DIFFUSION NORMALIZED LEAST MEAN SQUARES OVER WIRELESS SENSOR NETWORKS

BY

SYED ABDUL BAQI

A Thesis Presented to the
DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

ELECTRICAL ENGINEERING

MAY 2012

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS
DHAHRAN 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

This thesis, written by **Syed Abdul Baqi** under the direction of his thesis advisor and approved by his thesis committee, has been presented to and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE IN ELECTRICAL ENGINEERING.**

Thesis Committee

azzedine zerguine

Dr. Azzedine Zerguine (Advisor)

Dr. Abdelmalek Zidouri (Member)

Dr. Tareq Y. Al-Naffouri (Member)

Dr. Ali Ahmad Al-Shaikhi
(Department Chairman)

Dr. Salam A. Zummo
(Dean of Graduate Studies)

15/5/12

Date:

Dedicated to my loving Mother & Father

ACKNOWLEDGEMENTS

All praise be to Allah (The One and The Only Creator of everything) for His limitless blessings. May Allah bestow peace and His choicest blessings on His last prophet, Hazrat Muhammad (Peace Be Upon Him), his family (May Allah be pleased with them), his companions (May Allah be pleased with them) and his followers.

First and foremost gratitude is due to my esteemed university, King Fahd University of Petroleum and Minerals for my admittance, and for the opportunities for professional and personal development provided during my tenure.

I would like to acknowledge the advice and guidance of my thesis committee chairman and adviser, Dr. Azzedine Zerguine, whose expertise, understanding, and patience, added considerably to my graduate experience. I attribute the level of my Masters degree to his encouragement and effort and without him this thesis would not have been completed.

I would also like to thank my committee members, Dr. Abdelmalek Zidouri and Dr. Tareq Y. Al-Naffouri for their assistance and guidance during my research work. Their constructive and positive criticism helped me complete my thesis work. I would also like to make a special acknowledgement to Dr. Muhammad Omer Bin Saeed for his constant support and guidance throughout my time here at KFUPM, and especially during my research work.

Finally, I would like to thank my parents for their prayers and support from the bottom of my heart. I would also like to thank my friends, who provided me with the motivation and moral support to carry on with my work.

May Allah help us in following Islam according to Quran and Sunna!(Ameen)

TABLE OF CONTENTS

LIST OF FIGURES

THESIS ABSTRACT

Name:	Syed Abdul Baqi.
Title:	Diffusion Normalized Least Mean Squares over Wireless Sensor Networks.
Degree:	MASTER OF SCIENCE.
Major Field:	Electrical Engineering.
Date of Degree:	May 2012.

In recent work, distributed adaptive algorithms have been proposed to solve the problem of estimation over distributed networks. In diffusion protocol, each node in the network functions as an individual adaptive filter whose aim is to estimate a parameter of interest through local observations. All the estimates obtained from the nodes are then locally fused with their neighboring estimates in the network. Several algorithms have been proposed to exploit this distributed structure in order to improve estimation.

Diffusion techniques have been used based on the least mean square (LMS) or recursive least square (RLS) algorithm in wireless sensor networks. The LMS algorithm, unlike the RLS algorithm, is a very simple algorithm when the computational complexity is concerned. However, the performance of the LMS algorithm deteriorates as the amount of correlation increases among the input data.

To address this problem, in this network, a diffusion normalized least mean square (NLMS) algorithm is proposed. First, transient analysis of the proposed algorithm are derived. Second, the steady-state analysis are derived. Finally, simulation results are carried out to prove the better performance of the proposed algorithm and more importantly to corroborate the theoretical findings.

مُلخّص الأطروحة

الاسم: سيد عبد الباقي

العنوان: الانتشار المعدل ذو أقل متوسط تربيعي في شبكات الحساسات اللاسلكية

الدرجة: ماجستير في العلوم

التخصص: هندسة كهربائية

تاريخالدرجة: مايو 2012

حديثاً، تم اقتراح الخوارزميات التكيفية كحل لمشكلة التقدير في الشبكات الموزعة. كل نقطةفي بروتوكول الانتشارتتفرع في وظائف الشبكة تعمل كمرشح مستقل قابل للتكيف والذي يهدف الى تقدير المعاملات المرغوبة من خلال الملاحظات المحلية. كل التقديرات التي تم الحصول عليها من العقد يتم ربطها محليا بصمام مع التقديرات المجاورة لها في الشبكة.

وقد اقترحت عدة خوارزميات لاستغلال هذه البنية الموزعة من أجل تحسين التقدير. وقد استخدمت تقنيات نشر استنادا إلى خوارزميات أقل متوسط تربيعي (LMS)أو أقل تربيع تكراري (RLS)في شبكات الاستشعار اللاسلكية. على عكس خوارزميةأقل تربيع تكراري RLS فإن خوارزمية أقل متوسط تربيعي LMS ،هي خوارزمية بسيطة جدا عندما يتعلق الأمربالتعقيد الحسابي. ومع ذلك، فإن أداء خوارزمية أقل متوسط تربيعي LMSتتدهور كلما زادت العلاقة الإحصاية بينها وبين البيانات المدخلة.

لحل هذه المشكلة، في هذه الشبكة، اقترحتخوارزمية الانتشار: أقل متوسط تربيعي معدل (NLMS). في البداية نقوم بتحليل انتقالي من الخوارزمية المقترحة. ثانيا: نقوم باستنتاج حالة الاستقرار. أخيرا، يتم تنفيذ نتائج المحاكاة لإثبات الأداء الأفضل للخوارزمية المقترحة، والأهم من ذلك لثبيت النتائج النظرية.

NOMENCLATURE

Abbreviations

FC	:	Fusion Center
WSN	:	Wireless Sensor Network
LMS	:	Least Mean Square algorithm
NLMS	:	Normalized Least Mean Square algorithm
RLS	:	Recursive Least Square algorithm
APA	:	Affine Projection algorithm
DLMS	:	Diffusion Least Mean Square algorithm
DNLMS	:	Diffusion Normalized Least Mean Square algorithm
MSD	:	Mean Square Deviation
EMSE	:	Excess Mean Square Error

Notations

i	:	Iteration number
d	:	Measured value
$\boldsymbol{d}$	:	Measured value vector for entire network
$\boldsymbol{u}$	:	Input regressor vector
$\boldsymbol{U}$	:	Input regressor matrix for entire network
$\boldsymbol{w}^o$	:	Unknown vector
$\boldsymbol{w}^{(o)}$	:	Unknown vector for entire network
$\boldsymbol{\phi}_k$	:	Estimation vector for node k
$\boldsymbol{\phi}$	:	Estimation vector for entire network
$\boldsymbol{\psi}_k$	:	Intermediate estimation vector for node k
$\boldsymbol{\psi}$	:	Intermediate estimation vector for entire network
v	:	Scalar noise value
$\boldsymbol{v}$	:	Noise vector
J	:	Cost function
E[.]	:	Expectation operator
μ	:	Step-size
ε	:	Positive constant; which avoids the possibility of divide by zero
α	:	Correlation factor
M	:	Length of unknown vector
N	:	Number of nodes (for a network)
k, l	:	Node number
N_k	:	Number of neighbor nodes for node k
$\boldsymbol{R_u}$	:	Auto-correlation matrix for input regressor matrix for entire network
$\boldsymbol{R_{du}}$	:	Cross-correlation vector between input and output for entire network

$.^*$	:	Conjugate
$.^T$	:	Transpose
c_{kl}	:	Combiner weight between nodes k and l
$\boldsymbol{D}$	:	Step-size matrix for entire network
$\boldsymbol{C}, \boldsymbol{G}$	:	Combiner matrix
I	:	Identity matrix
λ	:	Eigenvalue
$\boldsymbol{\Lambda}$	:	Eigenvalue matrix
$\boldsymbol{\sigma}$	:	Weighting vector
$\boldsymbol{\Sigma}$	:	Weighting matrix
$\otimes$	:	Kronecker product
$\odot$	:	Block Kronecker product
bvec	:	Block vector operator
$\boldsymbol{R}_v$	:	Auto-Correlation matrix for noise
σ_v^2	:	Noise variance

CHAPTER 1

INTRODUCTION

Wireless sensor networks (WSNs) have become a hot topic of interest for researchers due to the multiplicity of their uses [1] - [4]. WSN has potential for numerous applications with several more in the pipeline [5]. Recently, WSN has also attracted much attention for decentralized estimation of signals of interest [6] - [15]. Here the objective is to enable each node to estimate a parameters of interest from the observed data. Each node in the network is provided with local learning abilities. Each node derives local estimates for the parameter of interest and shares it with their neighbors. The resulting algorithm is distributed; Distributed algorithm outperforms traditional non-cooperative schemes in terms of transient and steady-state mean-square error. Each node in the WSN functions as an individual adaptive filter to estimate the parameter of interest through local observations [16] - [18]. All the estimates obtained from the nodes are then locally fused with their neighboring estimates in the network.

Sensors need to be empowered with the required signal processing tools that fully utilize the distributive nature of the network as well as provide optimal results. This need has been addressed recently and several algorithms proposed. The aim of this thesis is to improve the signal processing capability of the distributed network in a novel way by applying diffusion normalized least mean squares (NLMS) algorithm.

The chapter is organized as follows. A background for wireless sensor networks and adaptive filtering is given in the context of the least mean square (LMS) and normalized least mean square (NLMS) algorithms. This is followed by a detailed literature survey. The aim of the thesis are then briefly explained.

1.1 Background

1.1.1 Wireless Sensor Network

A wireless sensor network (WSN) is a collection of spatially distributed autonomous devices that use sensors to monitor physical or environmental conditions [20] - [21]. The WSN comprises of “nodes”, where each node is connected to one (or more) sensors. Each sensor in the network node has typically several parts: a radio transceiver, an antenna, a microcontroller, an electronic circuit for interfacing with the sensors and a battery to fulfill energy requirement. The data collected from the different node sensors are fused together. Generally two types of WSNs are used in practice (see Fig. 1.1). One has a central processing unit known as a Fusion Center (FC). The sensors usually sense the required data and then transmit the data via a wireless channel to the fusion center. The sensors do not perform much processing except quantizing and coding the data before transmitting it to the fusion center. The fusion center acts as a data sink where data from all sensors is collected and then processed in order to ascertain the estimates of the parameters of interest. Unlike sensors, a fusion center has large processing capability as well as storage capacity.

A network devoid of a fusion network is generally termed as an ad hoc network. The sensors only communicate with neighboring sensors that are within communication range. In such a network, the sensors have access to data from their neighboring sensors only that can be attained via a wireless communication link between the sensors. The sensors are required to do a two-fold process in such cases. First, they need to acquire the available data from the nearby neighbors. Then each sensor performs some signal processing on the available data in order to estimate some parameter of interest.

Fusion Center-based WSN fails if the center fails, this is one of the major drawback. Also, the sensors located far from the center has problem communicating with the center. Such far located sensors would require higher power to send data to the center. However, this problem may be overcome by incorporating a multi-hop system but, this adds additional complexity to the system. In comparison,

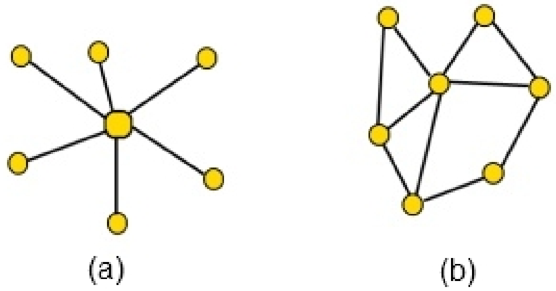


Figure 1.1: (a) A Fusion Center-based WSN; (b) An ad hoc topology

ad hoc networks are not prone to such limitation as they are working without any such processing center. Even if some sensors fail, ad hoc WSNs still continues to function with some performance degradation.

Currently, WSNs are beginning to be deployed at an accelerated pace. This new technology is exciting with unlimited potential for numerous application areas including medical, surveillance, environmental, military, localization, power spectrum estimation, target tracking and smart spaces [5]. However, recent study shows that without empowering the sensors with some signal processing capability, this goals cannot be achieved. Sensors need to be empowered with the required signal processing capability that fully utilize the distributive nature of the network for better energy performance. This need has been addressed recently and several algorithms proposed.

1.1.2 Adaptive Filtering

An adaptive filter can be defined as a self-designing system which tries to adjust its transfer function with the goal of meeting some well-defined target that depends upon the state of the system and its surroundings. So the system adjusts itself so as to respond to some phenomenon that is taking place in its surroundings. Adaptive algorithm avoids the need to know the exact signal statistics, which are rarely available in real practice. Also, these algorithms possess a tracking mechanism that enables them to track variation in the signal statistics. Because of these abilities adaptive filters are described as “smart systems” [18].

General application of adaptive filters is to estimate some unknown parameter. For example, in wireless communications the channel is usually unknown and is needed to be identified at the receiver in order to estimate the possible transformation that might have occurred on the transmitted information while propagating through the wireless channel. System needs to adapt itself until there is an approximate match that of the channel. This is usually an iterative process. At each iteration the adaptive filter outputs a certain value of the signal and tries to match it to the received signal.

Different stochastic-gradient algorithms are Least Mean Square (LMS), Normalized Least Mean Squares (NLMS), Affine Projective Algorithm (APA) and Recursive Least Squares Algorithm (RLS). The performance of these adaptive algorithms can be measured using the error. But the interesting point to note here is that the actual measure to check the performance is not the error itself. Error between the two outputs can be positive or negative and is generally a zero-mean process. Therefore, it is not reliable to develop an algorithm for adaptation based on just the error. A much better quantity would be the squared error or the absolute error. The simplest algorithms usually tend to minimize the mean square error. The error between the two outputs is squared and minimized. Repeating this process over several experiments generally gives a measure of how well the algorithm is performing. Hence the term mean square error (MSE). Recently, another measure is being adopted by researchers called mean square deviation (MSD). Instead of measuring the error between the channel output and the filter output, performance is measured by looking at the error between the coefficients of the channel and the filter. This shows how far off the filter is from reaching the actual channel.

1.1.3 LMS Algorithm

The LMS algorithm is based on stochastic gradient descent method that relate to producing the least mean squares of the error signal by mimicking a desired filter by adjusting its transfer function [27]. The LMS algorithm is an approximation of the steepest descent algorithm, which uses an instantaneous estimate of the gradient vector. The estimate of the gradient is based on sample values of the tap input vector and an error signal. The LMS algorithm incorporates an iterative procedure over each tap weight in the filter to makes successive corrections to the weight vector in the direction of the negative of the gradient vector which eventually leads to the minimum mean square error. The LMS algorithm is relatively simple when compared to other algorithms. Assume that we have access to several observations of regressor and desired data, given as

$$\{\boldsymbol{u}_0, \boldsymbol{u}_1, \boldsymbol{u}_2, ...\}$$

$$\{d(0), d(1), d(2), ...\}$$

Then LMS algorithm can be formulated as [27]:

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} + \mu \boldsymbol{u}_i{}^*[d(i) - \boldsymbol{u}_i \boldsymbol{w}_{i-1}] \tag{1.1}$$

The LMS algorithm is the simplest adaptive filtering algorithm described by (1.1). Because of its computational simplicity and ease of implementation, it is preferred in most applications.

1.1.4 ε-NLMS Algorithm

The main drawback of the LMS algorithm is that it is sensitive to the scaling of its input $\boldsymbol{u}_i$. This makes it very hard to choose a learning rate μ that guarantees stability of the algorithm. The ε-NLMS algorithm solves this problem by normalizing with the power of the input. The ε-NLMS algorithm can be formulated as [27]:

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} + \frac{\mu}{\varepsilon + ||\boldsymbol{u}_i||^2}\boldsymbol{u}_i{}^*[d(i) - \boldsymbol{u}_i\boldsymbol{w}_{i-1}] \tag{1.2}$$

In ε-NLMS algorithm the step size μ will be proportional to the power of $\boldsymbol{u}_i$. The positive constant ε avoids the possibility of division by zero, when the regressor is zero or close to zero. Hence, ε-NLMS algorithm can be considered as a special case of the LMS algorithm, where the step size depend upon the norm of $\boldsymbol{u}_i$. The regressor $\boldsymbol{u}_i$ with larger norm will lead to more substantial change to weight vector $\boldsymbol{w}_{i-1}$, when compared to regressor with smaller norm.

There has been research in the past focusing on the comparison between the LMS and the NLMS algorithms [22] - [24]. In 1993, Slock [24] studied the convergence behavior of both the algorithms and concluded that the NLMS algorithm is a potentially faster converging algorithm compared to the LMS algorithm. However, faster convergence comes at a cost of high computational complexity. More recent studies towards adjustable step-size tries to relax this trade-off [25] - [26].

1.2 Literature Survey

Different WSN topologies including Fusion Center based, ad hoc ones are lacking of hierarchies and depends on in-network processing to make agreement among sensors on the estimate of interest. In recent years, a great body of literature has been proposed, building up the field of consensus-based distributed signal processing. The tutorial in [28] gives idea about the general results and some list of related works which is good reading for a beginner in the field.

In [29] authors develop the best possible method for getting the consensus average in a distributed network by studying the results for several vast networks. In [30] authors discuss consensus issues with various types of graphs and provide some theoretical basis for their further development. Some authors suggests a decentralized algorithm that utilizes consensus to produce results similar to a centralized system [31] by projection into linear subspace. In all the schemes that are mentioned so far, all the data is collected by sensors at once and after locally exchanging messages it is reached to consensus.

In [32] authors discuss mobile environment tracking by providing algorithms for sensor fusion using novel consensus filters and suggest methods for designing such filters. In [33] authors suggest least squares solution in exchanging sequential peer-to-peer data. The algorithm is not robust enough to tackle the problem of estimating time-varying signals or dynamic systems. Apart from that, algorithm also suffers with high computational cost and requires extensive communication between sensors. Recently, ad hoc WSNs for distributed state estimation of dynamical systems has also received a lot of attention.

In many practical applications, sensors need to perform estimation in a constantly changing environment without having available a (statistical) model for the underlying processes of interest. This motivates the development of distributed adaptive estimation algorithms, the subject dealt with in the current work. Here each sensor transmits its update to the next sensor in the cycle, which then uses its own data to update this estimate. Such incremental schemes offer faster convergence than a centralized solution at a very low complexity cost which make the incremental algorithm very attractive. However, in case of any node failure, the cycle is broken and the network turns down.

In [14] author proposed a new algorithm that fully exploited the distributed nature of the network by getting rid of the topological constraints in [6]. The overall solution was more practical even with increased computational cost. The algorithm was termed as diffusion LMS. Each sensor forms a convex combination of the local estimates acquired from the nearby neighbors and this combined estimate is then used in the LMS recursion to update the local estimate. This new diffusion LMS algorithm outperforms the previous algorithm. In [9] the performance is further improved by diffusing not only the local estimates but also the sensor observations to nearby neighbors. This results in improving the flow of data across the WSN but can be computationally expensive, especially in the presence of communication noise. A new variant is suggested in [10], where the network was divided into several small networks, each with its own diffusion LMS algorithm network. This hierarchical structure provides improved performance but at the cost of extra computational cost. A distributed RLS scheme was introduced in [34] for applications where fast convergence is required and sensors can be burdened with increased computational load. A detailed analysis and design of the diffusion RLS scheme was given in [35].

In [36] authors discuss the diffusion algorithm used to synchronize the mobile sensors moving towards a specific target. The proposed algorithm is robust in estimating in constantly changing environment. Each sensor has access to a direction vector as well its own position. The sensed data by each node is simply the target position towards which the network has to travel. Due to the noise present in sensed data, the exact position has to be estimated and make sure that each node is moving in sync with the other nodes. Hence, target is estimated by each node by updating its own position and speed with sync to its neighboring sensors. Even though the work presented in [36] is application specific, yet it can be extended for both systems working in stationary as well as non-stationary environments.

So far discussed algorithms assume that each node has access to regressor data, but with the assumption that there is no correlation among the data. In a real scenario, however the absence of correlation cannot be neglected, therefore, this work addresses this issue. In a slowly changing environment, data will show high correlation. Moreover, one also needs to consider this correlation to fully exploit the distributed system for a better energy performance. The ε-NLMS algorithm is a variant of LMS algorithm that exhibits better performance in the presence of correlated data and provides generic solution.

1.3 Thesis Objectives

The aim of this thesis is to derive a distributed adaptive solution for ε-NLMS algorithm under diffusion protocol. The proposed diffusion ε-NLMS algorithm is then compared with diffusion LMS algorithm for non-white Gaussian data. The thesis objectives can be enumerated as:

1. To develop the diffusion ε-NLMS algorithm.
2. To find the range of values for which the step size guarantees convergence of the proposed algorithm.
3. To study the transient analysis of the diffusion ε-NLMS algorithm.
4. To study the steady-state analysis of the diffusion ε-NLMS algorithm.
5. To compare the performance of the diffusion ε-NLMS algorithm to that of the diffusion LMS algorithm.

CHAPTER 2

PROPOSED ADAPTIVE ALGORITHM

2.1 Introduction

In this chapter, we discuss the proposed ε-NLMS adaptive algorithm for diffusion protocol to implement cooperation among individual nodes in the network. We will begin our discussion by defining the problem statement, and deriving the proposed diffusion strategy for ε-NLMS algorithm. After that, the combined effect of several interconnected nodes in the network is discussed. Following this, in the coming sections we will discuss the transient and steady state analyses of the diffusion ε-NLMS algorithm.

Before we begin further, we shall introduce the data model that we adopted for the analysis, which has been used before in the adaptive literature, and rely on them to derive the performance measures.

A1 There exists a vector $\boldsymbol{w}^o$ at each node k such that $d_k(i) = \boldsymbol{u}_{k,i}\boldsymbol{w}^o + v_k(i)$.

A2 The noise sequence $v_k(i)$ is zero mean i.i.d. with variance $\sigma^2_{v,k}$, and with possibly different statistical profile for each node k.

A3 The noise sequence $v_k(i)$ is independent of $\boldsymbol{u}_{k,j}$ for all i, j.

A4 The initial condition $\boldsymbol{w}_k^{-1}$ at each node k is independent of all $\{d_k(i), \boldsymbol{u}_{k,i}, v_k(i)\}$.

2.2 Problem Statement

We have to estimate an Mx1 unknown vector $\boldsymbol{w}_0$ from the data collected from N distributed nodes (see Fig. 2.1). The scalar measurement sensed by node k, d_k at any time instant i, is given as

$$d_k(i) = \boldsymbol{u}_{k,i}\boldsymbol{w}^o + v_k(i), \tag{2.1}$$

where $v_k(i)$ is zero-mean additive white noise. The simplest solution to this estimation problem is for each node to estimate the unknown vector using only its own set of data. Such a case is termed as the no cooperation case as the nodes are not communicating with each other. The spatial diversity of the nodes is not being utilized here and so this case is counter productive as the poor performance of the nodes with low SNR will result in poor performance of the network. In

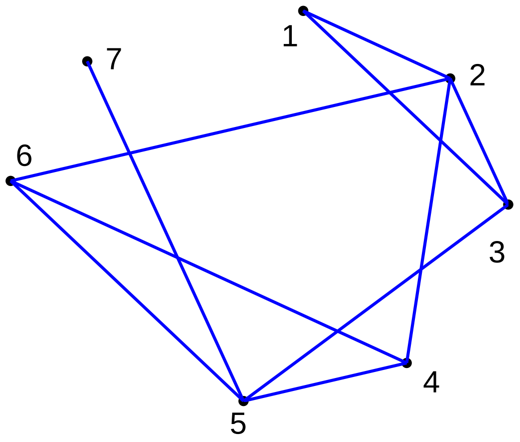


Figure 2.1: Adaptive Network of N (N=7) Nodes.

order to obtain a fully distributed estimation algorithm, a cost function is needed that defines the complete network. Thus, the cost function is defined as follows:

$$\begin{aligned} J(\boldsymbol{w}) &= \sum_{k=1}^{N} J_k(\boldsymbol{w}) \\ &= \sum_{k=1}^{N} \mathrm{E}\left[|d_k - \boldsymbol{u}_k \boldsymbol{w}|^2\right]. \end{aligned} \tag{2.2}$$

Consequently, the steepest descent solution for this problem is given as

$$\boldsymbol{w}_k^i = \boldsymbol{w}_k^{(i-1)} + \mu \sum_{k=1}^{N} \left(\boldsymbol{R}_{d\boldsymbol{u},k} - \boldsymbol{R}_{\boldsymbol{u},k} \boldsymbol{w}_k^{(i-1)} \right), \tag{2.3}$$

where $\boldsymbol{R}_{d\boldsymbol{u},k} = \mathrm{E}[d_k \boldsymbol{u}_k^*]$ is the cross-correlation between d_k and $\boldsymbol{u}_k$, and $\boldsymbol{R}_{\boldsymbol{u},k} = \mathrm{E}[\boldsymbol{u}_k^* \boldsymbol{u}_k]$ is the auto-correlation of $\boldsymbol{u}_k$. The recursion (2.3) requires full knowledge of the statistics of the entire network. Moreover, it requires exact statistical knowledge of the data, which is not possible in a practical scenario. A more practical solution utilizes the distributive nature of the network by dividing the cost function into local cost functions that add up to the global cost function. The solution to the local cost functions is similar to (2.3). However, a practical approach leads to the use of the least mean square (LMS) algorithm as a solution. The work in [14] gives a fully distributed solution, given as

$$\boldsymbol{\phi}_k^{(i-1)} = \sum_{l \in N_{k,i-1}} c_{kl} \boldsymbol{\psi}_l^{(i-1)} \tag{2.4a}$$

$$\boldsymbol{\psi}_k^i = \boldsymbol{\phi}_k^{(i-1)} + \mu_k \boldsymbol{u}_{k,i}{}^* (d_k(i) - \boldsymbol{u}_{k,i} \boldsymbol{\phi}_k^{(i-1)}) \tag{2.4b}$$

where $\boldsymbol{\psi}_k^i$ is the intermediate update, c_{kl} is the weight connecting node k to its neighboring node $l \in N_k$ and can be fixed according to a chosen rule [14], and μ_k is the step-size for the k^{th} node. Each node uses its own set of data, $\{d_k(i), \boldsymbol{u}_k(i)\}$, to get an intermediate update for the estimate. Then intermediate updates from neighbor nodes are combined together through a weighted sum to get the final update for the estimate.

The author in [14] propose diffusion LMS-based algorithms, on the assumption that there is no correlation among the data. In a real scenario, however the absence of correlation cannot be neglected. In a slowly changing environment, data will show high correlation. Moreover, one also needs to consider this correlation to fully exploit the distributed system for a better energy performance.

2.3 Diffusion ε-NLMS Algorithm

There are different types of cooperation strategy available in the literature. In diffusion strategy, every node k has access to estimates of its neighbors. The neighborhood of the nodes is defined in combiner matrix. Consider that at any given time $i-1$ node k has access to estimates $\boldsymbol{\psi}_k^{(i-1)}$ from its neighborhood.

Now these local estimates are fused together at node k

$$\boldsymbol{\phi}_k^{(i-1)} = \sum_{l \in N_{k,i-1}} c_{kl} \boldsymbol{\psi}_l^{(i-1)} \tag{2.5}$$

where $c_{kl} \geq 0$ are coefficients of combiner matrix. Here combiner matrix is generated using the Metropolis rule defined as follows

$$c_{kl} = \begin{cases} \frac{1}{\max(n_k, n_l)}, & \text{if}\, k \neq l\, are\, linked \\ 0, & \text{for}\, k\, and\, l\, not\, linked \\ 1 - \sum_{l \in N_k | k} c_{kl}, & \text{for}\, k = l \end{cases} \tag{2.6}$$

where n_k and n_l define the number of neighbor connections present for nodes k and l respectively. Combiner matrix calculated using the equation (2.6) defines the complete network topology. A non-zero entry c_{kl} states that nodes k and l are connected.

The coefficients of combiner matrix C must satisfy the following condition to ensure the stability and robustness of cooperative scheme

$$\sum_{l} c_{kl} = 1, \quad l \in N_{k,i-1} \tag{2.7}$$

Once we have the fused estimate $\boldsymbol{\phi}_k^{(i-1)}$ at node k, the ε-NLMS recursion at every node k can be given as

$$\boldsymbol{\psi}_k^i = \boldsymbol{\phi}_k^{(i-1)} + \frac{\mu_k}{\varepsilon + ||\boldsymbol{u}_{k,i}||^2} \boldsymbol{u}_{k,i}^* (d_k(i) - \boldsymbol{u}_{k,i} \boldsymbol{\phi}_k^{(i-1)}) \tag{2.8}$$

The above ε-NLMS recursion exhibits faster convergence behavior than LMS recursion for slow changing environment where data are expected to show high correlation. The positive constant ε avoids the possibility of division by zero, when the regressor is zero or close to zero.

In summary, the diffusion strategy for ε-NLMS recursion is given as

$$\boldsymbol{\phi}_k^{(i-1)} = \sum_{l \in N_{k,i-1}} c_{kl} \boldsymbol{\psi}_l^{(i-1)} \tag{2.9a}$$

$$\boldsymbol{\psi}_k^i = \boldsymbol{\phi}_k^{(i-1)} + \frac{\mu_k}{\varepsilon + ||\boldsymbol{u}_{k,i}||^2} \boldsymbol{u}_{k,i}^* (d_k(i) - \boldsymbol{u}_{k,i} \boldsymbol{\phi}_k^{(i-1)}) \tag{2.9b}$$

where c_{kl} are the coefficients of combiner matrix satisfying equation (2.6).

2.4 Network Global Model

We now proceed to state-space representation. Let us introduce some global quantities

$$\begin{aligned}
&\boldsymbol{\psi}^i = col\{\boldsymbol{\psi}_1^{(i)}, ..., \boldsymbol{\psi}_N^{(i)}\}, \qquad \boldsymbol{\phi}^{i-1} = col\{\boldsymbol{\phi}_1^{(i-1)}, ..., \boldsymbol{\phi}_N^{(i-1)}\},\\
&\boldsymbol{U}_i = diag\{\boldsymbol{u}_{1,i}, ..., \boldsymbol{u}_{N,i}\}, \qquad \boldsymbol{d}_i = col\{\boldsymbol{d}_1(i), ..., \boldsymbol{d}_N(i)\},\\
&\boldsymbol{H} = diag\{h_1(i)\mathbf{I}, ..., h_N(i)\mathbf{I}\},\\
&\boldsymbol{D} = diag\{\mu_1\mathbf{I}, \mu_2\mathbf{I}, ..., \mu_N\mathbf{I}\}
\end{aligned}$$

where $h_k(i)$ is a scalar normalization term at node k at time instance i. It is given as

$$h_k(i) = \frac{1}{\varepsilon + ||\boldsymbol{u}_{k,i}||^2}$$

The traditional model of the form

$$d_k(i) = \boldsymbol{u}_{k,i}\boldsymbol{w}^o + v_k(i) \tag{2.10}$$

where $v_k(i)$ is noise, which is independent in time and space with variance $\sigma_{v,k}^2$. Using the global quantities, equation (2.10) we can rewritten as

$$\boldsymbol{d}_i = \boldsymbol{U}_i\boldsymbol{w}^{(o)} + \boldsymbol{v}_i \tag{2.11}$$

where $\boldsymbol{w}^{(o)} = \boldsymbol{Q}\boldsymbol{w}^o$,

$\boldsymbol{Q} = col\{\mathbf{I_M}, ..., \mathbf{I_M}\}$ is $(NM \times M)$ with $\mathbf{I_M}$ the $M \times M$ identity matrix,

$\boldsymbol{v}_i = col\{v_1(i), v_2(i), ..., v_N(i)\}$ is $(N \times 1)$. Using the above relations, equation (2.9) can be represented in terms of global quantities:

$$\boldsymbol{\phi}^{i-1} = \boldsymbol{G}\boldsymbol{\psi}^{i-1} \tag{2.12a}$$

$$\boldsymbol{\psi}^{i} = \boldsymbol{\phi}^{i-1} + \boldsymbol{D}\boldsymbol{H}\boldsymbol{U}_i^*(\boldsymbol{d}_i - \boldsymbol{U}_i\boldsymbol{\phi}^{i-1}) \tag{2.12b}$$

where $\boldsymbol{G} = \boldsymbol{C} \otimes \mathbf{I_M}$ is the transition matrix of order $(NM \times NM)$. $\otimes$ represent kronecker product.

2.5 Comparison of Computational Complexity

In order to study the variation in performance of the two or more algorithms it is necessary to look at the computational cost as it tells us how much an algorithm gains in terms of computations as it loses in terms of performance. We first look at the complexity of the diffusion LMS algorithm and then move on to the diffusion ε-NLMS algorithm.

2.5.1 Diffusion LMS Algorithm

Consider a WSN of N nodes each with an unknown vector of length M. This means the complete data block matrix of size $N \times M$. The correlation matrix formed using this matrix will thus have the size $M \times M$. Let us analyze the computational complexity of each single node separately. The algorithm requires the evaluation of the inner product $\boldsymbol{u}_{k,i}\boldsymbol{\phi}_k^{(i-1)}$ between two vectors of size M which requires total of M multiplications and $M-1$ additions. After this, evaluation of the scalar $(d_k(i) - \boldsymbol{u}_{k,i}\boldsymbol{\phi}_k^{(i-1)})$ requires 1 additions. Evaluation of the product $\mu_k(d_k(i) - \boldsymbol{u}_{k,i}\boldsymbol{\phi}_k^{(i-1)})$ requires 1 multiplication. Further, multiplying the scalar $\mu_k(d_k(i) - \boldsymbol{u}_{k,i}\boldsymbol{\phi}_k^{(i-1)})$ with $\boldsymbol{u}^*_{k,i}$ requires M multiplications. The addition of two vectors $\mu_k\boldsymbol{u}^*_{k,i}(d_k(i) - \boldsymbol{u}_{k,i}\boldsymbol{\phi}_k^{(i-1)})$ and $\boldsymbol{\phi}_k^{(i-1)}$ requires M additions. Finally, the calculation of aggregate estimate $\sum\limits_{l \in N_{k,i-1}} c_{kl}\boldsymbol{\psi}_l^{(i-1)}$ at any node k requires NM multiplications and NM additions.

2.5.2 Diffusion ε-NLMS Algorithm

Consider a similar model of WSN with N nodes each with an unknown vector of length M. The computational complexity of ε-NLMS algorithm is same as LMS algorithm, expect multiplication of additional vector $\frac{1}{||\boldsymbol{u}_{k,i}||^2}$ with $\mu_k\boldsymbol{u}^*_{k,i}(d_k(i) - \boldsymbol{u}_{k,i}\boldsymbol{\phi}_k^{(i-1)})$, which requires additional M multiplications.

Table. 2.1 summarizes the computational complexity of diffusion LMS algorithm and diffusion ε-NLMS algorithm.

Algorithm Computation	Diffusion LMS	Diffusion ε-NLMS
Multiplication	$(2+N)M+1$	$(3+N)M+1$
Addition	$(2+N)M$	$(2+N)M$

Table 2.1: Computational complexity of Diffusion LMS and Diffusion ε-NLMS algorithms.

CHAPTER 3

TRANSIENT ANALYSIS OF DIFFUSION NORMALIZED LMS ALGORITHM

3.1 Introduction

The performance of the adaptive filters can be studied using transient analysis, which shows how fast and how stable adaptive filters can adapt to the changes in the signal. The study of the transient behavior of the adaptive algorithms is an essential part of adaptive filter performance analysis. In this chapter, we will discuss the stabilization effect of diffusion protocol on adaptive filter, in mean and mean square sense. Followed by, derivation of the learning curves of mean square deviation (MSD) and excess mean square error (EMSE).

3.2 Mean Analysis

We now focus our attention on mean analysis of adaptive filter for cooperative scheme. Recall global cooperative scheme defined by (2.12) can be represented as

$$\boldsymbol{\psi}^i = \boldsymbol{G}\boldsymbol{\psi}^{i-1} + \boldsymbol{D}\boldsymbol{H}\boldsymbol{U}_i^*(\boldsymbol{d}_i - \boldsymbol{U}_i\boldsymbol{G}\boldsymbol{\psi}^{i-1}) \tag{3.1}$$

Introduce the global weight error vector

$$\tilde{\boldsymbol{\psi}}^i = \boldsymbol{w}^{(o)} - \boldsymbol{\psi}^i \tag{3.2}$$

Now subtracting $\boldsymbol{w}^{(o)}$ from both sides of equation (3.1) and using the fact $\boldsymbol{G}\boldsymbol{w}^{(o)} = \boldsymbol{w}^{(o)}$, we get

$$\tilde{\boldsymbol{\psi}}^i = \boldsymbol{G}\,\tilde{\boldsymbol{\psi}}^{i-1} - \boldsymbol{D}\boldsymbol{H}\boldsymbol{U}_i{}^*(\boldsymbol{d}_i - \boldsymbol{U}_i\boldsymbol{G}\boldsymbol{\psi}^{i-1})$$

$$\tilde{\boldsymbol{\psi}}^i = \boldsymbol{G}\,\tilde{\boldsymbol{\psi}}^{i-1} - \boldsymbol{D}\boldsymbol{H}\boldsymbol{U}_i{}^*(\boldsymbol{U}_i\boldsymbol{w}^{(o)} + \boldsymbol{v}_i - \boldsymbol{U}_i\boldsymbol{G}\boldsymbol{\psi}^{i-1})$$

$$\tilde{\boldsymbol{\psi}}^i = \boldsymbol{G}\,\tilde{\boldsymbol{\psi}}^{i-1} - \boldsymbol{D}\boldsymbol{H}\boldsymbol{U}_i{}^*(\boldsymbol{U}_i\boldsymbol{G}\,\tilde{\boldsymbol{\psi}}^{i-1} + \boldsymbol{v}_i) \tag{3.3}$$

$$\tilde{\boldsymbol{\psi}}^i = (\mathbf{I_{NM}} - \boldsymbol{D}\boldsymbol{H}\boldsymbol{U}_i{}^*\boldsymbol{U}_i)\boldsymbol{G}\,\tilde{\boldsymbol{\psi}}^{i-1} - \boldsymbol{D}\boldsymbol{H}\boldsymbol{U}_i{}^*\boldsymbol{v}_i \tag{3.4}$$

Now taking expectation on both sides of (3.4) gives

$$\mathrm{E}[\tilde{\boldsymbol{\psi}}^i] = \{\mathbf{I_{NM}} - \boldsymbol{D}\,\mathrm{E}[\boldsymbol{H}\boldsymbol{U}_i{}^*\boldsymbol{U}_i]\}\boldsymbol{G}\,\mathrm{E}[\tilde{\boldsymbol{\psi}}^{i-1}] - \boldsymbol{D}\boldsymbol{H}\,\mathrm{E}[\boldsymbol{U}_i{}^*\boldsymbol{v}_i] \tag{3.5}$$

In above equation, second term on right-hand side becomes zero using independence assumption **A3**. Then equation (3.5) becomes

$$\mathrm{E}[\tilde{\boldsymbol{\psi}}^i] = \{\mathbf{I_{NM}} - \boldsymbol{D}\,\mathrm{E}[\boldsymbol{H}\boldsymbol{U}_i{}^*\boldsymbol{U}_i]\}\boldsymbol{G}\,\mathrm{E}[\tilde{\boldsymbol{\psi}}^{i-1}] \tag{3.6}$$

But unfortunately closed form for the term $\mathrm{E}[\boldsymbol{H}\boldsymbol{U}_i{}^*\boldsymbol{U}_i]$ is not available in general. And in order to continue our analysis, we consider the following assumption [19], which has been used before in the adaptive literature to derive closed form expressions.

$$\mathrm{E}\left[\frac{\boldsymbol{u}_{k,i}^*\boldsymbol{u}_{k,i}}{\varepsilon + \|\boldsymbol{u}_{k,i}\|^2}\right] \approx \frac{\mathrm{E}\left[\boldsymbol{u}_{k,i}^*\boldsymbol{u}_{k,i}\right]}{\mathrm{E}\left[\varepsilon + \|\boldsymbol{u}_{k,i}\|^2\right]} = \frac{\boldsymbol{R}_{u,k}}{\varepsilon + Tr(\boldsymbol{R}_{u,k})} \tag{3.7}$$

We will show using the simulations that this assumption leads to good results. Using assumption (3.7) equation (3.6) becomes

$$\mathrm{E}[\tilde{\boldsymbol{\psi}}^i] = (\mathbf{I_{NM}} - \boldsymbol{D}\boldsymbol{H}\boldsymbol{R}_u)\boldsymbol{G}\,\mathrm{E}[\tilde{\boldsymbol{\psi}}^{i-1}] \tag{3.8}$$

Now the stability in the mean sense is guaranteed if the eigenvalues of $\{\mathbf{I_{NM}} - \boldsymbol{D}\,\mathrm{E}[\boldsymbol{H}\boldsymbol{U}_i{}^*\boldsymbol{U}_i]\}\boldsymbol{G}$ lies inside a unit circle. Mathematically, the following condition must be satisfied:

$$|\lambda_{\max}\{\mathbf{I_{NM}} - \boldsymbol{D}\,\mathrm{E}[\boldsymbol{H}\boldsymbol{U}_i{}^*\boldsymbol{U}_i]\boldsymbol{G}\}| \leq 1 \tag{3.9}$$

Now using matrix 2-norms and the relation between transition matrix and combiner matrix ($\boldsymbol{G} = \boldsymbol{C} \otimes \mathbf{I_M}$) we get,

$$\|\boldsymbol{C}\|_2\,|\lambda_{\max}\{\mathbf{I_{NM}} - \boldsymbol{D}\mathrm{E}[\boldsymbol{H}\boldsymbol{U}_i{}^*\boldsymbol{U}_i]\}| \leq 1 \tag{3.10}$$

Since $\boldsymbol{C}$ is a stochastic and symmetric matrix whose coefficients are carefully selected satisfying the condition (2.7), we get

$$|\lambda_{\max}\{\mathbf{I_{NM}} - \boldsymbol{D}\mathrm{E}[\boldsymbol{H}\boldsymbol{U}_i{}^*\boldsymbol{U}_i]\}| \leq 1 \tag{3.11}$$

The convergence in the mean for single node is guaranteed for step size in the range [4]

$$0 < \mu < \frac{2}{\lambda_{\max}\left\{\mathrm{E}\left[\frac{\boldsymbol{U}_i{}^*\boldsymbol{U}_i}{\varepsilon + ||\boldsymbol{U}_i||^2}\right]\right\}} \tag{3.12}$$

Using Rayleigh-Ritz characterization for the maximum eigenvalue of any Hermitian matrix [4], and noting that $\boldsymbol{H}\boldsymbol{U}_i{}^*\boldsymbol{U}_i$ is block diagonal, we get

$$\begin{aligned}\lambda_{\max}\{\mathrm{E}[\boldsymbol{H}\boldsymbol{U}_i{}^*\boldsymbol{U}_i]\} &= \max_{||x||=1} x^*\mathrm{E}[\boldsymbol{H}\boldsymbol{U}_i{}^*\boldsymbol{U}_i]x \\ &= \max_{||x||=1} \mathrm{E}[x^*\boldsymbol{H}\boldsymbol{U}_i{}^*\boldsymbol{U}_i x] \\ &\leq 1\end{aligned} \tag{3.13}$$

Hence (3.12) can be rewritten as,

$$0 < \mu < 2. \tag{3.14}$$

Therefore, convergence in the mean sense is guaranteed, if step size μ satisfies the above condition. This is only a necessary condition for convergence in the mean square sense, which will be dealt in the coming section.

3.3 Mean Square Analysis

In this section we focus our attention towards mean-square analysis, which is concerned with the stability and convergence rate of the adaptive filter. This section will deal with formulating a suitable model to study the convergence behavior of adaptive filter for diffusion protocol, and the condition for which diffusion protocol has stabilizing effect on network.

3.3.1 Variance Relation

Let us begin our analysis by performing weighted energy balance and taking expectation on both sides of equation (3.3):

$$\mathrm{E}\left\|\tilde{\boldsymbol{\psi}}^i\right\|^2_{\boldsymbol{\Sigma}} = \mathrm{E}\left[\left(\boldsymbol{G}\tilde{\boldsymbol{\psi}}^{i-1} - \boldsymbol{DHU}_i{}^*(e^{\boldsymbol{G}}_{a,i}+\boldsymbol{v}_i)\right)^* \boldsymbol{\Sigma}\left(\boldsymbol{G}\tilde{\boldsymbol{\psi}}^{i-1} - \boldsymbol{DHU}_i{}^*(e^{\boldsymbol{G}}_{a,i}+\boldsymbol{v}_i)\right)\right] \tag{3.15}$$

where global a priori and a posteriori weighted estimated error is given as:

$$e^{\boldsymbol{G}}_{a,i} = \boldsymbol{U}_i\boldsymbol{G}\tilde{\boldsymbol{\psi}}^{i-1} \tag{3.16a}$$

$$e_{p,i} = \boldsymbol{U}_i\tilde{\boldsymbol{\psi}}^{i} \tag{3.16b}$$

$$\begin{aligned}\mathrm{E}\left\|\tilde{\boldsymbol{\psi}}^i\right\|^2_{\boldsymbol{\Sigma}} = {} & \mathrm{E}\left\|\tilde{\boldsymbol{\psi}}^{i-1}\right\|^2_{\boldsymbol{G}^*\boldsymbol{\Sigma}\boldsymbol{G}} - \mathrm{E}[\tilde{\boldsymbol{\psi}}^{*i-1}\boldsymbol{G}^*\boldsymbol{\Sigma}\boldsymbol{DHU}_i{}^*(e^{\boldsymbol{G}}_{a,i}+\boldsymbol{v}_i)] \\ & - \mathrm{E}[(e^{\boldsymbol{G}}_{a,i}+\boldsymbol{v}_i)\boldsymbol{U}_i\boldsymbol{H}^*\boldsymbol{D}^*\boldsymbol{\Sigma}\boldsymbol{G}\tilde{\boldsymbol{\psi}}^{i-1}] \\ & + \mathrm{E}[(e^{\boldsymbol{G}}_{a,i}+\boldsymbol{v}_i)^*\boldsymbol{U}_i\boldsymbol{H}^*\boldsymbol{D}^*\boldsymbol{\Sigma}\boldsymbol{DHU}_i{}^*(e^{\boldsymbol{G}}_{a,i}+\boldsymbol{v}_i)]\end{aligned} \tag{3.17}$$

Using the fact that $\boldsymbol{D}$ and $\boldsymbol{H}$ are diagonal matrices, transpose condition can be removed. And since input regressor $\boldsymbol{U}_i$ is independent of $\boldsymbol{v}_i$ above equation can be rewritten as:

$$\mathrm{E}\left\|\tilde{\boldsymbol{\psi}}^i\right\|^2_{\boldsymbol{\Sigma}} = \mathrm{E}\left\|\tilde{\boldsymbol{\psi}}^{i-1}\right\|^2_{\boldsymbol{G}^*\boldsymbol{\Sigma}\boldsymbol{G}} - \mathrm{E}[\tilde{\boldsymbol{\psi}}^{*i-1}\boldsymbol{G}^*\boldsymbol{\Sigma}\boldsymbol{D}\boldsymbol{H}\boldsymbol{U}_i{}^*(e^{\boldsymbol{G}}_{a,i})]$$
$$- \mathrm{E}[(e^{\boldsymbol{G}}_{a,i})^*\boldsymbol{U}_i\boldsymbol{H}\boldsymbol{D}\boldsymbol{\Sigma}\boldsymbol{G}\tilde{\boldsymbol{\psi}}^{i-1}]$$
$$+ \mathrm{E}[(e^{\boldsymbol{G}}_{a,i})^*\boldsymbol{U}_i\boldsymbol{H}\boldsymbol{D}\boldsymbol{\Sigma}\boldsymbol{D}\boldsymbol{H}\boldsymbol{U}_i{}^*(e^{\boldsymbol{G}}_{a,i})]$$
$$+ \mathrm{E}[\boldsymbol{v}_i{}^T\boldsymbol{U}_i\boldsymbol{H}\boldsymbol{D}\boldsymbol{\Sigma}\boldsymbol{D}\boldsymbol{H}\boldsymbol{U}_i{}^*\boldsymbol{v}_i] \qquad (3.18)$$

Substituting global a priori and a posteriori error (3.16) in above equation gives,

$$\mathrm{E}\left\|\tilde{\boldsymbol{\psi}}^i\right\|^2_{\boldsymbol{\Sigma}} = \mathrm{E}\left\|\tilde{\boldsymbol{\psi}}^{i-1}\right\|^2_{\boldsymbol{G}^*\boldsymbol{\Sigma}\boldsymbol{G}} - \mathrm{E}[(e^{\boldsymbol{H}\boldsymbol{D}\boldsymbol{\Sigma}\boldsymbol{G}}_{a,i})^*(e^{\boldsymbol{G}}_{a,i})] - \mathrm{E}[(e^{\boldsymbol{G}}_{a,i})^*(e^{\boldsymbol{H}\boldsymbol{D}\boldsymbol{\Sigma}\boldsymbol{G}}_{a,i})]$$
$$+ \mathrm{E}[(e^{\boldsymbol{G}}_{a,i})^*\boldsymbol{U}_i\boldsymbol{H}\boldsymbol{D}\boldsymbol{\Sigma}\boldsymbol{D}\boldsymbol{H}\boldsymbol{U}_i{}^*(e^{\boldsymbol{G}}_{a,i})] + \mathrm{E}[\boldsymbol{v}_i{}^*\boldsymbol{U}_i\boldsymbol{H}\boldsymbol{D}\boldsymbol{\Sigma}\boldsymbol{D}\boldsymbol{H}\boldsymbol{U}_i{}^*\boldsymbol{v}_i] \qquad (3.19)$$

This equality can be written more compactly as follows by introducing the random weighting matrix $\boldsymbol{\Sigma}'$ of order $(NM \times NM)$

$$\mathrm{E}\left\|\tilde{\boldsymbol{\psi}}^i\right\|^2_{\boldsymbol{\Sigma}} = \mathrm{E}\left\|\tilde{\boldsymbol{\psi}}^{i-1}\right\|^2_{\boldsymbol{\Sigma}'} + \mathrm{E}[v_i^*\boldsymbol{U}_i\boldsymbol{H}\boldsymbol{D}\boldsymbol{\Sigma}\boldsymbol{D}\boldsymbol{H}\boldsymbol{U}_i{}^*\boldsymbol{v}_i] \qquad (3.20)$$

where,

$$\boldsymbol{\Sigma}' = \boldsymbol{G}^*\boldsymbol{\Sigma}\boldsymbol{G} - \boldsymbol{G}^*\boldsymbol{\Sigma}\boldsymbol{H}\boldsymbol{D}\mathrm{E}[\boldsymbol{U}_i{}^*\boldsymbol{U}_i]\boldsymbol{G} - \boldsymbol{G}^*\mathrm{E}[\boldsymbol{U}_i{}^*\boldsymbol{U}_i]\boldsymbol{D}\boldsymbol{H}\boldsymbol{\Sigma}\boldsymbol{G}$$
$$+ \boldsymbol{G}^*\mathrm{E}[\boldsymbol{U}_i{}^*\boldsymbol{U}_i\boldsymbol{H}\boldsymbol{D}\boldsymbol{\Sigma}\boldsymbol{D}\boldsymbol{H}\boldsymbol{U}_i{}^*\boldsymbol{U}_i]\boldsymbol{G} \qquad (3.21)$$

Note that $\boldsymbol{H}$ is global normalization matrix defined earlier, which can be treated separately based on assumption (3.7). Above equation shows the time evolution of $\mathrm{E}\left\|\tilde{\boldsymbol{\psi}}^{i}\right\|_{\boldsymbol{\Sigma}}^{2}$ for some choices of interest for $\boldsymbol{\Sigma}$ of order $(NM \times NM)$. For mean square deviation (MSD) value of $\boldsymbol{\Sigma}$ is $\mathbf{I}$ and for excess mean square deviation (EMSE) value of $\boldsymbol{\Sigma}$ is $\boldsymbol{R}_u$.

3.3.2 Colored Gaussian Data

To continue with the analysis of mean square behavior we need to calculate the moments present in equation (3.20)-(3.21). For that we restrict our attention to colored Gaussian input with block diagonal correlation matrix $\boldsymbol{R}_u$. Let $\boldsymbol{R}_u = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^*$ denote the eigenvalue decomposition of the autocorrelation matrix, where $\boldsymbol{\Lambda}$ is a diagonal matrix given as $diag\{\boldsymbol{\Lambda}_1, \boldsymbol{\Lambda}_2, ..., \boldsymbol{\Lambda}_N\}$, and $\boldsymbol{Q}$ is unitary matrix $(\boldsymbol{Q}\boldsymbol{Q}^* = \boldsymbol{Q}^*\boldsymbol{Q} = \mathbf{I})$. The block diagonal correlation matrix $\boldsymbol{R}_u$ can be given as:

$$\boldsymbol{R}_u = \begin{bmatrix} \boldsymbol{R}_1 & 0 & \cdots & 0 \\ 0 & \boldsymbol{R}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & \boldsymbol{R}_N \end{bmatrix}$$

where,

$$\boldsymbol{R}_k = \begin{bmatrix} 1 & \alpha & \cdots & \alpha^{M-1} \\ \alpha & 1 & \cdots & \alpha^{M-2} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha^{M-1} & \alpha^{M-2} & \cdots & 1 \end{bmatrix} \quad 1 \leq k \leq N,$$

where M is the length of the unknown system, and $0 < \alpha < 1$ is the correlation coefficient. Evaluation of the moments in equation (3.21) can be simplified by using the assumption (3.7) and defining the following transformed quantities:

$$\begin{aligned} &\bar{\boldsymbol{\psi}}^i = \boldsymbol{Q}^* \tilde{\boldsymbol{\psi}}^i, \quad \bar{\boldsymbol{U}}_i = \boldsymbol{U}_i \boldsymbol{Q}, \quad \bar{\boldsymbol{G}} = \boldsymbol{Q}^* \boldsymbol{G} \boldsymbol{Q} \\ &\bar{\boldsymbol{\Sigma}} = \boldsymbol{Q}^* \boldsymbol{\Sigma} \boldsymbol{Q}, \quad \bar{\boldsymbol{\Sigma}}' = \boldsymbol{Q}^* \boldsymbol{\Sigma}' \boldsymbol{Q}, \quad \bar{\boldsymbol{D}} = \boldsymbol{Q}^* \boldsymbol{D} \boldsymbol{Q} \\ &\bar{\boldsymbol{H}} = \boldsymbol{Q}^* \boldsymbol{H} \boldsymbol{Q} \end{aligned}$$

Also $\bar{\boldsymbol{D}} = \boldsymbol{D}$ and $\bar{\boldsymbol{H}} = \boldsymbol{H}$ since both $\boldsymbol{D}$ and $\boldsymbol{H}$ are diagonal matrices. Using the above transformed quantities, variance relation (3.20)-(3.21) can be rewritten as,

$$\mathrm{E}\left\|\bar{\boldsymbol{\psi}}^i\right\|^2_{\bar{\boldsymbol{\Sigma}}} = \mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{i-1}\right\|^2_{\bar{\boldsymbol{\Sigma}}'} + \mathrm{E}[\boldsymbol{v}_i{}^* \bar{\boldsymbol{U}}_i \boldsymbol{H} \boldsymbol{D} \bar{\boldsymbol{\Sigma}} \boldsymbol{D} \boldsymbol{H} \bar{\boldsymbol{U}}_i{}^* \boldsymbol{v}_i] \tag{3.22}$$

$$\begin{aligned} \bar{\boldsymbol{\Sigma}}' = \bar{\boldsymbol{G}}^* \bar{\boldsymbol{\Sigma}} \bar{\boldsymbol{G}} &- \bar{\boldsymbol{G}}^* \bar{\boldsymbol{\Sigma}} \boldsymbol{D} \boldsymbol{H} \mathrm{E}[\bar{\boldsymbol{U}}_i{}^* \bar{\boldsymbol{U}}_i] \bar{\boldsymbol{G}} \\ &- \bar{\boldsymbol{G}}^* \mathrm{E}[\bar{\boldsymbol{U}}_i{}^* \bar{\boldsymbol{U}}_i] \boldsymbol{H} \boldsymbol{D} \bar{\boldsymbol{\Sigma}} \bar{\boldsymbol{G}} \\ &+ \bar{\boldsymbol{G}}^* \mathrm{E}[\bar{\boldsymbol{U}}_i{}^* \bar{\boldsymbol{U}}_i \boldsymbol{H} \boldsymbol{D} \bar{\boldsymbol{\Sigma}} \boldsymbol{D} \boldsymbol{H} \bar{\boldsymbol{U}}_i{}^* \bar{\boldsymbol{U}}_i] \bar{\boldsymbol{G}} \end{aligned} \tag{3.23}$$

Before we begin further in evaluating the required data moments in equations (3.22)-(3.23), we need to introduce some useful functions which will aid us in evaluating the data moments.

Block Vector Notation

We are already familiar with $vec\{.\}$ notation used for single node wireless sensor to replace an $M \times M$ arbitrary matrix by an $M^2 \times 1$ column vector by stacking the successive columns of matrix on top of each other [18]. For wireless sensor network with N nodes, we need to introduce a new notation $bvec\{.\}$ which has same function as $vec\{.\}$ except that it will process the data block-by-block. For block matrix $\boldsymbol{R}_u$ of order $NM \times NM$

$$\boldsymbol{R}_u = \begin{bmatrix} \boldsymbol{R}_{11} & \boldsymbol{R}_{12} & \cdots & \boldsymbol{R}_{1N} \\ \boldsymbol{R}_{21} & \boldsymbol{R}_{22} & \cdots & \boldsymbol{R}_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{R}_{N1} & \boldsymbol{R}_{N2} & \cdots & \boldsymbol{R}_{NN} \end{bmatrix} \tag{3.24}$$

where each block $\boldsymbol{R}_{kl}$ is of order $M \times M$. $k, l = 1, 2, ..., N$. Now consider the block columns are stacked on top of each other, yielding the $N^2 M \times M$ matrix

$$\boldsymbol{R}_u^c = \begin{bmatrix} \boldsymbol{R}_1 \\ \boldsymbol{R}_2 \\ \vdots \\ \boldsymbol{R}_N \end{bmatrix} \tag{3.25}$$

where $\boldsymbol{R}_l = col\{\boldsymbol{R}_{1l}, \boldsymbol{R}_{2l}, ..., \boldsymbol{R}_{Nl}\}$, $l = 1, 2, ..., N$. Once we have obtained $\boldsymbol{R}_u^c$, we can use the standard $vec\{.\}$ to vectorize individual block $\boldsymbol{R}_{kl}$, to obtain vector $\boldsymbol{r}_{kl}$ of order $M^2 \times 1$,

$$\boldsymbol{r}_{kl} = vec\{\boldsymbol{R}_{kl}\} \tag{3.26}$$

where r_{kl} are the coefficients of column matrix

$$\boldsymbol{r}_k = col\{r_{1l}, r_{2l}, ..., r_{Nl}\} \tag{3.27}$$

The final vectorized matrix $\boldsymbol{r}$ of order $(N^2 M^2 \times 1)$ can be represented as,

$$\boldsymbol{r} = bvec\{\boldsymbol{R}_u\} \tag{3.28}$$

The choice of notation is generally accepted as a two-directional operation, which maps block diagonal matrices to vectors and vectors to block diagonal matrices.

Therefore, we can write $\boldsymbol{R}_u = bvec^{-1}\{\boldsymbol{r}\}$ in order to recover the original block matrix from the column vector r.

Block Kronecker Product

The Kronecker product [18] of two matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, of order $M_a \times N_a$ and $M_b \times N_b$ respectively is denoted by $\boldsymbol{A} \otimes \boldsymbol{B}$ and is defined as the $M_a M_b \times N_a N_b$ matrix

$$\boldsymbol{A} \otimes \boldsymbol{B} = \begin{bmatrix} a_{11}\boldsymbol{B} & a_{12}\boldsymbol{B} & \cdots & a_{1N_a}\boldsymbol{B} \\ a_{21}\boldsymbol{B} & a_{22}\boldsymbol{B} & \cdots & a_{2N_a}\boldsymbol{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{M_a1}\boldsymbol{B} & a_{M_a2}\boldsymbol{B} & \cdots & a_{M_aN_a}\boldsymbol{B} \end{bmatrix} \tag{3.29}$$

This operation can be extended for wireless sensor network, where data will be processed block-by-block. Now consider two block matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, for which block Kronecker product is denoted by $\boldsymbol{A} \odot \boldsymbol{B}$. Its kl-block is represented as

$$[\boldsymbol{A} \odot \boldsymbol{B}]_{kl} = \begin{bmatrix} \boldsymbol{A}_{kl} \otimes \boldsymbol{B}_{11} & \boldsymbol{A}_{kl} \otimes \boldsymbol{B}_{12} & \cdots & \boldsymbol{A}_{kl} \otimes \boldsymbol{B}_{1N} \\ \boldsymbol{A}_{kl} \otimes \boldsymbol{B}_{21} & \boldsymbol{A}_{kl} \otimes \boldsymbol{B}_{22} & \cdots & \boldsymbol{A}_{kl} \otimes \boldsymbol{B}_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{A}_{kl} \otimes \boldsymbol{B}_{N1} & \boldsymbol{A}_{kl} \otimes \boldsymbol{B}_{N2} & \cdots & \boldsymbol{A}_{kl} \otimes \boldsymbol{B}_{NN} \end{bmatrix} \tag{3.30}$$

where $k, l = 1, 2, ..., N$. Block Kronecker product has several useful properties, but the one which we will be using is for evaluating the data moments as following.

For any matrices $\{\boldsymbol{A}, \boldsymbol{\Sigma}, \boldsymbol{B}\}$ of compatible dimensions, the following relation is satisfied

$$bvec\{\boldsymbol{A\Sigma B}\} = (\boldsymbol{B} \odot \boldsymbol{A}^T) bvec\{\boldsymbol{\Sigma}\} \tag{3.31}$$

$$bvec\{\boldsymbol{A\Sigma B}\} = (\boldsymbol{B} \odot \boldsymbol{A}^T)\boldsymbol{\sigma} \tag{3.32}$$

where $\boldsymbol{\sigma} = bvec\{\boldsymbol{\Sigma}\}$.

Now that we have defined block vector notation and block Kronecker product, we are equipped with the tools needed to evaluate the data moments present in (3.22)-(3.23), namely, $\mathrm{E}[\bar{\boldsymbol{U}}_i{}^*\bar{\boldsymbol{U}}_i]$, $\mathrm{E}[\boldsymbol{v}_i{}^*\bar{\boldsymbol{U}}_i\boldsymbol{HD}\bar{\boldsymbol{\Sigma}}\boldsymbol{DH}\bar{\boldsymbol{U}}_i{}^*\boldsymbol{v}_i]$ and $\mathrm{E}[\bar{\boldsymbol{U}}_i{}^*\bar{\boldsymbol{U}}_i\boldsymbol{HD}\bar{\boldsymbol{\Sigma}}\boldsymbol{DH}\bar{\boldsymbol{U}}_i{}^*\bar{\boldsymbol{U}}_i]$. We will derive these moments based upon the assumption stated in (3.7). This approximation is justified if fluctuation in the input signal from one iteration to the next is small enough [19].

The first moment is immediate to compute and is given by $\mathrm{E}[\bar{\boldsymbol{U}}_i{}^*\bar{\boldsymbol{U}}_i] = \boldsymbol{\Lambda}$. So that equation (3.23) can be rewritten as

$$\begin{aligned}\bar{\boldsymbol{\Sigma}}' = {} & \bar{\boldsymbol{G}}^*\bar{\boldsymbol{\Sigma}}\bar{\boldsymbol{G}} - \bar{\boldsymbol{G}}^*\bar{\boldsymbol{\Sigma}}\boldsymbol{DH\Lambda}\bar{\boldsymbol{G}} \\ & - \bar{\boldsymbol{G}}^*\boldsymbol{\Lambda HD}\bar{\boldsymbol{\Sigma}}\bar{\boldsymbol{G}} \\ & + \bar{\boldsymbol{G}}^*\mathrm{E}[\bar{\boldsymbol{U}}_i{}^*\bar{\boldsymbol{U}}_i\boldsymbol{HD}\bar{\boldsymbol{\Sigma}}\boldsymbol{DH}\bar{\boldsymbol{U}}_i{}^*\bar{\boldsymbol{U}}_i]\bar{\boldsymbol{G}} \end{aligned} \tag{3.33}$$

Second term on right-hand side of the above equation (3.33) can be evaluated using (3.32), which gives

$$\begin{aligned} bvec\{\bar{\boldsymbol{G}}^{*}\bar{\boldsymbol{\Sigma}}\boldsymbol{D}\boldsymbol{H}\boldsymbol{\Lambda}\bar{\boldsymbol{G}}\} &= (\bar{\boldsymbol{G}} \odot \bar{\boldsymbol{G}}^{*T})bvec\{\mathbf{I_{NM}}\bar{\boldsymbol{\Sigma}}\boldsymbol{D}\boldsymbol{H}\boldsymbol{\Lambda}\} \\ &= (\bar{\boldsymbol{G}} \odot \bar{\boldsymbol{G}}^{*T})(\boldsymbol{\Lambda}\boldsymbol{H}\boldsymbol{D} \odot \mathbf{I_{NM}})bvec\{\bar{\boldsymbol{\Sigma}}\} \\ &= (\bar{\boldsymbol{G}} \odot \bar{\boldsymbol{G}}^{*T})(\boldsymbol{\Lambda}\boldsymbol{H}\boldsymbol{D} \odot \mathbf{I_{NM}})\bar{\boldsymbol{\sigma}} \qquad (3.34) \end{aligned}$$

Similarly third term on right-hand side of the above equation (3.33) can be given as,

$$\begin{aligned} bvec\{\bar{\boldsymbol{G}}^{*}\boldsymbol{\Lambda}\boldsymbol{H}\boldsymbol{D}\bar{\boldsymbol{\Sigma}}\bar{\boldsymbol{G}}\} &= (\bar{\boldsymbol{G}} \odot \bar{\boldsymbol{G}}^{*T})bvec\{\boldsymbol{\Lambda}\boldsymbol{H}\boldsymbol{D}\bar{\boldsymbol{\Sigma}}\mathbf{I_{NM}}\} \\ &= (\bar{\boldsymbol{G}} \odot \bar{\boldsymbol{G}}^{*T})(\mathbf{I_{NM}} \odot \boldsymbol{\Lambda}\boldsymbol{H}\boldsymbol{D})bvec\{\bar{\boldsymbol{\Sigma}}\} \\ &= (\bar{\boldsymbol{G}} \odot \bar{\boldsymbol{G}}^{*T})(\mathbf{I_{NM}} \odot \boldsymbol{\Lambda}\boldsymbol{H}\boldsymbol{D})\bar{\boldsymbol{\sigma}} \qquad (3.35) \end{aligned}$$

To further proceed with the analysis we shall introduce the following assumption

$$\mathrm{E}\left[\frac{\boldsymbol{u}_{k,i}^{*}\boldsymbol{u}_{k,i}\boldsymbol{u}_{k,i}^{*}\boldsymbol{u}_{k,i}}{(\varepsilon + \|\boldsymbol{u}_{k,i}\|^{2})(\varepsilon + \|\boldsymbol{u}_{k,i}\|^{2})}\right] \approx \frac{\mathrm{E}\left[\boldsymbol{u}_{k,i}^{*}\boldsymbol{u}_{k,i}\boldsymbol{u}_{k,i}^{*}\boldsymbol{u}_{k,i}\right]}{\mathrm{E}\left[\varepsilon + \|\boldsymbol{u}_{k,i}\|^{2}\right]\mathrm{E}\left[\varepsilon + \|\boldsymbol{u}_{k,i}\|^{2}\right]} \qquad (3.36)$$

Since $\boldsymbol{D}$ and $\boldsymbol{H}$ are block diagonal matrices and using assumption (3.36), the fourth term on right-hand side of the equation (3.33) can be given as,

$$
\begin{aligned}
bvec\{\bar{\boldsymbol{G}}^{*}\mathrm{E}(\bar{\boldsymbol{U}}_i{}^{*}\bar{\boldsymbol{U}}_i\boldsymbol{H}\boldsymbol{D}\bar{\boldsymbol{\Sigma}}\boldsymbol{D}\boldsymbol{H}\bar{\boldsymbol{U}}_i{}^{*}\bar{\boldsymbol{U}}_i)\bar{\boldsymbol{G}}\} &= (\bar{\boldsymbol{G}} \odot \bar{\boldsymbol{G}}^{*T})bvec\{\mathrm{E}(\bar{\boldsymbol{U}}_i{}^{*}\bar{\boldsymbol{U}}_i\boldsymbol{H}\boldsymbol{D}\bar{\boldsymbol{\Sigma}}\boldsymbol{D}\boldsymbol{H}\bar{\boldsymbol{U}}_i{}^{*}\bar{\boldsymbol{U}}_i)\} \\
&= (\bar{\boldsymbol{G}} \odot \bar{\boldsymbol{G}}^{*T})(\boldsymbol{H}\boldsymbol{D} \odot \boldsymbol{D}\boldsymbol{H})bvec\{\mathrm{E}(\bar{\boldsymbol{U}}_i{}^{*}\bar{\boldsymbol{U}}_i\bar{\boldsymbol{\Sigma}}\bar{\boldsymbol{U}}_i{}^{*}\bar{\boldsymbol{U}}_i)\} \\
&= (\bar{\boldsymbol{G}} \odot \bar{\boldsymbol{G}}^{*T})(\boldsymbol{H}\boldsymbol{D} \odot \boldsymbol{D}\boldsymbol{H})bvec\{\boldsymbol{A}\}
\end{aligned}
\tag{3.37}
$$

where $\boldsymbol{A} = \mathrm{E}(\bar{\boldsymbol{U}}_i{}^{*}\bar{\boldsymbol{U}}_i\bar{\boldsymbol{\Sigma}}\bar{\boldsymbol{U}}_i{}^{*}\bar{\boldsymbol{U}}_i)$. For fourth-order moment of real Gaussian variables following condition [18] hold for kl-block of $\boldsymbol{A}$.

$$
\boldsymbol{A}_{kl} = \mathrm{E}\bar{\boldsymbol{u}}_{k,i}^{*}\bar{\boldsymbol{u}}_{k,i}\bar{\boldsymbol{\Sigma}}\bar{\boldsymbol{u}}_{l,i}^{*}\bar{\boldsymbol{u}}_{l,i} = \begin{cases} \boldsymbol{\Lambda}_k Tr(\boldsymbol{\Lambda}_k\bar{\boldsymbol{\Sigma}}_{kk}) + \gamma\boldsymbol{\Lambda}_k\bar{\boldsymbol{\Sigma}}_{kk}\boldsymbol{\Lambda}_k, & \text{for } k = l \\ \boldsymbol{\Lambda}_k\bar{\boldsymbol{\Sigma}}_{kl}, \boldsymbol{\Lambda}_l & \text{for } k \neq l \end{cases} \tag{3.38}
$$

where $\gamma = 1$ for complex data and $\gamma = 2$ for real data. Let us decompose matrix $\boldsymbol{A}$ as

$$
\boldsymbol{A} = [\boldsymbol{A}_1, \boldsymbol{A}_2, ..., \boldsymbol{A}_N] \tag{3.39}
$$

where $\boldsymbol{A}_l$ is the l^{th} block column of A

$$
\boldsymbol{A}_l = col\,\{\boldsymbol{A}_{1l}, \boldsymbol{A}_{2l}, ..., \boldsymbol{A}_{Nl}\} \tag{3.40}
$$

Now let us define block vectorized matrix of $\boldsymbol{A}$ as,

$$\boldsymbol{a} = bvec\{\boldsymbol{A}\} = col\{\boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_N\} \tag{3.41}$$

where $\boldsymbol{a}_l$ is a column vector defined as,

$$\boldsymbol{a}_l = col\{\boldsymbol{a}_{1l}, \boldsymbol{a}_{2l}, ..., \boldsymbol{a}_{Nl}\} \tag{3.42}$$

where $\boldsymbol{a}_{kl}$ is obtained by applying $vec\{.\}$ operation on $\boldsymbol{A}_{kl}$,

$$\boldsymbol{a}_{kl} = vec\{\boldsymbol{A}_{kl}\} = \begin{cases} (\boldsymbol{\lambda}_k \boldsymbol{\lambda}_k^T + \gamma \boldsymbol{\Lambda}_k \otimes \boldsymbol{\Lambda}_k)\bar{\boldsymbol{\sigma}}_{kk}, & \text{for } k = l \\ (\boldsymbol{\Lambda}_k \otimes \boldsymbol{\Lambda}_l)\bar{\boldsymbol{\sigma}}_{kl} & \text{for } k \neq l \end{cases} \tag{3.43}$$

where $\boldsymbol{\lambda}_k = vec\{\boldsymbol{\Lambda}_k\}$. Hence

$$\begin{aligned} \boldsymbol{a}_l = col\,\{&(\boldsymbol{\Lambda}_1 \otimes \boldsymbol{\Lambda}_l)\bar{\boldsymbol{\sigma}}_{1l}, (\boldsymbol{\Lambda}_1 \otimes \boldsymbol{\Lambda}_l)\bar{\boldsymbol{\sigma}}_{2l}\ , ..., \\ &(\boldsymbol{\lambda}_l \boldsymbol{\lambda}_l^T + \gamma \boldsymbol{\Lambda}_l \otimes \boldsymbol{\Lambda}_l)\bar{\boldsymbol{\sigma}}_{1l}, ..., (\boldsymbol{\Lambda}_N \otimes \boldsymbol{\Lambda}_l)\ \bar{\boldsymbol{\sigma}}_{Nl}\} \\ &= \mathbf{A_l}\bar{\boldsymbol{\sigma}}_l \end{aligned} \tag{3.44}$$

where, $\mathbf{A_l} = diag\,\{(\boldsymbol{\Lambda}_1 \otimes \boldsymbol{\Lambda}_l)\ , ..., (\boldsymbol{\lambda}_l \boldsymbol{\lambda}_l^T + \gamma \boldsymbol{\Lambda}_l \otimes \boldsymbol{\Lambda}_l), ..., (\boldsymbol{\Lambda}_N \otimes \boldsymbol{\Lambda}_l\ )\}$ and $\bar{\boldsymbol{\sigma}}_l = col\{\bar{\boldsymbol{\sigma}}_{1l}, \bar{\boldsymbol{\sigma}}_{2l}, ..., \bar{\boldsymbol{\sigma}}_{Nl}\}$. Hence,

$$bvec\{\boldsymbol{A}\} = \mathbf{A}\bar{\boldsymbol{\sigma}} \tag{3.45}$$

where, $\mathbf{A} = \mathbf{diag}\{\mathbf{A_1}, \mathbf{A_2}, ..., \mathbf{A_N}\}$ and $\bar{\boldsymbol{\sigma}} = bvec\{\bar{\boldsymbol{\Sigma}}\}$. Therefore substituting (3.45) in (3.37), we get the closed form for fourth-order moment

$$bvec\{\bar{\boldsymbol{G}}^{*}\mathrm{E}(\bar{\boldsymbol{U}}_i{}^{*}\bar{\boldsymbol{U}}_i\boldsymbol{H}\boldsymbol{D}\bar{\boldsymbol{\Sigma}}\boldsymbol{D}\boldsymbol{H}\bar{\boldsymbol{U}}_i{}^{*}\bar{\boldsymbol{U}}_i)\bar{\boldsymbol{G}}\} = (\bar{\boldsymbol{G}} \odot \bar{\boldsymbol{G}}^{*T})(\boldsymbol{H}\boldsymbol{D} \odot \boldsymbol{D}\boldsymbol{H})\mathrm{A}\bar{\boldsymbol{\sigma}} \quad (3.46)$$

Now let us find the closed form of second term in right-hand side of (3.22)

$$\begin{aligned}\mathrm{E}[\boldsymbol{v}_i{}^{*}\bar{\boldsymbol{U}}_i\boldsymbol{H}\boldsymbol{D}\bar{\boldsymbol{\Sigma}}\boldsymbol{D}\boldsymbol{H}\bar{\boldsymbol{U}}_i{}^{*}\boldsymbol{v}_i] &= \mathrm{E}[\mathrm{Tr}(\boldsymbol{v}_i\boldsymbol{v}_i{}^{*}\bar{\boldsymbol{U}}_i\boldsymbol{H}\boldsymbol{D}\bar{\boldsymbol{\Sigma}}\boldsymbol{D}\boldsymbol{H}\bar{\boldsymbol{U}}_i{}^{*})] \\ &= \mathrm{Tr}(\boldsymbol{\Lambda}_v\mathrm{E}[\bar{\boldsymbol{U}}_i\boldsymbol{H}\boldsymbol{D}\bar{\boldsymbol{\Sigma}}\boldsymbol{D}\boldsymbol{H}\bar{\boldsymbol{U}}_i{}^{*}]) \qquad (3.47)\end{aligned}$$

where $\boldsymbol{\Lambda}_v > 0$ is a diagonal matrix given by

$$\boldsymbol{\Lambda}_v = diag\left\{\boldsymbol{\sigma}_{v,1}^2, \boldsymbol{\sigma}_{v,2}^2, ..., \boldsymbol{\sigma}_{v,N}^2\right\}$$

The kl-block of $\mathrm{E}[\bar{\boldsymbol{U}}_i\boldsymbol{H}\boldsymbol{D}\bar{\boldsymbol{\Sigma}}\boldsymbol{D}\boldsymbol{H}\bar{\boldsymbol{U}}_i{}^{*}]$ is given by

$$\left\{\mathrm{E}[\bar{\boldsymbol{U}}_i\boldsymbol{H}\boldsymbol{D}\bar{\boldsymbol{\Sigma}}\boldsymbol{D}\boldsymbol{H}\bar{\boldsymbol{U}}_i{}^{*}]\right\}_{kl} = \begin{cases} 0 & \text{for } k \neq l \\ \mu_k^2 h_k^2 Tr(\boldsymbol{\Lambda}_k\bar{\boldsymbol{\Sigma}}_{kk}) = \mu_k^2 h_k^2 \boldsymbol{\lambda}_k^T \bar{\boldsymbol{\sigma}}_{kk} & \text{for } k = l \end{cases} \quad (3.48)$$

so that (3.47) can be written as

$$\mathrm{E}\boldsymbol{v}_i{}^{*}\bar{\boldsymbol{U}}_i\boldsymbol{H}\boldsymbol{D}\bar{\boldsymbol{\Sigma}}\boldsymbol{D}\boldsymbol{H}\bar{\boldsymbol{U}}_i{}^{*}\boldsymbol{v_i} = \boldsymbol{b}^T\bar{\boldsymbol{\sigma}} \quad (3.49)$$

where $\boldsymbol{b} = bvec\{\boldsymbol{R}_v\boldsymbol{D}^2\boldsymbol{H}^2\boldsymbol{\Lambda}\}$ and $\boldsymbol{R}_v = \boldsymbol{\Lambda}_v \odot \mathbf{I_M}$. Now substituting closed form of all the data moments in (3.22)-(3.23), we get

$$\mathrm{E}\left\|\bar{\boldsymbol{\psi}}^i\right\|^2_{\bar{\boldsymbol{\sigma}}} = \mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{i-1}\right\|^2_{\bar{\boldsymbol{F}}\bar{\boldsymbol{\sigma}}} + \boldsymbol{b}^T\bar{\boldsymbol{\sigma}} \tag{3.50}$$

$$\begin{aligned}\bar{\boldsymbol{F}} = (\bar{\boldsymbol{G}} \odot \bar{\boldsymbol{G}}^{*T})\,[\mathbf{I_{N^2M^2}} - (\mathbf{I_{NM}} \odot \boldsymbol{\Lambda}\boldsymbol{H}\boldsymbol{D})\\ -(\boldsymbol{\Lambda}\boldsymbol{H}\boldsymbol{D} \odot \mathbf{I_{NM}}) + (\boldsymbol{H}\boldsymbol{D} \odot \boldsymbol{D}\boldsymbol{H})\mathrm{A}]\end{aligned} \tag{3.51}$$

The transient behavior of the network is characterized by (3.50). The vector weighting factor $\{\bar{\boldsymbol{\sigma}}, \bar{\boldsymbol{F}}\bar{\boldsymbol{\sigma}}\}$ in this expression is compact representation for the actual weighting matrices $\{bvec\{\bar{\boldsymbol{\Sigma}}\}, bvec\{\bar{\boldsymbol{\Sigma}}'\}\}$. Equation (3.50) - (3.51) gives the variance relation used to characterize the mean-square behavior of the filter for diffusion protocol.

3.3.3 Constructing the Learning Curves

The desired quantity $\mathrm{E}\left\|\bar{\boldsymbol{\psi}}^i\right\|^2$ can be obtained from variance relation (3.50) if $\bar{\boldsymbol{\Sigma}}$ is chosen as $\bar{\boldsymbol{\Sigma}} = \mathbf{I_{NM}}$. This corresponds to choosing $\bar{\boldsymbol{\sigma}}$ as

$$\bar{\boldsymbol{\sigma}} = (1/N)bvec\{\mathbf{I_{NM}}\} = \boldsymbol{q}_n \tag{3.52}$$

Equation (3.50), for successive time instants,

$$\begin{aligned}
\mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{i}\right\|^{2}_{\boldsymbol{q}_n} &= \mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{i-1}\right\|^{2}_{\bar{\boldsymbol{F}}\boldsymbol{q}_n} + \boldsymbol{b}^{T}\boldsymbol{q}_n \\
\mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{i-1}\right\|^{2}_{\bar{\boldsymbol{F}}\boldsymbol{q}_n} &= \mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{i-2}\right\|^{2}_{\bar{\boldsymbol{F}}^{2}\boldsymbol{q}_n} + \boldsymbol{b}^{T}\bar{\boldsymbol{F}}\boldsymbol{q}_n \\
&\vdots \\
\mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{0}\right\|^{2}_{\bar{\boldsymbol{F}}^{i}\boldsymbol{q}_n} &= \mathrm{E}\left\|\bar{\boldsymbol{w}}^{(o)}\right\|^{2}_{\bar{\boldsymbol{F}}^{i+1}\boldsymbol{q}_n} + \boldsymbol{b}^{T}\bar{\boldsymbol{F}}^{i}\boldsymbol{q}_n
\end{aligned}$$

where $\bar{\boldsymbol{w}}^{(o)} = \boldsymbol{Q}^{*}\boldsymbol{w}^{(o)}$. Above equation can be written in more compact form as,

$$\mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{i}\right\|^{2}_{\boldsymbol{q}_n} = \mathrm{E}\left\|\bar{\boldsymbol{w}}^{(o)}\right\|^{2}_{\bar{\boldsymbol{F}}^{i+1}\boldsymbol{q}_n} + \boldsymbol{b}^{T}\left(\sum_{k=0}^{i}\bar{\boldsymbol{F}}^{k}\right)\boldsymbol{q}_n \tag{3.53}$$

From this result it can be verified that $\mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{i}\right\|^{2}$ satisfies the following recursion

$$\mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{i}\right\|^{2}_{\boldsymbol{q}_n} = \mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{i-1}\right\|^{2}_{\boldsymbol{q}_n} + \boldsymbol{b}^{T}\bar{\boldsymbol{F}}^{i}\boldsymbol{q}_n - \left\|\bar{\boldsymbol{w}}^{(o)}\right\|^{2}_{\bar{\boldsymbol{F}}^{i}(\mathbf{I}-\bar{\mathbf{F}})\mathbf{q_n}} \tag{3.54}$$

This recursion describes the time evolution of $\mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{i}\right\|^{2}$. Global learning curve for mean square deviation (MSD) can be obtained by iterating recursion (3.54). In similar way, by selecting $\bar{\boldsymbol{\sigma}} = (1/N)bvec\{\boldsymbol{\Lambda}\} = \boldsymbol{\lambda}_{\varsigma}$ gives global learning curve for excess mean square error as,

$$\mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{i}\right\|^{2}_{\boldsymbol{\lambda}_{\varsigma}} = \mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{i-1}\right\|^{2}_{\boldsymbol{\lambda}_{\varsigma}} + \boldsymbol{b}^{T}\bar{\boldsymbol{F}}^{i}\boldsymbol{\lambda}_{\varsigma} - \left\|\bar{\boldsymbol{w}}^{(o)}\right\|^{2}_{\bar{\boldsymbol{F}}^{i}(\mathrm{I}-\bar{\boldsymbol{F}})\boldsymbol{\lambda}_{\varsigma}} \tag{3.55}$$

We can now use the above learning curves (3.54) - (3.55) of mean square deviation (MSD) and excess mean square error (EMSE) to study the behavior of the proposed diffusion ε-NLMS adaptive algorithm for colored Gaussian input data.

3.3.4 Mean Square Stability

Variance relation (3.50) - (3.51) is used in this section to discuss the stability of the adaptive filter for diffusion protocol. Equation (3.51) can be rewritten in more compact form as,

$$\bar{\boldsymbol{F}} = (\bar{\boldsymbol{G}} \odot \bar{\boldsymbol{G}}^{*T})\mathbf{K} \tag{3.56}$$

where $\mathbf{K} = [\mathrm{I}_{\mathbf{N^2M^2}} - (\mathrm{I}_{\mathbf{NM}} \odot \boldsymbol{\Lambda}\mathbf{HD}) - (\boldsymbol{\Lambda}\mathbf{HD} \odot \mathrm{I}_{\mathbf{NM}}) + (\mathbf{HD} \odot \mathbf{DH})\mathrm{A}]$. Step size μ must satisfy $\left|\lambda(\bar{\boldsymbol{F}})\right| < 1$ in order to guarantee stability in mean square sense for cooperative strategy. Using matrix 2-norms

$$\begin{aligned} \left\|\bar{\boldsymbol{F}}\right\|_2 &= \left\|(\bar{\boldsymbol{G}} \odot \bar{\boldsymbol{G}}^{*T})\mathbf{K}\right\|_2 \\ &= \left\|\mathbf{P}^T(\bar{\boldsymbol{G}} \otimes \bar{\boldsymbol{G}}^{*T})\mathbf{P} \cdot \mathbf{K}\right\|_2 \\ &= \left\|\bar{\boldsymbol{G}}\right\|_2 \cdot \left\|\bar{\boldsymbol{G}}^{*T}\right\|_2 \cdot \|\mathbf{K}\|_2 \end{aligned} \tag{3.57}$$

where $\mathbf{P}$ is some permutation matrix. Recall $\boldsymbol{G} = \boldsymbol{C} \otimes \mathbf{I_M}$, we get

$$\left|\lambda_{\max}((\bar{\boldsymbol{G}} \odot \bar{\boldsymbol{G}}^{*T})\mathbf{K})\right| \leq \|\boldsymbol{C}\|_2^2 \cdot |\lambda_{\max}(\mathbf{K})| \tag{3.58}$$

Since $\boldsymbol{C}$ is a stochastic and symmetric matrix whose coefficients satisfy the condition (2.7), we get

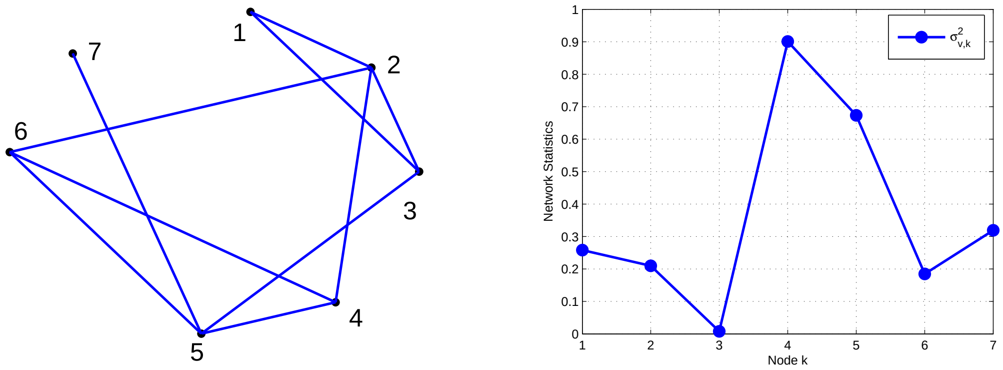
$$\left|\lambda_{\max}((\bar{\boldsymbol{G}} \odot \bar{\boldsymbol{G}}^{*T})\mathbf{K})\right| \leq |\lambda_{\max}(\mathbf{K})| \tag{3.59}$$

From the above relation it is clear that cooperation under diffusion protocol has stabilizing effect on the network in mean square sense. Since the largest eigenvalue of $(\bar{\boldsymbol{G}} \odot \bar{\boldsymbol{G}}^{*T})\mathbf{K}$ is smaller when compared to largest eigenvalue of $(\mathbf{K})$.

3.4 Simulation Results

In this section, the results of the computer simulations are presented which are made to investigate the performance behavior of the diffusion ε-NLMS algorithm. A number of simulation results are carried out to corroborate the theoretical finding.

First, we will show how the diffusion ε-NLMS algorithm provides better performance in terms of the mean-square deviation as compared to the diffusion LMS algorithm for correlated input data. After this, we will present a number of simulations which shows that there is a good match between the theoretical and simulation results of the diffusion ε-NLMS algorithm. Fig. 3.1 defines the network topological structure and statistics of the adaptive network following diffusion protocol.



(a)

(b)

Figure 3.1: Network topology and statistics.

3.4.1 Comparison of Diffusion LMS and Diffusion ε-NLMS

In this section, we will compare the diffusion ε-NLMS algorithm to the diffusion LMS algorithm and show that, the diffusion ε-NLMS algorithm outperforms diffusion LMS algorithm in mean square deviation and excess mean square error sense for correlated input data.

Consider a network of 7 nodes each with tap length of 10. A non-white Gaussian input process with correlation factor α was fed into both the diffusion LMS and diffusion ε-NLMS algorithms while the output noise was set as a zero mean random process with variance 0.01. The experiment was conducted for Gaussian noisy environment and the results were averaged over 100 experiments.

The experiment results show that the performance of diffusion LMS and diffusion ε-NLMS algorithms is almost identical for small correlation factor ($\alpha = 0.1$), which shows that the diffusion of data also helps to mitigate data correlation.

However, when the correlation factor α is increased, diffusion ε-NLMS algorithm provides better performance than diffusion LMS algorithm.

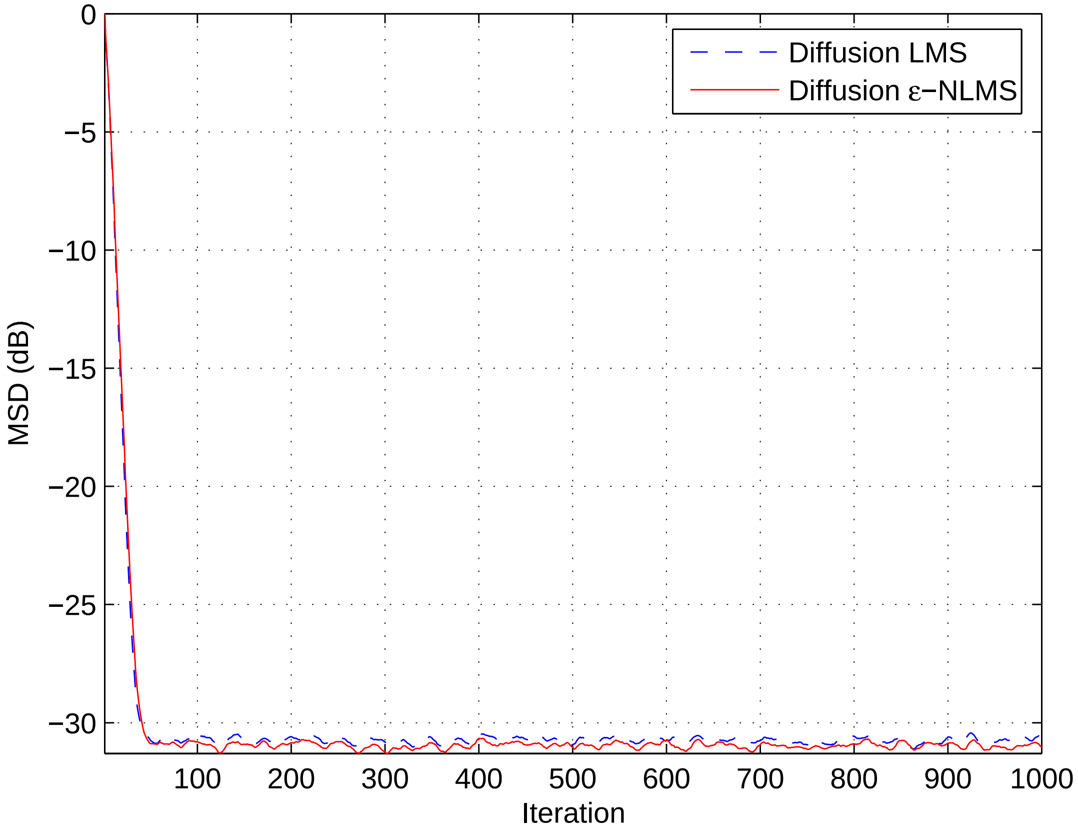


Figure 3.2: Global MSD of the diffusion LMS and diffusion ε-NLMS algorithm for $\alpha = 0.1$.

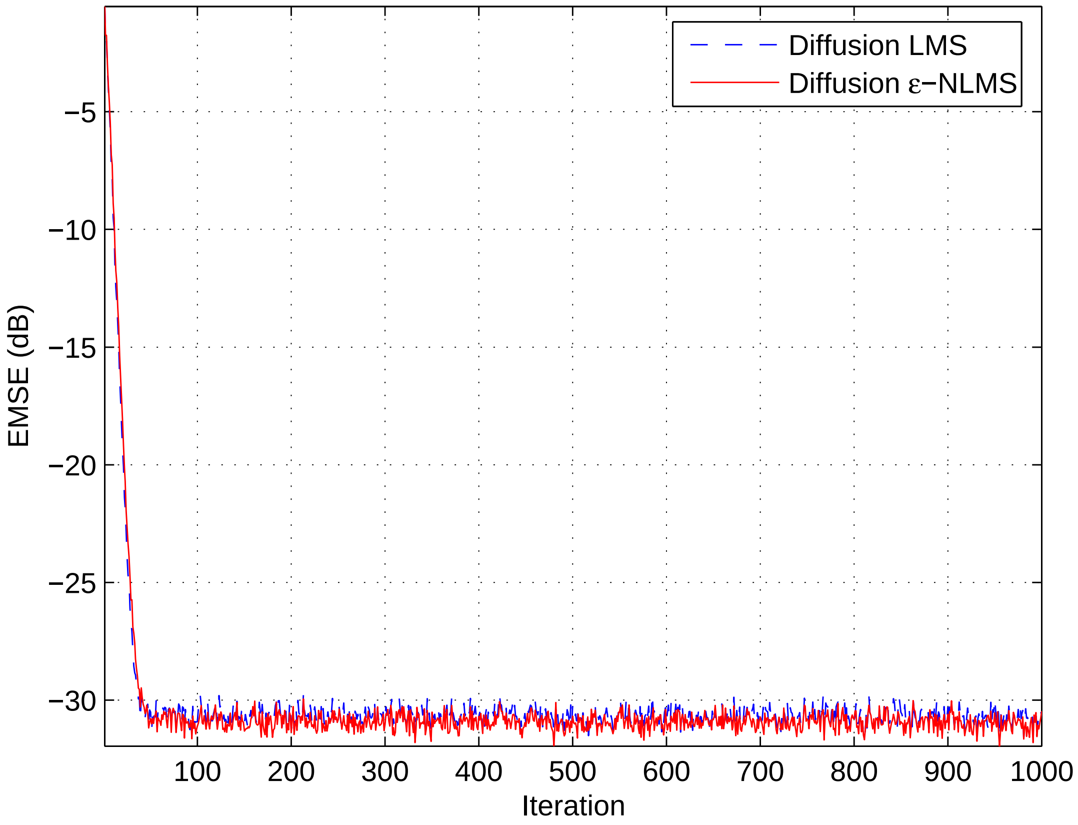


Figure 3.3: Global EMSE of the diffusion LMS and diffusion ε-NLMS algorithm for $\alpha = 0.1$.

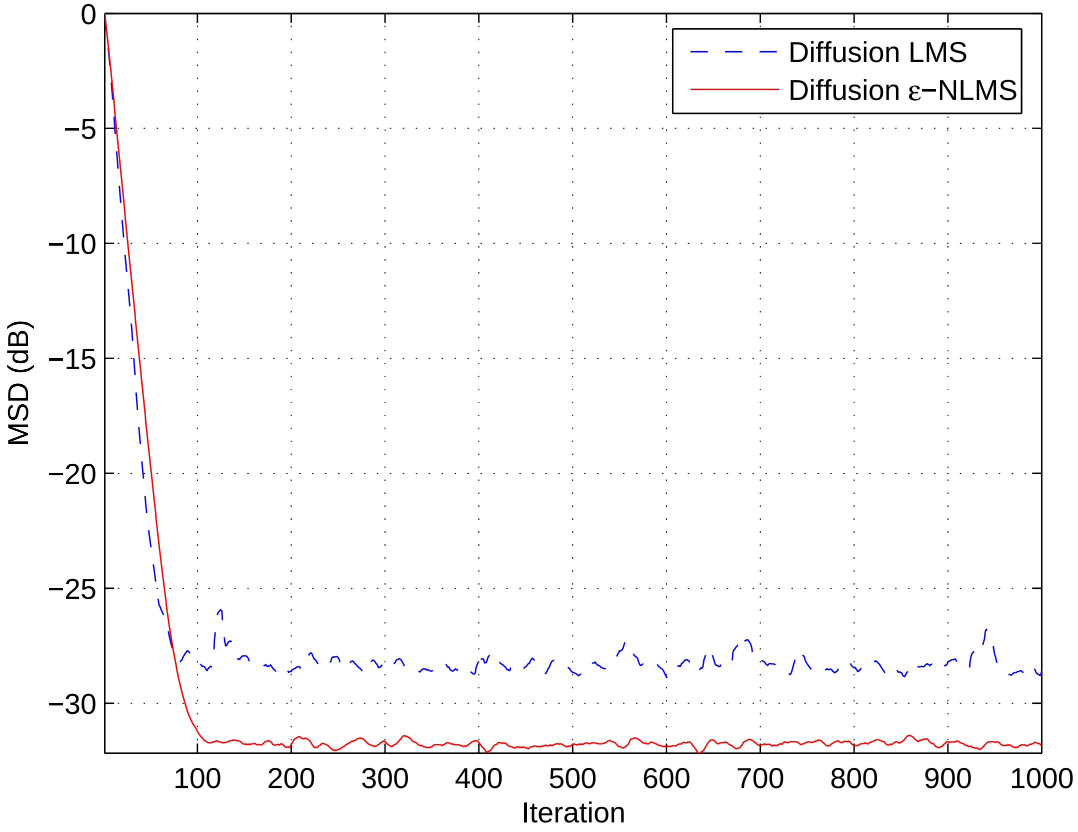


Figure 3.4: Global MSD of the diffusion LMS and diffusion ε-NLMS algorithm for $\alpha = 0.6$.

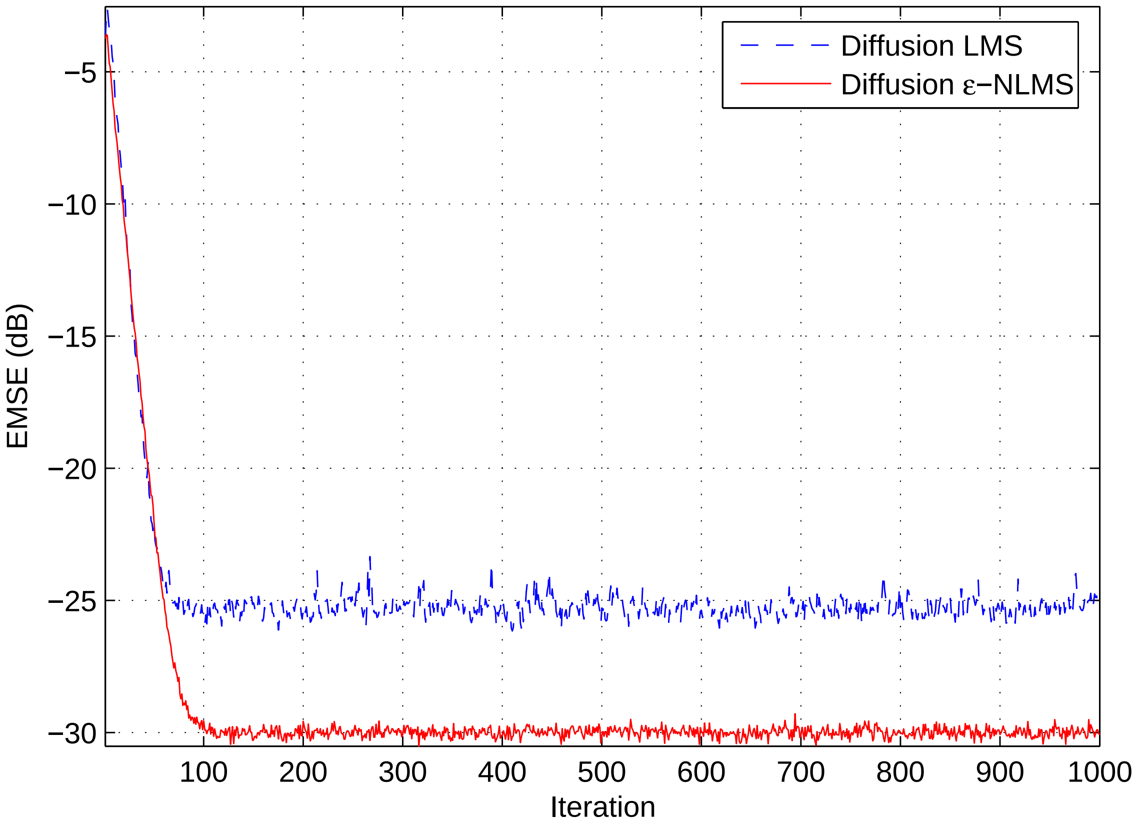


Figure 3.5: Global EMSE of the diffusion LMS and diffusion ε-NLMS algorithm for $\alpha = 0.6$.

3.4.2 Comparison of Theoretical and Simulation Results for Transient Analysis

In this section, we will try to see if the theoretical findings pertaining to the transient analysis of the diffusion ε-NLMS algorithm agree with the simulation results. A randomly generated normalized system weight vector with the number of taps set at 5.

Consider a network of 7 nodes each with tap length of 5. A non-white Gaussian input process with correlation factor α was fed into the diffusion ε-NLMS algorithm. The simulations were performed for Gaussian noise environment with noise variance values selected were 0.1, 0.01 and 0.001. The experiment is repeated for various correlation factors α. As we see from the figures, there is a good match between theory and simulation results.

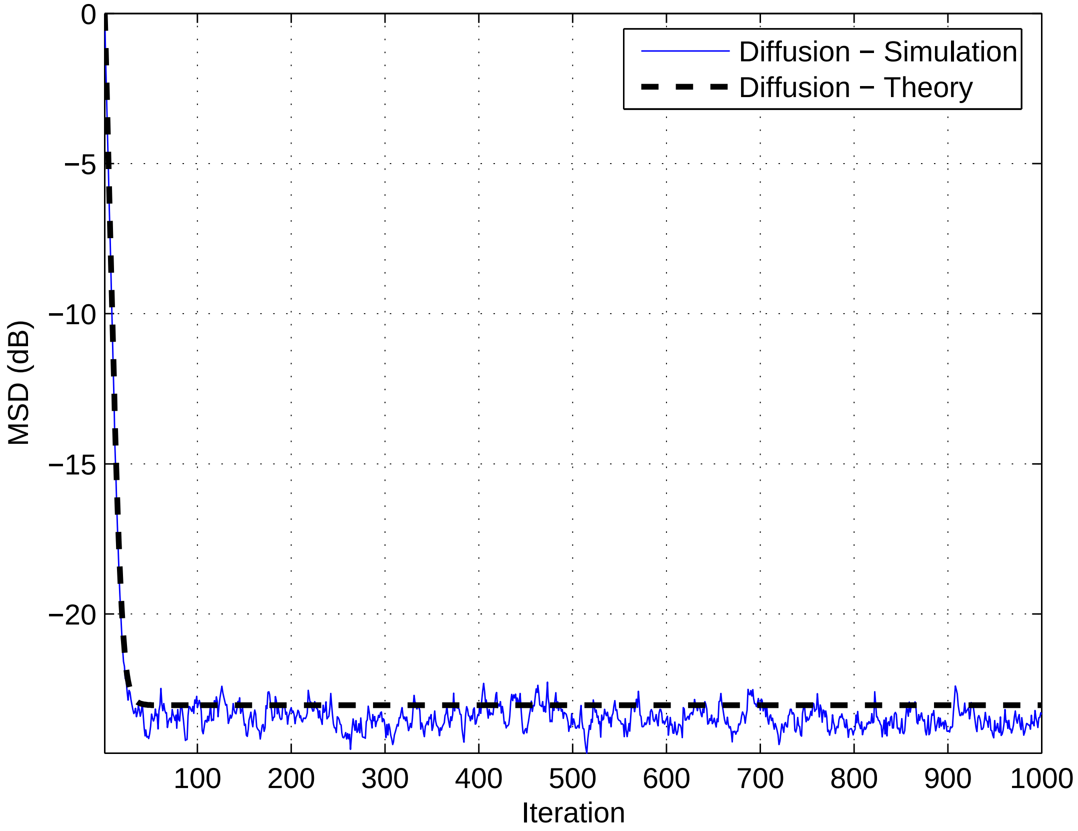


Figure 3.6: Global MSD of the diffusion ε-NLMS algorithm for $\alpha = 0.1$ and Noise Variance 0.1.

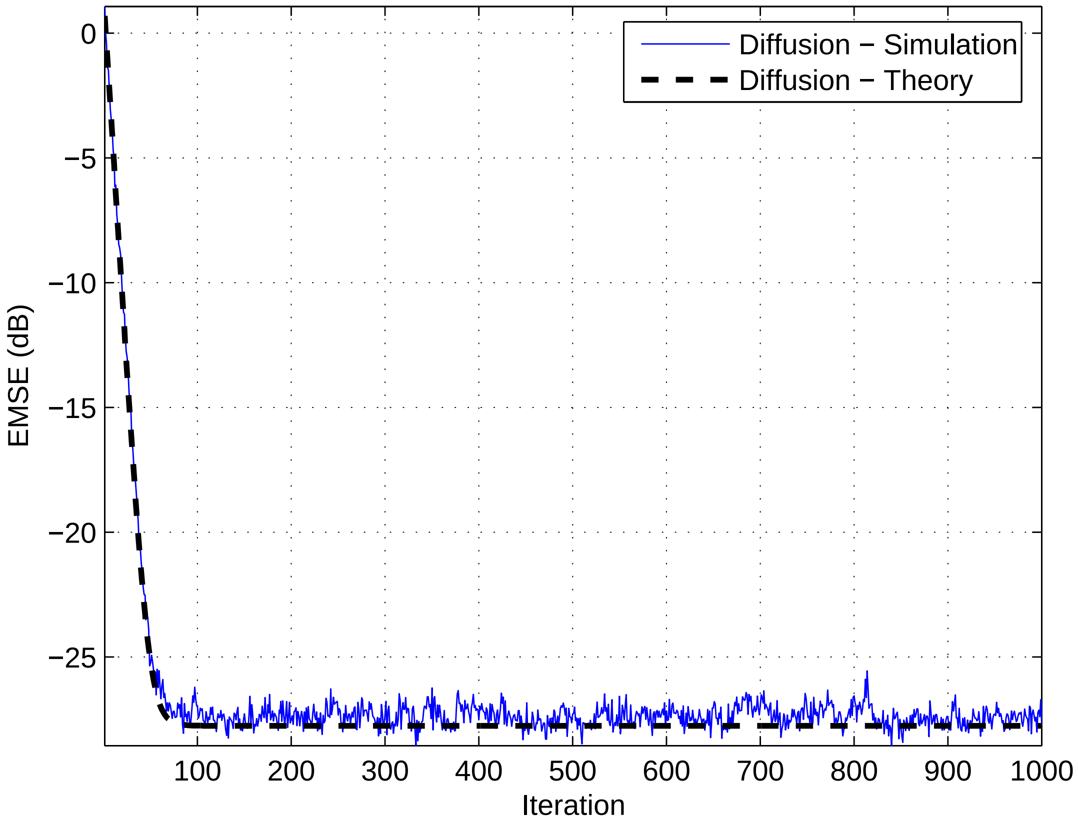


Figure 3.7: Global EMSE of the diffusion ε-NLMS algorithm for $\alpha = 0.1$ and Noise Variance 0.1.

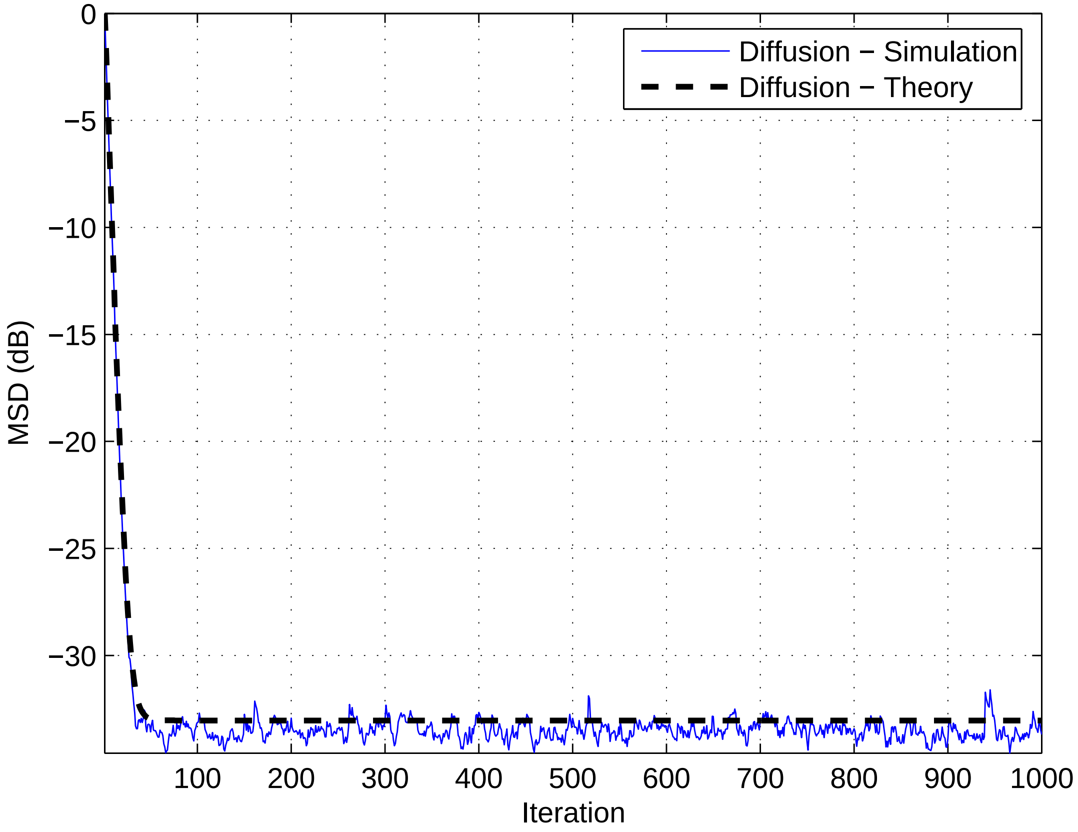


Figure 3.8: Global MSD of the diffusion ε-NLMS algorithm for $\alpha = 0.1$ and Noise Variance 0.01.

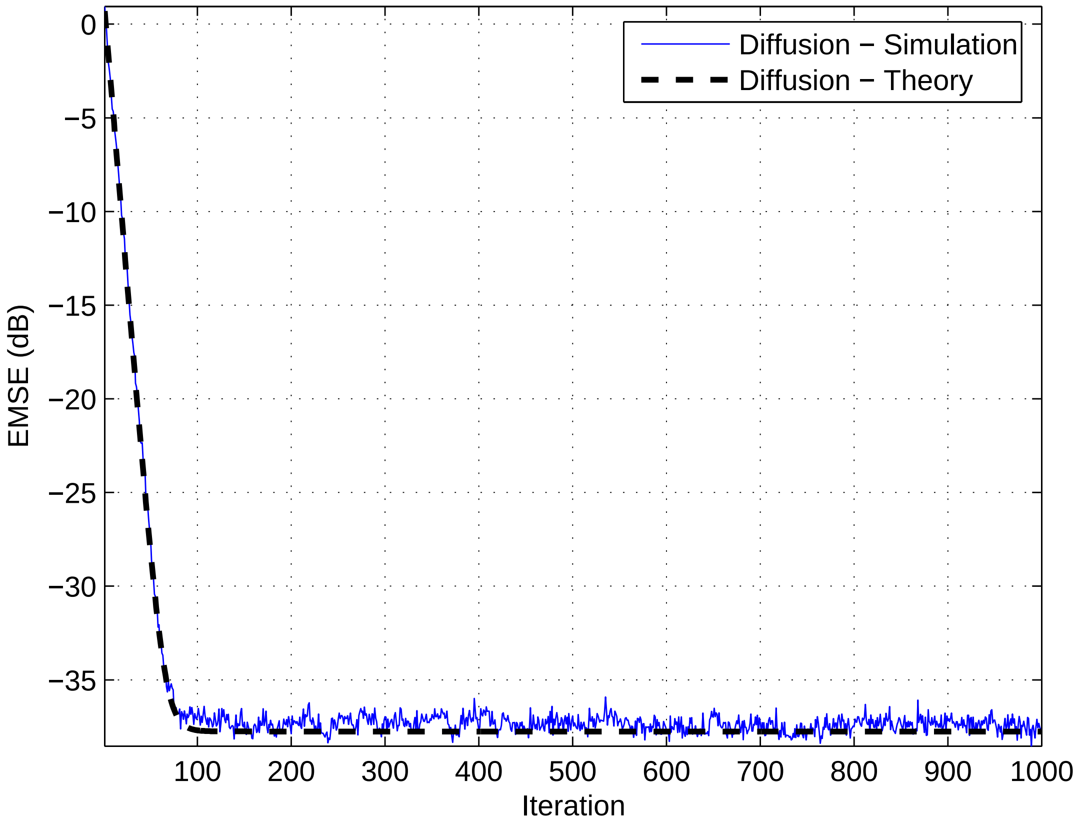


Figure 3.9: Global EMSE of the diffusion ε-NLMS algorithm for $\alpha = 0.1$ and Noise Variance 0.01.

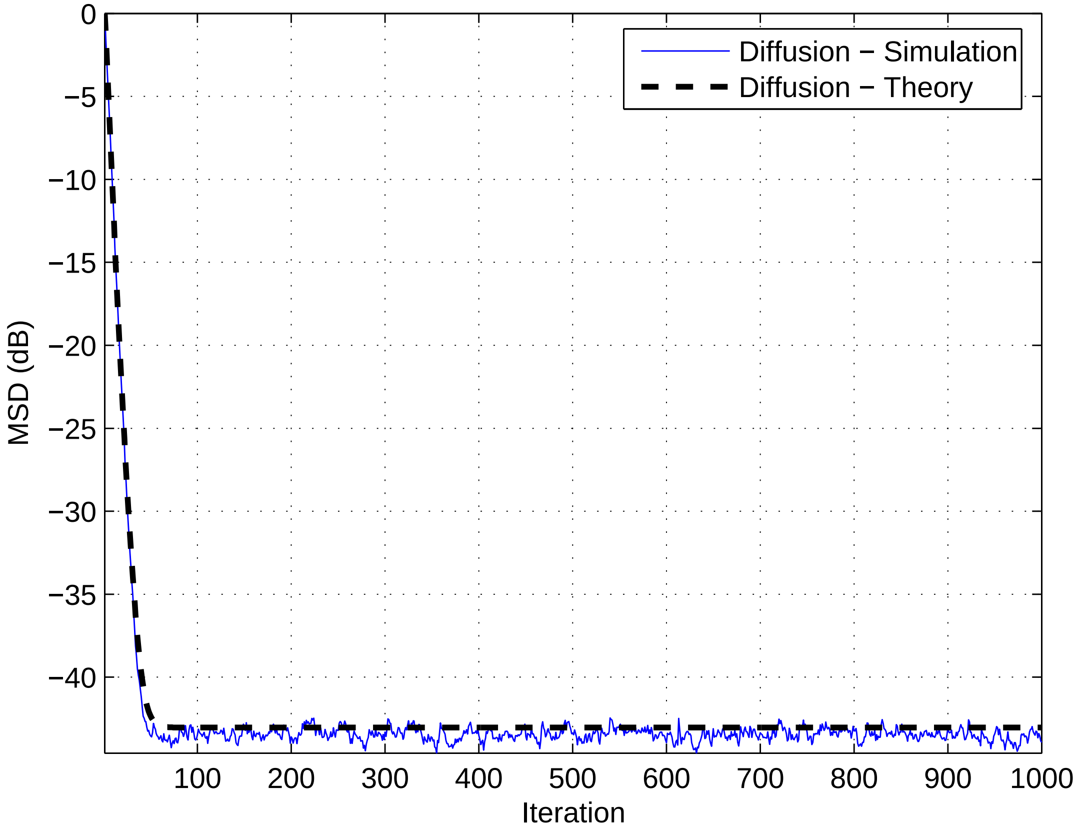


Figure 3.10: Global MSD of the diffusion ε-NLMS algorithm for $\alpha = 0.1$ and Noise Variance 0.001.

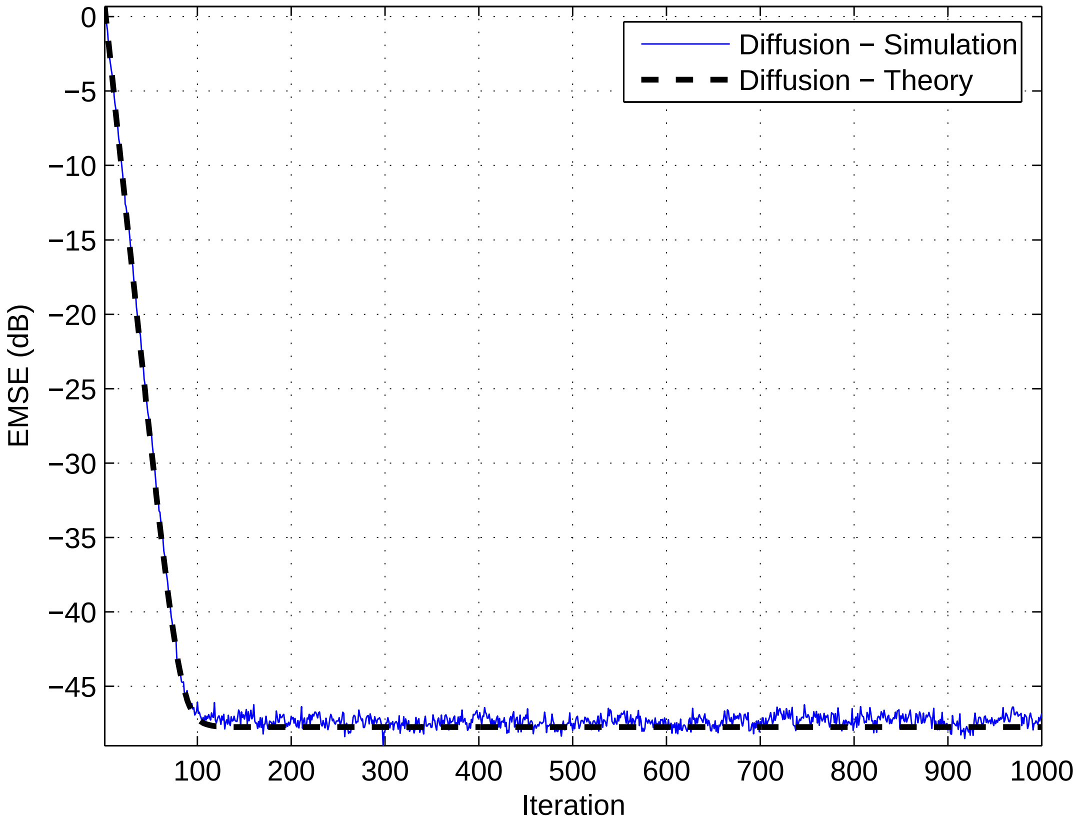


Figure 3.11: Global EMSE of the diffusion ε-NLMS algorithm for $\alpha = 0.1$ and Noise Variance 0.001.

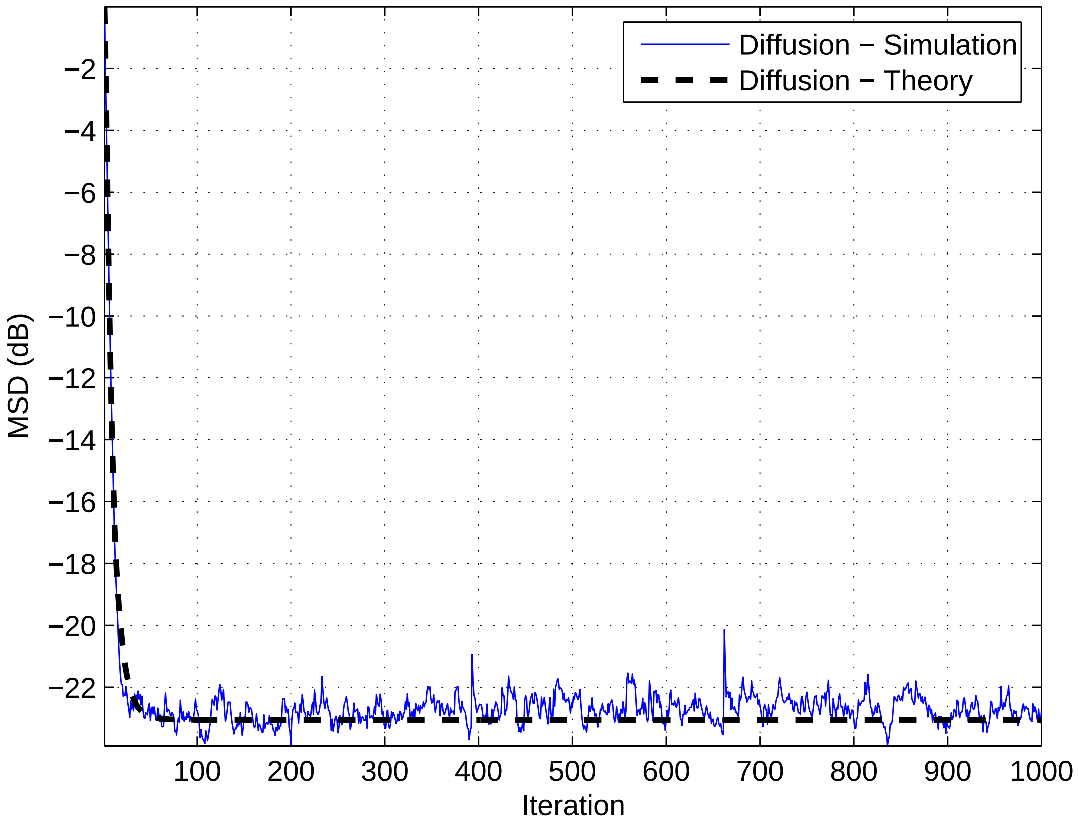


Figure 3.12: Global MSD of the diffusion ε-NLMS algorithm for $\alpha = 0.6$ and Noise Variance 0.1.

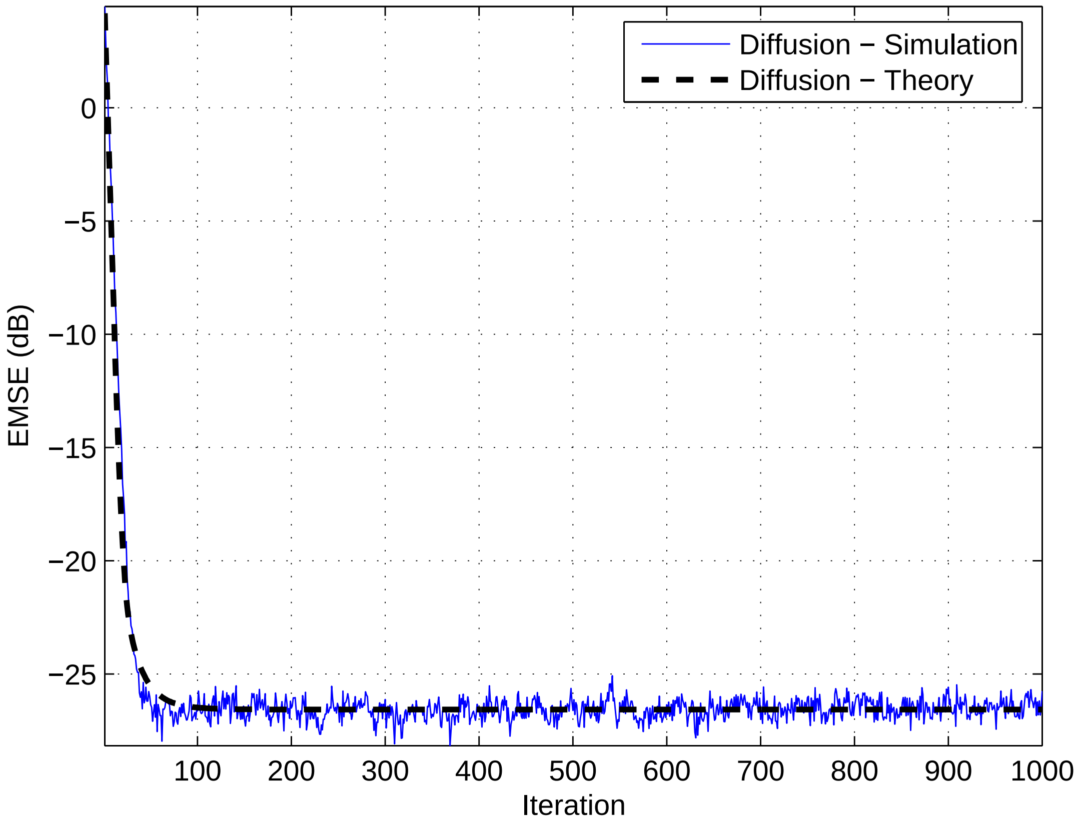


Figure 3.13: Global EMSE of the diffusion ε-NLMS algorithm for $\alpha = 0.6$ and Noise Variance 0.1.

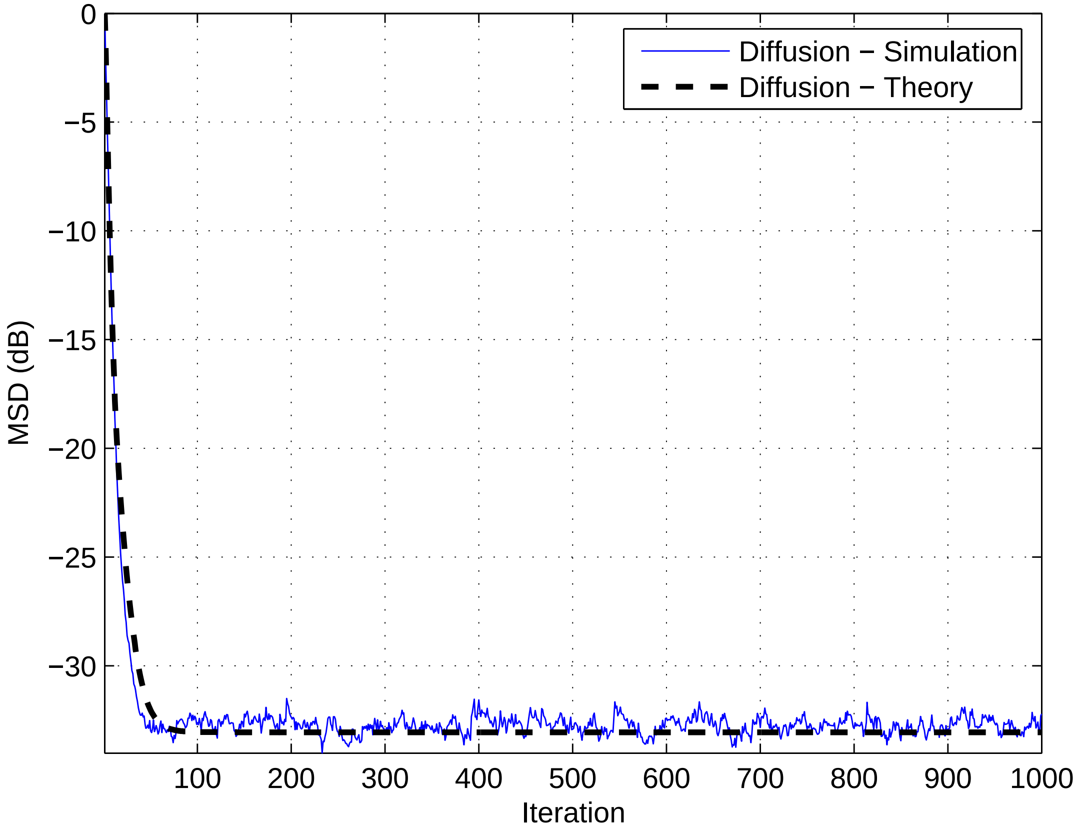


Figure 3.14: Global MSD of the diffusion ε-NLMS algorithm for $\alpha = 0.6$ and Noise Variance 0.01.

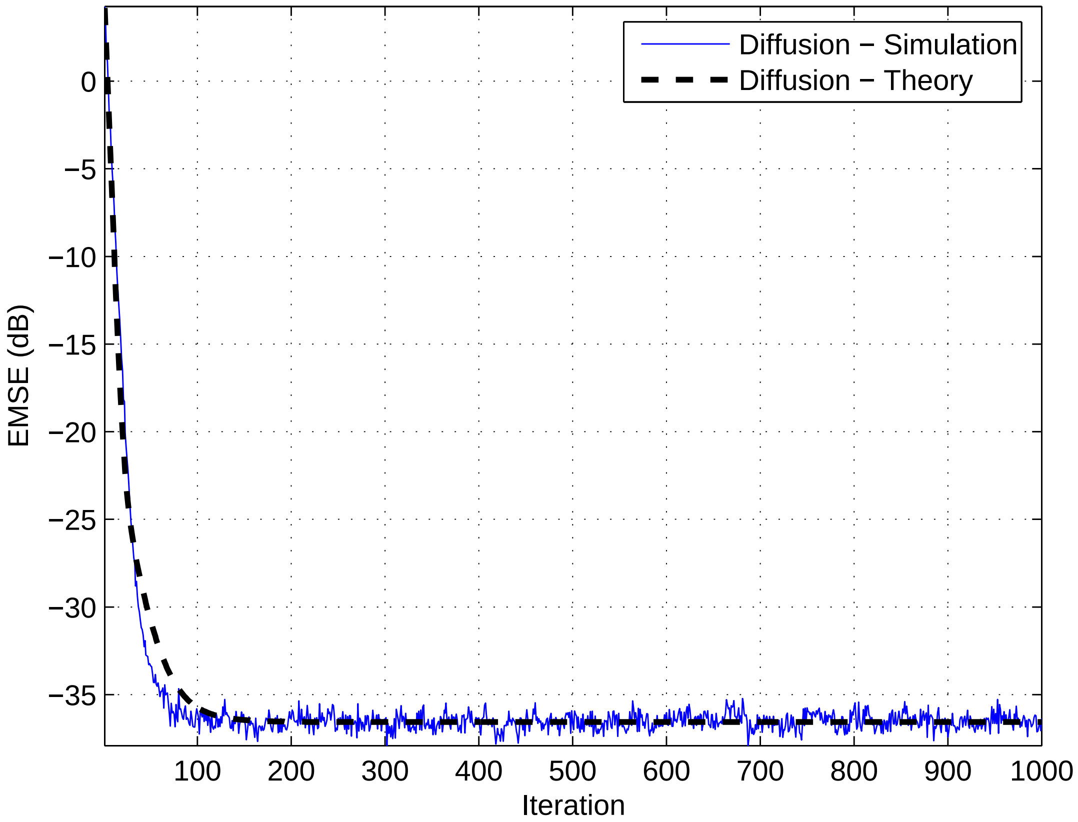


Figure 3.15: Global EMSE of the diffusion ε-NLMS algorithm for $\alpha = 0.6$ and Noise Variance 0.01.

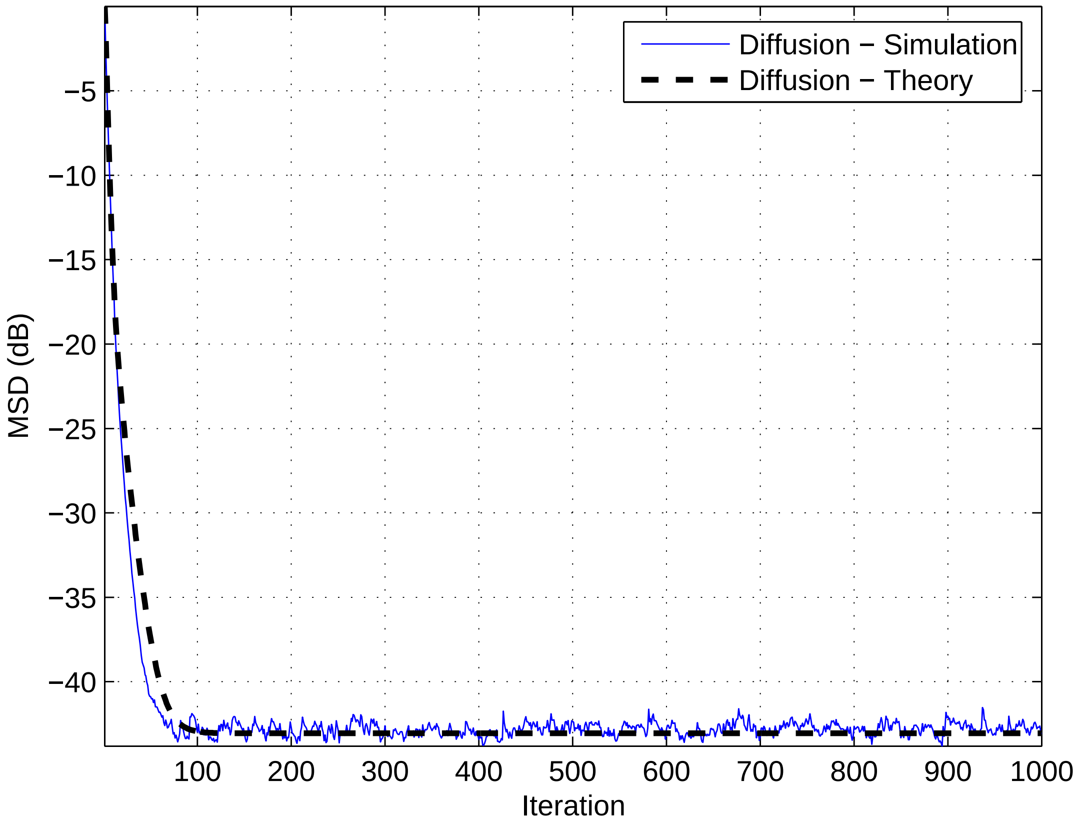


Figure 3.16: Global MSD of the diffusion ε-NLMS algorithm for $\alpha = 0.6$ and Noise Variance 0.001.

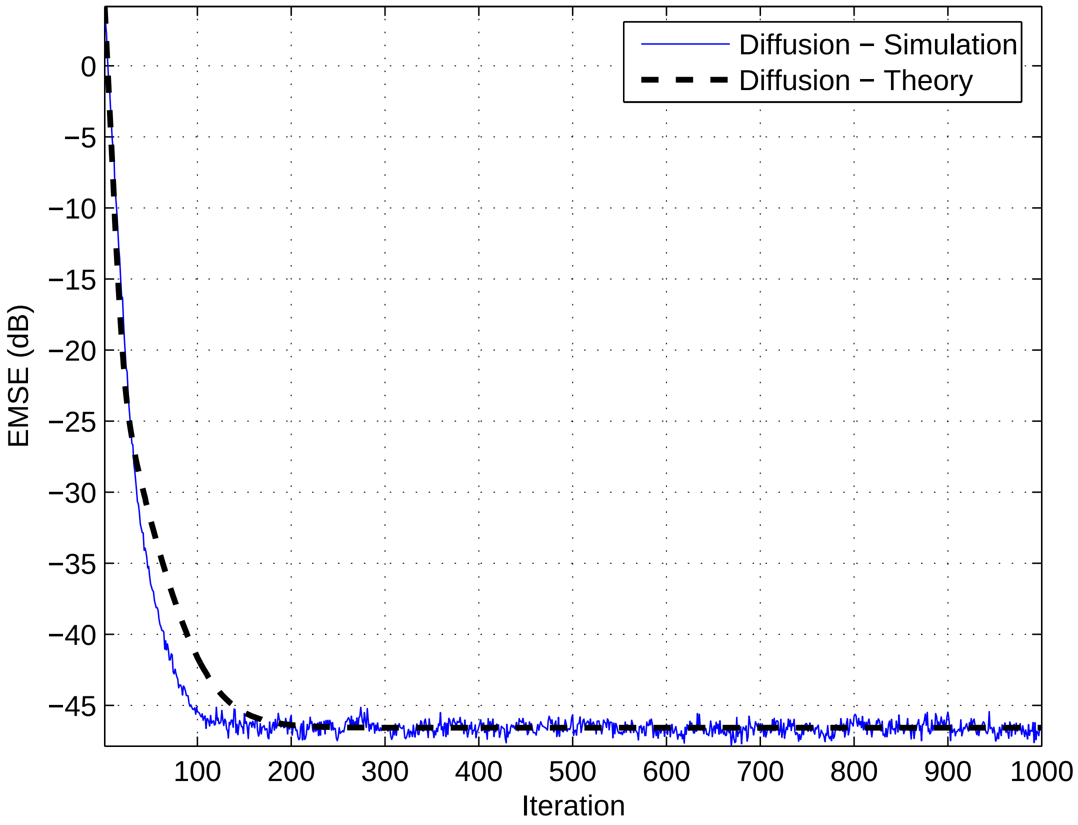


Figure 3.17: Global EMSE of the diffusion ε-NLMS algorithm for $\alpha = 0.6$ and Noise Variance 0.001.

Above figures shows good match between theoretical and simulation results in mean square deviation and excess mean square error sense for diffusion ε-NLMS algorithm for fixed correlation factor α for all N nodes. However, we may also need to consider the scenario when the correlation factor α of input Gaussian regressor varies from one node to another. Fig. 3.18 shows the network statistical settings for which inferences is drawn, which also shows good match between theory and simulation results.

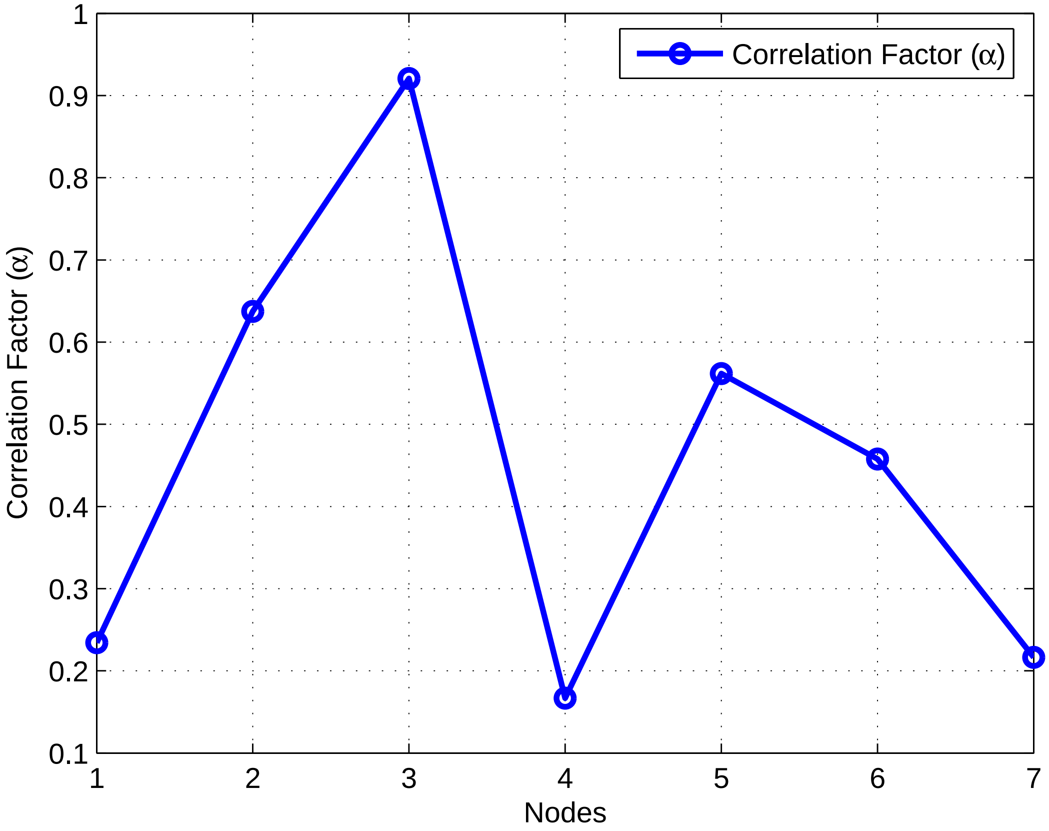


Figure 3.18: Network statistical settings.

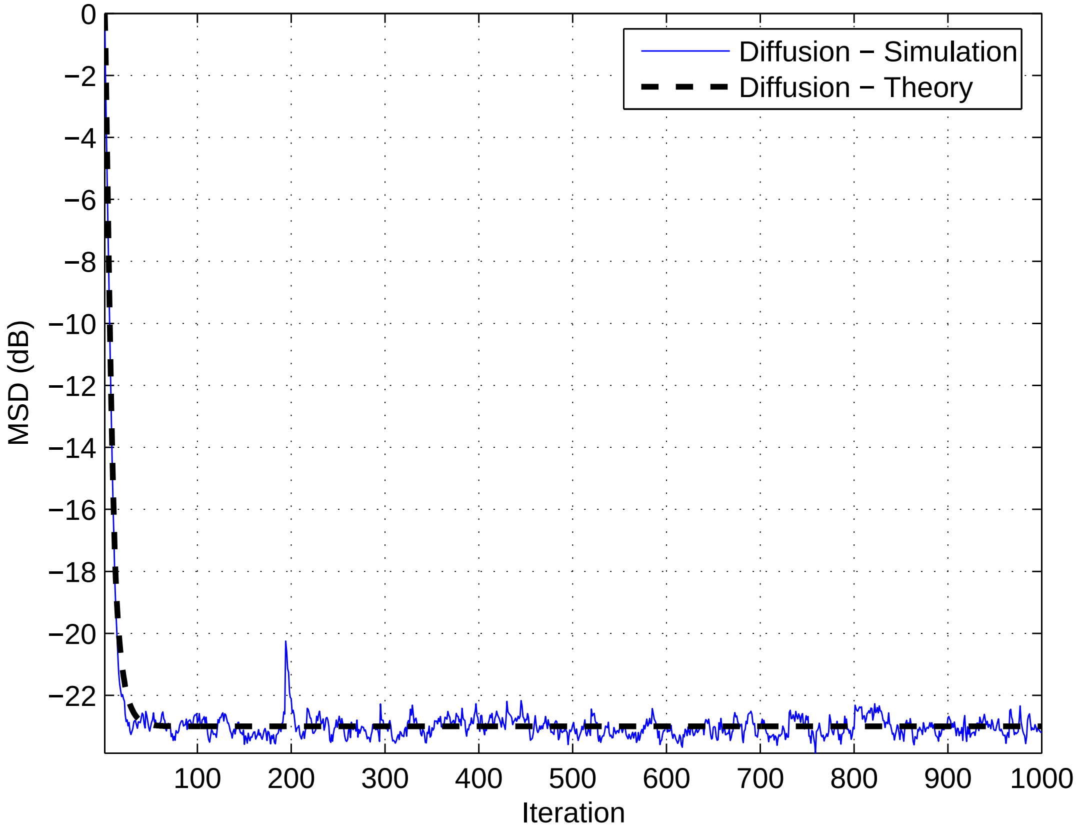


Figure 3.19: Global MSD of the diffusion ε-NLMS algorithm for Noise Variance 0.1.

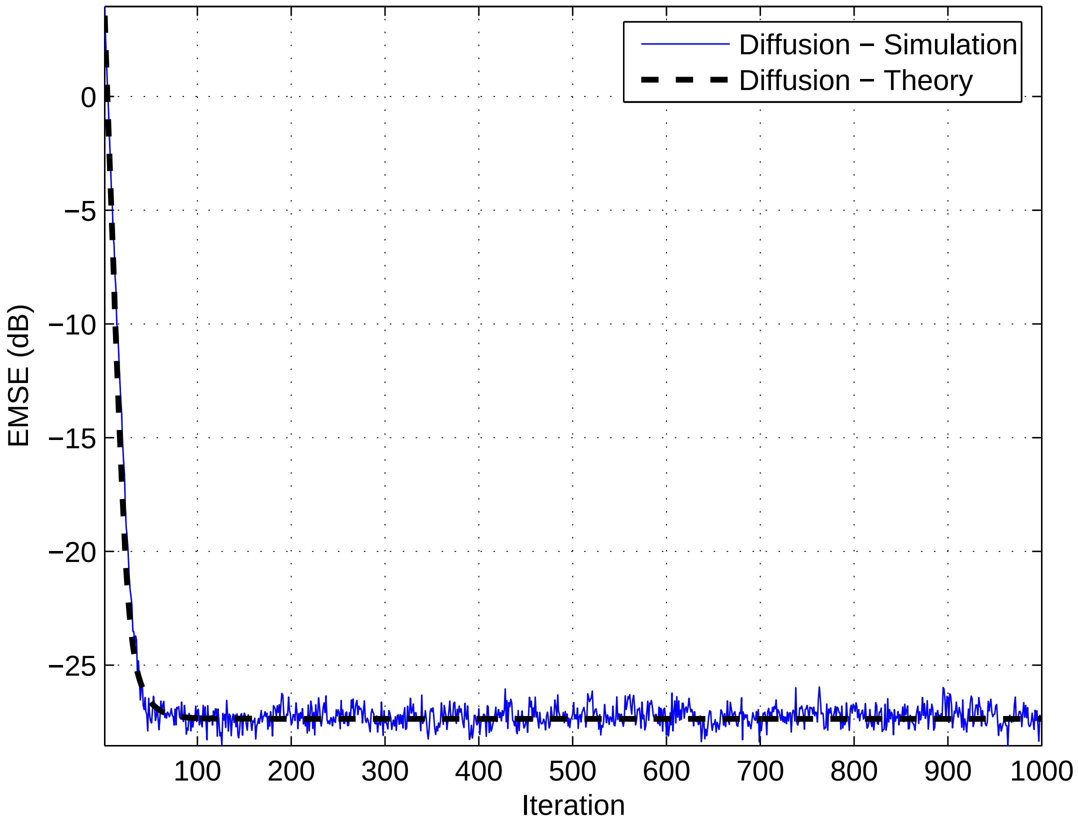


Figure 3.20: Global EMSE of the diffusion ε-NLMS algorithm for Noise Variance 0.1.

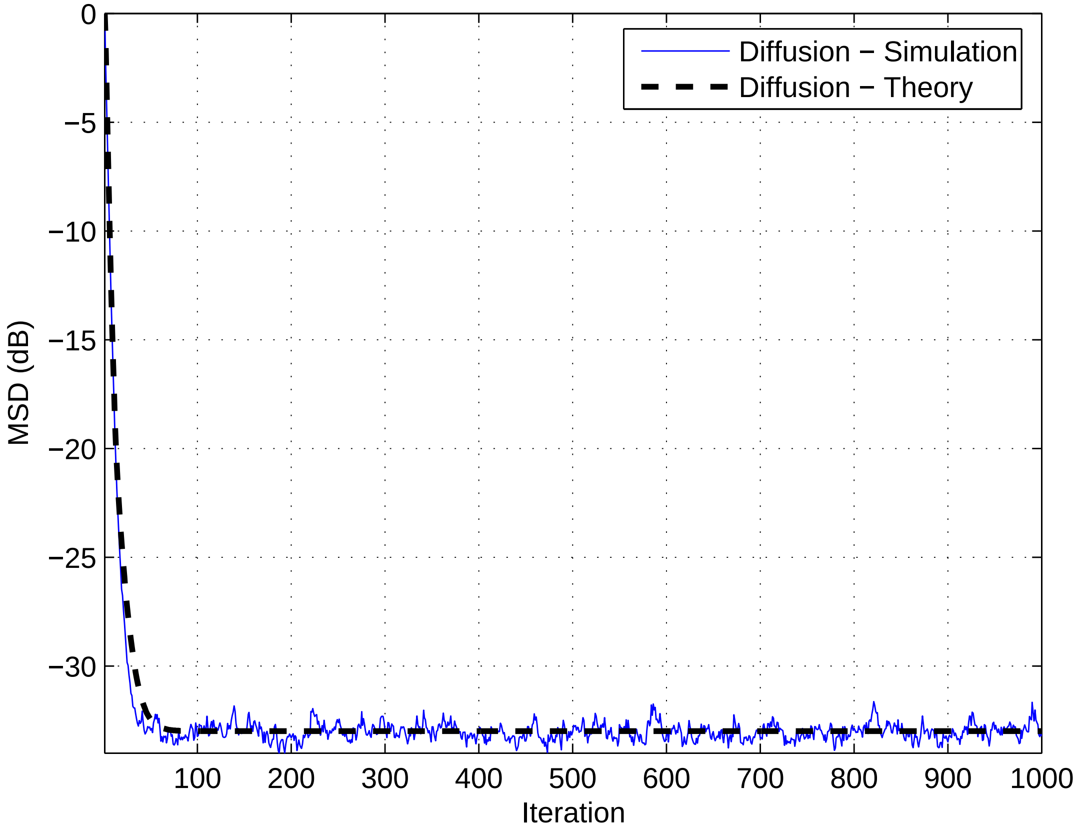


Figure 3.21: Global MSD of the diffusion ε-NLMS algorithm for Noise Variance 0.01.

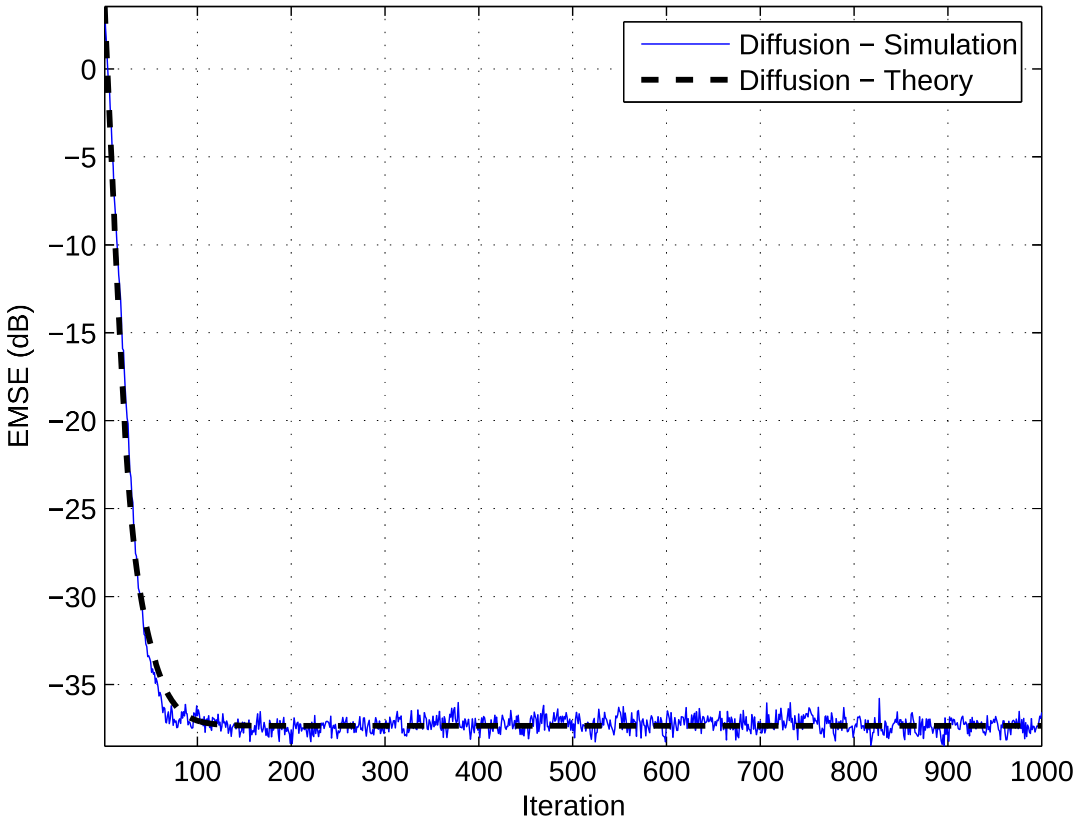


Figure 3.22: Global EMSE of the diffusion ε-NLMS algorithm for Noise Variance 0.01.

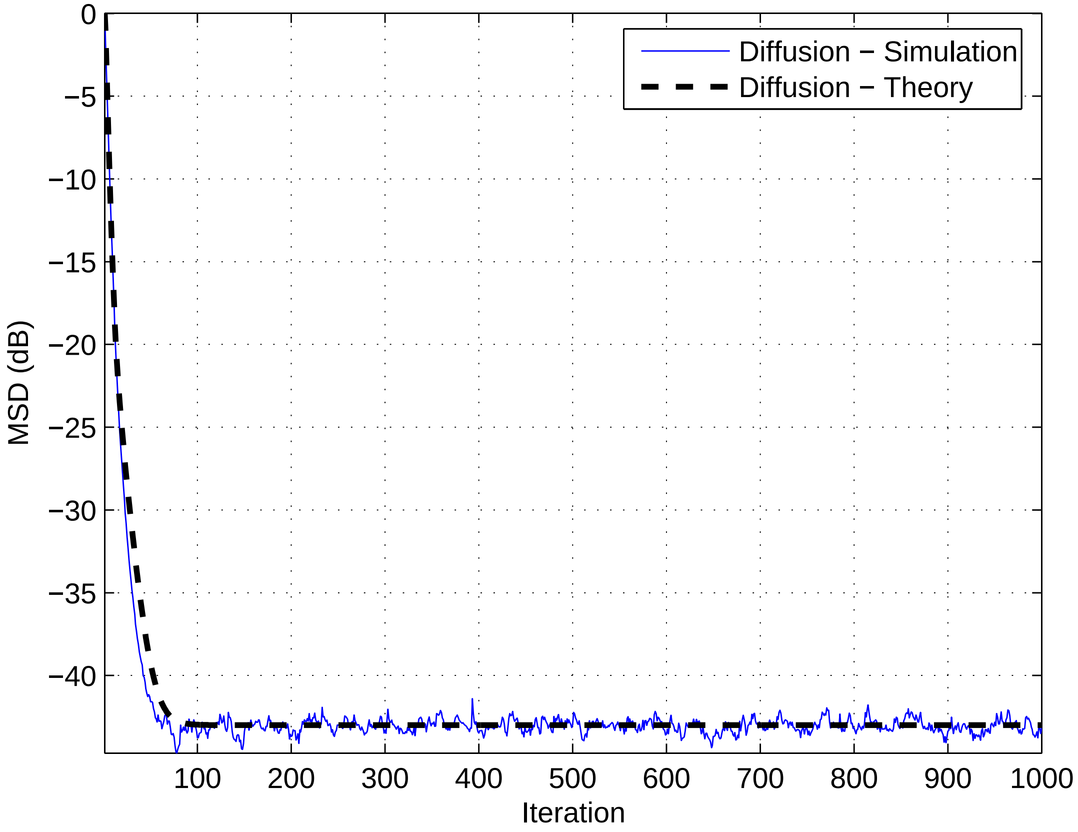


Figure 3.23: Global MSD of the diffusion ε-NLMS algorithm for Noise Variance 0.001.

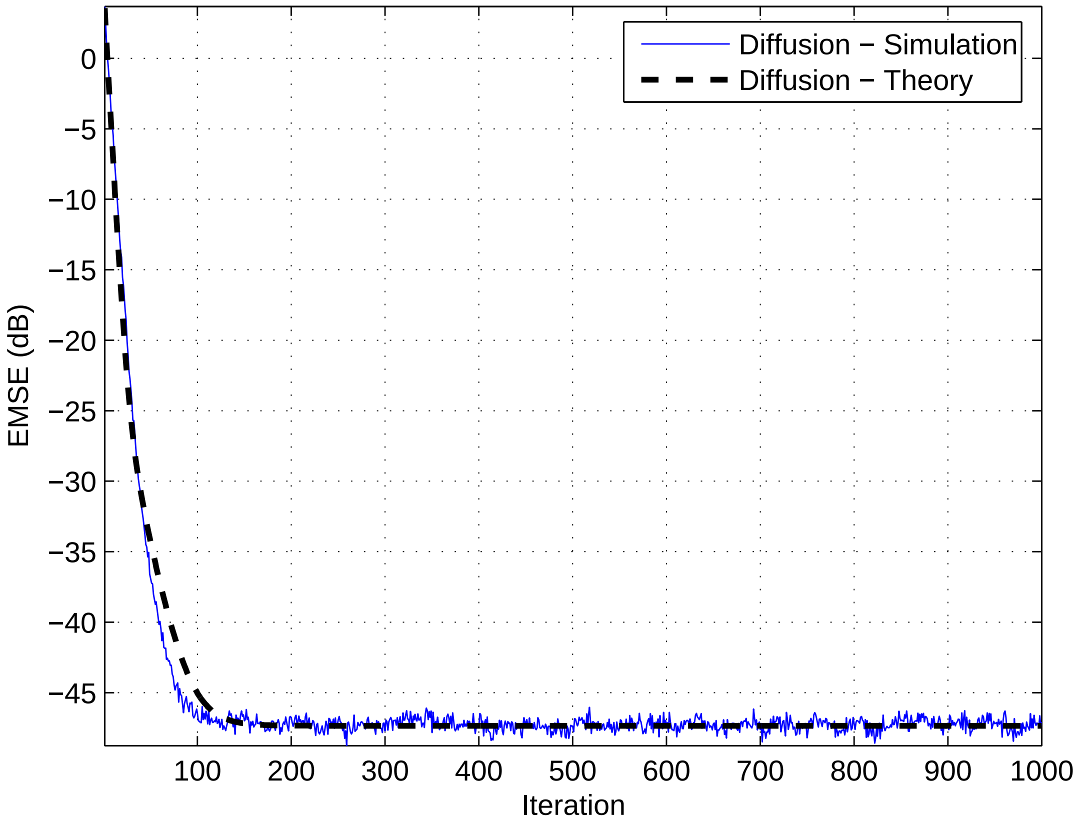


Figure 3.24: Global EMSE of the diffusion ε-NLMS algorithm for Noise Variance 0.001.

CHAPTER 4

STEADY STATE ANALYSIS OF DIFFUSION NORMALIZED LMS ALGORITHM

4.1 Introduction

In this chapter, steady-state analysis of the diffusion ε-NLMS adaptive algorithm is carried out. We will be using the same assumptions used in the previous chapters. Using steady-state analysis the performance of learning mechanism of the adaptive filter in terms of how well adaptive filter can learn the underlying signal statistics given sufficient time can be studied. In this chapter, we will derive mean square deviation (MSD) and excess mean square error (EMSE) expressions for steady-state which will describe the global network performance.

4.2 Steady State Analysis

Let us define the global mean square deviation and global excess mean square error as an average of the global quantities $\mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{i}\right\|^{2}$ and $\mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{i}\right\|_{\boldsymbol{\Lambda}}^{2}$ respectively. So we get,

$$MSD = (1/N)\,\mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{i}\right\|^{2} \tag{4.1a}$$

$$EMSE = (1/N)\,\mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{i}\right\|_{\boldsymbol{\Lambda}}^{2} \tag{4.1b}$$

Now in order to evaluate $\mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{i}\right\|^{2}$ and $\mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{i}\right\|_{\boldsymbol{\Lambda}}^{2}$, let us use equation (3.50) in steady state ($i \to \infty$), which leads to

$$\mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{\infty}\right\|_{\bar{\boldsymbol{\sigma}}}^{2} = \mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{\infty}\right\|_{\bar{\boldsymbol{F}}\bar{\boldsymbol{\sigma}}}^{2} + \boldsymbol{b}^{T}\bar{\boldsymbol{\sigma}} \tag{4.2}$$

This equation gives the steady-state performance measure for the entire network, which can be rewritten more compactly as,

$$\mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{\infty}\right\|_{(\mathbf{I}-\bar{\boldsymbol{F}})\bar{\boldsymbol{\sigma}}}^{2} = \boldsymbol{b}^{T}\bar{\boldsymbol{\sigma}} \tag{4.3}$$

Comparing (4.1) and (4.3), the possible values for $\bar{\boldsymbol{\sigma}}$ can be derived as,

$$(\mathbf{I} - \bar{\boldsymbol{F}})\bar{\boldsymbol{\sigma}}_{MSD} = \boldsymbol{q} \tag{4.4a}$$

$$(\mathbf{I} - \bar{\boldsymbol{F}})\bar{\boldsymbol{\sigma}}_{EMSE} = \boldsymbol{\lambda} \tag{4.4b}$$

where $\boldsymbol{q} = bvec\{\mathbf{I_{NM}}\}$ and $\boldsymbol{\lambda} = bvec\{\boldsymbol{\Lambda}\}$. Above equation (4.4) can be rewritten as,

$$\bar{\boldsymbol{\sigma}}_{MSD} = (\mathbf{I} - \bar{\boldsymbol{F}})^{-1}\boldsymbol{q} \tag{4.5a}$$

$$\bar{\boldsymbol{\sigma}}_{EMSE} = (\mathbf{I} - \bar{\boldsymbol{F}})^{-1}\boldsymbol{\lambda} \tag{4.5b}$$

Substituting (4.5) in (4.3) leads to,

$$\mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{\infty}\right\|^2_{(\mathbf{I}-\bar{\boldsymbol{F}})\bar{\boldsymbol{\sigma}}} = \boldsymbol{b}^T(\mathbf{I} - \bar{\boldsymbol{F}})^{-1}\boldsymbol{q} \ \ (MSD) \tag{4.6a}$$

$$\mathrm{E}\left\|\bar{\boldsymbol{\psi}}^{\infty}\right\|^2_{(\mathbf{I}-\bar{\boldsymbol{F}})\bar{\boldsymbol{\sigma}}} = \boldsymbol{b}^T(\mathbf{I} - \bar{\boldsymbol{F}})^{-1}\boldsymbol{\lambda} \ \ (EMSE) \tag{4.6b}$$

Now again substitute (4.6) in (4.1) which gives,

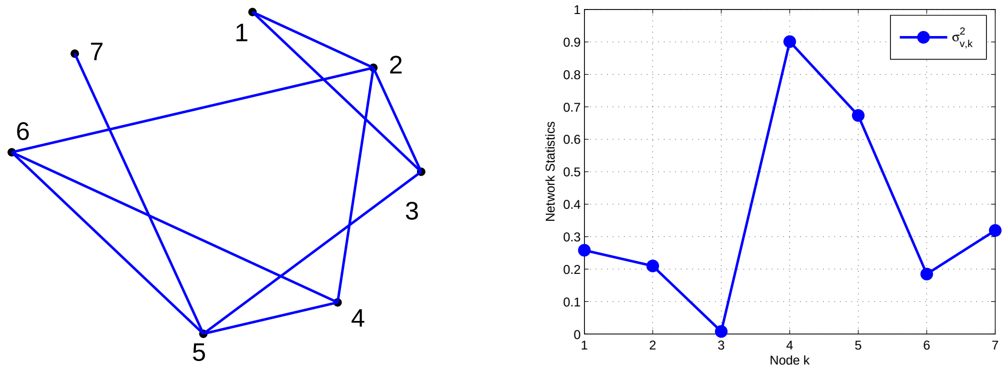
$$MSD = (1/N)\, \boldsymbol{b}^T(\mathbf{I} - \bar{\boldsymbol{F}})^{-1}\boldsymbol{q} \tag{4.7a}$$

$$EMSE = (1/N)\, \boldsymbol{b}^T(\mathbf{I} - \bar{\boldsymbol{F}})^{-1}\boldsymbol{\lambda} \tag{4.7b}$$

Equation (4.7) defines the global network performance for cooperative strategy in steady-state for MSD and EMSE.

4.3 Simulation Results

In this section, the results of the computer simulations are presented which are made to investigate the performance behavior of the diffusion ε-NLMS algorithm. A number of simulation results are carried out to corroborate the theoretical finding. We will present a number of simulations which shows that there is a good match between the theoretical and simulation results of diffusion ε-NLMS algorithm. Figure. 4.1 defines the network topological structure and statistics of the adaptive network following diffusion protocol.



(a)

(b)

Figure 4.1: Network topology and statistics.

4.3.1 Comparison of Diffusion LMS and Diffusion ε-NLMS for Steady-state Analysis

In this section, the effectiveness of the proposed algorithm is illustrated by comparing the steady-state performance for individual nodes of diffusion LMS and diffusion ε-NLMS algorithms.

A non-white Gaussian input process with correlation factor α was fed into the diffusion LMS and diffusion ε-NLMS algorithms. The simulations were performed for Gaussian noise environment with noise variance 0.01. The experiment is repeated for varying correlation factors α.

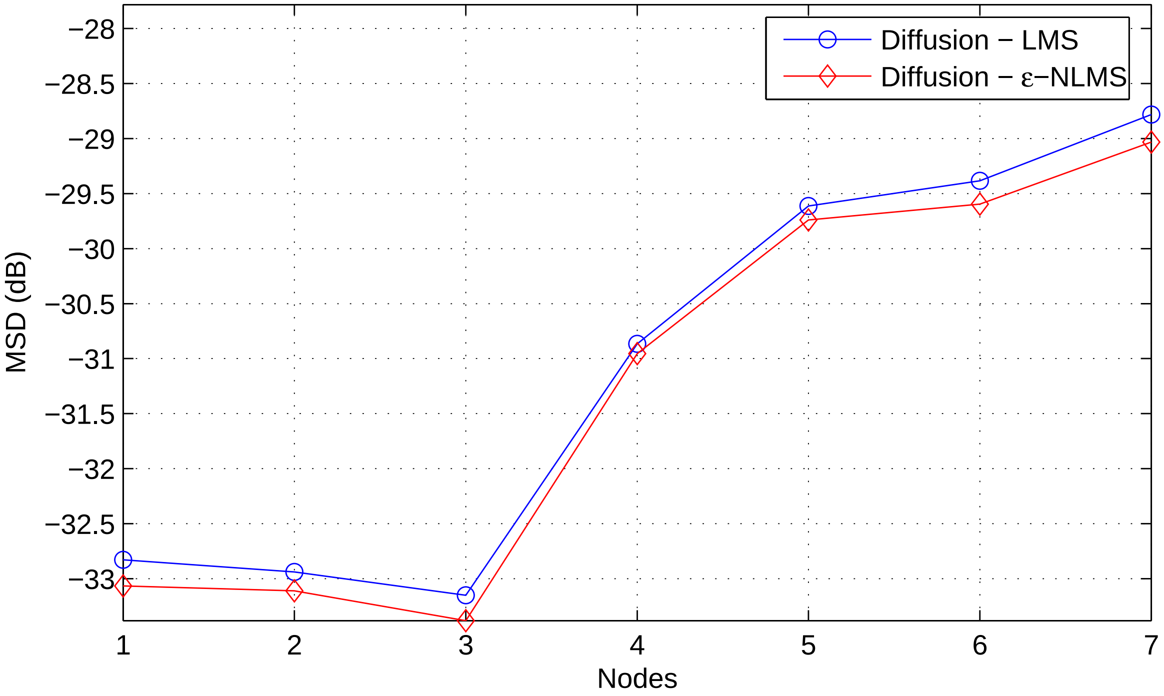


Figure 4.2: Local MSD performance for $\alpha = 0.1$ and Noise Variance 0.01.

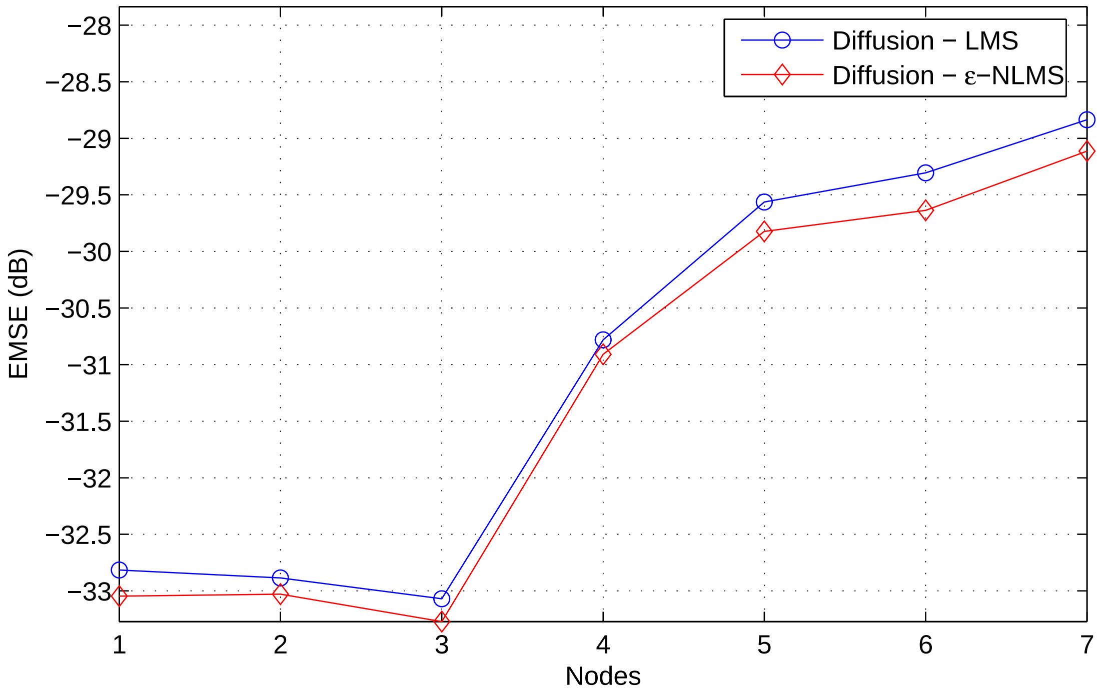


Figure 4.3: Local EMSE performance for $\alpha = 0.1$ and Noise Variance 0.01.

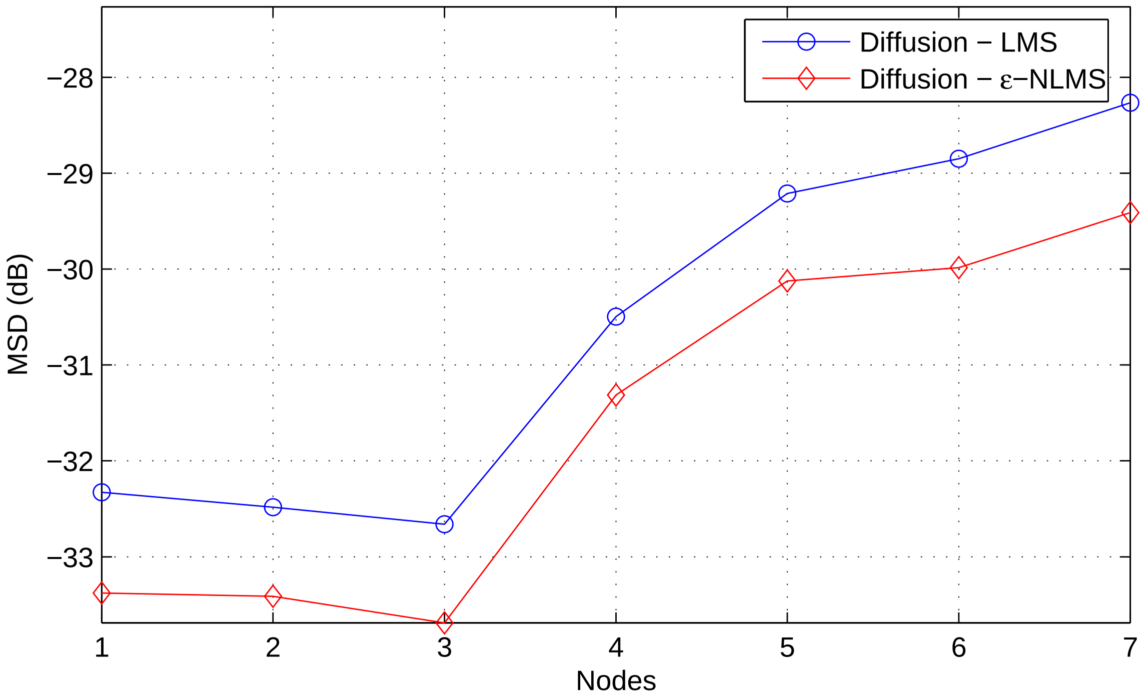


Figure 4.4: Local MSD performance for $\alpha = 0.4$ and Noise Variance 0.01.

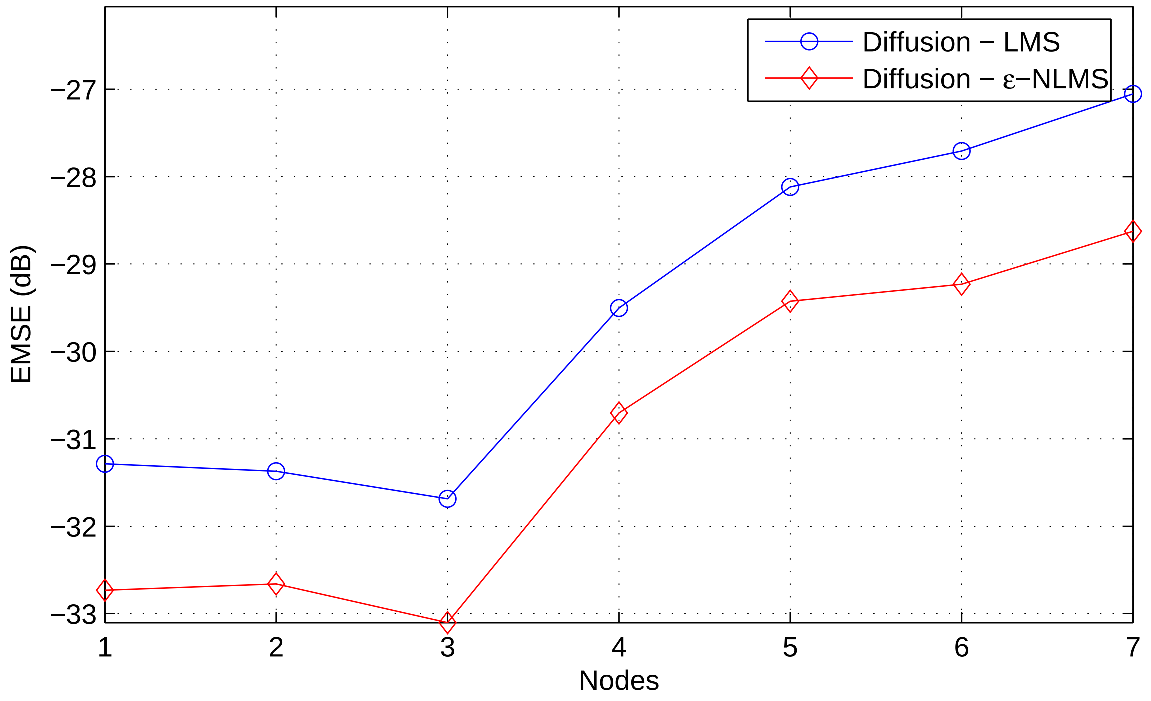


Figure 4.5: Local EMSE performance for $\alpha = 0.4$ and Noise Variance 0.01.

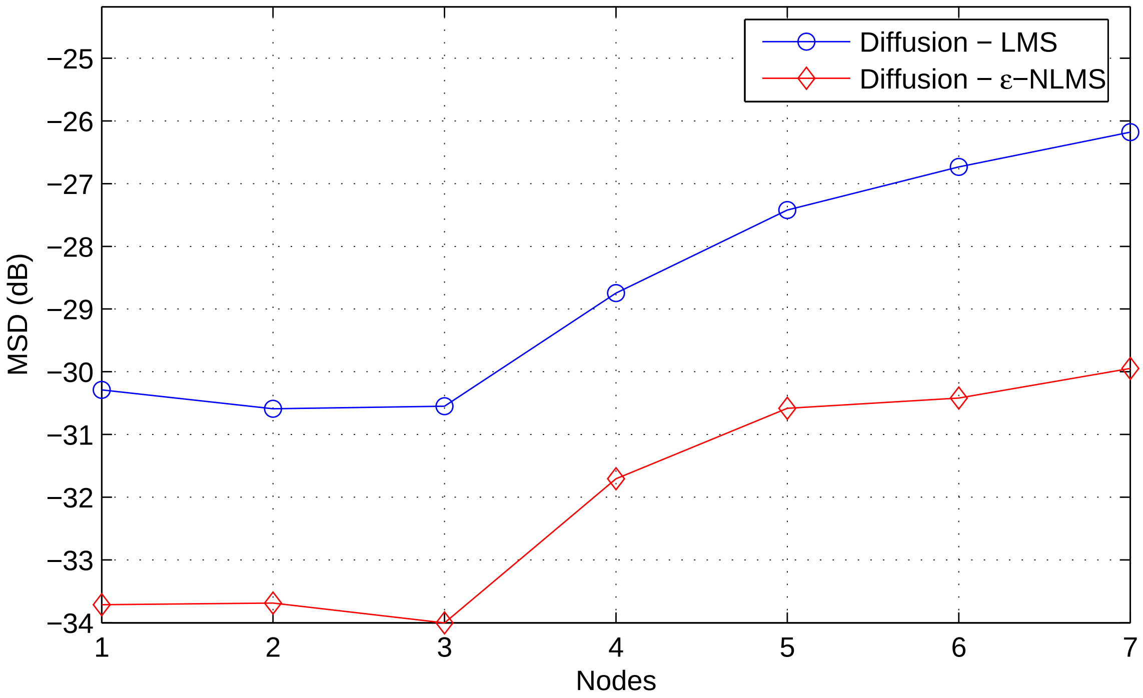


Figure 4.6: Local MSD performance for $\alpha = 0.6$ and Noise Variance 0.01.

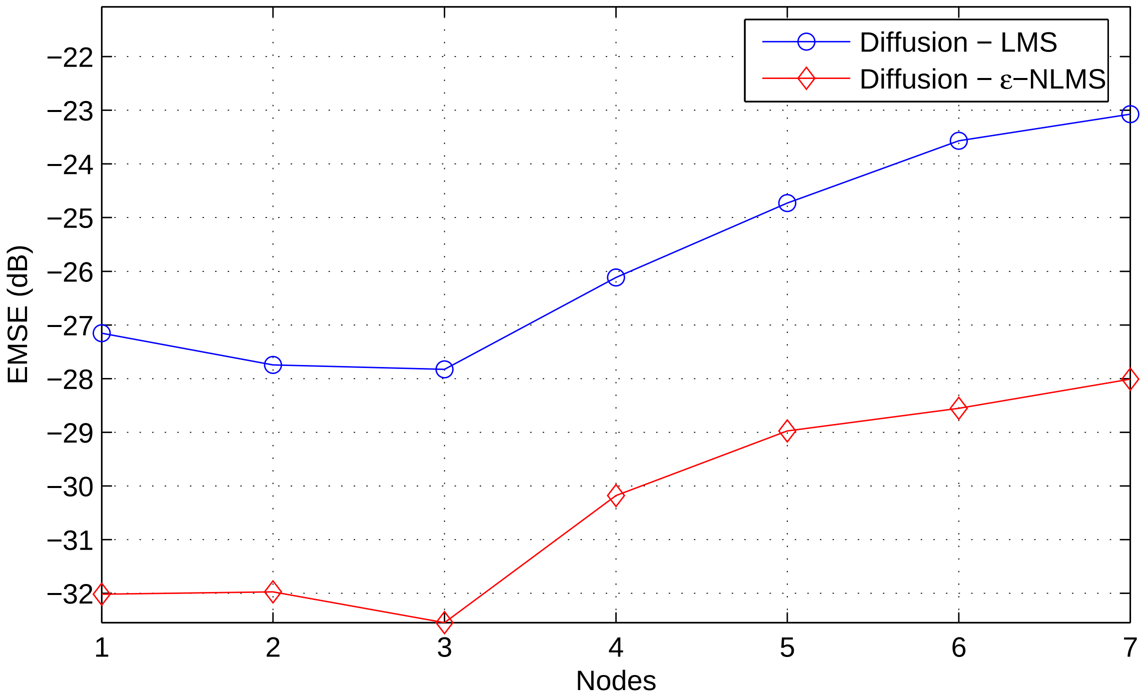


Figure 4.7: Local EMSE performance for $\alpha = 0.6$ and Noise Variance 0.01.

4.3.2 Comparison of Theoretical and Simulation Results for Steady-state Analysis

In this section, we will try to see if the theoretical findings pertaining to the steady-state analysis of the diffusion ε-NLMS algorithm agree with the simulation results. All other parameters are same as for the transient analysis.

A non-white Gaussian input process with correlation factor α was fed into the diffusion ε-NLMS algorithm. The simulations were performed for Gaussian noise environment with noise variance 0.01. The experiment is repeated for varying correlation factors α. We can see from the figures that, there is a good match between theoretical and simulation results.

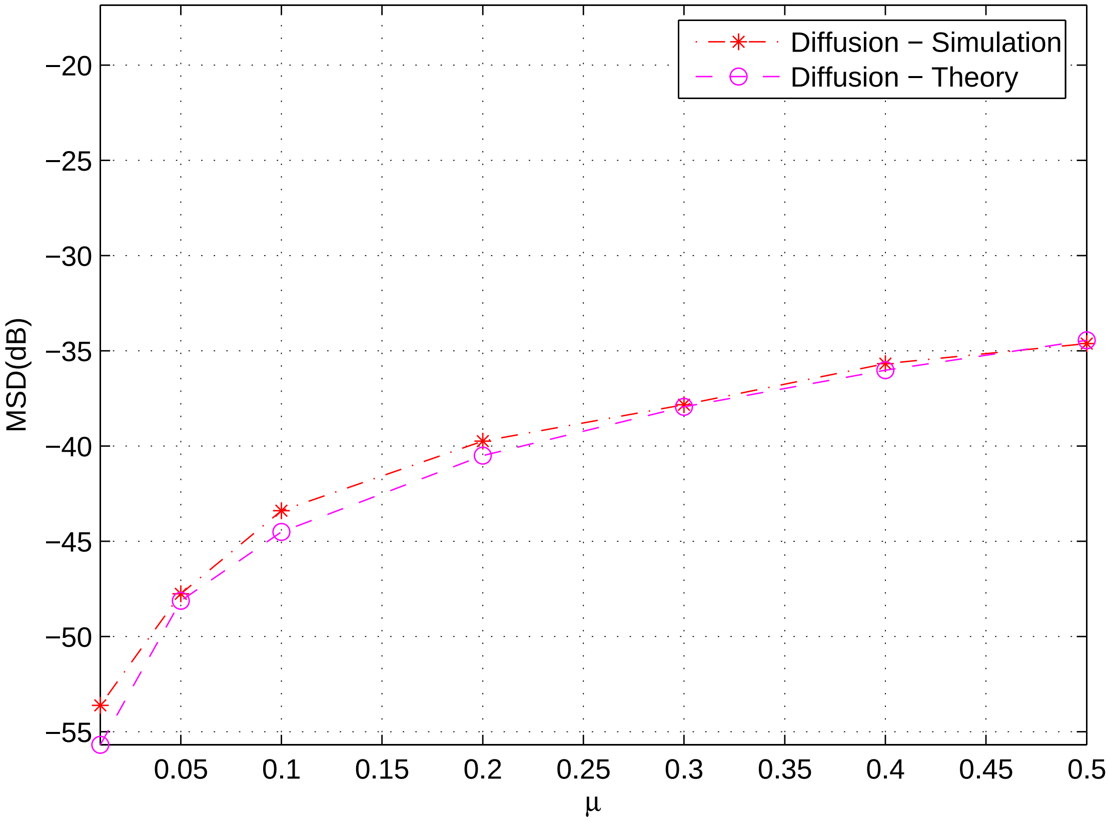


Figure 4.8: Global MSD of the diffusion ε-NLMS algorithm for $\alpha = 0.1$ and Noise Variance 0.01.

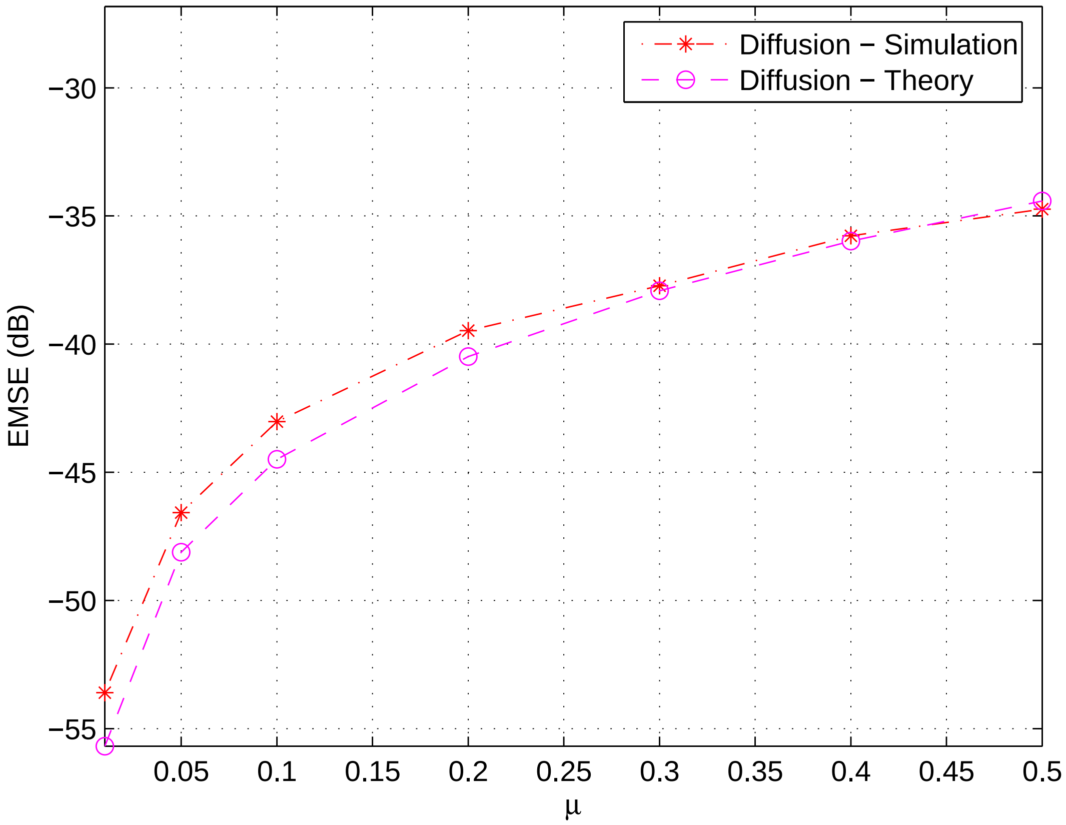


Figure 4.9: Global EMSE of the diffusion ε-NLMS algorithm for $\alpha = 0.1$ and Noise Variance 0.01.

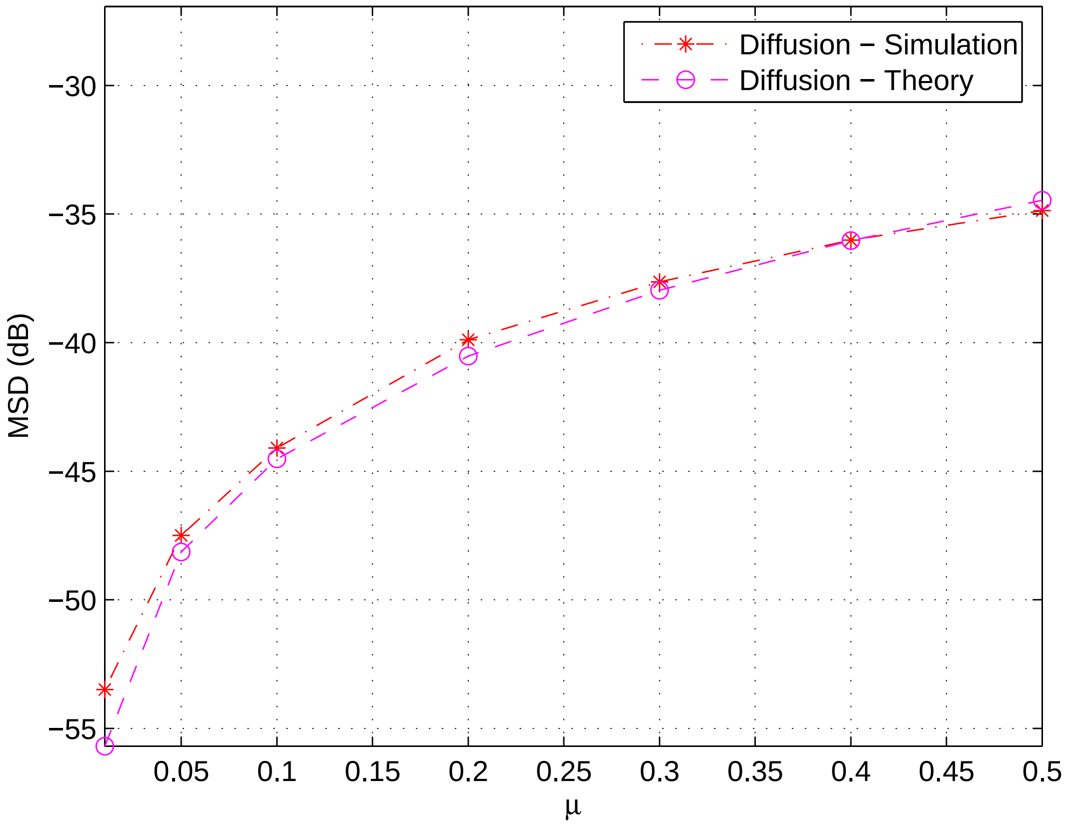


Figure 4.10: Global MSD of the diffusion ε-NLMS algorithm for $\alpha = 0.4$ and Noise Variance 0.01.

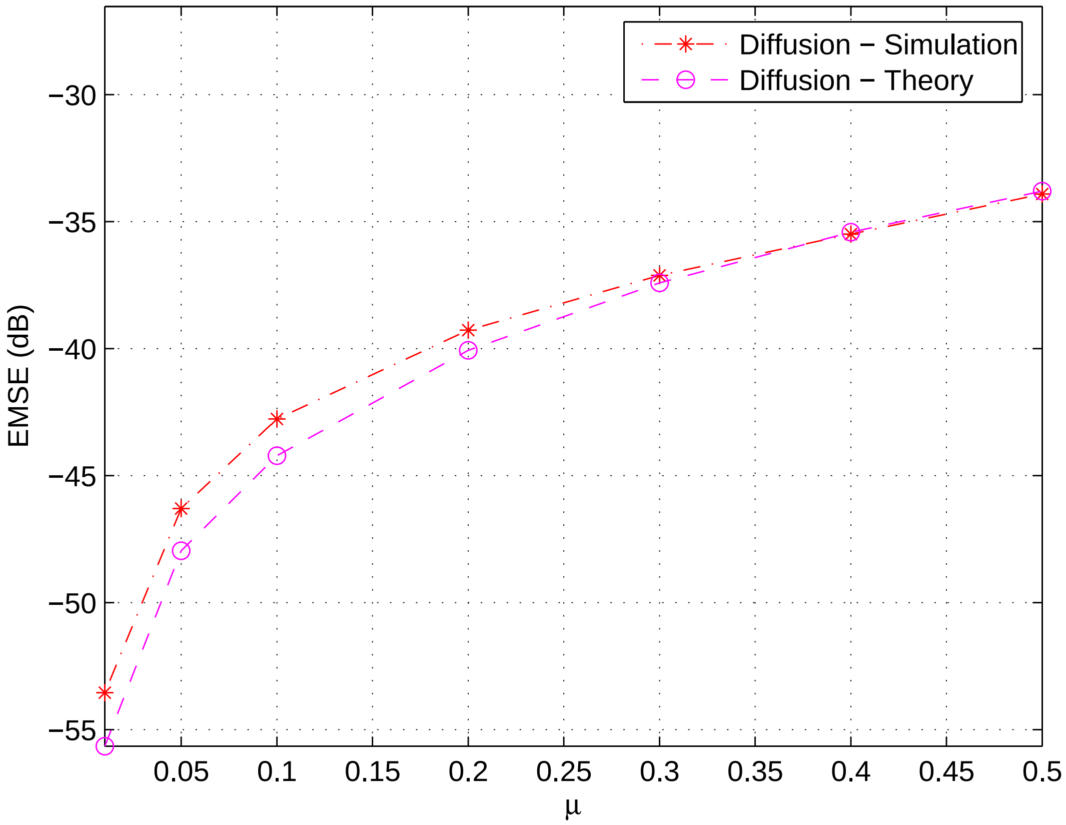


Figure 4.11: Global EMSE of the diffusion ε-NLMS algorithm for $\alpha = 0.4$ and Noise Variance 0.01.

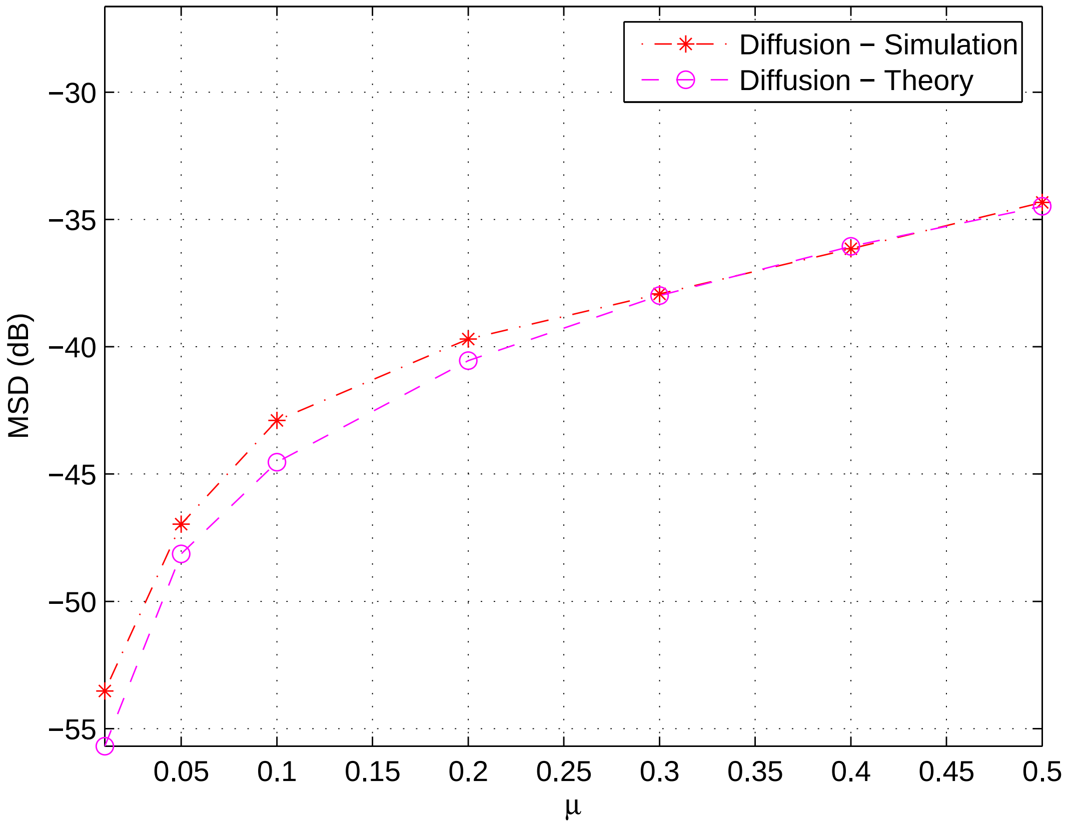


Figure 4.12: Global MSD of the diffusion ε-NLMS algorithm for $\alpha = 0.6$ and Noise Variance 0.01.

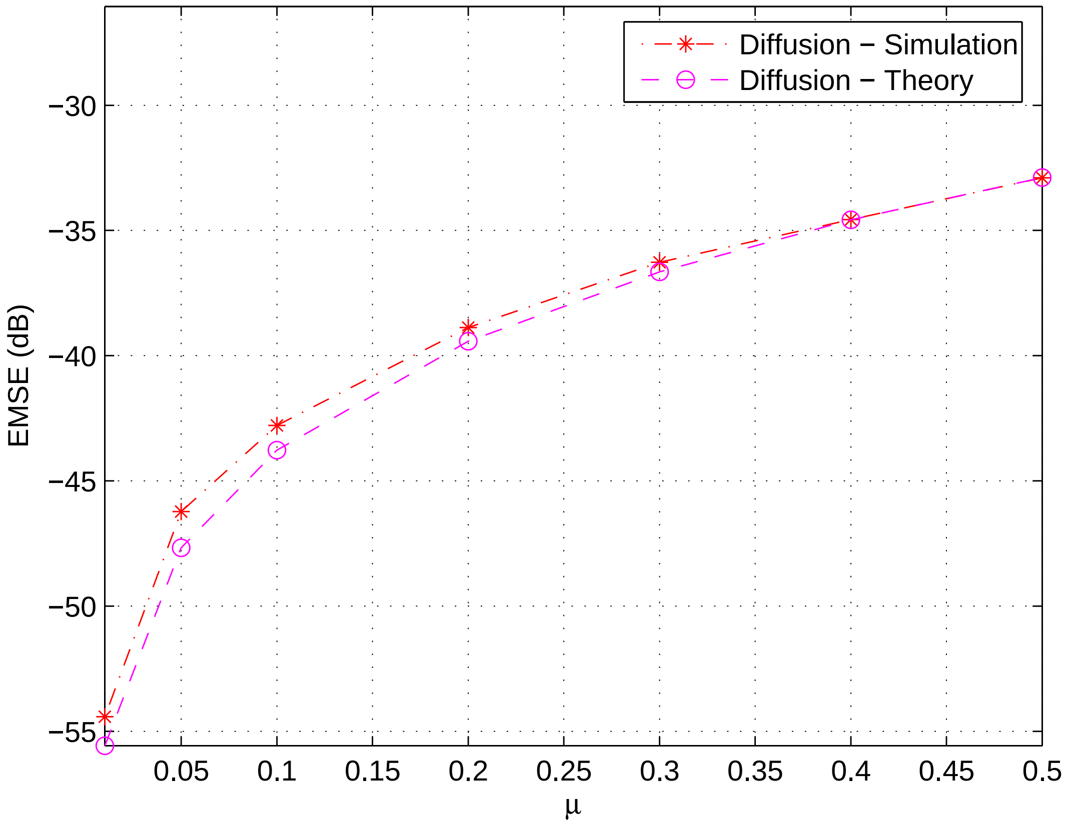


Figure 4.13: Global EMSE of the diffusion ε-NLMS algorithm for $\alpha = 0.6$ and Noise Variance 0.01.

CHAPTER 5

THESIS CONTRIBUTIONS AND RECOMMENDATIONS FOR FUTURE WORK

5.1 Thesis Contributions

This work successfully presented the diffusion ε-NLMS over wireless sensor networks. This algorithm was analyzed in terms of convergence properties, steady-state and transient behavior. The performance of the algorithm has been supported by presenting the simulation results to assess the performance of the diffusion ε-NLMS algorithm under various scenarios. The diffusion ε-NLMS algorithm is formulated recursively and then applied for estimation in a wireless sensor network environment. Following are the contribution of this thesis work:

1. Developed diffusion based ε-NLMS adaptive algorithm.

2. The transient analysis of the algorithm has been derived in mean and mean-square sense.

3. The steady state analysis of the algorithm has been carried.

4. The analytical results were compared with the experimental results which support the analyses.

5. Finally, the diffusion ε-NLMS algorithm is compared with diffusion LMS algorithm and the computational complexity of the algorithms is compared.

5.2 Recommendations for Future Work

Based on the results achieved in this work, there are few suggestions regarding the future work.

1. The proposed algorithm should be applied to applications such as radar detection and tracking, medical imaging, and spectrum sensing in order to study its effectiveness.

2. The work in this thesis is done using real-valued data only. The work should be extended to complex-valued data sets. Further, the analysis should be studied without using the independence assumptions.

3. In this thesis, we have only considered inter-correlation present between the regressor for which diffusion ε-NLMS algorithm shows impressive improvement in performance over diffusion LMS algorithm. However, in many cases, where nodes are placed in closed vicinity, data are expected to show spatial correlation. One needs to further exploit these correlations for better energy performance.

REFERENCES

[1] I.F. Akyildiz and I. H. Kasimoglu, “Wireless Sensor and Actor Networks: Research Challenges,” *Ad Hoc Networks Journal, vol. 2, pp. 351367*, Oct. 2004.

[2] I.F. Akyildiz and E. Stuntebeck, “Wireless Underground Sensor Networks: Research Challenges,” *Proc. Ad Hoc Networks Journal, vol. 4, pp. 669686*, Nov. 2006.

[3] I.F. Akyildiz and E. Stuntebeck, “Underwater Acoustic Sensor Networks: Research Challenges,” *Ad Hoc Networks Journal, vol. 3, pp. 257279*, Mar. 2005.

[4] I.F. Akyildiz, B. F. Lo, and R. Balakrishnan, “Cooperative Spectrum Sensing in Cognitive Radio Networks: A Survey,” *Physical Communication, vol. 4, pp. 4062*, Mar. 2011.

[5] Ning Xu, “A survey of sensor network applications,” *IEEE Communications Magazine, vol. 40, no. 8, pp. 102114*, 2002.

[6] C.G. Lopes and A.H. Sayed, "Incremental adaptive strategies over distributed networks," *IEEE Transactions on Signal Processing, vol. 55, no. 8, pp. 40644077*, 2007.

[7] G. Mateos, I.D. Schizas, and G.B. Giannakis, "Distributed recursive least squares for consensus-based in-network adaptive estimation," *IEEE Transactions on Signal Processing, vol. 57, no. 11, pp. 45834588*, 2009.

[8] G. Mateos, Ioannis D. Schizas, and Georgios B. Giannakis, "Performance Analysis of the Consensus-Based Distributed LMS Algorithm," *EURASIP Journal on Advances in Signal Processing, vol. 2009, pp. 120*, 2009.

[9] F.S. Cattivelli and a.H. Sayed, "Diffusion LMS Strategies for Distributed Estimation," *IEEE Transactions on Signal Processing, vol. 58, no. 3, pp. 10351048*, Mar. 2010.

[10] F.S. Cattivelli and A.H. Sayed, "Multi-level diffusion adaptive networks," *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP), pp. 27892792*, 2009.

[11] C.G. Lopes and A.H. Sayed, "Randomized incremental protocols over adaptive networks," *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP), pp. 35143517*, 2010.

[12] M.O. Bin Saeed, A. Zerguine, and S.A. Zummo, “Noise Constrained Diffusion Least Mean Squares over adaptive networks,” *IEEE International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC), number i, pp. 288292*, 2010.

[13] M.O. Bin Saeed, A. Zerguine, and S.A. Zummo, “Variable step-size least mean square algorithms over adaptive networks,” *10th International Conference on Information Sciences Signal Processing and their Applications (ISSPA), number ISSPA, pp. 381384*, 2010.

[14] 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. 31223136*, 2008.

[15] I.D. Schizas, Alejandro Ribeiro, and G.B. Giannakis, “Consensus in ad hoc WSNs with noisy linksPart I: Distributed estimation of deterministic signals,” *IEEE Transactions on Signal Processing, vol. 56, no. 1, pp. 350364*, 2008.

[16] N. Kalouptsidis and S. Theodoridis, “Adaptive System Identification and Signal Processing Algorithms,” *Englewood Cliffs, NJ: Prentice-Hall*, 1993.

[17] H. Sakai, J. M. Yang, and T. Oka, “Exact convergence analysis of adaptive filter algorithms without the persistently exciting condition,” *IEEE Trans. Signal Process., vol. 55, no. 5, pp. 2077-2083*, May. 2007.

[18] A. H. Sayed, "Fundamentals of Adaptive Filtering," *Wiley, NJ*, 2003.

[19] S. Haykin, "Adaptive Filter Theory," *Englewood Cliffs, NJ: Prentice Hall*, 2002.

[20] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine, vol. 40, no. 8, pp. 102114*, 2002.

[21] I.F. Akyildiz and I. H. Kasimoglu, "Wireless Sensor and Actor Networks: Research Challenges," *Ad Hoc Networks Journal, vol. 2, pp. 351367*, Oct. 2004.

[22] J. M. Grriz, Javier Ramrez, S. Cruces-Alvarez, Carlos G. Puntonet, Elmar W. Lang, and Deniz Erdogmus, "A Novel LMS Algorithm Applied to Adaptive Noise Cancellation," *IEEE Signal Processing Letters, VOL. 16, NO. 1*, Jan. 2009.

[23] Markus Rupp and Rudi Frenzel, "Analysis of LMS and NLMS Algorithms with Delayed Coefficient Update Under the Presence of Spherically Invariant Processes," *IEEE Transactions on Signal Processing, VOL. 42, NO. 3*, Mar. 1994.

[24] Slock, D.T.M., "On the convergence behavior of the LMS and the normalized LMS algorithms," *IEEE Transactions on Signal Processing, ISSN: 1053-5878, Vol. 41, Issue 9, pp. 2811-2825*, 1993.

[25] Kim, J., Poularikas, A.D., "Comparison of two proposed methods in adaptive noise canceling," *Proceedings of the 35th Southeastern Symposium on System Theory. ISSN: 0094-2898, pp. 400-403*, 2003.

[26] Sulyman, A.I., Zerguine, A., "Convergence and steady-state analysis of a variable step-size normalized LMS algorithm," *Proc. Seventh International Symposium on Signal Processing and Its Applications, Vol. 2, pp. 591-594*, 2003.

[27] S. Haykin, "Adaptive Filter Theory," *Englewood Cliffs, NJ: Prentice Hall*, 2002.

[28] R. Olfati-Saber, J. Alex Fax, and Richard M. Murray, "Consensus and Cooperation in Networked Multi-Agent Systems," *Proceedings of the IEEE, vol. 95, no. 1, pp. 215233*, Jan. 2007.

[29] L Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Systems & Control Letters, vol. 53, no. 1, pp. 6578*, Sept. 2004.

[30] R. Olfati-Saber and R.M. Murray, “Consensus problems in networks of agents with switching topology and time-delays,” *IEEE Transactions on Automatic Control, vol. 49, no. 9, pp. 15201533*, 2004.

[31] S Barbarossa, G Scutari, and T Battisti, “Distributed signal subspace projection algorithms with maximum convergence rate for sensor networks with topological constraints,” *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP), pp. 28932896*, 2009.

[32] R. Olfati-Saber and J.S. Shamma, “Consensus filters for sensor networks and distributed sensor fusion,” *IEEE Conference on Decision and Control and European Control Conference. CDC-ECC05, number 1, pp. 66986703*, 2005.

[33] L. Xiao, S. Boyd, and S. Lall, “A space-time diffusion scheme for peer-to-peer least-squares estimation,” *Proceedings of the 5th international conference on Information processing in sensor networks, pp. 168176, ACM*, 2006.

[34] C.G. Lopes and A.H. Sayed, “Distributed recursive least-squares strategies over adaptive networks,” *Asilomar Conference on Signals, Systems and Computers, number 1, pp. 233237*, 2006.

[35] 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. 18651877*, 2008.

[36] S.Y. Tu and A.H. Sayed, "Mobile adaptive networks with self-organization abilities," *7th International Symposium on Wireless Communication Systems (ISWCS), number 4, pp. 379383*, 2010.

Vitae

- Syed Abdul Baqi.

- Born in Hyderabad, India on June 12, 1985.

- Received Bachelor of Technology (B.Tech.) in Electronics and Communications Engineering from Jawaharlal Nehru Technological University, India in 2007.

- Joined King Fahd University of Petroleum and Minerals in February 2010 as a Research Assistant in Electrical Engineering Department.

- **Contact details:**

- **Present Address:** King Fahd University of Petroleum and Minerals, P.O. Box 8653, Dhahran 31261, Saudi Arabia.

- **E-mail Address:** g200905410@kfupm.edu.sa

- **Permanent Address:** 16-9-409/P/6, Old Malakpet, Hyderabad 500036, Andhra Pradesh, India.

- **E-mail Address:** baqi.syed@gmail.com