EURASIP Journal on Advances in Signal Processing
a SpringerOpen Journal

RESEARCH Open Access

A variable step-size strategy for distributed estimation over adaptive networks

Muhammad O Bin Saeed, Azzedine Zerguine* and Salam A Zummo

Abstract

A lot of work has been done recently to develop algorithms that utilize the distributed structure of an *ad hoc* wireless sensor network to estimate a certain parameter of interest. One such algorithm is called diffusion least-mean squares (DLMS). This algorithm estimates the parameter of interest using the cooperation between neighboring sensors within the network. The present work proposes an improvement on the DLMS algorithm by using a variable step-size LMS (VSSLMS) algorithm. In this work, first, the well-known variants of VSSLMS algorithms are compared with each other in order to select the most suitable algorithm which provides the best trade-off between performance and complexity. Second, the detailed convergence and steady-state analyses of the selected VSSLMS algorithm are performed. Finally, extensive simulations are carried out to test the robustness of the proposed algorithm under different scenarios. Moreover, the simulation results are found to corroborate the theoretical findings very well.

Keywords: Variable step-size least mean square algorithm; Diffusion least mean square algorithm; Distributed network

1 Introduction

The advent of *ad-hoc* wireless sensor networks has created renewed interest in distributed computing and opened up new avenues for research in the areas of estimation and tracking of parameters of interest where a robust, scalable, and low-cost solution is required. To illustrate this point clearly, consider a set of N sensor nodes spread over a wide geographic area, as shown in Figure 1. Each node takes sensor measurements at every sampling instant. The goal here is to estimate a certain unknown parameter of interest using these sensed measurements. In a centralized network, all nodes transmit their readings to a fusion center where the processing takes place. However, this system is prone to any failure of the fusion center. Furthermore, large amounts of energy and communication resources may be required for the complete signal processing task between the center and the entire network to be successfully carried out. These resources needed could become considerable, as the distance between the nodes and the center increases [1].

An *ad hoc* network, on the other hand, depends on distributed processing with the nodes communicating only with their neighbors and the processing taking place at the nodes themselves. As no hierarchical structure is involved, any node failure would not result in the failure of the entire network. Extensive research has been done in the field of consensus-based distributed signal processing and resulted in a variety of algorithms [2].

In this work, a brief overview of the virtues and limitations of these algorithms [3-10] is conducted, thus providing the background against which our contribution is justified. Incremental algorithms organize the network in a Hamiltonian cycle [3]. The estimation is completed by passing the estimate from node to node, improving the accuracy of the estimate with each new set of data per node. This algorithm is termed the incremental least mean squares (ILMS) algorithm as it uses the LMS algorithm [11] with incremental steps within each iteration [3]. A general incremental stochastic gradient descent algorithm was developed in [4], for which [3] can be considered a special case. These algorithms are heavily dependent on the Hamiltonian cycle and in case of a node failure, a new cycle has to be initiated. The problem of reconstructing the cycle is non-deterministic polynomial-time hard [12]. A quantized version of the ILMS algorithm was suggested in [5]. A fully distributed algorithm was proposed in [6], called diffusion least mean squares

*Correspondence: azzedine@kfupm.edu.sa
Electrical Engineering Department, King Fahd University of Petroleum and Minerals, Dhahran 31261, Saudi Arabia

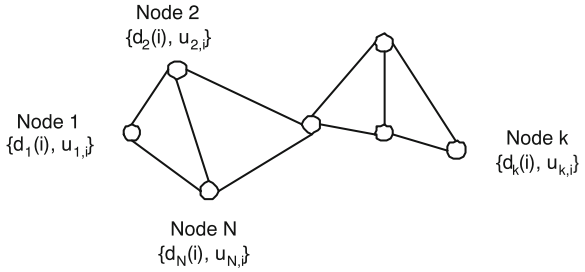

Figure 1 Adaptive network of *N* nodes.

(DLMS) algorithm. In the DLMS algorithm, neighbor nodes share their estimates in order to improve the overall performance; see also [7]. Ultimately, this algorithm is robust to node failure as the network is not dependent on any single node. On the other hand, the authors in [8] introduced a scheme to adapt the combination weights at each iteration for each node instead of having as fixed weights for the shared data. In this case, the performance is improved if more weight is given to the estimates of the neighbor nodes that are providing more improvement per iteration.

All the previously mentioned algorithms are generally based on a non-constrained optimization technique, except in [8] which uses constrained optimization to adapt the weights only. However, the authors in [9] use the constraint that all nodes converge to the same steady-state estimate to derive the distributed LMS algorithm. This algorithm is unfortunately hierarchical, thus making it complex and not completely distributed. To remedy this situation, a fully distributed algorithm based on the same constraint was suggested in [10]. The algorithms in [9,10] have been shown to be robust to inter-sensor noise, a property not possessed by the diffusion-based algorithms. However, it has been shown in [7] that diffusion-based algorithms perform better in general.

All the above-mentioned algorithms use a fixed step-size. In general, the step-size is kept the same for all nodes. As a result, the nodes with better signal-to-noise ratio (SNR) may converge quicker and provide a reasonably good performance. However, the nodes with poor SNR will not provide similar performance despite cooperation from neighboring nodes. As a result, a distributed algorithm may provide improvement in average performance but individually, some nodes will still not be performing similarly to the other nodes. One solution for this problem was provided by [8]. The work in [8] provides a computationally complex method to improve the performance of the DLMS algorithm. For every iteration, an error correlation matrix is formed for each node. A decision is made based on this decision as to the weights of the neighbors. Thus, the combiner weights are adapted at every iteration according to the performance of the neighbors of each node. Simulation results from [8] have shown slight improvement in the performance, but this has been achieved at the cost of high computational complexity.

In comparison, a much simpler solution was suggested in [13], using a variable step-size LMS algorithm. This resulted in the variable step-size diffusion LMS (VSSDLMS) algorithm. Preliminary results showed remarkable improvement in performance at a reasonably low computational cost. The idea is to vary the step-size at each iteration based on the error performance. Each node will alter its step-size according to its own individual performance. As a result, not only the average performance improves remarkably but the individual performances of the nodes also get better.

A different diffusion-based algorithm was proposed in [14] using the recursive least squares (RLS) algorithm to obtain the diffusion RLS (DRLS) algorithm. This DRLS algorithm provided exceptional results in both speed and performance. Another RLS-based distributed estimation algorithm has been studied in [15,16]. The latter algorithm is hierarchical in nature, which makes its complexity higher than that of the DRLS algorithm. The RLS algorithm is inherently far more complex compared with the LMS algorithm. In this work, it is shown that despite the LMS algorithm being inferior to the RLS algorithm, using a variable step-size allows the LMS algorithm to achieve performance very close to that of the RLS algorithm.

In order to achieve better performance, various other algorithms have been proposed in the literature, such as in [17-19]. The works in [17,18] propose distributed Kalman filtering algorithms that provide efficient solutions for several applications. A survey of distributed particle filtering is provided in [19]. This work takes a look at several solutions proposed for nonlinear distributed estimation. However, the focus of this work is primarily on LMS-based algorithms, so these algorithms will not be included in any further discussion.

Our work extends that of [13] and investigates in detail the performance of the VSSDLMS algorithm. Here, the most popular variable step-size LMS (VSSLMS) algorithms are first investigated and compared with each other. Based on the best complexity-performance trade-off, one variant of the VSSLMS algorithms is chosen for the proposed algorithm. Next, the incorporation of the selected VSSLMS algorithm in the diffusion scheme is described, and complete convergence and steady-state analyses are carried out. The stability of the algorithm is also analyzed. Finally, some simulations are carried out to first determine which of the various selected VSSLMS algorithms provide the best trade-off between performance and complexity, and then to compare the proposed

algorithm with similar existing algorithms. Note that the performance of the proposed algorithm is assessed under different conditions. Interestingly, the theoretical findings are found to corroborate the simulation results very well. Moreover, a sensitivity analysis is performed on the proposed algorithm.

The paper is organized as follows. Section 2 describes the problem statement and briefly introduces the DLMS algorithm. Section 3 incorporates the VSSLMS algorithm into the DLMS algorithm, and then complete convergence and stability analyses are carried out. Simulation results are given in Section 4 followed by a thorough discussion of the results. Finally, Section 5 concludes the work.

Notation. Boldface letters are used for vectors/matrices and normal font for scalar quantities. Matrices are defined by capital letters and small letters are used for vectors. The notation $(.)^{\mathrm{T}}$ stands for transposition operation for vectors and matrices and expectation operation is denoted by $E[.]$. Any other mathematical operators that have been used will be defined as they are introduced in the paper.

2 Problem statement

Consider a network of N sensor nodes deployed over a geographical area for estimating an unknown parameter vector $\mathbf{w}^{\mathrm{o}}$ of size $(M \times 1)$, as shown in Figure 1. Each node k has access to a time realization of a known regressor vector $\mathbf{u}_k(i)$ of size $(1 \times M)$ and a scalar measurement $d_k(i)$ that are related by

$$d_k(i) = \mathbf{u}_k(i)\mathbf{w}^{\mathrm{o}} + v_k(i), 1 \leq k \leq N \tag{1}$$

where $v_k(i)$ is a spatially uncorrelated zero-mean additive white Gaussian noise with variance $\sigma_{v_k}^2$ and i denotes the time index. The measurements, $d_k(i)$ and $\mathbf{u}_k(i)$, are used to generate an estimate, $\mathbf{w}_k(i)$ of size $(M \times 1)$, of the unknown vector $\mathbf{w}^{\mathrm{o}}$. Assuming that each node cooperates only with its neighbors, then each node k has access to updates $\mathbf{w}_l(i)$, from its $\mathcal{N}_k$ neighbor nodes at every time instant i, where $l \underset{l \neq k}{\in} \mathcal{N}_k$, in addition to its own estimate, $\mathbf{w}_k(i)$. Two different schemes have been introduced in the literature for the diffusion algorithm. The adapt-then-combine (ATC) scheme [7] first updates the local estimate using the adaptive algorithm used and then the intermediate estimates from the neighbors are fused together. The second scheme, called combine-then-adapt (CTA) [6], reverses the order. It is found that the ATC scheme outperforms the CTA scheme [7] and therefore, this work uses the ATC scheme.

The objective of the adaptive algorithm is to minimize the following global cost function given by

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

The steepest descent algorithm is given as

$$\mathbf{w}_k(i+1) = \mathbf{w}_k(i) + \mu \sum_{k=1}^{N} \left(\mathbf{r}_{d\mathbf{u},k} - \mathbf{R}_{\mathbf{u},k}\mathbf{w}_k(i)\right), \tag{3}$$

where $\mathbf{r}_{d\mathbf{u},k} = \mathrm{E}\left[d_k\mathbf{u}_k^{\mathrm{T}}\right]$ is the cross-correlation between d_k and $\mathbf{u}_k$, $\mathbf{R}_{\mathbf{u},k} = \mathrm{E}\left[\mathbf{u}_k^{\mathrm{T}}\mathbf{u}_k\right]$ is the auto-correlation of $\mathbf{u}_k$, and μ is the step-size. The recursion defined in (3) requires full knowledge of the statistics of the entire network. A more practical solution utilizes the distributive nature of the network. The work in [6] gives a fully distributed solution, which has been modified and improved in [7]. The update equation is divided into two steps. The first step performs the adaptation, while the second step combines the intermediate updates from neighboring nodes. The resulting scheme is called *adapt-then-combine* or ATC. Using the ATC scheme, the diffusion LMS algorithm is given as

$$\begin{aligned} \mathbf{\Psi}_k(i+1) &= \mathbf{w}_k(i) + \mu_k\mathbf{u}_k(i)\left[d_k(i) - \mathbf{u}_k(i)\mathbf{w}_k(i)\right] \\ \mathbf{w}_k(i+1) &= \sum_{l \in \mathcal{N}_k} c_{lk}\mathbf{\Psi}_l(i+1), \end{aligned} \tag{4}$$

where $\mathbf{\Psi}_k(i+1)$ is the intermediate update, μ_k is the step-size for node k, and c_{lk} is the weight connecting node k to its neighboring node $l \in \mathcal{N}_k$, where $\mathcal{N}_k$ includes node k, and $\sum c_{lk} = 1$. Further details on the formation of the weights c_{lk} can be found in [6,7].

3 Variable step-size diffusion LMS algorithm

The VSSLMS algorithms show marked improvement over the LMS algorithm at a low computational complexity [20-25]. Therefore, this variation is inserted in the distributed algorithm to inherit the improved performance of the VSSLMS algorithm. Different variations have their own advantages and disadvantages. A complex step-size adaptation algorithm would not be suitable because of the physical limitations of the sensor node. As shown in [23], the algorithm proposed by [20] shows the best performance as well as having low complexity. Therefore, it is well suited for this application. A further comparison of performance of these variants in the present scenario confirm our choice of the VSSLMS algorithm.

The proposed algorithm simply incorporates the VSSLMS algorithm into the diffusion scheme given by (4). Using a VSSLMS algorithm, the step-size will also become a variable in this system of equations defining

the proposed distributed algorithm. Then the VSSDLMS algorithm is governed by the following:

$$\begin{aligned}
\boldsymbol{\Psi}_k(i+1) &= \mathbf{w}_k(i) + \mu_k(i)\mathbf{u}_k(i)\left(d_k(i) - \mathbf{u}_k(i)\mathbf{w}_k(i)\right),\\
\mu_k(i+1) &= f\left[\mu_k(i)\right],\\
\mathbf{w}_k(i+1) &= \sum_{l\in\mathcal{N}_k} c_{lk}\boldsymbol{\Psi}_l(i+1),
\end{aligned} \tag{5}$$

where $f\left[\mu_k(i)\right]$ is the step-size adaptation function that is defined using the VSSLMS adaptation given in [20] where the update equation is given by

$$\begin{aligned}
\mu_k(i+1) &= \alpha\mu_k(i) + \gamma\left(d_k(i) - \mathbf{u}_k(i)\mathbf{w}_k(i)\right)^2\\
&= \alpha\mu_k(i) + \gamma e_k^2(i),
\end{aligned} \tag{6}$$

where $e_k(i) = d_k(i) - \mathbf{u}_k(i)\mathbf{w}_k(i)$, $0 < \alpha < 1$ and $\gamma > 0$.

Since nodes exchange data amongst themselves, their current update will then be affected by the weighted average of the previous estimates. Therefore, to account for this inter-node dependence, it is suitable to study the performance of the whole network. Hence, some new variables need to be introduced and the local ones are transformed into global variables as follows:

$$\begin{aligned}
\mathbf{w}(i) &= \mathrm{col}\left\{\mathbf{w}_1(i),\ldots,\mathbf{w}_N(i)\right\},\\
\boldsymbol{\Psi}(i) &= \mathrm{col}\left\{\boldsymbol{\Psi}_1(i),\ldots,\boldsymbol{\Psi}_N(i)\right\},\\
\mathbf{U}(i) &= \mathrm{diag}\left\{\mathbf{u}_1(i),\ldots,\mathbf{u}_N(i)\right\},\\
\mathbf{D}(i) &= \mathrm{diag}\left\{\mu_1(i)\mathbf{I}_M,\ldots,\mu_N(i)\mathbf{I}_M\right\},\\
\mathbf{d}(i) &= \mathrm{col}\left\{d_1(i),\ldots,d_N(i)\right\},\\
\mathbf{v}(i) &= \mathrm{col}\left\{v_1(i),\ldots,v_N(i)\right\}.
\end{aligned}$$

From these new variables, a completely new set of equations representing the entire network is formed, starting with the relation between the measurements

$$\mathbf{d}(i) = \mathbf{U}(i)\mathbf{w}^{(o)} + \mathbf{v}(i), \tag{7}$$

where $\mathbf{w}^{(o)} = \mathbf{Q}\mathbf{w}^o$, and $\mathbf{Q} = col\left\{\mathbf{I}_M, \mathbf{I}_M, \ldots, \mathbf{I}_M\right\}$ is a $MN \times M$ matrix. Similarly, the update equations can be remodeled to represent the entire network

$$\begin{aligned}
\boldsymbol{\Psi}(i+1) &= \mathbf{w}(i) + \mathbf{D}(i)\mathbf{U}^{\mathrm{T}}(i)\left(\mathbf{d}(i) - \mathbf{U}(i)\mathbf{w}(i)\right),\\
\mathbf{D}(i+1) &= \alpha\mathbf{D}(i) + \gamma\mathbf{E}(i),\\
\mathbf{w}(i+1) &= \mathbf{G}\boldsymbol{\Psi}(i+1),
\end{aligned} \tag{8}$$

where $\mathbf{G} = \mathbf{C}\otimes\mathbf{I}_M$; $\mathbf{C}$ is an $N\times N$ weighting matrix, where $\{\mathbf{C}\}_{lk} = c_{lk}$; $\otimes$ is the Kronecker product; $\mathbf{D}(i)$ is the diagonal step-size matrix; and the error energy matrix, $\mathbf{E}(i)$, is given by

$$\mathbf{E}(i) = \mathrm{diag}\left\{e_1^2(i)\,\mathbf{I}_M, e_2^2(i)\,\mathbf{I}_M, \ldots, e_N^2(i)\,\mathbf{I}_M\right\}. \tag{9}$$

Considering the above set of equations, the mean and mean-square analyses and the steady-state behavior of the VSSDLMS algorithm are carried out as shown next. The mean analysis considers the stability of the algorithm and derives a bound for the step-size which would guarantee convergence. The mean-square analysis also derives transient and steady-state expressions for the mean square deviation (MSD) and the excess mean square error (EMSE). The MSD is defined as the error in the estimate of the unknown vector. The weight-error vector for node k is given by

$$\tilde{\mathbf{w}}_k(i) = \mathbf{w}^o - \mathbf{w}_k(i), \tag{10}$$

then the MSD can be simply defined as

$$\mathrm{MSD} = \mathrm{E}\left[\left\|\tilde{\mathbf{w}}_k(i)\right\|^2\right] = \mathrm{E}\left[\left\|\mathbf{w}^o - \mathbf{w}_k(i)\right\|^2\right]. \tag{11}$$

Similarly, the EMSE is derived from the error equation as follows:

$$\mathrm{EMSE} = \mathrm{E}\left[|e_k(i)|^2\right] - \sigma_{vk}^2 = \mathrm{E}\left[|d_k(i) - \mathbf{u}_k(i)\mathbf{w}_k(i)|^2\right] - \sigma_{vk}^2,$$

which can be solved further to get the following expression for the EMSE:

$$\mathrm{EMSE} = \mathrm{E}\left[\left\|\tilde{\mathbf{w}}_k(i)\right\|_{\mathbf{R}_k}^2\right], \tag{12}$$

where $\mathbf{R}_k$ is the autocorrelation matrix for node k.

3.1 Mean analysis

To begin with, let us introduce the global weight-error vector defined in [6,26] as

$$\tilde{\mathbf{w}}(i) = \mathbf{w}^{(o)} - \mathbf{w}(i). \tag{13}$$

Since $\mathbf{G}\mathbf{w}^{(o)} \triangleq \mathbf{w}^{(o)}$, by incorporating the global weight-error vector into (8), we get

$$\begin{aligned}
\tilde{\mathbf{w}}(i+1) &= \mathbf{G}\tilde{\boldsymbol{\Psi}}(i+1)\\
&= \mathbf{G}\tilde{\mathbf{w}}(i) - \mathbf{G}\mathbf{D}(i)\mathbf{U}^{\mathrm{T}}(i)\left(\mathbf{U}(i)\tilde{\mathbf{w}}(i) + \mathbf{v}(i)\right)\\
&= \mathbf{G}\left(\mathbf{I}_{MN} - \mathbf{D}(i)\mathbf{U}^{\mathrm{T}}(i)\mathbf{U}(i)\right)\tilde{\mathbf{w}}(i) - \mathbf{G}\mathbf{D}(i)\mathbf{U}^{\mathrm{T}}(i)\mathbf{v}(i).
\end{aligned} \tag{14}$$

Here we use the assumption that the step-size matrix $\mathbf{D}(i)$ is independent of the regressor matrix $\mathbf{U}(i)$ [20]. Accordingly, for small values of γ in (6), the following relation holds true asymptotically

$$\mathrm{E}\left[\mathbf{D}(i)\mathbf{U}^{\mathrm{T}}(i)\mathbf{U}(i)\right] \approx \mathrm{E}\left[\mathbf{D}(i)\right]\mathrm{E}\left[\mathbf{U}^{\mathrm{T}}(i)\mathbf{U}(i)\right], \tag{15}$$

where $\mathrm{E}\left[\mathbf{U}^{\mathrm{T}}(i)\mathbf{U}(i)\right] = \mathbf{R}_{\mathbf{U}}$ is the auto-correklation matrix of $\mathbf{U}(i)$, and taking the expectation on both sides of (14) gives

$$\mathrm{E}\left[\tilde{\mathbf{w}}(i+1)\right] = \mathbf{G}\left(\mathbf{I}_{MN} - \mathrm{E}\left[\mathbf{D}(i)\right]\mathbf{R}_{\mathbf{U}}\right)\mathrm{E}\left[\tilde{\mathbf{w}}(i)\right], \quad (16)$$

where the expectation of the second term of the right-hand side of (14) is 0 since the measurement noise is spatially uncorrelated with the regressor and zero-mean, as explained earlier.

From (16), we see that for stability in the mean we must have $|\lambda_{\max}(\mathbf{GB})| < 1$, where $\mathbf{B} = (\mathbf{I}_{MN} - \mathrm{E}\left[\mathbf{D}(i)\right]\mathbf{R}_{\mathbf{U}})$. Since $\mathbf{G}$ comes from $\mathbf{C}$ and we know that $\|\mathbf{GB}\|_2 \leq \|\mathbf{G}\|_2 . \|\mathbf{B}\|_2$, we can safely infer that

$$|\lambda_{\max}(\mathbf{GB})| \leq \|\mathbf{C}\|_2 . |\lambda_{\max}(\mathbf{B})| . \quad (17)$$

Since there is already a condition that $\|\mathbf{C}\|_2 = 1$ and for noncooperative schemes, we have ($\mathbf{G} = \mathbf{I}_{MN}$), we can safely conclude that

$$|\lambda_{\max}(\mathbf{GB})| \leq |\lambda_{\max}(\mathbf{B})| . \quad (18)$$

So we can see that the cooperation mode only enhances the stability of the system (for further details, refer to [6,7]). Since stability is also dependent on the step-size, then the algorithm will be stable in the mean if

$$\prod_{i=0}^{n}\left(\mathbf{I} - \mathrm{E}\left[\mu_k(i)\right]\mathbf{R}_{\mathbf{u},k}\right) \to 0, as\, n \to \infty \quad (19)$$

which holds true if the mean of the step-size is governed by

$$0 < \mathrm{E}\left[\mu_k(i)\right] < \frac{2}{\lambda_{\max}\left(\mathbf{R}_{\mathbf{u},k}\right)}, 1 \leq k \leq N, \quad (20)$$

where $\lambda_{\max}\left(\mathbf{R}_{\mathbf{u},k}\right)$ is the maximum eigenvalue of the auto-correlation matrix $\mathbf{R}_{\mathbf{u},k}$. This scenario is different from that of the fixed step-size as in this case where the system is stable only when the mean of the step-size is within the limits defined by (20).

3.2 Mean-square analysis

In this section, the mean-square analysis of the VSS-DLMS algorithm is investigated. Here, the weighted norm has been used instead of the regular norm. The motivation behind using a weighted norm stems from the fact that even though the MSD does not require a weighted norm, the evaluation of the EMSE depends on a weighted norm. In order to accommodate both these measures, a general analysis is conducted using a weighted norm, where a weighting matrix is replaced by an identity matrix for the case of MSD, where a weighting matrix is not required [26].

We take the weighted norm of (14) and then apply the expectation operator to both of its sides. This yields the following:

$$\begin{aligned}\mathrm{E}\left[\|\tilde{\mathbf{w}}(i+1)\|^2_{\mathbf{\Sigma}}\right] &= \mathrm{E}\left[\left\|\mathbf{G}\left(\mathbf{I}_{MN} - \mathbf{D}(i)\mathbf{U}^{\mathrm{T}}(i)\mathbf{U}(i)\right)\tilde{\mathbf{w}}(i)\right.\right.\\ &\quad \left.\left. - \mathbf{GD}(i)\mathbf{U}^{\mathrm{T}}(i)\mathbf{v}(i)\right\|^2_{\mathbf{\Sigma}}\right]\\ &= \mathrm{E}\left[\|\tilde{\mathbf{w}}(i)\|^2_{\mathbf{G}^{\mathrm{T}}\mathbf{\Sigma}\mathbf{G}}\right] - \mathrm{E}\left[\|\tilde{\mathbf{w}}(i)\|^2_{\mathbf{G}^{\mathrm{T}}\mathbf{\Sigma}\mathbf{Y}(i)\mathbf{U}(i)}\right]\\ &\quad - \mathrm{E}\left[\|\tilde{\mathbf{w}}(i)\|^2_{\mathbf{U}^{\mathrm{T}}(i)\mathbf{Y}^{\mathrm{T}}(i)\mathbf{\Sigma}\mathbf{G}}\right]\\ &\quad + \mathrm{E}\left[\|\tilde{\mathbf{w}}(i)\|^2_{\mathbf{U}^{\mathrm{T}}(i)\mathbf{Y}^{\mathrm{T}}(i)\mathbf{\Sigma}\mathbf{Y}(i)\mathbf{U}(i)}\right]\\ &\quad + \mathrm{E}\left[\mathbf{v}^{\mathrm{T}}(i)\mathbf{Y}^{\mathrm{T}}(i)\mathbf{\Sigma}\mathbf{Y}(i)\mathbf{v}(i)\right]\\ &= \mathrm{E}\left[\|\tilde{\mathbf{w}}(i)\|^2_{\hat{\Sigma}}\right] + \mathrm{E}\left[\mathbf{v}^{\mathrm{T}}(i)\mathbf{Y}^{\mathrm{T}}(i)\mathbf{\Sigma}\mathbf{Y}(i)\mathbf{v}(i)\right], \end{aligned} \quad (21)$$

where

$$\mathbf{Y}(i) = \mathbf{GD}(i)\mathbf{U}^{\mathrm{T}}(i) \quad (22)$$

$$\begin{aligned}\hat{\Sigma} = {} & \mathbf{G}^{\mathrm{T}}\mathbf{\Sigma}\mathbf{G} - \mathbf{G}^{\mathrm{T}}\mathbf{\Sigma}\mathbf{Y}(i)\mathbf{U}(i) - \mathbf{U}^{\mathrm{T}}(i)\mathbf{Y}^{\mathrm{T}}(i)\mathbf{\Sigma}\mathbf{G}\\ & + \mathbf{U}^{\mathrm{T}}(i)\mathbf{Y}^{\mathrm{T}}(i)\mathbf{\Sigma}\mathbf{Y}(i)\mathbf{U}(i).\end{aligned} \quad (23)$$

Using the data independence assumption [26] and applying the expectation operator gives

$$\begin{aligned}\mathrm{E}\left[\hat{\Sigma}\right] &= \mathbf{G}^{\mathrm{T}}\mathbf{\Sigma}\mathbf{G} - \mathbf{G}^{\mathrm{T}}\mathbf{\Sigma}\mathrm{E}\left[\mathbf{Y}(i)\mathbf{U}(i)\right] - \mathrm{E}\left[\mathbf{U}^{\mathrm{T}}(i)\mathbf{Y}^{\mathrm{T}}(i)\right]\mathbf{\Sigma}\mathbf{G}\\ &\quad + \mathrm{E}\left[\mathbf{U}^{\mathrm{T}}(i)\mathbf{Y}^{\mathrm{T}}(i)\mathbf{\Sigma}\mathbf{Y}(i)\mathbf{U}(i)\right]\\ &= \mathbf{G}^{\mathrm{T}}\mathbf{\Sigma}\mathbf{G} - \mathbf{G}^{\mathrm{T}}\mathbf{\Sigma}\mathbf{G}\mathrm{E}\left[\mathbf{D}(i)\right]\mathrm{E}\left[\mathbf{U}^{\mathrm{T}}(i)\mathbf{U}(i)\right]\\ &\quad - \mathrm{E}\left[\mathbf{U}^{\mathrm{T}}(i)\mathbf{U}(i)\right]\mathrm{E}\left[\mathbf{D}(i)\right]\mathbf{G}^{\mathrm{T}}\mathbf{\Sigma}\mathbf{G}\\ &\quad + \mathrm{E}\left[\mathbf{U}^{\mathrm{T}}(i)\mathbf{Y}^{\mathrm{T}}(i)\mathbf{\Sigma}\mathbf{Y}(i)\mathbf{U}(i)\right]. \end{aligned} \quad (24)$$

For ease of notation, we denote $\mathrm{E}\left[\hat{\Sigma}\right] = \mathbf{\Sigma}'$ for the remaining analysis.

3.2.1 Mean-square analysis for Gaussian data

The evaluation of the expectations in (24) is quite tedious for non-Gaussian data. Therefore, it is assumed here that the data is Gaussian in order to evaluate (24). The auto-correlation matrix can be decomposed as $\mathbf{R}_{\mathbf{U}} = \mathbf{T}\mathbf{\Lambda}\mathbf{T}^{\mathrm{T}}$, where $\mathbf{\Lambda}$ is a diagonal matrix containing the eigenvalues for the entire network and $\mathbf{T}$ is a matrix containing the eigenvectors corresponding to these eigenvalues. Using this eigenvalue decomposition, we define the following relations

$$\bar{\mathbf{w}}(i) = \mathbf{T}^{\mathrm{T}}\tilde{\mathbf{w}}(i) \quad \bar{\mathbf{U}}(i) = \mathbf{U}(i)\mathbf{T} \quad \bar{\mathbf{G}} = \mathbf{T}^{\mathrm{T}}\mathbf{G}\mathbf{T}$$
$$\bar{\mathbf{\Sigma}} = \mathbf{T}^{\mathrm{T}}\mathbf{\Sigma}\mathbf{T} \quad \bar{\mathbf{\Sigma}}' = \mathbf{T}^{\mathrm{T}}\mathbf{\Sigma}'\mathbf{T} \quad \bar{\mathbf{D}}(i) = \mathbf{T}^{\mathrm{T}}\mathbf{D}(i)\mathbf{T} = \mathbf{D}(i),$$

where the input regressors are considered independent of each other at each node and the step-size matrix $\mathbf{D}(i)$ is block-diagonal, so it does not transform since $\mathbf{T}^{\mathrm{T}}\mathbf{T} = \mathbf{I}$. Using these relations, (21) and (24) can be rewritten, respectively, as

$$\mathrm{E}\left[\|\bar{\mathbf{w}}(i+1)\|^2_{\bar{\boldsymbol{\Sigma}}}\right] = \mathrm{E}\left[\|\bar{\mathbf{w}}(i)\|^2_{\bar{\boldsymbol{\Sigma}}'}\right] + \mathrm{E}\left[\mathbf{v}^{\mathrm{T}}(i)\bar{\mathbf{Y}}^{\mathrm{T}}(i)\bar{\boldsymbol{\Sigma}}\bar{\mathbf{Y}}(i)\mathbf{v}(i)\right], \tag{25}$$

and

$$\begin{aligned}\bar{\boldsymbol{\Sigma}}' = {} & \bar{\mathbf{G}}^{\mathrm{T}}\bar{\boldsymbol{\Sigma}}\bar{\mathbf{G}} - \bar{\mathbf{G}}^{\mathrm{T}}\bar{\boldsymbol{\Sigma}}\bar{\mathbf{G}}\mathrm{E}\left[\mathbf{D}(i)\right]\mathrm{E}\left[\bar{\mathbf{U}}^{\mathrm{T}}(i)\bar{\mathbf{U}}(i)\right] \\ & - \mathrm{E}\left[\bar{\mathbf{U}}^{\mathrm{T}}(i)\bar{\mathbf{U}}(i)\right]\mathrm{E}\left[\mathbf{D}(i)\right]\bar{\mathbf{G}}^{\mathrm{T}}\bar{\boldsymbol{\Sigma}}\bar{\mathbf{G}} \\ & + \mathrm{E}\left[\bar{\mathbf{U}}^{\mathrm{T}}(i)\bar{\mathbf{Y}}(i)\bar{\boldsymbol{\Sigma}}\bar{\mathbf{Y}}(i)\bar{\mathbf{U}}(i)\right],\end{aligned} \tag{26}$$

where $\bar{\mathbf{Y}}(i) = \bar{\mathbf{G}}\mathbf{D}(i)\bar{\mathbf{U}}^{\mathrm{T}}(i)$.

It can be seen that $\mathrm{E}\left[\bar{\mathbf{U}}^{\mathrm{T}}(i)\bar{\mathbf{U}}(i)\right] = \boldsymbol{\Lambda}$. Also, using the bvec operator [27], we have $\bar{\sigma} = \mathrm{bvec}\left\{\bar{\boldsymbol{\Sigma}}\right\}$, where the bvec operator divides the matrix into smaller blocks and then applies the vec operator to each of the smaller blocks. Now, let $\mathbf{R}_{\mathbf{v}} = \boldsymbol{\Lambda}_{\mathbf{v}} \odot \mathbf{I}_M$ denote the block diagonal noise covariance matrix for the entire network, where $\odot$ denotes the block Kronecker product [27] and $\boldsymbol{\Lambda}_{\mathbf{v}}$ is a diagonal noise variance matrix for the network. Hence, the second term of the right-hand side of (25) is

$$\mathrm{E}\left[\mathbf{v}^{\mathrm{T}}(i)\bar{\mathbf{Y}}^{\mathrm{T}}(i)\bar{\boldsymbol{\Sigma}}\bar{\mathbf{Y}}(i)\mathbf{v}(i)\right] = \mathbf{b}^{\mathrm{T}}(i)\bar{\boldsymbol{\sigma}}, \tag{27}$$

where $\mathbf{b}(i) = \mathrm{bvec}\left\{\mathbf{R}_{\mathbf{v}}\mathrm{E}\left[\mathbf{D}^2(i)\right]\boldsymbol{\Lambda}\right\}$.

The fourth-order moment $\mathrm{E}\left[\bar{\mathbf{U}}^{\mathrm{T}}(i)\bar{\mathbf{Y}}^{\mathrm{T}}(i)\bar{\boldsymbol{\Sigma}}\bar{\mathbf{Y}}(i)\bar{\mathbf{U}}(i)\right]$ in (26) remains to be evaluated. Using the step-size independence assumption and the $\odot$ operator, we get

$$\begin{aligned}\mathrm{bvec}\left\{\mathrm{E}\left[\bar{\mathbf{U}}^{\mathrm{T}}(i)\bar{\mathbf{Y}}^{\mathrm{T}}(i)\bar{\boldsymbol{\Sigma}}\bar{\mathbf{Y}}(i)\bar{\mathbf{U}}(i)\right]\right\} = {} & \left(\mathrm{E}\left[\mathbf{D}(i)\odot\mathbf{D}(i)\right]\right) \\ & \times \mathbf{A}\left(\mathbf{G}^{\mathrm{T}}\odot\mathbf{G}^{\mathrm{T}}\right)\bar{\boldsymbol{\sigma}},\end{aligned} \tag{28}$$

where we have from [6]

$$\mathbf{A} = \mathrm{diag}\left\{\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_N\right\}, \tag{29}$$

and each matrix $\mathbf{A}_k$ is given by

$$\mathbf{A}_k = \mathrm{diag}\left\{\boldsymbol{\Lambda}_1\otimes\boldsymbol{\Lambda}_k, \ldots, \lambda_k\lambda_k^{\mathrm{T}} + 2\boldsymbol{\Lambda}_k\otimes\boldsymbol{\Lambda}_k, \ldots, \boldsymbol{\Lambda}_N\otimes\boldsymbol{\Lambda}_k\right\}, \tag{30}$$

where $\boldsymbol{\Lambda}_k$ defines the diagonal eigenvalue matrix and λ_k is the eigenvalue vector for node k.

The output of the matrix $\mathrm{E}\left[\mathbf{D}(i)\odot\mathbf{D}(i)\right]$ can be written as

$$\begin{aligned}\left(\mathrm{E}\left[\mathbf{D}(i)\odot\mathbf{D}(i)\right]\right)_{kk} = {} & \mathrm{E}\left[\mathrm{diag}\left\{\mu_k(i)\mathbf{I}_M\otimes\mu_1(i)\mathbf{I}_M, \ldots,\right.\right. \\ & \mu_k(i)\mathbf{I}_M\otimes\mu_k(i)\mathbf{I}_M, \ldots, \\ & \left.\left.\mu_k(i)\mathbf{I}_M\otimes\mu_N(i)\mathbf{I}_M\right\}\right] \\ = {} & \mathrm{E}\left[\mathrm{diag}\left\{\mu_k(i)\mu_1(i)\mathbf{I}_{M^2}, \ldots,\right.\right. \\ & \left.\left.\mu_k^2(i)\mathbf{I}_{M^2}, \ldots, \mu_k(i)\mu_N(i)\mathbf{I}_{M^2}\right\}\right] \\ = {} & \mathrm{diag}\left\{\mathrm{E}\left[\mu_k(i)\right]\mathrm{E}\left[\mu_1(i)\right]\mathbf{I}_{M^2}, \ldots,\right. \\ & \mathrm{E}\left[\mu_k^2(i)\right]\mathbf{I}_{M^2}, \ldots, \\ & \left.\mathrm{E}\left[\mu_k(i)\right]\mathrm{E}\left[\mu_N(i)\right]\mathbf{I}_{M^2}\right\}.\end{aligned} \tag{31}$$

Now applying the bvec operator to the weighting matrix $\bar{\boldsymbol{\Sigma}}'$ using the relation $\mathrm{bvec}\left\{\bar{\boldsymbol{\Sigma}}'\right\} = \bar{\sigma}'$, where we can get back the original $\bar{\boldsymbol{\Sigma}}'$ through $\mathrm{bvec}\left\{\bar{\sigma}'\right\} = \bar{\boldsymbol{\Sigma}}'$, we get

$$\begin{aligned}\mathrm{bvec}\left\{\bar{\boldsymbol{\Sigma}}'\right\} = \bar{\sigma} = {} & \left[\mathbf{I}_{M^2N^2} - \left(\mathbf{I}_{MN}\odot\boldsymbol{\Lambda}\mathrm{E}\left[\mathbf{D}(i)\right]\right)\right. \\ & - \left(\boldsymbol{\Lambda}\mathrm{E}\left[\mathbf{D}(i)\right]\odot\mathbf{I}_{MN}\right) \\ & \left.+ \left(\mathrm{E}\left[\mathbf{D}(i)\odot\mathbf{D}(i)\right]\right)\mathbf{A}\right]\left(\mathbf{G}^{\mathrm{T}}\odot\mathbf{G}^{\mathrm{T}}\right)\bar{\boldsymbol{\sigma}} \\ = {} & \mathbf{F}(i)\bar{\boldsymbol{\sigma}},\end{aligned} \tag{32}$$

where

$$\begin{aligned}\mathbf{F}(i) = {} & \left[\mathbf{I}_{M^2N^2} - \left(\mathbf{I}_{MN}\odot\boldsymbol{\Lambda}\mathrm{E}\left[\mathbf{D}(i)\right]\right) - \left(\boldsymbol{\Lambda}\mathrm{E}\left[\mathbf{D}(i)\right]\odot\mathbf{I}_{MN}\right)\right. \\ & \left.+ \left(\mathrm{E}\left[\mathbf{D}(i)\odot\mathbf{D}(i)\right]\right)\mathbf{A}\right]\left(\mathbf{G}^{\mathrm{T}}\odot\mathbf{G}^{\mathrm{T}}\right).\end{aligned} \tag{33}$$

Then (21) will take on the following form:

$$\mathrm{E}\left[\|\bar{\mathbf{w}}(i+1)\|^2_{\bar{\sigma}}\right] = \mathrm{E}\left[\|\bar{\mathbf{w}}(i)\|^2_{\mathbf{F}(i)\bar{\sigma}}\right] + \mathbf{b}^{\mathrm{T}}(i)\bar{\sigma}, \tag{34}$$

which characterizes the transient behavior of the network. Although (34) does not explicitly show the effect of the variable step (VSS) algorithm on the network's performance, this effect is in fact subsumed in the weighting matrix, $\mathbf{F}(i)$ which varies for each iteration, unlike in the fixed step-size LMS algorithm where the analysis shows that this weighting matrix remains fixed at all iterations. Also, (33) clearly shows the effect of the VSS algorithm on the performance of the algorithm through the presence of the diagonal step-size matrix $\mathbf{D}(i)$.

3.2.2 Learning behavior of the proposed algorithm

In this section, the learning behavior (which shows how the algorithm evolves with time) of the VSSDLMS algorithm is evaluated. Starting with $\bar{\mathbf{w}}_0 = \mathbf{w}^{(o)}$ and $\mathbf{D}_0 = \mu_0\mathbf{I}_{MN}$, we have for iteration $(i+1)$

$$\begin{aligned}\mathcal{E}(i-1) = \mathrm{diag}\left\{\left(\mathrm{E}\left[\|\bar{\mathbf{w}}(i-1)\|^2_{\boldsymbol{\lambda}}\right] + \sigma^2_{v,1}\right)\mathbf{I}_M, \ldots,\right. \\ \left.\left(\mathrm{E}\left[\|\bar{\mathbf{w}}(i-1)\|^2_{\boldsymbol{\lambda}}\right] + \sigma^2_{v,N}\right)\mathbf{I}_M\right\}\end{aligned} \tag{35}$$

$$\mathrm{E}\left[\mathbf{D}(i)\right] = \alpha \mathrm{E}\left[\mathbf{D}\left(i-1\right)\right] + \gamma \mathcal{E}\left(i-1\right) \tag{36}$$

$$\begin{aligned}\mathrm{E}\left[\mathbf{D}^2(i)\right] &= \alpha^2 \mathrm{E}\left[\mathbf{D}^2\left(i-1\right)\right] + 2\alpha\gamma \mathcal{E}\left(i-1\right) \\ &\quad + \gamma^2 \mathcal{E}^2\left(i-1\right)\end{aligned} \tag{37}$$

$$\begin{aligned}\mathbf{F}(i) = \big[&\mathbf{I}_{M^2N^2} - \left(\mathbf{I}_{MN} \odot \mathbf{\Lambda}\mathrm{E}\left[\mathbf{D}\left(i\right)\right]\right) \\ &- \left(\mathbf{\Lambda}\mathrm{E}\left[\mathbf{D}\left(i\right)\right] \odot \mathbf{I}_{MN}\right) \\ &+ \left(\mathrm{E}\left[\mathbf{D}\left(i\right) \odot \mathbf{D}\left(i\right)\right]\right)\mathbf{A}\big]\left(\mathbf{G}^{\mathrm{T}} \odot \mathbf{G}^{\mathrm{T}}\right)\end{aligned} \tag{38}$$

$$\mathbf{b}(i) = \mathrm{bvec}\left\{\mathbf{R}_{\mathbf{v}}\mathrm{E}\left[\mathbf{D}^2(i)\right]\mathbf{\Lambda}\right\}; \tag{39}$$

then incorporating the above relations in (34) gives

$$\begin{aligned}\mathrm{E}\left[\|\bar{\mathbf{w}}(i+1)\|^2_{\bar{\sigma}}\right] &= \mathrm{E}\left[\|\bar{\mathbf{w}}(i)\|^2_{\mathbf{F}(i)\bar{\sigma}}\right] + \mathbf{b}^{\mathrm{T}}(i)\bar{\sigma} \\ &= \left\|\bar{\mathbf{w}}^{(o)}\right\|^2_{\left(\prod_{m=0}^{i}\mathbf{F}(m)\right)\bar{\sigma}} \\ &\quad + \left[\sum_{m=0}^{i-1}\mathbf{b}^{\mathrm{T}}(m)\left(\prod_{n=m+1}^{i}\mathbf{F}(n)\right)\right. \\ &\quad \left. + \mathbf{b}^{\mathrm{T}}(i)\mathbf{I}_{MN}\right]\bar{\sigma}.\end{aligned} \tag{40}$$

Now, after subtracting the results of iteration i from those of iteration $(i+1)$ and simplifying them, we get

$$\begin{aligned}\mathrm{E}\left[\|\bar{\mathbf{w}}(i+1)\|^2_{\bar{\sigma}}\right] &= \mathrm{E}\left[\|\bar{\mathbf{w}}(i)\|^2_{\bar{\sigma}}\right] + \left\|\bar{\mathbf{w}}^{(o)}\right\|^2_{\mathbf{F}'(i)(\mathbf{F}(i)-\mathbf{I}_{MN})\bar{\sigma}} \\ &\quad + \left[\mathbf{F}''(i)\left(\mathbf{F}(i)-\mathbf{I}_{MN}\right) + \mathbf{b}^{\mathrm{T}}(i)\mathbf{I}_{MN}\right]\bar{\sigma},\end{aligned} \tag{41}$$

where

$$\mathbf{F}'(i) = \prod_{m=0}^{i-1}\mathbf{F}(m), \tag{42}$$

$$\mathbf{F}''(i) = \sum_{m=0}^{i-2}\mathbf{b}^{\mathrm{T}}(m)\left(\prod_{n=m+1}^{i-1}\mathbf{F}(n)\right) + \mathbf{b}^{\mathrm{T}}(i)\mathbf{I}_{MN}, \tag{43}$$

which can be defined iteratively as

$$\mathbf{F}'(i+1) = \mathbf{F}'(i)\mathbf{F}(i), \tag{44}$$

$$\mathbf{F}''(i+1) = \mathbf{F}''(i)\mathbf{F}(i) + \mathbf{b}^{\mathrm{T}}(i)\mathbf{I}_{MN}. \tag{45}$$

In order to evaluate the MSD and EMSE, we need to define the corresponding weighting matrix for each of them. Taking $\bar{\sigma} = (1/N)\,\mathrm{bvec}\left\{\mathbf{I}_{MN}\right\} = \mathbf{q}_\eta$ and $\eta(i) = (1/N)\,\mathrm{E}\left[\|\bar{\mathbf{w}}(i)\|^2\right]$ for the MSD, we get

$$\begin{aligned}\eta\left(i\right) &= \eta\left(i-1\right) + \left\|\bar{\mathbf{w}}^{(o)}\right\|^2_{\mathbf{F}'(i)(\mathbf{F}(i)-\mathbf{I}_{MN})\mathbf{q}_\eta} \\ &\quad + \left[\mathbf{F}''(i)\left(\mathbf{F}(i)-\mathbf{I}_{MN}\right) + \mathbf{b}^{\mathrm{T}}(i)\mathbf{I}_{MN}\right]\mathbf{q}_\eta.\end{aligned} \tag{46}$$

Similarly, taking $\bar{\sigma} = (1/N)\,\mathrm{bvec}\left\{\mathbf{\Lambda}\right\} = \lambda_\zeta$ and $\zeta(i) = (1/N)\,\mathrm{E}\left[\|\bar{\mathbf{w}}(i)\|^2_{\mathbf{\Lambda}}\right]$, the EMSE behavior is governed by

$$\begin{aligned}\zeta\left(i\right) &= \zeta\left(i-1\right) + \left\|\bar{\mathbf{w}}^{(o)}\right\|^2_{\mathbf{F}'(i)(\mathbf{F}(i)-\mathbf{I}_{MN})\lambda_\zeta} \\ &\quad + \left[\mathbf{F}''(i)\left(\mathbf{F}(i)-\mathbf{I}_{MN}\right) + \mathbf{b}^{\mathrm{T}}(i)\mathbf{I}_{MN}\right]\lambda_\zeta.\end{aligned} \tag{47}$$

The relations in (46) and (47) govern the transient behavior of the MSD and EMSE of the proposed algorithm. These relations show how the effect on the proposed algorithm's transient behavior of the weighting matrix varies from one iteration to the next as the weighting matrix itself varies at each iteration. This is not the case in the simple fixed step-size DLMS in [6] where the weighting matrix remains constant for all iterations. Since the weighting matrix depends on the step-size matrix, which becomes very small asymptotically, then both the norm and influence of the weighting matrix also become asymptotically small. From the above relations, it is seen that both the MSD and EMSE become very small at steady-state because the weighting matrix itself becomes small at steady-state and these relations will then depend only on the product of the weighting matrices at each iteration.

3.3 Steady-state analysis

From the second relation in (8), it is seen that the step-size for each node is independent of the data received from other nodes. Even though the connectivity matrix, $\mathbf{G}$, does not permit the weighting matrix, $\mathbf{F}(\mathbf{i})$, to be evaluated separately for each node, this is not the case for the determination of the step-size at any node. Here, we define the misadjustment as the ratio of the EMSE to the minimum mean square error. The misadjustment value is used in determining the steady-state performance of the algorithm [11]. Therefore, taking the approach of [20], we first find the misadjustment, as given by

$$\mathcal{M}_k = \frac{1 - \left[1 - 2\frac{(3-\alpha)\gamma\sigma^2_{v,k}}{1-\alpha^2}\mathrm{tr}\left(\mathbf{\Lambda}_k\right)\right]^{1/2}}{1 + \left[1 - 2\frac{(3-\alpha)\gamma\sigma^2_{v,k}}{1-\alpha^2}\mathrm{tr}\left(\mathbf{\Lambda}_k\right)\right]^{1/2}}. \tag{48}$$

Then solving (36) and (37) along with (48) leads to the steady-state values for the step-size and its square for each node

$$\mu_{ss,k} = \frac{\gamma\sigma^2_{v,k}\left(1+\mathcal{M}_k\right)}{1-\alpha}, \tag{49}$$

$$\mu^2_{ss,k} = \frac{2\alpha\gamma\mu_{ss,k}\sigma^2_{v,k}\left(1+\mathcal{M}_k\right) + \gamma^2\sigma^4_{v,k}\left(1+\mathcal{M}_k\right)^2}{1-\alpha^2}. \tag{50}$$

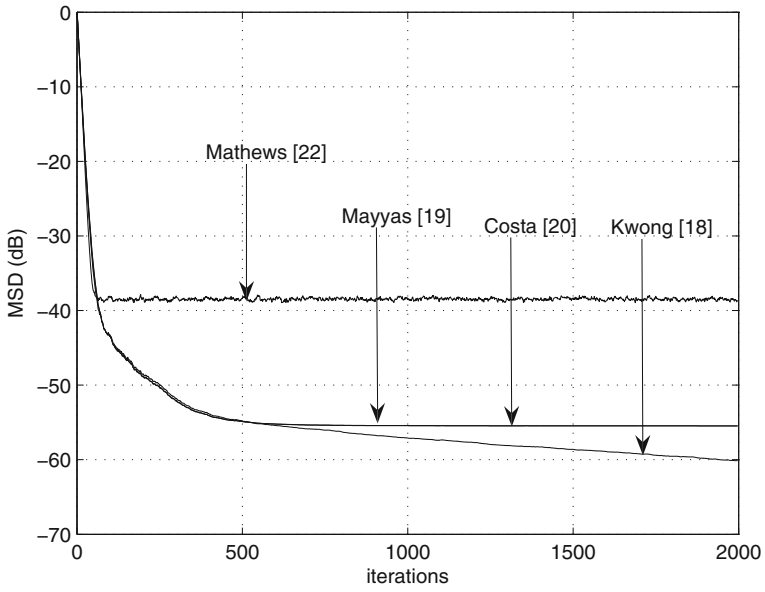

Figure 2 MSD for various VSSLMS algorithms applied to the diffusion scheme.

Incorporating these two steady-state relations in (33) yields the steady-state weighting matrix as

$$\mathbf{F}_{ss} = \left[\mathbf{I}_{M^2N^2} - (\mathbf{I}_{MN} \odot \Lambda \mathrm{E}\left[\mathbf{D}_{ss}\right]) - (\Lambda \mathrm{E}\left[\mathbf{D}_{ss}\right] \odot \mathbf{I}_{MN}) + (\mathrm{E}\left[\mathbf{D}_{ss} \odot \mathbf{D}_{ss}\right]) A\right] \left(\mathbf{G}^{\mathrm{T}} \odot \mathbf{G}^{\mathrm{T}}\right), \quad (51)$$

where $\mathbf{D}_{ss} = \mathrm{diag}\left\{\mu_{ss,k}\mathbf{I}_M\right\}$.

Thus, the steady-state mean-square behavior is given by

$$\mathrm{E}\left[\|\bar{\mathbf{w}}_{ss}\|^2_{\overline{\sigma}}\right] = \mathrm{E}\left[\|\bar{\mathbf{w}}_{ss}\|^2_{\mathbf{F}_{ss}\overline{\sigma}}\right] + \mathbf{b}^{\mathrm{T}}_{ss}\overline{\sigma}, \quad (52)$$

where $\mathbf{b}_{ss} = \mathbf{R}_{\mathbf{v}}\mathbf{D}^2_{ss}\mathbf{\Lambda}$ and $\mathbf{D}^2_{ss} = \mathrm{diag}\left\{\mu^2_{ss,k}\mathbf{I}_M\right\}$. Now solving (52), we get

$$\mathrm{E}\left[\|\bar{\mathbf{w}}_{ss}\|^2_{\overline{\sigma}}\right] = \mathbf{b}^{\mathrm{T}}_{ss}\left[\mathbf{I}_{M^2N^2} - \mathbf{F}_{ss}\right]^{-1}\overline{\sigma}. \quad (53)$$

This equation gives the steady-state performance measure for the entire network. In order to solve for the steady-state values of MSD and EMSE, we take $\bar{\sigma} = \mathbf{q}_\eta$

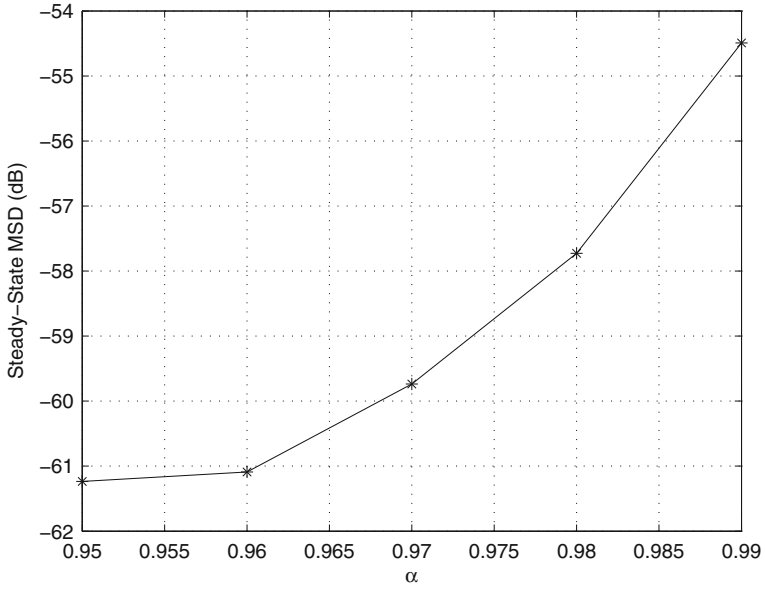

Figure 3 Steady-state MSD values for varying values of α.

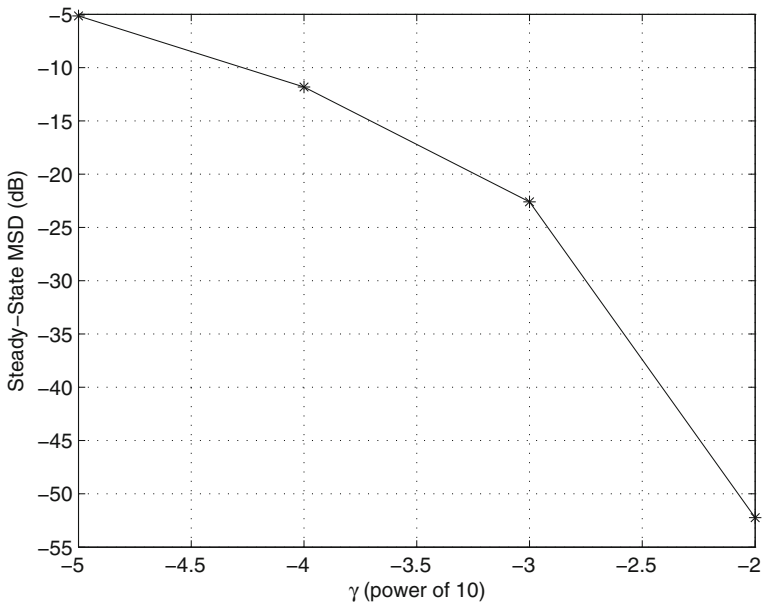

Figure 4 Steady-state MSD values for varying values of γ.

and $\bar{\sigma} = \lambda_\zeta$, respectively, as in (46) and (47). This gives us the steady-state values for MSD and EMSE as follows:

$$\eta_{ss} = \mathbf{b}_{ss}^{\mathrm{T}} \left[\mathbf{I}_{M^2N^2} - \mathbf{F}_{ss}\right]^{-1} \mathbf{q}_\eta, \tag{54}$$

$$\zeta_{ss} = \mathbf{b}_{ss}^{\mathrm{T}} \left[\mathbf{I}_{M^2N^2} - \mathbf{F}_{ss}\right]^{-1} \lambda_\zeta. \tag{55}$$

The above two steady-state relationships depend on the steady-state weighting matrix which becomes very small at steady-state, as explained before. As a result, the steady-state results for the proposed algorithms become very small compared to those for the fixed step-size DLMS algorithm.

4 Numerical results

In this section, several simulation scenarios are considered and discussed to assess the performance of the proposed VSSDLMS algorithm. Results have been conducted

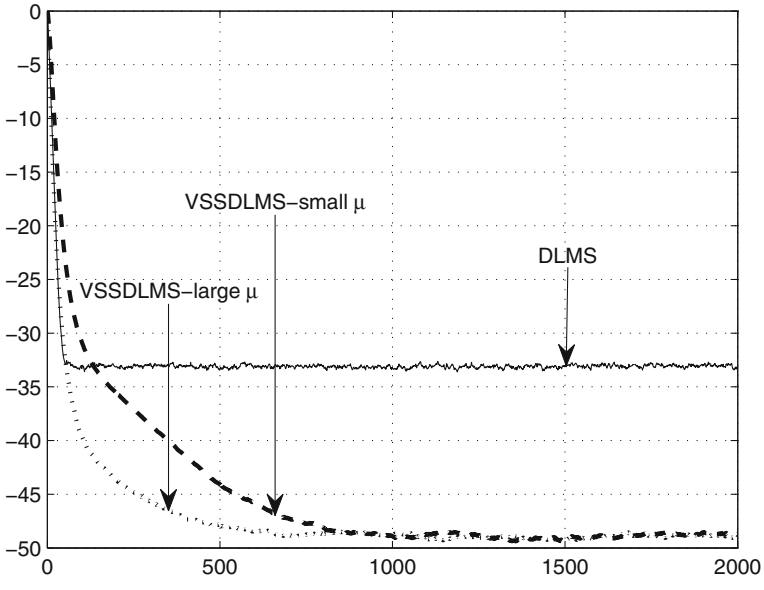

Figure 5 Comparison with the DLMS algorithm having a high step-size.

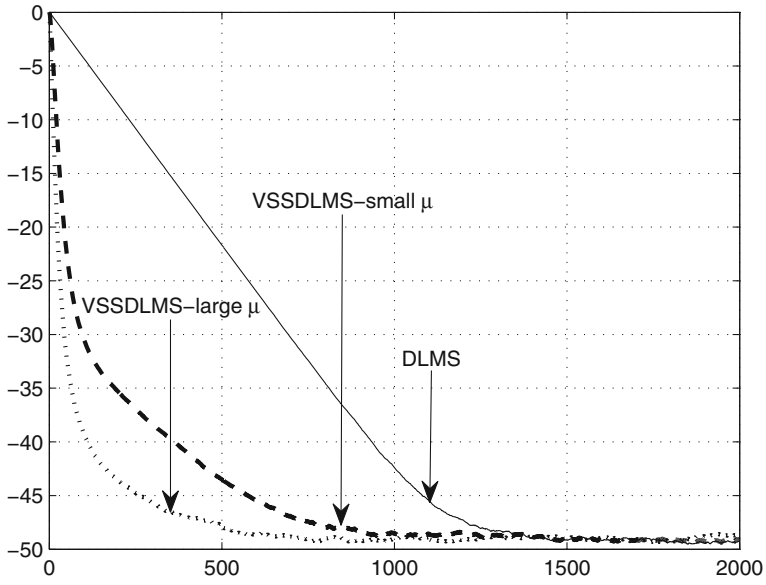

Figure 6 Comparison with the DLMS algorithm having a low step-size.

for different average SNR values. The performance measure used throughout these simulations is the MSD. The length of the unknown vector is taken as $M = 4$. The size of the network is $N = 20$ nodes. The sensors are randomly placed in an area of 1 unit square. The input regressor vector is assumed to be white Gaussian with auto-correlation matrix having the same variance for all nodes. Results averaged over 100 experiments are shown for the SNR value of 20 dB, a normalized communication range of 0.3, and a Gaussian environment.

First, the discussed that VSSLMS algorithms are compared with each other for the case of SNR 20 dB and the results of this comparison are reported in Figure 2. As can be depicted from Figure 2, the algorithm of [20] performs the best and is therefore chosen for our proposed VSSDLMS algorithm.

The sensitivity analysis for the selected VSSDLMS algorithm operating in the scenario explained above is discussed next. Since the VSSDLMS algorithm depends upon the choice of α and γ, these values are varied to check

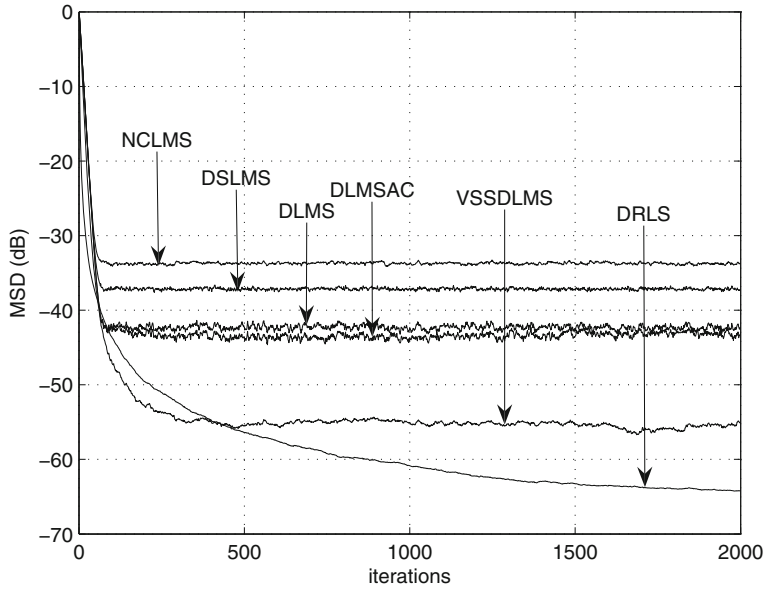

Figure 7 MSD for 20 nodes and SNR = 20 dB.

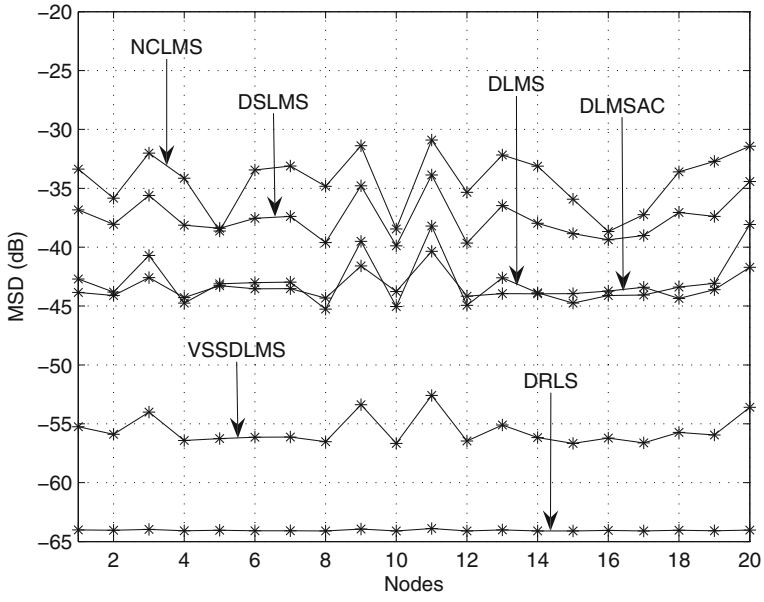

Figure 8 MSD at steady-state for 20 nodes and SNR = 20 dB.

their effect on the performance of the algorithm. As can be seen from Figure 3, the performance of the VSSDLMS algorithm degrades as α gets larger. Similarly, the performance of the proposed algorithm improves as γ increases as depicted in Figure 4. This investigation therefore allows for a proper choice of α and γ to be made.

In order to show the importance of varying the step-size, two experiments were run separately. In the first experiment, the DLMS algorithm was simulated with a high step-size while the initial value for the proposed algorithm was kept both low and high. In the second experiment, the step-size of the DLMS algorithm was changed to a low value. As can be seen from Figures 5 and 6, the proposed algorithm converges to the same error floor for both scenarios. However, the DLMS algorithm converges fast but at a higher error floor in Figure 5. The low value of step-size results in the DLMS algorithm converging at the same error floor as the proposed algorithm but very slowly.

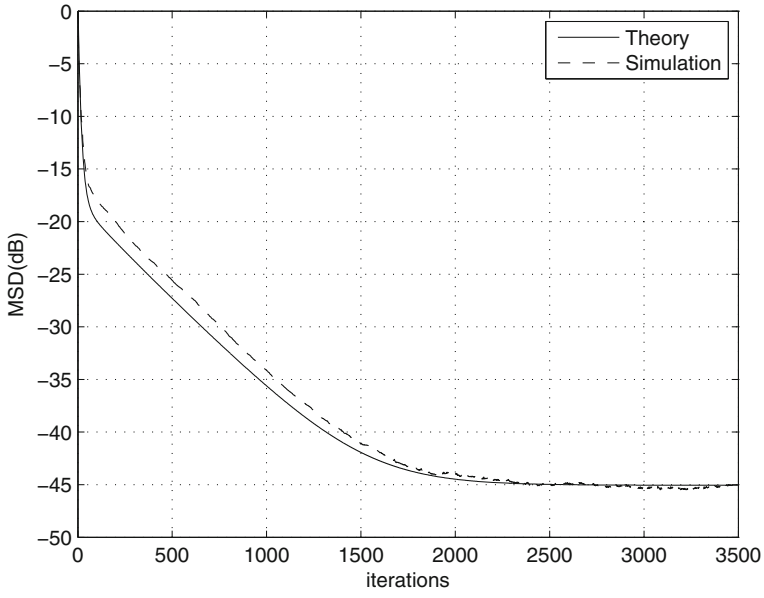

Figure 9 Theoretical and simulated MSD with $\alpha = 0.95$ and $\gamma = 0.001$.

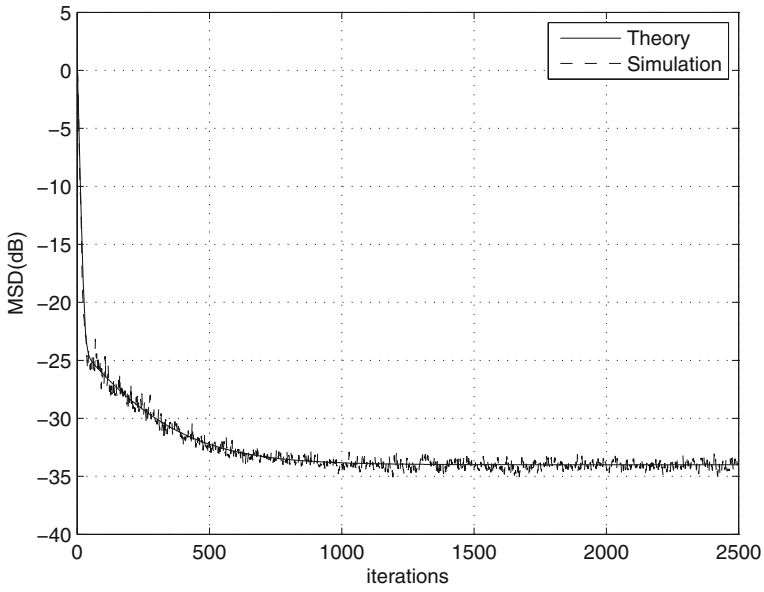

Figure 10 Theoretical and simulated MSD with $\alpha = 0.995$ and $\gamma = 0.001$.

Thus, the proposed algorithm provides fast convergence as well as better performance.

Next, the proposed algorithm is compared with some key existing algorithms, which are the no-cooperation LMS, the distributed LMS [10], the DLMS [6], the DLMS with adaptive combiners (DLMSAC) [8] and the DRLS [14]. Figure 7 reports the performance behavior of these different algorithms. As can be seen from this figure, the performance of the proposed VSSDLMS algorithm is second only that of the DRLS algorithm. However, the gap in performance is narrow. These results show that when compared with other algorithms of similar complexity, the VSSDLMS algorithm exhibits a significant improvement in performance. A similar performance in the steady-state behavior of the proposed VSSDLMS algorithm is obtained as shown in Figure 8. As expected,

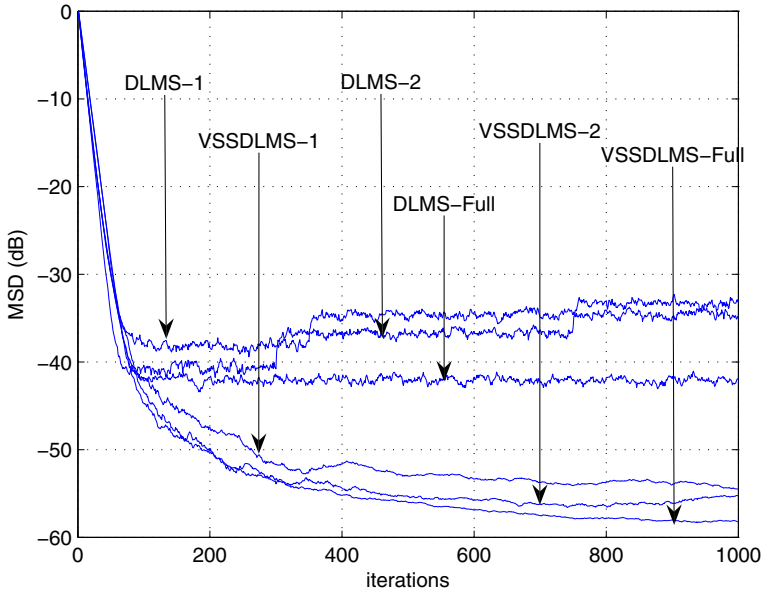

Figure 11 Robustness of algorithm at SNR = 20 dB.

Table 1 Steady-state values for MSD and EMSE

$\mathbf{\Lambda}$	MSD (dB)		EMSE (dB)	
	Equation (54)	Simulations	Equation (55)	Simulations
$\mathbf{I}_{MN}$	−63.7800	−63.2838	−63.7800	−63.2814
Diag $\left\{\sigma^2_{\mathbf{u},k}\mathbf{I}_M\right\}$	−63.3310	−63.5882	−58.4950	−58.8067

the DRLS algorithm performs better than all other algorithms included in this comparison, but the proposed algorithm remains second only to the DRLS algorithm in the steady-state mode. Also, the diffusion process can be appropriately viewed as an efficient and indirect way of adjusting the step-size in all neighboring nodes, which resulted in keeping the steady-state MSD for all nodes nearly the same for all cases.

Next, the comparison between the results predicted by the theoretical analysis of the proposed algorithm and the simulation results is reported in Figures 9 and 10. As can be seen from these figures, the simulation analysis corroborates the theoretical findings very well. This is done for a network of 15 nodes with $M = 2$ and a communication range of 0.35. Two values for α, namely $\alpha = 0.995$ and $\alpha = 0.95$, are chosen whereas $\gamma = 0.001$.

An important aspect of working with sensor nodes is the possibility of a node switching off. In such a case the network may be required to adapt itself. The diffusion scheme is robust to such a change, and this scenario has been considered here and results are shown in Figure 11. A network of 50 nodes is chosen so that enough nodes can be switched off in order to study the performance of the proposed algorithm in this scenario. Two cases are considered. In the first case, 15 nodes are turned off after 50 iterations and then a further 15 nodes are switched off after 300 iterations. In the second case, 15 nodes are switched off after 250 iterations and the next 15 nodes are switched off after 750 iterations. In both cases, the performance degrades initially but recovers to give a similar performance to the case where there are no nodes being switched off. The difference between the best and worst case scenarios is only about 2 dB. For the DLMS algorithm, however, the performance is worse off the earlier the nodes are switched off. The difference between the best and worst case scenarios is almost 9 dB, which further enhances the robustness of the proposed algorithm.

Finally, the comparison between the theoretical and simulated steady-state values for the MSD and EMSE for two different input regressor auto-correlation matrices is given in Table 1. As can be seen from this table, a close agreement between theory and simulations has been obtained.

5 Conclusions

The proposed variable step-size diffusion LMS (VSS-DLMS) algorithm has been discussed in detail. Several popular VSSLMS algorithms are investigated and the algorithm providing the best trade-off between complexity and performance is chosen as the proposed VSSDLMS algorithm. Complete convergence and steady-state analyses have been carried out to assess the performance of the proposed algorithm. Simulations have been carried out under different scenarios and with different SNR values. A sensitivity analysis has been carried out through extensive simulations. Based on the results of this analysis, the values for the parameters of the VSSLMS algorithm were chosen. The proposed algorithm has been compared with existing algorithms of similar complexity and it has been shown that the proposed algorithm performed significantly better. Theoretical results were also compared with simulation results and the two were found to be in close agreement with each other. The proposed algorithm was then tested under different scenarios to assess its robustness. Finally, a steady-state comparison between theoretical and simulated results was carried out and tabulated and the results were also found to be in close agreement with each other.

Competing interests
The authors declare that they have no competing interests.

Acknowledgements
This research work is funded by King Fahd University of Petroleum and Minerals (KFUPM) under research grants FT100012 and RG1112-1&2.

Received: 15 July 2013 Accepted: 26 July 2013
Published: 6 August 2013

References

1. D Estrin, L Girod, G Pottie, M Srivastava, Instrumenting the world with wireless sensor networks, in *Proceedings of the ICASSP* (Salt Lake City, 07–11 May 2001), pp. 2033–2036
2. R Olfati-Saber, JA Fax, RM Murray, Consensus and cooperation in networked multi-agent systems. Proc. IEEE. **95**(1), 215–233 (2007)
3. CG Lopes, AH Sayed, Incremental adaptive strategies over distributed networks. IEEE Trans. Signal Process. **55**(8), 4064–4077 (2007)
4. SS Ram, A Nedic, VV Veeravalli, Stochastic incremental gradient descent for estimation in sensor networks, in *Proceedings of the 41st Asilomar Conference on Signals, Systems and Computers* (Pacific Grove, 4–7 November 2007), pp. 582–586
5. A Rastegarnia, MA Tinati, A Khalili, Performance analysis of quantized incremental LMS algorithm for distributed adaptive estimation. Signal Process. **90**(8), 2621–2627 (2010)
6. CG Lopes, AH Sayed, Diffusion least-mean squares over adaptive networks: formulation and performance analysis. IEEE Trans. Signal Process. **56**(7), 3122–3136 (2008)
7. F Cattivelli, AH Sayed, Diffusion LMS strategies for distributed estimation. IEEE Trans. Signal Process. **58**(3), 1035–1048 (2010)
8. N Takahashi, I Yamada, AH Sayed, Diffusion least mean squares with adaptive combiners: formulation and performance analysis. IEEE Trans. Signal Process. **58**(9), 4795–4810 (2010)

9. ID Schizas, G Mateos, GB Giannakis, Distributed LMS for consensus-based in-network adaptive processing. IEEE Trans. Signal Process. **57**(6), 2365–2382 (2009)
10. G Mateos, ID Schizas, GB Giannakis, Performance analysis of the consensus-based distributed LMS algorithm. EURASIP J. Adv. Signal Process (2009). doi:10.1155/2009/981030
11. S Haykin, *Adaptive Filter Theory*, 4th edn. (Prentice-Hall, Englewood Cliffs, 2000)
12. CH Papadimitriou, *Computational Complexity* (Addison-Wesley, Reading, MA, 1993)
13. MO BinSaeed, A Zerguine, SA Zummo, Variable step size least mean square algorithms over adaptive networks, in *Proceedings of ISSPA 2010* (Kuala Lampur, 10–13 May 2010), pp. 381–384
14. FS Cattivelli, CG Lopes, AH Sayed, Diffusion recursive least-squares for distributed estimation over adaptive networks. IEEE Trans. Signal Process. **56**(5), 1865–1877 (2008)
15. G Mateos, ID Schizas, GB Giannakis, Distributed recursive least-squares for consensus-based in-network adaptive estimation. IEEE Trans. Signal Process. **57**(11), 4583–4588 (2009)
16. G Mateos, GB Giannakis, Distributed recursive least-squares: stability and performance analysis. IEEE Trans. Signal Process. **60**(7), 3740–3754 (2012)
17. F Cattivelli, AH Sayed, Diffusion strategies for distributed Kalman filtering and smoothing. IEEE Trans. Automatic Control. **55**(9), 2069–2084 (2010)
18. R Olfati-Saber, Distributed Kalman filtering for sensor networks, in *Proceedings of IEEE CDC 2007* (New Orleans, 12–14 December 2007), pp. 5492–5498
19. O Hlinka, F Hlawatsch, PM Djuric, Distributed particle filtering in agent networks: a survey, classification, and comparison. IEEE Signal Proc. Mag. **30**(1), 61–81 (2013)
20. RH Kwong, EW Johnston, A variable step size LMS algorithm. IEEE Trans. Signal Process. **40**(7), 1633–1642 (1992)
21. T Aboulnasr, K Mayyas, A robust variable step-size LMS-type algorithm: analysis and simulations. IEEE Trans. Signal Process. **45**(3), 631–639 (1997)
22. MH Costa, JCM Bermudez, A robust variable step-size algorithm for LMS adaptive filters, in *Proceedings of the ICASSP* (Toulouse, 14–19 May 2006), pp. 93–96
23. CG Lopes, JCM Bermudez, Evaluation and design of variable step size adaptive algorithms, in *Proceedings of the ICASSP* (Salt Lake City, 7–11 May 2001), pp. 3845–3848
24. VJ Mathews, Z Xie, A stochastic gradient adaptive filter with gradient adaptive step size. IEEE Trans. Signal Process. **41**(6), 2075–2087 (1993)
25. AI Sulyman, A Zerguine, Convergence and steady-state analysis of a variable step-size NLMS algorithm. Signal Process. **83**(6), 1255–1273 (2003)
26. AH Sayed, *Fundamentals of Adaptive Filtering* (Wiley, New York, 2003)
27. RH Koning, H Neudecker, T Wansbeek, Block Kronecker products and the vecb operator. Economics Department, Institute of Economics Research, University of Groningen, Groningen, The Netherlands, Research Memo no. 351, 1990

doi:10.1186/1687-6180-2013-135
Cite this article as: Bin Saeed *et al.*: **A variable step-size strategy for distributed estimation over adaptive networks.** *EURASIP Journal on Advances in Signal Processing* 2013 **2013**:135.