributed approach with no need of centralized processor at least in the case where the whole network observes a common event.

A strategy that has received significant attention in the last few years is on the development of distributed optimization methods using consensus algorithms. The basic idea is that if the network is fully connected then the local exchange of information among nearby sensors is sufficient to reach global consensus. In one of the approaches each node in the network receives the updated estimation from its neighbour nodes, fuses it by using some rules to make it initial or previous guess and then updates it by using its own observed data [40].

Depending on the manner by which the nodes cooperate with each other, they are referred to as incremental algorithm or diffusion algorithm. In an incremental mode of co-operation, a cyclic path through the network is required, so that information is communicated sequentially from one node to another [31]. In the other hand in diffusion method the nodes communicate with all of their neighbors as dictated by the network topology [35, 36]. The diffusion estimation is considered as a parallel architecture where all sensor nodes deliver crude estimation by using its own observed data and information form neighbor nodes. The detailed investigation of spatial adaptive filtering in WSN is dealt in [18, 21, 32]. Recently a generalized distributed signal processing is introduced in [94, 95] in which the authors have suggested a distributed node specific signal estimation algorithm. In the network all nodes contribute a common goal i.e to estimate sources DOA in present scenario which is the same for all nodes. We have considered a special problem in which each node in the network estimates the node specific parameters from its locally defined cost function. This means that all nodes have same local objective, which may improvise their own performance through cooperation with others unlike all the nodes have different local objectives in [94].

In literature attempts have been made to estimate source location by measuring DOA with an antenna array at each sensor node [49, 50, 59] or by taking group of sensor subarrays [51]. The process is adapted to the DOA estimate from each subarray of sensors. Then the triangulation process of determining the intersection of these cross bearing DOA angles can be used to estimate the source location. The diffusion co-operation concept could be employed to estimate the DOA for source

localization.

There are different optimization techniques available in literature for optimization of ML function like AP-AML [63], fast EM and SAGE algorithms [96] and a local search technique like Quasi-Newton methods to estimate different parameters of the source. All these techniques have several limitations because the ML cost function is multidimensional which needs extensive computation. A good initialization is also crucial for achieving global optimization and it is not guarantee that these local search techniques always have global converge.

The evolutionary algorithms like genetic algorithm (GA) [97], particle swarm optimization (PSO) and simulated annealing (SA) [98, 99] can be designed to optimize the ML function. PSO is a recent evolutionary algorithm first introduced by Eberhart and Kennedy in 1995 [100, 101]. As an emerging technology, PSO has attracted a lot of attention in recent years, and has been successfully applied in many fields, such as phased array synthesis [102, 103], electromagnetic optimization [104], and etc. Most of the applications demonstrated that PSO could give competitive or even better results in a faster and cheaper way, compared with other heuristic methods such as GA. Genetic algorithm [45] and particle swarm optimization [46] had already used as a global optimization technique to estimate the DOA. The author in [46] said that PSO-EML is around 20 times more efficient that GA-EML. The distributed PSO has been applied for nonlinear system identification problem in both incremental and distributed way in [105].

Clustering is a standard approach used in WSNs in order to achieve efficient and scalable performance [52,106,107]. Clustering groups the nodes and saves energy and reduces networks contention as nodes communicate their data over shorter distance to their respective cluster heads. In this chapter we proposed a hybrid form of aforesaid techniques and formulate a distributed in-cluster approach. There exists a multihop path within each cluster that transits the observation data to the cluster head once in each experiment [108]. As results the clusters operates similar to the distributed in-network scheme to achieve the global performance. The clustering alleviates the inherent inflexibility that exists in both the centralized as well as

distributed in-network algorithms. The centralized algorithm gives more accurate estimation regardless of the communication cost. On the other hand the distributed in-network algorithm fails to produce accurate estimate if the node connectivity is less. Therefore, when the application demands the most energy efficient algorithm which can provide the performance nearly equal to that of centralized algorithm, then the distributed in-cluster algorithm is the best suitable. We have the flexibility to adjust the number of clusters, equivalently the cluster size to accommodate the WSNs requirements [109].

2.5 Concluding Remarks

In this chapter, related works on some of the current issues present in WSNs are discussed. The works done on distributed estimation of parameters in WSNs are briefly presented. Similarly the work done related to the robust distributed estimation in WSNs when node/link failed is discussed. Further the work done on distributed source localization using WSNs are also presented in this chapter. It is found that the current versions of the distributed estimation algorithms need more communication overheads. Therefore, it motivate to the development of the distributed algorithm in order to minimize the communication overhead and latency.

Chapter 3

Distributed Estimation in Wireless Sensor Networks using Block Least-Mean Squares Algorithm

Chapter 3

Distributed Estimation in Wireless Sensor Networks using Block Least Mean Squares Algorithm

Two new distributed algorithms namely the block diffusion least mean squares (BDLMS) and block incremental least mean squares (BILMS) are proposed by extending the concept of block adaptive filtering technique to distributed adaptation scenario. The convergence analysis of proposed algorithms obtained from the simulation study is also found to be in agreement with the theoretical analysis. The remarkable and interesting aspect of the proposed block-based algorithms is that their communication overheads per node and latencies are less than those of the conventional algorithms by a factor as high as the block-size used in the algorithms.

3.1 Introduction

The WSNs consist of a group of sensor nodes which perform distributed sensing by coordinating themselves through wireless links. Since the nodes in a WSNs function with limited battery power, it is important to design the networks with less communication among the nodes to estimate the required parameter vector [5, 64]. According to the energy estimation scheme based on the 4th power loss model with Raleyigh fading, the transmission of 1 Kb of data over a distance of 100m, operating at 1 GHz using binary-phase-shift-keying (BPSK) modulation with 10^{-6} bit-error rate, an ideal receiver (no noise) requires approximately 3J of energy [110]. The same energy can be used for executing 300M instructions in a 100 MIPS/watt general

purpose processor. This indicates that *the transmission of a single bit in WSNs may require the same energy as the processing of* $30,000$ *instructions in a general purpose machine; and therefore it is of great importance to minimize the communication among nodes by maximizing the local processing in the sensor nodes.*

If the intended application and the sensor architecture allow more local processing, then it would be more energy-efficient compared to communication extensive centralized processing. Alternatively, each node in the network can function as an individual adaptive filter to estimate the parameter from the local observation and by cooperating with the neighbours. So there is a need to search for new distributed adaptive algorithms to reduce communication overhead for low-power consumption and low-latency system for real-time operation [25].

In block filtering technique [66, 67], the filter coefficients are adjusted once for each new block of data in contrast to once for each new input sample as in the case of LMS algorithm. In addition the block adaptive filter permits faster implementation while maintaining equivalent performance compared to that of conventional LMS adaptive filter. Therefore the block LMS (BLMS) algorithm can be potentially used at each node to reduce the communication overheads.

Keeping these in view, in the present work, a block formulation of the existing cooperative algorithm [19,30,32,65] has been suggested. Accordingly a block adaptive mechanism is proposed in which the nodes of the same neighborhood communicates with each other after processing a block of data, instead of communicating the estimates to the neighbors after every sample of input data. As a result, the average bandwidth for communication among the neighboring nodes decreases by a factor equal to the block-size of the algorithm. In real time scenario the nodes in the sensor network follow a particular protocol for communication [111–113], where the communication time is much more than the processing time. The proposed block distributed algorithm, provides an excellent balance between the message transmission delay and processing delay, by increasing the interval between two messages and by increasing the computational load on each node in the interval between two successive transmissions. Therefore the main motivation here is to propose

communication-efficient block distributed LMS algorithms (both incremental and diffusion type) applicable for sensor networks. The performances of the proposed algorithms are analyzed and compare them with those of existing distributed LMS algorithms.

3.2 Problem Formulation

Consider a sensor network consisting of N nodes distributed over some region of interest. The set of nodes connected to node k (including itself because a node is always connected to itself) is called the neighborhood of node k and is denoted by $\mathcal{N}_k$. The number of neighbors of kth node is called degree of node k and is denoted by n_k.

Now, the objective is to estimate an $M \times 1$ unknown vector $\mathbf{w}^\circ$ from the measurements of N sensor nodes. The estimated weight vector of the kth node at time n is denoted as $\hat{\mathbf{w}}_k(n)$. Let $u_k(n)$ be the input data of kth node at time instant n then the input vector to the filter at time instant n is

$$\mathbf{u}_{k,n} = [u_k(n), u_k(n-1), \ldots u_k(n-M+1)]^T \tag{3.1}$$

The corresponding desired output of the node for the input vector $\mathbf{u}_k(n)$ is modeled as [83, 84]

$$d_k(n) = \mathbf{u}_{k,n}^T \mathbf{w}^\circ + v_k(n) \tag{3.2}$$

where $v_k(n)$ denotes a temporally and spatially uncorrelated white noise with variance $\sigma_{v,k}^2$. The regression and measurement data are collected across all nodes and is represented in form of two global matrices. By dropping the time index for compactness of notation, the global data across the network are as follows

$$\mathbf{U}_g = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_N]^T \tag{3.3}$$

$$\mathbf{d}_g = [d_1, d_2, \ldots, d_N]^T \tag{3.4}$$

Now the objective is to estimate the vector $\mathbf{w}^{\circ}$ that solves the cost function [18]

$$J(\mathbf{w}) = E\|\mathbf{d}_g - \mathbf{U}_g\mathbf{w}\|^2 \tag{3.5}$$

where E is the expectation operator. The cost function $J(\mathbf{w})$ is denotes the mean square error (MSE).

The optimum weight $\mathbf{w}^{\circ}$ can be estimated either by centralized approach or by distributed approach described in Section 1.2. The distributed methods like incremental LMS [18], diffusion LMS [32] need more communication overhead because each node communicate with other nodes after processing each data. In order to overcome this disadvantage in these distributed methods, the block distributed adaptive algorithm is proposed here.

3.2.1 Block Adaptive Distributed Solution

In the proposed approach, every node is modeled as a block adaptive linear filter where each node updates its estimated parameters by using the set of errors observed in the estimated output vector and broadcasts that to its neighbours. To define the sensor nodes data in block format, the literatures [18, 32, 84] are followed. The *block index* j is related to the time index n as

$$n = jL + i, \quad i = 0, 1, 2, \ldots, L-1, \quad j = 1, 2, \ldots$$

where L is the *block length*. The jth block contains time indices $n = [jL, jL + 1, \ldots, jL + L - 1]$. The input vectors of kth node for block j is combined to form a matrix given by

$$\mathbf{X}_k^j = [\mathbf{u}_{k,jL}, \mathbf{u}_{k,jL+1}, \ldots, \mathbf{u}_{k,jL+L-1}]^T \tag{3.6}$$

The corresponding desired response at jth block index of kth node is represented as

$$\mathbf{d}_k^j = [d_k(jL), d_k(jL+1), \ldots, d_k(jL+L-1)]^T \tag{3.7}$$

Let $\mathbf{e}_k^j$ represent the $L \times 1$ error signal vector for jth block of kth node, and is defined as [66, 67]

$$\mathbf{e}_k^j = \mathbf{d}_k^j - \mathbf{X}_k^j \hat{\mathbf{w}}_k^j \tag{3.8}$$

where $\hat{\mathbf{w}}_k^j$ denotes the estimated $M \times 1$ weight vector of the filter when jth block of the data acts as input at the kth node.

The regression input data and corresponding desired responses are distributed across all the nodes and are represented in two global matrices as

$$\mathbf{X}_{bg}^j = \text{col}\{\mathbf{X}_1^j, \mathbf{X}_2^j, \ldots, \mathbf{X}_N^j\} \tag{3.9a}$$

$$\mathbf{d}_{bg}^j = \text{col}\{\mathbf{d}_1^j, \mathbf{d}_2^j, \ldots, \mathbf{d}_N^j\} \tag{3.9b}$$

By using this global data, the block error vector for the whole network is [66, 67]

$$\mathbf{e}_{bg}^j = \mathbf{d}_{bg}^j - \mathbf{X}_{bg}^j \hat{\mathbf{w}} \tag{3.10}$$

For simpler mathematical analysis, henceforth the block index is dropped. Therefore the new notation for aforementioned variables along with their sizes are as follows: $\mathbf{d}_{bg}(NL \times 1)$, $\mathbf{X}_{bg}(NL \times M)$. The optimal weight vector $\hat{\mathbf{w}}$ can be estimated by minimizing the MSE function as [27, 84]

$$\min_{\hat{\mathbf{w}}} E \parallel \mathbf{d}_{bg} - \mathbf{X}_{bg} \hat{\mathbf{w}} \parallel^2 \tag{3.11}$$

where $\hat{\mathbf{w}}$ is a column vector of size $M \times 1$. Since the quantities are collected data across the network in block format therefore the cost function to be minimized is the block mean square error (BMSE). The BMSE is given by [27, 84]

$$BMSE = \frac{1}{L}[E[\mathbf{d}_{bg}^T \mathbf{d}_{bg}] - E[\mathbf{d}_{bg}^T \mathbf{X}_{bg}]\hat{\mathbf{w}} - \hat{\mathbf{w}}^T E[\mathbf{X}_{bg}^T \mathbf{d}_{bg}] - \hat{\mathbf{w}}^T E[\mathbf{X}_{bg}^T \mathbf{X}_{bg}]\hat{\mathbf{w}}] \tag{3.12}$$

The factor $1/L$ is used to find the mean square error for a single block [66]. Let the input regression data $\mathbf{u}$ is Gaussian and defined by the correlation function $r(l) = \sigma^2 \alpha^{|l|}$ in the covariance matrix, where α is the correlation index and σ^2 is the variance of the input regression data, then the relation between correlation

and cross-correlation quantities among blocked and unblocked data can be denoted as [66]

$$\mathrm{R}_X^{bg} = L\mathrm{R}_U^g,\ \mathrm{R}_{dX}^{bg} = L\mathrm{R}_{du}^g,\ \mathrm{R}_d^{bg} = L\mathrm{R}_d^g \tag{3.13}$$

where $\mathrm{R}_X^{bg} = E[\mathbf{X}_{bg}^T\mathbf{X}_{bg}]$, $\mathrm{R}_{dX}^{bg} = E[\mathbf{X}_{bg}^T\mathbf{d}_{bg}]$ and $\mathrm{R}_d^{bg} = E[\mathbf{d}_{bg}^T\mathbf{d}_{bg}]$, are the auto-correlation and cross-correlation matrices for global data in block form. Similarly the correlation matrices for unblocked data are defined as $\mathrm{R}_U^g = E[\mathbf{U}_g^T\mathbf{U}_g]$, $\mathrm{R}_{du}^g = E[\mathbf{U}_g^T\mathbf{d}_g]$ and $\mathrm{R}_d^g = E[\mathbf{d}_g^T\mathbf{d}_g]$ where the global distribution of data across the network is represented as $\mathbf{U}_g = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_N]^T$ and $\mathbf{d}_g = [d_1, d_2, \ldots, d_N]^T$. These relations are also valid for the data at individual nodes.

The BMSE in (3.12) is then reduced to

$$\begin{aligned} BMSE &= \frac{1}{L}[L\mathrm{R}_d^g - L\mathrm{R}_{du}^g\hat{\mathbf{w}} - L\hat{\mathbf{w}}^T\mathrm{R}_{du}^g - L\mathbf{w}^T\mathrm{R}_u^g\hat{\mathbf{w}}] \\ &= \mathrm{R}_d^g - \mathrm{R}_{du}^g\hat{\mathbf{w}} - \hat{\mathbf{w}}^T\mathrm{R}_{du}^g - \mathbf{w}^T\mathrm{R}_u^g\hat{\mathbf{w}} = MSE \end{aligned} \tag{3.14}$$

Comparing (3.14) with the MSE of conventional LMS for global data [18, 84], it can be concluded that the MSE in both the cases are same. Hence BLMS algorithm has similar properties as that of the conventional LMS algorithm. Now the equation (3.11) for block data can be reduced to a form similar to that of unblocked data as

$$\min_{\hat{\mathbf{w}}} E \parallel \mathbf{d}_g - \mathbf{U}_g\hat{\mathbf{w}} \parallel^2 \tag{3.15}$$

The basic difference between block and conventional LMS lies in the estimation of the gradient vector used in their respective implementation. The BLMS algorithm uses a more accurately estimated gradient because of using the time averaging. The accuracy increases with increase in block size [84]. Taking into account, the advantages of BLMS algorithm over conventional LMS algorithm, the distributed BLMS algorithm is proposed in the next subsection.

3.2.2 Adaptive Block Distributed Algorithms

In adaptive BLMS algorithm, each node k in the network receives the estimates from its neighboring nodes after each block of input data to adapt the local changes in

the environment. Two different types of distributed LMS algorithms in WSN have been reported in literature namely ILMS and DLMS [18, 19, 32]. These algorithms are based on conventional LMS for local learning process which in terms needs large communication resources. In order to achieve same performance with less communication resource, the block distributed LMS is proposed here.

The Block Incremental LMS (BILMS) Algorithm

In an incremental mode of cooperation, the information flows in a sequential manner from one node to the adjacent one in the network after processing of each sample of data [18, 30, 31]. The communications in the incremental way of cooperation can be reduced if each node need to communicate only after processing a block of data. For any block of data j, it is assumed that node k has access to the $\hat{\boldsymbol{w}}_{k-1}^{j}$ estimates from its predecessor node, as defined by the network topology and constitution. Based on these assumptions, the proposed BILMS algorithm can be derived by reducing the conventional ILMS algorithm ((16) in [18]) to a blocked data form as follows,

$$\begin{aligned}
\hat{\boldsymbol{w}}_{0}^{j} &= \hat{\boldsymbol{w}}^{j-1} \\
\hat{\boldsymbol{w}}_{k}^{j} &= \hat{\boldsymbol{w}}_{k-1}^{j} + \frac{\mu_k}{L} \sum_{q=0}^{L-1} \mathbf{u}_{k,jL+q} \left(d_k(jL+q) - \mathbf{u}_{k,jL+q}^{T} \hat{\boldsymbol{w}}_{k-1}^{j} \right) \\
&= \hat{\boldsymbol{w}}_{k-1}^{j} + \frac{\mu_k}{L} \mathbf{X}^{jT}_{k} \left(\mathbf{d}_{k}^{j} - \mathbf{X}_{k}^{j} \hat{\boldsymbol{w}}_{k-1}^{j} \right), \text{ for } k = 1, 2, \cdots N \\
\hat{\boldsymbol{w}}^{j} &= \hat{\boldsymbol{w}}_{N}^{j}
\end{aligned} \tag{3.16}$$

where μ_k is the local step size and L is the block size. The main steps of the BILMS algorithm are given in Algorithm 1.

The Block Diffusion LMS (BDLMS) Algorithm

Here each node k updates its estimate by using a simple local rule based on the average of its own estimates plus the information received from its neighbor $\mathcal{N}_k$. In this case, for every jth block of data at the kth node, the node has access to a set of estimates from its neighbors $\mathcal{N}_k$. Similar to BILMS algorithm, the proposed block diffusion strategy for a set of local combiners c_{kl} and for local step size μ_k can be

Algorithm 1: Main steps of BILMS algorithm

Setup Problem:
• Define WSNs with topology.
• Construct a path through the network which passes through all nodes just once.
• Define input signal power, correlation index for input signal and noise variance at each node.
• Find desire data at each node using (3.2).
• initialize the estimated weight $\hat{\boldsymbol{w}}^0 = 0$.
for *each block* **do**
  for *each node* **do**
    Receive the updated weights from the previous node;
    Calculate block of errors corresponding to block of data and update the weights using (3.16) ;
    Calculate the performance parameters;
    Send the updated weight to the next neighbour node;
  end
end
The Nth node has global estimated weight.

described as a reduced form of conventional DLMS algorithm [19, 32, 114] as

$$\boldsymbol{\theta}_k^{j-1} = \sum_{l \in \mathcal{N}_{k,j-1}} c_{kl} \hat{\mathbf{w}}_k^{j-1}, \qquad \boldsymbol{\theta}_k^{-1} = 0 \tag{3.17a}$$

$$\hat{\mathbf{w}}_k^{j} = \boldsymbol{\theta}_k^{j-1} + \frac{\mu_k}{L} \sum_{q=0}^{L-1} \mathbf{u}_{k,jL+q} \left(d_k(jL+q) - \mathbf{u}_{k,jL+q}^T \boldsymbol{\theta}_k^{j-1} \right) \tag{3.17b}$$

The weight update equation can be rewritten in more compact form by using the data in block format given in (3.6) and (3.7) as

$$\hat{\mathbf{w}}_k^{j} = \boldsymbol{\theta}_k^{j-1} + \frac{\mu_k}{L} \boldsymbol{X}_k^{j\,T} \left(\boldsymbol{d}_k^j - \boldsymbol{X}_k^j \boldsymbol{\theta}_k^{j-1} \right) \tag{3.18}$$

Comparing (3.17) with equation (19) in [32] it is concluded that the weight update equation is modified into block format.The main steps of the BILMS algorithm are given in Algorithm 2.

3.3 Performance Analysis of BDLMS Algorithm

The performance of an adaptive filter is evaluated in terms of its transient and steady state behaviors, which respectively provide the information about how fast and how well a filter is capable to learn. Such performance analysis are usually

Algorithm 2: Main steps of BDLMS algorithm

Setup Problem:
- Define WSNs with topology.
- Construct a path through the network which passes through all nodes just once.
- Calculate metropolis weight for diffusion.
- Define input signal power, correlation index for input signal and noise variance at each node.
- Find desire data at each node using (3.2).
- initialize the estimated weight $\hat{\boldsymbol{w}}^0 = 0$.

for *each block* **do**
  for *each node* **do**
    Receive the updated weights from the neighbouring nodes;
    Diffuse the received weights using (3.17a);
    Calculate block of errors corresponding to block of data and update the weights using (3.17b) ;
    Calculate the performance parameters;
    Send the updated weight to the neighbouring nodes;
  end
  Each node shares their updated weights.
end
Each node has the global estimated weights.

challenging in interconnected network because each node k is influenced by local data with local statistics $\{R_{dx,k}, R_{X,k}\}$, by its neighbourhood nodes through local diffusion and by local noise with variance $\sigma^2_{v,k}$. In case of block distributed system, the analysis becomes more challenging as it has to handle data in block form. The key performance metrics used in the analysis are MSD (mean square deviation), EMSE (excess mean square error) and MSE (mean square error) for local and also for global networks and are defined [27] as:

$$\eta_k^j = E\|\tilde{\mathbf{w}}_{k-1}^j\|^2 \quad (\mathbf{MSD}) \tag{3.19a}$$

$$\zeta_k^j = E\|e_{a,k}^j\|^2 \quad (\mathbf{EMSE}) \tag{3.19b}$$

$$\xi_k^j = E\|e_k(j)\|^2 = \zeta_k^j + \sigma_{v,k}^2 \quad (\mathbf{MSE}) \tag{3.19c}$$

and the local error signals such as weight error vector and *a priori* error at kth node for jth block are given as

$$\tilde{\mathbf{w}}_{k-1}^j = \mathbf{w}^o - \hat{\mathbf{w}}_{k-1}^j \tag{3.20}$$

$$e_{a,k}^{j} = \mathbf{u}_{k}^{j}\tilde{\mathbf{w}}_{k-1}^{j} \tag{3.21}$$

The algorithm described in (3.17) appears as the interconnection of block adaptive filters instead of conventional LMS adaptive algorithm among all the nodes across the network. As shown in (3.14) the BLMS algorithm has similar properties to those of the conventional LMS algorithm, the convergence analysis of the proposed BDLMS algorithm can be carried out in similar to the DLMS algorithm described in [27, 32].

The estimated weight vector for jth block across the network is defined as

$$\hat{\mathbf{w}}^{j} = [\hat{\mathbf{w}}_{1}^{j}; \ldots; \hat{\mathbf{w}}_{N}^{j}]$$

Let C is the $N \times N$ metropolis with entries $[c_{kl}]$, then the global transaction combiner matrix G is defined as $G = C \otimes I_M$. The diffusion global vector for jth block is defined as

$$\boldsymbol{\theta}^{j} = G\hat{\mathbf{w}}^{j} \tag{3.22}$$

Now the input data vector at jth block is defined as

$$\mathbf{X}^{j} = \text{diag}\{\mathbf{X}_{1}^{j}, \ldots, \mathbf{X}_{N}^{j}\}$$

The desired block response at each node k is assumed have to follow the traditional data model used in literature [27, 83, 84] i.e.

$$\mathbf{d}_{k}^{j} = \mathbf{X}_{k}^{j}\mathbf{w}^{\circ} + \mathbf{v}_{k}^{j} \tag{3.23}$$

where $\mathbf{v}_{k}^{j}$ is the background noise vector of length L. The noise is assumed to be spatially and temporarily independent with variance $\sigma_{v,k}^{2}$. Using blocked desired response for single node (3.19) the global response for kth block can be modeled as

$$\mathbf{d}_{bg}^{j} = \mathbf{X}_{bg}^{j}\mathbf{w}_{g}^{\circ} + \mathbf{v}^{j} \tag{3.24}$$

where $\mathbf{w}_{g}^{\circ}$ is the optimum global weight vector defined for every node and is written

as $\mathbf{w}_g^\circ = [\mathbf{w}^\circ; , \ldots , ; \mathbf{w}^\circ]$ and

$$\mathbf{v}^j = [\mathbf{v}_1^j; , \ldots , ; \mathbf{v}_N^j] \qquad (LN \times 1)$$

is the additive Gaussian noise for jth block index.

Using the relations defined above, the block diffusion strategy in (3.17) can be written in global form as:

$$\hat{\mathbf{w}}^j = \boldsymbol{\theta}^{j-1} + \frac{1}{L}\mathbf{S}\mathbf{X}^j(\mathbf{d}_{bg}^j - \mathbf{X}^j\boldsymbol{\theta}^{j-1}) \tag{3.25}$$

where the stepsizes for all the nodes are embedded in a matrix $\mathbf{S}$

$$\mathbf{S} = \text{diag}\{\mu_1 \mathrm{I}_M, \mu_2 \mathrm{I}_M, \ldots, \mu_N \mathrm{I}_M\} \qquad (NM \times NM) \tag{3.26}$$

Using (3.22) it can be written as

$$\hat{\mathbf{w}}^j = G\hat{\mathbf{w}}^{j-1} + \frac{1}{L}\mathbf{S}\mathbf{X}^j(\mathbf{d}_{bg}^j - \mathbf{X}^j G\hat{\mathbf{w}}^{j-1}) \tag{3.27}$$

3.3.1 Mean Transient Analysis of BDLMS Algorithm

The mean behavior of the proposed BDLMS algorithm is similar to DLMS algorithm given in [32]. The mean error vector signal is given as

$$E[\tilde{\mathbf{w}}^j] = \left(\mathrm{I}_{NM} - \frac{1}{L}\mathbf{S}\mathrm{R}_X\right) GE[\tilde{\mathbf{w}}^{j-1}] \tag{3.28}$$

where $\mathrm{R}_X = \text{diag}\{\mathrm{R}_{X,1}, \mathrm{R}_{X,2}, \ldots, \mathrm{R}_{X,N}\}$ is a block diagonal matrix and

$$\mathrm{R}_{X,k} = E[{\mathbf{X}_k^j}^T \mathbf{X}_k^j] = LE[\mathbf{U}_k^T \mathbf{U}_k] = L\mathrm{R}_U$$

Hence (3.28) can be written as

$$E[\tilde{\mathbf{w}}^j] = (\mathrm{I}_{NM} - \mathbf{S}\mathrm{R}_U)\, GE[\tilde{\mathbf{w}}^{j-1}] \tag{3.29}$$

Comparing (3.29) with that of DLMS algorithm ((31) in [32]), it is observed that both BDLMS and DLMS algorithms yield the same characteristic equation for the convergence of mean; and hence it can be concluded that block diffusion protocol defined in (3.17) has the same stabilizing effect on the network as DLMS. Ideally

the initial and final values of the error vector $E[\tilde{\mathbf{w}}^j]$ are $\mathbf{w}^o$ and zero. The duration of the transient depends on the step size.

3.3.2 Mean-Square Transient Analysis of BDLMS Algorithm

Since the ideal steady state value of $E[\tilde{\mathbf{w}}^j]$ is zero, the mean square is same as the variance for the error vector. The variance estimate is a key performance indicator in mean-square transient analysis of any adaptive system. The variance relation for block data is similar to that of conventional DLMS algorithm.

$$E\| \tilde{\mathbf{w}}^j \|_{\Sigma}^2 = E\| \tilde{\mathbf{w}}^{j-1} \|_{\Sigma'}^2 + \frac{1}{L^2} E[\mathbf{v}^{j^T} \mathbf{X}^j \mathbf{S}\Sigma\mathbf{S}\mathbf{X}^{j^T} \mathbf{v}^j] \tag{3.30}$$

$$\Sigma' = G^T\Sigma G - \frac{1}{L} G^T\Sigma\mathbf{S}E[\mathbf{X}^{j^T}\mathbf{X}^j]G - \frac{1}{L} G^T E[\mathbf{X}^{j^T}\mathbf{X}^j]\mathbf{S}\Sigma G + \frac{1}{L^2} G^T E[\mathbf{X}^{j^T}\mathbf{X}^j]\mathbf{S}\Sigma\mathbf{S}E[\mathbf{X}^{j^T}\mathbf{X}^j]G \tag{3.31}$$

Using $E[\mathbf{X}^{j^T}\mathbf{X}^j] = LE[\mathbf{U}^{j^T}\mathbf{U}^j]$ from the definition in (3.31), we obtain

$$\Sigma' = G^T\Sigma G - G^T\Sigma\mathbf{S}E[\mathbf{U}^{j^T}\mathbf{U}^j]G - G^T E[\mathbf{U}(j)^T\mathbf{U}^j]\mathbf{S}\Sigma G + G^T E[\mathbf{U}^{j^T}\mathbf{U}^j]\mathbf{S}\Sigma\mathbf{S}E[\mathbf{U}^{j^T}\mathbf{U}^j]G \tag{3.32}$$

which is similar to (45) in [32]. Using the properties of expectation and trace [27] the second term of (3.30) is solved as

$$\begin{aligned}
&\frac{1}{L^2} E\mathbf{v}^{j^T}\mathbf{X}^j\mathbf{S}\Sigma\mathbf{S}\mathbf{X}^{j^T}\mathbf{v}^j \\
&= \frac{1}{L^2} E\left[\| \mathbf{X}_j \|_{\mathbf{S}\Sigma\mathbf{S}}^2 \mathbf{v}_j^T\mathbf{v}_j\right] \\
&= \frac{1}{L^2} E\left[\| \mathbf{X}_j \|_{\mathbf{S}\Sigma\mathbf{S}}^2\right] E\left[\mathbf{v}_j^T\mathbf{v}_j\right] \\
&= \frac{1}{L^2} E\left[\mathrm{tr}\left[\mathbf{X}^j\mathbf{S}\Sigma\mathbf{S}\mathbf{X}^{j^T}\right]\right] E\left[\mathbf{v}^{j^T}\mathbf{v}^j\right] = E\left[v^{j^T}\mathbf{U}^j\mathbf{S}\Sigma\mathbf{S}\mathbf{U}^{j^T}v^j\right]
\end{aligned} \tag{3.33}$$

where the noise variance vector v^j is not in block form and it is assumed that the noise is stationary Gaussian. Equations (3.30) and (3.31) may therefore be written

as

$$E\| \tilde{\mathbf{w}}^j \|_{\Sigma}^2 = E\| \tilde{\mathbf{w}}^{j-1} \|_{\Sigma'}^2 + \frac{1}{L^2} E v^{j^T} \mathbf{U}^j \mathbf{S} \Sigma \mathbf{S} \mathbf{U}^{j^T} v(j) \tag{3.34}$$

$$\Sigma' = G^T \Sigma G - G^T \Sigma \mathbf{S} E(\mathbf{U}^{j^T} \mathbf{U}^j) G - G^T E(\mathbf{U}^{j^T} \mathbf{U}^j) \mathbf{S} \Sigma G + G^T E(\mathbf{U}^{j^T} \mathbf{U}^j) \mathbf{S} \Sigma \mathbf{S} E(\mathbf{U}^{j^T} \mathbf{U}^j) G \tag{3.35}$$

It may be noted that variance estimate (3.35) for BDLMS algorithm is exactly the same as that of DLMS. In the BDLMS algorithm, the local step-size is chosen to be L times that of the local step-size of DLMS algorithm in order to have the same level of performance. As the proposed algorithm and the DLMS algorithm have similar properties, the evolution of their variances is also similar. Therefore the recursion equation of the global variances for BDLMS is similar to (73) and (74) in [32]. In the same way the local node performance is similar to (89) and (91) of [32].

3.3.3 Learning Behavior of BDLMS Algorithm

The learning behaviour of BDLMS algorithm is examined using simulation study. The characteristic or variance curves are plotted for BDLMS algorithm and are compared with that of DLMS algorithm. The row regressors with shift invariance input [27] are used with each regressor having data as

$$\mathbf{u}_{k,i} = [u_k(i), u_k(i-1), \ldots, u_k(i-M+1)]^T \tag{3.36}$$

In BLMS, the regressors for $L = 3$ and $M = 3$ are given by

$$\mathbf{X}_k(1) = \begin{pmatrix} u_k(1) & 0 & 0 \\ u_k(2) & u_k(1) & 0 \\ u_k(3) & u_k(2) & u_k(1) \end{pmatrix}$$

$$\mathbf{X}_k(2) = \begin{pmatrix} u_k(4) & u_k(3) & u_k(2) \\ u_k(5) & u_k(4) & u_k(3) \\ u_k(6) & u_k(5) & u_k(4) \end{pmatrix} \tag{3.37}$$

The desired data are generated according to the model given in literature [27]. The unknown vector $\mathbf{w}^\circ$ is set to $[1, 1, \ldots, 1]^T/\sqrt{M}$.

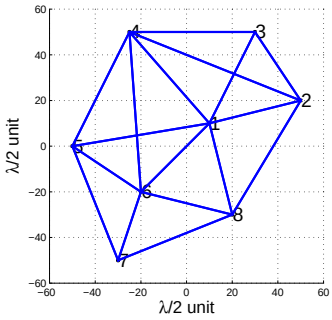


Figure 3.1: Network topology used for the simulation of BDLMS algorithm

The input sequence $\{u_k(n)\}$ is assumed to be spatially correlated and is generated using a first-order auto-regressive model with transfer function $\sqrt{\sigma^2(1-\alpha^2)}/(1-\beta z^{-1})$ [27]

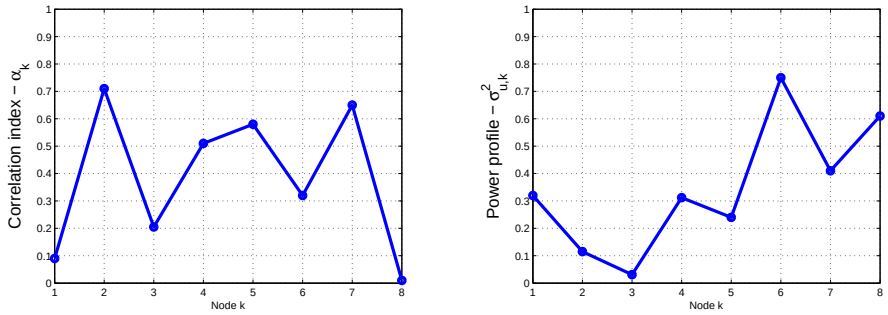
$$u_k(n) = \alpha_k \cdot u_k(n-1) + \beta_k \cdot v_k(n), \qquad i > -\infty \tag{3.38}$$

Here, $\alpha_k \in [0,1)$ is the correlation index and $v_k(n)$ is a spatially independent white Gaussian process with unit variance and $\beta_k = \sqrt{\sigma^2_{u,k} \cdot (1-\alpha_k^2)}$. The regressors power profile is given by $\{\sigma^2_{u,k}\} \in (0,1]$. The resulting regressors have Toeplitz covariance with co-relation sequence $r_k(i) = \sigma^2_{u,k} \cdot (\alpha_k)^{|i|}, i = 0,1,2,\ldots,M-1$.

Figure 3.1 shows an eight node network topology used in the simulation study. The randomly generated input correlation index α_k and and input signal power $\sigma^2_{u,k}$ are shown in Figure 3.2. The variable β_k is computed using α_k and $\sigma^2_{u,k}$ shown in Figure 3.2. The input power profile shown in Figure 3.2(b) tries to imitate the practical scenario faced in WSNs. These values of input power along with the noise variance are used to obtain the signal to noise ratio profile.

3.3.4 The Simulation Conditions

The algorithm is valid for any block of length greater than one [66], while $L = M$ is the most preferable and optimum choice. The back ground noise is assumed to be Gaussian white noise of variance $\sigma^2_{v,k} = 10^{-3}$ and the data used in the study is generated using (3.2). The step size μ is 0.05 and 0.5 for DLMS and BDLMS respectively. In order to generate the performance curves, 100 independent experi-



(a) (b)

Figure 3.2: Network statistics used for the simulation of BDLMS algorithm. (a) Network co-relation index per node. (b) Regressor power profile.

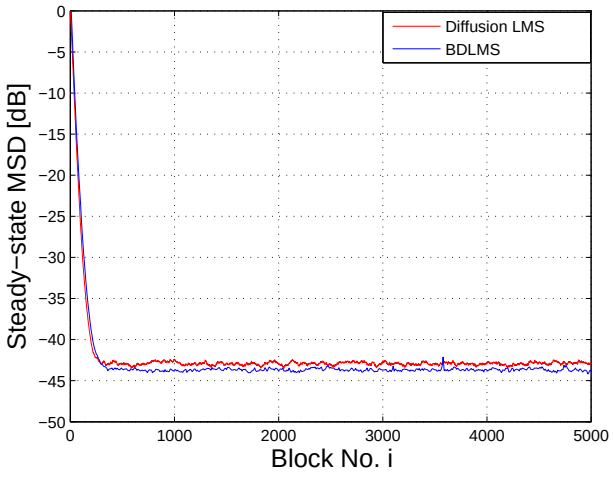


Figure 3.3: Global MSD curve for DLMS and BDLMS algorithms.

ments are performed and averaged. The global MSD curve is shown in Figure 3.3. This is obtained by averaging $E \parallel \tilde{\mathbf{w}}_k^{j-1} \parallel^2$ across all the nodes over 100 experiments. Similarly, the global EMSE curve obtained by averaging $E \parallel \mathbf{e}^{\mathbf{j}}_{a,k} \parallel^2$, where $\mathbf{e}^{\mathbf{j}}_{a,k} = \mathbf{x}_k^j \tilde{w}_k^{j-1}$, across all the nodes is displayed in Figure 3.4. The global MSE is depicted in Figure 3.5. It shows that in both the cases the global MSE curve is exactly matching.

Since the weights are updated and then communicated for local diffusion after every L data samples, the number of communications between neighbours are reduced by L times compared to that of the DLMS case where the weights are updated and communicated after each sample of data.

The global performances are the contributions of all individual nodes and it is

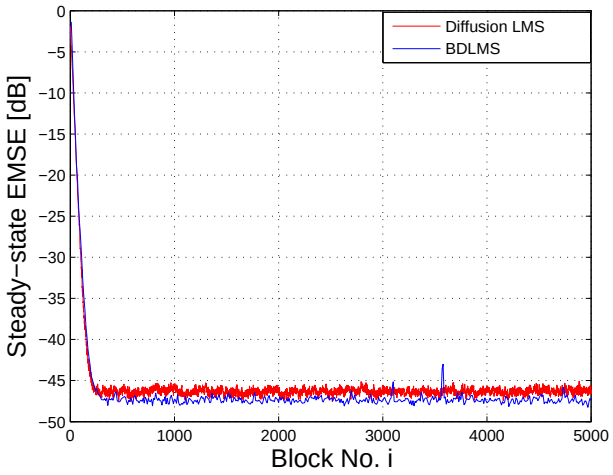


Figure 3.4: Global EMSE curve for DLMS and BDLMS algorithms.

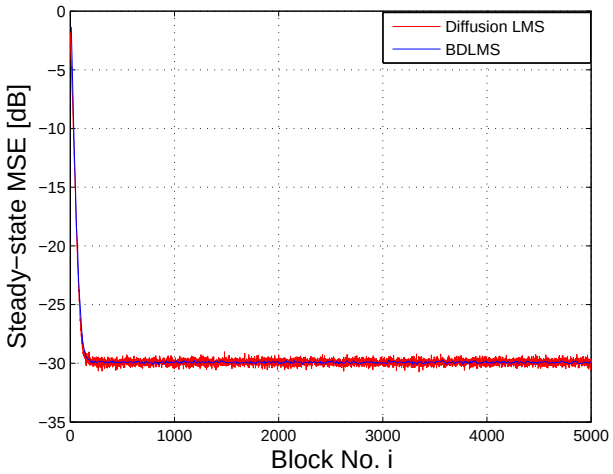
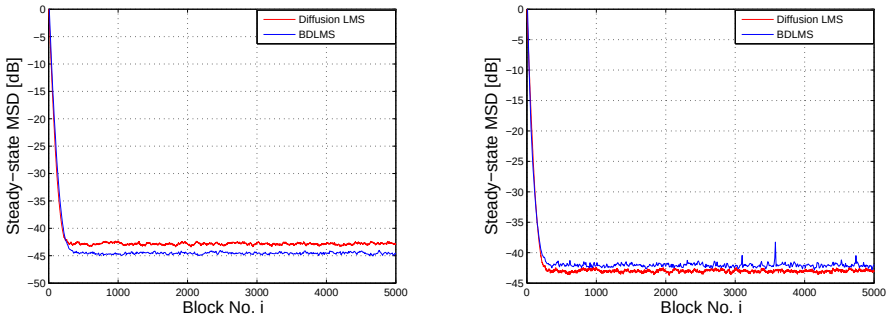


Figure 3.5: Global MSE curve for DLMS and BDLMS algorithms.

obtained by taking the mean performance of all the nodes. The simulation results are provided to compare with that obtained by DLMS algorithm for individual node. The local MSD evolution at nodes 1 and 5 are given in Figure 3.6(a) and Figure 3.6(b) respectively. Similarly, the local EMSE evolution at nodes 1 and 7 is depicted in Figure 3.7. The convergence speed is nearly same in both MSD and EMSE evolution, but the performance is slightly degraded in case of BDLMS algorithm. The detoriation of performance in case of proposed BDLMS algorithm can be compensated with the huge reduction in of communication bandwidth.



(a) (b)

Figure 3.6: Local MSD: comparison between DLMS and BDLMS algorithms. (a) MSD curve at node 1. (b) MSD curve at node 7.

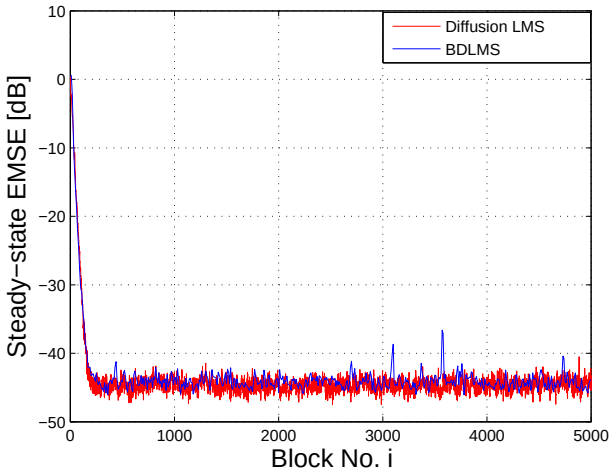


Figure 3.7: Local EMSE at nodes 7 for the same network

3.4 Performance Analysis of BILMS Algorithm

To show that the proposed BILMS algorithm has guaranteed convergence, the steady-state performance analysis of the algorithm using the same data model as the one which is commonly used in the conventional sequential adaptive algorithms [31, 115, 116] is carried out.

Following the steps in [27] the weight-energy relation is derived by using the definition of weighted *a priori* and *a posteriori* error

$$\|\tilde{\boldsymbol{w}}_k^j\|_\Sigma^2 + \frac{|e^{j}_{a,k}{}^{\Sigma}|^2}{\|\mathbf{X}_k^j\|_\Sigma^2} = \|\tilde{\boldsymbol{w}}_{k-1}^j\|_\Sigma^2 + \frac{|e^{j}_{p,k}{}^{\Sigma}|^2}{\|\mathbf{X}_k^j\|_\Sigma^2} \tag{3.39}$$

Since (3.39) is similar to that of (35) in [18] the performance of BILMS algorithm

is similar to that of ILMS algorithm. The variance expression is obtained from the energy relation (3.39) by replacing a *posteriori error* by its equivalent expression, and then averaging both the sides,

$$E[\|\tilde{\boldsymbol{w}}_k^j\|_\Sigma^2] = E[\|\tilde{\boldsymbol{w}}_{k-1}^j\|_{\Sigma'}^2] + |\frac{\mu_k}{L}|^2 E[V^j{}_k^T \mathbf{X}_k^j \Sigma \mathbf{X}^j{}_k^T V_k^j]$$
$$\Sigma' = \Sigma - \frac{\mu_k}{L}\left(\Sigma \mathbf{X}^j{}_k^T \mathbf{X}_k^j + \mathbf{X}^j{}_k^T \mathbf{X}_k^j \Sigma\right) + |\frac{\mu_k}{L}|^2 \mathbf{X}^j{}_k^T \mathbf{X}_k^j \Sigma \mathbf{X}^j{}_k^T \mathbf{X}_k^j \qquad (3.40)$$

The variance relation in (3.40) is similar to that of ILMS algorithm in [18]. The performance of ILMS algorithm is studied in detail in literature. It is observed that the theoretical performance of BILMS algorithm and conventional ILMS algorithm are similar because both have the same variance expressions. Simulation results provide the validation of this analysis.

3.4.1 Simulation Results of BILMS Algorithm

From the simulation study of BILMS algorithm and to facilitate comparison on the performance the same desired data have been used in the case of BDLMS algorithm. The time correlated sequences are generated at every node according to the network statistics. The same network has been chosen as defined for block diffusion network

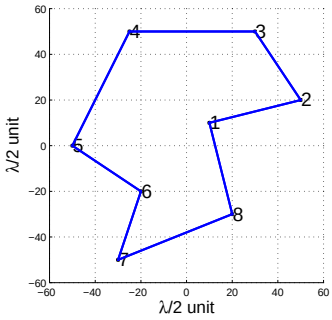


Figure 3.8: Network Topology

in Section 3.3.3. The ring topology shown in Figure 3.8 is used for the simulation study. It is assumed that the background noise to be temporarily and spatially uncorrelated additive white Gaussian noise with a variance of $\sigma_{v,k}^2 = 10^{-3}$. The learning curves are obtained by averaging the performance of 100 independent experiments, generated by 5,000 samples in the network. The results are obtained

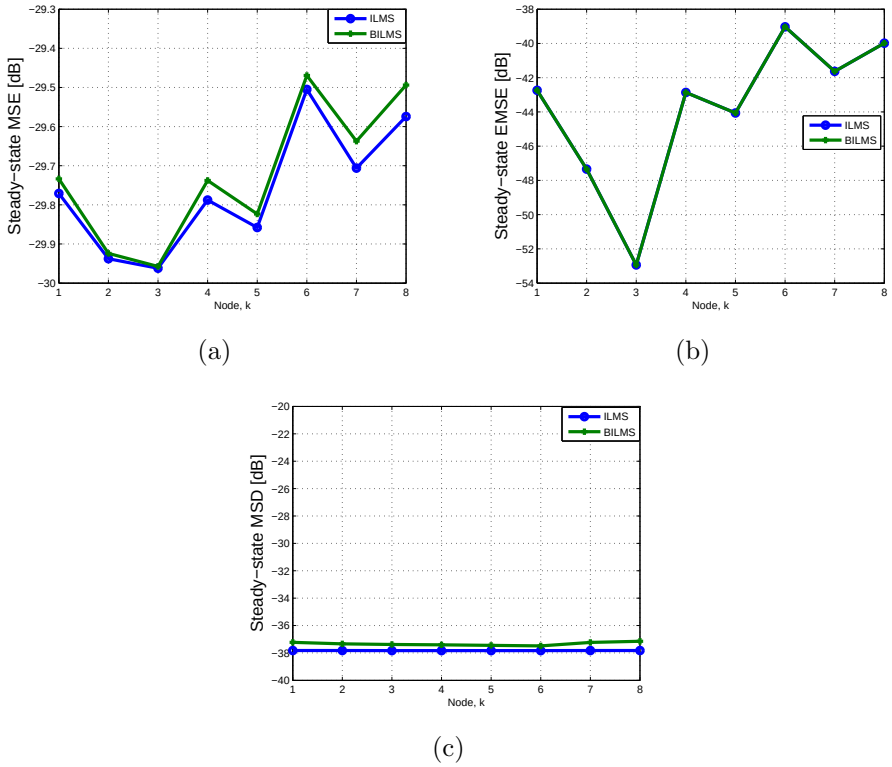


Figure 3.9: Network nodal performance. (a) MSE versus node. (b) EMSE versus node. (c) MSD versus node.

by averaging the last 50 samples of the corresponding learning curves. It can be observed from figures that the steady state performances at different nodes of the network achieved by BILMS are in close agreement with that of ILMS algorithm. The EMSE plots which are more sensitive to local statistics are depicted in Figures 3.9(a) and 3.9(b). A good match between BILMS and ILMS is observed from these plots. In [18], the authors have already proved the theoretical matching of steady-state nodal performance with simulation results. As the MSE roughly reflects the noise power and the plot indicates the good performance of the adaptive network, it may be inferred that the adaptive node performs well in the steady state.

The global MSD curve shown in Figure 3.10 is obtained by averaging $E \parallel \tilde{\boldsymbol{\psi}}_k^{(j-1)} \parallel^2$ across all the nodes and over 100 experiments. Similarly the global EMSE and MSE plots are displayed in Figure 3.11 and 3.12, respectively. These are obtained by averaging $E \parallel \mathbf{e}_{a,k}(j) \parallel^2$, where $\mathbf{e}_{a,k}(j) = \mathbf{u}_{k,j}\tilde{\boldsymbol{\psi}}_k^{(j-1)}$ across all the nodes.

If the weights are updated after L data points and then communicated for local

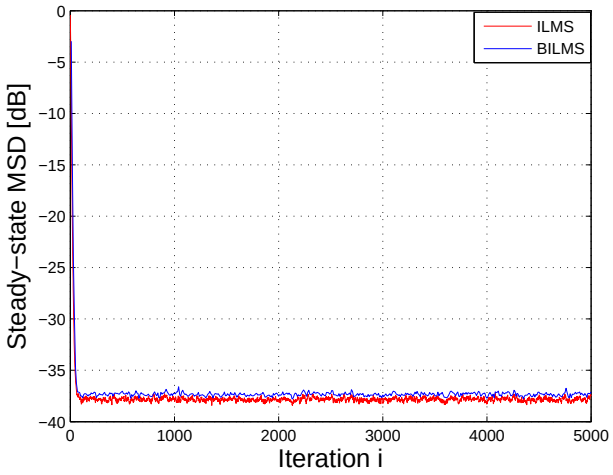


Figure 3.10: Global MSD curve for ILMS and BILMS algorithms.

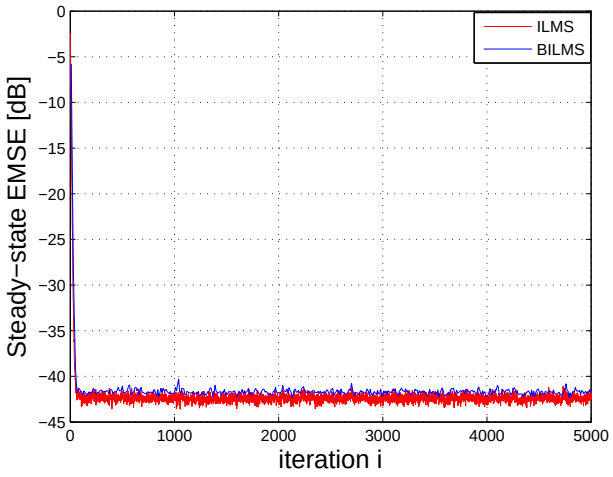


Figure 3.11: Global EMSE curve for ILMS and BILMS algorithms.

diffusion, the number of communications between neighbors is reduced by L times than that of ILMS where the weights are updated after processing each sample of data. Therefore, similar to BDLMS the communication overhead in BILMS algorithm also get reduced by L times than that of ILMS algorithm.

The performance comparison between two proposed algorithms BDLMS and BILMS algorithms for the same network are shown in Figures 3.13-3.15. One can observe from Figure 3.13 that the MSE for BILMS algorithm converges faster than BDLMS. Since same noise model is used for both the algorithms, therefore after convergence the steady state performance is same for both the cases. But in case

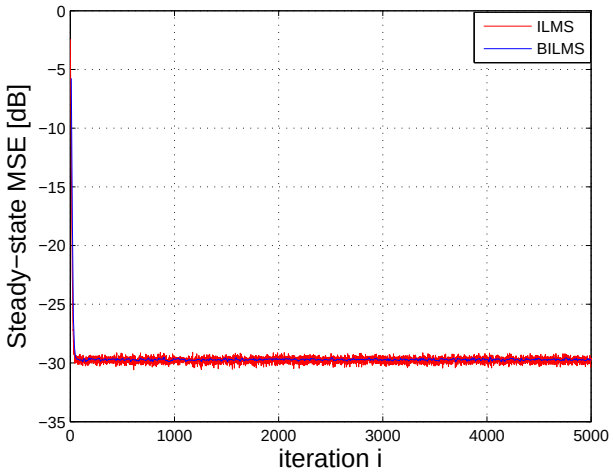


Figure 3.12: Global MSE curve for ILMS and BILMS algorithms.

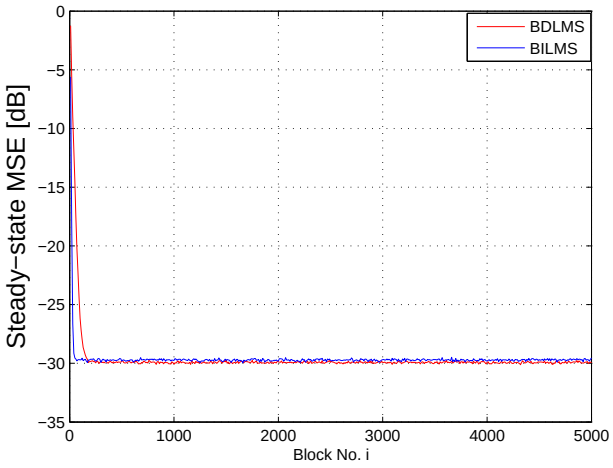


Figure 3.13: Global MSE curve for BILMS and BDLMS algorithms.

of MSD and EMSE performance of Figures 3.14 and 3.15, a little difference is observed. It is because different cooperation schemes are used for different algorithms. However, the diffusion cooperation scheme is more adaptive to the environmental change compared to the incremental cooperation. But more number of communications overhead is required for BDLMS than the BILMS algorithm.

3.5 Performance Comparison

In this section, an analysis of communication cost and latency is presented to have a theoretical comparison of the performance of distributed LMS with block distributed

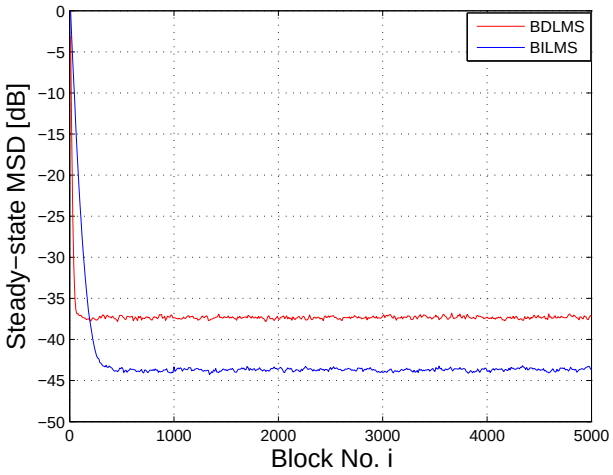


Figure 3.14: Global MSD curve for BILMS and BDLMS algorithms.

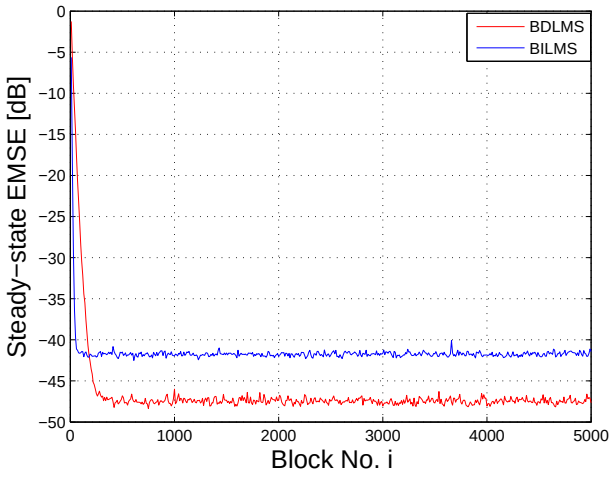


Figure 3.15: Global EMSE curve for BILMS and BDLMS algorithms.

LMS algorithm.

3.5.1 Analysis of Communication Cost

Assuming that the messages are of fixed bit-length, the communication cost is modeled as the number of messages transmitted to achieve the steady-state value in the network. Let N be the number of nodes in the network and M be the filter length. The block length L is chosen to be the same as the filter length. Let h be the average time required for the transmission of one message i.e. for one communication between the nodes [117–119].

ILMS and BILMS Algorithms

In the incremental mode of cooperation, every node sends its own estimated weight vector to its adjacent node in a unidirectional cyclic manner. Since at any instant of time, only one node is active/allowed to transmit to only one designated node, the number of messages transmitted in one complete cycle is $N-1$. Let K be the number of cycles required to attain the steady state value in the network. Therefore, the total number communications required to converge the system to steady state is given by

$$C_{ILMS} = (N-1)K \tag{3.41}$$

In case of BILMS also, at any instant of time, only one node in the network is active/allowed to transmit to one designated follower node, as in the case of ILMS. But, in case of BILMS algorithm, each node sends its estimated weight vector to its follower node in the network after an interval of L sample periods. Therefore, the number of messages sent by a node in this case is reduced to K/L, and accordingly, the total communication cost is given by

$$C_{BILMS} = (N-1)K/L \tag{3.42}$$

DLMS and BDLMS Algorithms

The diffusion-based algorithms are communication intensive. In DLMS mode of cooperation, in each cycle, each node in the network sends its estimated information to all its connected nodes in the network. So the total number of messages transmitted by all the nodes in a cycle is

$$c = \sum_{i=1}^{N} n_i \tag{3.43}$$

where n_i is the number of nodes connected to the ith node and the total communication cost to attain convergence is given by

$$C_{DLMS} = cK \tag{3.44}$$

In this proposed block diffusion strategy, the number of connected nodes n_i and the total size of the messages remain the same as that of DLMS algorithm. But, in BDLMS algorithm, each node distributes the message after processing L data samples. Therefore the communication is reduced by a factor equal to the block length and the total communication cost in this case is given by

$$C_{BDLMS} = cK/L \tag{3.45}$$

3.5.2 Analysis of Duration for Convergence

The time interval between the arrival of input to a node and the time of reception of corresponding updates by the designated node(s) may be assumed to be comprised of two major components: processing delay to perform the necessary computations in a node to obtain the estimates to be updated and the communication delay involved in transferring the message to the receiver node(s). The processing delay depends on the hardware architecture of the nodes to perform the computation which could be widely varying. But, without losing much of the generality of analysis, it is assumed that each node has M parallel multipliers and one full adder to implement the LMS algorithm. Let T_M and T_A be the time required for executing a multiplication and an addition, respectively. Therefore, the processing delay needed for single update in LMS is

$$D = 2T_M + (M+1)T_A \tag{3.46}$$

The communication delay is mostly due to the implementation of protocols for transmission and reception, which remains almost same for different nodes. The location of nodes will not have any major contribution to the delay unless the destination node is far apart and a relay node is required to make the message reach the destination. In this backdrop it is assumed that the same average delay h is required to transfer each message for all receiver-transmitter pairs in the network.

Estimation of Delays for the ILMS and BILMS Algorithms

In case of ILMS, the duration of each updating cycle by all the nodes is

$$ND + (N-1)h \tag{3.47}$$

and the total duration for convergence of the network is given as

$$L_{ILMS} = [ND + (N-1)h]K \tag{3.48}$$

If the same hardware as that of ILMS is used for the implementation of BILMS, the delay for processing one block of data is $2MT_M + M(M+1)T_A = MD$. Then the duration of one cycle of update by the BILMS is $N\{2MT_M + M(M+1)T_A\} + (N-1)h$ and the duration of convergence of this algorithm is

$$L_{BILMS} = [NMD + (N-1)h]K/L \tag{3.49}$$

For $L = M$, the above expression is reduced to

$$L_{BILMS} = \Big[ND + \frac{(N-1)h}{L}\Big]K \tag{3.50}$$

Comparing (3.50) with (3.48), it is found that in BILMS the processing delay remains the same as that in ILMS, but the communication overhead is reduced by L times.

Estimation of Delays for the DLMS and BDLMS Algorithms

It is also assumed in case of DLMS algorithm that the average delay h in the arrival of the updates of a node at the other connected nodes in the network is assumed. Therefore, the communication delay remains the same as that of ILMS algorithm but in this case it needs more processing delay to process the unbiased estimates received from the connected neighbouring nodes. The total communication delay in a cycle in this case is given by $cT_A + NT_M$, where c is the total number of messages transferred in a cycle given by (3.43). Now the total duration of a cycle in DLMS algorithm with same hardware constraints is given by

$$L_{DLMS} = [cT_A + NT_M + ND + (N-1)h]K \tag{3.51}$$

Table 3.1: Comparison of Performances of Distributed Algorithms and Block Distributed Algorithms

Parameter	ILMS Algorithm	BILMS Algorithm	DLMS Algorithm	BDLMS Algorithm
Communication Cost	$(N-1)K$	$(N-1)K/L$	cK	cK/L
Duration of convergence	$[ND + (N-1)h]K$	$[NMD + (N-1)h]K/L$	$[cT_A + NT_M + ND + (N-1)h]K$	$[cT_A + NT_M + ND + (N-1)h]K/L$

Table 3.2: Numerical Comparison of Performances of Sequential and Block Distributed Adaptive Algorithms

Parameter	ILMS Algorithm	BILMS Algorithm	DLMS Algorithm	BDLMS Algorithm
Communication Cost	950	95	4500	450
Duration of Convergence	$9.5s$	$0.95s$	$9.5s$	$0.95s$

In case of DBLMS, the total communication delay per cycle is reduced by a factor of L and is expressed as

$$L_{BDLMS} = [cT_A + NT_M + NMD + (N-1)h]K/L \tag{3.52}$$

The mathematical expressions of communication cost and latency for the distributed LMS and the block distributed LMS algorithms are summarized in Table 3.1. A numerical example is given in Table 3.2 to show the advantage of block distributed algorithms over the sequential distributed algorithms. The authors have simulated the hardware for 8-bit multiplication and addition in TSMC 90 nm. The multiplication and addition time are found to be $T_A = 10^{-5}ns, T_M = 10^{-3}ns$. It is assumed that the transmission delay is $h = 10^{-2}s$. The convergence curves obtained from the simulation study shows that the network attains steady-state value after 250 input data DLMS and 50 input data in case of ILMS. The filter length M as well as the block size L are taken to be 10 in the numerical study.

3.6 Conclusion

This chapter proposes the block implementation of distributed LMS algorithms for WSNs. The theoretical analysis and the corresponding simulation results demonstrate that the performance of the block distributed LMS algorithms is similar to that of the sequential distributed LMS. The significant achievement of the proposed algorithms is that a node performs L (block size) times lesser communication compared to conventional sequential distributed LMS algorithms. This would be of greater advantage to reduce the communication bandwidth and power consumptions involved in the transmission and reception of messages across the resource-constraint nodes in a WSN. In the coming years, with continuing advances in microelectronics, enough computing resources in the nodes can be accommodated to reduce the processing delays in the nodes, but the communication bandwidth and communication delay could be the major operational bottlenecks in WSNs. Therefore the proposed block formulation would certainly be advantageous compared to its sequential counterpart.

Chapter 4

Robust Distributed Estimation in Wireless Sensor Networks

Chapter 4

Robust Distributed Estimation in Wireless Sensor Networks

The cooperative schemes conventionally used in sensor network give better performance when the noise is considered as Gaussian. In general practice the data collected by sensor nodes over a geographical region are contaminated with Gaussian as well as impulsive noise. Under that situation the gradient based distributed adaptive estimation algorithms exhibit poor performance. To alleviate this shortcoming a robust distributed strategy is proposed here based on error saturation nonlinearity in impulsive noise environment. The proposed strategy is introduced first in incremental cooperative distributed network to estimate the desired parameters in presence of faulty sensor node and Gaussian contaminated impulsive noise. The steady-state analysis of saturation nonlinearity ILMS (SNILMS) is carried out by employing spatial-temporal energy conservation principle. Both the theoretical as well as the simulation result show that the proposed algorithm is robust to impulsive noise. Further the robust estimation techniques are incorporated in diffusion way of estimation. The rank based cost function known as Wilcoxon norm is used. For this a robust BDLMS is formulated. The simulation results show that the error saturation nonlinearity diffusion LMS (SNDLMS) provides better performance compared to that of Wilcoxon norm diffusion LMS (WNDLMS) algorithm.

4.1 Introduction

Substantial effort is required to devise algorithms which would be able to improve the estimation performance of the parameters of interest by employing every node and exchanging local estimates between them. During estimation process each node tries to optimize a cost function that depends on all information in the network [22, 23]. The main challenges in optimizing such functions are (i) no node has direct access to all the information, (ii) the network topology can change over time (due to link failures, change in the node position, and/or reachability problems) and (iii) the presence of impulsive noise.

The first challenge can be overcome by centralized processing where all the nodes send their data to the central processor by using either single hop or multi hop communication. To overcome the disadvantages of centralized processing distributed signal processing is proposed, especially in large-scale WSNs systems [12, 17, 24, 41]. In distributed approach, every node in the network communicates with a subset of the nodes, and processing task is distributed among all the nodes in the network. The performances of distributed algorithms depend on the mode of cooperation among the nodes. The incremental and diffusion algorithm are the two major co-operative techniques used in WSNs. In an incremental mode of co-operation, a cyclic path through the network is required so that information is communicated sequentially from one node to another [31, 115, 116]. On the other hand in the diffusion method, the nodes communicate with all of their neighbors as dictated by the network topology [32]. Both the diffusion as well as incremental strategies are robust to the change in topology. Because in case of incremental cooperation only a sequential path is needed this can be reconstituted after change in the topology. Similarly in [35] author mentioned that the diffusion adaptive strategy can work for change in networks topology environment.

It is known that when data is contaminated with non-Gaussian noise, the conventional estimation algorithms that minimize least square or mean square criterion yields poor performance. This leads to a new domain of research in communication systems, where the performance is limited by the interference of impulsive noise. In

many physical environments the additive noise is modeled as impulsive or Gaussian mixture [73]. Nonlinear techniques are employed to reduce the effect of impulsive interference on the systems.

Considering the current research work discussed in Chapter 2, a new generalized distributed algorithm which would offer robustness to impulsive noise is to be developed. This study has given rise to steady-state analysis of saturation nonlinearity incremental LMS (SNILMS) algorithm in presence of impulsive noise. Both the theoretical and simulation results exhibit better its robustness of this algorithm over the conventional ILMS algorithm. The performance equations are derived by assuming that the input data is Gaussian. Finally it is shown that the theoretical performance curves have excellent agreement with the corresponding simulation results.

Further, two robust DLMS algorithms are proposed by incorporating the robust function in diffusion adaptation. The rank based robust cost function known as Wilcoxon norm is used for this purpose. The simulation study reveals the robust performance of the proposed algorithms against impulsive noise.

4.2 Problem Formulation

Suppose that the WSNs have been deployed over a region to find the average temperature. Each sensor node collects a set of M temperature measurements, $\{x_{k,i}\}_{i=1}^{M}, k = 1, 2, \ldots, N$ over some period. At the end of the day the mean temperature

$$\hat{\theta} = \frac{1}{MN} \sum_{k=1}^{N} \sum_{i=1}^{M} x_{k,i} \tag{4.1}$$

is to be calculated. Let us assume that the measurements are iid and the variance of each measurement is σ^2. However, some fraction say 10% of sensors are damaged or mis-calibrated, so that recorded data would have variance of $100\sigma^2$. Then the estimator variance increases by a factor of 10. Ideally, these bad measurements should be identified and discarded from the estimation process. Robust estimation techniques aim to achieve this strategy by modifying the cost function.

Let θ is the average temperature to be estimated, and $f(\theta)$ is the cost function

which can be expressed as a sum of N *local* cost functions $\{f_k(\theta)\}_N^{k=1}$ in which $f_k(\theta)$ only depends on the data measured at sensor k and is given by,

$$f_k(\theta) = \frac{1}{M}\sum_{i=1}^{M}(x_{k,i} - \theta)^2 \tag{4.2}$$

Hence the global cost function can be written in terms of sum of local cost functions as

$$f(\theta) = \frac{1}{MN}\sum_{k=1}^{N}\sum_{i=1}^{M}(x_{k,i} - \theta)^2 \tag{4.3}$$

Averages can be viewed as the values minimizing quadratic cost functions. Quadratic optimization problems have solutions which are linear functions of the data. A simple accumulation of parameter estimate leads to a solution. General optimization problems can often be solved using this simple, distributed algorithms described in Section 1.2.

4.2.1 Decentralized Incremental Estimation

The optimization problems can be solved iteratively by using gradient and subgradient methods. The update equation for a centralized subgradient descent approach to solve (4.3) is

$$\hat{\theta}_{n+1} = \hat{\theta}_n - \alpha\sum_{k=1}^{N} g_{k,n} \tag{4.4}$$

where $g_{k,n} \in \partial f_k(\hat{\theta}^{(n)}), \alpha$ is a positive step size, and n is the iteration number. In this approach each update step uses data from all the nodes.

In a decentralized incremental approach, each iteration (4.4) is divided into N subiterations. In jth subiteration, kth node updates its local parameter

estimate $f_k(\theta)$. The algorithm can be written as:

$$\begin{aligned}
\psi_0^{(n)} &= \hat{\theta}^{(n-1)} \\
\psi_k^{(n)} &= \psi_{k-1}^{(n)} - \alpha g_{k,n}, \quad k = 1, 2, \ldots, N \\
\hat{\theta}^{(n)} &= \psi_N^{(n)}
\end{aligned} \tag{4.5}$$

where $\hat{\theta}^{(n)}$ is the estimated parameter obtained after n iterations and $\psi_N^{(n)}$ is the parameter estimate of Nth node in nth iteration. For analyzing the rate of convergence an arbitrary starting point is assumed.

In the proposed approach, an estimate of the parameter θ at a node is passed to a neighbouring node. Each node updates the parameters to reduce its local cost (4.2) and then passes its updated parameter to the next node. The flow of information from first node to the last node forms a single cycle. Several cycles through the network are required to obtain a desired solution. Thus these distributed algorithms can be viewed as incremental subgradient optimization procedure, and the number of cycles required to obtain a good solution can be characterized theoretically. If M and N are large, then a high quality estimate can be obtained using a distributed optimization algorithm with less energy and communications compared to the centralized approach. The simulation result for the linear estimate problem with and

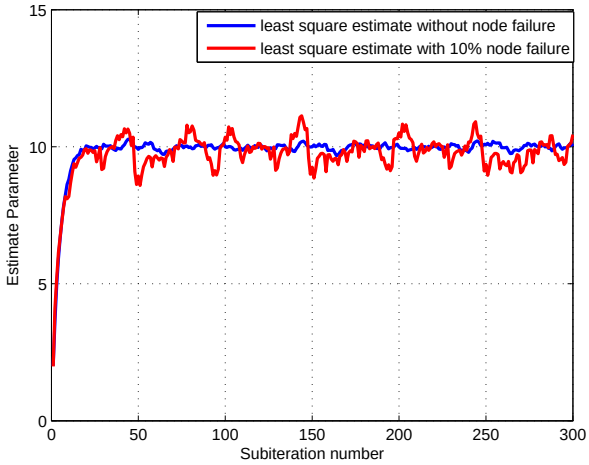


Figure 4.1: Least square estimate without and with 10% node failure

without node failure is plotted in Figure 4.2.1. In the Figure 4.2.1, it is clearly shown

that the least square method provides more variance of the measurement error when the 10% nodes are failed. The same situation may occur when the data is corrupted by impulsive noise.

Therefore the main challenges in accurately estimating parameters in WSNs are link or node failure and presence of impulsive noise. A noise level that fluctuates over a range greater than 10 dB during observation is classified as impulsive noise. In this chapter, distributed estimation algorithms are proposed for WSNs environment which are robust to link/node failure as well as impulsive noise while maintaining faster convergence and low residual MSE.

4.3 Robust Distributed Estimation

In a general estimation problem, the classical least-square loss function, $\|d_k(i) - \mathbf{u}_{k,i}\mathbf{w}\|^2$, defined in Section 1.2 is used. For robust estimation, this cost function is replaced with a robust function, $h(d_k(i), \mathbf{w})$. The robust function $h(d_k(i), \mathbf{w})$ is chosen to assign lesser weights to data points which deviate greatly from the desired parameter. So the cost function (1.7) is suitably modified for a robust estimation and is given as

$$f_{robust}(\mathbf{w}) = \sum_{i=1}^{N} h(d_k(i), \mathbf{w}) \tag{4.6}$$

4.3.1 Robust Cost Functions

Several robust functions are available in literature. The standard robust cost function is the Huber loss function [41] used in several literature and is defined as

$$h(d_k(i); \mathbf{w}) = \begin{cases} \|d_k(i) - \mathbf{u}_{k,i}\mathbf{w}\|^2/2, & \text{for } \|d_k(i) - \mathbf{u}_{k,i}\mathbf{w}\| \leq \gamma \\ \gamma\|d_k(i) - \mathbf{u}_{k,i}\mathbf{w}\| - \gamma^2/2, & \text{for } \|d_k(i) - \mathbf{u}_{k,i}\mathbf{w}\| > \gamma \end{cases} \tag{4.7}$$

This function acts as usual squared error loss function if the distance between the data point $d_k(i)$ and γ is within a threshold value that means if $\mathbf{w}$ is close to $\mathbf{w}^\circ$. However, it gives less weights to points outside a radius γ.

Another function known as error saturation non-linearity [42, 43] is used in litera-

ture for robust estimation in Gaussian-contaminated impulsive noise. The saturation nonlinearity cost function for error $e = \|d_k(i) - \mathbf{u}_{k,i}\mathbf{w}\|$ is defined as

$$h(e) = \int_0^e \exp[-u^2/2\sigma_s^2]du = \sqrt{\frac{\pi}{2}}erf\left[\frac{e}{\sqrt{2}\sigma_s}\right] \tag{4.8}$$

where σ_s^2 is the saturation variance which defines the degree of saturation.

4.3.2 Model for Impulsive Noise

The performance analysis of adaptive filters that are available in literature is for the case of white Gaussian noise. But in any practical situation the impulsive noise is encountered which is modeled as two components of the Gaussian mixture [54,55,73] and may be written as

$$v_k(i) = v_k^g(i) + v_k^{im}(i) = v_k^g(i) + b(i)v_k^w(i) \tag{4.9}$$

where $v_k^g(i)$ and $v_k^w(i)$ are independent zero mean Gaussian noise with variances σ_g^2 and σ_w^2, respectively; $b(i)$ is a switch sequence of ones and zeros and is modeled as an independent and identically distributed (iid) Bernoulli random process with occurrence probability $P_r(b(i) = 1) = p_r$ and $P_r(b(i) = 0) = 1 - p_r$. The variance of $v_k^w(i)$ is chosen to be very large than that of $v_k^g(i)$ so that when $b(i) = 1$, a large impulse is experienced in $v_k(i)$. The corresponding pdf of $v_k(i)$ in (4.9) is given by

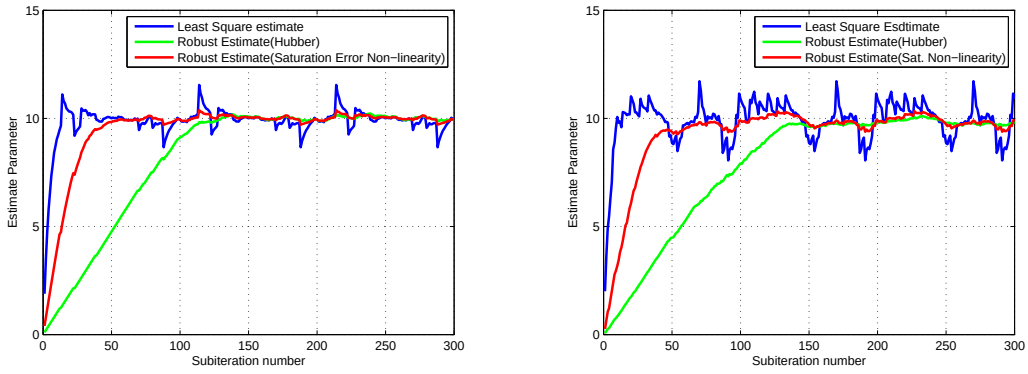
$$f_{v_k}(x) = \frac{1-p_r}{\sqrt{2\pi}\sigma_g}\exp\left(\frac{-x^2}{2\sigma_g^2}\right) + \frac{p_r}{\sqrt{2\pi}\sigma_T}\exp\left(\frac{-x^2}{2\sigma_T^2}\right) \tag{4.10}$$

where $\sigma_T^2 = \sigma_g^2 + \sigma_w^2$ and $E[v^2(i)] = \sigma_v^2 = \sigma_g^2 + p_r\sigma_w^2$. It is noted that when $p_r = 0$ or 1, $v_k(i)$ is a zero-mean Gaussian variate.

4.3.3 Robust Incremental Estimation During Node Failure

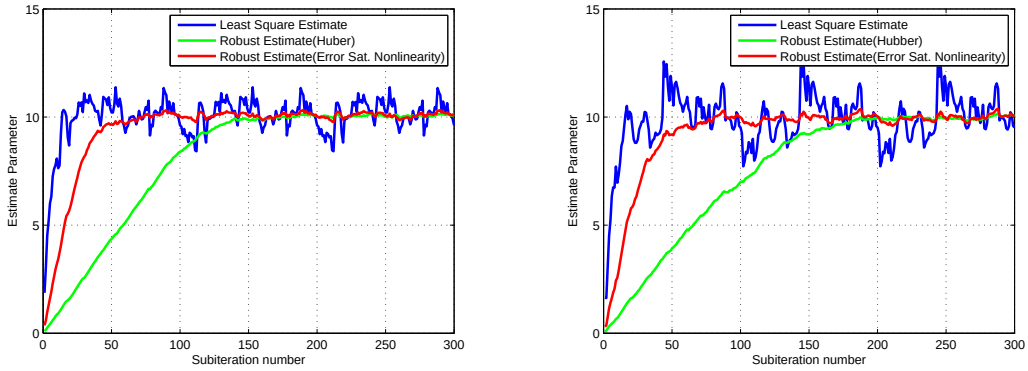
To show the robustness of different functions, the simulation work is carried out using a network where sensors are uniformly distributed over a homogeneous region, measurements are iid and corrupted by additive white Gaussian noise. In this example $N = 100$ sensor nodes are considered with each node collecting $M = 10$ measurements. However some sensors are assumed to be damaged and hence give noisy

measurements. Let $N(\mu, \sigma^2)$ denote the Gaussian distribution with mean μ and variance σ^2. A sensor which is working collects data with distribution $x_{k,i} \sim N(10, 1)$, and a damaged sensor collects data with distribution $x_{k,i} \sim N(10, 100)$. The error saturation non-linearity function with $\sigma_s^2 = 10$ is used and compared its performance with the Huber loss function.



(a) 10% of the sensors are damaged (b) 50% of the sensors are damaged

Figure 4.2: Robust incremental estimation procedures using Hubber function and error saturation nonlinearity with nodal failure (a) 10%.(b) 50% of the sensors are damaged



(a) 10% of the sensors are damaged (b) 50% of the sensors are damaged

Figure 4.3: Robust incremental estimation procedures using Hubber function and error saturation nonlinearity when some nodes are damaged and in presence of impulsive noise (a) 10%.(b) 50% of the sensors are damaged

The convergence characteristics of least-square estimate and the incremental robust estimate for both the robust functions are shown in Figure 4.2(a), when 10% of the sensors are being damaged where as Figure 4.2(b) depicts the convergence be-

havior when 50% nodes are damaged. It is observed that the least square estimate converges faster than the robust estimate, but the variance of the distributed estimator is large. The results show that the robust estiamate obtained using saturation non-linearity, converges faster compared to that of Huber loss function.

4.3.4 Robust Incremental Estimation During Node Failure and Impulsive Noise Condition

The performance of the robust estimation algorithms in presence of 20% impulsive noise (20% of data are corrupted by impulsive noise and rest 80% are corrupted by AWGN) is depicted in Figures 4.3(a) and 4.3(b). The parameters of the algorithm are taken as follows: the step size $\alpha = 0.1$ and $\sigma_s^2 = 10, \sigma_g^2 = 10^{-3}, \sigma_w^2 = 10^4\sigma_g^2$. The results show that the incremental robust estimation algorithms are robust to impulsive noise. The simulation results also reveal that the incremental robust estimate using error saturation non-linearity, converges faster compared to the one based on Huber loss function.

4.4 Error Saturation Nonlinearity Incremental LMS Algorithm

In Section 4.3.3 it has been observed that the performance of robust estimation using error saturation nonlinearity is better compared to that of Huber Cost function. Further, it is also discussed in Introduction section that theory is available for the analysis of error saturation nonlinearity LMS in impulsive noise. Therefore robust ILMS using error saturation nonlinearity known as saturation nonlinearity incremental LMS (SNILMS) is presented here.

Consider a sensor network with N number of nodes. Each node k has access to environment data $\{d_k(i), \mathbf{u}_{k,i}\}$ where $k = 1, 2, 3, ...N$, $d_k(i)$ represents a scalar measurement and $\mathbf{u}_{k,i}$ as a $1 \times M$ regression row vector at time i and is given by:

$$\mathbf{u}_{k,i} = [u_k(i), u_k(i-1), \ldots u_k(i-M+1)]$$

The symbols used for the mathematical analysis are

- $\mathbf{w}^{\circ}$ is $M \times 1$ optimum weight vector
- $\boldsymbol{\psi}_k^{(i)}$ is the estimate of optimum weight by node k at time instant i
- $\mathbf{w}_{i-1}$ is the current global estimate of $\mathbf{w}^{\circ}$ at time instant i
- $\boldsymbol{\psi}_0^{(i)}$ is the initial condition at time instant i and is equal to $\mathbf{w}_{i-1}$
- $\boldsymbol{\psi}_N^{(i)}$ is the local estimate vector at node N which is assigned as $\mathbf{w}_i$
- $v_k(i)$ is represents the impulsive noise for node k at time instant i which is temporally and spatially independent of $\{d_k(i), \mathbf{u}_{k,i}\}$

In incremental mode of cooperation each node k has access only to its immediate neighboring nodes estimate $\boldsymbol{\psi}_{k-1}^{(i)}$.

4.4.1 Adaptive Algorithm with Error Saturation Nonlinearity

At every discrete time instant i, the local scalar measurement $d_k(i)$ is related to the input data vector $\mathbf{u}_k(i)$ as

$$d_k(i) = \mathbf{u}_{k,i}\mathbf{w}^{\circ} + v_k(i) \tag{4.11}$$

It is assumed that the input data to the nodes are spatially and temporally independent. The weight update equation of well known LMS algorithm is given by [27, 83, 84]

$$\boldsymbol{\psi}_k^{(i)} = \boldsymbol{\psi}_k^{(i-1)} + \mu_k e_k(i)\mathbf{u}_{k,i}^T \tag{4.12}$$

The update equation using error saturation nonlinearity LMS [42] is given as

$$\boldsymbol{\psi}_k^{(i)} = \boldsymbol{\psi}_k^{(i-1)} + \mu_k h(e_k(i))\mathbf{u}_{k,i}^T \quad i \geq 0 \tag{4.13}$$

where μ_k is the step size of the kth node.

The error term for kth node at time instant i is defined as

$$e_k(i) = d_k(i) - \mathbf{u}_{k,i}\boldsymbol{\psi}_k^{(i-1)} \tag{4.14}$$

and the error saturation nonlinearity function is defined in (4.8). Now in the following sections robust ILMS using the above discussed error saturation nonlinearity LMS algorithm is developed.

4.4.2 Distributed Error Saturation Nonlinearity ILMS Algorithm

The impulsive type of noise not occurs frequently, but when it occur the LMS based algorithms fail to perform satisfactorily. So the distributed ILMS algorithm [18] is modified by adding non-linearity in the error term which is defined in (4.14). The algorithm is concisely presented as follows.
For each time $i \geq 0$, repeat:

$$\begin{aligned} k &= 1, \ldots, N \\ \boldsymbol{\psi}_0^{(i)} &= \mathbf{w}_{i-1} \\ e_k(i) &= d_k(i) - \mathbf{u}_{k,i}\boldsymbol{\psi}_{k-1}^{(i)} \\ \boldsymbol{\psi}_k^{(i)} &= \boldsymbol{\psi}_{k-1}^{(i)} + \mu_k \mathbf{u}_{k,i}^T h(e_k(i)) \\ \mathbf{w}_i &= \boldsymbol{\psi}_N^{(i)} \end{aligned} \tag{4.15}$$

In this case the steady-state performance can be controlled by choosing suitable value of local step size μ_k and saturation variance in the presence of Gaussian noise. In complexity point of view this algorithm requires little more operation than ILMS as the former involves error nonlinearity.

4.5 Performance Analysis of Robust SNILMS

The performance of an adaptive algorithm is evaluated in terms of its transient behavior which provides the information about how fast a system learns and its steady-state behavior gives information about how well a system learns. Such performance analysis is usually challenging when the noise is non-stationary and the algorithm contains nonlinearity. The performance analysis of ILMS is studied in [18] by exploiting the spatial energy conservation argument. The chapter provides the steady-state performance analysis of SNLMS algorithm which guaranteed the con-

vergence in presence of impulsive noise.

The weighted error which is the difference between estimated weight and the optimum weight at kth node is given as

$$\tilde{\boldsymbol{\psi}}_k^{(i)} = \mathbf{w}^\circ - \boldsymbol{\psi}_k^{(i)} \tag{4.16}$$

The local error signals at each kth node are defined as

$$e_{a,k}(i) = \mathbf{u}_{k,i}\tilde{\boldsymbol{\psi}}_{k-1}^{(i)} \quad (a\ priori \text{ error})$$
$$e_{p,k}(i) = \mathbf{u}_{k,i}\tilde{\boldsymbol{\psi}}_{k}^{(i)} \quad (a\ posteriori \text{ error})$$

The output error $e_k(i)$ defined in (4.14) can be rewritten in terms of *a priori* error as

$$e_k(i) = v_k(i) + \mathbf{u}_{k,i}\tilde{\boldsymbol{\psi}}_{k-1}^{(i)} \tag{4.17}$$

By using the definition of *a priori* and the data model (4.11), the output error is obtained as

$$e_k(i) = e_{a,k}(i) + v_k(i) \tag{4.18}$$

Assuming that the noise is independent of weight and input data, and then the variance relation can be written as

$$E\|e_k(i)\|^2 = E\|e_{a,k}\|^2 + E\|v_k(i)\|^2 \tag{4.19}$$

The objective in this section is to evaluate the steady-state values of the variances such as MSE, EMSE, MSD for every node. These quantities are defined as [27]:

$$\eta_k = E\|\tilde{\boldsymbol{\psi}}_{k-1}(\infty)\|^2 \quad (\mathbf{MSD}) \tag{4.20a}$$
$$\zeta_k = E|e_{a,k}(\infty)|^2 \quad (\mathbf{EMSE}) \tag{4.20b}$$
$$\xi_k = E|e_k(\infty)|^2 = \zeta_k + E\|v_k(i)\|^2 \quad (\mathbf{MSE}) \tag{4.20c}$$

In order to study these parameters, the weighted *a priori* and *a posteriori* error

terms are introduced as

$$e_{a,k}^{\Sigma}(i) = \mathbf{u}_{k,i}\Sigma\tilde{\boldsymbol{\psi}}_{k-1}^{(i)} \qquad \text{and} \qquad e_{p,k}^{\Sigma}(i) = \mathbf{u}_{k,i}\Sigma\tilde{\boldsymbol{\psi}}_{k}^{(i)} \tag{4.21}$$

where Σ is a symmetric positive definite weighting matrix. It may be noted that different choice of Σ allows evaluation of different performance.

Subtracting $\mathbf{w}^{\circ}$ from both sides of weight update equation in (4.15) results

$$\tilde{\boldsymbol{\psi}}_{k}^{(i)} = \tilde{\boldsymbol{\psi}}_{k-1}^{(i)} - \mu_k \mathbf{u}_{k,i}^{\mathrm{T}} f(e_k(i)) \tag{4.22}$$

Relation between various error terms $e_{a,k}^{\Sigma}(i)$, $e_{p,k}^{\Sigma}(i)$ and $e_k(i)$ is obtained by premultiplying both sides of (4.22) by $\mathbf{u}_{k,i}\Sigma$

$$\begin{aligned} e_{p,k}^{\Sigma}(i) &= e_{a,k}^{\Sigma}(i) - \mu_k \|\mathbf{u}_{k,i}\|_{\Sigma}^2 f[e_k(i)] \\ f[e_k(i)] &= \frac{1}{\mu_k} \frac{e_{a,k}^{\Sigma}(i) - e_{p,k}^{\Sigma}(i)}{\|\mathbf{u}_{k,i}\|_{\Sigma}^2} \end{aligned} \tag{4.23}$$

4.5.1 Weight-Energy Relation

Combining (4.23) and (4.22) and after removing the nonlinearity term $f[e(i)]$, we get

$$\tilde{\boldsymbol{\psi}}_{k}^{(i)} + \frac{\mathbf{u}_{k,i} e_{a,k}^{\Sigma}(i)}{\|\mathbf{u}_{k,i}\|_{\Sigma}^2} = \tilde{\boldsymbol{\psi}}_{k-1}^{(i)} + \frac{\mathbf{u}_{k,i} e_{p,k}^{\Sigma}(i)}{\|\mathbf{u}_{k,i}\|_{\Sigma}^2} \tag{4.24}$$

Taking weighted energy on both sides of (4.24) and canceling identical will lead to

$$\|\tilde{\boldsymbol{\psi}}_{k}^{(i)}\|_{\Sigma}^2 + \frac{|e_{a,k}^{\Sigma}(i)|^2}{\|\mathbf{u}_{k,i}\|_{\Sigma}^2} = \|\tilde{\boldsymbol{\psi}}_{k-1}^{(i)}\|_{\Sigma}^2 + \frac{|e_{p,k}^{\Sigma}(i)|^2}{\|\mathbf{u}_{k,i}\|_{\Sigma}^2} \tag{4.25}$$

Equation (4.25) represents the well known weighted energy relation which is same as in ILMS [18, 32]. However (4.25) involves weighted *a priori* and *a posteriori* errors which associate error nonlinearity term.

4.5.2 Variance Relation

Replacing *a posteriori* error in (4.25) by its equivalent expression in (4.23) and then taking expectation on both the sides leads to

$$E[\|\tilde{\boldsymbol{\psi}}_k^{(i)}\|_\Sigma^2] = E[\|\tilde{\boldsymbol{\psi}}_{k-1}^{(i)}\|_\Sigma^2] - 2\mu_k E[e_{a,k}^\Sigma(i) f[e_k(i)]] + \mu_k^2 E[\|\mathbf{u}_{k,i}\|_\Sigma^2 f^2[e_k(i)]] \tag{4.26}$$

Evaluation (4.26) is complex as it contains the nonlinearity term (2nd and 3rd terms on RHS). To evaluate the transient analysis the following assumptions which are common in literature are made. To evaluate the steady state analysis of the proposed SNILMS the following assumptions are made.

- The noise sequence $v_k(i)$ is independence of $\mathbf{u}_{k,i}$
- For any constant matrix Σ and for all i, $e_{a,k}(i)$ and $e_{a,k}^\Sigma(i)$ are jointly Gaussian.
- The adaptive filter is long enough such that the weighted norm of input regressor and the square of error nonlinearity i.e. $f^2[e_k(i)]$ are uncorrelated.

In order to proceed for further analysis of (4.26) the nonlinear terms on the right hand side need to be evaluated. Price's theorem [120, 121] plays an important role to analyze such non-linear terms is given as

$$E[af[b+c]] = \frac{E[ab]}{E[b^2]} E[bf[b+c]] \tag{4.27}$$

where a and b are jointly Gaussian random variables that are independent from the third random variable c. The noise is considered to be impulsive and is independent of the errors. By using the Price's theorem (4.27) and assuming that the impulsive noise is independent of errors $e_{a,k}(i)$ and $e_{a,k}^\Sigma(i)$ then the 2nd term in (4.26) is given as

$$E[e_{a,k}^\Sigma(i) f[e_k(i)]] = E[e_{a,k}^\Sigma(i) e_{a,k}(i)] A_k[E[e_{a,k}^2(i)]] \tag{4.28}$$

where A_k contains the non-linear term and is written as

$$A_k[E[e_{a,k}^2(i)]] = \frac{E[e_{a,k}(i) f[e_{a,k}(i) + v_k(i)]]}{E[e_{a,k}^2(i)]} \tag{4.29}$$

The general expression of A_k for any type of noise is given as

$$A_k = \frac{\sigma_s}{\sqrt{E[e_{a,k}^2(i)] + \sigma_s^2}} E\left[\exp\left[-\frac{v_k^2(i)}{2(E[e_{a,k}^2(i)] + \sigma_s^2)}\right]\right]$$

The polarization property of weighted norm states that

$$(\mathbf{u}\Sigma_1\mathbf{w})(\mathbf{u}\Sigma_2\mathbf{w}) = \|\mathbf{w}\|^2_{\Sigma_1\mathbf{u}^T\mathbf{u}\Sigma_2}$$

By using this property the above equation can be written as

$$E[e_{a,k}^{\Sigma}(i)e_{a,k}(i)] = E\|\tilde{\boldsymbol{\psi}}_{k-1}^{(i)}\|^2_{\Sigma\mathbf{u}_{k,i}^T\mathbf{u}_{k,i}} \tag{4.30}$$

Therefore (4.28) is written as

$$E[e_{a,k}^{\Sigma}(i)f[e_k(i)]] = E\|\tilde{\boldsymbol{\psi}}_{k-1}^{(i)}\|^2_{\Sigma\mathbf{u}_{k,i}^T\mathbf{u}_{k,i}} A_k[E[e_{a,k}^2(i)]] \tag{4.31}$$

In similar way the third term of (4.26) is evaluated by taking long filter assumption for which the weighted norm of input data and the squared error nonlinearity are uncorrelated [85]. Therefore third term is obtained as

$$E[\|\mathbf{u}_{k,i}\|^2_{\Sigma} f^2[e_k(i)]] = E[\|\mathbf{u}_{k,i}\|^2_{\Sigma}] B_k[E[e_{a,k}^2(i)]] \tag{4.32}$$

where the nonlinear component B_k is

$$B_k[E[e_{a,k}^2(i)]] = E[f^2[e_k(i)]] \tag{4.33}$$

Assuming that the *a priori* error is Gaussian the general expression of B_k for any type of noise which is independent of input data is given below [85, 122].

$$B_k = 2\pi\sigma_s^2\left(\frac{1}{4} - \frac{1}{\pi}\int_{\pi/4}^{\pi/2}\sqrt{\frac{\sigma_s^2\sin^2(\theta)}{E[e_{a,k}^2] + \sigma_s^2\sin^2(\theta)}}\cdot E\left[\exp\left[-\frac{v_k^2(i)}{2(E[e_{a,k}^2] + \sigma_s^2\sin^2(\theta))}\right]\right]d\theta\right) \tag{4.34}$$

After evaluating the second and third terms from the variance relation is obtained as

$$E[\|\tilde{\boldsymbol{\psi}}_k^{(i)}\|_\Sigma^2] = E[\|\tilde{\boldsymbol{\psi}}_{k-1}^{(i)}\|_\Sigma^2] - 2\mu_k E\|\tilde{\boldsymbol{\psi}}_{k-1}^{(i)}\|^2_{\Sigma\mathbf{u}_{k,i}^T\mathbf{u}_{k,i}} A_k[E[e_{a,k}^2(i)]] + \mu_k^2 E[\|\mathbf{u}_{k,i}\|_\Sigma^2] B_k[E[e_{a,k}^2(i)]] \tag{4.35}$$

4.5.3 Evaluation of non-linear term A and B

In the present case of Gaussian contaminated impulsive noise with pdf defined in (4.10), the equations of A_k and B_k are derived to be

$$A_k = \frac{(1-p_r)\sigma_s}{\sqrt{E[e_{a,k}^2] + \sigma_{g,k}^2 + \sigma_s^2}} + \frac{(1-p_r)\sigma_s}{\sqrt{E[e_{a,k}^2] + \sigma_T^2 + \sigma_s^2}} \tag{4.36a}$$

$$B_k = (1-p_r)\sigma_s^2 \sin^{-1}\left(\frac{\sigma_{g,k}^2 + E[e_{a,k}^2]}{E[e_{a,k}^2] + \sigma_s^2 + \sigma_{g,k}^2}\right) + p_r\sigma_s^2 \sin^{-1}\left(\frac{\sigma_T^2 + E[e_{a,k}^2]}{E[e_{a,k}^2] + \sigma_s^2 + \sigma_T^2}\right) \tag{4.36b}$$

Using the independence of the regression data $\{\mathbf{u}_k\}$ allows us to write

$$E[e_{a,k}^2(i)] = E\|\tilde{\boldsymbol{\psi}}_{k-1}^{(i)}\|^2_{\mathbf{u}_{k,i}^T\mathbf{u}_{k,i}} = E\|\tilde{\boldsymbol{\psi}}_{k-1}^{(i)}\|^2_{R_{u,k}} \tag{4.37}$$

which is the excess-mean-square error at time instant i of node k in the network. hence A_k and B_k are expressed as

$$A_k^{(i)} = \frac{(1-p_r)\sigma_s}{\sqrt{E\|\tilde{\boldsymbol{\psi}}_{k-1}^{(i)}\|^2_{R_{u,k}} + \sigma_{g,k}^2 + \sigma_s^2}} + \frac{p_r\sigma_s}{\sqrt{E\|\tilde{\boldsymbol{\psi}}_{k-1}^{(i)}\|^2_{R_{u,k}} + \sigma_T^2 + \sigma_s^2}} \tag{4.38a}$$

$$B_k^{(i)} = (1-p_r)\sigma_s^2 \sin^{-1}\left(\frac{\sigma_{g,k}^2 + E\|\tilde{\psi}_{k-1}^{(i)}\|^2_{R_{u,k}}}{E\|\tilde{\boldsymbol{\psi}}_{k-1}^{(i)}\|^2_{R_{u,k}} + \sigma_s^2 + \sigma_{g,k}^2}\right) + p_r\sigma_s^2 \sin^{-1}\left(\frac{\sigma_T^2 + E\|\tilde{\boldsymbol{\psi}}_{k-1}^{(i)}\|^2_{R_{u,k}}}{E\|\tilde{\boldsymbol{\psi}}_{k-1}^{(i)}\|^2_{R_{u,k}} + \sigma_s^2 + \sigma_T^2}\right) \tag{4.38b}$$

It is assumed that the regressors $\{\mathbf{u}_{k,i}\}$ arises from Gaussian distribution with covariance matrix $\mathbf{R}_{u,k} = E[u_k^T u_k]$. Now introduce the eigen decomposition $\mathbf{R}_{u,k} = U_k \wedge_k U_k^T$, where U_k is unitary (*i.e.* $U_k^T U_K = U_k U_k^T = I$) and $\wedge_k$ is a diagonal matrix

with eigenvalues of $\mathbf{R}_{u,k}$. Then, the transformed quantities are defined as

$$\overline{\boldsymbol{\psi}}_k = U_k^T \tilde{\boldsymbol{\psi}}_k \qquad \overline{\mathbf{u}}_k = \mathbf{u}_k U_k \qquad \overline{\Sigma} = U_k^T \Sigma U_k$$

Since U_K is unitary matrix, it is easy to find that

$$\|\overline{\boldsymbol{\psi}}_k\|^2_{\overline{\Sigma}} = \|\boldsymbol{\psi}_k\|^2_{\Sigma} \qquad \|\overline{\mathbf{u}}_k\|^2_{\overline{\Sigma}} = \|\mathbf{u}_k\|^2_{\Sigma}$$

Using the change of variables, the variance relation (4.35) is rewritten as

$$E[\|\overline{\boldsymbol{\psi}}_k^{(i)}\|^2_{\overline{\Sigma}}] = E[\|\overline{\boldsymbol{\psi}}_{k-1}^{(i)}\|^2_{\overline{\Sigma}}] - 2\mu_k E\|\overline{\boldsymbol{\psi}}_{k-1}^{(i)}\|^2_{\overline{\Sigma}\overline{\mathbf{u}}_{k,i}^T \overline{\mathbf{u}}_{k,i}} A_k^{(i)} + \mu_k^2 E[\|\overline{\mathbf{u}}_{k,i}\|^2_{\overline{\Sigma}}] B_k^{(i)} \tag{4.39}$$

The derivation of moments needed for the steady state analysis are given by

$$E\|\overline{\mathbf{u}}_k\|^2_{\overline{\Sigma}} = \mathrm{Tr}(\wedge_k \overline{\Sigma}) \quad \text{and} \quad E[\overline{\mathbf{u}}_k^T \overline{\mathbf{u}}_k] = \wedge_k \tag{4.40}$$

Using (4.40) and invoking the independence of the regression data $\{\mathbf{u}_k\}$, (4.39) is solved to be

$$E\|\overline{\boldsymbol{\psi}}_k^{(i)}\|^2_{\overline{\Sigma}} = E\|\overline{\boldsymbol{\psi}}_{k-1}^{(i)}\|^2_{\overline{\Sigma}} - 2\mu_k E\|\overline{\boldsymbol{\psi}}_{k-1}^{(i)}\|^2_{\overline{\Sigma}\wedge_k} A_k^{(i)} + \mu_k^2 \mathrm{Tr}(\wedge_k \overline{\Sigma}) B_k^{(i)} \tag{4.41}$$

4.5.4 Steady-State Behavior

When $i \to \infty$, let us take

$$\overline{\mathbf{h}}_k = \overline{\boldsymbol{\psi}}_k^{\infty}$$

Then for $i \to \infty$(i.e. in steady state), the variance relation (4.41) gives

$$E\|\overline{\mathbf{h}}_k\|^2_{\overline{\Sigma}} = E\|\overline{\mathbf{h}}_{k-1}\|^2_{\overline{\Sigma}} - 2\mu_k E\|\overline{\mathbf{h}}_{k-1}\|^2_{\overline{\Sigma}\wedge_k} A_k + \mu_k^2 \mathrm{Tr}(\wedge_k \overline{\Sigma}) B_k \tag{4.42}$$

The convergence analysis of the LMS algorithm with general error nonlinearity and an iid input are studied in [86, 123]. The theory suggests that the saturation LMS algorithm converge in its mean and variance. The performance of this algorithm also studied [42], where the analysis proves the robustness to impulsive noise. Therefore in the present case it is assuming that the variance converges to the minimum value. Considering this assumption the nonlinear parameters in the above variance relation

are approximated as.

$$\begin{aligned} A_k^{(\infty)} &= \frac{(1-p_r)\sigma_s}{\sqrt{E\|\overline{\mathbf{h}}_{k-1}\|^2_{\wedge_k} + \sigma^2_{g,k} + \sigma^2_s}} + \frac{p_r\sigma_s}{\sqrt{E\|\overline{\mathbf{h}}_{k-1}\|^2_{\wedge_k} + \sigma^2_T + \sigma^2_s}} \\ &\approx \frac{(1-p_r)\sigma_s}{\sqrt{\sigma^2_{g,k} + \sigma^2_s}} + \frac{p_r\sigma_s}{\sqrt{\sigma^2_T + \sigma^2_s}} \end{aligned} \tag{4.43}$$

In similar way the other nonlinear parameter B_k is given as

$$\begin{aligned} B_k^{(\infty)} &= (1-p_r)\sigma_s^2 \sin^{-1}\left(\frac{\sigma^2_{g,k} + E\|\overline{\mathbf{h}}_{k-1}\|^2_{\wedge_k}}{E\|\overline{\mathbf{h}}_{k-1}\|^2_{\wedge_k} + \sigma^2_s + \sigma^2_{g,k}}\right) \\ &\quad + p_r\sigma_s^2 \sin^{-1}\left(\frac{\sigma^2_T + E\|\overline{\mathbf{h}}_{k-1}\|^2_{\wedge_k}}{E\|\overline{\mathbf{h}}_{k-1}\|^2_{\wedge_k} + \sigma^2_s + \sigma^2_\Sigma}\right) \\ &\approx (1-p_r)\sigma_s^2 \sin^{-1}\left(\frac{\sigma^2_{g,k}}{\sigma^2_s + \sigma^2_{g,k}}\right) + p_r\sigma_s^2 \sin^{-1}\left(\frac{\sigma^2_T}{\sigma^2_s + \sigma^2_T}\right) \end{aligned} \tag{4.44}$$

Now these parameters are independent of weighted matrix Σ, so that they are considered as constant in subsequent analysis. Therefore (4.42) is rewritten as

$$E\|\overline{\mathbf{h}}_k\|^2_{\overline{\Sigma}} = E\|\overline{\mathbf{h}}_{k-1}\|^2_{\overline{\Sigma}} - E\|\overline{\mathbf{h}}_{k-1}\|^2_{2\mu_k A_k \overline{\Sigma} \wedge_k} + \mu_k^2 \mathrm{Tr}(\wedge_k \overline{\Sigma}) B_k \tag{4.45}$$

The superposition property of *weighted-norm* states that

$$a_1\|x\|^2_{W_1} + a_2\|x\|^2_{W_2} = \|x\|^2_{a_1W_1 + a_2W_2}$$

Using this property, (4.45) can is rewritten in compact form as

$$E\|\overline{\mathbf{h}}_k\|^2_{\overline{\Sigma}} = E\|\overline{\mathbf{h}}_{k-1}\|^2_{\overline{\Sigma}'} + \mu_k^2 \mathrm{Tr}(\wedge_k \overline{\Sigma}) B_k \tag{4.46}$$

where

$$\overline{\Sigma}' = \overline{\Sigma} - 2\mu_k A_k \overline{\Sigma} \wedge_k \tag{4.47}$$

The interesting fact is that $\overline{\Sigma}'$ is diagonal if $\overline{\Sigma}$ is diagonal. The choice of weighted matrix Σ is made in (4.47) in such a way that both $\overline{\Sigma}$ and $\overline{\Sigma}'$ are diagonal. Under this condition, it is possible to rewrite (4.46) in a more compact form in terms of diagonal entries of $\{\overline{\Sigma}, \overline{\Sigma}'\}$.

The steady-state performance defined in (4.20) can be measured by using suitable

method. For this purpose the following terms

$$\eta_k = E\|\overline{\mathbf{h}}_{k-1}\|_q^2, \quad q = \underline{1} \quad (\mathbf{MSD}) \tag{4.48a}$$

$$\zeta_k = E|\overline{\mathbf{h}}_{k-1}|_{\lambda_k}^2, \quad \lambda_k = \mathrm{diag}\{\wedge_k\} \quad (\mathbf{EMSE}) \tag{4.48b}$$

$$\xi_k = \zeta_k + E\|v_k(i)\|^2 \quad (\mathbf{MSE}) \tag{4.48c}$$

The standard assumptions of Gaussian noise that is iid and independent of input are also used in the present analysis except that the variance of the noise is given as $\sigma_{v,k}^2 = \sigma_{g,k}^2 + p_r\sigma_w^2$. The approximate steady-state expressions of MSD and EMSE for small step size are given as:

$$\eta_k \approx (\mu_1^2 B_1 \lambda_1^T + \ldots + \mu_N^2 B_N \lambda_N^T) \times (2\mu_1 A_1 \wedge_1 + 2\mu_2 A_2 \wedge_2 + \ldots + 2\mu_N A_N \wedge_N)^{-1} q \tag{4.49}$$

In the similar way, the EMSE can approximate for small step size as

$$\zeta_k \approx (\mu_1^2 B_1 \lambda_1^T + \ldots + \mu_N^2 B_N \lambda_N^T) \times (2\mu_1 A_1 \wedge_1 + 2\mu_2 A_2 \wedge_2 + \ldots + 2\mu_N A_N \wedge_N)^{-1} \lambda_k \tag{4.50}$$

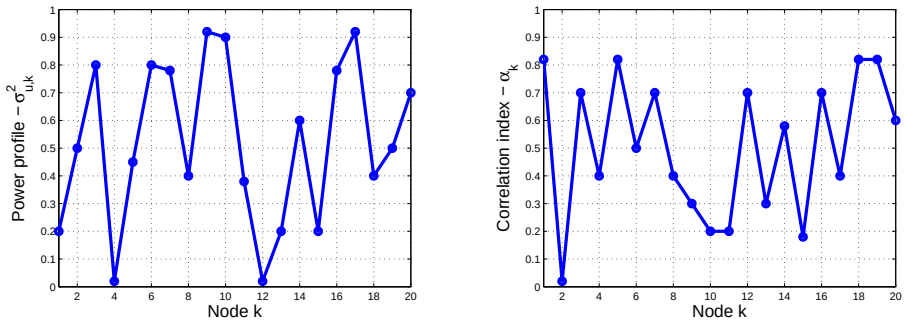
Both MSE and EMSE are going to zero asymptotically when the step size $\mu_l \to 0$. At that time the MSE is achieving the background impulsive noise level at each node in the network.

4.6 Simulation Results for Robust ILMS

The performance of the proposed SNILMS algorithm is evaluated through simulation study and the results are compared with those obtained by theoretical investigation. Further to assess the performance of the proposed method both theoretical and simulation results of ILMS are also plotted. All simulations are carried out using regressor input with shift structure. The desired data are generated according to the model given in (4.11), and the unknown vector $\mathbf{w}^\circ$ is set to $[1, 1, \ldots, 1]^T/\sqrt{M}$.

The network consists of 20 nodes with each regressor size of (1×10) collecting data through a correlated process given in (3.38) (details is discussed in 3.3.3).

The network parameters are taken same as in [18] for comparison purpose and are depicted in Figures 4.4(a) and 4.4(b). The background Gaussian noise with variance $\sigma^2_{g,k}$ are also generated randomly and is shown in Figure 4.5. The rationale behind the assumption of a particular noise power profile is based on the fact that each individual node has its own signal and noise power which are different for all other nodes. Hence the noise powers are chosen in between 0 to 0.01 randomly with Gaussian distribution which are assumed to match with real time environment.



(a) (b)

Figure 4.4: (a) Regressor power profile. (b) Correlation index per node

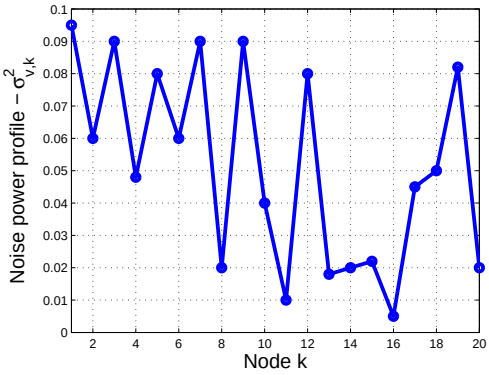


Figure 4.5: Noise power profile

To generate the performance curves, hundred independent experiments are performed and the results are averaged. The steady-state curves are generated by running the networks for 3000 iterations. The quantities such as the MSD, EMSE and MSE are obtained by averaging the last 300 samples of the corresponding learning curves.

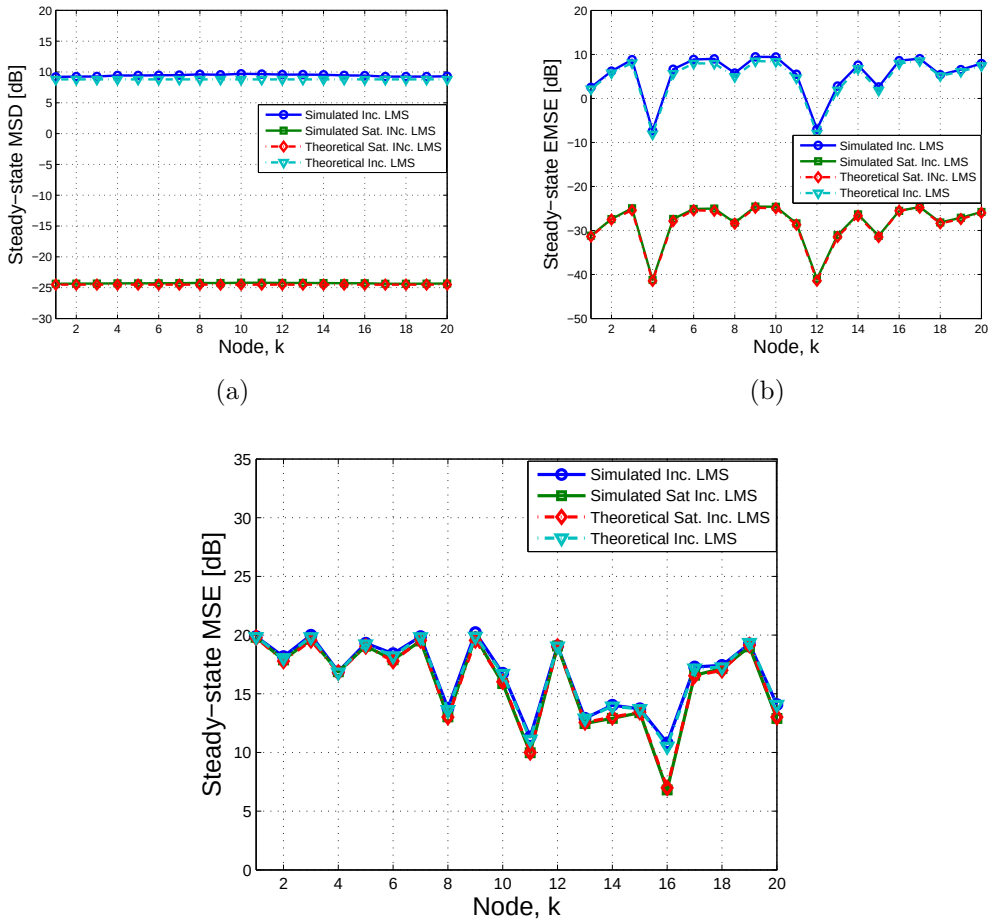


Figure 4.6: Theoretical and simulated MSD, EMSE and MSE curves for $p_r = 0.1\sigma_s^2 = 0.01, \mu = 0.03$ (a) MSD Vs nodes. (b) EMSE Vs nodes (c) MSE Vs nodes

The robustness of the algorithm in 10% impulsive noise is tested. The variance of impulsive noise is defined as 10^4 times the back ground Gaussian noise variance defined for each node in Figure 4.5. Figures 4.6(a), 4.6(b) and 4.6(c) clearly show the robustness of algorithm in impulsive noise over ILMS. The steady-state values attained by both the MSD and EMSE are around -25 dB for MSD and -30 dB for EMSE respectively where as the ILMS attains poor performances such as 10 dB and 5 dB for MSD and EMSE respectively.

The main objectives of all the adaptive algorithms are to estimate weights that should approach to the optimum. If the noise is stationary, then the mean-square error should be close to the background noise. But when the noise is non stationary,

then the MSE will not converge. In this scenario, the estimated weights diverge from the desire ones. It is because the error is used directly in weight update equation of the adaptive algorithm. Therefore the MSD and EMSE are not converging and their steady-state values are very high in case of distributed ILMS.

Where as in case of the saturation error nonlinearity distributed incremental algorithm, the error is not used directly in the weight update equation. Here the error is feedback through the Gaussian nonlinearity function, where the error is mapped within limit of $[-\sqrt{\frac{\pi}{2}}\sigma_s, \sqrt{\frac{\pi}{2}}\sigma_s]$ again that can also limited by the proper choice of σ_s. The error may be high enough due to impulsive, but that mapped to small value within the defined limit. So that the estimated weights are approaching towards the desired weight due to the presence of error nonlinearity in update equation. This reflects in the steady-state performance of filter. The MSD and EMSE are very low indicating $\boldsymbol{\psi}_k^{\infty}$ is a good estimate for $\mathbf{w}^{\circ}$. But the error is remains unchanged, so the steady-state MSE is not converging in both the cases.

After showing the robustness of algorithm in 10% impulsive noise then tested it for 50%. The Figures 4.7(a), 4.7(b) and 4.7(c) show that the error saturation nonlinearity incremental algorithms robustness towards the impulsive noise. It is observed that the simulation results exactly match to the plots obtained from the equations obtained from the analysis. Figures 4.7(a) and 4.7(b) shows the plot of the MSD and EMSE performance of the proposed method as well as the conventional LMS based method. The proposed method exhibits nearly 35 dB improvement in performance compared to the LMS one which strongly supports the improved robustness of the new method. Though the probability of occurrence of impulsive noise is 50%, still the estimated parameters are approaching towards desire.

The above results depicts the steady-state quantities as a function of node k for particular value of step size μ_k. When step size is small then the convergence speed is less, but the steady state performance is better. If high step size is chosen then the algorithm converges faster yielding less steady state performance. Therefore for a better steady state performance, smaller step size has to be chosen. The performance of the proposed and the existing algorithms have been compared by taking identical

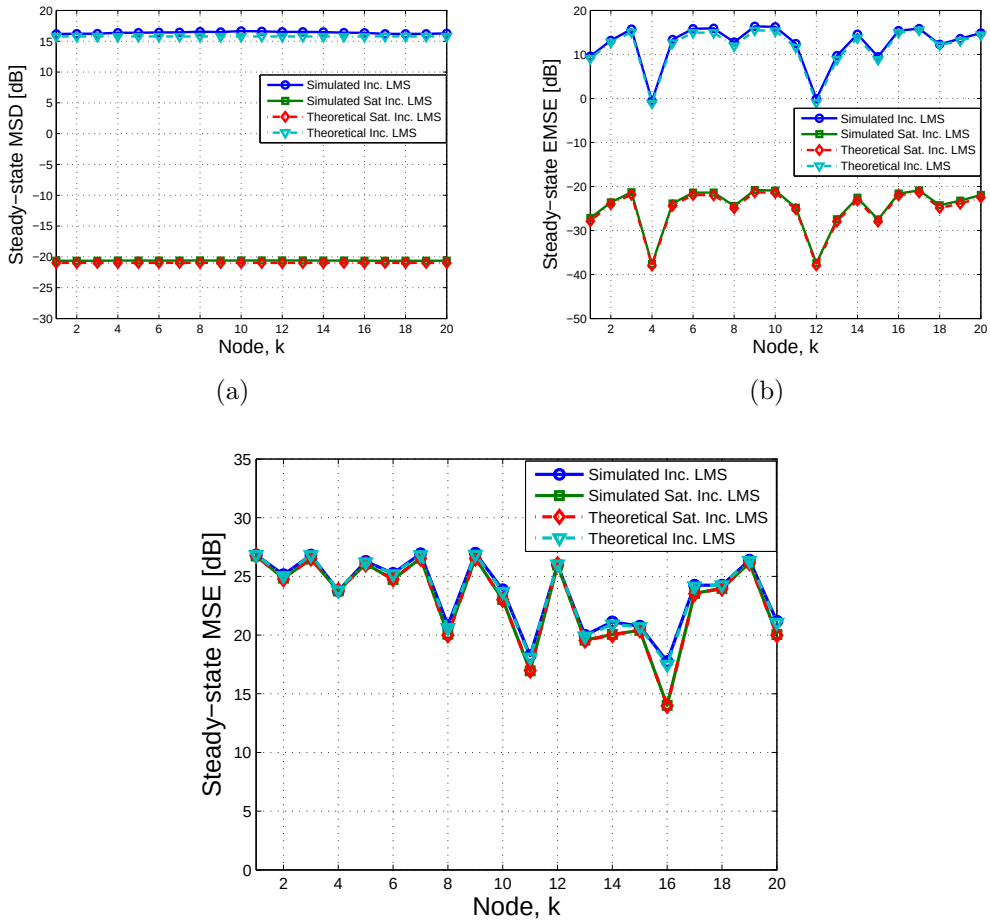


Figure 4.7: Theoretical and simulated MSD, EMSE and MSE curves for $p_r = 0.5\sigma_s^2 = 0.01, \mu = 0.03$, (a) MSD Vs nodes. (b) EMSE Vs nodes (c) MSE Vs nodes

step size in both cases. These curves help the designer how one can adjust step size at a certain node to compensate signal power increased in nearby nodes. One may observe the close match between simulation and theory, as well as improvement in performance over ILMS in presence of impulsive noise.

In this chapter, a second kind of curve that shows the behavior of the steady-state quantities as a function of the step size for a particular node is also provided. These curves evaluate the quality of theoretical model and the assumptions used to derive the theoretical model. A large deviation between theory and simulation are expected for bigger step size. It is because when the step-size becomes large, the simplifying assumptions adopted in the analysis are no longer reasonable. The

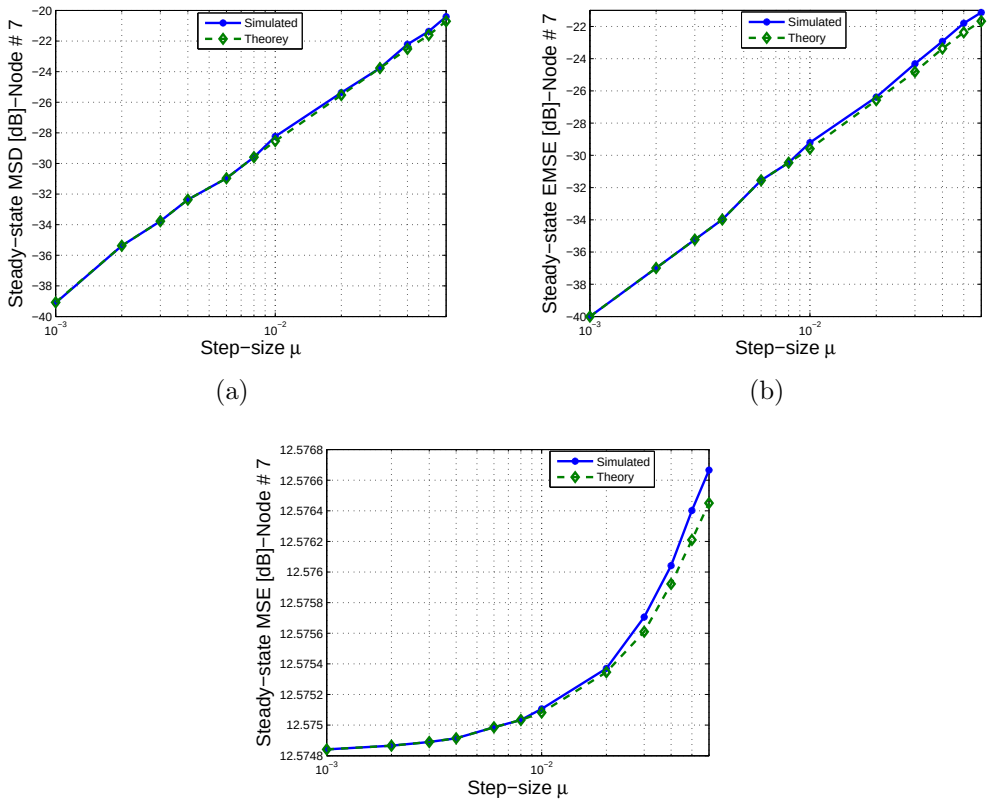


Figure 4.8: Theoretical and simulated performance curves for $p_r = 0.2$, $\sigma_s^2 = 0.01$, (a) MSD Vs step-size at node-7. (b) EMSE Vs step-size at node-7 (c) MSE Vs step-size at node-7

Figures 4.8(a), 4.8(b) and 4.8(c) show the performance with respect to step-size. When the step size is increasing, the theoretical values start diverting.

A robust adaptive algorithm, together with a good step-size and proper cooperative scheme may take advantage of the spatial diversity provided by the adaptive network. It will be better if the performance of individual node in the network is equalized. This may possible by choosing the proper step-size. The nodes presenting poor performance, or high noise level should assign with small step-size to equalize the performance.

4.7 Robust Diffusion LMS Algorithm

It is a fact that the gradient based DLMS [32] and BDLMS algorithms described in Chapter 3 are not robust to impulsive type of noise present. To make the algorithm

robust to impulsive noise, a new class of distributed diffusion algorithm based on robust function defined in Section 4.3 or using rank based Wilcoxon norm has been proposed. The objective of using robust block LMS instead of simple DLMS is that, the Wilcoxon norm need block of error to evaluate the norm. The BDLMS algorithm is incorporated with Wilcoxon norm and the robust algorithm is also energy efficient like BDLMS.

4.7.1 Robust BDLMS Algorithm using Robust Cost functions

This algorithm is similar to block diffusion LMS algorithm discussed in Chapter 3, but here the weight update equation after local diffusion is modified accordingly the cost function defined in Section 4.3. The algorithm is given as:

$$\boldsymbol{\theta}_k^{j-1} = \sum_{l \in \mathcal{N}_k} c_{kl} \hat{\mathbf{w}}_k^{j-1}, \qquad \boldsymbol{\theta}_k(-1) = 0 \tag{4.51a}$$

$$\hat{\mathbf{w}}_k^{j} = \boldsymbol{\theta}_k^{j-1} + \frac{\mu_k}{L} \sum_{q=0}^{L-1} \mathbf{u}_k(jL+q) h\left(d_k(jL+q) - \mathbf{u}_k^T(jL+q)\boldsymbol{\theta}_k^{j-1}\right) \tag{4.51b}$$

where $h()$ is either error saturation nonlinearity function or Huber's function. Since the Hubers function based adaptive algorithms are very slow in convergence, therefore here error saturation nonlinearity function is used for robustness.

4.7.2 Wilcoxon Norm based Robust BDLMS Algorithm

Before describing the robust BDLMS algorithm using Wilcoxon norm, let us introduce Wilcoxon norm first.

Wilcoxon Least Mean Square Algorithm (WLMS)

Consider a linear combiner with its M weights to be updated by an algorithm derived using the Wilcoxon norm as cost function. In developing the WLMS algorithm, epoch based training is employed. Let the weights of the combiner be trained for L samples in each experiments j. To define Wilcoxon norm of an error vector $\mathbf{e}$ of length L [44, 82], it needs a score function φ: $[0,1] \rightarrow \mathcal{R}$ which is non decreasing

such that

$$\int_0^1 \varphi^2(u)du < \infty.$$

The score associated with the score function φ is defined by

$$a(i) = \varphi\left(\frac{i}{L+1}\right), \qquad i \in \underline{l}$$

where l is a fixed positive integer.

It can be shown that the following cost function which is a pseudonorm on $\mathcal{R}^l$:

$$J(j) = \| \mathbf{e} \|_W = \sum_{i=1}^{L} a\left(R(e_i)\right) e_i = \sum_{i=1}^{L} a(i) e_{(i)} \tag{4.52}$$

where $R(e_i)$ denotes the rank of e_i among $e_1, \ldots, e_l, e_{(1)} \leq \ldots \leq e_{(l)}$ are the ordered values of $e_1, \ldots, e_l, a(i) = \varphi\left(\frac{i}{l+1}\right)$ and $\varphi(u) = \sqrt{12}(u - 0.5)$. Then $\| \mathbf{e} \|_W$ defined in (4.52) is the Wilcoxon Norm of the vector $\mathbf{e}$.

To derive the adaptive algorithm, the steepest-descent method is used:

$$\mathbf{w}_j = \mathbf{w}_{j-1} + \eta(\nabla_w J(j))^* \tag{4.53}$$

where $\nabla_w J(j)$ is defined as

$$\nabla_w J(j) = \frac{\partial J(j)}{\partial \mathbf{w}_{j-1}} = \sum_{i=1}^{L} a\left(R(e_i)\right) e_i = \sum_{i=1}^{L} a(i) u_{(i)} \tag{4.54}$$

The term η is is a gain constant that regulates the speed and stability of adaptation.

The proposed robust diffusion strategy based on Wilcoxon norm can therefore be described in general form as follows:

$$\begin{aligned} \boldsymbol{\theta}_k^{j-1} &= \sum_{l \in \mathcal{N}_k} c_{kl} \hat{\mathbf{w}}_k^{j-1}, \qquad \boldsymbol{\theta}_k(-1) = 0 \\ \hat{\mathbf{w}}_k^{(j)} &= \boldsymbol{\theta}_k^{(j-1)} + \eta_k \sum_{q=(j-1)L+1}^{jL} \sqrt{12}\left(\frac{q}{L+1} - 0.5\right) \mathbf{u}_{(q)k} \end{aligned} \tag{4.55}$$

for local step size η_k. The combiner may be non-linear, or even time variant, to reflect the changing topologies. That can be implemented by assuming the neighbourhood $\mathcal{N}_k$ is time variant. Here linear combiner model has been used for fixed network.

4.8 Simulation Results for Robust BDLMS Algorithm

The performance of the error saturation nonlinearity DLMS and diffusion Wilcoxon norm are evaluated through simulation study. Further, to assess the performance of the proposed method both theoretical and simulation results of ILMS are also plotted. All simulations are carried out using regressor input with shift structure. The desired data are generated according to the model discussed in Section 4.6.

The same example as in [32] is simulated for facilitating comparison. The network consists of seven nodes with each regressor size of (1×10) collecting data through a correlated process given by (3.38). The parameters are chosen randomly which

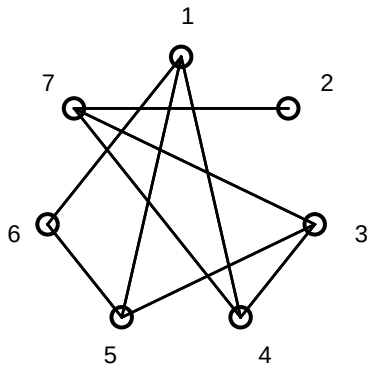
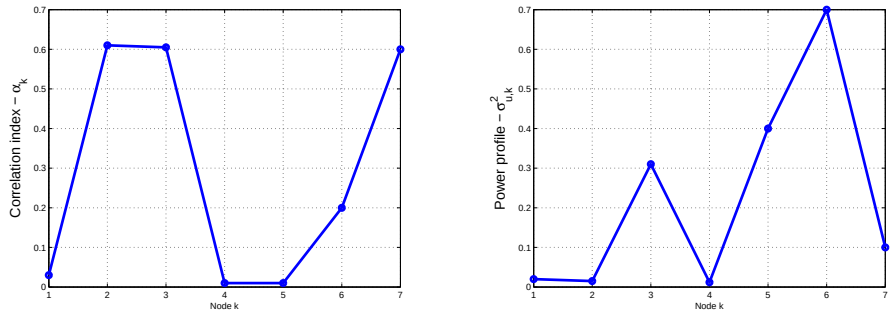


Figure 4.9: Network Topology

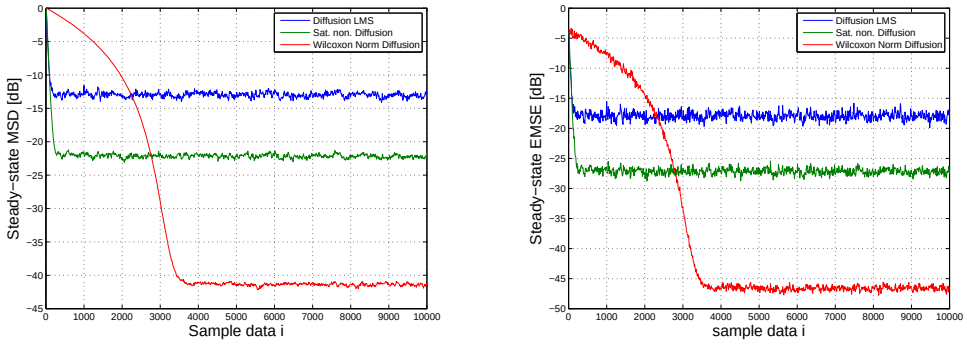
are same as in literature for comparison purpose and are depicted in Figures 4.9 and 4.10(a). The background Gaussian noise with variance $\sigma^2_{g,k}$ is also generated randomly and is shown in Figure 4.10(b).

To generate the performance curves, hundred independent experiments are performed and the results are averaged. The steady-state curves are generated by running the networks for 10000 iterations. The impulsive noise of having 10% occurrence is assumed. Both the saturation nonlinearity and Wilcoxon diffusion algorithms are robust to this. In terms of performance, the saturation nonlinearity diffusion algorithm gives fast convergence with inferrior performance compared to



(a) (b)

Figure 4.10: Network statistics. (a) Network co-relation index per node. (b) Regressor power profile.

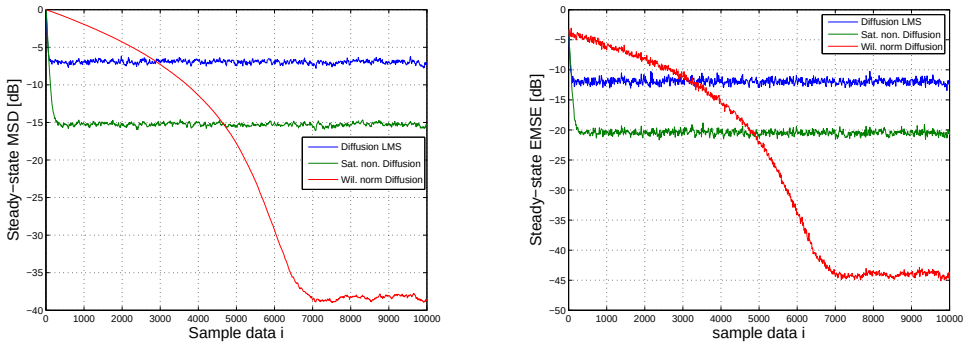


(a) (b)

Figure 4.11: Performance of algorithms with for $\mu = 0.05, \sigma^2_{sat} = 1, \sigma^2_g = 10^{-3}, \sigma^2_w = 10000\sigma^2_g, p_r = 0.10$ and the steps size in Wilcoxon norm is $\eta = 0.002$. (a) Global MSD. (b) Global EMSE

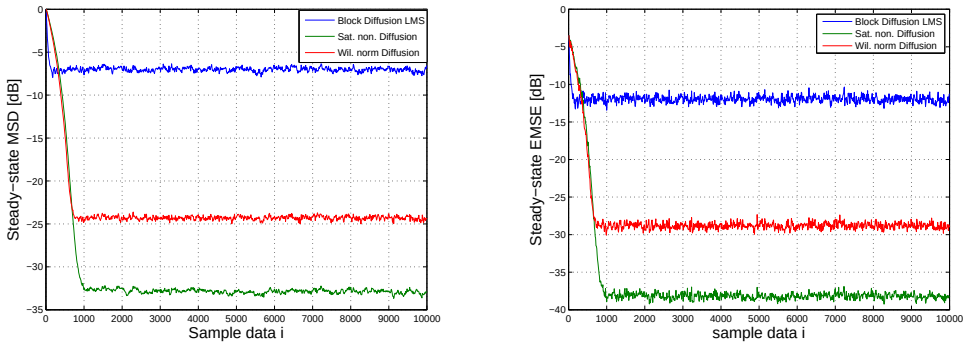
Wilcoxon diffusion. It is becase, in case of error saturatioin nonliniearty the error after mapping is used for weight updation where as in case of Wilcoxon norm algorithm the the norm based on the rank of error is minimized and also used for weight update. Thus what ever may be the error, the convergnce speed depends on the step size and the block size used to evalute the Wilcoxon norm. If the step size is increased , η then the convergnce speed increases, but the steady state perfromance decreases. The global MSD and EMSE curves are depicted in Figure 4.11(a) and Figure 4.11(b) respectively. If the probability of occurrence increased from 10% to 40%, the algorithms are still robust to outliers. This is shown in Figures 4.12(a) and 4.12(b) respectively.

But if the performance of the algorithms in terms of their convergence speed



(a) (b)

Figure 4.12: Performance of algorithms with for $\mu = 0.05, \sigma_{sat}^2 = 1, \sigma_g^2 = 10^{-3}, \sigma_w^2 = 10000\sigma_g^2, p_r = 0.40$ and the steps size in Wilcoxon norm is $\eta = 0.002$. (a) Global MSD. (b) Global EMSE



(a) (b)

Figure 4.13: Performance of algorithms with for $\mu = 0.05, \sigma_{sat}^2 = 0.01, \sigma_g^2 = 10^{-3}, \sigma_w^2 = 10000\sigma_g^2, p_r = 0.40$ and the steps size in Wilcoxon norm is $\eta = 0.02$. (a) Global MSD. (b) Global EMSE

are compared, the saturation nonlinearity diffusion algorithm gives much better performance compared to the Wilcoxon diffusion algorithm. This can be illustrated by simulating the network in presence of 40 percentage impulsive noise and changing some parameter of the algorithms. The MSD and EMSE curves are shown in Figure 4.13(a) and 4.13(b) respectively. From these Figures it is observed that for the same convergence speed, the error saturation nonlinearity based algorithm gives better steady state performance compared to the Wilcoxon diffusion algorithm.

In case of Wilcoxon norm diffusion algorithm, the error vector is weighted. The maximum error is then multiplied with less weight to minimize its affect on weight

update process. The computation involved in this method is more compared to saturation diffusion algorithm because of every time the errors must be ascending ordered in order to find the Wilcoxon norm.

4.9 Conclusion

This chapter presents the steady-state performance of error saturation nonlinearity incremental algorithm under the contaminated Gaussian impulsive noise which exhibits robust performance over the conventional ILMS. One of the key results of this work is to show the robust estimation in impulsive noise environment over wireless sensor network. The proposed algorithms using incremental distributed scheme need same communication resources as required in case of ILMS.

The steady-state expressions for MSE, EMSE and MSD have been derived which agree very well with corresponding simulation results.

The robust DLMS algorithm based on Wilcoxon norm and error saturation nonlinearity robust function also show robust performance against impulsive noise. The simulation results exhibits that the SNBLMS algorithm provides better performance compared to that of the WNDLMS.

In this work the error saturation nonlinearity LMS implementation operating with Gaussian inputs have been studied. This investigation can further be extended to other family of error nonlinearities like the LMF, Sign error *etc.* When both the desired and input data corrupted by impulsive noise, the Hubber Prior Error Feedback-Least Square Lattice (H-PEF-LSL) algorithm [54] have been employed in literature. Some other robust algorithms are also available in the literature which are robust to changing topology, nodal failure and link failures. In future the proposed analysis can be extended to study such situations using both incremental and diffusion distributed cooperative schemes over WSNs.

Chapter 5

Distributed DOA Estimation in Wireless Sensor Networks

Chapter 5

Distributed DOA Estimation in Wireless Sensor Networks

A novel distributed ML bearing estimation strategy is developed following the diffusion principle in WSNs. In this approach each sensor node estimates the source bearing by optimizing the ML function locally after forming a random array with its neighbours. Diffusion particle swarm optimization (DPSO) is proposed to optimize the ML function. During the optimization process each associated node shares its best information with the other nodes so that global estimation is achieved. The simulation results exhibit improved performance for the proposed method compared to that offered by the centralized MUSIC algorithm but the same is slightly inferior compared to the global ML-PSO algorithm.

5.1 Introduction

Bearing estimation is not possible using the observed data of a single node with a single sensor alone. It needs at least one neighbour to form an array, so that any existing array processing algorithm can be used to estimate the bearing. An alternative approach is to send all observations from various nodes to a central processor by considering the whole network as an arbitrary array. Centralized processing may not be practicable for a large network as it requires excessive communication overhead to deliver the data to, and process it at, a central processor. Further, it may not be possible to maintain the coherence between signals received from widely separated sensor nodes in large distributed WSNs. However, such centralized methods

have the potential to achieve optimum solution due to the availability of all data at processing point.

The DOA estimation is truly a distributed estimation problem. Accordingly a fully distributed localization algorithm in which nodes organize themselves into groups and then collaborate to locate the source is proposed here. In this formulation each node makes an arbitrary array with its neighbouring nodes and then estimates the DOA by optimizing a local ML function using diffusive mode of cooperation. For this problem the node has to be be aware of its locations because the array characteristic is dependent on the position of nodes. For effective localization, the best solution to have a global position system (GPS) receiver at every node. But the GPS receiver cannot be fitted in each and every sensor node because it increases both cost and size of the node, but also limits the outdoor applications. An alternate way to localize the node is to use small number of reference node called beacon. These mobile beacons assists the unknown nodes in localizing themselves [124].

Iterative search techniques discussed in Chapter 2 have been applied for optimization of the ML function. Accordingly the PSO algorithm is selected here and its diffusive distributed version [105] is proposed for estimation of the DOA. In this approach each node in the network optimizes its own local ML function and passes its estimate to its neighbours for local diffusion.

In other words each node in the network forms an arbitrary array to optimize the ML function. After collection of spatially uncorrelated data a node shares to its neighbours once in an experiment with their relative positions. The ML function is optimized at each node using local data and PSO tool. Subsequently the estimated parameters are diffused for further estimation. After learning properly with the use of distributed consensus algorithm each node in the network achieves global angle of incidence. The performance in terms of probability of resolution (PR) and root mean-square error (RMSE) are evaluated. It is observed that the proposed distributed estimation provides better performance at each node of the network compared to that obtained by non-cooperative way of estimation.

5.2 Data Model and Problem Formulation

Consider a sensor network that consists of N of sensor nodes distributed in the wavefield generated by $M(> N)$ narrowband far-field sources. The sources are positioned at $\boldsymbol{\theta} = [\theta_1, \theta_2, \ldots, \theta_M]^T$ where $[\cdot]^T$ represents the transpose operator. For far-field sources it is assumed that the sensors and the sources lie in the same plane [125]. Let the $N \times 1$ vector $\mathbf{a}_g(\theta)$ be the complex response of the global array to a source in direction θ. The complex $N \times M$ global steering matrix of an array of sensor nodes $\mathbf{A}_g(\boldsymbol{\theta}) = [\mathbf{a}_g(\theta_1), \ldots \mathbf{a}_g(\theta_M)]$ depends on their physical positions in the network. The (k,l)th element $\mathbf{A}_g(k,l)$ of the steering matrix $\mathbf{A}_g(\boldsymbol{\theta})$ is modeled by the complex gain (with respect to a reference point) of the lth signal at the kth node, i.e.

$$\mathbf{A}_g(k,l) = \exp\left\{ j\frac{2\pi}{\lambda}[x_k \sin\theta_l + y_k \cos\theta_l] \right\}$$
$$k = 1, 2, \ldots, N, \quad l = 1, 2, \ldots, M \tag{5.1}$$

in which λ is the wavelength of the signal and (x_k, y_k) is the position of the kth node. The complex observed N-vector $\mathbf{y}_g(i)$ from sensor nodes is modeled by the standard equation as in [126]

$$\mathbf{y}_g(i) = \mathbf{A}_g(\boldsymbol{\theta})\mathbf{s}(i) + \mathbf{v}_g(i), \qquad i = 1, 2, \ldots, L \tag{5.2}$$

where $\mathbf{s}(i)$ is the complex $M \times 1$ vector of signal source which is considered to be stochastic. The unpredicted noise vector $\mathbf{v}_g(i)$ is independent, identically complex normally distributed with zero mean and covariance matrix $\sigma^2\mathbf{I}_N$, where σ^2 is noise variance, $\mathbf{I}_N$ is the identity matrix of order $N \times N$ and L is the number of data samples (snapshots). Further, the vectors of signal and noise are assumed to be uncorrelated. The array covariance matrix is given by

$$\mathbf{R}_g = E[\mathbf{y}_g(i)\mathbf{y}_g^{\mathrm{H}}(i)] = \mathbf{A}_g(\boldsymbol{\theta})\mathbf{S}\mathbf{A}_g^{\mathrm{H}}(\boldsymbol{\theta}) + \sigma^2\mathbf{I}_N \tag{5.3}$$

where $\mathbf{S} = E[\mathbf{s}(i)\mathbf{s}^{\mathrm{H}}(i)]$ is the signal covariance matrix, $(\cdot)^{\mathrm{H}}$ denotes complex conjugate transpose and $E(\cdot)$ stands for expectation operator.

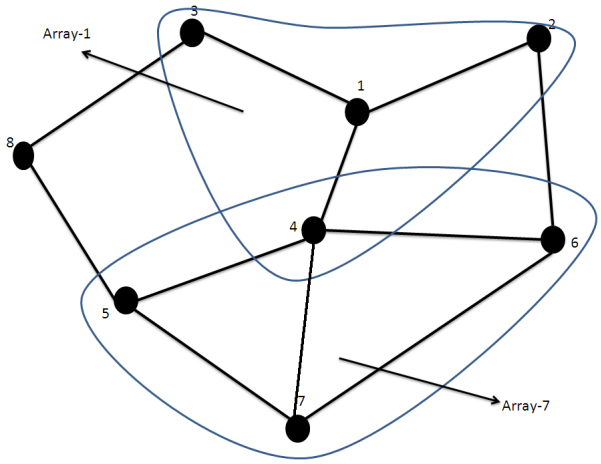


Figure 5.1: Formation of a random array at each node in the network

The problem of interest is to estimate the DOA $\boldsymbol{\theta}$ from the given observations $\{\mathbf{y}_g(i)\}_1^L$ in a distributed manner. Let $\mathcal{N}_k$ represents a set of sensor nodes in the neighbourhood of kth node including itself. The number of nodes connected to node k is called the degree of kth node, and is denoted by n_k. How the nodes constitute random array is shown in Figure 5.1. Subarray 1 is formed at node 1 with nodes 2,3,4 and 1 itself. Similarly subarray 7 is constituted at the 7th node with nodes 4,5,6 and 7. In this way every node in the network forms a random array with its immediate neighbours. Now the complex n_k-vector $\mathbf{y}_k(i)$ for the output of kth subarray becomes

$$\mathbf{y}_k(i) = \mathbf{A}_k(\boldsymbol{\theta})\mathbf{s}(i) + \mathbf{v}_k(i), \qquad i = 1, 2, \ldots, L \tag{5.4}$$

where $\mathbf{A}_k(\boldsymbol{\theta})$ is the $n_k \times M$ steering matrix and the complex n_k-vector $\mathbf{v}_k(i)$ is the noise process.

Now the DOA at the kth subarray can be estimated by using algorithms like MUSIC or the Capon beam former etc. In MUSIC, for example, the sample covariance matrix of the measured data is estimated first as

$$\hat{\mathbf{R}}_k = \frac{1}{L}\sum_{i=1}^{L} \mathbf{y}_k(i)\mathbf{y}_k^{\mathrm{H}}(i) \tag{5.5}$$

and its eigenvalues and corresponding eigenvectors are evaluated from which the

noise subspace is determined to give an estimate of the spatial spectrum and corresponding estimates of the source DOA's. After estimation at each subarray, each node shares its estimated values with its neighbours. For diffusion mode learning, each node then needs to update its own estimate using those from its neighbours. There does not appear to be a mechanism for doing this within the context of the MUSIC or Capon source location without passing additional data or covariance matrices between nodes. Therefore these algorithms have not been employed for diffusion based bearing estimation. On the other hand ML based DOA estimation not only has well known performance advantages over these algorithms at low signal-noise ratio and low sample support but also is formulated as a cost function optimization task that permits the passing of DOA estimates in the network.

5.2.1 Maximum Likelihood Estimation

Two different types of ML models are used; conditional model (CM) which assumes the signal to be deterministic and unconditional model (UM) which assumes the signals is random. But the authors in [127] proved that for uncorrelated sources, the statistical performances of CML and UML are similar; while for highly correlated or coherent sources, the performance of UML is significantly superior. Therefore, in order to obtain the bearing estimation in distributed manner, the proposed work is on optimizing the UML function.

Following [126], the UML estimates of DOA at kth node is obtained by minimizing negative log-likelihood function (5.6)

$$f_k(\boldsymbol{\theta}) = \log \left| \mathbf{P}_{\mathbf{A}_k} \hat{\mathbf{R}}_k \mathbf{P}_{\mathbf{A}_k}^H + \frac{\mathbf{P}_{\mathbf{A}_k}^{\perp}}{N-M} \mathrm{tr}\left[\mathbf{P}_{\mathbf{A}_k}^{\perp} \hat{\mathbf{R}}_k \right] \right| \tag{5.6}$$

where tr[$\bullet$] is the trace of the bracketed matrix; $\mathbf{P}_{\mathbf{A}_k} = \mathbf{A}_k(\mathbf{A}_k^H \mathbf{A}_k)^{-1}\mathbf{A}_k^H$ is the projection of matrix $\mathbf{A}_k$ and $\mathbf{P}_{\mathbf{A}_k}^{\perp} = \mathbf{I} - \mathbf{P}_{\mathbf{A}_k}$ is the orthogonal complementary. The sample covariance matrix $\hat{\mathbf{R}}_k$ is defined in (5.5). The dependence of $\mathbf{A}_k$ on $\boldsymbol{\theta}$ is suppressed here for notational simplicity.

However the UML for the global array, where the whole network acts as an random array, is obtained by replacing the local steering matrix $\mathbf{A}_k$ by the

global steering matrix $\mathbf{A}_g$ and the local sample covariance matrix $\hat{\mathbf{R}}_k$ by $\hat{\mathbf{R}}_g = \frac{1}{L}\sum_{i=1}^{L}\mathbf{y}_g(i)\mathbf{y}_g^{\mathrm{H}}(i)$. The global UML is given by

$$f_g(\boldsymbol{\theta}) = \log\left|\mathbf{P}_{\mathbf{A}_g}\hat{\mathbf{R}}_g\mathbf{P}_{\mathbf{A}_g}^{H} + \frac{\mathbf{P}_{\mathbf{A}_g}^{\perp}}{N-M}\mathrm{tr}\left[\mathbf{P}_{\mathbf{A}_g}^{\perp}\hat{\mathbf{R}}_g\right]\right| \tag{5.7}$$

Now each node has its local UML function $f_k(\boldsymbol{\theta})$ in (5.6) which is different for different node. The difficulty in ML estimation is the multimodal nature of the cost function. This is mostly overcome in PSO based ML solutions [46, 128]. During the evolution of PSO each node can share their estimated angle of arrival with their neighbours. Therefore every node can diffuse the neighbour's estimated DOA in order to make faster and accurate estimation. Now it can incorporate the idea of diffusion PSO with distributed ML estimation.

5.3 Distributed DOA Estimation

The bearing estimation problem in distributed WSNs is formulated as distributed optimization problem. After collecting the required observed data, each sensor node formulates its ML cost function and then attempts to estimate DOA individually by solving its own cost function. Then the probability density function (pdf) for single snapshots at kth node is given as [88, 126]

$$p_k(\mathbf{y}_k(i)|\boldsymbol{\theta}) = \frac{1}{\det[\pi\mathbf{R}_k]}\exp\left\{-(\mathbf{y}_k^{H}\mathbf{R}_k^{-1}\mathbf{y}_k)\right\} \tag{5.8}$$

where $\mathbf{R}_k$ is the covariance matrix of the kth subarray. The joint pdf $p_k(\mathbf{y}(i)|\boldsymbol{\theta})$ for L number of independent snapshots is defined as [88, 126]:

$$p_k(\mathbf{y}_k|\boldsymbol{\theta}) = \prod_{i=1}^{L}\frac{1}{\det[\pi\mathbf{R}_k]}\exp\left\{-\mathbf{y}_k^{H}(i)\mathbf{R}_k^{-1}\mathbf{y}_k(i)\right\} \tag{5.9}$$

For bearing estimation, the cost function is a log-likelihood of the form [88, 126]

$$f(\boldsymbol{\theta}) = \ln\prod_{i=1}^{L}p(\mathbf{y}(i)|\boldsymbol{\theta}) = \sum_{i}\ln p(\mathbf{y}(i)|\boldsymbol{\theta}) \tag{5.10}$$

The use of simple product of probabilities assumes that the snapshots are independent random events. The log-likelihood expression is convenient because it trans-

forms the product to summation and in addition to Gaussian densities the log function inverts the exponential function.

For distributed bearing estimation, further approximation made in (5.11) is valid.

$$p(\mathbf{y}(i)|\boldsymbol{\theta}) \approx \prod_{k=1}^{N} p_k(\mathbf{y}_k(i)|\boldsymbol{\theta}) \tag{5.11}$$

where $p_k(\mathbf{y}_k(i)|\theta)$ is the likelihood function of observing data for given θ. As the snapshots at neighbouring nodes share common data sample, the use of product to describe the probability of simultaneously observing all the node snapshot vectors $\{\mathbf{y}(i)\}_k$ at time i is not technically correct. The events are clearly not independent. However the use of the approximation by substituting (5.11) in (5.10) yields

$$f(\boldsymbol{\theta}) \approx \sum_{k=1}^{N} \left\{ \sum_{i=1}^{L} \ln p_k(\mathbf{y}(i)|\boldsymbol{\theta}) \right\} \tag{5.12}$$

which is identical to the form to costs functions used in distributed optimization [22–24]. Thus in invoking a distributed optimization strategy, it would expect some loss in performance associated with the approximation of (5.12).

In distributed optimization methods using consensus algorithm [22], the problem is to optimize the sum of convex functions corresponding to N sensor nodes. The goal of each node is to co-operatively solve the optimization problem

$$\text{minimize} \sum_{k=1}^{N} f_k(\boldsymbol{\theta}) \tag{5.13}$$

subject to $\theta \in \mathbb{R}^M$

where each $f_k(\theta)$ is assumed to be a convex function, representing the local objective function of sensor node k which is known to that particular node only.

Comparing (5.12) and (5.13) the local cost function is given as

$$f_k(\boldsymbol{\theta}) \approx \sum_{i=1}^{L} \ln p_k(\mathbf{y}(i)|\boldsymbol{\theta}) \tag{5.14}$$

The right hand side of (5.14) can be formulated as the local ML function given in (5.6).

In essence it implies that each node in the network has a different cost function

that completely depends on the connectivity among its neighbours. Since the ML function is multimodal, the cost functions at each node are not convex. However they share a common global maximum (all provide unbiased bearing estimates) but may not share local maxima. In a small enough region around this global maximum, all the functions are approximately convex. Therefore, if it is possible to initialize an iterative search algorithm within this region at each of the node, the theory [22] will apply and consensus will be achieved. PSO has the ability to get close to the global maximum at each of the nodes to facilitate consensus.

The effectiveness of the PSO algorithm is that in a multimodal function it can search the region where the global solution lies. Further it enables the individual sensor nodes to interact with other neighbouring nodes to make the search process faster and accurate. So it needs a consensus algorithm that serves a basic mechanism for distributing the optimization among the sensor nodes and allowing us to solve the problem in a decentralized manner. The global objective is to minimize the cost function defined in (5.13). Considering the nature of the present problem and the capability of PSO, the distributed PSO algorithm is introduced for solving (5.13).

In this formulation each node runs the PSO algorithm for the optimization of its own cost function and simultaneously shares the best solution with the neighbouring nodes. In particular, each sensor node starts with an initial estimate $\boldsymbol{\theta}_k(0)$ and updates its estimate at discrete times $t_i, i = 1, 2, ...$ Let $\boldsymbol{\theta}_k(i)$ be the bearing vector estimate by the kth node at time t_i. While updating, a sensor node combines its current estimate $\boldsymbol{\theta}_k(i)$ with the estimates $\boldsymbol{\theta}_j(i)$ received from its neighbouring nodes j. The update equation is given as

$$\boldsymbol{\theta}_k(i+1) = \sum_{j=1}^{N} c_{kj}(i)\boldsymbol{\theta}_j(i) \tag{5.15}$$

where the scalars $c_{kj}(i)$ are the factor by which the estimates of others nodes $\boldsymbol{\theta}_1(i), \ldots, \boldsymbol{\theta}_N(i)$ contribute to the kth node. These weights are assumed to be the same for static network where the topology remains unchanged with time. In particular, when the neighbour j communicates with node k which includes the node itself, the weight assigned is given as c_{kj}, which is positive. If a neighbour j do

not communicate with k, then the weight assigned is $c_{kj} = 0$. In [35, 36] different methods have been proposed to calculate these weights according to the network topology. In the present case the metropolis weight selection rule is used as it is best suited for distributed implementation [22, 35].

5.3.1 Communication Overhead of the Proposed Algorithm

In a sensor network, communication is a key energy consumer [5]. Therefore it is of great importance to minimize the communication among nodes by processing the data as much as possible inside the network locally. In the present case an upper bound of the average number of messages transmitted by a node and the overall communication for the network is determined.

In multihop communication, let h be the maximum distance between any node in the network and the FC, where the distance between any two nodes is the minimum number of hops between them. Let d be the average degree of a node. The proposed algorithm involves two phases: a data sharing phase and the distributed estimation phase. The overall communication overhead is the sum of the number of messages in these two phases. In the data sharing phase, each node shares L complex data samples with its neighbours. For this the number of messages transmitted is $O(NLd)$. During the estimation phase, at each iteration every node in the network shares M ($M << L$) real data samples with its neighbours. So the total number of messages transmitted by all the nodes in a cycle is $\sum_{k=1}^{N} n_k \approx Nd$. Let I be the number of cycles required to attain the steady state value in the network. Then, the total communication cost for the estimation phase is $O(NMId)$. The overall communication overhead for the network is $O(NLd) + O(NMId)$. Since $L < MI$, the number of messages needed is

$$\mathcal{C}_{MLDPSO} = O(NMId) \tag{5.16}$$

In centralized estimation, each node in the network sends L received data samples to the FC. In the worst case scenario where message aggregation is not used, the message unicast from a leaf node in the spanning tree, will be forwarded h times till

Table 5.1: Comparison of Communication Overheads for Different Algorithms

Algorithm	Distributed PSO-ML	Centralized PSO-ML	Centralized MUSIC
Overall communication overhead	$O(NMId)$	$O(NLdh^3)$	$O(NL'dh^3)$

it reaches the FC. Therefore, the overall communication overhead for the network in the centralized case is [109]

$$\mathcal{C}_{MLPSO} = O(NLdh^3) \tag{5.17}$$

In general, a greater number of snapshots are required for the centralized MUSIC algorithm say $L' > L$. The communication overhead for this algorithm is

$$\mathcal{C}_{MUSIC} = O(NL'dh^3) \tag{5.18}$$

The overall comparison between all the algorithms are given in Table 5.1. From the table it is observed that the saving in overall communication overhead is $O(Lh^3/MI)$. In a large distributed multihop network, h is large enough for sending the data to the FC. The number of cycles needed to attain the steady state value I for PSO is less (maximum 50). So the ratio $Lh^3/MI >> 1$. So there is substantial saving of at least $O(h^2)$ number of communications.

5.4 Distributed Particle Swarm Optimization (DPSO)

The PSO is a population based stochastic optimization technique developed in 1995 [100, 101] which is inspired by social behavior such as bird flocking or fish schooling. The PSO starts with random positions of the individuals which represent random gausses in the search space. These individuals are candidate solutions to the problem under consideration. Each member of the swarm associated with a random velocity, helps to update its position. Each of the particle iteratively updates its position and velocity according to its own as well as the group's flying experiences that is the *pbest* (own flying experience) and *gbest* (groups flying experience) information.

Consider a swarm at each node having P particles. The particles are represented by the M dimensional positions and velocity vectors. Let the position and velocity vectors of the jth particle for the kth node be denoted by $\mathbf{x}_{jk}^i = [x_{jk1}^i, x_{jk2}^i, \ldots, x_{jkM}^i]^T$ and $\mathbf{v}_{jk}^i = [v_{jk1}^i, v_{jk2}^i, \ldots, v_{jkM}^i]^T$ respectively. These particles fly through the hyperspace (i.e., $\mathbb{R}^M$) and have two essential reference points: their own best as well as the global or their neighborhood's best position. Let the *pbest* of the jth particle at the kth node be $\mathbf{p}_{jk}^i = [p_{jk1}^i, p_{jk2}^i, \ldots, p_{jkM}^i]$ and that of *gbest* be $\mathbf{p}_{gk}^i = [p_{gk1}^i, p_{gk2}^i, \ldots, p_{gkM}^i]$. In a minimization formulation each particle of the swarm at the kth node updates its velocity and position according to the following equations:

$$\mathbf{v}_{jk}^{i+1} = \omega^i \mathbf{v}_{jk}^i + c_1 \mathbf{r}_1^i \odot (\mathbf{p}_{jk}^i - \mathbf{x}_{jk}^i) + c_2 \mathbf{r}_2^i \odot (\mathbf{p}_{gk}^i - \mathbf{x}_{jk}^i) \tag{5.19}$$

$$\mathbf{x}_{jk}^{i+1} = \mathbf{x}_j^i + \mathbf{v}_{jk}^{i+1} \tag{5.20}$$

where $\odot$ denotes the element-wise product, $j = 1, 2, \ldots, P$; $i = 1, 2, \ldots,$ indicates the iterations and $k = 1, 2, \ldots, N$ indicates the node number. The velocity is updated based on the current value scaled by *inertial weight* ω and increased in the direction of *pbest* and *gbest*. The constants c_1 and c_2 are scaling factors that determines the relative pull of the particles' best location *pbest* and global best location *gbest*. These positive constants are sometimes referred to as *cognitive* and *social* parameters respectively [101, 104]. Two independent M-dimensional random vectors r_1 and r_2 consisting of independent random numbers uniformly distributed between 0 and 1 are used to stochastically vary the relative pull of *pbest* and *gbest*. In order to simulate the unpredictable components of the natural swarm behavior, these random elements are introduced in the optimization process [104].

In a distributed bearing estimation, the problem is how do different nodes mutually share their *gbest* information with other nodes. For this purpose, diffusion based distributed PSO algorithm is chosen using a similar cooperation strategy described in [32]. In DPSO each node updates its particle's positions, velocities and *pbest* using its local objective function and shares its *gbest* vector with the other neighbouring

nodes of the network. There are two ways a node can utilize the received *gbest*.

In the first the estimated global best information of different nodes are exchanged among neighbouring nodes. Then each node locally compares all the received *gbest* including its own and then selects the best of the global best positions by comparing their fitness values and use the same for updating the velocity and position of particles of that node. The velocity and position update equations remain same as given in (5.19) and (5.20). This approach may not be proper because of the difference in the cost function.

In the second case, at every node a consensus mechanism given in (5.21) is used to fuse the received *gbest* estimates,

$$\mathbf{p}_{gk}^{(i-1)} = \sum_{l \in \mathcal{N}_k} c_{kl} \mathbf{p}_{gk}^{(i-1)} \tag{5.21}$$

where $\mathcal{N}_k$ is the set of neighbouring nodes of kth node. The metropolis weight selection procedure to determine the combiner coefficients c_{kl} is [35]

$$c_{kl} = \begin{cases} \frac{1}{\max(n_k, n_l)}, & \text{if } k \neq l \text{ are linked} \\ 0, & \text{for } k \text{ and } l \text{ are not linked} \\ 1 - \sum_{l \in \mathcal{N}_k / k} c_{kl}, & \text{for } k = l \end{cases} \tag{5.22}$$

where n_k and n_l denote the degree for nodes k and l.

The fused $\mathbf{p}_{gk}^{(i-1)}$ is used in the local optimization process to estimate the global DOA, so that it can rapidly respond to change in its neighborhood. Subsequently the position and velocities of all particles at nodes are simultaneously evaluated by taking diffused *gbest* as the previous *gbest* of the node. The computed global best information is exchanged between all participating nodes for further processing.

5.5 DPSO for DOA Estimation in WSNs

The formulation of the DPSO algorithm for ML optimization to estimate source DOA's is dealt in this section. At first, every node in the network starts by initializing a population of particles in the search space with random positions constrained between 0 and π in each dimension and random velocities in the range of 0 to π.

The M-dimensional position vector of the kth node of jth particle takes the form $x_{kj} = [\theta_{k1}, \ldots, \theta_{kM}]$, where $\boldsymbol{\theta_k}$ represents the DOAs of the kth node. A particle position vector is converted to a candidate solution vector in the problem space. The score of the mapped vector evaluated by a likelihood function f_k which is given in (5.6), is regarded as the fitness of the corresponding particle.

The velocity update equation of (5.19) acts as a key to the entire optimization process. Three components typically contribute to the new velocity. The first part refers to the inertial effect of the movement, which is just proportional to the old velocity and has a tendency to guide particles to proceed in the same direction. The inertial weight w is considered critical for the convergence behavior of PSO [129]. A larger w facilitates searching new areas and global exploration while a smaller w facilitates local exploitation in the current search area. In this study, w is selected so that it decreases during the optimization process, thus the PSO tends to have more global search ability at the beginning of the run while having more local search ability towards the end.

Given a maximum value $w_{\max}$ and a minimum value $w_{\min}$, w is updated as follows:

$$w^i = \begin{cases} w_{\max} - \frac{w_{\max} - w_{\min}}{rK}(i-1), & \text{if } 1 \leq i \leq [rK] \\ w_{\min}, & \text{for } [rK]+1 \leq i \leq K \end{cases} \tag{5.23}$$

where $[rK]$ is the number of iterations with time decreasing inertial weight, $0 < r < 1$ is a ratio, K is the maximum iteration number and $[\cdot]$ is a rounding operator. Based on the literatures [130], in the present case, select $w_{\max} = 0.9, w_{\min} = 0.4$, and $r = 0.4 \sim 0.8$ are selected.

The second and third components of the velocity update equation introduces stochastic tendencies to return towards the particle's own as well as the group's best historical positions. The constants c_1 and c_2 are used to bias the particle's search towards the two best locations. Following common practice in the literature [104], $c_1 = c_2 = 2$, have been chosen. Since there is no mechanism for controlling the velocity of a particle, it is necessary to define a maximum velocity to avoid the swarm divergence [131]. The velocity limit can be applied along each dimension at

every node as

$$v_{jk}^{i} = \begin{cases} V_{\text{MAX}}, & \text{if } v_{jk}^{i} > V_{\text{MAX}} \\ V_{\text{MIN}}, & \text{if } v_{jk}^{i} < V_{\text{MIN}} \end{cases} \tag{5.24}$$

where $k = 1, \ldots, N$. In this work, V_{MAX} is limited to be half of the dynamic range. The new particle position is calculated using (5.11). If any dimension of the new position vector is less than zero or more than π, it is adjusted to lie within this range. The iteration is terminated if the specified maximum iteration number K is reached. The final global best position p_{kg} is taken as the ML estimates of source DOA. The main steps of DPSO-UML algorithms are outlined in Algorithm 3.

5.6 Simulation Results and Discussions

Here a numerical example is presented to demonstrate the performance of the distributed PSO based DOA estimation against centralized PSO-ML and MUSIC [91] which is the best known and well investigated algorithm. The performances of these methods are compared in two ways: (a) RMSE, which is calculated as [45]

$$\text{RMSE} = \sqrt{\frac{1}{MN_{\text{run}}} \sum_{i=1}^{N_{\text{run}}} \sum_{l=1}^{N} \left(\hat{\theta}_l(i) - \theta_l\right)^2} \tag{5.25}$$

where M is the number of sources, $\hat{\theta}_l(i)$ is the estimate of the lth DOA achieved in the ith run, θ_l is the true DOA of the lth source; and (b) the ability to resolve closely spaced sources known as PR. By definition, two sources are said to be resolved in a given run if both $|\hat{\theta}_1 - \theta_1|$ and $|\hat{\theta}_2 - \theta_2|$ are smaller than $|\theta_1 - \theta_2|/2$.

At first the network elements are initialized such that each node is aware of its position and the knowledge of connectivity with neighbour sensor nodes. According to the network topology, the combiner matrix known as metropolis weight is determined for local diffusion. The metropolis weights are computed with two rounds of communication between every pair of neighboring nodes [35, 36]. In the first round, each node calculates its degree by counting the number of its neighbours. In the second round, each node sends its degree information to all its neighbors. While calculating this, a node does not need any global knowledge of whole network, or

Algorithm 3: Main steps of DPSO-ML algorithm

Setup Problem:
- Define WSNs with topology.
- Initialize the location of each node.
- Calculate metropolis for diffusion.
- Set up random array at each node with its connecting neighbours.
- Define problem space.
- Define local fitness function at each node.
- Select PSO Parameter.

Swarm Initialization at each node:
- Random Positions.
- Random velocities.

for *each iteration* **do**
  for *each node* **do**
    for *each particle* **do**
      Map particle location to solution vector in solution space;
      Evaluate the objective function of current iteration of the node according to its local ML function (5.6);
      According to fitness value update particles best location p_k^i and group best location g_k^i;
      Update particles velocity according to (5.19);
      if *velocity exceeds the limits* **then**
        Limit particles velocity using (5.24);
      end
      Update particle position using (5.20);
      if *particles position out of solution boundaries* **then**
        Clip or adjust particles position;
      end
    end
    Share its *gbest* to all neighbour nodes;
  end
  Do the local diffusion according to (5.21);
  Check termination criterion;
end

even the number of nodes present in the network. In case of static network this metropolis remains constant, but for dynamic network it changes accordingly to the changing topology.

The performances obtained through the simulation study are plotted by varying SNR from -20 dB to 30 dB with a step size of 1 dB by using PSO, diffusion PSO and MUSIC algorithms. The MUSIC algorithm was run for 500 Monte Carlo (MC) simulations and for 2000 snapshots to achieve the RMSE close to the theoretical CRLB. But the DPSO-UML and PSO-UML algorithm run for 100 MC simulations by taking only 20 snapshots. For comparison purposes, a subarray formed at node 1 with the nodes connecting to its immediate neighbours has taken and estimate the DOA using the PSO-ML algorithm without any co-operation. In present study a static network with $N = 16$ nodes is considered. In the network every node is equipped with an arbitrary array. It is further assumed that two equal power uncorrelated signal sources are impinging on the sensor network with true DOA's $[130^\circ\ 138^\circ]$ with respect to the x-axis. Two examples are provided here with different connectivity.

5.6.1 Example 1

In the first example a network with less connectivity among the nodes has been chosen. The network topology is shown in Figure 5.2 by taking axis scales as $\lambda/2$ units. The network can be enlarged to any number of nodes, but to show the validity of the proposed algorithm, a network with 16 nodes is considered as an illustration. Further, if the scale of the network is enlarged, then the computer simulation will require longer time for the ensemble average results. It is because of the fact that in the proposed algorithm individual node runs the PSO algorithm for certain number of iteration, say I. In order to get PR performance the experiments are repeated for L times. Again the simulation is done for certain range of SNRs say R. Now the PSO algorithm is running for $N * L * R$ times. In the present example $N = 16$, $L = 100$, and $R = 51$. Therefore in order to ensembles the results of the example provided in the manuscript, the PSO (30 particles, 200 iterations) is running for 81600 times, for this simulation, around 30 Hr (PC: Windows XP, Intel Core 2 duo, 4GB RAM)

is required. In a real scenario, the situation is different where the PSO is running parallely once by the individual node in the network to estimate the bearing of the sources.

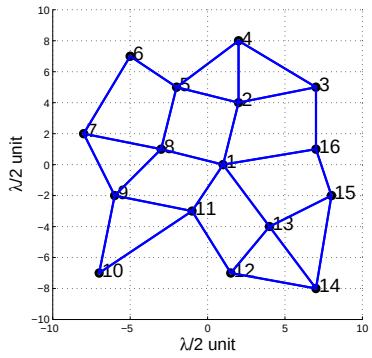


Figure 5.2: Network topology used in Example 1

At first the DOA is estimated by optimizing the ML function globally by the use of PSO. It is considered that the node #1 as a central node to compute the global DOA and all other nodes shared their received data to it. Subsequently every node in the network estimate the bearing in completely distributed manner by optimizing f_{ML} locally. Figure 5.3 describes the RMSE in DOA estimation obtained using global PSO-ML, DPSO-ML and MUSIC as a function of SNR. Figure 5.4 shows the resolution probability achieved by these methods.

It is evident from Figure 5.3 and 5.4 that the distributed PSO-ML produces superior performance compared to those obtained by global MUSIC algorithm as it provides lower RMSE and higher PR. The global PSO-ML offers the best performance than the distributed PSO-ML algorithm, because in the centralized estimation all informations are available while estimating the DOA. To achieve theoretical bound CRLB the global MUSIC algorithm needs 2000 number of snapshots where as the PSO-ML requires only 20 snapshots. The number of snapshots for PSO-ML is chosen keeping in view the communication burden in the distributed estimation.

Figure 5.5(a) shows that the global ML cost function is like a dip well on the flat surface where minima lies exactly at the true solution. Due to such characteristics of the fitness function, at times it is difficult for the PSO particles to search for the global minima. However such ambiguity can be overcome by imposing some more

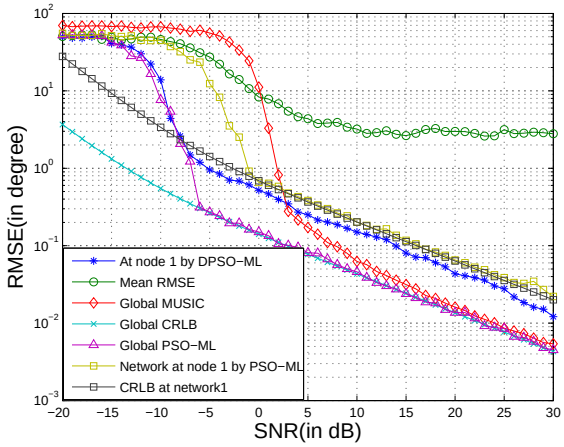


Figure 5.3: RMSE vrs. SNR plots in DOA estimation by global PSO, DPSO and MUSIC methods

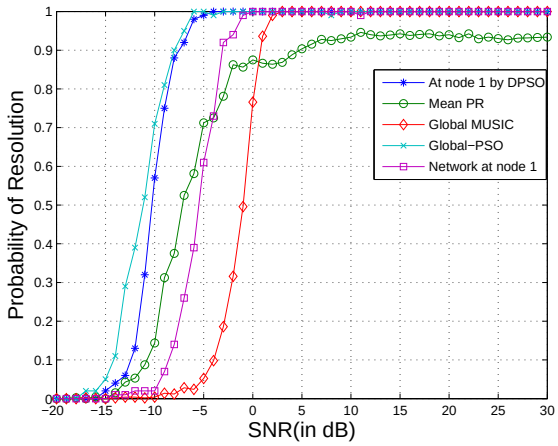
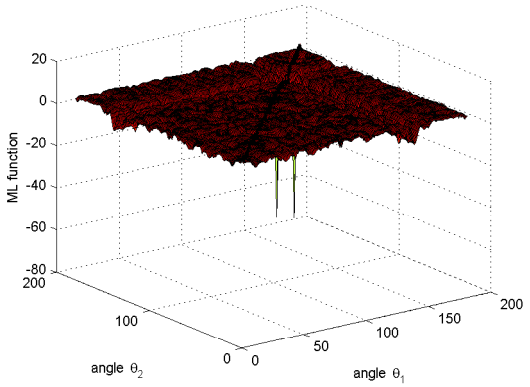
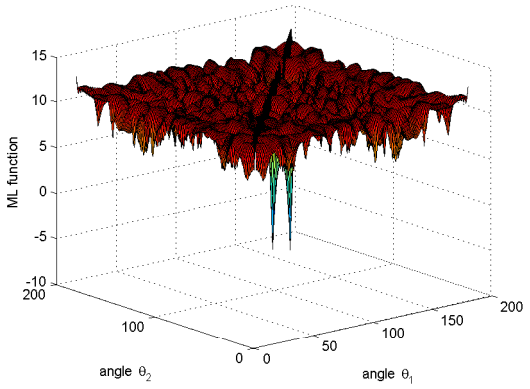


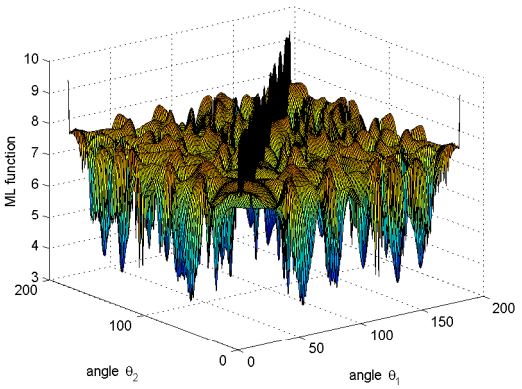
Figure 5.4: PR vrs. SNR plots in DOA estimation by global PSO, DPSO and MUSIC methods



(a) ML function of global network



(b) ML function at node 1



(c) ML function at node 6

Figure 5.5: ML functions of Example 1

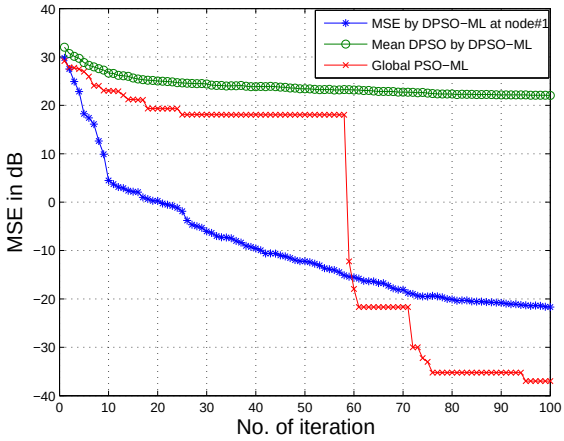


Figure 5.6: MSE variations curve by global PSO and DPSO

constraints in PSO algorithm. The local cost function which is shown in Figure 5.5(b) contains a global minima on flat surface by which it provides large space to search. Hence the probability of falling to minima is more and particles tend to find the minima in the vicinity of global minima.

When the network is fully connected and proper mixing of nodes estimates is ensured, then it is expected that all the nodes of distributed optimization method using consensus algorithm attain the same estimate after some recursions. This performance should be at par or better than that provided by the centralized method. This is the philosophy behind the development of any distributed algorithm and also behind the analysis of its performance. But in the present case every node could not give identical performance. The node which is connected to more number of neighbour nodes yields better estimation compared to the nodes which have less connectivity. This observation is evident from Figure 5.5(c) where the ML function at node 6 is displayed. There are so many other minima whose values are at par or more than that of global minima specially those which are located at the edge of the network having less connection. Therefore the average performance degrades. At lower SNR the performance is observed to be much better than that of MUSIC algorithm where as at high SNR the resolution PR does not reach unity because of edge node performance. Some of the nodes are connected to only two neighbours as a results three sensors forms a random array to estimate two sources DOA. Therefore

Table 5.2: Comparison of Performances for Different Algorithms in Example 1

RMSE performance in dB	At node 1 using DPSO-ML	Average of all nodes using DPSO-ML	At node 1 using PSO-ML without co-operation	Global PSO-ML	Global MUSIC
At -7 dB SNR	1.72	15.56	14.01	0.88	17.69
At -5 dB SNR	-0.21	14.36	10.93	-5.66	17.46
At 0 dB SNR	-2.85	9.19	-1.61	-8.23	10.47
At 5 dB SNR	-6.01	6.39	-4.35	-10.91	-7.66
Overall communication overhead	$O(6.99\times10^3)$	$O(6.99\times10^3)$	$O(1.4\times10^3)$	$O(1.12\times10^4)$	$O(1.12\times10^6)$

their performances are poorer than the interior nodes in the network, as a result the average performance degrades.

In general, the bearing estimation ability of an array increases with the number of elements presents in the array. If the degree of the connectivity of a node is two and trying to estimate the DOA of two sources, then whatever may be the algorithm used the performance will be very poor. Practically the nodes present in the interior of the network have degree more than two compare to the nodes present at the edges of the network. Therefore if the average degree of the nodes is less, then the overall performance is poor. But at the same time the overall communication is also less. So we have to compromise between communication overhead and performance. For any connectivity the performance of the distributed algorithm is better compared to individual subarray produced at each node.

From the Figures 5.3 and 5.4 and the Table 5.2, it is found that the performance (both in PR and RMSE) of the proposed algorithm is same with that of global PSO-ML up to -7 dB SNR. In between -7 dB to -1 dB the RMSE Performance of the DPSO-ML algorithm at node 1 is better than the PSO-ML algorithm. But at high SNR, nearly 2dB gain over non-cooperative estimation is achieved. The communication overheads required for the proposed method is in between non-cooperative

PSO-ML and global PSO-ML algorithm. The performance at every node is better in distributed estimation compared to non-cooperative estimation. If the connectivity of the network is increased, then the distributed performance is also increased. This can be observed in the following Example 2.

The manner in which the mean square error (MSE) decreases with evolution is depicted in Figure 5.6. It is observed that the MSE in DPSO-ML at node 1 converges much faster than that of global PSO-ML which is used for local diffusion. However its performance is not good compared to that of global PSO-ML. In case of global PSO-ML it requires more number of iterations to find the minima, but its performance is superior.

5.6.2 Example 2

In order to achieve better performance, in this example a network having more connectivity is considered (i.e. the minimum degree of connectivity is not less than 3). A sensor network with $N = 16$ with same position as in Example 1 is considered here. The only difference is the connectivity. The network topology is shown in

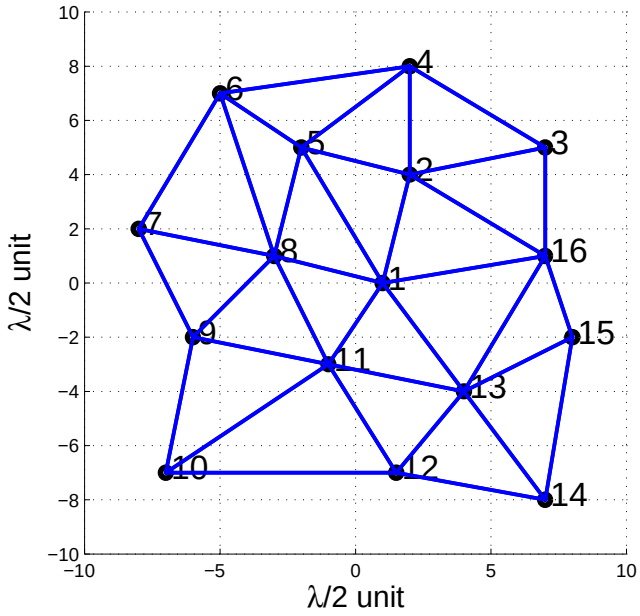


Figure 5.7: Network topology used in Example 2

Figure 5.7. The performance obtained through the simulation study are plotted in

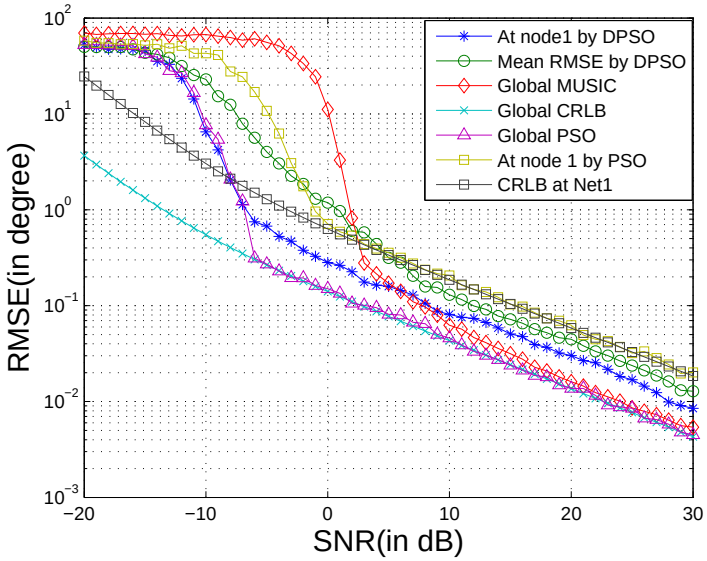


Figure 5.8: RMSE vrs. SNR plots in DOA estimation by global PSO, DPSO and MUSIC methods

Figure 5.8 and 5.9 by varying SNR from -20 dB to 30 dB with a step size of 1 dB by using global PSO-ML, DPSO-ML and MUSIC algorithms.

It is shown in these figures that each of these algorithms has provided better performance if the estimation is carried out globally. As can be seen from Figure 5.8, the RMSE performance at node 1 is in between that obtained with global PSO-ML and performance of the subarray at node 1 without diffusion. At lower SNR (around -20 dB to -5 dB) the overall performance of distributed estimation using diffusion PSO of node 1 and that using the global PSO is nearly the same. As

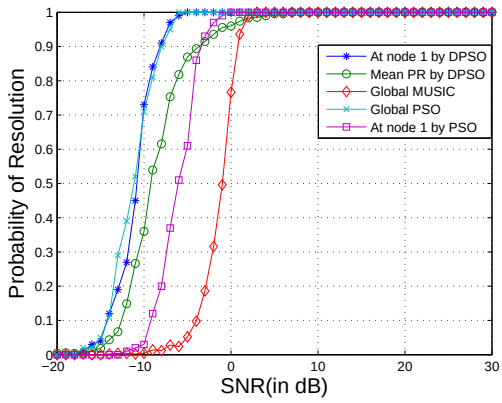


Figure 5.9: PR vrs. SNR plots in DOA estimation by global PSO, DPSO and MUSIC methods

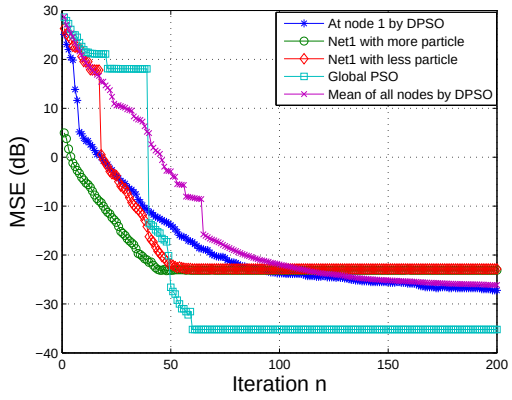


Figure 5.10: MSE(dB) variation by global PSO and DPSO based methods

the SNR increases the global bearing estimation is better because of the nature of the ML cost function. At high SNR the global maxima of global ML function is quite distinguishable than other local maxima, whereas at low SNR the local ML function at each node is almost identical. But due to distributed optimization 5 dB gain is achieved in case of RMSE performance at high SNR compared to the corresponding performance of node 1, not being in the network. The global MUSIC shows comparable performance at higher SNR but at the expense of using more snapshots.

It is observed from Figure 5.9 that the distributed PSO performance at node 1 in terms of PR is at par with that of the global ML-PSO, but the average performance of all the nodes is found to be a little less than global one. If the performance of DPSO at node 1 is compared with that of a subarray at node 1 without co-operation and centralized MUSIC with higher number of snapshots, the performance achievement is nearly 5 dB and 10 dB respectively.

Hence it is observed that the distributed PSO-ML algorithm provides better performance in terms of both PR and RMSE compared to that obtained by non distributed case. But it is difficult to maintain the same at all the nodes because the ML cost function is unable to estimate the proper DOA at the edge nodes where they are connected to smaller number of neighbours.

In a network if the average degree of a node is reduced, then the overall performance of both the individual subarrays and the distributed algorithms degrades.

Table 5.3: Comparison of Performances for Different Algorithms in Example 2

RMSE performance in dB	At node 1 using DPSO-ML	Average of all nodes using DPSO-ML	At node 1 using PSO-ML without co-operation
At -7 dB SNR	0.54	8.97	13.85
At -5 dB SNR	-1.70	6.03	10.31
At 0 dB SNR	-5.50	0.71	-1.47
At 5 dB SNR	-8.18	-5.02	-4.48

However for any particular connectivity the performance of the distributed algorithm is better than the individual network produced at each node.

The rate of convergence of the PSO algorithm at 20 dB SNR for a particular experiment is depicted in Figure 5.10. The global MSE converges to the lowest value but at the cost of more communication overhead ($O(h^2)$) more than distributed) compared to other methods. In distributed estimation the convergence is slightly slower at node 1 compared to non-co-operative method. In this case the performance is compromised with the number of communication overhead. It is also shown that if an increased numbers of particles (210 particles) are taken then the convergence becomes faster, but the performance does not increase with that of PSO-ML with 30 numbers of particles.

From the Table 5.3 one can observe that at -7 dB SNR, the networks with more connectivity, the RMSE performance of DPSO-ML algorithm at node 1 is nearly same as that of global PSO-ML (from Table 5.2). However at higher SNR (above -7 dB SNR) the RMSE performance is 2 dB less than the global PSO-ML algorithms performance, but there is a 4 dB improvement than the non-cooperative method at node 1. This amount of achievement is good, because if the number of nodes in a array increased from 7 (network at node 1) to 16 (global network), there is 6 dB gain is achieved. But in case of distributed estimation the same network is giving more than 4 dB gain without increasing the number of nodes. Definitely the proposed method needs little more communication overhead compared to the case for network

at node 1 and much less than the global method.

The performance of ML estimation algorithm depends on the size of the sub-array at each node. This can also be represented in terms of the connectivity of the network. When the degree of connectivity is less than 2, the performance of the proposed approach degrades, thereby supporting the hypothesis of minimum degree of connectivity higher than or equal to 2. Although this constraint reduces the applicability of the proposed approach, still it remains as one of the competitive alternatives for distributed DOA estimation.

5.7 Conclusion

In this chapter distributed ML bearing estimation is proposed using a distributed PSO based method. In this case every node in the network is capable of estimating the DOA by optimizing the local ML function in a diffusive way of cooperation among the neighbours. This algorithm is energy efficient because it needs $O(h^2)$ less communication overhead compared to conventional centralized method. When the nodes estimate the source bearing in a distributed manner the PR is nearly the same as that of global estimation. The RMSE performance is equal at lower SNR; but at higher SNR the distributed estimation has a gain of 5 dB in comparison to the small network when it is not a part of the global network.

The performance of the proposed method depends on the connectivity of the network. For lesser connectivity the performance is also less, but at the same time a reduced amount of communication overhead is required. In any network topology, the overall performance of the distributed network with cooperation is better than the individual network produced at each node with neighbours without cooperation.

In the present case even though each node is capable of estimating the bearing of the sources the RMSE performance of an individual node at high SNR is not at par with global one. It is because of the inherent assumptions is the distributed formulation of the ML cost function revealed in Section 5.3. Hence a problem is to formulate the global likelihood function as a sum of local likelihood function without any assumption. Further investigation can be made on evaluating the performance

of the proposed algorithm with changes in the position of the node. The overall communication overhead of the algorithm depends on the rate of convergence of the PSO for which improved variants of PSO can be chosen.

Chapter 6

Clustering-Based Distributed DOA Estimation Using Diffusion PSO Algorithm in Wireless Sensor Networks

Chapter 6

Clustering-Based Distributed DOA Estimation in Wireless Sensor Networks

This chapter proposes a distributed DOA estimation technique using clustering of sensor nodes and distributed PSO algorithm. The sensor nodes are suitably by clustered to act as random arrays. Each cluster estimates the source bearing by optimizing the ML function locally with cooperation of other clusters. During the estimation process each cluster shares its best information obtained by DPSO with other clusters so that the global estimation is achieved. The performance of the proposed technique has been evaluated through simulation study and is compared with that of obtained by the centralized and decentralized MUSIC algorithms and the distributed in-network algorithm discussed in Chapter 5. The results demonstrate improved performance of the proposed method compared to others. However, the new method exhibits slightly inferior performance compared to the centralized PSO-ML algorithm. Further the proposed method offers low communication overheads compared to other methods.

6.1 Introduction

Chapter 5 introduced a novel distributed DOA estimation techniques using DPSO algorithm. The distributed ML estimation for DOA is formulated. The performance obtained by individual nodes increases with cooperative estimation compared to that of non-cooperative approach. But that approach failed to attain the global

performance at each sensor node due to the assumption taken for the distributed ML formulation for DOA estimation. Since a node shares its observed data to all the neighbours, the snapshots collected at each node are not independent. So it causes a problem for the formulation of global cost function as the sum of all local cost functions. The performance of the algorithm is also influenced by the connectivity of the nodes in the network. It is because the ability of an array to find DOA depends on the number of nodes present in that array. If the nodes connected in a array is less, then their cost function can not provide better performance. In general the connectivity is less in WSNs as the nodes present in the edge of a network always have less connectivity. Therefore in the present chapter clustering based diffusion co-operation is proposed for DOA estimation.

A distributed localization algorithm is proposed here in which the nodes organize themselves into groups known as clusters and collaborate to locate the sources. Each cluster makes an arbitrary array with the nodes present inside the cluster and then estimates the DOA by optimizing a local ML function using diffusive mode of cooperation among the clusters. For this problem the node has to be aware of its location because the array characteristic is dependent on the position of nodes.

Iterative search techniques have been reported in the literature for optimization of ML function (discussed in Section 5.1). Based on the potentiality, the same DPSO algorithm is selected here. In this approach each cluster in the network tries to optimize the local ML function and communicates its best estimate values to its neighbors for local diffusion.

The proposed distributed in-cluster algorithm also reduces the computational burden on individual nodes compared to the distributed in-network algorithm. In case of distributed in-network algorithm, each node optimizes the local ML function using PSO. As a result the memory usage is also more, because PSO with P number of M-dimensional particles that run for K iteration requires $K \times P$ fitness evaluations and memory for $P \times M$ variables for position, velocities, and *pbest*, plus M variables for *gbest*. This may be expensive for sensor nodes. On the other hand in proposed algorithm only the cluster head runs the PSO algorithm for ML optimization instead

of running the PSO in each node. Further, the number of communications among the cluster can be minimized by using block concepts in Chapter 3. Initially the PSO needs more information from the global network to search for the global optimum in their local cost functions. So that each cluster shares the estimated value to the other clusters, up to certain iterations. Then the clusters may share their estimated information after some fixed interval in order to further reduce the communication overheads. In this chapter, the estimation accuracy among the distributed in-cluster, distributed in-network, and centralized algorithm are compared.

6.2 Problem Formulation

Consider a sensor network with N sensor nodes distributed over some geographical region to collect data for source localization. These nodes receive the signal from $M(> N)$ number of narrowband far-field sources from direction angles $\boldsymbol{\theta} = [\theta_1, \theta_2, \ldots, \theta_M]^T$. The same data model is used here as it is given in Section 5.2.

The problem is to estimate the DOA of the sources $\boldsymbol{\theta}$ from the given observation $\{\mathbf{y}_g(i)\}_1^L$ in distributed manner. Now the question arises which of the bearing estimation algorithms can be used to make it distributed. Of course, the best choice is the ML estimator because of its superior statistical property and the distributed MLE for sensor network is discussed in [132].

6.2.1 Maximum Likelihood DOA Estimation

Since the data received at different nodes are spatially uncorrelated, therefore the probability density function for nodes single snap shot is given as

$$p_g(\mathbf{y_g}|\theta) = \frac{1}{\det[\pi R_g(\theta)]} \exp\left\{-(\mathbf{y}_g^{\mathrm{H}} R_g^{-1} \mathbf{y}_g)\right\} \tag{6.1}$$

The joint probability density function (pdf) $p_g(\mathbf{y}(j)|\boldsymbol{\theta})$ for L independent snapshots is defined as:

$$p_g(\mathbf{y}_g|\boldsymbol{\theta}) = \prod_{i=1}^{L} \frac{1}{\det[\pi R_g(\theta)]} \exp\left\{-(\mathbf{y}_g^{\mathrm{H}}(i) R_{\mathbf{y}_g}^{-1} \mathbf{y}_g(i))\right\} \tag{6.2}$$

For bearing estimation, the cost function is a log-likelihood of the form

$$f_g(\boldsymbol{\theta}) = \ln \prod_{i=1}^{L} p_g(\mathbf{y}_g(i)|\boldsymbol{\theta}) = \sum_i \ln p(\mathbf{y}_g(i)|\boldsymbol{\theta}) \tag{6.3}$$

For convenience, the dependency of f on the parameter in the notation is suppressed and let $f(\mathbf{y}_g(i)|\boldsymbol{\theta}) = f(\boldsymbol{\theta})$. The use of simple product of probabilities assumes that the snapshots are independent random events. Further, following [126] the the global ML cost function is given in (5.7) where the sample covariance matrix $\hat{\mathbf{R}}_g$ is defined as

$$\hat{\mathbf{R}}_g = \frac{1}{L} \sum_{i=1}^{L} \mathbf{y}_g(i) \mathbf{y}_g^{\mathrm{H}}(i) \tag{6.4}$$

It is required to obtain a the optimal estimate of $\boldsymbol{\theta}$ that minimizes the global likelihood function

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} f_g(\boldsymbol{\theta}) \tag{6.5}$$

Note that $\mathbf{y}_g(i)$ is to be maintained constant while estimating $\boldsymbol{\theta}$.

6.2.2 Local Bearing Estimation (Distributed In-Network)

In order to decompose the global estimation problem (6.5) into a distributed optimization problem, one has to assume that the cost function has an additive structure [11]. Now the question is how individual node formulates its own cost function in order to estimate sources direction. In Section 5.2, how N number of arbitrary arrays are formed for local bearing estimation at each node is discussed.

While decomposing the global ML function into the sum of local ML function, the approximation used in (5.11) is incorrect. To overcome this approximation, in the next section a novel consider distributed in-clustering strategy is proposed.

6.3 Distributed In-Clustering DOA Estimation

In this section the distributed in-cluster DOA estimation algorithm applicable to sensor network is developed. Let us assume that all the sensor nodes in the SN is divided into N_c number of clusters. Let the set of nodes belonging to jth cluster is

denoted as $\mathcal{N}_j^c$ and the number of nodes in jth cluster is represented as n_j^c. Let the average cluster size be n_c. Note that distributed in-network algorithm can be viewed as a special case of the distributed in-cluster algorithm where each node acts as its own cluster such that $n_j^c = 1$ and $N_c = N$. After clustering of the nodes is formed

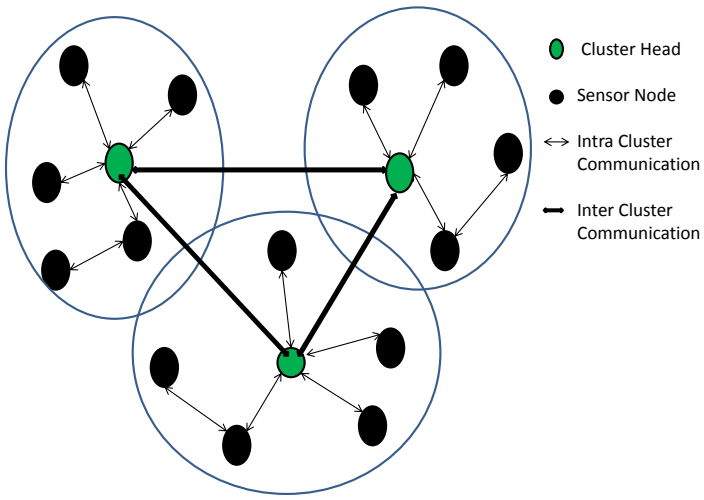


Figure 6.1: Formation of random array at each cluster head in the network

each of them shares the received data to the cluster head by using either single hop or multihop communication mode which is shown in Figure 6.1. Now, the sensor nodes belonging to a cluster constitute an arbitrary array. The data model for jth array formed at cluster j is given as

$$\mathbf{y}_j^c(i) = \mathbf{A}_j^c(\boldsymbol{\theta})\mathbf{s}(i) + \mathbf{v}_j^c(i), \qquad i = 1, 2, \ldots, L \tag{6.6}$$

where $\mathbf{y}_j^c(i)$ is the data vector of an arbitrary sensor array formed at jth cluster; $\mathbf{A}_j^c(\boldsymbol{\theta})$ denotes the steering matrix of that array and $\mathbf{v}_j^c(i)$ represents the unpredicted noise process at the jth cluster.

The joint probability density function (pdf) $p_c(\mathbf{y}_c(i)|\boldsymbol{\theta})$ for L number of independent snapshots is defined as

$$p_j^c(\mathbf{y}_j^c|\boldsymbol{\theta}) = \prod_{i=1}^{L} \frac{1}{\det[\pi R_{\mathbf{y}_j^c}(\theta)]} \exp\left\{-(\mathbf{y}_j^c(i)^H R_{\mathbf{y}_j^c}^{-1} \mathbf{y}_j^c(i))\right\} \tag{6.7}$$

and the cost function of the jth cluster in log-likelihood form is given by

$$f_j^c(\boldsymbol{\theta}) = \ln \prod_{i=1}^{L} p_j^c(\mathbf{y}_c(i)|\boldsymbol{\theta}) = \sum_i \ln p_j^c(\mathbf{y}_j^c(i)|\boldsymbol{\theta}) \tag{6.8}$$

where $\mathbf{y}_j^c(i)$ is the snapshot vector available at cluster j at time i and $p_j^c(\mathbf{y}_j^c(i)|\theta)$ is the likelihood function of observed data for a given θ.

Now the global joint pdf can written in terms of individual clusters pdf for distributed bearing estimation as.

$$p_g(\mathbf{y}_g(i)|\boldsymbol{\theta}) = \prod_{j=1}^{N_c} p_j^c(\mathbf{y}_j^c(i)|\boldsymbol{\theta}) \tag{6.9}$$

In the above equation no approximation has been made as a node shares its data to a particular to which the node belongs to. Further, a node always belongs to a specific cluster only. Therefore the data are clearly uncorrelated among clusters. Taking log-likelihood the cost function has an additive form given as

$$f_g(\boldsymbol{\theta}) = \sum_{j=1}^{N_c} \left\{ \sum_{i=1}^{L} \ln p_j^c(\mathbf{y}(i)|\boldsymbol{\theta}) \right\} \tag{6.10}$$

Using (6.8) the global cost function can be rewritten as the sum of local cost functions of the corresponding clusters as

$$f_g(\boldsymbol{\theta}) = \sum_{j=1}^{N_c} f_j^c(\mathbf{y}_j^c(i)|\boldsymbol{\theta}) \tag{6.11}$$

Further, following [126] the UML estimates of DOA at different cluster is

$$f_j^c(\boldsymbol{\theta}) = \log \left| \mathbf{P}_{\mathbf{A}_j^c} \hat{\mathbf{R}}_j^c \mathbf{P}_{\mathbf{A}_j^c}^H + \frac{\mathbf{P}_{\mathbf{A}_j^c}^{\perp}}{N-M} \mathrm{tr}\left[\mathbf{P}_{\mathbf{A}_j^c}^{\perp} \hat{\mathbf{R}}_j^c \right] \right| \tag{6.12}$$

where tr[$\cdot$] is the trace of the bracketed matrix; $\mathbf{P}_{\mathbf{A}_j^c} = \mathbf{A}_j^c (\mathbf{A}_j^{cH} \mathbf{A}_j^c)^{-1} \mathbf{A}_j^{cH}$ is the projection of matrix $\mathbf{A}_j^c$ and $\mathbf{P}_{\mathbf{A}_j^c}^{\perp} = \mathbf{I} - \mathbf{P}_{\mathbf{A}_j^c}$ is the orthogonal complementary. The sample covariance matrix $\hat{\mathbf{R}}_j^c$ is defined as

$$\hat{\mathbf{R}}_j^c = \frac{1}{L} \sum_{i=1}^{L} \mathbf{y}_j^c(i) \mathbf{y}_j^c(i)^{\mathrm{H}} \tag{6.13}$$

In this case the dependence of $\mathbf{A}_j^c$ on $\boldsymbol{\theta}$ is suppressed for notational simplicity.

Now each cluster has local UML function $f_j^c(\boldsymbol{\theta})$ in (6.12) which is different for different clusters. It is because of the differences in the array response vector for different array formed at different clusters. But the difficulty in ML estimation is its multimodal nature which is mostly overcome in PSO based ML solution [46, 128]. During the evolution of PSO each node can share their estimated angle of arrival to other clusters. Therefore every cluster can diffuse the neighbour's estimated DOA in order to make faster and accurate global estimation. The working principle of distributed PSO has been explained in [105]. Th diffusion PSO, a distributed version of PSO is employed for distributed ML-DOA estimation.

6.3.1 Distributed DOA Estimation using Clusters

In distributed optimization methods using consensus algorithm, the goal of each cluster is to solve the optimization problem in a cooperative manner. Assume that the clusters have high energy and they are communicating with each other by inter cluster path [106]. Thus, the problem of optimization is to minimize the global cost function $f_g(\boldsymbol{\theta})$.

$$f_g(\boldsymbol{\theta}) = \sum_{j=1}^{N_c} f_j^c(\boldsymbol{\theta}) \tag{6.14}$$

subject to $\theta \in \mathtt{R}^M$. Since the ML function is multimodal, its cost functions are not convex, but shares a common global solution (all provides unbiased bearing estimates). In the vicinity i.e., a small region around the solution, all the functions are convex. Therefore it is possible to initialize an optimization procedure within this region.

The beauty of PSO algorithm is that in a multimodal problem it can search the region where the global solution lies. Further it enables the individual clusters to interact with other clusters to make the search process faster and accurate. So it needs a consensus algorithm that serves a basic mechanism for distributing the optimization among the clusters. The primary objective is to minimize the global cost function defined in (6.14).

Considering the nature of the present problem and potentiality of PSO, the

distributed version of the diffusion PSO algorithm is chosen for solving (6.14). In this formulation each cluster runs the PSO algorithm for optimizing its own cost function and simultaneously shares the best solution to other clusters. Therefore the algorithm has the uniqueness of sharing the best results of cluster among other clusters. In particular, each cluster starts with an initial estimate $\boldsymbol{\theta}_j(0)$ and update its estimate at discrete times $i = 1, 2, ...$ Let $\boldsymbol{\theta}_j(i)$ be the bearing vector estimate by jth cluster at time i. While updating, sensor node combines its current estimate $\boldsymbol{\theta}_j(i)$ with the estimate $\boldsymbol{\theta}_k(i)$ received from other clusters l. Hence update equation is given as

$$\boldsymbol{\theta}_j^c(i+1) = \sum_{l=1}^{N_c} a_{jl}(i)\boldsymbol{\theta}_l^c(i) \tag{6.15}$$

where the scalars $a_{jl}(i)$ represents a factor by which the estimates of others clusters $\boldsymbol{\theta}_1(i), \ldots, \boldsymbol{\theta}_{N_c}(i)$ contribute to the lth cluster. These weights are assumed to be same as for static network where the topology as well as clusters remain unchanged with time. In literature [35, 36] different methods have been proposed to calculate these weights according to the network topology.

6.4 Distributed Particle Swarm Optimization

Consider a swarm at each cluster having P particles. The particles are represented by M dimensional positions (because M number of sources are considered) and associated velocity vectors. Let the position and velocity vectors of the l-th particle for j-th cluster are denoted as $\mathbf{x}_{lj}^i = [x_{lj1}^i, x_{lj2}^i, \ldots, x_{ljM}^i]^T$ and $\mathbf{v}_{lj}^i = [v_{lj1}^i, v_{lj2}^i, \ldots, v_{ljM}^i]^T$ respectively. These particles fly through the hyperspace (i.e., $\mathbb{R}^M$) and have two essential information: their own best as well as the global or their neighborhood's best position. Let the *pbest* of the l-th particle at the jth cluster be $\mathbf{p}_{lj}^i = [p_{lj1}^i, p_{lj2}^i, \ldots, p_{ljM}^i]$ and that of *gbest* be $\mathbf{p}_{gl}^i = [p_{gl1}^i, p_{gl2}^i, \ldots, p_{glM}^i]$. In a minimization formulation each particle of the swarm at kth node updates its velocity and position according to

$$\mathbf{v}_{lj}^{i+1} = \omega^i \mathbf{v}_{lj}^i + c_1 \mathbf{r}_1^i \odot (\mathbf{p}_{lj}^i - \mathbf{x}_{lj}^i) + c_2 \mathbf{r}_2^i \odot (\mathbf{p}_{gl}^i - \mathbf{x}_{lj}^i) \tag{6.16}$$

$$\mathbf{x}_{lj}^{i+1} = \mathbf{x}_{lj}^{i} + \mathbf{v}_{lj}^{i+1} \tag{6.17}$$

where $\odot$ denotes the element-wise product, $l = 1, 2, \ldots, P$, $i = 1, 2, \ldots,$ indicates the iteration numbers and $j = 1, 2, \ldots, n_c$ represents the cluster number. The velocity is updated based on the current value scaled by *inertial weight* ω and increased in the direction of *pbest* and *gbest*. The constants c_1 and c_2 are scaling factors that determine the relative pull of the particles' best location *pbest* and global best location *gbest*. These positive constants are sometimes referred to as *cognitive* and *social* parameters respectively [101, 104]. Two independent M-dimensional random vectors r_1 and r_2 consisting of independent random numbers uniformly distributed between 0 and 1 are used to stochastically vary the relative pull of *pbest* and *gbest*. In order to simulate the unpredictable components of the natural swarm behavior, these random elements are introduced in the optimization process [104].

In a distributed DOA estimation, the problem is how do different clusters mutually share their *gbest* information with other clusters. For this purpose, diffusion based distributed PSO (DPSO) algorithm is chosen using similar cooperation strategy described in [32]. In DPSO each node updates its particle's position, velocity and *pbest* using its local objective function $f_j^c(\boldsymbol{\theta})$ and shares its *gbest* vector to all other clusters of the network.

At every cluster a consensus mechanism given in (6.18) is used to fuse the received *gbest* estimates.

$$\mathbf{p}_{gj}^{(i-1)} = \sum_{n=1}^{N_c} a_{jn} \mathbf{p}_{gn}^{(i-1)} \tag{6.18}$$

The combination coefficients $\{a_{jn} \geq 0\}$ are determined from the network topology. The various methods of selecting the weighted coefficients are the metropolis weight, the maximum-degree weight and the nearest neighbor rule. However in the present case the metropolis weight selection rule is used as it is best suited for distributed implementation [35]. It is because these weights preserve the average, are easily computed and provide guaranteed asymptotic average consensus. The metropolis

weight selection procedure is given as

$$a_{kl} = \begin{cases} \frac{1}{\max(n_k, n_l)}, & \text{for } k \neq l, \text{ nodes } k \text{ and } l \text{ are linked} \\ 0, & \text{for } k \neq l, \text{ nodes } k \text{ and } l \text{ are not linked} \\ 1 - \sum_{l \in \mathcal{N}_k / k} a_{kl}, & \text{for } k = l \end{cases} \tag{6.19}$$

where n_k and n_l denote the degree for nodes k and l. But in the present case cluster of nodes are cooperated for distributed DOA estimation. Again it is also assumed that the clusters are fully connected which means all the clusters are connected to each others. Therefore the combiner co-efficients are constant and equal to

$$a_{kl} = 1/N_c \tag{6.20}$$

The fused $\mathbf{p}_{gj}^{(i-1)}$ is used in the local optimization process to estimate the global DOA so that it can rapidly respond to change in its neighborhood. Subsequently the position and velocities of all particles at nodes are simultaneously evaluated by taking diffused *gbest* as the previous *gbest* of the node. The computed global best information is exchanged between all participating nodes for further processing.

6.5 Diffusion PSO for Clustering Based DOA Estimation

The formulation of the DPSO algorithm for clustering based distributed ML optimization to estimate the source DOA's in SN is dealt in this section. Instead of nodes, the cluster heads estimates the DOA. To achieve it every cluster head in the network begins by initializing a population of particles in the search space with random positions constrained between 0 and π in each dimension and random velocities in the range of 0 to π. The M-dimensional position vector of the j-th cluster of l-th particle takes the form $x_{jl} = [\theta_{j1}, \ldots, \theta_{jM}]$, where θ_j represents the DOAs of j-th cluster. A particle position vector is converted to a candidate solution vector in the problem space. The score of the mapped vector evaluated by a likelihood function $f_j^c(\boldsymbol{\theta})$ which is given in (6.12), represents the fitness value of the corresponding particle.

During the evolution of the algorithm in each iteration each cluster updates its velocity and position of all associated particles and then evaluates *gbest* and shares with its neighbour nodes for diffusion operation. The velocity update of (6.16) acts as a key to the entire optimization process. Three components typically contribute to the new velocity. The first part refers to the inertial effect of the movement which is just proportional to the old velocity and has a tendency to guide particle to proceed in the same direction. The inertial weight w is considered critical for the convergence behavior of PSO [129]. A larger w facilitates the search in new area and global exploration while a smaller w helps local exploitation in the current search area. In this study, w is selected so that it decreases during the optimization process, thus the PSO tends to have more global search ability at the beginning of the run while having more local search ability towards the end.

Let the maximum and minimum values of w be $w_{\max}$ and $w_{\min}$ respectively. The value of w in ith iteration is

$$w^i = \begin{cases} w_{\max} - \frac{w_{\max}-w_{\min}}{rK}(i-1), & \text{if } 1 \leq i \leq [rK] \\ w_{\min}, & \text{for } [rK]+1 \leq i \leq K \end{cases} \tag{6.21}$$

where $[rK]$ is the number of iterations with time decreasing inertial weight, $0 < r < 1$ represents ratio, K is the maximum iteration number and $[\cdot]$ is a rounding operator.

The second and third components of the velocity update equation introduces stochastic tendencies to return towards the particle's own as well as the group's best historical positions. The constants c_1 and c_2 are used to bias the particle's search towards the two best locations. Following the common practice reported in the literature [101], $c_1 = c_2 = 2$ have been chosen. Since there is no mechanism for controlling the velocity of a particle, it is necessary to define a maximum velocity to avoid the swarm divergence [131]. The velocity limit which has been applied along each dimension at every node as

$$v_{lj}^i = \begin{cases} V_{\text{MAX}}, & \text{if } v_{lj}^i > V_{\text{MAX}} \\ V_{\text{MIN}}, & \text{if } v_{lj}^i < V_{\text{MIN}} \end{cases} \tag{6.22}$$

where $j = 1, \ldots, n_c$. In this work, V_{MAX} is limited to half of the dynamic range. The

new particle position is calculated using (6.17). The algorithm is stopped when the specified maximum iteration number K is reached. The final global best position p_{kg} is taken as the ML estimates of source DOA. The main steps involved in DPSO-ML algorithms are outlined in Algorithm 4.

Algorithm 4: Main steps of distributed in-cluster algorithm

Setup Problem:
- Define the WSNs with topology.
- Initialize the location of each node.
- Form of clusters using K-means clustering algorithm.
- Choose the cluster head.
- Calculate weighted coefficients for diffusion.
- Set up random array at each cluster by connecting to the nodes present in that cluster.
- Define problem space.
- Define local fitness function at each cluster.
- Select PSO Parameter.

Swarm Initialization at each cluster:
- Random Positions.
- Random velocities.

```
for each iteration do
    for each cluster do
        for each particle do
            Map particle location to solution vector in solution space;
            Evaluate the objective function of current iteration of the node
            according to its local ML function (6.12);
            Update particles best location p_k^i and group best location g_k^i according
            to fitness value;
            Update particles velocity according to (6.16);
            if velocity exceeds the limits then
                Limit particles velocity using (6.22);
            end
            Update particle position using (6.17);
            if particles position out of solution boundaries then
                Clip or adjust particles position;
            end
        end
        Share its gbest to all neighbour nodes;
    end
    Perform the local diffusion according to (6.18);
    Check termination criterion;
end
```

6.5.1 Block distributed in-cluster algorithm

Each cluster runs the PSO algorithm locally and shares their best estimated value in each iteration with other clusters so that they may achieve the global performance. In this way the DPSO also converges faster than the PSO algorithm. For this purpose initially the algorithm needs more co-operations to search the region of the global optimum in the local cost function. After finding the global optimum region, the PSO then searches the actual optimum value in the vicinity of it so that the sharing of their best estimated data may not be required in each iteration. Therefore a block distributed in-cluster algorithm (Algorithm 2) is proposed where the individual cluster shares its best estimate up to K_B iterations for local diffusion. After K_B iterations, the clusters share their information in the interval of B iterations until the stopping criterion is satisfied. The main steps involved in this algorithm is presented as Algorithm 5.

Algorithm 5: Main steps of block distributed in-cluster algorithm

```
Setup Problem:
As in Algorithm 1
Swarm Initialization at each cluster:
As in Algorithm 1
for each iteration do
    for each cluster do
        for each particle do
            These steps are same as in Algorithm 1
        end
        if iteration >= K_B & mod(iteration, B)=0 then
            Share its gbest to all neighbour nodes;
        end
    end
    Carry out the local diffusion operation according to (6.18);
    Check termination criterion;
end
```

6.5.2 Communication Overhead of the Proposed Algorithms

In a sensor network, the communication of information consumes most of the energy [5]. Therefore it is of great importance to minimize the communication requirements

among nodes. In the present case an upper bound of the average number of messages transmitted by a node and the overall communication for the network are determined for the proposed algorithms and are also compared with those of existing algorithms.

A node communicates directly only with other sensor nodes that are within a small distance known as transmission radius (T_r). In order to avoid communication between sensors within each other's communication range, the sensor nodes form a multihop network. In multihop communication the distance between any two nodes is defined as the minimum number of hops between them. Let h_N be the maximum distance between any node in the network and the fusion center (FC). Similarly, let h_C be the maximum distance between any node in the cluster and the cluster head (CH). Let d be the average degree of a node in the network where the degree of a node is the number of nodes that serve as neighbours to it.

The proposed algorithm involves two phases: a data sharing phase and the distributed estimation phase. In fact the clustering is also a part of this algorithm, but the amount of communication required for clustering are not considering here. It is because the position based K-Means clustering is used in the case where the nodes need not have to communicate anything. The clustering is carried out by the FC and the cluster information is sent to all the nodes of the network. Thus excluding the clustering phase, the overall communication overheads required are the sum of the number of messages in two aforesaid phases.

In the data sharing phase, each node shares L complex data samples to its CH and the average number of messages unicast from nodes to the CH node can be calculated. In the present case aggregation of message is not used and it provides worst case value. Under these conditions the number of messages transmitted becomes $O(dh_c^3)$ [109]. The overall communication overheads for all clusters of the network during data sharing phase is given by

$$\mathcal{M}_{sharing} = O(N_c L d h_c^3) \tag{6.23}$$

During the estimation phase, at each iteration every cluster in the network shares M ($M << L$) real data samples with all N_c clusters. Let us assume that the in

the worst case, CH's are present at the center. If one cluster sends data to another neighbouring cluster, it needs at most $2h_c$ hops (h_c hops for the cluster sending the data and another h_c hops for the cluster sending to a node). Therefore, total number of messages transmitted by all the clusters in a cycle is $2h_cN_cM$ (assuming that cluster are communicating is $2h_c$ hops). Let I be the number of cycles required to attain the steady state value in the network. Then, the total communication cost for the estimation phase is $\mathcal{M}_{est} = O(N_cMI)$.

Therefore the overall communication overhead for the network using Algorithm 1 is $O(N_cLdh_c^3) + O(2h_cN_cMI)$. Since the number of clusters are always less than the cluster size, the number of hops between clusters can be minimized and sharing the information between the clusters can further be reduced by using block concept (Algorithm 2). Again in estimation phase the number messages are also depends on the number of cycles required to get steady state value. The convergence speed of PSO can be improved if the PSO is initialized by AP-AML [46] or MUSIC algorithm. Therefore the total number of messages needed for Algorithm 1 is

$$\mathcal{M}_{alg1} = O(N_cLdh_c^3) \tag{6.24}$$

In case of distributed in-network algorithm each node shares its information in every cycle. Hence the number of messages required for distributed in-network algorithm (Algorithm 3) is given as

$$\mathcal{M}_{alg3} = O(NMId) \tag{6.25}$$

In centralized estimation, each node in the network sends L number of data samples to the FC. In the worst case scenario where message aggregation is not used, the message unicast from a leaf node in the spanning tree, is forwarded h_N times till it reaches the FC. Therefore, the overall communication overhead for the network in the centralized case is

$$\mathcal{M}_{cent} = O(NLdh_N^3) \tag{6.26}$$

In general, a more number of snapshots are required for the centralized MUSIC

Table 6.1: Comparison of Communication Overheads for Different Algorithms

Algorithm	Distributed in-cluster algorithm	Distributed in-network algorithm	Centralized PSO-ML	Centralized MUSIC
Overall communication overhead	$O(N_c L d h_c^3)$	$O(NMId)$	$O(NLdh_N^3)$	$O(NL'dh_N^3)$

algorithm say $L' > L$. The communication overhead for this algorithm is computed as

$$\mathcal{M}_{MUSIC} = O(NL'dh_N^3) \tag{6.27}$$

Thus the centralized MUSIC algorithm needs more communication than the centralized PSO-ML algorithm. The overall comparison between all the algorithms are given in Table 6.1. The value of h_c is small because it may approximately equal to $\lceil h_N/N_C \rceil$. Therefore the saving in overall communication overhead is $O(NMI/N_c L h_c^3)$ over distributed in-network algorithm. Especially for large network the ratio N/N_c is large, hence the saving of communication overhead is large. Already in Section 5.3.1 it is shown that the distributed in-network algorithm needs less communication overhead compared to other centralized algorithms. Therefore it may be conclude that the proposed distributed in-cluster algorithm uses less communication overhead compared to others.

6.6 Simulation Results and Discussions

In this section an illustration is given to show the advantages of distributed in-clustering algorithm over distributed in-network and also to examine the performance of the proposed algorithm against decentralized MUSIC, centralized MUSIC and PSO-ML algorithms. The performances of these estimators are evaluated in term of the RMSE and the PR. These are defined in Sectiion 5.6 All the algorithms have been simulated using MATLAB for 100 Monte Carlo (MC) for each point of the plot on an Intel Core™ Duo, 2.8 GHz, 4GB RAM PC. The PSO parameters are chosen for the experiments are: the constants $c_1 = c_2 = 2$, maximum number of iteration $K = 100$, maximum velocity limit $V_{max} = 0.5 * \pi$, the range of time varying

inertial weight is $w_{max} = 0.9, w_{min} = 0.4$ and $r = 0.5$. These values can be tuned for better optimization performance. The PSO algorithm starts with random initialization at diffusion center for centralized estimation and at cluster heads for distributed in-clustering algorithm respectively. The algorithm is stopped when maximum number of iteration is reached. Since the main objective is to compare the performance of the distributed algorithm with the centralized one no other algorithms for solving ML like the GA, alternating projection (AP) or Newton techniques are used for comparison purpose. Further, the PSO-ML provides superior performance over all other algorithms dealt before [46].

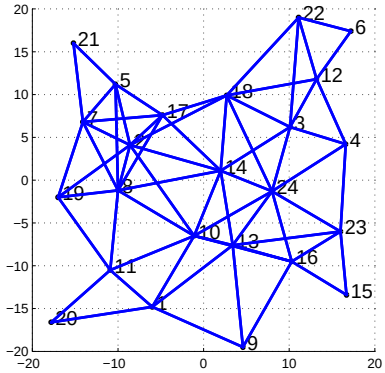


Figure 6.2: Network Topology used in Example 1

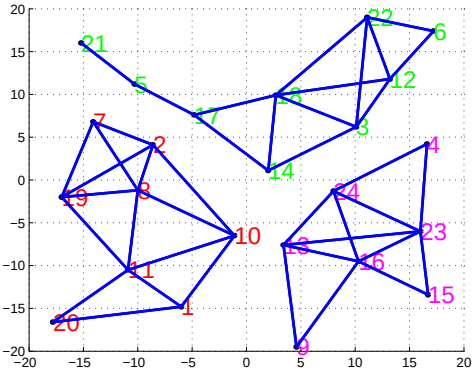


Figure 6.3: Clusters

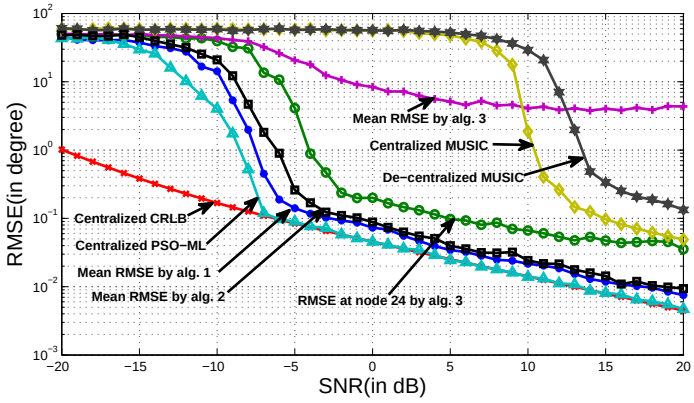


Figure 6.4: RMSE vrs. SNR plots in DOA estimation by centralized PSO-ML, centralized MUSIC, decentralized MUSIC, proposed distributed in-clustering Algorithms 1 and 2, distributed in-network (Algorithm 3) and theoretical centralized CRLB.

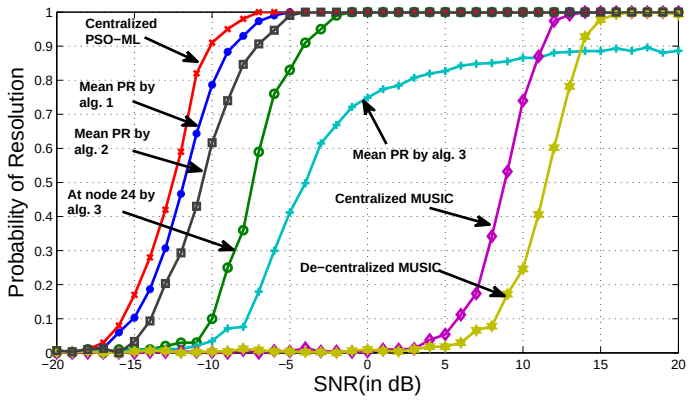


Figure 6.5: PR vrs. SNR plots in DOA estimation by centralized PSO-ML, centralized and decentralized MUSIC algorithms, proposed distributed in-clustering Algorithms 1 and 2 and distributed in-network Algorithm 3

6.6.1 Example 1

Consider a sensor network having $N = 24$ (number of identical sensor nodes) distributed randomly in an area of 20×20. It is assumed that the sensor nodes are aware of their position co-ordinates. Then the topology of the network is developed on the basis of transmission radius of the nodes which is shown in Figure 6.2. Two nodes are connected if they all lie in their transmission range. Then the K-means clustering algorithm is used to make the networks into 3 numbers of non-overlapping clusters. The clusters are shown in Figure 6.3. The main objective of this proposed

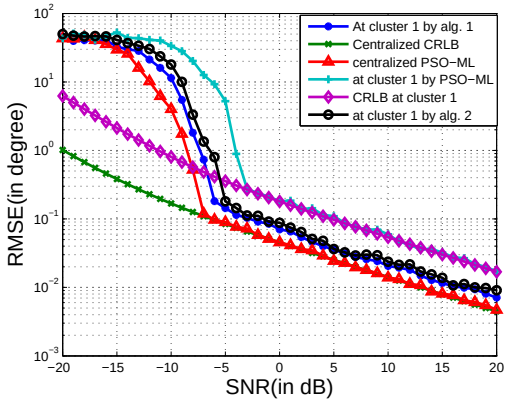


Figure 6.6: RMSE vrs. SNR plots in DOA estimation by centralized PSO-ML, at cluster 1 by using Algorithms 1 and 2, at cluster 1 by PSO-ML, theoretical CRLB at cluster 1 and theoretical centralized CRLB.

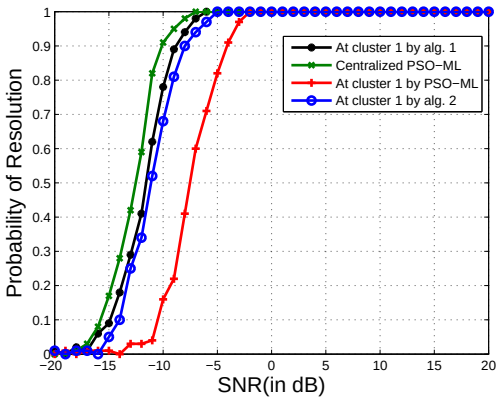


Figure 6.7: PR vrs. SNR plots in DOA estimation by centralized PSO-ML, at cluster 1 by using Algorithms 1 and 2 and at cluster 1 by PSO-ML

work is to make the algorithm distributed, and hence effort has not been given for efficient clustering. In Figure 6.3 the nodes belongs to cluster 1, 2 and 3 are shown. The nodes 8, 18 and 16 serve as the respective cluster heads (CH). The choice of CH is such that the cluster head distance should be minimized. This depends upon the clustering algorithm used. But in the present case, the central node of cluster chose as the CH. It may require more number of multi-hop communications to interact with other CHs if proper CH is not chosen.

It is assumed that two uncorrelated equal signals power impinge on the distributed network at 130° and 138°. The number of snapshots taken are $L = 20$ and 60 are taken for PSO-ML and MUSIC algorithms respectively. Higher number

of snapshots for MUSIC compared to that of PSO-ML is used because minimum number of snapshots required for MUSIC to get satisfactory performance is equal to twice the number of sensors where as the ML estimator can provide similar performance even by using fewer snapshots. Further, in SN data communication is reduced to make the network energy efficient. The functions $f_g(\boldsymbol{\theta})$ and $f_k^c(\boldsymbol{\theta})$ are computed for global and local estimation using the PSO. Figure 6.4 shows the estimation RMSE obtained by using distributed in-clustering Algorithm 1, centralized PSO-ML, Centralized MUSIC, decentralized MUSIC and distributed in-network algorithm and block distributed in-clustering algorithm. The performance of all those algorithms is compared with the corresponding Cramer-Rao lower bound (CRLB) which is based on stochastic signal assumption [126]. Figure 6.5 depicts the probability of resolution obtained using these methods. The SNR is varied from -20 dB to 20 dB with step size of 1 dB.

It is evident from Figures 6.4 and 6.5, that the centralized PSO-ML provides better performance over all other algorithms by demonstrating lower estimation RMSE and higher probability of resolution. The RMSE is asymptotically attains the CRLB at -7 dB SNR. The accurate DOA estimation is because of use of the ML function (which is statistically optimal) and the PSO algorithm (which is robust and reliable for global optimization). The performance of the proposed in-clustering Algorithm 1 and 2 are closer to global performance. The performances (both RMSE and PR) are not exactly matching with that of global network's performance, but it is much better than the distributed in-network Algorithm 3. The average performance of Algorithm 3 is inferior to that of Algorithms 1 and 2. It is know that the DOA estimation performance of an array decreases with the decrease in number of sensors in the network. In case of in-network distributed algorithm, few nodes present in the edge of a network constitutes random array with their immediate neighbours having only 3 sensor nodes including it. In such situation the ML-DPSO algorithm fails to estimate two sources DOA using 3 nodes. Though in distributed scenario the performance of the individual node increases, yet it could not achieve the global estimation performance due to their nature of ML cost function, $f_k(\boldsymbol{\theta})$. Thus the average per-

formance is degraded due to poorer edge node's performance. On the other hand the nodes which reside in the interior of the network have better connectivity, and hence they yield better performance. In Figure 6.4 and 6.5 it is observed that the best performance is obtained at node 24 using Algorithm 3. Thus it may be concluded that the proposed in-clustering algorithm is provides better performance compared to all other distributed algorithms. At lower SNRs this performance is better than the MUSIC and the decentralized MUSIC algorithm. These two algorithms show comparable performance at high SNRs.

In order to show the performance improvement of distributed in-clustering estimation over non-cooperating estimation, consider Cluster 1 of the network estimates the same sources DOA by using PSO-ML without any cooperation with other clusters. The performances of Cluster 1 using PSO-ML and Algorithms 1 and 2 are plotted in Figures 6.6 and 6.7. Since the objective of distributed algorithm is to achieve the centralized performance, the performance of clust1 is compared with the centralized network performances. The optimum performance of an array is the CRLB. In case of no-cooperative estimation, the PSO-ML attains the CRLB asymptotically at -3 dB. But the same network gives better performance than the optimum theoretical value, if the estimation is co-operative. It is because in distributed estimation by using proper consensus algorithm for a fully connected network, each agent tries to achieve the global performance. In case of block in-clustering algorithm, the performance is slightly inferior to that of Algorithm 1. But the block method saves communication overheads. So when energy is major constraint for designing SN, the block distributed in-clustering algorithm is preferable with slight compromise with the performance.

In Table 6.2 the RMSE performance of proposed Algorithms 1 and 2 at cluster 1 are compared with centralized PSO-ML, centralized and decentralized MUSIC algorithms and at cluster 1 using PSO-ML without cooperation at SNRs -5 dB, 0 dB, 5 dB, 10 dB and 15 dB. From the table it is observed that the proposed method RMSE performance is much better than (around 6 dB) the non-cooperative estimation techniques when the SNR is less than or equal to -5 dB. At high SNR (> 0 dB),

Table 6.2: Comparison of Performances for Different Algorithms in Example 1 at Different SNRs

RMSE performance in dB	At Cluster 1 using Algorithm 1	At Cluster 1 using Algorithm 1	At Cluster 1 using PSO-ML without cooperation	Centralized PSO-ML	Decentralized MUSIC	Centralized MUSIC
At -5 dB SNR	-8.42	-7.44	7.18	-10.47	17.58	17.64
At 0 dB SNR	-11.52	-10.64	-7.45	-13.45	17.55	17.47
At 5 dB SNR	-14.47	-14.28	-9.61	-16.61	17.51	16.82
At 15 dB SNR	-19.35	-18.79	-14.92	-20.91	-4.73	-10.11

the gain of approximately 4.5 dB is achieved over non-cooperative estimates which is less than only 2 dB compared to that of centralized PSO-ML. If the performance of centralized PSO-ML is compared with that of PSO-ML performance at cluster1, then there will be 6 dB better RMSE gain after -3 dB SNR. For this the number of nodes is increased from 8 to 24 (3 times) and this the number of communication overheads increased in the order of $O(n_c^4) = O(81)$. If the DOA is estimated by using proposed algorithm, then nearly same number of communication required as in case of cluster 1 without cooperation (because the communication overhead during estimation technique are neglected) and giving $4 \sim 5$ dB of gain. The MUSIC and decentralized MUSIC algorithms are giving comparable performance at higher SNRs (¿15 dB). In this example there are 15 dB and 10 dB gain of Algorithm 1 over centralized MUSIC and decentralized MUSIC algorithms respectively. It is also noted that the MUSIC and decentralized MUSIC algorithm uses 60 snapshots compared to 20 snapshots in case of PSO-ML algorithm. The algorithm 2 provides little less performance than Algorithm 1 over all SNRs. It may conclude that for lower SNR the proposed method archives best performance with least communication over head compared to all the existing algorithms. Again at high SNR it has been achieved comparative results with centralized PSO-ML. But in practice the SNR is work-

ing at low SNR. So this proposed algorithm is a better choice for distributed DOA estimation.

6.6.2 Example 2

In the second example, the proposed technique is examined in presence of two closely spaced sources by using a large sensor array of $N = 40$ nodes. In WSNs two nodes are connected if they lie in the transmission radius of $T_r = 10$. From the topology shown in Figure 6.8 the average node degree $d = 6$. This value can be reduced by minimizing communication link between nodes (i.e. by using multi hop network) because performance of the algorithm does not depend on connectivity of the nodes unlike distributed in-network algorithm. The clustering operation is carried out by unsupervised K-means algorithm. In this case the whole network in sub-groups into 4 clusters having 11, 10, 9 and 10 cluster nodes (the average cluster size is 10) respectively. The clusters are shown in Figure 6.9. The central nodes 2, 22, 17 and 39 are chosen CH's of cluster 1, 2, 3, and 4 respectively.

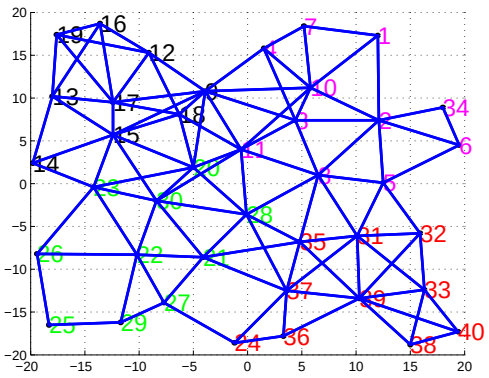


Figure 6.8: Network topology used in Example 2

Table 6.3 at 5 dB of SNR and after 100 iterations.

The distributed SN receives signals from two sources present at DOAs 80° and 82° relative to x-axis. The number of snapshots is 20. In Figure 6.10, the RMSE values obtained by global network using PSO-ML, small network at cluster 1 without any co-operation by PSO-ML and distributed in cluster Algorithm 1 are plotted and compared with the theoretical CRLB. The probability of resolution for the

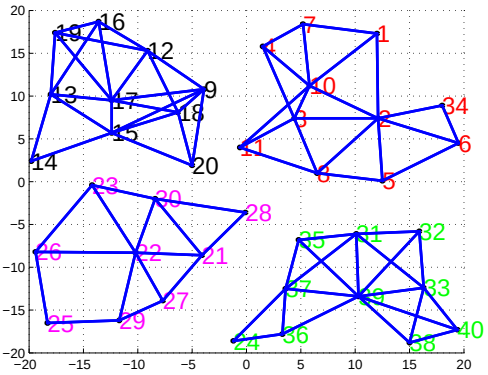


Figure 6.9: The network divided into 4 numbers of clusters

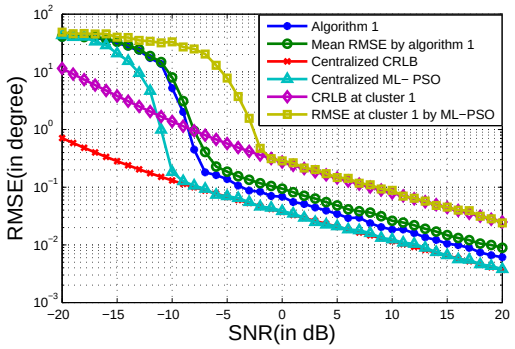


Figure 6.10: RMSE vrs. SNR plots in DOA estimation by PSO-ML, distributed in-clustering Algorithm 1, and theoretical centralized CRLB.

same methods are shown in Figure 6.11. The results of MUSIC and decentralized MUSIC algorithms are not plotted here, because the MUSIC is able to resolve two sources at SNRs higher than 20 dB. As evidents from Figures 6.10 and 6.11 the distributed in-cluster algorithm at cluster 1 outperforms the individual cluster without co-operation. But in this distributed estimation the centralized networks performance is not achieved. It is because of the nature of likelihood function. However, if the numbers of sensors are more, then the minima of the negative likelihood function are clearly distinguishable even at lower SNRs, but it provides better performance by involving high communication overhead. The performance comparison in terms of RMSE is listed in

The results of this table demonstrate that due to distributed estimation, nearly 6 dB gain in RMSE is achieved over non co-operative estimation. The distributed

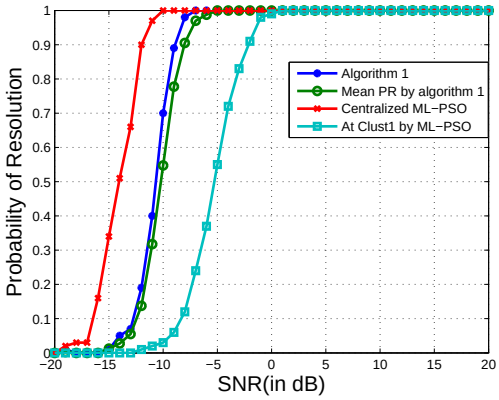


Figure 6.11: PR vrs. SNR plots in DOA estimation by centralized PSO-ML, proposed distributed in-clustering Algorithms 1 and 2 and distributed in-network Algorithm 3

Table 6.3: Comparison of Performances for Different Algorithms

Algorithm	At cluster 1 using algorithm 1	At cluster 1 using PSO-ML without co-operation	Centralized PSO-ML
RMSE performance in dB	-14.63	-8.32	-16.58
Overall communication overhead	$O(9.6 \times 10^3)$	$O(9.6 \times 10^3)$	$O(6 \times 10^5)$

RMSE is 2 dB less than the centralized one. But at the same time substantial saving $O(100)$ of communication overhead is achieved.

6.7 Conclusion

In this chapter a distributed *in-cluster* ML bearing estimation in a sensor network using DPSO algorithm is proposed. In the network the nodes form clusters to estimate the DOA by optimizing the ML function locally in diffusion mode of cooperation among clusters using distributed PSO algorithm. This approach is independent of the connectivity of sensor nodes unlike distributed in-network algorithm. When the clusters are formed in the network and when it estimates the source bearing in distributed manner, its RMSE performance at lower SNR is nearly the same as that computed centrally. However at high SNR the distributed estimation provides 6 dB gain compared to that achieved by small network.

Chapter 7

Conclusion and Future Work

Chapter 7

Conclusion and Future Work

In this chapter the overall contributions of the thesis are reported. Future research problems are also outlined for further investigation on the same/related topics.

7.1 Conclusions

Block implementation of two distributed LMS algorithms i.e. the BDLMS and BILMS algorithms have been proposed to minimize the amount of communication overheads present in conventional distributed LMS. Performance analysis of these two algorithms has been carried out using the weighted energy conservation approach. Both the theoretical and simulation results have demonstrated that the performance of the new algorithm is similar to that of the conventional distributed LMS algorithm. The convergence delay and communication cost of the new algorithms have been investigated and found to be reduced by L (block size) times compared to the conventional distributed LMS algorithms. This not only reduces the communication bandwidth but also the power consumption associated with the transmission and reception of messages across the resource constraint nodes in WSNs.

In order to reduce the effects of impulsive noise on a WSNs, distributed algorithms using robust cost functions such as Huber's cost function, error saturation nonlinearity, and Wilcoxon norm have been proposed. It has been seen from the study that robust algorithm based on Huber's cost function converges slower compared to one based on error saturation nonlinearity. Based on above findings, the SNILMS and SNDLMS algorithms have been developed. The steady-state perfor-

mance of the SNILMS algorithm in presence of contaminated Gaussian impulsive noise is investigated using the spatial energy conservation method. It has been observed through the experimental and theoretical studies that these algorithms are robust. Robust DLMS algorithm based on Wilcoxon norm has also been found to be robust against impulsive noise. The simulation results have shown that the SNDLMS algorithm is a better candidate for robust estimation in comparison to the WNDLMS algorithm.

A novel distributed ML bearing estimation for WSNs have been proposed based on DPSO algorithm. Each node in the network is capable of estimating the DOA by optimizing the local ML function in a diffusive way of cooperation among the neighbours. This algorithm needs $O(h^2)$ less communication overhead compared to that of the conventional centralized method and hence energy efficient. The performance the algorithm in terms of PR has been found to be nearly the same as that of global estimation algorithm. Comparing the RMSE values of the proposed algorithm with the global estimation algorithm, it has been found to be same at low SNR but less at high SNR. However, the RMSE performance of a small network with cooperation is better than without cooperation.

In order to make the distributed bearing estimation algorithm independent of node connectivity, a distributed in-cluster bearing estimation algorithm has been proposed. In this approach the nodes are arranged in clusters and each cluster optimizes the ML function locally by using diffusion mode of cooperation and the PSO algorithm. This distributed in-cluster approach is independent of the connectivity of sensor nodes and provides better performance in terms of PR, RMSE and communication overheads compared to distributed in-network algorithm. Further the block concept has been used along with the diffusion cooperation strategy among the clusters to reduce the communication overheads.

7.2 Contributions Achieved

The contributions achieved in this thesis are listed below.

- Developed two new block distributed algorithms namely the BDLMS and BILMS to reduce the delay and communication overheads present in the conventional distributed algorithms.
- Distributed robust adaptive algorithms namely SNILMS, SNDLMS and WNDLMS based on error saturation nonlinearity and Wilcoxon norm have been proposed. Transient analysis of SNILMS algorithm in presence of impulsive noise has been carried out.
- Developed a distributed ML-DOA estimation technique for WSNs using DPSO algorithm.
- Developed a distributed *in-cluster* ML-DOA estimation algorithm for WSNs using DPSO algorithm in order to make the distributed algorithm more energy efficient and independent of node connectivity.

7.3 Suggestions for Future Work

- The results of the theoretical study has been supported by those obtained in simulation study using MATLAB. It will be of great interest, if the proposed algorithms are simulated using network simulators like NS3, Qualnet to assess the real time performance.
- In this thesis, the error saturation nonlinearity LMS algorithm has been analysed in presence of Gaussian inputs. This investigation can further be extended to other family of LMS algorithms like the LMF, sign error LMS algorithm etc.
- Outliers present both in the desired and input data can be studied using the robust algorithms. The analysis carried out in this thesis may be suitably changed for such cases in accordance with the distributed cooperative scheme.

- Further investigation is required to evaluate the performance of the proposed algorithms when the nodes change their positions.
- The overall communication overhead of the distributed ML-DOA estimation algorithm depends on the rate of convergence of the PSO. Hence improved variants of PSO can be chosen and applied to achieve lower communication overheads and power consumption.

Bibliography

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, no. 8, pp. 102–114, Aug. 2002.

[2] D. Puccinelli and M. Haenggi, "Wireless sensor networks: applications and challenges of ubiquitous sensing," *IEEE Circuits and Systems Magazine*, vol. 5, no. 3, pp. 19 – 31, 2005.

[3] A. Hac, *Wireless Sensor Network Designs*. John Wiley and Sons, 2003.

[4] C. Raghavendra, K. M. Sivalingam, and T. Znati, *Wireless Sensor Networks*. Springer, 2006.

[5] D. Estrin and M. Srivastav, "Instrumenting the world with wireless sensor networks," in *IEEE International conference on Acoustics, Speech, Signal Processing (ICASSP'01)*, May 2001, pp. 2033–2036.

[6] C.-Y. Chong and S. Kumar, "Sensor networks: evolution, opportunities, and challenges," *Proceedings of the IEEE*, vol. 91, no. 8, pp. 1247 – 1256, aug. 2003.

[7] Y. Li and M. T. E. Thai, *Wireless sensor networks and applications*. Springer, 2008.

[8] E. Serpedin, H. Li, A. Dogandic, H. Dai, and P. Cotae, "Distributed signal processing techniques for wireless sensor networks," *EURASIP Journal on Applied Signal Processing*, vol. 2008, pp. 2–12, 2008.

[9] G. Sharma and R. Mazumdar, "A case for hybrid sensor networks," *Networking, IEEE/ACM Transactions on*, vol. 16, no. 5, pp. 1121–1132, Oct. 2008.

[10] V. Lesser, C. L. Ortiz, and M. Tambe, *Distributed Sensor Networks: A Multiagent Perspective*. Kluwer, 2003.

[11] D. Blatt, A. Hero, and H. Gauchman, "A convergent incremental gradient method with constant step size," *SIAM Journal on Optimization*, vol. 18, pp. 29–51, Feb. 2007.

[12] B. Johansson, T. Keviczky, M. Johansson, and K. H. Johansson, "Subgradient methods and consensus algorithms for solving convex optimization problems," in *47th*

IEEE Conference on Decision and Control, 2008. CDC 2008., Cancun, Dec. 1994, pp. 4185–4190.

[13] B. Johansson, A. Speranzon, M. Johansson, and K. H. Johansson, "Technical communique: On decentralized negotiation of optimal consensus," *Automatica*, vol. 44, pp. 1175–1179, April 2008.

[14] D. Blatt and A. O. Hero, "Energy-based sensor network source localization via projection onto convex sets," *IEEE Transactions on Signal Processing*, vol. 54, no. 9, pp. 3614–3619, 2006.

[15] S. M. Williams, P. L. Schmidt, and K. D. Frampton, "Distributed source localization in a wireless sensor network," *Journal of the Acoustical Society of America*, vol. 117, no. 4, p. 2610, 2005.

[16] W. Qiu and E. Skafidas, "Distributed source localization based on TOA measurements in wireless sensor networks," *Research Letters in Electronics*, vol. 2009, 2009.

[17] R. L. G. Cavalcante, I. Yamada, and B. Mulgrew, "An adaptive projected subgradient approach to learning in diffusion networks," *IEEE Transactions on Signal Processing*, vol. 57, pp. 2762–2774, July 2009.

[18] C. G. Lopes and A. H. Sayed, "Incremental adaptive strategies over distributed networks," *IEEE Transactions on Signal Processing*, vol. 55, no. 8, pp. 4064–4077, Aug. 2007.

[19] F. S. Cattivelli and A. H. Sayed, "Diffusion LMS strategies for distributed estimation," *IEEE Transactions on Signal Processing*, vol. 58, no. 3, pp. 1035 –1048, March 2010.

[20] F. Cattivelli and A. Sayed, "Diffusion strategies for distributed Kalman filtering and smoothing," *IEEE Transactions on Automatic Control,*, vol. 55, no. 9, pp. 2069 –2084, sept. 2010.

[21] F. S. Cattivelli, C. G. Lopes, and A. H. Sayed, "Diffusion recursive least-squares for distributed estimation over adaptive networks," *IEEE Transactions on Signal Processing*, vol. 56, no. 5, pp. 1865–1877, May 2008.

[22] A. Nedic and A. Ozdaglar, "Cooperative distributed multi-agent optimization," in *Convex Optimization in Signal Processing and Communications*, Y. Eldar and D. Palomar, Eds. Cambridge University Press, 2008, pp. 340–386.

[23] ——, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, Jan. 2009.

[24] B. Johansson, C. Carretti, and M. Johansson, "On distributed optimization using peer-to-peer communications in wireless sensor networks," in *5th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, 2008. SECON 08.*, June 2008, pp. 497–505.

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

[26] A. Jadbabaie, J. Lin, and A. S. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *IEEE Trans. on Autom. Control*, vol. 48, no. 6, pp. 988–1001, June 2003.

[27] A. H. Sayed, *Fundamentals of Adaptive Filtering*. John Wiley and Sons. Inc. Publication, 2003.

[28] S. M. Kay, *Fundamentals of statistical signal processing: Estimation Theory (Vol. 1)*. Prentice-Hall Signal Processing Series, 2009.

[29] I. Schizas, G. Mateos, and G. Giannakis, "Distributed lms for consensus-based in-network adaptive processing," *IEEE Transactions on Signal Processing*, vol. 57, no. 6, pp. 2365 –2382, june 2009.

[30] M. Rabbat and R. Nowak, "Decentralized source localization and tracking [wireless sensor networks]," in *IEEE International Conference on Acoustics, Speech, Signal Processing (ICASSP'04)*, vol. 3, May 2004, pp. iii–921–4.

[31] A. Nedic and D. P. Bertsekas, "Incremental subgradient methods for nondifferentiable optimization," *SIAM Journal on Optimization*, vol. 12, no. 1, pp. 109–138, 2001.

[32] C. G. Lopes and A. H. Sayed, "Diffusion least-mean squares over adaptive networks: Formulation and performance analysis," *IEEE Transactions on Signal Processing*, vol. 56, no. 7, pp. 3122–3136, July 2008.

[33] ——, "Diffusion adaptive networks with changing topologies," in *IEEE International conference on Acoustics, Speech, Signal Processing (ICASSP'08)*, Las Vegas, NV, April 2008, pp. 3285–3288.

[34] N. Takahashi, I. Yamada, and A. H. Sayed, "Diffusion adaptive networks with changing topologies," in *IEEE International conference on Acoustics, Speech, Signal Processing (ICASSP'09)*, April 2009, pp. 2845–2849.

[35] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *Proceedings of 4th International Symposium on Information Processing in Sensor Networks*, Loss Angles, CA, April 2005, pp. 63–70.

[36] S. Boyd, L. Xiao, and S. Lall, "A space time diffusion scheme for peer-to-peer least-square estimation," in *Proceedings of 5th International Symposium Information Processing in Sensor Networks*, Nashvile, TN, April 2006.

[37] F. S. Cattivelli and A. H. Sayed, "Distributed detection over adaptive networks using diffusion adaptation," *IEEE Transactions on Signal Processing*, vol. 59, no. 5, pp. 1917–1932, May 2011.

[38] Z. K. andEdmund M. Yeh, "Distributed energy management algorithm for large-scale wireless sensor networks," in *Proc. of the 8th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'2007)*, Montreal, Quebec, Canada, Sep. 2007, pp. 209–218.

[39] G. Anastasi, M. Conti, M. D. Francesco, and A. Passarella, "Energy conservation in wireless sensor networks: a survey," *Ad Hoc Networks*, vol. 7, pp. 537–568, May 2009.

[40] S. Marano, V. Matta, and P. Willett, "Distributed estimation in large wireless sensor networks via a locally optimum approach," *IEEE Transactions on Signal Processing*, vol. 56, no. 2, pp. 748 – 756, Feb. 2008.

[41] M. Rabbat and R. Nowak, "Distributed optimization in sensor networks," in *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks*, April 2004, pp. 20–27.

[42] N. J. Bershad, "On error saturation nonlinearities for LMS adaptation in impulsive noise," *IEEE Transactions on Signal Processing*, vol. 56, no. 9, pp. 4526–4530, Sep. 2008.

[43] ——, "On error saturation nonlinearities for LMS adaptation," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 36, no. 4, pp. 440–452, April 1988.

[44] B. Majhi, G. Panda, and B. Mulgrew, "Robust identification using new Wilcoxon least mean square algorithm," *IEEE Electronics Letter*, vol. 45, no. 6, pp. 334–335, March 2009.

[45] M. Li and Y. Lu, "A refined genetic algorithm for accurate and reliable DOA estimation with a sensor array," *Wirel. Pers. Commun.*, vol. 43, no. 2, pp. 533–547, 2007.

[46] ——, "Maximum likelihood DOA estimation in unknown colored noise fields," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 44, no. 3, pp. 1079–1090, July 2008.

[47] M. Wax and T. Kailath, "Decentralized processing in sensor array," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 33, pp. 1123–1129, Oct. 1985.

[48] T. Söderström and P. Stoica, "Statistical analysis of decentralized MUSIC," *Circuits Systems Signal Processing*, vol. 11, no. 4, pp. 443–454, 1992.

[49] R. L. Moses, O. L. Moses, D. Krishnamurthy, and R. Patterson, "A self-localization method for wireless sensor networks," *EURASIP Journal on Applied Signal Processing*, vol. 4, pp. 348–358, 2002.

[50] M. Abedin and A. Sanagavarapu, "Localization of near-field radiating sources with an arbitrary antenna array," in *Antennas and Propagation Society International Symposium, ed NA*. Marrakech, Morocco: IEEE Computer Soc, San Diego, USA, July 2008, pp. 572–577.

[51] J. C. Chen, K. Yao, and R. E. Hudson, "Source localization and beamforming," *IEEE signal Processing Magazine*, pp. 30–39, March 2002.

[52] A. Abbasi and M. Younis, "A survey on clustering algorithms for wireless sensor network," *Computer Communication*, vol. 30, pp. 2826–2841, Oct 2007.

[53] S. C. Bang and S. Ann, "A robust adaptive algorithm and its performance analysis with contaminated-Gaussian noise," in *Procedeenigs of ISPACS*, Seol, Korea, Oct. 1994, pp. 295–300.

[54] S. C. Chan and Y. X. Zou, "A recursive least m-estimate algorithm for robust adaptive filtering in impulsive noise: Fast algorithm and convergence performance analysis," *IEEE Transactions on Signal Processing*, vol. 52, no. 4, pp. 975–991, April 2004.

[55] S. R. Kim and A. Efron, "Adaptive robust impulsive noise filtering," *IEEE Transactions on Signal Processing*, vol. 43, no. 8, pp. 1855–1866, Aug. 1995.

[56] W. Dargie and C. Poellabauer, *Fundamentals of Wireless Sensor Networks: Theory and Practice*. John Wiley and Sons, 2010.

[57] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, no. 4, pp. 393 – 422, 2002. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128601003024

[58] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer Network.*, vol. 52, no. 12, pp. 2292–2330, Aug. 2008.

[59] J. Chen, R. Hudson, and K. Yao, "Maximum-likelihood source localization and unknown sensor location estimation for wideband signals in the near-field," *IEEE Transactions on Signal Processing*, vol. 50, no. 8, pp. 1843–1854, Aug 2002.

[60] A. Swami, Q. Zhao, Y.-W. Hong, and L. Tong, Eds., *Wireless Sensor Networks: Signal Processing and Communications Perspectives*. John Wiley and Sons, 2007.

[61] M. Wang, L. Ci, P. Zhan, and Y. Xu, "Acoustic source localization in wireless sensor networks," in *IITA '07: Proceedings of the Workshop on Intelligent Information Technology Application*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 196–199.

[62] D. D. Lee, R. L. Kashyap, and R. N. Madan, "Robust decentralized direction-of-arrival estimation in contaminated noise," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 38, pp. 496–505, March 1990.

[63] I. Ziskind and M. Wax, "Maximum likelihood localization of multiple sources by alternating projection,," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 36, no. 10, pp. 1553–1560, Oct. 1988.

[64] D. Estin, R. Govindan, J. Heidemann, and S. Kumar, "Next century chanlanges:Scalable cordination in sensor network," in *Proceedings of the ACM/IEEE MobiComm'99*, Aug. 1999, pp. 263–270.

[65] F. S. Cattivelli and A. H. Sayed, "Analysis of spatial and incremental LMS processing for distributed estimation," *IEEE Transactions on Signal Processing*, vol. 59, no. 4, pp. 1465–1480, April 2011.

[66] G. A. Clark, S. K. Mitra, and S. R. Parker, "Block implementation of adaptive digital filters," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 29, no. 3, pp. 744–752, June 1981.

[67] K. Berberidis and S. Theodoridis, "A new fast block adaptive algorithm," *IEEE Trans. Signal Process.*, vol. 47, no. 1, pp. 75–87, Jan 1999.

[68] A. Feuer, "Performance analysis of the block least mean square algorithm," *IEEE Trans. on Circuit System.*, vol. 32, no. 9, pp. 960–963, Sep. 1985.

[69] G. Panda, B. Mulgrew, and C. F. N. Cowan, "A self-orthogonalizing efficient block adaptibe filter," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 34, no. 6, pp. 1573–1582, Dec. 1986.

[70] C. Burrus, "Block implementation of digital filters," *IEEE Trans. Circuit Theory*, vol. 18, no. 6, pp. 697–701, Nov. 1971.

[71] J. S. Lim and C. K. Un, "Block conjugate gradient algorithms for adaptive filtering," *Signal Processing, Elsevier*, vol. 55, pp. 65–77, 1996.

[72] J. S. Lim, "Block adaptive filtering algorithm based on the preconditioned conjugate gradient method," *Signal Processing, Elsevier*, vol. 71, pp. 79–84, 1998.

[73] X. Wang and H. V. Poor, "Joint channel estimation and symbol detection in rayleigh flat-fading channels with impulsive noise," *IEEE Communications Letters*, vol. 1, no. 1, pp. 19–21, Jan. 1997.

[74] N. J. Bershad, "On weight update saturation nonlinearities in LMS adaptation," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 38, no. 2, pp. 623–630, Feb. 1990.

[75] H. Fan and R. Vemuri, "Robust adaptive algorithms for active noise and vibration control," *IEEE International conference on Acoustics, Speech, Signal Processing (ICASSP'90)*, pp. 1137–1140 vol.2, April 1990.

[76] T. I. Haweel and P. Clarkson, "A class of order stastistic LMS algorithms," *IEEE Transactions on Signal Processing,*, vol. 40, no. 1, pp. 44–53, Jan. 1992.

[77] S. koike, "Adaptive threshold nonlinear algorithm for adaptive filters with robustness against impulse noise," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 45, no. 9, pp. 2391–2395, Sep. 1997.

[78] O. Abu-Ella and B. El-Jabu, "Optimal robust adaptive LMS algorithm without adaptation step-size," *Millimeter Waves, 2008. GSMM 2008. Global Symposium on*, pp. 249–251, April 2008.

[79] N. J. Bershad and M. Bonnet, "Saturation effects in LMS adaptive echo cancellation for binary data," *IEEE Transactions on Signal Processing*, vol. 38, no. 10, pp. 1687–1696, Oct. 1990.

[80] V. Delouille, R. Neelamani, and R. G. Baraniuk, "Robust distributed estimation using the embedded subgraphs algorithm," *IEEE Transactions on Signal Processing*, vol. 54, no. 8, pp. 2998–3010, Aug. 2006.

[81] S. J. Ban and S. W. Kim, "Wilcoxon adaptive algorithms for robust identification," *IEEE Electronics Letter*, vol. 45, no. 18, pp. 334–335, Aug. 2009.

[82] J. G. Hsieh, Y. L. Lin, and J. H. Jeng, "Preliminary study on Wilcoxon learning machines," *IEEE Transactions on Neural Networks*, vol. 19, no. 2, pp. 201–11, Feb. 2008.

[83] B. Widrow and S. D. Strearns, *Adaptive Signal Processing.* Englewood Cliffs, NJ:Prentice-Hall, 1985.

[84] S. Haykin, *Adaptive Filter Theory.* Englewood Cliffs, NJ:Prentice-Hall, 2001.

[85] T. Y. Al-Naffouri and A. H. Sayed, "Transient analysis of adaptive filters with error nonlinearities," *IEEE Transactions on Signal Processing*, vol. 51, no. 3, pp. 653–663, March 2003.

[86] ——, "Adaptive filters with error nonlinearities: Mean-square analysis and optimum design," *EURASIP Journal on Applied Signal Processing*, pp. 192–205, Oct. 2001.

[87] T. Panigrahi, G. Panda, and B. Mulgrew, "The performance analysis of error saturation nonlinearity LMS in impulsive noise based on weighted-energy conservation," *International Journal of Information and Communication Engineering*, vol. 6, no. 3, pp. 158–162, March 2010.

[88] B. Ottersten, M. Viberg, P. Stoica, and A. Nehorai, "Exact and large sample ML techniques for parameter estimation and detection in array processing," *Radar Array Processing, Simon Haykin (ed.), Springer-Verlag.*, pp. 99–152, Feb. 1993.

[89] P. Stoica and K. Sharman, "Maximum likelihood methods for direction-of-arrival estimation," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 38, no. 7, pp. 1132–1143, July 1990.

[90] J. Capon, "High-resolution frequency-wavenumber spectrum analysis," *Proc. IEEE*, vol. 57, no. 8, pp. 1408–1418, Aug. 1969.

[91] R. Schmidt, "Multiple emitter location and signal parameter estimation," *IEEE Transactions on Antennas and Propagation*, vol. 34, no. 3, pp. 276–280, March 1986.

[92] P. Stoica and A. Nehorai, "MUSIC, maximum likelihood and Cramer-Rao bound," vol. 37, no. 5, pp. 720–741, May 1989.

[93] P. Stoica, A. Nehorai, and T. Söderström, "Decentralized array processing using the MODE algorithm," *Circuits Syst. Signal Process.*, vol. 14, no. 1, pp. 17–38, 1995.

[94] A. Bertrand and M. Moonen, "Distributed adaptive node-specific signal estimation in fully connected sensor networks – part I: Sequential node updating," *IEEE Transactions on Signal Processing*, vol. 58, no. 10, pp. 5277 –5291, oct. 2010.

[95] ——, "Distributed adaptive node-specific signal estimation in fully connected sensor networks – part II: Simultaneous and asynchronous node updating," *IEEE Transactions on Signal Processing*, vol. 58, no. 10, pp. 5292 –5306, Oct. 2010.

[96] P. J. Chung and J. F. Böhme, "DOA estimation using fast EM and SAGE algorithms," *Signal Processing*, vol. 82, no. 11, pp. 1753–1762, 2002.

[97] B.-K. Yeo and Y. Lu, "Array failure correction with a genetic algorithm," *IEEE Transactions on Antennas and Propagation*, vol. 47, no. 5, pp. 823–828, May 1999.

[98] P. J. M. van Laarhoven and E. H. L. Aarts, *Simulated annealing: theory and applications (Mathematics and Its Applications)*. Springer publisher, 1987.

[99] D. Thompson and G. Bilbro, "Sample-sort simulated annealing," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 35, no. 3, pp. 625–632, June 2005.

[100] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *IEEE International Conference on Neural Networks, 1995*, vol. 4, nov/dec 1995, pp. 1942 –1948 vol.4.

[101] Eberhart and Y. Shi, "Particle swarm optimization: developments, applications and resources," in *Proceedings of the 2001 Congress on Evolutionary Computation, 2001*, vol. 1, 2001, pp. 81 – 86 vol. 1.

[102] T. Panigrahi, A. Patnaik, S. Sinha, and C. Christodoulou, "Amplitude only compensation for failed antenna array using particle swarm optimization," in *Antennas and Propagation Society International Symposium, 2008. AP-S 2008. IEEE*, July 2008, pp. 1–4.

[103] D. Boeringer and D. Werner, "Particle swarm optimization versus genetic algorithms for phased array synthesis," *IEEE Transactions on Antennas and Propagation*, vol. 52, no. 3, pp. 771–779, March 2004.

[104] J. Robinson and Y. Rahmat-Samii, "Particle swarm optimization in electromagnetics," *IEEE Transactions on Antennas and Propagation*, vol. 52, no. 2, pp. 397–407, Feb. 2004.

[105] B. Majhi, G. Panda, and B. Mulgrew, "Distributed identification of nonlinear processes using incremental and diffusion type pso algorithms," in *CEC'09: Proceedings of the Eleventh conference on Congress on Evolutionary Computation*. Piscataway, NJ, USA: IEEE Press, 2009, pp. 2076–2082.

[106] O. Younis, M. Krunz, and S. Ramasubramanian, "Node clustering in wireless sensor networks: recent developments and deployment challenges," *IEEE Network*, vol. 20, no. 3, pp. 20 – 25, may-june 2006.

[107] O. Younis and S. Fahmy, "Heed: A hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks," *IEEE Transactions on Mobile Computing*, vol. 3, pp. 366–379, 2004.

[108] X. Xu, N. Yuruk, Z. Feng, and T. Schweiger, "SCAN: A Structural Clustering Algorithm for Networks," in *Proceedings of the 13th International Conference on Knowledge Discovery and Data Mining (KDD '07)*, New York NY, 2007, pp. 824–833.

[109] M. Youssef, A. M. Youssef, and M. F. Younis, "Overlapping multi-hop clustering for wireless sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, pp. 1844–1856, Dec. 2009.

[110] G. J. Pottie and W. J. Kaiser, "Wireless integrated network sensors," *ACM communication*, vol. 43, pp. 51–53, May 2000.

[111] X. Liu, Q. Huang, and Y. Zhang, "Combs, Needles, Haystacks: Balancing push and pull for discovery in large scale sensor network," in *Proceedings of Second International Conference Embedded Network Sensor System*, Nov. 2004, pp. 122–133.

[112] S. Madden, M. Franklin, J. Hellerstein, and W. Wong, "TAG: a tiny aggregation service for ad-hoc sensor network." in *Proceedings of the ACM Symposium on Operating System Design and Implementation (OSDI)*, 2002.

[113] J. Ahn and B. Krishnamachari, "Modelling search cost in wireless sensor network," in *Technical Report CENG-2007-1, Computer Science Dept., Unversity of southern California*, 2007.

[114] D. P. Bertsekas and J. N. Tsitsiklis, "Comments on "coordination of groups of mobile autonomous agents using nearest neighbor rules"," *IEEE Transactions on Automatic Control*, vol. 52, no. 5, pp. 968 –969, May 2007.

[115] A. Geary and D. Bertsekas, "Incremental subgradient methods for nondifferentiable optimization," *Proceedings of the 38th IEEE Conference on Decision and Control, 1999*, vol. 1, pp. 907–912, 1999.

[116] D. P. Bertsekas, "A new class of incremental gradient methods for least squares problems," *SIAM J. Optim*, vol. 7, pp. 913–926, 1997.

[117] R. F. Woods, J. V. Mccanny, and J. G. Mcwhirter, "From bit level systolic arrays to HDTV processor chips," *Journal of Signal Process. Syst.*, vol. 53, no. 1-2, pp. 35–49, 2008.

[118] L.-K. Ting, R. Woods, and C. F. N. Cowan, "Virtex FPGA implementation of a pipelined adaptive LMS predictor for electronic support measures receivers," *IEEE Trans. on Very Large Scale Integr. Syst.*, vol. 13, no. 1, pp. 86–95, 2005.

[119] P. Fletcher and M. Dean, "Low complexity implementation of LMS algorithm," *Electronics Letter*, vol. 38, no. 15, pp. 836–837, 2002.

[120] R. Price, "A usefull theorem for non-linear devices having Gaussian inputs," *IRE Transactions on Information Theory*, vol. IT-4, pp. 69–72, June 1958.

[121] R. Pawula, "A modified version of Price's theorem," *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 285–288, April 1967.

[122] M. K. Simon and M.-S. Alouini, "A unified approach to the probability of error for noncoherent and differentially coherent modulations over generalized fading channels," *IEEE Trans. Commun.*, vol. 46, pp. 1625–1638, 1998.

[123] T. Y. Al-Naffouri, A. Zerguine, and M. Bettayeb, "Convergence analysis of the LMS algorithm with a general error nonlinearity and an IID input," *Conference Record of the Thirty-Second Asilomar Conference on Signals, Systems and Computers, 1998*, vol. 1, pp. 556–559, Nov 1998.

[124] N. Patwari, J. N. Ash, S. Kyperountas, A. O. H. III, R. L. Moses, and N. S. Correal, "Locating the nodes (cooperative localization in wireless sensor networks)," *IEEE Signal Processing Magazine*, vol. 22, no. 4, pp. 54–69, 2005.

[125] G. Gera and B. Mulgrew, "Vertically challenged array design for doa estimation," in *IEEE International conference on Acoustics, Speech, Signal Processing (ICASSP-2010)*, March 2010, pp. 2630 –2633.

[126] H. V. Trees, *Optimum array processing.* Wiley-Interscience Publication, 2002.

[127] P. Stoica and A. Nehorai, "Performance study of conditional and unconditional direction-of-arrival estimation," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 38, no. 10, pp. 1783–1795, Oct. 1990.

[128] T. Panigrahi, G. Panda, B. Mulgrew, and B. Majhi, "Maximum likelihood source localization in wireless sensor network using particle swarm optimization," in *the proceeding of International Conference on Electronics Systems (ICES-11)*, Jan. 2011, pp. 111–115.

[129] K. E. Parsopoulos and M. N. Vrahatis, "Recent approaches to global optimization problems through particle swarm optimization," *Natural Computing*, vol. 1, pp. 235–306, 2002.

[130] Y. Shi and R. C. Eberhart, "Parameter selection in particle swarm optimization," in *EP '98: Proceedings of the 7th International Conference on Evolutionary Programming VII*. London, UK: Springer-Verlag, 1998, pp. 591–600.

[131] M. Clerc and J. Kennedy, "The particle swarm - explosion, stability, and convergence in a multidimensional complex space," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, Feb 2002.

[132] D. Blatt and A. Hero, "Distributed maximum likelihood estimation for sensor networks," in *International Conference on Acoustics, Speech, and Signal Processing, (ICASP'04)*, 2004.

Related Publications

Journals

[1] **T. Panigrahi**, G. Panda and B. Mulgrew, "Novel Distributed Bearing Estimation Technique using Diffusion PSO Algorithm," *IET Wireless Sensor Systems*, in press.

[2] **T. Panigrahi**, P. M. Pradhan, G. Panda and B. Mulgrew, "Block Least Mean Squares Algorithm Over Distributed Wireless Sensor Network," *Journal of Computer Networks and Communication*, volume 2012, article ID 601287.

[3] **T. Panigrahi**, G. Panda and B. Mulgrew, "Distributed DOA Estimation Using Clustering of Sensor Nodes and Diffusion PSO Algorithm," *Journal of Swarm and Evolutionary Computing, Elsevier*, submitted after first revision.

[4] **T. Panigrahi**, G. Panda, B. Mulgrew, and B. Majhi, "Error Saturation Nonlinearities for Robust Incremental LMS over Wireless Sensor Network in Impulsive Noise," *Digital Signal Processing, Elsevier*, submitted after second revision.

Conferences

[1] **T. Panigrahi**, G. Panda and B. Mulgrew, "Robust Distributed Block LMS over Wireless Sensor Network in Impulsive Noise," in *8th International Conference on Distributed Computing and Internet Technology, ICDCIT-2012*, at KIIT University, pp. 261-292, Bhubaneswar, India, Feb 2012, LNCS Springer publisher.

[2] **T. Panigrahi**, B. Majhi and B. Mulgrew, "Robust Distributed Linear Parameter Estimation in Wireless Sensor Network," in *IEEE International Conference on Energy, Automation and Signals, ICEAS-2011*, SOA University, Bhubaneswar, INDIA, Dec. 2011.

[3] **T. Panigrahi**, D Hanumant Rao, G. Panda, B Mulgrew and B. Majhi, "Maximum Likelihood DOA Estimation in Distributed Wireless Sensor Network using Adaptive Particle Swarm Optimization", in *ACM International Conference on Communication, Computing and Security, ICCCS-2011*, at NIT Rourkela, Feb 2011, PP. 134-136.

[4] **T. Panigrahi**, G. Panda, B. Mulgrew and B. Majhi, "Maximum Likelihood Source Localization in Wireless Sensor Network using Particle Swarm Optimization", in

the International Conference on Electronics Systems, ICES-2011, at NIT Rourkela, Jan 2011, PP. 111-115.

[5] **T. Panigrahi**, G. Panda, B. Mulgrew and B. Majhi, "Robust Incremental LMS over Wireless Sensor Network in Impulsive Noise", in *IEEE International Conference on Computational Intelligence and Communication Network, CICN-2010*, at Bhopal, Nov 2010, pp. 205-209, (DOI 10.1109/CICN.2010.50).

[6] **T. Panigrahi**, P. M. Pradhan, G. Panda, and B. Mulgrew, "Transient Analysis of Error Saturation Nonlinearity LMS in Impulsive Noise", in *Silver Jubilee Conference on Communication Technology and VLSI Design, CommV-2009*, VIT University, Tamil Nadu, INDIA, Oct 2009, pp. 218–221.

[7] **T. Panigrahi**, P. M. Pradhan, G. Panda, B. Majhi, and B. Mulgrew, "Robust Distributed Optimization in Wireless Sensor Network," in *IEEE International Conference on Advances in Recent Technologies in Communication and Computing, ARTCom-2009*, The Windsor Castle, Kottayam, INDIA, Oct 2009.

BIO-DATA

Trilochan Panigrahi

Date of Birth: 28^{th} May, 1980

Correspondence:
PhD Scholar, Department of Electronics and Communication Engineering,
National Institute of Technology Rourkela, India – 769 008.

Ph: +91 94378 25979 (M)

e-mail: panigrahit@nitrkl.ac.in, tpanigrahi80@gmail.com

Qualification

- Ph.D. (Continuing)
  National Institute of Technology Rourkela, Odisha, India
- M.Tech. (Electronics and Communication Engineering)
  Biju Patnaik University of Technology, Odisha, India [First division]
- B.Sc.
  Berhampur University, Berhampur, Odisha, India [First class with distinction]
- M.Sc. (Electronics Science)
  Berhampur University, Berhampur, Odisha, India [First class with distinction]
- +2 (Science)
  Council of Higher Secondary Education, Odisha, India [First division]
- 10th
  Board of Secondary Education, Odisha, India [First division]

Professional Experience

Sr. Lecturer, NIST, Berhampur, Odisha, India, June 2005 – June 2008

Publications

- 06 Journal Articles
- 17 Conference Articles