ADAPTIVE DIFFUSION LMS STRATEGIES

by

Altynbek Isabekov

A Thesis Submitted to the
Graduate School of Engineering
in Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in

Electrical & Computer Engineering

Koç University

September, 2011

Koç University

Graduate School of Sciences and Engineering

This is to certify that I have examined this copy of a master's thesis by

Altynbek Isabekov

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Committee Members:

Assist. Prof. Dr. Süleyman Serdar Kozat

Assoc. Prof. Dr. Engin Erzin

Assist. Prof. Dr. Fatma Sibel Salman

Date:

To my family

ABSTRACT

In this thesis new distributed adaptive algorithms for the in-networking parameter estimation problem are proposed. They are cooperative and resistant to link failures. The individual nodes run local least-mean squares (LMS) algorithm to estimate the common parameter of interest and then share these estimates with nodes in vicinity. Neighbor nodes use these data to update their own estimates by combining received estimates and processing the resulting aggregate estimates in the local adaptive LMS filters. This strategy is known as the diffusion LMS algorithm.

In the first chapter of the thesis stability and convergence of the diffusion LMS algorithm is introduced. Theoretical statement of the evolution of the mean-square deviation (MSD) and excess mean-square error (EMSE) are given in short as stated in literature. Simulations show perfect match between experimental and theoretical evolution of these error measures for diffusion algorithm. Also experiments show that this algorithm has a faster convergence and better performance (tens of dB difference in MSD and EMSE) compared to noncooperative LMS.

The second chapter of the thesis contains main contributions of the research. In the diffusion LMS algorithm, aggregation step comprises of combining neighbor estimates by weighing them with constant coefficients. Contrary to this approach, in proposed adaptive diffusion algorithms another adaptation layer is introduced to update these weighing coefficients at every iteration. The weights are constrained to produce a) convex, b) affine combination or c) may not have any constraints. For adaptation purpose gradient-descent algorithm is used. Simulations show that in some cases adaptive diffusion LMS algorithms have faster convergence than classical diffusion algorithm with penalty in larger MSD and EMSE values in the steady-state.

ÖZETÇE

Bu tezde, ağ içindeki parametre kestirimi problemi için yeni uyarlanır algoritmalar önerilmektedir. Bu algoritmalar işbirlikseldirler ve düğümlerin arasındaki bağlantı kopmalarına karşı dayanıklıdır. Bireysel düğümler ortak bir parametreyi kestirmek için en küçük ortalama kare (LMS) algoritmasını çalıştırmaktadırlar ve elde edilen kestirimlerini komşu düğümlerle paylaşmaktadırlar. Komşu düğümler alınan tahminleri birleştirerek, oluşan toplam kestirimle kendi (yerel) kestirimlerini güncellemek için çalışan uyarlamalı LMS süzgeçlerini beslemektedirler. Bu strateji, yayınım LMS algoritması olarak bilinir.

Tezin ilk bölümünde yayınım LMS algoritmasının kararlılığı ve yakınsaması literatürde verildiği gibi incelenmektedir. Ayrıca ortalama karesel sapma (MSD) ve fazlalık ortalama karesel hata (EMSE)'nın zamanla gelişiminin teorik analizi verilmiştir. Benzetimlerde, yayınım algoritmasına ait MSD ve EMSE hatalarının deneysel ve teorik gelişimlerinin arasında mükemmel bir uyumun sağlandığını gösterilmiştir. Ayrıca deneyler, yayınım LMS algoritmasının işbirliksel olmayan LMS algoritmasına göre daha hızlı bir yakınsamaya ve daha iyi bir performansa (MSD ve EMSE değerlerinde on dB'lik fark mertebesinde) sahip olduğunu göstermektedir.

Tezin ikinci bölümü, yürütülen araştırmanın ana katkılarını içermektedir. Yayınım LMS algoritması komşu kestirimleri birleştirirken onları sabit katsayılarla çarparak elde edilen kestirimleri toplamaktadır. Bu yaklaşımın aksine, önerilen uyarlamalı yayınım algoritmalarda her yinelemede bu katsayıları güncellemek için başka bir uyarlama katmanı kullanılmaktadır. Bu ağırlık katsayıları a) tümsek, b) ılgın kombinasyonu oluşturabilir, ya da c) katsayılar için herhangi bir kısıtlama olmayabilir. İkincil uyarlama katmanında katsayıları güncellemek için en dik iniş (steepest-descent) algoritması kullanılmaktadır. Deneysel sonuçlar, bazı durumlarda uyarlamalı yayınım LMS algoritmalarının geleneksel yayınım algoritmasından daha iyi bir performans sergilediğini ve çoğu zaman hızlı yakınsamayı elde etmek için kararlı durumda daha büyük bir MSD ve EMSE değerleriyle telafi etmenin gerektiğini göstermektedir.

ACKNOWLEDGMENTS

I wish to express my genuine gratitude to my advisor, Dr. Süleyman Serdar Kozat. His insight, experience, guidance and encouragement were very helpful in this study. I consider myself to be truly fortunate to have an opportunity to work with him. I am very thankful to members of my MSc thesis committee, Dr. Sibel Salman and Dr. Engin Erzin. Also I would like to thank Dr. Alper T. Erdoğan for teaching me concepts of estimation.

On a personal note, I express my gratitude to my family for their lifetime support, encouragement and love.

TABLE OF CONTENTS

LIST OF FIGURES

NOMENCLATURE

col{.}	:	Column vector operator, concatenates entries into a column vector
vec{.}	:	Matrix vectorization operator, produces a column vector by concatenating columns of the matrix
diag{.}	:	Diagonalization and block-diagonal matrix operator, puts entries on the matrix diagonals
$\mathbb{E}[.]$	:	Expectation operator
$A \otimes B$	:	Kronecker product of the matrices A and B
$A \odot B$	:	Tracy-Singh product of the matrices A and B
$\|a\|$	:	Euclidian norm of the vector a
$\|a\|_W$	:	Weighted norm of the vector a
$\|A\|_2$	:	2-norm of the matrix A (the largest singular value of the matrix)

Chapter 1

INTRODUCTION

Distributed computing gains importance with development and cheapening of sensor devices which are widely used in industrial, military and environmental applications [1, 2]. Unlike centralized approach, distributed computing benefits from in-network cooperation making the network more scalable and reducing computational load on the central node.

In this thesis, the problem of distributed estimation as a further development of the framework introduced in [3, 4] is studied. According to this framework, nodes deployed on a certain region cooperate in the *diffusion* mode, i.e. they share their estimates with neighbors and combine collected estimates linearly to process it in a local adaptive least-mean squares (LMS) filter. Depending on nature of combination weights diffusion LMS strategies can be classified into two categories: linear combination with 1) constant and 2) adaptive weights.

1.1 Diffusion LMS Strategies with Constant Weights

In this setup neighbor estimates are linearly combined using constant weights. These combination weights are computed according to network topology and connectivity information. Several combination rules such as Metropolis, Laplacian and nearest-neighborhood were introduced in the literature [3]. Although it is a simple strategy, the collaboration among nodes make estimation robust to link failures.

1.2 Diffusion LMS Strategies with Adaptive Weights

Contrary to previous approach neighbor estimates are combined using adaptive weights which are updated to minimize mean-square error (MSE). Three adaptive mixture algorithms applied to two operation modes are studied. The mixture algorithms produce either convex combination (where the linear mixture weights are constrained to be nonnegative and sum up to one), affine combination (where the linear mixture weights are constrained to sum up to one) or unconstrained linear combination of nodes' estimates in the neighborhood. These mixture approaches applied to adaptive filters showed performance gain in steady-state [5, 6], which makes it reasonable to apply these algorithms in the diffusion framework. The first operation mode is a *pairwise combination* of node's estimate and averaged value of its neighbor estimates. The second operation mode is a *mixture* of all nodes' estimates within vicinity (including the host node itself) with assigned individual weights. The mixture weights are updated according to gradient-descent algorithm, hence forming second layer of adaptation in addition to the existing adaptation used in local LMS filters.

1.3 Contributions

In this thesis five new adaptive diffusion algorithms are proposed and their performance is compared:

1. Adaptive diffusion using convex mixture;
2. Adaptive diffusion using affine pairwise combination;
3. Adaptive diffusion using affine mixture;
4. Adaptive diffusion using unconstrained pairwise combination;
5. Adaptive diffusion using unconstrained mixture.

Simulations show that in some cases adaptive diffusion LMS algorithms have faster convergence than nonadaptive diffusion algorithm with a cost of bigger MSD and EMSE values in the steady-state.

1.4 Content

The thesis has the following organization. Chapter 2 covers the problem statement. Chapter 3 describes the non-adaptive diffusion algorithm with constant coefficients. Chapter 4 describes the adaptive diffusion algorithm. Sections 4.3, 4.4, 4.5 describe derivation and formulation of convex, affine and unconstrained mixtures respectively operating in two modes depicted above. Numerical simulations and performance analysis of all six algorithms are given within corresponding sections. Conlusion, discussion and future work are given in the final Chapter 5.

Chapter 2

PROBLEM STATEMENT

The collaborative estimation problem has the following setup [3]. Assume that there are $N > 1$ nodes randomly distributed in some space. Each node k has access to time realizations $\{d_k(t), u_k(t)\}, k = 1, \ldots, N$ of zero-mean random real data $\{\boldsymbol{d}_k(t), \boldsymbol{u}_k(t)\}$, with $\boldsymbol{d}_k(t)$ a scalar measurement and $\boldsymbol{u}_k(t)$ regression row vector of size $(1 \times M)$ at time t. The aim is to estimate $M \times 1$ unknown vector w^0 from these measurements, given that:

$$d_k(t) = u_k(t)w^0 + v_k(t), \qquad k = 1, \ldots, N. \tag{2.1}$$

Regression vectors and measurement data are gathered among all N nodes into two global matrices:

$$U_c \triangleq \text{col}\{u_1, u_2, \ldots, u_N\} \qquad (N \times M) \tag{2.2}$$

$$d \triangleq \text{col}\{d_1, d_2, \ldots d_N\} \qquad (N \times 1). \tag{2.3}$$

The estimation problem is defined as finding such a vector w which minimizes the mean-square error:

$$\min_w \mathbb{E}\|d - U_c w\|^2 \tag{2.4}$$

where $\mathbb{E}$ is an expectation operator. The optimal (in MSE sense) solution is found by using pojection theorem which implies orthogonality condition [7]:

$$\mathbb{E}\left[U_c^T(d - U_c w^0)\right] = 0, \tag{2.5}$$

so the solution can be expressed in terms of

$$w^0 = (R_u^c)^{-1} R_{du}^c \tag{2.6}$$

where

$$R_u^c = \mathbb{E}\left[U_c^T U_c\right] \ (M \times M) \quad \text{and} \quad R_{du}^c = \mathbb{E}\left[U_c^T d\right] \ (M \times 1). \tag{2.7}$$

Instead of calculating these correlations using time-averaging with ergodicity assumption, it is more convenient to use real-time implementation based on adaptive filtering. The block-diagram of the adaptive filtering approach is shown in Fig. 2.1. Among all adaptive filters the least-mean squares (LMS) filter is a simple one and it is easy to implement.

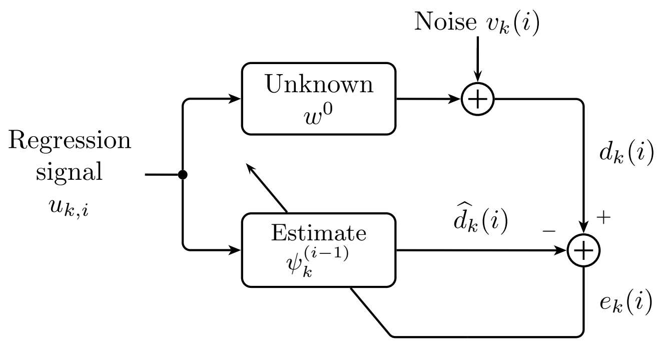


Figure 2.1: Block-diagram of the adaptive filtering procedure.

2.1 Noncooperative LMS

Firstly, consider a single node case without any collaboration. The goal of the adaptive filtering is to find an estimate ψ which minimizes the mean-square error (MSE) in the iterative way:

$$\psi(i) = \arg\min_{w} \, [e(i)^2]. \tag{2.8}$$

A simple solution is to run the LMS algorithm at every node k independently:

$$\begin{aligned}
\psi_k^{(i)} &= \psi_k^{(i-1)} - \frac{\mu}{2}\left[\frac{\partial e_k^2(i)}{\psi_k^{(i-1)}}\right], \\
\psi_k^{(i)} &= \psi_k^{(i-1)} - \mu e_k(i)\left[\frac{\partial e_k(i)}{\partial \psi_k^{(i-1)}}\right], \\
\psi_k^{(i)} &= \psi_k^{(i-1)} + \mu e_k(i) u_{k,i}^T.
\end{aligned} \tag{2.9}$$

This approach does not take advantage of cooperation among nodes.

2.2 Diffusion LMS

In case of collaboration among nodes, the estimates of neighbor nodes are combined to form a more reliable (in some sense) estimate which is further processed in a local adaptive LMS filter. One of such distributed algorithms is called *combine-then-adapt diffusion least-mean squares algorithm* and defined as [3]:

$$\begin{aligned} \phi_k^{(i-1)} &= f_k\left(\psi_l^{(i-1)}; l \in \mathcal{N}_k\right), \quad \phi_k^{(i-1)} = 0 \\ \psi_k^{(i)} &= \phi_k^{(i-1)} + \mu_k u_{k,i}^T \left(d_k(i) - u_{k,i}^T \phi_k^{(i-1)}\right), \end{aligned} \tag{2.10}$$

where $\mathcal{N}_k$ is a neighborhood of k-th node (nodes connected to k including itself), $\psi_k^{(i-1)}$ and $\phi_k^{(i-1)}$ are local and aggregate estimates of w^0 respectively, at node k at time $i-1$ and $f_k(.)$ is a combiner function. There is another class of algorithms similar to (2.10) with inverse order of operations where the adaptation step precedes combination. Analysis of both combine-then-adapt (CTA) and adapt-then-combine (ATC) algorithms are given in [8]. In this thesis, only the CTA strategy is examined. The *CTA diffusion LMS* algorithm will be referred as *diffusion LMS* algorithm.

Possible alternatives for the combiner function $f_k\left(\psi_l^{(i-1)}; l \in \mathcal{N}_k\right)$ are:

1. Linear combination with constant coefficients (diffusion LMS),
2. Linear combination with adaptive coefficients (adaptive diffusion LMS).

Both variants have some advantages and disadvantages in terms of convergence, implementation and computational cost.

Chapter 3

DIFFUSION LEAST-MEAN SQUARES

As it was stated earlier the diffusion LMS algorithm is described using following equations:

$$\begin{aligned} \phi_k^{(i-1)} &= f_k\left(\psi_l^{(i-1)}; l \in \mathcal{N}_k\right), \quad \phi_k^{(i-1)} = 0 \\ \psi_k^{(i)} &= \phi_k^{(i-1)} + \mu_k u_{k,i}^T \left(d_k(i) - u_{k,i}^T \phi_k^{(i-1)}\right), \end{aligned} \tag{3.1}$$

In the diffusion LMS algorithm combiner function has a form of:

$$\phi_k^{(i-1)} = \sum_{l \in \mathcal{N}_k} c_{k,l} \psi_l^{(i-1)}, \tag{3.2}$$

with nonnegative constant coefficients $c_{k,l}$. The aggregate estimate $\phi_k^{(i-1)}$ can be considered as a weighted least-squares estimate of w^0 given estimates of the node k's neighbors. If these estimates $\{\psi_l^{(i-1)},\ l \in \mathcal{N}_k\}$ are put in a column vector

$$\psi_{\mathcal{N}_k} \triangleq \text{col}\{\psi_l^{(i-1)}\}_{l \in \mathcal{N}_k}, \tag{3.3}$$

the local weighted least-squares problem can be formed:

$$\min_{\phi} \|\psi_{\mathcal{N}_k} - Q\phi_k\|_{C_k}^2 \tag{3.4}$$

where

$$\begin{gathered} Q = \text{col}\{I_M, I_M, \ldots, I_M\} \quad \text{and} \\ C_k = \text{diag}\{c_{k,1} I_M, \ldots, c_{k,\mathcal{N}_k} I_M\}. \end{gathered} \tag{3.5}$$

The solution to (3.4) has a form of

$$\phi_k^{(i-1)} = \sum_{l \in \mathcal{N}_k} \frac{c_{k,l}}{\sum\limits_{p \in \mathcal{N}_k} c_{k,p}} \psi_l^{(i-1)}, \tag{3.6}$$

which puts a constraint on new $c_{k,l} \leftarrow c_{k,l} / \sum_{p \in \mathcal{N}_k} c_{k,p}$:

$$\sum_{l \in \mathcal{N}_k} c_{k,l} = 1, \quad \forall k. \tag{3.7}$$

Nonnegative weights $c_{k,l},\ l \in \mathcal{N}_k$ which sup up to one form a convex combination which can be chosen according to some rule [9–11]. The popular ones are:

1. Metropolis rule,
2. Laplacian rule,
3. Nearest-neighborhood rule.

These rules for choosing coefficients of the combiner function are purely based on network topology. A useful term describing network topology is degree of a node n_k. It is defined as a number of neighbors connected to the node (including node itself), such that $n_k = |\mathcal{N}_k|$.

To deal with network topology more compact representation of weghts $c_{k,l}$ is introduced. $(N \times N)$ diffusion combination matrix C is a matrix representation of the weight coefficients:

$$[C]_{k,l} = c_{k,l} \tag{3.8}$$

Also note that $C\mathbb{1} = \mathbb{1}$. Using above equations combiner functions are easily defined.

Metropolis rule sets coefficients $c_{k,l}$ according to degrees of nodes, n_k and n_l:

$$\begin{aligned}
c_{k,l} &= \frac{1}{\max(n_k, n_l)}, && \text{if } k \neq l \text{ are linked,} \\
c_{k,l} &= 0, && \text{if } k \text{ and } l \text{ not linked,} \\
c_{k,k} &= 1 - \sum_{l \in \mathcal{N}_k \setminus k} c_{k,l}, && \text{if } k = l.
\end{aligned} \tag{3.9}$$

Laplacian rule is described by the following equations:

$$\begin{aligned}
&C = I_N - \kappa\mathcal{L} \quad \text{where}\\
&\mathcal{L} = \mathcal{D} - A_d \quad \text{with}\\
&\mathcal{D} = \operatorname{diag}\{n_1, n_2, \ldots, n_N\} \text{ and}\\
&\kappa = 1/n_{max}
\end{aligned} \tag{3.10}$$

where A_d is $N \times N$ network adjacent matrix formed as

$$[A_d]_{kl} = \begin{cases} 1, & \text{if } k \text{ and } l \text{ are linked,}\\ 0, & \text{otherwise.}\end{cases} \tag{3.11}$$

Nearest-neighborhood rule is defined as:

$$\begin{aligned}
&c_{k,l} = \frac{1}{|\mathcal{N}_k|}, && l \in \mathcal{N}_k,\\
&c_{k,l} = 0, && \text{otherwise.}
\end{aligned} \tag{3.12}$$

3.1 Centralized Network Model

In order to perform analysis, interdependency among nodes should be considered and the state of the whole network should be accessible. Since lots of linear algebra is involved in the analysis, all variable are collected into global quantities which are represented by vectors and matrices:

$$\begin{aligned}
&\psi^{(i)} \triangleq \operatorname{col}\{\psi_k^{(i-1)}, \ldots, \psi_N^{(i-1)}\} \quad (NM \times 1), \quad &&\phi^{(i)} \triangleq \operatorname{col}\{\phi_k^{(i-1)}, \ldots, \phi_N^{(i-1)}\} \quad (NM \times 1),\\
&U_i \triangleq \operatorname{diag}\{u_{1,i}, \ldots, u_{N,i}\} \quad (N \times NM), \quad &&d_i \triangleq \operatorname{col}\{d_1(i), \ldots, d_N(i)\} \quad (N \times 1).
\end{aligned}$$

The measurement $d_k(i)$ obeys a typical model:

$$d_k(i) = u_{k,i} w^0 + v_k(i) \tag{3.13}$$

where $v_k(i)$ is an additive noise, with assumption of spatial and temporal independence [3]. By introducing

$$\begin{aligned} v_i &= \mathrm{col}\{v_1(i), \ldots, v_N(i)\} \quad \text{and} \\ w^{(0)} &= Qw^0 \end{aligned} \tag{3.14}$$

with Q defined in (3.5) it is possible to express all measurements $\{d_k(i), k = 1, \ldots, N\}$ using single equation:

$$d_i = U_i w^{(0)} + v_i. \tag{3.15}$$

The diffusion LMS algorithm described by (3.1) is represented using global model with these equations:

$$\begin{aligned} \phi^{(i-1)} &= G\psi^{(i-1)} \\ \psi^{(i)} &= \phi^{(i-1)} + DU_i^T(d_i - U_i\phi^{(i-1)}) \end{aligned} \tag{3.16}$$

where

$$D = \mathrm{diag}\{\mu_1 I_M, \mu_2 I_M, \ldots, \mu_N I_M\} \quad (NM \times NM) \tag{3.17}$$

is a diagonal matrix with local step sizes and

$$G = C \otimes I_M \quad (NM \times NM) \tag{3.18}$$

is a transition matrix with $N \times N$ diffusion combination matrix C filled with nonnegative coefficients $c_{k,l}$. After substitution (3.16) reduces to a single equation:

$$\psi^{(i)} = G\psi^{(i-1)} + DU_i^T(d_i - U_i G\phi^{(i-1)}). \tag{3.19}$$

3.2 Stability Analysis of the Diffusion Algorithm

During implementation of the diffusion LMS algorithm the proper choice of the step sizes μ_k is important. If the step size is very small, convergence will be slow. On contrary, if the step size is too big, the algorithm will probably diverge. In some applications the statistics of the regression signal $u_{k,i}$ and background noise $v_k(i)$ may be known a priori or estimated before.

In this section stability analysis of the diffusion LMS algorithm is revealed as it was reported in [3]. The deviation of the nodes' estimates from original unknown vector w^0 is defined as a global error vector:

$$\widetilde{\psi}^{(i)} = w^{(0)} - \psi^{(i)}. \tag{3.20}$$

The evolution of global error vector $\widetilde{\psi}^{(i)}$ in terms of its previous values and noise is represented using centralized network model (3.19). Recalling convex combination constraint of diffusion combination matrix C it is easy to note that $Gw^{(0)} = w^{(0)}$. By substituting $w^{(0)}$ and $Gw^{(0)}$ into (3.19), the evolution of $\widetilde{\psi}^{(i)}$ is found to be:

$$\begin{aligned}\widetilde{\psi}^{(i)} &= Gw^{(0)} - G\psi^{(i-1)} - DU_i^T\left(U_i w^{(0)} + v_i - U_i G\psi^{(i-1)}\right)\\ &= G\widetilde{\psi}^{(i-1)} - DU_i^T\left(U_i G\widetilde{\psi}^{(i-1)} + v_i\right)\\ &= \left(I_{NM} - DU_i^T U_i\right) G\widetilde{\psi}^{(i-1)} - DU_i^T v_i. \end{aligned} \tag{3.21}$$

Assuming independence of $u_{k,i}$ from v_i and taking expectations of both sides of (3.21) results in:

$$\mathbb{E}[\widetilde{\psi}^{(i)}] = (I_{NM} - DR_u)\, G\mathbb{E}[\widetilde{\psi}^{(i-1)}] \tag{3.22}$$

where $R_u = \text{diag}\{R_{u,1}, \ldots, R_{u,N}\}$ with $R_{u,k} = \mathbb{E}[u_{k,i}^T u_{k,i}]$, i.e. evolution of $\widetilde{\psi}^{(i)}$ depends on network statistics $(I_{NM} - DR_u)$ and topology G. Therefore for stability in the mean magnitude of all eigenvalues of $(I_{NM} - DR_u)\, G$ must be smaller than one:

$$|\lambda\left\{(I_{NM} - DR_u)\, G\right\}| < 1. \tag{3.23}$$

The step sizes μ_k in matrix D should be chosen accordingly to satisfy this criterion:

$$0 < \mu_k < \frac{2}{\max\limits_{m=1,\ldots,N} \lambda_{max}\left(R_{u,m}\right)}. \tag{3.24}$$

Using matrix 2-norm which is defined as a largest singular value of a matrix [12], it is easy to show that:

$$\|(I_{NM} - DR_u)\, G\|_2 \leq \|(I_{NM} - DR_u)\|_2 \cdot \|G\|_2. \tag{3.25}$$

Noting that $(I_{NM} - DR_u)$ is a symmetric matrix and $G = C \otimes I_M$, (3.25) reduces to:

$$|\lambda_{\text{max}}\left((I_{NM} - DR_u)\, G\right)| \leq \|C\|_2 \cdot |\lambda_{\text{max}}\left((I_{NM} - DR_u)\right)|. \tag{3.26}$$

Although not all diffusion combination matrices C are symmetric, the ones that are generated using Metropolis, Laplacian and nearest-neighborhood rules have 2-norm equal to one, $\|C\|_2 = 1$, which implies the following relation:

$$|\lambda_{\text{max}}\left((I_{NM} - DR_u)\, G\right)| \leq |\lambda_{\text{max}}\left((I_{NM} - DR_u)\right)|, \tag{3.27}$$

note that right side of (3.27) is a 2-norm of a noncooperative scheme with $G = I_{NM}$ which means that usually cooperation among nodes has a stabilizing effect on the network [3].

3.3 Mean-Square Transient Analysis

After introducing stability conditions, transient analysis of the network behavior and evolution of error measures, particularly mean-square deviation (MSD) and excess mean-square error (EMSE) is given. This is a good tool to compare theoretical expectations and experimental results. Obviously, the analysis is challenging due to interdependency of nodes in the network. Its detailed derivation is made in [3]. In this thesis only resulting theorem is given and justified using numerical tools.

Necessary definitions of the quantities used in Theorem 1 are given below. Eigendecomposition $R_u = S\Lambda S^T$ with $\Lambda = \text{diag}\{\Lambda_1, \ldots, \Lambda_N\}$ is used to transform several quantities including:

$$\begin{aligned} \overline{G} &= S^T G S \\ \overline{w}^{(0)} &= S^T w^{(0)}. \end{aligned} \tag{3.28}$$

The global mean-square deviaton (MSD), $\eta(i)$ is defined as an average of all nodes' mean-square deviation values:

$$\eta(i) \triangleq \frac{1}{N} \sum_{k=1}^{N} \mathbb{E}\|\widetilde{\psi}_k^{(i)}\|^2 = \frac{1}{N}\mathbb{E}\|\overline{\psi}^{(i)}\|^2. \tag{3.29}$$

In a similar way, the global excess mean-square error (EMSE), $\zeta(i)$ is defined as:

$$\zeta(i) \triangleq \frac{1}{N}\sum_{k=1}^{N}\mathbb{E}\|u_{k,i}\widetilde{\psi}_k^{(i)}\|^2 = \frac{1}{N}\mathbb{E}\|U_i\widetilde{\psi}^{(i)}\|^2 = \frac{1}{N}\mathbb{E}\|\overline{\psi}^{(i)}\|_\Lambda^2. \tag{3.30}$$

3.3.1 Block Operations

In order to compute correlation terms resulting from manipulation with (3.21) and to deal with block-diagonal matrices (R_u and other which are not given here) some advanced linear algebra methods are needed. Two useful tools, the bvec{} operator and the block Kronecker product are given below.

Let Σ be a $(NM \times NM)$ block matrix of the form:

$$\Sigma = \begin{bmatrix} \Sigma_{11} & \dots & \Sigma_{1N} \\ \vdots & \ddots & \vdots \\ \Sigma_{N1} & \dots & \Sigma_{NN} \end{bmatrix} \tag{3.31}$$

with blocks $\Sigma_{kl} \in \mathbb{R}^{M\times M}$. The block vectorization operator bvec{} converts matrix Σ into a $(N^2M^2 \times 1)$ vector σ in two steps [13, 14]. Recall that the standart vec{} operator, which converts a matrix to a single column vector by vertically concatenating columns of the matrix, is used to obtain a $(M^2N \times N)$ matrix Σ_v:

$$\Sigma_v = \begin{bmatrix} \text{vec}\{\Sigma_{11}\} & \dots & \text{vec}\{\Sigma_{1N}\} \\ \vdots & \ddots & \vdots \\ \text{vec}\{\Sigma_{N1}\} & \dots & \text{vec}\{\Sigma_{NN}\} \end{bmatrix} \tag{3.32}$$

The same operator vec{} converts Σ_v into a single column vector σ:

$$\sigma = \text{bvec}\{\Sigma\} \triangleq \text{vec}\{\Sigma_v\}. \tag{3.33}$$

Inverse operation is done using $\text{bvec}^{-1}\{\}$ operator, which converts vector σ into a $(NM \times NM)$ matrix Σ.

The block Kronecker product (aslo referred as Tracy-Singh product) of two $(NM\times NM)$ block matrices A and B, which is denoted by $A \odot B$, is defined as a $(N^2M^2 \times N^2M^2)$ block matrix whose k, l-th block is given by the $M^2N \times M^2N$ matrix

$$[A \odot B]_{kl} \triangleq \begin{bmatrix} A_{kl} \otimes B_{11} & \dots & A_{kl} \otimes B_{1N} \\ \vdots & \ddots & \vdots \\ A_{kl} \otimes B_{N1} & \dots & A_{kl} \otimes B_{NN} \end{bmatrix} \tag{3.34}$$

where $\otimes$ denotes the standart Kronecker product. These block operators are used to evaluate weighted norms of matrices [3]. The most valuable properties of these operators are [13, 14]:

$$\text{bvec}\{A\Sigma B\} = (B^T \odot A)\text{bvec}\{\Sigma\} \tag{3.35}$$

$$\text{Tr}(A^T B) = \text{bvec}\{A\}^T \text{bvec}\{B\}. \tag{3.36}$$

3.3.2 Transient Analysis: Evolution of the Mean-Square Deviation and the Excess Mean-Square Error in the Network

Theorem 1: Consider an adaptive network operating under the diffusion protocol (3.1) with space-time data $\{d_k(i), u_{k,i}\}$ satisfying (3.13). Assume further that the regressors $u_{k,i}$ are circularly Gaussian and independent over time and space. The network mean-square deviation (MSD) and excess mean-square error (EMSE) evolve as follows:

$$\eta(i) = \eta(i-1) + b^T \overline{F}^i q_n - \|\overline{w}^{(0)}\|^2_{\overline{F}^i (I-\overline{F}) q_n} \quad \text{(MSD)} \tag{3.37}$$

$$\zeta(i) = \zeta(i-1) + b^T \overline{F}^i \lambda_\zeta - \|\overline{w}^{(0)}\|^2_{\overline{F}^i (I-\overline{F}) \lambda_\zeta} \quad \text{(EMSE)} \tag{3.38}$$

with initial conditions $\eta(-1) = \|w^0\|^2$ and $\zeta(-1) = \frac{1}{N}\|\overline{w}^{(0)}\|^2_\Lambda$, respectively [a].
$b = \text{bvec}\{(\Lambda_v \odot I_M) D^2 \Lambda\}$ where $\Lambda_v = \text{diag}\{\sigma^2_{v,1}, \ldots, \sigma^2_{v,N}\}$ with $\sigma^2_{v,k}$ being a variance of the additive noise at node k. Vectors q_n and λ_ζ of size $(N^2M^2 \times 1)$ are defined as:

$$q_n = \frac{1}{N}\text{bvec}\{I_{NM}\} \quad \text{and} \quad \lambda_\zeta = \frac{1}{N}\text{bvec}\{\Lambda\}. \tag{3.39}$$

Matrix $\overline{F}$ is given by[a]:

$$\overline{F} = (\overline{G} \odot \overline{G}) \left[I_{N^2M^2} - (I_{NM} \odot \Lambda D) - (\Lambda D \odot I_{NM}) + (D \odot D)\mathcal{A} \right] \tag{3.40}$$

where $\mathcal{A} = \text{diag}\{\mathcal{A}_1, \ldots, \mathcal{A}_N\}$ with diagonal terms

$$\mathcal{A}_k = \text{diag}\{\Lambda_1 \otimes \Lambda_k, \ldots, \lambda_k \lambda_k^T + \gamma \Lambda_k \otimes \Lambda_k, \ldots, \Lambda_N \otimes \Lambda_k\} \tag{3.41}$$

In the k-th entry of (3.41) $\lambda_k \lambda_K^T$ is an outer product, where $\lambda_k = \text{vec}\{\Lambda_k\}$ and $\gamma = 2$ (assuming all data is real).

[a]Note some typographical errors in the original article [3]. In Theorem 1, normalization terms $\frac{1}{N}$ were omitted in $\eta(-1)$ and $\zeta(-1)$. Also an unnecessary σ term exists in the definition of $\overline{F}$ in Eq. 69. on page 3128. The errors were corrected in the next paper on this subject [4].

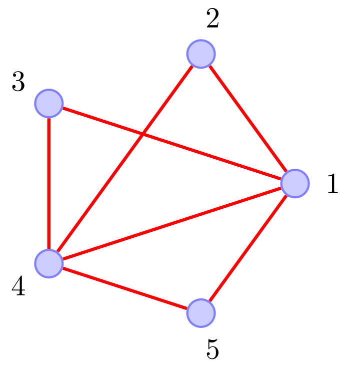
3.3.3 Experimental Results versus Theory

In order to compare experimental results and theoretical framework, evolution of MSD and EMSE was simulated using (3.37) and (3.38) of Theorem 1. A sample network composed of $N = 5$ nodes with topology shown in Fig. 3.1 was chosen for Example 1. A vector to be estimated is $w^0 = [0.3487, -0.8245,\ 0.2955,\ -0.2331]^T$ of length $M = 4$. All regression signals $u_{k,i}$ are realizations of 1-st order Gaussian-Markov process which is a discrete-time auto-regressive stochastic process obeying the following model:

$$u_{k,i} = \alpha u_{k,i-1} + \sqrt{\sigma^2_{u,k}(1-\alpha^2)}n(i) \tag{3.42}$$

where α_k is the correlation coefficient and $n(i)$ is a zero-mean unit-variance white Gaussian noise, $n(i) \sim \mathcal{N}(0,1)$. It should be noted that 1-st order Gaussian-Markov process has a correlation function $r_{u,k}(l) = \sigma^2_{u,k}\alpha_k^{|l|}$. Background noise $v_k(i)$ is a zero-mean white Gaussian noise with variance $\sigma_{v,k}$. Values of α_k, $\sigma^2_{u,k}$ and $\sigma^2_{v,k}$ are shown in Fig. 3.1. For both noncooperative and diffusion LMS algorithms the same step size $\mu_k = 0.007$ was chosen for all nodes. Diffusion LMS strategy uses Metropolis rule for combiner function. In the experimental part, noncooperative LMS algorithm (2.9) and diffusion algorithm (3.1) were implemented corresponding MSD and EMSE values were measured and average of 50 experiments was taken. In the theoretical part Eqs. 3.37 and 3.38 were evaluated using given parameters for diffusion algorithm and using $G = I_{NM}$ for noncollaborative scenario.

As illustrated in the Fig. 3.2 and Fig. 3.3 experimental results justify the correctness of Theorem 1. Both MSD and EMSE curves perfectly fit the theoretical expectation. Use of diffusion strategy improves the accuracy of the LMS algorithm, reduces MSD and EMSE by 7 dB (in other cases even larger) compared to noncooperative algorithm in the steady-state and fastens convergence.



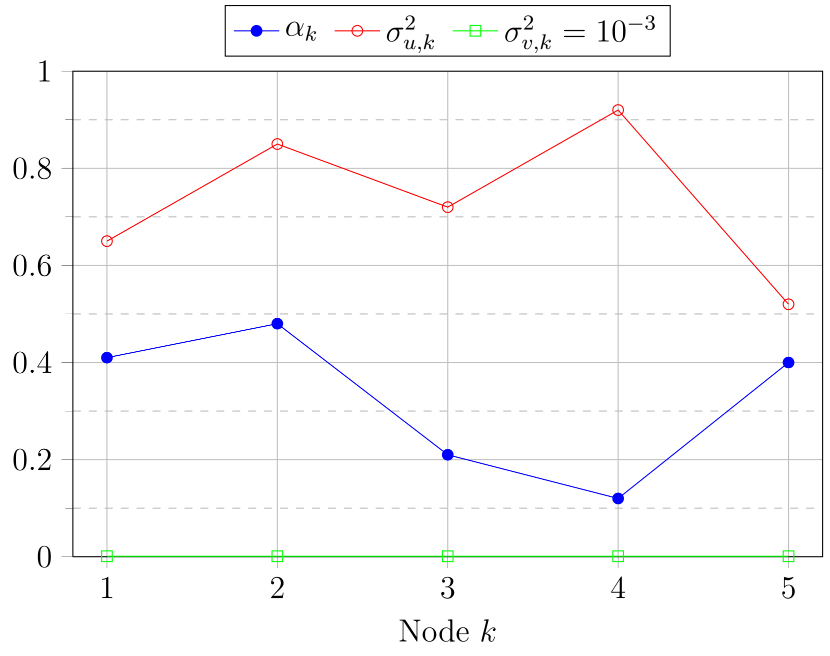


Figure 3.1: Example 1: Network topology and statistics.

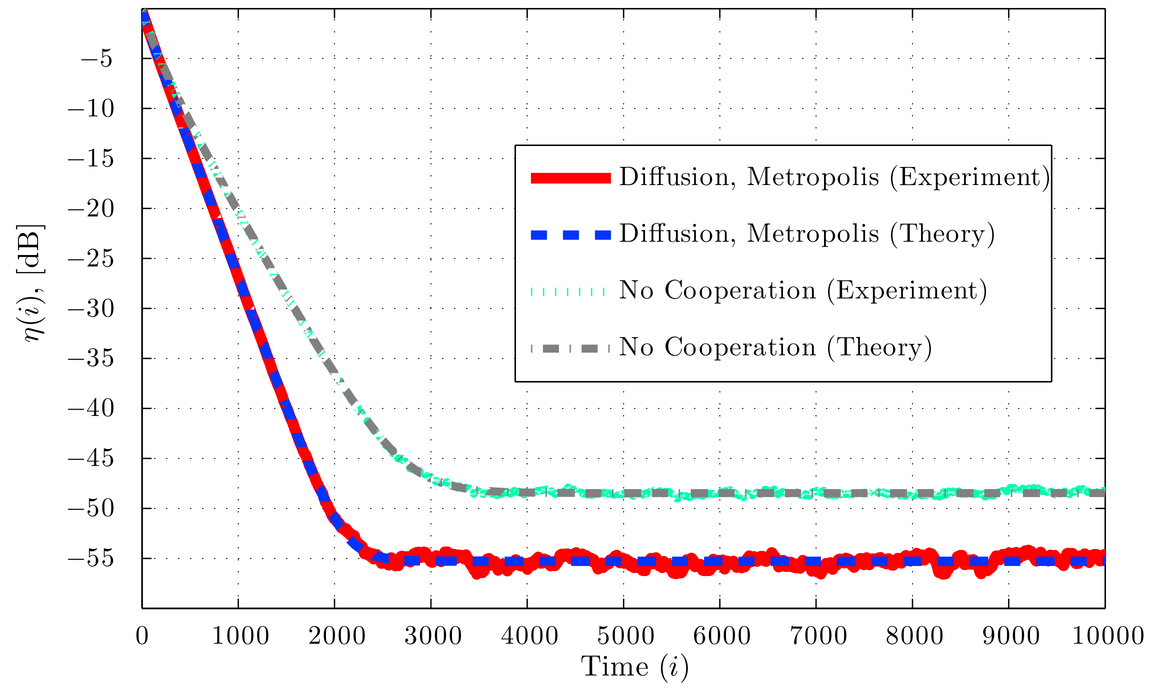


Figure 3.2: Example 1: Evolution of global mean-square deviation (MSD) for experimental (3.29) and theoretical (3.37) behavior.

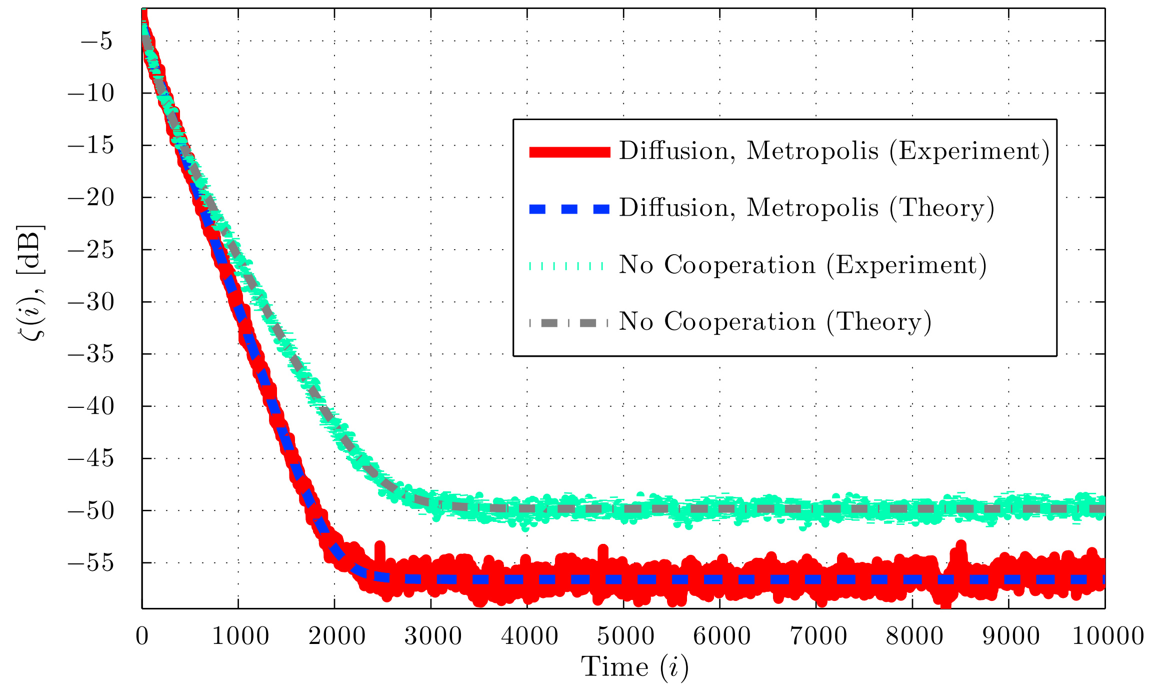


Figure 3.3: Example 1: Evolution of global excess mean-square error (EMSE) for experimental (3.30) and theoretical (3.38) behavior.

Chapter 4

ADAPTIVE DIFFUSION LEAST-MEAN SQUARES

Constant combination weights $c_{k,l}$ formed using network rules (Metropolis etc.) rely only on network topology and do not take into account different SNR values at the nodes. It would be more useful to add new level of adaption to the system, making combination weights adaptive and resistant to SNR changes in the network. The aim of the adaptive diffusion algorithms is to train weighing coefficients such that nodes with higher SNR are given more weight.

In the adaptive diffusion LMS setup, two different approaches for designing update rule for combination weights used in formation of aggregate estimate are studied. First class of algorithms is based on *pairwise combination* of the node's own estimate and averaged value of neighbors' estimates, second class of algorithms has completely adaptive nature in which *mixture* of all nodes' estimates in the neighborhood (including the node itself) is used as an aggregate estimate.

4.1 Pairwise Combination Algorithms

In this class of algorithms only two weight coefficients are adapted. The first coefficient measures contribution of the node's local estimate $\psi_k^{(i-1)}$. The other one measures the contribution of all neighbor estimates on the average. In a more detailed format, the aggregate estimate $\phi_k^{(i-1)}$ has a form of:

$$\phi_k^{i-1} = \lambda_k^{(i)} \psi_k^{(i-1)} + \widehat{\lambda}_k^{(i)} \widehat{\psi}_k^{(i-1)}, \tag{4.1}$$

$$\widehat{\psi}_k^{(i-1)} = \frac{1}{n_k - 1} \sum_{l \in \mathcal{N}_k \setminus k} \psi_l^{(i-1)}, \tag{4.2}$$

where $\lambda_k^{(i)}$ is the weight of the host node's estimate $\psi_k^{(i-1)}$ and $\widehat{\lambda}_k^{(i)}$ is the weight of the averaged value of the neighbors' estimates $\widehat{\psi}_k^{(i-1)}$. The block-diagram of the pairwise combination

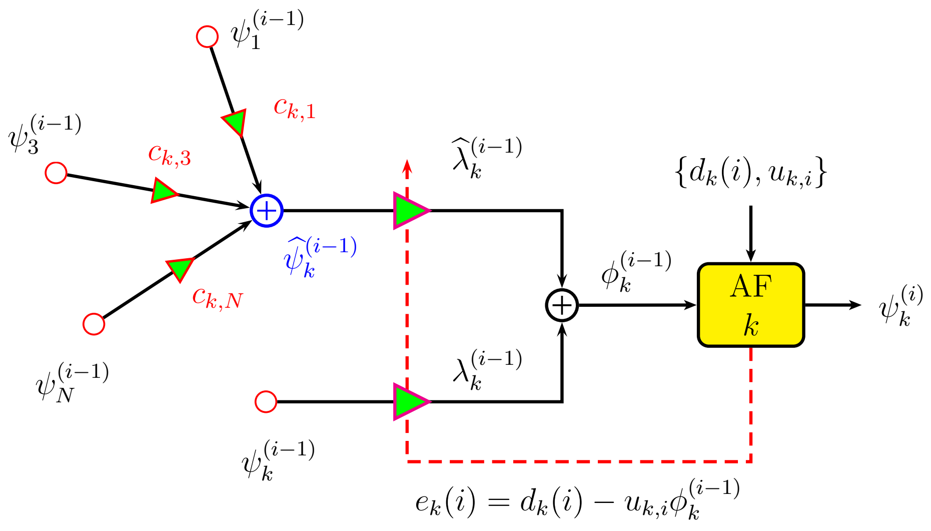


Figure 4.1: Adaptive diffusion strategy: pairwise combination setup.

setup is shown on Fig. 4.1[1].

Three possible choices of the weighing coefficients $\lambda_k^{(i)}$ and $\widehat{\lambda}_k^{(i)}$ are listed below.

1. Convex combination: any $\lambda_k^{(i)}, \widehat{\lambda}_k^{(i)}$ obeying
   $\lambda_k^{(i)} + \widehat{\lambda}_k^{(i)} = 1, \quad \lambda_k^{(i)}, \widehat{\lambda}_k^{(i)} \in (0,1);$
2. Affine combination: any $\lambda_k^{(i)}, \widehat{\lambda}_k^{(i)}$ obeying
   $\lambda_k^{(i)} + \widehat{\lambda}_k^{(i)} = 1;$
3. Unconstrained combination: any $\lambda_k^{(i)}, \widehat{\lambda}_k^{(i)}$.

The coefficients $\lambda_k^{(i)}, \widehat{\lambda}_k^{(i)}$ are updated using normalized LMS type of update. Derivation and use of adaptive diffusion algorithm with pairwise combination setup are explained in the next sections.

4.2 Mixture Algorithms

This class of algorithms contains adaptation rules for weighing coefficients of *all* nodes' estimates in the neighborhood (including the node itself), i.e. *mixture* of all estimates. The

[1]Original version of the figure can be found in [15].

aggregate estimate has a form of:

$$\phi_k^{i-1} = \sum_{l \in \mathcal{N}_k} c_{k,l}^{(i)} \psi_l^{i-1}, \tag{4.3}$$

where individual $c_{k,l}^{(i)}$ are updated at every time instant i. The block-diagram of this mixture setup is shown in Fig. 4.2. Three possible choices of the weighing coefficients $c_{k,l}^{(i)}$ are listed below.

1. Convex combination: $c_{k,l}^{(i)}$ obeying
   $\sum_{l \in \mathcal{N}_k} c_{k,l}^{(i)} = 1 \quad \text{and} \quad \forall l \in \mathcal{N}_k, \quad c_{k,l}^{(i)} \in (0,1);$
2. Affine combination: $c_{k,l}^{(i)}$ obeying
   $\sum_{l \in \mathcal{N}_k} c_{k,l}^{(i)} = 1;$
3. Unconstrained combination: $c_{k,l}^{(i)}$ with no constraint.

Adaptation rules for weight coefficients $c_{k,l}^{(i)}$ are based on normalized LMS type of algorithm. Next sections describe adaptive diffusion LMS algorithms grouped with respect to their constraints on selection of the combiner function (convex, affine, unconstrained combinations).

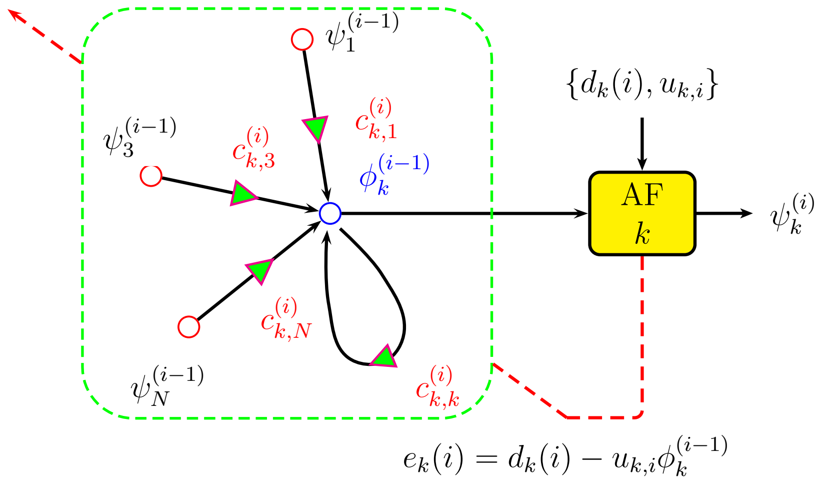


Figure 4.2: Adaptive diffusion strategy: mixture setup.

4.3 Convex Combiner Function

A convex combiner function is a linear operator which combines estimated vectors into an aggregate estimate by weighing them with nonnegative scalar coefficients $c_{k,l}^{(i)}$ which sum up to one:

$$\begin{aligned} &\forall l \in \mathcal{N}_k, \quad c_{k,l}^{(i)} \in (0,1), \\ &\sum_{l \in \mathcal{N}_k} c_{k,l}^{(i)} = 1, \end{aligned}$$

for k-th node at time index i. This means that convex combiner function completely satisties constraints peculiar to nonadaptive combiner function in classical diffusion algorithm (see Eq. 3.6).

4.3.1 Adapting Pairwise Convex Combination Weights Using the LMS Update

Aggregate estimate $\phi_k^{(i-1)}$ is obtained by summing weighted by some $\lambda_k^{(i)}$ estimate of the local node $\psi_k^{(i-1)}$ and weighted by $(1-\lambda_k^{(i)})$ average of all neighbors' estimates $\widehat{\psi}_k^{(i-1)}$ as described in [3], i.e.

$$\phi_k^{(i-1)} = \lambda_k^{(i)} \psi_k^{(i-1)} + \left(1 - \lambda_k^{(i)}\right) \widehat{\psi}_k^{(i-1)}, \tag{4.4}$$

where

$$\widehat{\psi}_k^{(i-1)} = \frac{1}{n_k - 1} \sum_{l \in \mathcal{N}_k \setminus k} \psi_l^{(i-1)}, \tag{4.5}$$

and n_k is a degree of the node k.

Obtained $\phi_k^{(i-1)}$ is used to update local estimate $\psi_k^{(i-1)}$:

$$e_k(i) = d_k(i) - u_{k,i} \phi_k^{(i-1)}, \tag{4.6}$$

$$\psi_k^{(i)} = \phi_k^{(i-1)} + \mu_k u_{k,i}^T e_k(i). \tag{4.7}$$

In order to satisfy convex criterion of the combination, the weight $\lambda_k^{(i)}$ should be in the range of $[0,1]$. This condition is guarantied if $\lambda_k^{(i)}$ is a sigmoidal function of some parameter a_k:

$$\lambda_k^{(i)} = \frac{1}{1 + \exp\left(-a_k^{(i-1)}\right)}. \tag{4.8}$$

Updating λ_k indirectly is achievable by training a_k which should minimize the local mean-square error:

$$a_k^{(i)} = a_k^{(i-1)} - \frac{\mu_a}{2} \frac{1}{\|u_{k,i}\|^\beta} \left(\frac{\partial e_k^2(i)}{\partial a_k^{(i-1)}} \right) \tag{4.9}$$

$$a_k^{(i)} = a_k^{(i-1)} - \mu_a \frac{e_k(i)}{\|u_{k,i}\|^\beta} \left(\frac{\partial e_k(i)}{\partial a_k^{(i-1)}} \right), \tag{4.10}$$

where β is a nonnegative coefficient inserted for normalization purpose. By applying chain rule we obtain:

$$\frac{\partial e_k(i)}{\partial a_k^{(i-1)}} = \frac{\partial e_k(i)}{\partial \lambda_k^{(i)}} \frac{\partial \lambda_k^{(i)}}{\partial a_k^{(i-1)}}, \tag{4.11}$$

with first term calculated from (4.4):

$$\frac{\partial e_k(i)}{\partial \lambda_k^{(i)}} = -u_{k,i} \left[\psi_k^{(i-1)} - \widehat{\psi}_k^{(i-1)} \right] \tag{4.12}$$

and second term computed using (4.8),

$$\frac{\partial \lambda_k^{(i)}}{\partial a_k^{(i-1)}} = \lambda_k^{(i-1)}(1 - \lambda_k^{(i-1)}). \tag{4.13}$$

After substituting (4.12) and (4.13), the adaptation rule for a_k takes a form of:

$$\begin{aligned} h_k^{(i)} &= \left(u_{k,i} \left[\psi_k^{(i-1)} - \widehat{\psi}_k^{(i-1)} \right] \right) e_k(i) \lambda_k^{(i)} (1 - \lambda_k^{(i)}) \\ a_k^{(i)} &= a_k^{(i-1)} + \mu_a \frac{1}{||u_{k,i}||^\beta} h_k^{(i)}. \end{aligned} \tag{4.14}$$

In summary, the pairwise convex combination adaptive diffusion alogrithm is depicted using following equations:

$$\boxed{\begin{aligned}
\lambda_k^{(i)} &= \frac{1}{1+\exp\left(-a_k^{(i-1)}\right)} \\
\widehat{\psi}_k^{(i-1)} &= \frac{1}{n_k-1}\sum_{l\in\mathcal{N}_k\setminus k}\psi_l^{(i-1)} \\
\phi_k^{(i-1)} &= \lambda_k^{(i)}\psi_k^{(i-1)} + \left(1-\lambda_k^{(i)}\right)\widehat{\psi}_k^{(i-1)} \\
e_k(i) &= d_k(i) - u_{k,i}\phi_k^{(i-1)} \\
h_k^{(i)} &= \left(u_{k,i}\left[\psi_k^{(i-1)} - \widehat{\psi}_k^{(i-1)}\right]\right)e_k(i)\lambda_k^{(i)}(1-\lambda_k^{(i)}) \\
a_k^{(i)} &= a_k^{(i-1)} + \mu_a\frac{1}{||u_{k,i}||^{\beta}}h_k^{(i)} \\
\psi_k^{(i)} &= \phi_k^{(i-1)} + \mu_k u_{k,i}^T e_k(i).
\end{aligned} \qquad (4.15)}$$

4.3.2 Adapting Convex Mixture Weights Using the LMS Update

Contrary to pairwise combination depicted previously in (4.4), this method of aggregating estimates gives *individual* adaptive weights to all neighbor estimates. The adaptation rules used for this purpose are taken from [16] and [17].

Combiner weights are exponential functions of another parameter $a_{k,l}^{(i-1)}$ which are normalized to satisfy convex criterion:

$$c_{k,l}^{(i)} = \frac{\exp\left(a_{k,l}^{(i-1)}\right)}{\sum\limits_{m\in\mathcal{N}_k}\exp\left(a_{k,m}^{(i-1)}\right)}. \qquad (4.16)$$

The parameter $a_{k,l}^{(i-1)}$ is updated using the LMS algorithm:

$$a_{k,l}^{(i)} = a_{k,l}^{(i-1)} - \frac{\mu_a}{2}\frac{1}{\|u_{k,i}\|^{\beta}}\left(\frac{\partial e_k^2(i)}{\partial a_{k,l}^{(i-1)}}\right) \qquad (4.17)$$

$$a_{k,l}^{(i)} = a_{k,l}^{(i-1)} - \mu_a\frac{e_k(i)}{\|u_{k,i}\|^{\beta}}\left(\frac{\partial e_k(i)}{\partial a_{k,l}^{(i-1)}}\right), \qquad (4.18)$$

where β is a nonnegative coefficient inserted for normalization purpose. By using chain rule the partial derivative takes a form of:

$$\frac{\partial e_k(i)}{\partial a_{k,l}^{(i-1)}} = \sum_{m\in\mathcal{N}_k} \frac{\partial e_k^{(i)}}{\partial c_{k,m}^{(i)}} \frac{\partial c_{k,m}^{(i)}}{\partial a_{k,l}^{(i-1)}} \tag{4.19}$$

with first term

$$\frac{\partial e_k^{(i)}}{\partial c_{k,m}^{(i)}} = -u_{k,i}\psi_m^{(i-1)} \tag{4.20}$$

and second term

$$\frac{\partial c_{k,m}^{(i)}}{\partial a_{k,l}^{(i-1)}} = \frac{\partial}{\partial a_{k,l}^{(i-1)}} \left(\frac{\exp\left(a_{k,m}^{(i-1)}\right)}{\sum_{p\in\mathcal{N}_k} \exp\left(a_{k,p}^{(i-1)}\right)} \right), \tag{4.21}$$

After taking the derivative the second terms becomes:

$$\frac{\partial c_{k,m}^{(i)}}{\partial a_{k,l}^{(i-1)}} = \begin{cases} c_{k,l}^{(i)}(1-c_{k,l}^{(i)}), & \text{if } m = l, \\ -c_{k,l}^{(i)}c_{k,m}^{(i)}, & \text{if } m \neq l. \end{cases} \tag{4.22}$$

The final adaptation rule is obtained by summarizing 4.3, 4.6, 4.7, 4.16 and above equations:

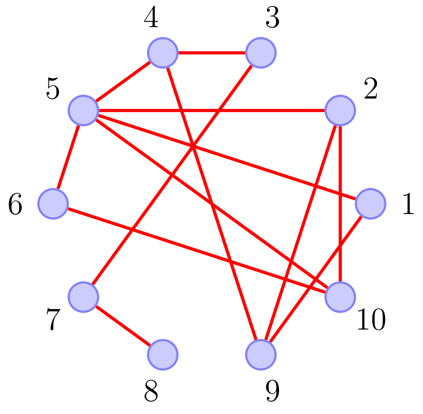
$$\boxed{\begin{aligned} c_{k,l}^{(i)} &= \frac{\exp\left(a_{k,l}^{(i-1)}\right)}{\sum\limits_{m\in\mathcal{N}_k} \exp\left(a_{k,m}^{(i-1)}\right)} \\ \phi_k^{i-1} &= \sum_{l\in\mathcal{N}_k} c_{k,l}^{(i)}\psi_l^{i-1} \\ e_k(i) &= d_k(i) - u_{k,i}\phi_k^{(i-1)} \\ h_{k,l}^{(i)} &= u_{k,i}\left[\psi_l^{(i-1)} - \phi_k^{(i-1)}\right] e_k(i)c_{k,l}^{(i)} \\ a_{k,l}^{(i)} &= a_{k,l}^{(i-1)} + \mu_a \frac{1}{||u_{k,i}||^{\beta}} h_{k,l}^{(i)} \\ \psi_k^{(i)} &= \phi_k^{(i-1)} + \mu_k u_{k,i}^T e_k(i). \end{aligned}} \tag{4.23}$$

4.3.3 Simulations

In this experiment (Example 2) performance of adaptive diffusion algorithms with convex combination constraint are compared. Network topology is shown on the left side of Fig. 4.3 on page 26. Parameters of the networks (α_k, $\sigma^2_{u,k}$, $\sigma^2_{v,k}$ - see Example 1) are given on the right side of the same figure. A 10×1 vector to be estimated is given below:

$$w^0 = [2.3459, 0.0893, 2.2103, 0.7440, 0.6762, -0.4959, 1.0007, -1.8874, -1.2499, -0.2327]^T.$$

The same step size $\mu_k = 0.007$ at all nodes was chosen for all diffusion strategies. In all evaluations average of 25 experiments is taken. The goal of this experiment is to see whether the adaptive strategies improve the accuracy of the agreggate estimate $\phi_k^{(i-1)}$ which consequently affects the performance of the core LMS algorithm. Insertion of the second adaptation layer introduces a source of *gradient noise* [3] due to interdependency created by collaboration among nodes. MSD performance of the adaptive strategies are compared to classical diffusion algorithm with Metropolis rule used as a combiner function. As seen from Fig. 4.4, with a small penalty of ∼1 dB (@54 dB) difference in MSD in steady-state adaptive diffusion with pairwise convex combination constraint converges to steady-state faster than classical diffusion LMS by 1000 time samples. If convergence is a priority, adaptive diffusion algorithm with convex mixture constraint learns even faster (2000 time sample difference) but with higher penalty of 3 dB (@54 dB). In Fig. 4.5 evolution of particuar $\lambda_k^{(i)}$ of the adaptive diffusion with pairwise covex combination constraint tuned with $\mu_a = 5$ and $\beta = 6$ for some nodes is shown. Similarly, evolution of $c_{k,l}$ weights of the adaptive diffusion algorithm with convex mixture constraint for $\mu_a = 2$, $\beta = 5$ and $k = 4$ is shown. Node k is connected to three neighbors: nodes 3,5, and 9 (see Fig. 4.3). The coefficients convergence to steady values during first 1500 time samples. In Fig. 4.7 and 4.8 evolution of global MSD errors for adaptive diffusion LMS with pairwise convex combiner function is shown. In Fig. 4.7, normalization exponent $\beta = 2$ is kept constant and effect of changing μ_a is observed. Increasing the step size μ_a, slightly fastens convergence but leads to higher MSD values. In Fig. 4.8 the step size μ_a is kept constant and performance for different β's is observed. Increasing normalization exponent β leads to lowering of MSD level and fastens convergence at the same time. The same observation is valid for convex mixture case (see Fig. 4.9 and 4.10).



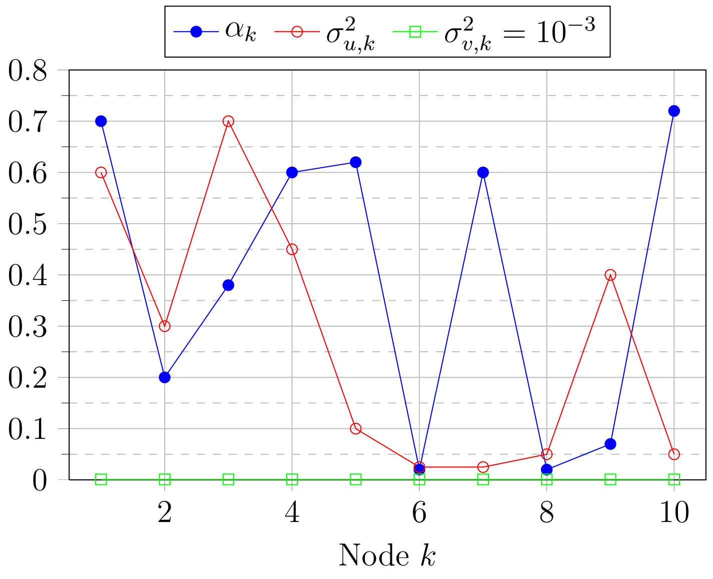


Figure 4.3: Example 2: Network topology (left) and statistics (right).

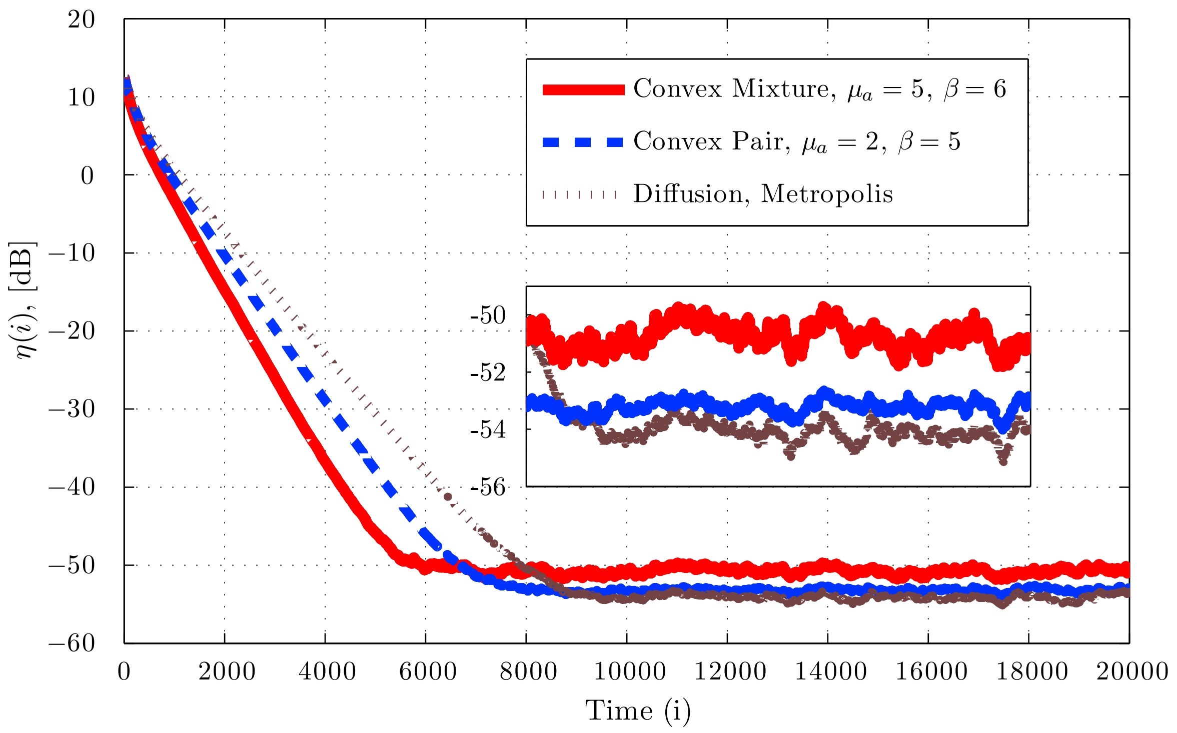


Figure 4.4: Example 2: Evolution of global mean-square deviation (MSD) (3.29) for adaptive diffusion LMS with convex combiner functions.

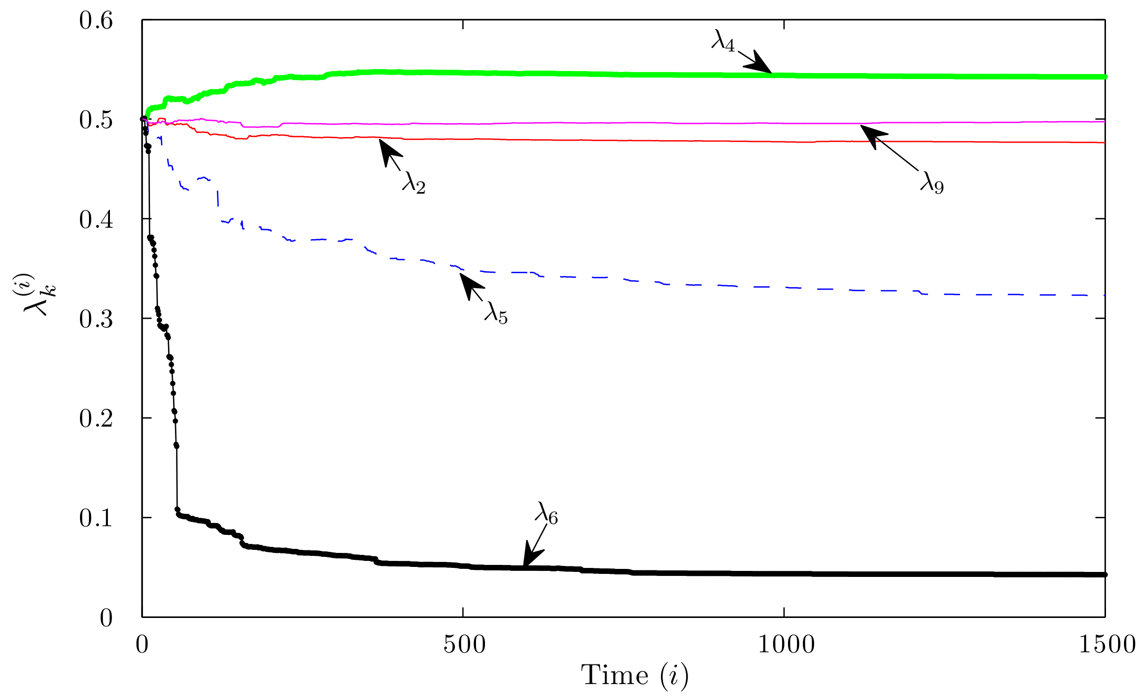


Figure 4.5: Example 2: Evolution of weight coefficients $\lambda_k^{(i)}$ for pairwise convex combination with $\mu_a = 2$ and $\beta = 5$.

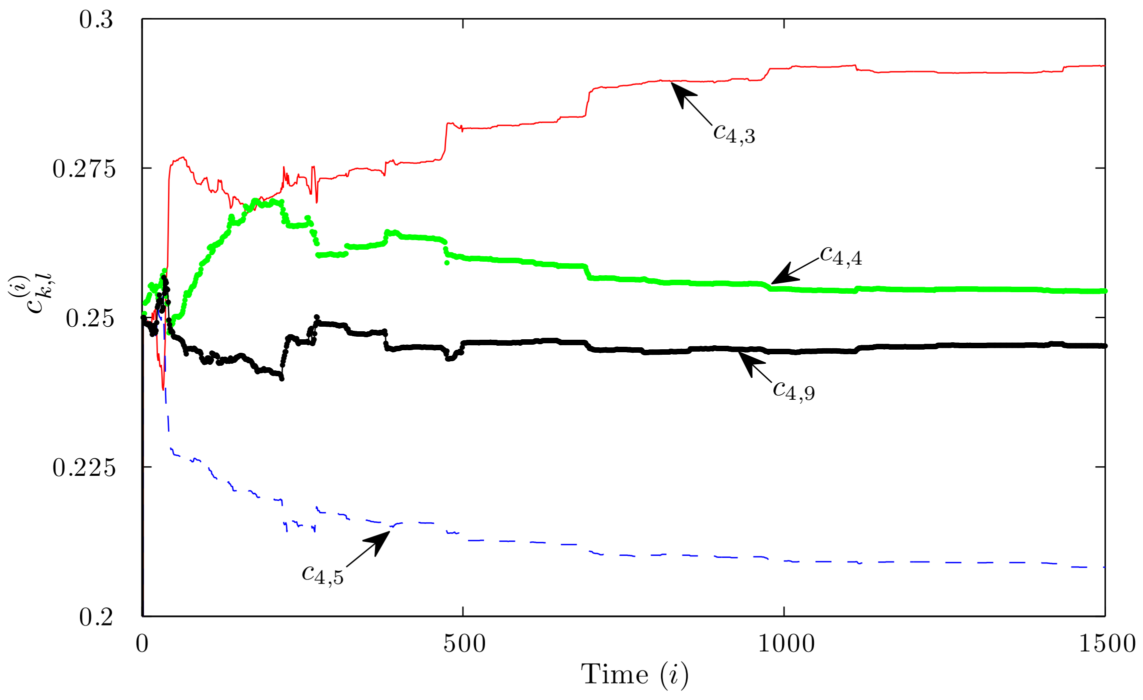


Figure 4.6: Example 2: Evolution of weight coefficients $c_{k,l}^{(i)}$ for $k = 4$ of convex mixture with $\mu_a = 5$ and $\beta = 6$.

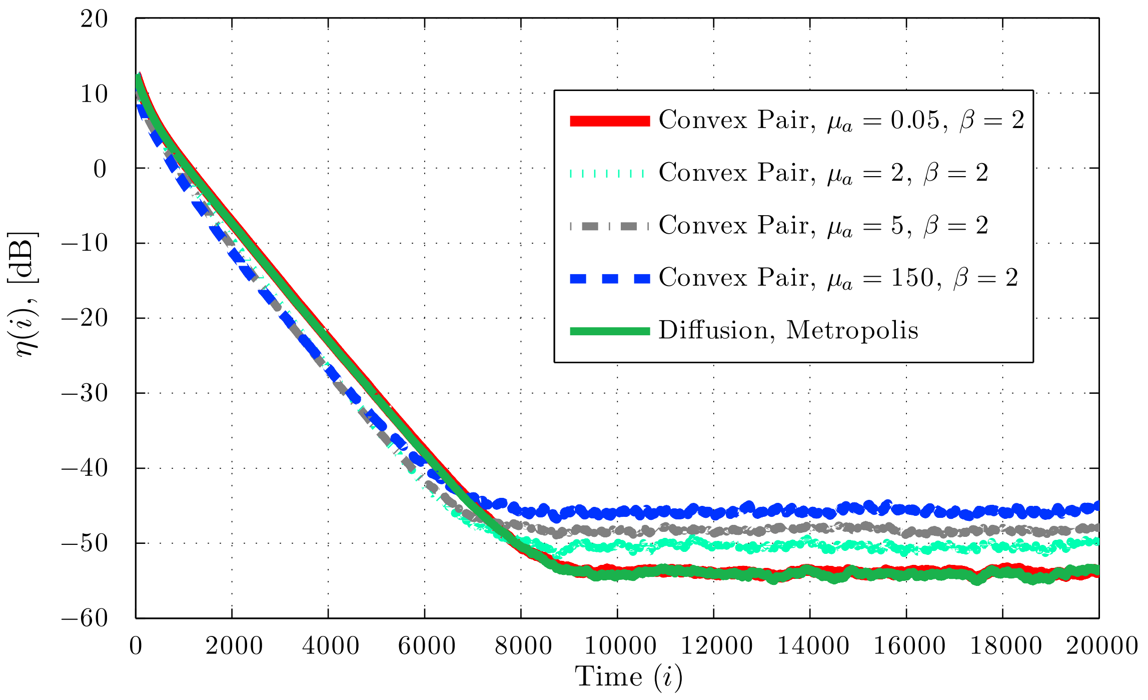


Figure 4.7: Example 2: Evolution of global mean-square deviation (MSD) (3.29) for adaptive diffusion LMS with pairwise convex combiner function with constant $\beta = 2$.

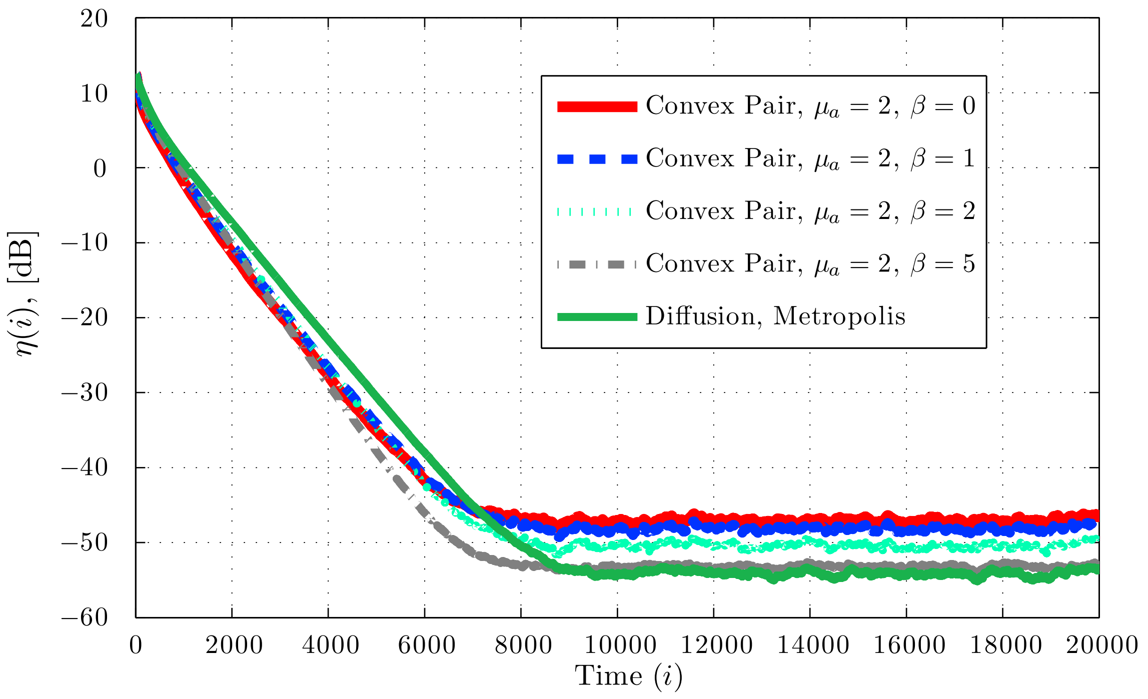


Figure 4.8: Example 2: Evolution of global mean-square deviation (MSD) (3.29) for adaptive diffusion LMS with pairwise convex combiner function with constant $\mu_a = 2$.

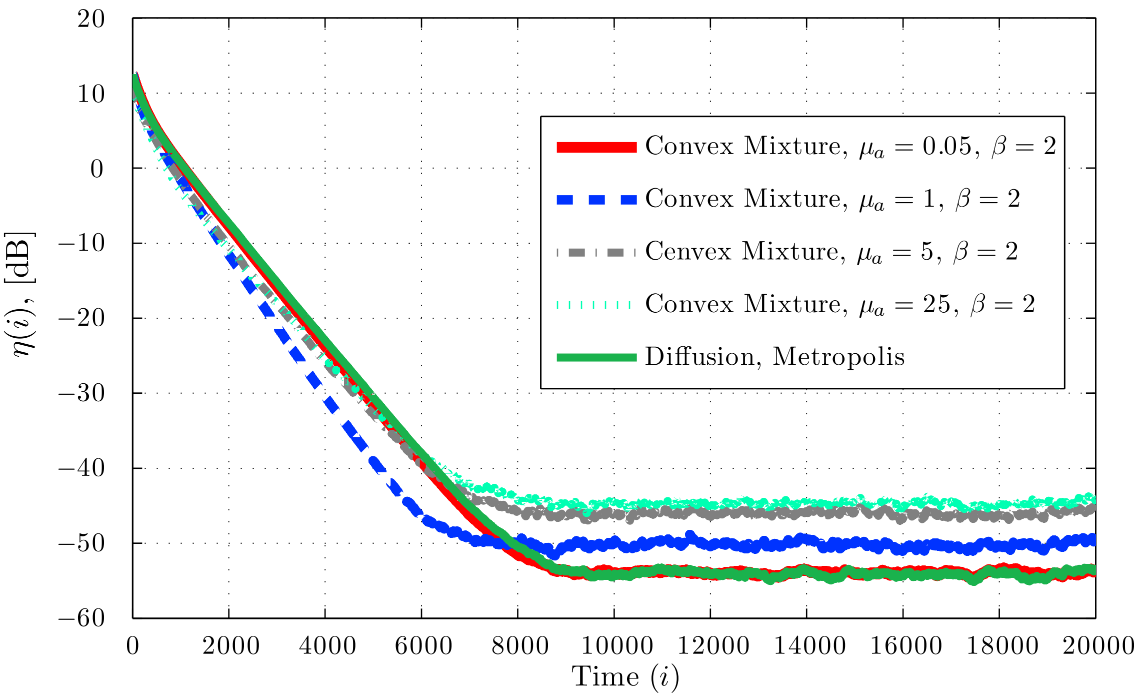


Figure 4.9: Example 2: Evolution of global mean-square deviation (MSD) (3.29) for adaptive diffusion LMS with convex mixture combiner function with constant $\beta = 2$.

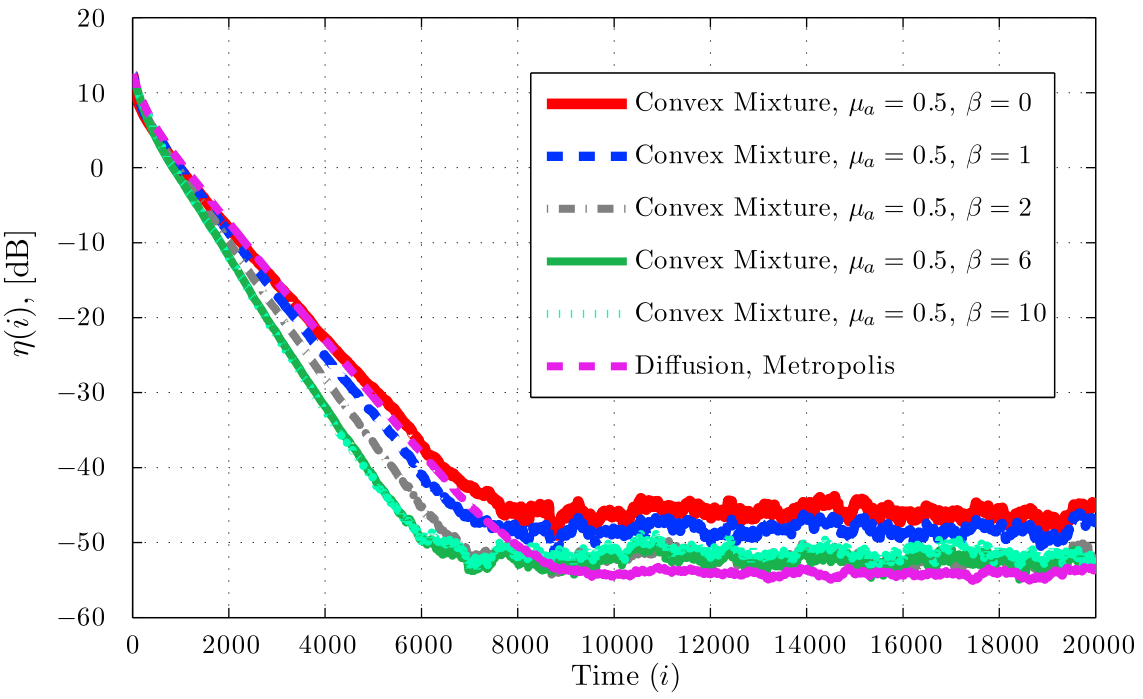


Figure 4.10: Example 2: Evolution of global mean-square deviation (MSD) (3.29) for adaptive diffusion LMS with convex mixture combiner function with constant $\mu_a = 0.5$.

4.4 Affine Combiner Function

An affine combiner function is a linear operator which combines estimated vectors into an aggregate estimate by weighing them with scalar coefficients $c_{k,l}^{(i)}$ which sum up to one:

$$\sum_{l \in \mathcal{N}_k} c_{k,l}^{(i)} = 1,$$

for k-th node at time index i. This implies that affine combiner function partially satisties constraints specific to nonadaptive combiner function in classical diffusion algorithm (see 3.6).

4.4.1 Adapting Pairwise Affine Combination Weights Using the LMS Update

Aggregate function used in this algorithm is similar to those in (4.4). The adaptation rule for $\lambda_k^{(i-1)}$ is given as:

$$\begin{aligned}
\lambda_k^{(i)} &= \lambda_k^{(i-1)} - \frac{\mu_{af}}{2} \frac{1}{\|u_{k,i}\|^\beta} \left(\frac{\partial e_k^2(i)}{\partial \lambda_k^{(i-1)}} \right) \\
\lambda_k^{(i)} &= \lambda_k^{(i-1)} - \mu_{af} \frac{e_k(i)}{\|u_{k,i}\|^\beta} \left(\frac{\partial e_k(i)}{\partial \lambda_k^{(i-1)}} \right) \\
\lambda_k^{(i)} &= \lambda_k^{(i-1)} + \mu_{af} \frac{e_k(i)}{\|u_{k,i}\|^\beta} u_{k,i} \left[\psi_k^{(i-1)} - \widehat{\psi}_k^{(i-1)} \right],
\end{aligned}$$

where β is a nonnegative coefficient inserted for normalization purpose. Combining the above equation with (4.4), (4.5), (4.6) and (4.7) yields the adaptation rule:

$$\boxed{\begin{aligned}
\widehat{\psi}_k^{(i-1)} &= \frac{1}{n_k - 1} \sum_{l \in \mathcal{N}_k \setminus k} \psi_l^{(i-1)} \\
\phi_k^{(i-1)} &= \lambda_k^{(i-1)} \psi_k^{(i-1)} + \left(1 - \lambda_k^{(i-1)}\right) \widehat{\psi}_k^{(i-1)} \\
e_k(i) &= d_k(i) - u_{k,i} \phi_k^{(i-1)} \\
\psi_k^{(i)} &= \phi_k^{(i-1)} + \mu_k u_{k,i}^T e_k(i) \\
\lambda_k^{(i)} &= \lambda_k^{(i-1)} + \mu_{af} \frac{e_k(i)}{\|u_{k,i}\|^\beta} u_{k,i} \left[\psi_k^{(i-1)} - \widehat{\psi}_k^{(i-1)} \right].
\end{aligned} \qquad (4.24)}$$

4.4.2 Adapting Affine Mixture Weights Using the LMS Update

For this diffusion strategy affine combination algorithm described in [18] is adopted. The aggregate estimate $\phi_k^{(i-1)}$ is formed using the following equation:

$$\begin{aligned}\phi_k^{(i-1)} &= \sum_{l\in\mathcal{N}_k\setminus k} c_{k,l}^{(i-1)}\psi_l^{(i-1)} + \left(1-\sum_{l\in\mathcal{N}_k\setminus k} c_{k,l}^{(i-1)}\right)\psi_k^{(i-1)}\\ \phi_k^{(i-1)} &= \psi_k^{(i-1)} + \sum_{l\in\mathcal{N}_k\setminus k} c_{k,l}^{(i-1)}\left[\psi_l^{(i-1)} - \psi_k^{(i-1)}\right].\end{aligned} \tag{4.25}$$

Coeffcients $c_{k,l}$ are updated according to the rule given below:

$$\begin{aligned}c_{k,l}^{(i)} &= c_{k,l}^{(i-1)} - \frac{\mu_{af}}{2}\frac{1}{\|u_{k,i}\|^{\beta}}\left(\frac{\partial e_k^2(i)}{\partial c_{k,l}^{(i-1)}}\right)\\ c_{k,l}^{(i)} &= c_{k,l}^{(i-1)} - \mu_{af}\frac{e_k(i)}{\|u_{k,i}\|^{\beta}}\left(\frac{\partial e_k(i)}{\partial c_{k,l}^{(i-1)}}\right)\\ c_{k,l}^{(i)} &= c_{k,l}^{(i-1)} + \mu_{af}\frac{e_k(i)}{\|u_{k,i}\|^{\beta}}u_{k,i}\left[\psi_l^{(i-1)} - \psi_k^{(i-1)}\right],\end{aligned}$$

where β is a nonnegative coefficient inserted for normalization purpose. The final adaptation rule is obtained by summarizing 4.3, 4.6, 4.7 and the above equations:

$$\boxed{\begin{aligned}\phi_k^{(i-1)} &= \sum_{l\in\mathcal{N}_k\setminus k} c_{k,l}^{(i-1)}\psi_l^{(i-1)} + \left(1-\sum_{l\in\mathcal{N}_k\setminus k} c_{k,l}^{(i-1)}\right)\psi_k^{(i-1)}\\ e_k(i) &= d_k(i) - u_{k,i}\phi_k^{(i-1)}\\ \psi_k^{(i)} &= \phi_k^{(i-1)} + \mu_k u_{k,i}^T e_k(i)\\ &\forall\ l\in\mathcal{N}_k\setminus k \quad \text{update}\\ c_{k,l}^{(i)} &= c_{k,l}^{(i-1)} + \mu_{af}\frac{e_k(i)}{\|u_{k,i}\|^{\beta}}u_{k,i}\left[\psi_l^{(i-1)} - \psi_k^{(i-1)}\right].\end{aligned} \qquad (4.26)}$$

4.4.3 Simulations

In this experiment the same setup of Example 2 is used including network topology, network statistics (see Fig. 4.3 on page 26), w^0 and universal $\mu_k = 0.007$.

In Fig. 4.11 and 4.12 evolution of global MSD errors for adaptive diffusion LMS with pairwise affine combiner function is shown. In Fig. 4.11, normalization exponent $\beta = 2$ is kept constant and effect of changing μ_{af} is observed. Increasing the step size μ_{af}, slightly fastens convergence but leads to higher MSD values. In Fig. 4.12 the step size μ_{af} is kept constant and performance for different β's is observed. Increasing normalization exponent β leads to lowering of MSD level and fastens convergence at the same time. In mixture case, changes in performance are more noticeable. When β is kept constant, increasing the value of μ_{af} interestingly slows down convergence and decreases MSD error (see Fig. 4.13). If the step size μ_{af} is kept constant and normalization exponent β is increased the convergence fastens with penalty of increasing MSD value (see Fig. 4.14). In Fig. 4.15 evolution of particuar $\lambda_k^{(i)}$ of the adaptive diffusion with pairwise affine combination constraint tuned with $\mu_{af} = 0.01$ and $\beta = 2$ for some nodes is shown. Similarly, in Fig. 4.16 evolution of $c_{k,l}$ weights of the adaptive diffusion algorithm with affine mixture constraint for $\mu_{af} = 0.03$, $\beta = 2$ and $k = 4$ is shown. Node k is connected to three neighbors: nodes 3,5, and 9 (see Fig. 4.3). The coefficients convergence to steady values during first 2000 time samples.

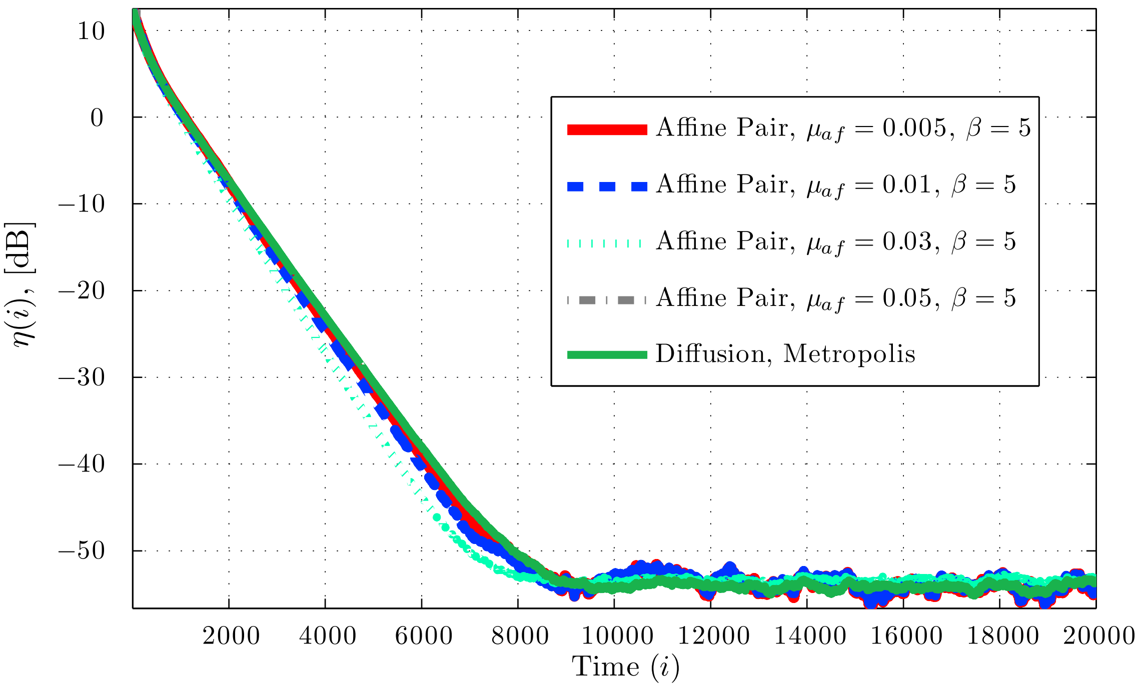


Figure 4.11: Example 2: Evolution of global mean-square deviation (MSD) (3.29) for adaptive diffusion LMS with pairwise affine combiner function with constant $\beta = 5$.

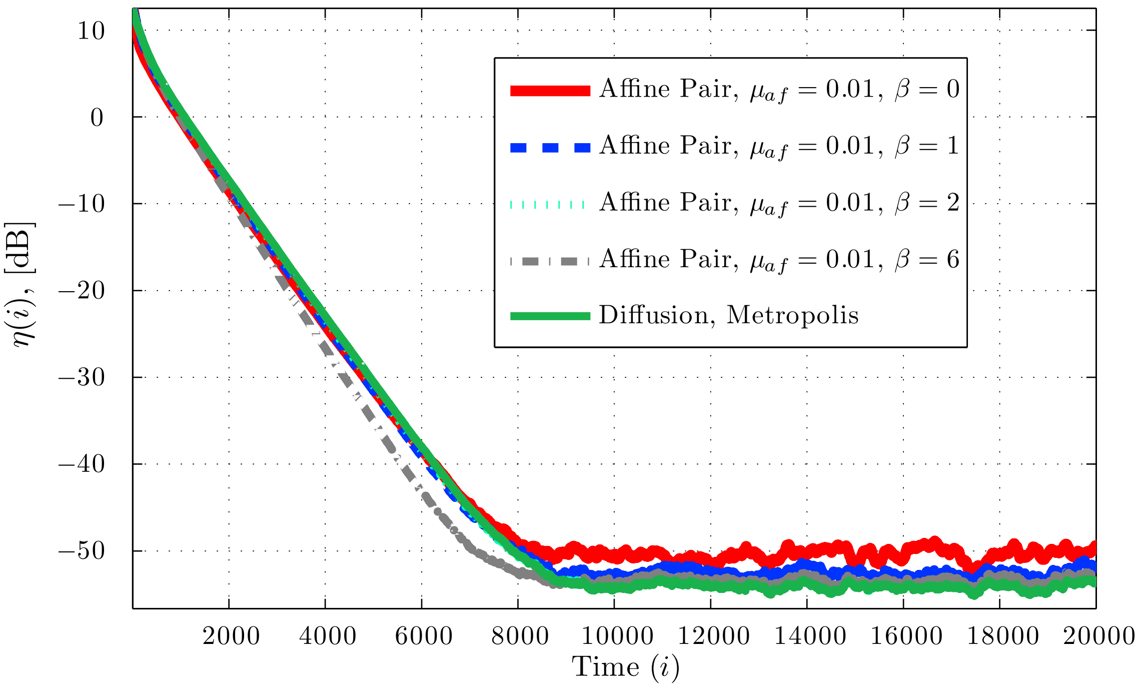


Figure 4.12: Example 2: Evolution of global mean-square deviation (MSD) (3.29) for adaptive diffusion LMS with pairwise affine combiner function with constant $\mu_{af} = 0.01$.

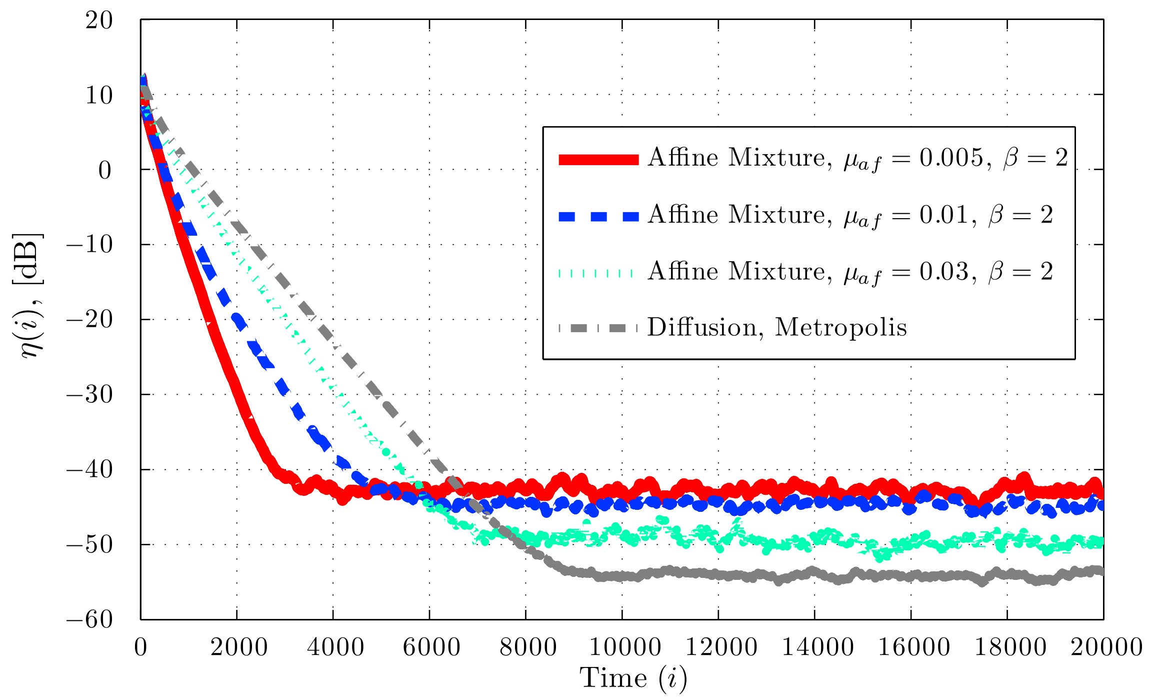


Figure 4.13: Example 2: Evolution of global mean-square deviation (MSD) (3.29) for adaptive diffusion LMS with affine mixture combiner function with constant $\beta = 2$.

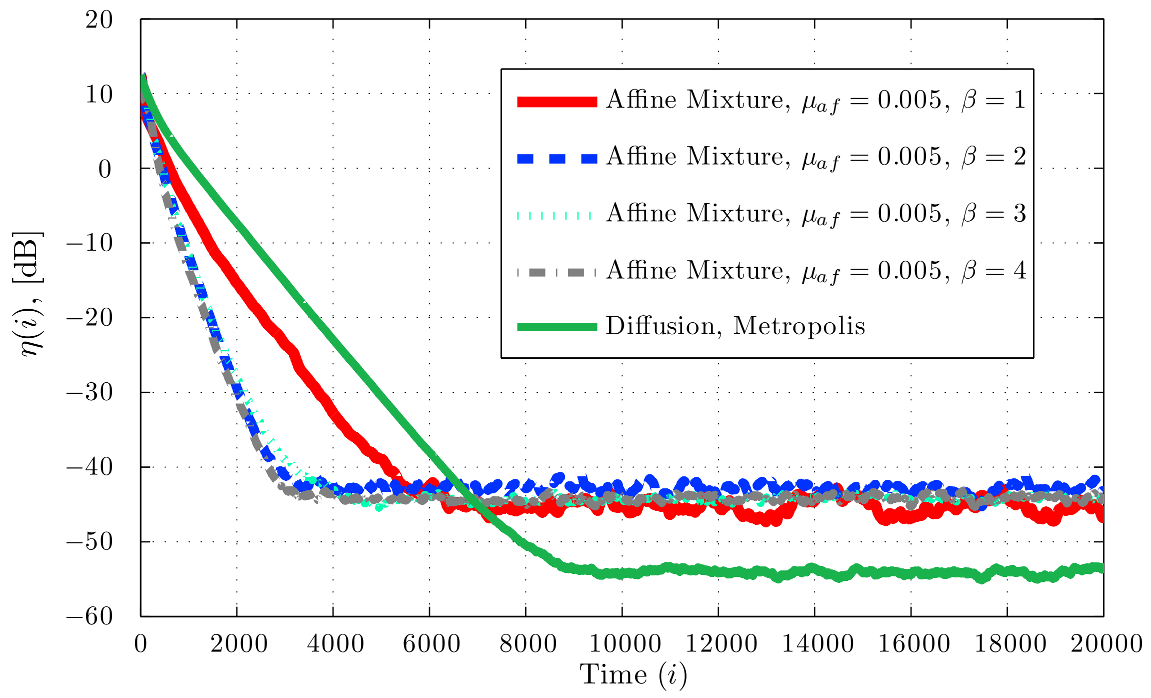


Figure 4.14: Example 2: Evolution of global mean-square deviation (MSD) (3.29) for adaptive diffusion LMS with affine mixture combiner function with constant $\mu_{af} = 0.005$.

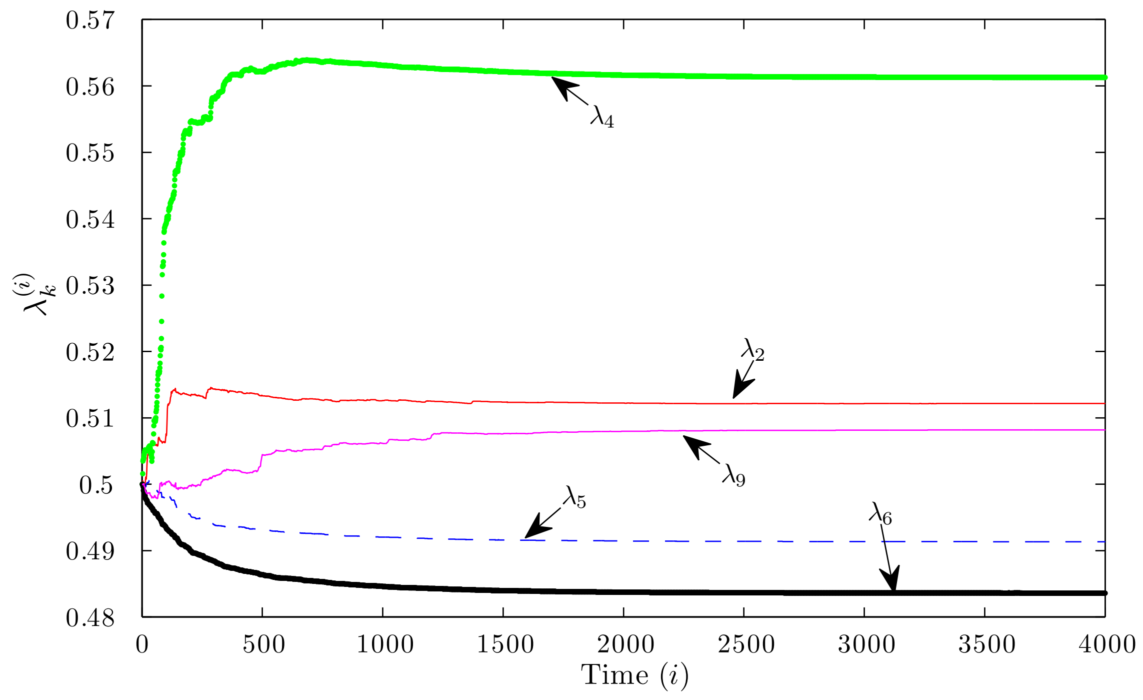


Figure 4.15: Example 2: Evolution of weight coefficients $\lambda_k^{(i)}$ for pairwise affine combination with $\mu_{af} = 0.01$ and $\beta = 2$.

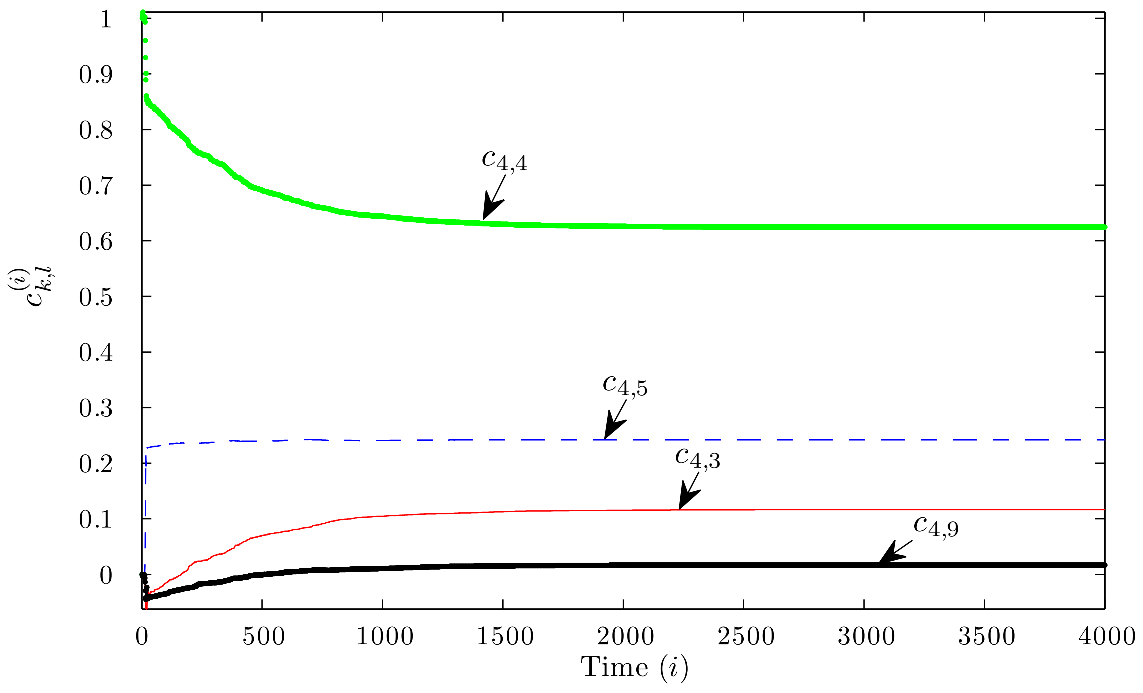


Figure 4.16: Example 2: Evolution of weight coefficients $c_{k,l}^{(i)}$ for $k = 4$ of affine mixture with $\mu_{af} = 0.03$ and $\beta = 2$.

4.5 Unconstrained Combiner Function

An unconstrained combiner function is a linear operator which combines estimated vectors into an aggregate estimate by weighing them with scalar coefficients $c_{k,l}^{(i)}$ which do not carry any constraint. This implies that unconstrained combiner function does not satisty constraints specific to nonadaptive combiner function in classical diffusion algorithm (see 3.6).

4.5.1 Adapting Pairwise Unconstrained Combination Weights Using the LMS Update

This algorithm does not put any constraint on combination weights $\lambda_k^{(i-1)}, \widehat{\lambda}_k^{(i-1)}$. The weights are updated using LMS alogrithm [17]:

$$\begin{bmatrix} \lambda_k^{(i)} \\ \widehat{\lambda}_k^{(i)} \end{bmatrix} = \begin{bmatrix} \lambda_k^{(i-1)} \\ \widehat{\lambda}_k^{(i-1)} \end{bmatrix} - \frac{\mu_{un}}{2} \frac{1}{\|u_{k,i}\|^{\beta}} \begin{bmatrix} \frac{\partial e_k^2(i)}{\partial \lambda_k^{(i-1)}} \\ \frac{\partial e_k^2(i)}{\partial \widehat{\lambda}_k^{(i-1)}} \end{bmatrix}$$

$$\begin{bmatrix} \lambda_k^{(i)} \\ \widehat{\lambda}_k^{(i)} \end{bmatrix} = \begin{bmatrix} \lambda_k^{(i-1)} \\ \widehat{\lambda}_k^{(i-1)} \end{bmatrix} - \frac{\mu_{un}}{2} \frac{e_k(i)}{\|u_{k,i}\|^{\beta}} \begin{bmatrix} \frac{\partial e_k(i)}{\partial \lambda_k^{(i-1)}} \\ \frac{\partial e_k(i)}{\partial \widehat{\lambda}_k^{(i-1)}} \end{bmatrix}$$

$$\begin{bmatrix} \lambda_k^{(i)} \\ \widehat{\lambda}_k^{(i)} \end{bmatrix} = \begin{bmatrix} \lambda_k^{(i-1)} \\ \widehat{\lambda}_k^{(i-1)} \end{bmatrix} + \mu_{un} \frac{e_k(i)}{\|u_{k,i}\|^{\beta}} \begin{bmatrix} u_{k,i}\psi_k^{(i-1)} \\ u_{k,i}\widehat{\psi}_k^{(i-1)} \end{bmatrix},$$

where β is a nonnegative coefficient inserted for normalization purpose. Combining the above equation with (4.1), (4.5), (4.6) and (4.7) yields the adaptation rule:

$$\boxed{\begin{aligned} \widehat{\psi}_k^{(i-1)} &= \frac{1}{n_k - 1} \sum_{l \in \mathcal{N}_k \setminus k} \psi_l^{(i-1)} \\ \phi_k^{(i-1)} &= \lambda_k^{(i-1)} \psi_k^{(i-1)} + \widehat{\lambda}_k^{(i-1)} \widehat{\psi}_k^{(i-1)} \\ e_k(i) &= d_k(i) - u_{k,i}\phi_k^{(i-1)} \\ \psi_k^{(i)} &= \phi_k^{(i-1)} + \mu_k u_{k,i}^T e_k(i) \\ \begin{bmatrix} \lambda_k^{(i)} \\ \widehat{\lambda}_k^{(i)} \end{bmatrix} &= \begin{bmatrix} \lambda_k^{(i-1)} \\ \widehat{\lambda}_k^{(i-1)} \end{bmatrix} + \mu_{un} \frac{e_k(i)}{\|u_{k,i}\|^{\beta}} \begin{bmatrix} u_{k,i}\psi_k^{(i-1)} \\ u_{k,i}\widehat{\psi}_k^{(i-1)} \end{bmatrix}. \end{aligned} \qquad (4.27)}$$

4.5.2 Adapting Unconstrained Mixture Weights Using the LMS Update

This algorithm is similar to pairwise unconstrained combination update except that adaptation is extended to all weights.

$$\begin{aligned}
c_{k,l}^{(i)} &= c_{k,l}^{(i-1)} - \frac{\mu_{un}}{2}\frac{1}{\|u_{k,i}\|^{\beta}}\left(\frac{\partial e_k^2(i)}{\partial c_{k,l}^{(i-1)}}\right) \\
c_{k,l}^{(i)} &= c_{k,l}^{(i-1)} - \mu_{un}\frac{e_k(i)}{\|u_{k,i}\|^{\beta}}\left(\frac{\partial e_k(i)}{\partial c_{k,l}^{(i-1)}}\right) \\
c_{k,l}^{(i)} &= c_{k,l}^{(i-1)} + \mu_{un}\frac{e_k(i)}{\|u_{k,i}\|^{\beta}}u_{k,i}\psi_l^{(i-1)}
\end{aligned}$$

where β is a nonnegative coefficient inserted for normalization purpose. The final adaptation rule is obtained by summarizing 4.3, 4.6, 4.7 and the above equations:

$$\boxed{\begin{aligned}
\phi_k^{(i-1)} &= \sum_{l\in\mathcal{N}_k} c_{k,l}^{(i-1)}\psi_l^{(i-1)} \\
e_k(i) &= d_k(i) - u_{k,i}\phi_k^{(i-1)} \\
\psi_k^{(i)} &= \phi_k^{(i-1)} + \mu_k u_{k,i}^T e_k(i) \\
c_{k,l}^{(i)} &= c_{k,l}^{(i-1)} + \mu_{un}\frac{e_k(i)}{\|u_{k,i}\|^{\beta}}u_{k,i}\psi_l^{(i-1)}
\end{aligned}} \tag{4.28}$$

4.5.3 Simulations

In this experiment the same setup of Example 2 is used including network topology, network statistics (see Fig. 4.3 on page 26), w^0 and universal $\mu_k = 0.007$.

In Fig. 4.17 and 4.18 evolution of global MSD errors for adaptive diffusion LMS with pairwise unconstrained combiner function is shown. In Fig. 4.17, normalization exponent $\beta = 1$ is kept constant and effect of changing μ_{un} is observed. Increasing the step size μ_{un}, does not fasten convergence but leads to higher MSD values. In Fig. 4.18 the step size μ_{un} is kept constant and performance for different β's is observed. Increasing normalization exponent β also increases MSD. For unconstrained mixture case, increasing μ_{un} and keeping β constant in some cases fastens convergence (see Fig. 4.19). However, algorithm diverges for small μ_{un} and different β's (see Fig. 4.20). In Fig. 4.21 evolution of particuar $\lambda_k^{(i)}$ of the adaptive diffusion with pairwise unconstrained combination tuned with $\mu_{un} = 0.005$ and $\beta = 1$ for some nodes is shown. Abrupt changes in evolution of λ_k's are noticable. Similarly, in Fig. 4.22 evolution of $c_{k,l}$ weights of the adaptive diffusion with unconstrained mixture tuned with $\mu_{un} = 0.001$ and $\beta = 2$ for $k = 4$ is shown. Node k is connected to three neighbors: nodes 3,5, and 9 (see Fig. 4.3). The coefficients convergence to steady values during first 10000 time samples.

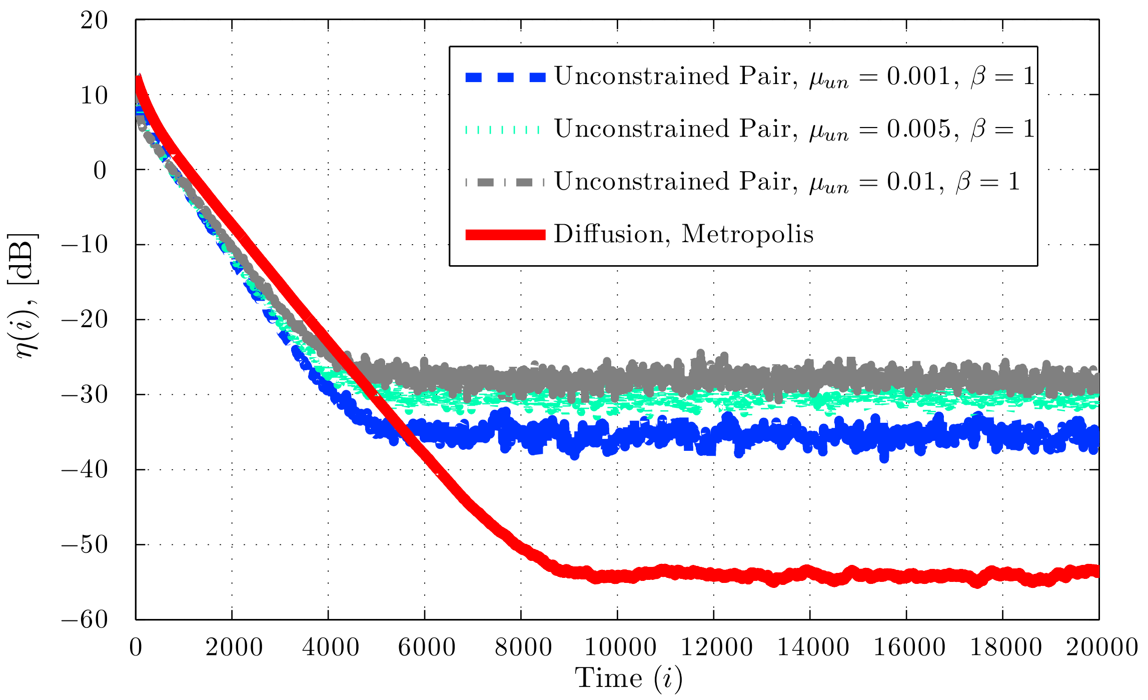


Figure 4.17: Example 2: Evolution of global mean-square deviation (MSD) (3.29) for adaptive diffusion LMS with pairwise unconstrained combiner function with constant $\beta = 1$.

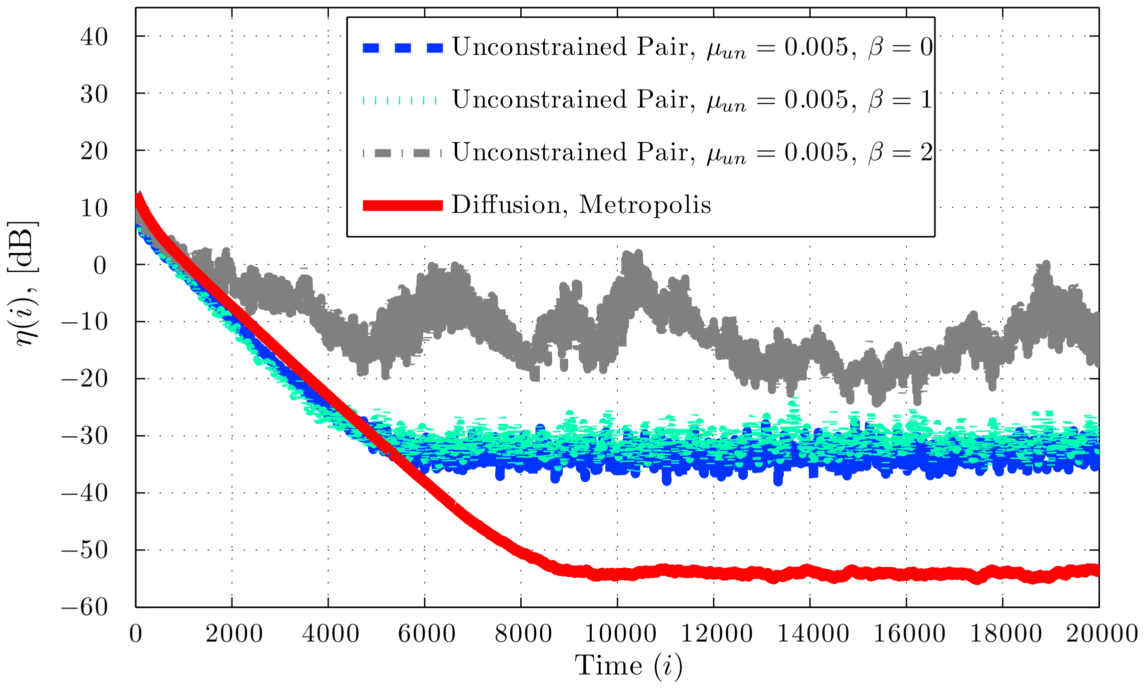


Figure 4.18: Example 2: Evolution of global mean-square deviation (MSD) (3.29) for adaptive diffusion LMS with pairwise unconstrained combiner function with constant $\mu_{un} = 0.005$.

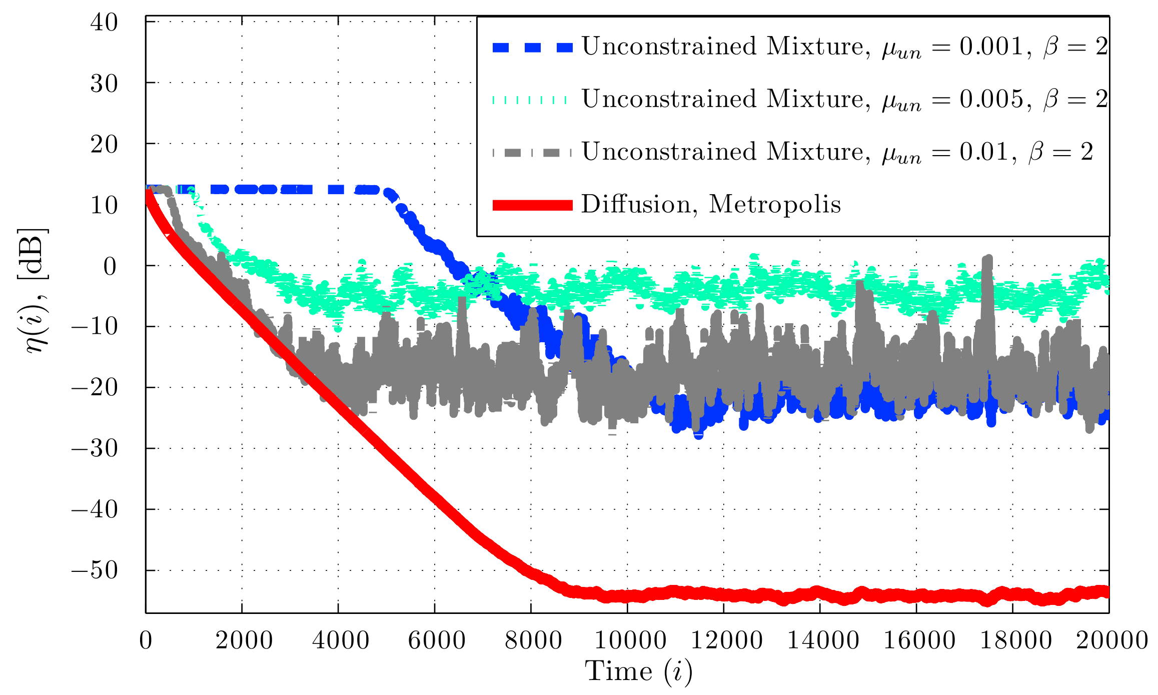


Figure 4.19: Example 2: Evolution of global mean-square deviation (MSD) (3.29) for adaptive diffusion LMS with unconstrained mixture combiner function with constant $\beta = 2$.

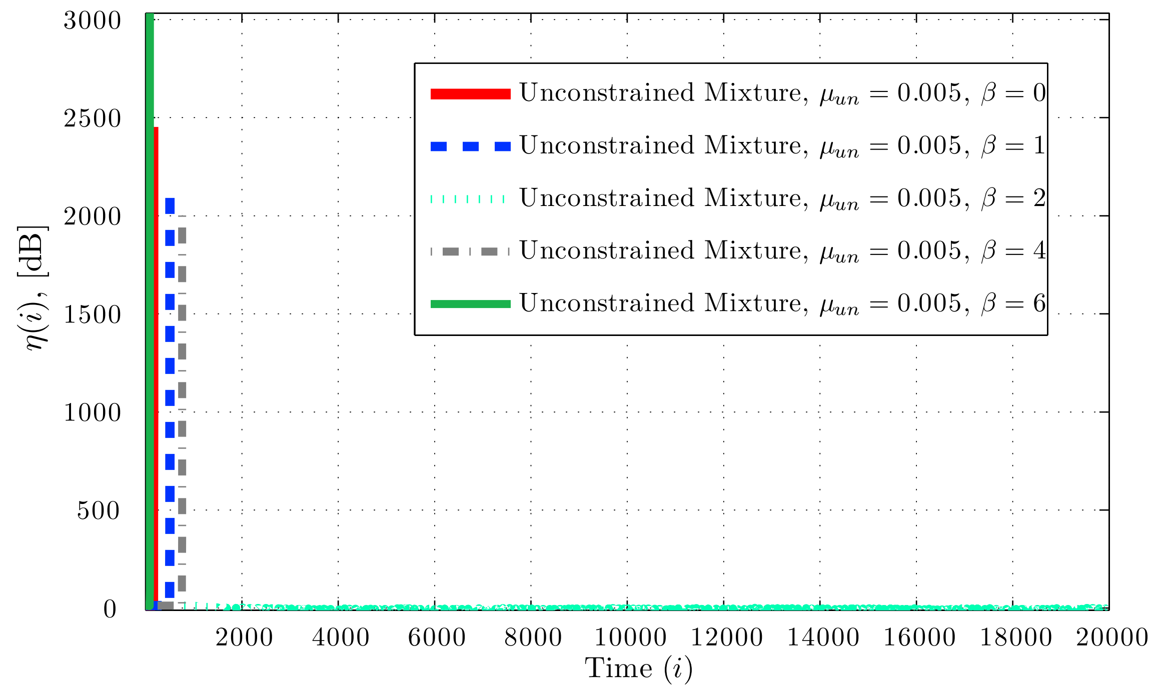


Figure 4.20: Example 2: Evolution of global mean-square deviation (MSD) (3.29) for adaptive diffusion LMS with unconstrained mixture combiner function with constant $\mu_{un} = 0.005$.

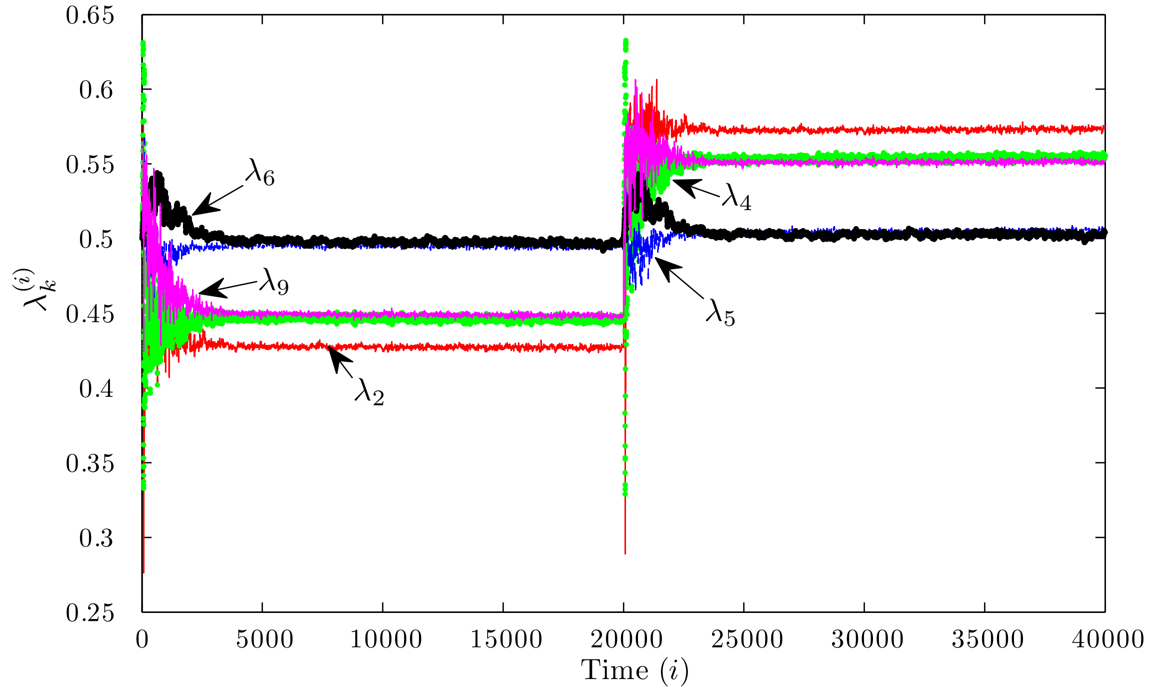


Figure 4.21: Example 2: Evolution of weight coefficients $\lambda_k^{(i)}$ for pairwise unconstrained combination with $\mu_{un} = 0.005$ and $\beta = 1$.

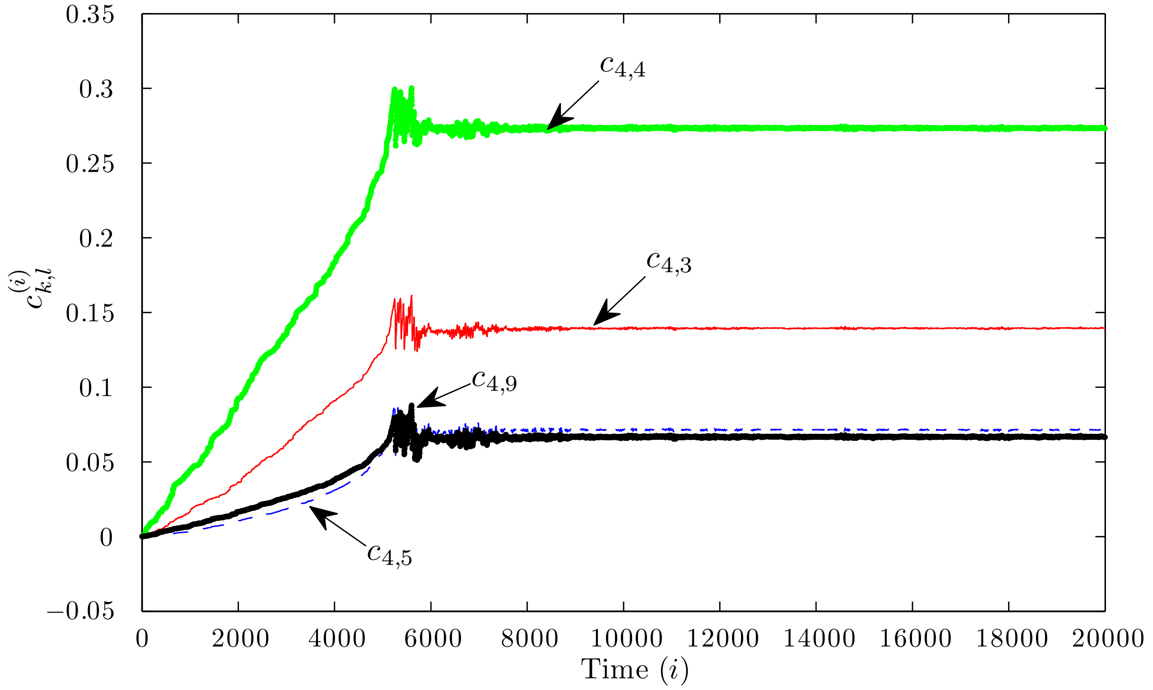


Figure 4.22: Example 2: Evolution of weight coefficients $c_{k,l}^{(i)}$ for $k = 4$ of unconstrained mixture with $\mu_{un} = 0.001$ and $\beta = 2$.

Chapter 5

CONCLUSION

In this thesis, mean-square transient analysis of MSD and EMSE behavior for diffusion algorithm described in [3] was implemented. Simulations showed exact match between theoretical and experimental evaluations of this behavior. Also it was justified that diffusion strategies have better MSD and EMSE performance compared to noncooperative LMS algorithm. Five new alogrithms introduced here are highly parameter-dependent and do not converge to the same MSD and EMSE values in the steady-state when tuned up differently. This fact makes the comparison with classical diffusion algorithm unfair. However, experimental results showed that in some cases adaptive diffusion LMS algorithm with convex and affine combination constraints may have faster convergence than nonadaptive diffusion algorithm reaching the same MSD and EMSE values in the steady-state. Also it should be noted that adaptive diffusion algorithm with unconstrained combiner function demonstrates always greater MSD and EMSE values due to dissatisfaction of condition in (3.7).

The main result of this research is that second layer of adaptation does not necessarily improve performance of the diffusion strategy. Although motivatation is very logical: nodes with *higher* SNR, and hence, with more precise estimates, should contribute *more* to formation the aggregate estimate, other way to get rid of gradient noise introduces by second adaptation layer should be investigated. On the other hand, other classes of algorithms such as variable-step size LMS can be used for improvement of the MSD and EMSE performance of diffusion strategies [19, 20].

BIBLIOGRAPHY

[1] Dan Li, Kerry D. Wong, Yu H. Hu, and Akbar M. Sayeed. Detection, classification and tracking of targets in distributed sensor networks. In *IEEE Signal Processing Magazine*, pages 17–29, 2002.

[2] D. Estrin, L. Girod, G. Pottie, and M. Srivastava. Instrumenting the world with wireless sensor networks. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 2033–2036, 2001.

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

[4] Noriyuki Takahashi, Isao Yamada, and Ali H. Sayed. Diffusion least-mean squares with adaptive combiners: formulation and performance analysis. *IEEE Transactions on Signal Processing*, 58:4795–4810, September 2010.

[5] Jerónimo Arenas-García, Aníbal R. Figueiras-Vidal, and Ali H. Sayed. Mean-square performance of a convex combination of two adaptive filters. *IEEE Transactions on Signal Processing*, 54:1078–1090, 2006.

[6] Süleyman S. Kozat, Alper T. Erdoğan, Andrew C. Singer, and Ali H. Sayed. Steady-state MSE performance analysis of mixture approaches to adaptive filtering. *IEEE Transactions on Signal Processing*, 58(8):4050–4063, August 2010.

[7] Ali H. Sayed. *Fundamentals of Adaptive Filtering.* New York: Wiley, 2003.

[8] Federico S. Cattivelli and Ali H. Sayed. Diffusion LMS strategies for distributed estimation. *IEEE Transactions on Signal Processing*, 2010.

[9] L. Xiao and S. Boyd. Fast linear iterations for distributed averaging. *Systems and Control Letters*, 53(1):65–78, September 2004.

[10] R. Olfati-Saber and R. M. Murray. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Transactions on Automatic Control*, 49:1520–1533, September 2004.

[11] A. Jadbabaie, J. Lin, and A. S. Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48(6), June 2003.

[12] Carl D. Meyer. *Matrix Analysis and Applied Linear Algebra.* SIAM, 2000.

[13] D. S. Tracy and R. P. Singh. A new matrix product and its applications in partitioned matrix differentiation. *Statistica Neerlandica*, 26(4):143–157, 1972.

[14] Ruud H. Koning, Heinz Neudecker, and Tom Wansbeek. Block Kronecker products and the vecb operator. Technical Report 351, Institure of Economic Research, University of Groningen, Groningen, The Netherlands, January 1990.

[15] Cassio G. Lopes and Ali H. Sayed. Diffusion least-mean squares over adaptive networks. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 3, pages 917–920, Honolulu, Hawaii, April 2007.

[16] Jerónimo Arenas-García, Manel Martínez-Ramón, Vanessa Gómez-Verdejo, and Aníbal R. Figueriras-Vidal. Multiple plant identifier via adaptive LMS convex combination. In *Proc. of the 2003 IEEE International Symposium on Intelligent Signal Processing*, pages 137–142, Budapest, Hungary, September 2003.

[17] Süleyman S. Kozat and Andrew C. Singer. Multi-stage adaptive signal processing algorithms. In *in Proc. Sensor Array and Multichannel Processing Workshop*, pages 380–384, 2000.

[18] Luis Antonio Azpicueta-Ruiz, Marcus Zeller, Aníbal R. Figueiras-Vidal, and Jerónimo Arenas-García. Least-squares adaptation of affine combinations of multiple adaptive

filters. In *IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 2976–2979, Paris, France, May 2010. IEEE.

[19] Muhammad Omer Bin Saeed, Azzedine Zerguine, and Salam A. Zummo. Variable step-size least-mean square algorithms over adaptive networks. In *10th International Conference on Information Science, Signal Processing and their Applications (ISSPA)*, pages 381–384, May 2010.

[20] Hyun-Chool Shin, Ali H. Sayed, and Woo-Jin Song. Variable step-size NLMS and affine projection algorithms. *IEEE Signal Processing Letters*, 2, February 2004.