INTERNATIONAL JOURNAL OF ADAPTIVE CONTROL AND SIGNAL PROCESSING
Int. J. Adapt. Control Signal Process. 2013; **27**:827–845
Published online 28 October 2012 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/acs.2358

A noise-constrained algorithm for estimation over distributed networks

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

King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia

SUMMARY

Much research has been devoted recently to the development of algorithms to utilize the distributed structure of an ad hoc wireless sensor network for the estimation of a certain parameter of interest. A successful solution is the algorithm called the diffusion least mean squares algorithm. The algorithm estimates the parameter of interest by employing cooperation between neighboring sensor nodes within the network. The present work derives a new algorithm by using the noise constraint that is based on and improves the diffusion least mean squares algorithm. In this work, first the derivation of the noise constraint-based algorithm is given. Second, detailed convergence and steady-state analyses are carried out, including analyses for the case where there is mismatch in the noise variance estimate. Finally, extensive simulations are carried out to test the robustness of the proposed algorithm under different scenarios, especially the mismatch scenario. Moreover, the simulation results are found to corroborate the theoretical results very well. Copyright © 2012 John Wiley & Sons, Ltd.

Received 20 April 2011; Revised 19 September 2012; Accepted 6 October 2012

KEY WORDS: noise-constrained LMS algorithm; diffusion LMS algorithm; distributed networks

1. INTRODUCTION

Ad hoc wireless sensor networks have renewed an interest in distributed computing and opened several venues for research in the estimation and tracking of parameters of interest in situations where a robust, scalable, and low cost solution is required. To illustrate the subject matter more clearly, consider a set of N sensor nodes spread over a geographic area, as shown in Figure 1, with each node taking sensor measurements at every time instant. The goal is to estimate an unknown parameter of interest by using these measurements. In a centralized network, each node transmits its readings to a fusion center for processing. However, this system is prone to center failure. Furthermore, large amounts of energy and communication resources are often required for the complete signal processing to take place between the center and the network. This could become considerable as the distance between the nodes and the center increases [1].

On the other hand, an ad hoc network depends on distributed processing, and the nodes need only to communicate with their neighbors. The processing takes place at each node. Because no hierarchical structure is involved, the network is robust to node failure. Extensive research has been carried out in the area of consensus-based distributed signal processing and resulted in a variety of algorithms [2].

We begin with a brief overview of these algorithms [3–9], looking at their virtues and limitations and therefore, justifying our contribution. The first algorithm organizes the network by using a Hamiltonian cycle [3]. The estimation is carried out by passing the estimate cyclicly from node

*Correspondence to: Azzedine Zerguine, King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia.
†E-mail: azzedine@kfupm.edu.sa

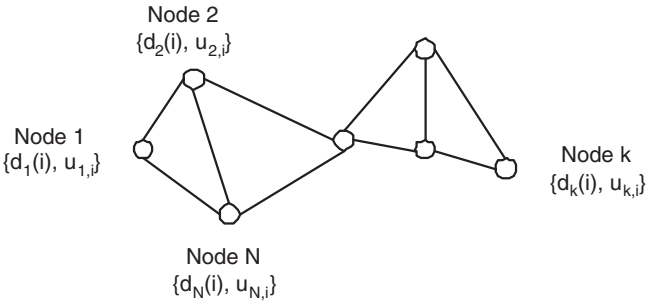

Figure 1. Adaptive network of N nodes.

to node, the estimate being improved with each new set of data per node. This algorithm is termed incremental least mean squares (LMS) algorithm as it uses the LMS algorithm [10] with incremental steps within every iteration [3]. These algorithms are heavily dependent on the Hamiltonian cycle and are prone to node failure, which would mean that a new cycle has to be reestablished and that becomes a nondeterministic polynomial-time hard problem [11]. A fully distributed algorithm was later proposed in [4] and termed diffusion LMS (DLMS) algorithm. The neighbor nodes share their estimates in this algorithm in order the overall performance. A modification incorporated to this algorithm [5] further improved the performance of the algorithm. Ultimately, this algorithm is robust to node failure as the network is not dependent on any single node or any cyclic structuring of the nodes. The authors in [4] introduced a scheme to adapt the combination weights at each iteration for each node instead of having fixed weights for the shared data. This scheme was further improved upon in [6]. The performance of the DLMS algorithm improved in this case if more weight is given to the estimates of the neighbor nodes that are providing more improvement per iteration.

All the previously mentioned algorithms were designed on a nonconstrained optimization technique. However, the authors in [7] used 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 using the same constraint was suggested in [8]. Compared with the work in [6], a much simpler solution was suggested in [9], using a variable step size LMS algorithm. This resulted in the variable step size DLMS (VSSDLMS) algorithm.

The performance of each node is affected by additive noise. If the noise variance is known at each node, the performance of the algorithm is likely to improve. Inspired by the noise constraint-based algorithm in [12], a new algorithm was devised in [13], termed the noise-constrained DLMS (NCDLMS) algorithm. Preliminary results showed that remarkable improvement in performance at low computational complexity was achieved using the proposed NCDLMS algorithm.

This work extends that of [13] and investigates in detail the NCDLMS algorithm where in particular the transient, steady-state, and sensitivity analyses are for the first time reported in this work. First, the NCDLMS algorithm is derived by rearranging the cost function to incorporate the noise constraint. Second, complete convergence and steady-state analyses are carried out. The case where there exists a mismatch in the estimation of noise variance is also included. Also, the stability of the algorithm is derived. Finally, simulations are carried out for the proposed algorithm and existing algorithms of similar complexity. The performance of the proposed algorithm is assessed under different conditions. The theoretical results are found to match the simulation results remarkably well.

The paper is divided as follows. Section 2 describes the problem statement and briefly introduces the DLMS algorithm. Section 3 derives the proposed NCDLMS algorithm. Complete convergence and stability analyses are carried out, including the case of noise variance estimate mismatch, in Section 4. Simulation results are given in Section 5 followed by a discussion on the results. Finally, Section 6 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 $(.)^T$

DOI: 10.1002/acs

stands for transposition for vectors and matrices and expectation operation is denoted by $E[.]$. Any mathematical operators that have been used will be defined as they are introduced during the paper.

2. PROBLEM STATEMENT

Consider a network of N sensor nodes deployed over a geographical area for estimating an unknown parameter vector $\mathbf{w}^o$ 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

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

where $v_k(i)$ is spatially uncorrelated zero-mean additive white Gaussian noise with variance $\sigma^2_{v,k}$ and i denotes the discrete time index. The measurements, $d_k(i)$ and $\mathbf{u}_k(i)$, are used to generate an estimate $\mathbf{w}_k(i)$ of the unknown parameter vector $\mathbf{w}^o$. By assuming that each node cooperates only with its neighbors, each node k has access to updates $\mathbf{w}_l(i)$, from its $\mathcal{N}_k$ neighbor nodes at every time instant i, where $l \in \mathcal{N}_k$, in addition to its own estimate, $\mathbf{w}_k(i)$. Here, the neighborhood $l \neq k$ of node k is defined as those nodes to which node k is communicating directly and includes node k itself. Two different schemes have been introduced in the literature for the diffusion algorithm. The adapt-then-combine (ATC) scheme [5] first updates the local estimate by using the adaptive algorithm, and then, the intermediate estimates from the neighbors are fused together. The second scheme, called combine-then-adapt [4], reverses the order. It is found that the ATC scheme outperforms the combine-then-adapt scheme [5], and therefore, this work uses the ATC scheme.

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

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

where $\mathbf{w}$ is the estimate of $\mathbf{w}^o$. The steepest descent algorithm is given as

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

where $\mathbf{r}_{d\mathbf{u},k} = \mathrm{E}\left[d_k\mathbf{u}_k^T\right]$ is the cross-correlation between d_k and $\mathbf{u}_k$, and $\mathbf{R}_{\mathbf{u},k} = \mathrm{E}\left[\mathbf{u}_k^T\mathbf{u}_k\right]$ is the autocorrelation of $\mathbf{u}_k$. 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 [4] gives a fully distributed solution, which is modified and improved in [5]. By using the ATC scheme, the DLMS algorithm is given as [5]

$$\mathbf{\Psi}_k(i) = \mathbf{w}_k(i-1) + \mu_k\mathbf{u}_k^T(i)\left[d_k(i) - \mathbf{u}_k(i)\mathbf{w}_k(i-1)\right], \tag{4}$$

$$\mathbf{w}_k(i) = \sum_{l \in \mathcal{N}_k} c_{lk}\mathbf{\Psi}_l(i), \tag{5}$$

where $\mathbf{\Psi}_k(i)$ is the intermediate update and c_{lk} is a real nonnegative scalar weight connecting node k to its neighboring node $l \in \mathcal{N}_k$, where $\sum_{l \in \mathcal{N}_k} c_{lk} = 1$.

The DLMS algorithm, defined by (4) and (5), uses a fixed step size. Using a constraint would make the step size in the DLMS algorithm act as a variable step size, for example, the normalized LMS algorithm [14]. As a result, the algorithm would converge faster and give a lower steady-state error, which is not the case for the DLMS algorithm. Ultimately, a new algorithm based on the noise constraint is being presented next.

DOI: 10.1002/acs

3. NOISE-CONSTRAINED DIFFUSION LMS ALGORITHM

From (2), we can write the local cost function for each node k as

$$J_k(\mathbf{w}) = \mathrm{E}\left[|d_k - \mathbf{u}_k\mathbf{w}|^2\right], \tag{6}$$

where completing the squares and letting $\mathrm{E}\left[\mathbf{u}_k^T\mathbf{u}_k\right] = \mathbf{R}_{\mathbf{u},k}$ gives

$$J_k(\mathbf{w}) = \|\mathbf{w} - \mathbf{w}_k\|^2_{\mathbf{R}_{\mathbf{u},k}} + \mathrm{MMSE}, \tag{7}$$

where $\mathbf{w}$ is the global estimate, $\mathbf{w}_k$ is the local estimate at node k, and MMSE represents the terms that do not include $\mathbf{w}$ and can, therefore, be ignored. Incorporating (7) into (2), the global cost function can be written as

$$\begin{aligned} J_{\mathrm{gl}}(\mathbf{w}) &= J_k(\mathbf{w}) + \sum_{l\neq k}^{N} J_l(\mathbf{w}) \\ &= \mathrm{E}\left[|d_k - \mathbf{u}_k\mathbf{w}|^2\right] + \sum_{l\neq k}^{N} \|\mathbf{w} - \mathbf{w}_l\|^2_{\mathbf{R}_{\mathbf{u},l}}. \end{aligned} \tag{8}$$

This model assumes that any node k has access to data across the entire network. However, this is not a practical assumption as node k has access only to its neighbors. As a result, the cost function is approximated with data from neighbors being shared at each node. The resulting weighting matrix for the second term in (8) changes from $\mathbf{R}_{\mathbf{u},l}$ to a nonnegative constant weighting factor b_{lk}, where the subscript lk denotes the connection between node k with its neighbor node l and $\sum_{l\in\mathcal{N}_k} b_{lk} = 1$. The approximated cost function looks like

$$\begin{aligned} J(\mathbf{w}_k) &= \mathrm{E}\left[|d_k - \mathbf{u}_k\mathbf{w}_k|^2\right] + \sum_{\substack{l\in\mathcal{N}_k \\ l\neq k}} b_{lk}\,\|\mathbf{w}_k - \mathbf{w}_l\|^2 \\ &= J_k(\mathbf{w}_k) + \sum_{\substack{l\in\mathcal{N}_k \\ l\neq k}} b_{lk}\,\|\mathbf{w}_k - \mathbf{w}_l\|^2, \end{aligned} \tag{9}$$

where the cost function now becomes as no node has access to the data from the entire network. By assuming that the additive noise variance, $\sigma^2_{v,k}$, is known, the cost function can be modified using Lagrange multipliers as follows:

$$J'(\mathbf{w}_k) = J_k(\mathbf{w}_k) + \sum_{\substack{l\in\mathcal{N}_k \\ l\neq k}} b_{lk}\,\|\mathbf{w}_k - \mathbf{w}_l\|^2 + \gamma\beta_k\left(J_k(\mathbf{w}_k) - \sigma^2_{v,k}\right) - \gamma\beta_k^2, \tag{10}$$

where $\gamma > 0$ and the last term is a correction term added because the critical values of β_k are not unique (or bounded) and this may present a potential problem for an adaptive filter [12, 15, 16].

The solution for (10) can be obtained from the Robbins–Monro algorithm [12, 15–17]

$$\mathbf{w}_k(i) = \mathbf{w}_k(i-1) - \mu_k\frac{\partial J'_k(\mathbf{w}_k)}{\partial\mathbf{w}_k}, \tag{11}$$

$$\beta_k(i+1) = \beta_k(i) + \alpha\frac{\partial J'_k(\mathbf{w}_k)}{\partial\beta_k}. \tag{12}$$

3.1. Steepest descent solution

Solution of the first partial derivative is given by

$$\frac{\partial J_k(\mathbf{w}_k)}{\partial\mathbf{w}_k} = (1+\gamma\beta_k)(\mathbf{R}_{\mathbf{u},k}\mathbf{w}_k - \mathbf{r}_{d\mathbf{u},k}) + 2\sum_{\substack{l\in\mathcal{N}_k \\ l\neq k}} b_{lk}(\mathbf{w}_k - \mathbf{w}_l). \tag{13}$$

DOI: 10.1002/acs

Similarly, the solution of the second partial derivative is

$$\frac{\partial J'_k(\mathbf{w}_k)}{\partial \beta_k} = \gamma \left(\mathrm{E}\left[|d_k - \mathbf{u}_k \mathbf{w}_k|^2 \right] - \sigma_{v,k}^2 \right) - 2\gamma\beta_k, \tag{14}$$

which results in

$$\beta_k(i+1) = \beta_k(i) + \alpha\gamma \left(\mathrm{E}\left[|d_k - \mathbf{u}_k \mathbf{w}_k|^2 \right] - \sigma_{v,k}^2 \right) - 2\alpha\gamma\beta_k(i). \tag{15}$$

If we replace $\alpha\gamma$ by $\alpha/2$ and then insert the solutions to the partial derivatives into the algorithm, we obtain the resulting steepest descent solution

$$\begin{aligned} \mathbf{w}_k(i) = \mathbf{w}_k(i-1) &+ \mu_k(1+\gamma\beta_k(i)) \left(\mathbf{R}_{\mathbf{u},k} \mathbf{w}_k(i-1) - \mathbf{r}_{d\mathbf{u},k} \right) \\ &+ \upsilon_k \sum_{\substack{l \in \mathcal{N}_k \\ l \neq k}} b_{lk} (\mathbf{w}_l(i-1) - \mathbf{w}_k(i-1)), \end{aligned} \tag{16}$$

$$\beta_k(i+1) = (1-\alpha)\beta_k(i) + \frac{\alpha}{2} \left(\mathrm{E}\left[|d_k - \mathbf{u}_k \mathbf{w}_k|^2 \right] - \sigma_{v,k}^2 \right), \tag{17}$$

where υ_k is a step size different from μ_k. Now, (16) can be written as a two-step process

$$\boldsymbol{\Psi}_k(i) = \mathbf{w}_k(i-1) + \mu_k(1+\gamma\beta_k(i))(\mathbf{R}_{\mathbf{u},k}\mathbf{w}_k - \mathbf{r}_{d\mathbf{u},k}), \tag{18}$$

$$\begin{aligned} \mathbf{w}_k(i) &= \boldsymbol{\Psi}_k(i) + \upsilon_k \sum_{\substack{l \in \mathcal{N}_k \\ l \neq k}} b_{lk} (\boldsymbol{\Psi}_l(i) - \boldsymbol{\Psi}_k(i)) \\ &= \boldsymbol{\Psi}_k(i)(1 - \upsilon_k + b_{kk}\upsilon_k) + \upsilon_k \sum_{\substack{l \in \mathcal{N}_k \\ l \neq k}} b_{lk} \boldsymbol{\Psi}_l(i) \\ &= \sum_{l \in \mathcal{N}_k} c_{lk} \boldsymbol{\Psi}_l(i), \end{aligned} \tag{19}$$

where

$$c_{lk} = \begin{cases} 1 - \upsilon_k + \upsilon_k b_{kk}, & l = k \\ \upsilon_k b_{lk}, & l \neq k \end{cases}, \quad s.t. \sum_{l \in \mathcal{N}_k} c_{lk} = 1, \quad \forall\{k,l\}. \tag{20}$$

Combining (18) and (19) with (17) results in the steepest descent solution to the noise-constrained diffusion problem.

3.2. The proposed algorithm

The steepest descent solution requires complete statistical knowledge of the data. For a practical adaptive solution, we simply replace $\mathbf{R}_{\mathbf{u},k}$, $\mathbf{R}_{d\mathbf{u},k}$, and $\mathrm{E}\left[|d_k - \mathbf{u}_k \mathbf{w}|^2 \right]$ by their instantaneous values. Noting that $e_k(i) = d_k(i) - \mathbf{u}_k(i)\mathbf{w}_k(i-1)$, we obtain

$$\boldsymbol{\Psi}_k(i) = \mathbf{w}_k(i-1) + \mu_k(1+\gamma\beta_k(i)) \mathbf{u}_k^T(i) e_k(i), \tag{21}$$

$$\mathbf{w}_k(i) = \sum_{l \in \mathcal{N}_k} c_{lk} \boldsymbol{\Psi}_l(i), \tag{22}$$

$$\beta_k(i+1) = (1-\alpha)\beta_k(i) + \frac{\alpha}{2} \left(e_k^2(i) - \sigma_{v,k}^2 \right), \tag{23}$$

where c_{lk} are defined in (20).

When compared with the DLMS algorithm, (21) has an extra factor consisting of the Lagrangian multiplier, which is recursively updated according to (23). Hence, (21)–(23) form the proposed NCDLMS algorithm.

DOI: 10.1002/acs

4. ANALYSIS OF THE PROPOSED ALGORITHM

To perform the analysis, the whole network needs to be looked at because any node k is being affected by its neighbors and the neighbors in turn are affected by their respective neighbors. Therefore, we introduce new terms to study the performance of the network in a global manner. Similarly, as it was carried out in [4], the local variables are transformed into global variables as follows:

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

By using these new variables, a completely new set of equations is formed, representing the entire network. Starting with the relation between the measurements, we have

$$\mathbf{d}(i) = \mathbf{U}(i)\mathbf{w}^{(o)} + \mathbf{v}(i), \tag{24}$$

where $\mathbf{w}^{(o)} = \mathbf{Q}\mathbf{w}^o$ and $\mathbf{Q} = \mathrm{col}\{\mathbf{I}_M, \mathbf{I}_M, \ldots, \mathbf{I}_M\}$ is a $MN \times M$ matrix. Similarly, the update equations can be remodeled to represent the entire network instead of representing just a single node as follows:

$$\boldsymbol{\Psi}(i) = \mathbf{w}(i-1) + \mathbf{D}\left(\mathbf{I}_{MN} + \gamma\mathbf{B}(i)\right)\mathbf{U}^T(i)\left(\mathbf{d}(i) - \mathbf{U}(i)\mathbf{w}(i-1)\right), \tag{25}$$

$$\mathbf{w}(i) = \mathbf{G}\boldsymbol{\Psi}(i), \tag{26}$$

$$\mathbf{B}(i+1) = (1-\alpha)\mathbf{B}(i) + \frac{\alpha}{2}\left(\mathcal{E}(i) - \mathbf{S}\right), \tag{27}$$

where $\mathbf{G} = \mathbf{C} \otimes \mathbf{I}_M$, $\mathbf{C}$ is the $N \times N$ weighting matrix, $\otimes$ is the Kronecker operator, $\mathbf{B}(i) = \mathrm{diag}\{\beta_1\mathbf{I}_M, \ldots, \beta_N\mathbf{I}_M\}$ is the diagonal update matrix for the Lagrange multipliers, $\mathcal{E}(i) = \mathrm{diag}\left\{e_1^2(i)\mathbf{I}_M, \ldots, e_N^2(i)\mathbf{I}_M\right\}$ is the diagonal matrix for instantaneous error, and $\mathbf{S} = \mathrm{diag}\left\{\sigma_1^2\mathbf{I}_M, \ldots, \sigma_N^2\mathbf{I}_M\right\}$ is the diagonal matrix containing the estimated noise variances for all nodes. Here, it is assumed that the noise variances have been estimated exactly.

4.1. Mean analysis

To begin with, let us introduce the global weight-error vector

$$\tilde{\mathbf{w}}(i) = \mathbf{w}^{(o)} - \mathbf{w}(i). \tag{28}$$

Because $\mathbf{G}\mathbf{w}^{(o)} \triangleq= \mathbf{w}^{(o)}$, incorporating the global weight-error vector into (25) and (26), we obtain

$$\begin{aligned} \tilde{\mathbf{w}}(i) &= \mathbf{G}\tilde{\boldsymbol{\Psi}}(i) \\ &= \mathbf{G}\tilde{\mathbf{w}}(i-1) - \mathbf{G}\mathbf{D}\left(\mathbf{I}_{MN} + \gamma\mathbf{B}(i)\right)\mathbf{U}^T(i)\left(\mathbf{U}(i)\tilde{\mathbf{w}}(i-1) + \mathbf{v}(i)\right) \\ &= \mathbf{G}\left(\mathbf{I}_{MN} - \mathcal{D}(i)\mathbf{U}^T(i)\mathbf{U}(i)\right)\tilde{\mathbf{w}}(i-1) - \mathbf{G}\mathcal{D}(i)\mathbf{U}^T(i)\mathbf{v}(i). \end{aligned} \tag{29}$$

Taking the expectation on both sides of the aforementioned equation gives

$$\mathrm{E}\left[\tilde{\mathbf{w}}(i)\right] = \mathbf{G}\left(\mathbf{I}_{MN} - \mathcal{D}(i)\mathbf{R}_{\mathbf{U}}\right)\mathrm{E}\left[\tilde{\mathbf{w}}(i-1)\right], \tag{30}$$

where $\mathbf{R}_{\mathbf{U}} = \mathrm{E}\left[\mathbf{U}^T\mathbf{U}\right]$ is the regressor autocorrelation matrix for the entire network and the matrix $\mathcal{D}(i) = \mathbf{D}\left(\mathbf{I}_{MN} + \gamma\mathbf{B}(i)\right)$ is assumed to be independent of the regressor matrix as it depends only on the values from the previous $(i-1)$ iterations. Also, because the measurement noise is spatially uncorrelated, the expectation of the second part of the right-hand side of (29) is zero.

DOI: 10.1002/acs

From [5], we see that the diffusion algorithm is stable if the combination weights are restricted to within the unit circle. However, in this case, stability is also dependent on the Lagrange multipliers. In this case, the algorithm will be stable if, for each node,

$$\prod_{i=1}^{n} \left(\mathbf{I}_M - \mu_k \left(1 + \gamma \mathrm{E}[\beta(i)]\right) \mathbf{R}_{\mathbf{u},k}\right) \to 0, \quad \text{as } n \to \infty, \tag{31}$$

which holds true if

$$0 < \mu_k < \frac{2}{\left(1 + \gamma \mathrm{E}[\beta(i)]\right) \lambda_{\max}\left(\mathbf{R}_{\mathbf{u},k}\right)}, \quad 1 \leqslant k \leqslant N, \tag{32}$$

where $\lambda_{\max}\left(\mathbf{R}_{\mathbf{u},k}\right)$ is the maximum eigenvalue of the autocorrelation matrix $\mathbf{R}_{\mathbf{u},k}$. The step size limit is dependent on the stability of the Lagrangian multiplier. When the expectation operator is applied to (27), the Lagrange multiplier for each node gets updated as follows:

$$\begin{aligned} \mathrm{E}[\beta_k\,(i+1)] &= (1-\alpha)\,\mathrm{E}[\beta_k(i)] + \frac{\alpha}{2}\left(\mathrm{E}\left[e_k^2(i)\right] - \sigma_{v,k}^2\right) \\ &= (1-\alpha)\,\mathrm{E}[\beta_k(i)] + \frac{\alpha}{2}\mathrm{EMSE}_k(i), \end{aligned} \tag{33}$$

where

$$\mathrm{EMSE}_k(i) = \mathrm{E}\left[e_k^2(i)\right] - \sigma_{v,k}^2 \tag{34}$$

is the excess mean square error for node k at iteration i, so the choice of the step size depends on the choice of the parameters α and γ. The product of the step size and γ decide how much impact the Larangian multiplier will have on the adaptive algorithm. This product has to be small for the algorithm to converge. By considering $(1-\alpha)$ as a forgetting factor here, the value of α needs to be small so that the forgetting factor value is close to one. Simulation results support this argument.

4.1.1. Effect of noise variance estimate mismatch. The previous analysis assumed perfect noise variance estimation. However, in a practical system, it is not always possible to have an exact estimate, and a mismatch can occur. The analysis in this case may be altered slightly to include the effect of the mismatch. The Lagrangian does not converge as in (32) because of the mismatch. Taking the expectation of (23), we have

$$\begin{aligned} \mathrm{E}\left[\beta_k\,(i+1)\right] &= (1-\alpha)\,\mathrm{E}[\beta_k(i)] + \frac{\alpha}{2}\mathrm{E}\left[e_k^2(i) - \hat{\sigma}_{v,k}^2\right] \\ &= (1-\alpha)\,\mathrm{E}[\beta_k(i)] + \frac{\alpha}{2}\left(\mathrm{EMSE}\,(i) + \sigma_{v,k}^2 - \hat{\sigma}_{v,k}^2\right) \\ &= (1-\alpha)\,\mathrm{E}\left[\beta_{k,i-1}\right] + \frac{\alpha}{2}\left(\mathrm{EMSE}\,(i) + \tilde{\sigma}_{v,k}^2\right), \end{aligned} \tag{35}$$

where $\hat{\sigma}_{v,k}^2$ is the imperfect estimate of the noise variance for node k and $\tilde{\sigma}_{v,k}^2$ is the noise variance mismatch. Thus, at steady-state, the Lagrange multiplier becomes

$$\beta_{k,ss} = \frac{1}{2}\left(\mathrm{EMSE}_{ss} + \tilde{\sigma}_{v,k}^2\right), \tag{36}$$

which is simply a summation of the steady-state EMSE and the noise power mismatch. Because the value of EMSE is reasonably small, the bound on the step size can be approximated by

$$0 < \mu_k < \frac{2}{\left(1 + \tilde{\sigma}_{v,k}^2/2\right) \lambda_{\max}\left(\mathbf{R}_{\mathbf{u},k}\right)}. \tag{37}$$

DOI: 10.1002/acs

In case of the zero NCDLMS (ZNCDLMS) algorithm, there is no estimate for the noise power, so the limit can be approximated as

$$0 < \mu_k < \frac{2}{\left(1 + \sigma_{v,k}^2/2\right) \lambda_{\max}\left(\mathbf{R}_{\mathbf{u},k}\right)}, \tag{38}$$

where the mismatch of the estimate, $\tilde{\sigma}_{v,k}^2$, is replaced by the actual noise power, $\sigma_{v,k}^2$.

4.2. Mean square analysis

Taking the weighted norm of (29) and applying the expectation operator yield

$$\begin{aligned}
\mathrm{E}\left[\|\tilde{\mathbf{w}}(i)\|_{\boldsymbol{\Sigma}}^2\right] &= \mathrm{E}\left[\left\|\mathbf{G}\left(\mathbf{I}_{MN} - \mathcal{D}(i)\mathbf{U}^T(i)\mathbf{U}(i)\right)\tilde{\mathbf{w}}(i-1) - \mathbf{G}\mathcal{D}(i)\mathbf{U}^T(i)\mathbf{v}(i)\right\|_{\boldsymbol{\Sigma}}^2\right] \\
&= \mathrm{E}\left[\|\tilde{\mathbf{w}}(i-1)\|_{\mathbf{G}^T\boldsymbol{\Sigma}\mathbf{G}}^2\right] - \mathrm{E}\left[\|\tilde{\mathbf{w}}(i-1)\|_{\mathbf{G}^T\boldsymbol{\Sigma}\mathbf{L}(i)\mathbf{U}(i)}^2\right] - \mathrm{E}\left[\|\tilde{\mathbf{w}}(i-1)\|_{\mathbf{U}^T(i)\mathbf{L}^T(i)\boldsymbol{\Sigma}\mathbf{G}}^2\right] \\
&\quad + \mathrm{E}\left[\|\tilde{\mathbf{w}}(i-1)\|_{\mathbf{U}^T(i)\mathbf{L}^T(i)\boldsymbol{\Sigma}\mathbf{L}(i)\mathbf{U}(i)}^2\right] + \mathrm{E}\left[\mathbf{v}^T(i)\mathbf{L}^T(i)\boldsymbol{\Sigma}\mathbf{L}(i)\mathbf{v}(i)\right] \\
&= \mathrm{E}\left[\|\tilde{\mathbf{w}}(i-1)\|_{\boldsymbol{\Sigma}'}^2\right] + \mathrm{E}\left[\mathbf{v}^T(i)\mathbf{L}^T(i)\boldsymbol{\Sigma}\mathbf{L}(i)\mathbf{v}(i)\right],
\end{aligned} \tag{39}$$

where

$$\mathbf{L}(i) = \mathbf{G}\mathcal{D}(i)\mathbf{U}^T(i) \tag{40}$$

$$\boldsymbol{\Sigma}' = \mathbf{G}^T\boldsymbol{\Sigma}\mathbf{G} - \mathbf{G}^T\boldsymbol{\Sigma}\mathbf{L}(i)\mathbf{U}(i) - \mathbf{U}^T(i)\mathbf{L}^T(i)\boldsymbol{\Sigma}\mathbf{G} + \mathbf{U}^T(i)\mathbf{L}^T(i)\boldsymbol{\Sigma}\mathbf{L}(i)\mathbf{U}(i). \tag{41}$$

Using the data independence assumption [18] and applying the expectation operator directly on (41), we have

$$\begin{aligned}
\boldsymbol{\Sigma}' &= \mathbf{G}^T\boldsymbol{\Sigma}\mathbf{G} - \mathbf{G}^T\boldsymbol{\Sigma}\,\mathrm{E}\left[\mathbf{L}(i)\mathbf{U}(i)\right] - \mathrm{E}[\mathbf{U}^T(i)\mathbf{L}^T(i)]\boldsymbol{\Sigma}\mathbf{G} + \mathrm{E}\left[\mathbf{U}^T(i)\mathbf{L}^T(i)\boldsymbol{\Sigma}\mathbf{L}(i)\mathbf{U}(i)\right] \\
&= \mathbf{G}^T\boldsymbol{\Sigma}\mathbf{G} - \mathbf{G}^T\boldsymbol{\Sigma}\mathbf{G}\mathrm{E}[\mathcal{D}(i)]\mathrm{E}[\mathbf{U}^T(i)\mathbf{U}(i)] - \mathrm{E}[\mathbf{U}^T(i)\mathbf{U}(i)]\mathrm{E}[\mathcal{D}(i)]\mathbf{G}^T\boldsymbol{\Sigma}\mathbf{G} \\
&\quad + \mathrm{E}[\mathbf{U}^T(i)\mathbf{L}^T(i)\boldsymbol{\Sigma}\mathbf{L}(i)\mathbf{U}(i)],
\end{aligned} \tag{42}$$

where $\mathrm{E}[\mathcal{D}(i)] = \mathbf{D}\left(\mathbf{I}_{MN} + \gamma\mathrm{E}[\mathbf{B}(i)]\right)$.

4.2.1. Gaussian data. The evaluation of the expectations in (42) is very complex for non-Gaussian data. Therefore, it is assumed here that the data is Gaussian to evaluate (42). For Gaussian data, the autocorrelation matrix can be decomposed as $\mathbf{R}_{\mathbf{U}} = \mathbf{T}\boldsymbol{\Lambda}\mathbf{T}^T$, where $\boldsymbol{\Lambda}$ 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:

$$\begin{aligned}
&\bar{\mathbf{w}}(i) = \mathbf{T}^T\tilde{\mathbf{w}}(i), \quad \bar{\mathbf{U}}(i) = \mathbf{U}(i)\mathbf{T}, \qquad \bar{\mathbf{G}} = \mathbf{T}^T\mathbf{G}\mathbf{T}, \\
&\bar{\boldsymbol{\Sigma}} = \mathbf{T}^T\boldsymbol{\Sigma}\mathbf{T}, \qquad \bar{\boldsymbol{\Sigma}}' = \mathbf{T}^T\boldsymbol{\Sigma}'\mathbf{T}, \quad \bar{\mathcal{D}}(i) = \mathbf{T}^T\mathcal{D}(i)\mathbf{T} = \mathcal{D}(i).
\end{aligned}$$

Now using these relations, we rewrite (39) and (42) as

$$\mathrm{E}\left[\|\bar{\mathbf{w}}(i)\|_{\bar{\boldsymbol{\Sigma}}}^2\right] = \mathrm{E}\left[\|\bar{\mathbf{w}}(i-1)\|_{\bar{\boldsymbol{\Sigma}}'}^2\right] + \mathrm{E}\left[\mathbf{v}^T(i)\bar{\mathbf{L}}^T(i)\bar{\boldsymbol{\Sigma}}\bar{\mathbf{L}}(i)\mathbf{v}(i)\right] \tag{43}$$

and

$$\begin{aligned}
\bar{\boldsymbol{\Sigma}}' &= \bar{\mathbf{G}}^T\bar{\boldsymbol{\Sigma}}\bar{\mathbf{G}} - \bar{\mathbf{G}}^T\bar{\boldsymbol{\Sigma}}\bar{\mathbf{G}}\mathrm{E}[\mathcal{D}(i)]\mathrm{E}\left[\bar{\mathbf{U}}^T(i)\bar{\mathbf{U}}(i)\right] - \mathrm{E}\left[\bar{\mathbf{U}}^T(i)\bar{\mathbf{U}}(i)\right]\mathrm{E}[\mathcal{D}(i)]\bar{\mathbf{G}}^T\bar{\boldsymbol{\Sigma}}\bar{\mathbf{G}} \\
&\quad + \mathrm{E}\left[\bar{\mathbf{U}}^T(i)\bar{\mathbf{L}}^T(i)\bar{\boldsymbol{\Sigma}}\bar{\mathbf{L}}(i)\bar{\mathbf{U}}(i)\right],
\end{aligned} \tag{44}$$

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

DOI: 10.1002/acs

It can be seen that $\mathrm{E}\left[\bar{\mathbf{U}}^T(i)\bar{\mathbf{U}}(i)\right] = \mathbf{\Lambda}$. Also, using the bvec operator [19], we have $\bar{\sigma} = \mathrm{bvec}\left\{\bar{\mathbf{\Sigma}}\right\}$. Now, let $\mathbf{R}_{\mathbf{v}} = \mathbf{\Lambda}_{\mathbf{v}} \odot \mathbf{I}_M$ denote the noise variance matrix for the entire network, where $\odot$ denotes the block Kronecker product [19]. Hence, the second term of the right-hand side of (43) is

$$\mathrm{E}\left[\mathbf{v}^T(i)\bar{\mathbf{L}}^T(i)\bar{\mathbf{\Sigma}}\bar{\mathbf{L}}(i)\mathbf{v}(i)\right] = \mathbf{b}^T(i)\bar{\sigma}, \tag{45}$$

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

The fourth-order moment $\mathrm{E}\left[\bar{\mathbf{U}}^T(i)\bar{\mathbf{L}}^T(i)\bar{\mathbf{\Sigma}}\bar{\mathbf{L}}(i)\bar{\mathbf{U}}(i)\right]$ remains to be evaluated. Because the Lagrange multiplier is independent of the regressor data and using the $\odot$ operator, we obtain

$$\mathrm{bvec}\left\{\mathrm{E}\left[\bar{\mathbf{U}}^T(i)\bar{\mathbf{L}}^T(i)\bar{\mathbf{\Sigma}}\bar{\mathbf{L}}(i)\bar{\mathbf{U}}(i)\right]\right\} = \left(\mathrm{E}\left[\mathcal{D}(i)\odot\mathcal{D}(i)\right]\right)\mathbf{A}\left(\mathbf{G}^T\odot\mathbf{G}^T\right)\bar{\sigma}, \tag{46}$$

where we have from [4]

$$\mathbf{A} = \mathrm{diag}\left\{\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_N\right\}, \tag{47}$$

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

$$\mathbf{A}_k = \mathrm{diag}\left\{\mathbf{\Lambda}_1\otimes\mathbf{\Lambda}_k, \ldots, \lambda_k\lambda_k^T + 2\mathbf{\Lambda}_k\otimes\mathbf{\Lambda}_k, \ldots, \mathbf{\Lambda}_N\otimes\mathbf{\Lambda}_k\right\}. \tag{48}$$

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

$$\begin{aligned}
\left(\mathrm{E}\left[\mathcal{D}(i)\odot\mathcal{D}(i)\right]\right)_{kk} &= \mathrm{E}\left[\mathrm{diag}\left\{\mu_k\left(1+\gamma\beta_k(i)\right)\mathbf{I}_M\otimes\mu_1\left(1+\gamma\beta_1(i)\right)\mathbf{I}_M,\right.\right.\\
&\quad \ldots,\mu_k\left(1+\gamma\beta_k(i)\right)\mathbf{I}_M\otimes\mu_k\left(1+\gamma\beta_k(i)\right)\mathbf{I}_M,\\
&\quad \left.\left.\ldots,\mu_k\left(1+\gamma\beta_k(i)\right)\mathbf{I}_M\otimes\mu_N\left(1+\gamma\beta_N(i)\right)\mathbf{I}_M\right\}\right]\\
&= \mathrm{E}\left[\mathrm{diag}\left\{\mu_k\mu_1\left(1+\gamma\beta_k(i)\right)\left(1+\gamma\beta_1(i)\right)\mathbf{I}_{M^2},\right.\right.\\
&\quad \ldots,\mu_k^2\left(1+\gamma\beta_k(i)\right)^2\mathbf{I}_{M^2},\\
&\quad \left.\left.\ldots,\mu_k\mu_N\left(1+\gamma\beta_k(i)\right)\left(1+\gamma\beta_N(i)\right)\mathbf{I}_{M^2}\right\}\right]\\
&= \mathrm{diag}\left\{\mu_k\mu_1\left(1+\gamma\mathrm{E}\left[\beta_k(i)\right]\right)\left(1+\gamma\mathrm{E}\left[\beta_1(i)\right]\right)\mathbf{I}_{M^2},\right.\\
&\quad \ldots,\mu_k^2\mathrm{E}\left[\left(1+\gamma\beta_k(i)\right)^2\right]\mathbf{I}_{M^2},\\
&\quad \left.\ldots,\mu_k\mu_N\left(1+\gamma\mathrm{E}\left[\beta_k(i)\right]\right)\left(1+\gamma\mathrm{E}\left[\beta_N(i)\right]\right)\mathbf{I}_{M^2}\right\}.
\end{aligned} \tag{49}$$

Now applying the bvec operator on the weighting matrix $\bar{\mathbf{\Sigma}}'$, we obtain

$$\begin{aligned}
\mathrm{bvec}\left\{\bar{\mathbf{\Sigma}}'\right\} &= \bar{\sigma}'\\
&= \left[\mathbf{I}_{M^2N^2} - \left(\mathbf{I}_{MN}\odot\mathbf{\Lambda}\mathrm{E}\left[\mathcal{D}(i)\right]\right) - \left(\mathbf{\Lambda}\mathrm{E}\left[\mathcal{D}(i)\right]\odot\mathbf{I}_{MN}\right)\right.\\
&\quad \left.+\left(\mathrm{E}\left[\mathcal{D}(i)\odot\mathcal{D}(i)\right]\right)\mathbf{A}\right]\left(\mathbf{G}^T\odot\mathbf{G}^T\right)\bar{\mathbf{f}}\bar{\mathbf{f}}\\
&= \mathbf{F}(i)\bar{\sigma},
\end{aligned} \tag{50}$$

where

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

Then, (43) will look like the following:

$$\mathrm{E}\left[\|\bar{\mathbf{w}}(i)\|_{\bar{\sigma}}^2\right] = \mathrm{E}\left[\|\bar{\mathbf{w}}(i-1)\|_{\mathbf{F}(i)\bar{\sigma}}^2\right] + \mathbf{b}^T(i)\bar{\sigma}, \tag{52}$$

and hence, the transient behavior of the network is characterized by (52).

DOI: 10.1002/acs

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

$$\begin{aligned}
\mathcal{E}(i-1) &= \operatorname{diag}\left\{\left(\mathrm{E}\left[\|\bar{\mathbf{w}}(i-1)\|_{\lambda}^{2}\right] + \tilde{\sigma}_{v,1}^{2}\right)\mathbf{I}_M, \ldots, \left(\mathrm{E}\left[\|\bar{\mathbf{w}}(i-1)\|_{\lambda}^{2}\right] + \tilde{\sigma}_{v,N}^{2}\right)\mathbf{I}_M\right\} \\
&= \operatorname{diag}\left\{\left(\mathrm{E}\left[\|\bar{\mathbf{w}}(i-1)\|_{\lambda}^{2}\right] + (1-a)\,\sigma_{v,1}^{2}\right)\mathbf{I}_M, \right. \\
&\quad \left. \ldots, \left(\mathrm{E}\left[\|\bar{\mathbf{w}}(i-1)\|_{\lambda}^{2}\right] + (1-a)\sigma_{v,N}^{2}\right)\mathbf{I}_M\right\}, \\
\mathrm{E}[\mathbf{B}(i)] &= (1-\alpha)\,\mathrm{E}[\mathbf{B}(i-1)] + \frac{\alpha}{2}\mathcal{E}(i-1), \\
\mathrm{E}\left[\mathbf{B}^2(i)\right] &= (1-\alpha)^2\,\mathrm{E}\left[\mathbf{B}^2(i-1)\right] + \alpha\,(1-\alpha)\,\mathrm{E}[\mathbf{B}(i-1)]\,\mathcal{E}(i-1) + \frac{\alpha^2}{4}\left(\mathcal{E}(i-1)\right)^2, \\
\mathrm{E}[\mathcal{D}(i)] &= \mathbf{D}\left(\mathbf{I}_{MN} + \gamma \mathrm{E}[\mathbf{B}(i)]\right), \\
\mathrm{E}\left[\mathcal{D}^2(i)\right] &= \mathbf{D}^2\left(\mathbf{I}_{MN} + 2\gamma \mathrm{E}[\mathbf{B}(i)] + \gamma^2 \mathrm{E}\left[\mathbf{B}^2(i)\right]\right), \\
\mathbf{F}(i) &= [\mathbf{I}_{M^2N^2} - (\mathbf{I}_{MN} \odot \Lambda \mathrm{E}[\mathcal{D}(i)]) - (\Lambda \mathrm{E}[\mathcal{D}(i)] \odot \mathbf{I}_{MN}) \\
&\quad + (\mathrm{E}[\mathcal{D}(i) \odot \mathcal{D}(i)])\,\mathbf{A}]\left(\mathbf{G}^T \odot \mathbf{G}^T\right), \\
\mathbf{b}(i) &= bvec\left\{\mathbf{R}_{\mathbf{v}} \mathrm{E}\left[\mathcal{D}^2(i)\right]\Lambda\right\},
\end{aligned}$$

where $a = \hat{\sigma}_{v,k}^2 / \sigma_{v,k}^2$ is the ratio between the estimated and actual noise power at node k. For a perfect estimate, this would result in $a = 1$; for a mismatch, $0 < a < 1$; and for the case of ZNCDLMS, $a = 0$. Incorporating the aforementioned relations in (52) gives

$$\begin{aligned}
\mathrm{E}\left[\|\bar{\mathbf{w}}(i)\|_{\bar{\mathbf{f}}\bar{\mathbf{f}}}^{2}\right] &= \mathrm{E}\left[\|\bar{\mathbf{w}}(i-1)\|_{\mathbf{F}(i)\bar{\sigma}}^{2}\right] + \mathbf{b}^T(i)\bar{\sigma} \\
&= \left\|\bar{\mathbf{w}}^{(o)}\right\|^2_{\left(\prod_{m=0}^{i}\mathbf{F}(m)\right)\bar{\sigma}} + \left[\sum_{m=0}^{i-1}\mathbf{b}^T(m)\left(\prod_{n=m+1}^{i}\mathbf{F}(n)\right) + \mathbf{b}^T(i)\mathbf{I}_{MN}\right]\bar{\sigma}. \qquad (53)
\end{aligned}$$

Now, subtracting the results of iteration i from those of iteration $i+1$ and simplifying, we obtain

$$\begin{aligned}
\mathrm{E}\left[\|\bar{\mathbf{w}}(i)\|_{\bar{\sigma}}^{2}\right] &= \mathrm{E}\left[\|\bar{\mathbf{w}}(i-1)\|_{\bar{\sigma}}^{2}\right] + \left\|\bar{\mathbf{w}}^{(o)}\right\|^2_{\mathbf{F}'(i)(\mathbf{F}(i)-\mathbf{I}_{MN})\bar{\sigma}} \\
&\quad + \left[\mathbf{F}''(i)\left(\mathbf{F}(i) - \mathbf{I}_{MN}\right) + \mathbf{b}^T(i)\,\mathbf{I}_{MN}\right]\bar{\sigma}, \qquad (54)
\end{aligned}$$

where

$$\mathbf{F}'(i) = \prod_{m=0}^{i-1}\mathbf{F}(m), \qquad (55)$$

$$\mathbf{F}''(i) = \sum_{m=0}^{i-2}\mathbf{b}^T(m)\left(\prod_{n=m+1}^{i-1}\mathbf{F}(n)\right) + \mathbf{b}^T(i)\,\mathbf{I}_{MN}, \qquad (56)$$

which can be defined iteratively as

$$\mathbf{F}'(i+1) = \mathbf{F}'(i)\,\mathbf{F}(i), \qquad (57)$$

$$\mathbf{F}''(i+1) = \mathbf{F}''(i)\,\mathbf{F}(i) + \mathbf{b}^T(i)\,\mathbf{I}_{MN}. \qquad (58)$$

DOI: 10.1002/acs

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

$$\eta(i) = \eta(i-1) + \left\|\bar{\mathbf{w}}^{(o)}\right\|^2_{\mathbf{F}'(i)(\mathbf{F}(i)-\mathbf{I}_{MN})\mathbf{q}_\eta} + \left[\mathbf{F}''(i)\,(\mathbf{F}(i)-\mathbf{I}_{MN}) + \mathbf{b}^T(i)\,\mathbf{I}_{MN}\right]\mathbf{q}_\eta. \tag{59}$$

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

$$\zeta(i) = \zeta(i-1) + \left\|\bar{\mathbf{w}}^{(o)}\right\|^2_{\mathbf{F}'(i)(\mathbf{F}(i)-\mathbf{I}_{MN})\lambda_\zeta} + \left[\mathbf{F}''(i)\,(\mathbf{F}(i)-\mathbf{I}_{MN}) + \mathbf{b}^T(i)\,\mathbf{I}_{MN}\right]\lambda_\zeta. \tag{60}$$

4.3. Steady-state analysis

From (27), it is seen that the Lagrangian multiplier update for each node is independent of data from other nodes. Even though the connectivity matrix, $\mathbf{G}$, does not permit the weighting matrix, $\mathbf{F}(i)$, to be evaluated separately for each node, this is not the case for the step size of any node. Therefore, taking the approach of [12], we first find the misadjustment, given by

$$\mathcal{M}_k = \frac{\mu_k Tr\{\mathbf{R}_{\mathbf{u},k}\}}{2}\left(1 + \frac{\gamma\sigma^2_{v,k}(1-a)}{2} + \frac{\gamma^2\alpha\sigma^2_{v,k}}{2(2-\alpha)\left(1+\gamma\sigma^2_{v,k}(1-a)/2\right)}\right), \tag{61}$$

which leads to the steady-state values for the Lagrange multiplier update and its square for each node

$$\begin{aligned}\beta_{k,ss} &= \frac{1}{2}\left(\mathrm{EMSE}_{ss} + (1-a)\sigma^2_{v,k}\right)\\ &= \frac{1}{2}\left(\mathcal{M}_k\sigma^2_{v,k} + (1-a)\,\sigma^2_{v,k}\right)\\ &= \frac{\sigma^2_{v,k}}{2}\left(\mathcal{M}_k + 1 - a\right),\end{aligned} \tag{62}$$

$$\beta^2_{k,ss} = \frac{\alpha}{(1-\alpha)}\beta_{k,ss}\sigma^2_{v,k}\left(\mathcal{M}_k + 1 - a\right) + \frac{\sigma^4_{v,k}}{4(1-\alpha)^2}\left(\mathcal{M}_k + 1 - a\right)^2. \tag{63}$$

Incorporating these relations in (51) to obtain the steady-state weighting matrix as

$$\begin{aligned}\mathbf{F}_{ss} = [&\mathbf{I}_{M^2N^2} - (\mathbf{I}_{MN}\odot\mathit{\Lambda}\mathrm{E}[\mathcal{D}_{ss}]) - (\mathit{\Lambda}\mathrm{E}[\mathcal{D}_{ss}]\odot\mathbf{I}_{MN})\\ &+ (\mathrm{E}[\mathcal{D}_{ss}\odot\mathcal{D}_{ss}])\,\mathbf{A}]\left(\mathbf{G}^T\odot\mathbf{G}^T\right),\end{aligned} \tag{64}$$

where $\mathcal{D}_{ss} = \mathrm{diag}\{\mu_k(1+\gamma\beta_{k,ss})\mathbf{I}_M\}$.

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

$$\mathrm{E}\left[\|\bar{\mathbf{w}}_{ss}\|^2_{\bar{\sigma}}\right] = \mathrm{E}\left[\|\bar{\mathbf{w}}_{ss}\|^2_{\mathbf{F}_{ss}\bar{\sigma}}\right] + \mathbf{b}^T_{ss}\bar{\sigma}, \tag{65}$$

where $\mathbf{b}_{ss} = \mathbf{R}_{\mathbf{v}}\tilde{\mathcal{D}^2_{ss}}$ and $\mathcal{D}^2_{ss} = \mathrm{diag}\left\{\mu^2_k(1+\gamma\beta_{k,ss})^2\mathbf{I}_M\right\}$. Now solving (65), we obtain

$$\mathrm{E}\left[\|\bar{\mathbf{w}}_{ss}\|^2_{\bar{\sigma}}\right] = \mathbf{b}^T_{ss}\left[\mathbf{I}_{M^2N^2} - \mathbf{F}_{ss}\right]^{-1}\bar{\sigma}. \tag{66}$$

This equation gives the steady-state performance measure for the entire network. To solve for steady-state values of MSD and EMSE, we take $\bar{\sigma} = \mathbf{q}_\eta$ and $\bar{\sigma} = \lambda_\zeta$, respectively, as in (59) and (60). This gives us the steady-state values for MSD and EMSE as follows:

$$\eta_{ss} = \mathbf{b}^T_{ss}\left[\mathbf{I}_{M^2N^2} - \mathbf{F}_{ss}\right]^{-1}\mathbf{q}_\eta, \tag{67}$$

$$\zeta_{ss} = \mathbf{b}^T_{ss}\left[\mathbf{I}_{M^2N^2} - \mathbf{F}_{ss}\right]^{-1}\lambda_\zeta. \tag{68}$$

DOI: 10.1002/acs

4.4. Complexity analysis

To truly appreciate the improvement gained from the proposed algorithm, we need to see the cost of this improvement. Because (23) is the only extra equation comparing with the DLMS algorithm, it is sufficient to see the number of computations in this equation to know the extra cost of the proposed algorithm. There are two multiplications and two additions in (23). The other extra computation required is the estimation of noise variance, but that operation is performed only once. Comparing with the improvement in performance gained through the proposed algorithm, the cost is reasonably low. Hence, the proposed algorithm proves to be a very cost-effective improvement over the DLMS algorithm.

5. NUMERICAL RESULTS

In this section, several simulation scenarios are considered and discussed to assess the performance of the proposed NCDLMS algorithm. Results have been conducted for different average signal-to-noise ratio (SNR) values. The performance measure is the MSD.

First, the proposed algorithm is compared with existing algorithms, which are the no cooperation case, the distributed LMS [8], the DLMS [4], the DLMS with adaptive combiners [6], the VSSDLMS algorithm [9], and the diffusion recursive least squares (DRLS) [20]. The length of the unknown parameter vector is taken as $M = 4$. The size of the network is $N = 20$. The input regressor vector is assumed to be white Gaussian with autocorrelation matrix having the same variance for all nodes. Results are shown for two different values of SNR and communication range 0.3. The convergence speed is kept similar for all algorithms so that the steady-state MSD can be compared. Figure 2 reports the performance behavior of the different algorithms at an SNR of 10 dB. As can be seen from this figure, the performance of the proposed NCDLMS algorithm comes after that of the DRLS algorithm. The performance of the NCDLMS algorithm improves better for an SNR of 20 dB as depicted in Figure 3. In both of these figures, when compared with other algorithms of similar complexity, the improvement in performance of the NCDLMS algorithm is very significant. Similar performance for the steady-state behavior is obtained by the proposed NCDLMS algorithm at SNR of 10 and 20 dB as shown, respectively, in Figures 4 and 5. The DRLS algorithm performs better as expected, but the proposed algorithm is clearly better than the remaining algorithms, both in convergence speed and steady-state error. Also, diffusion results in effecting the step size variation of neighboring nodes, and as a result, the steady-state MSD for all nodes is nearly the same for all cases. This is in contrast with other algorithms for which the steady-state MSD is effected by the SNR at each node, even when the SNR is high.

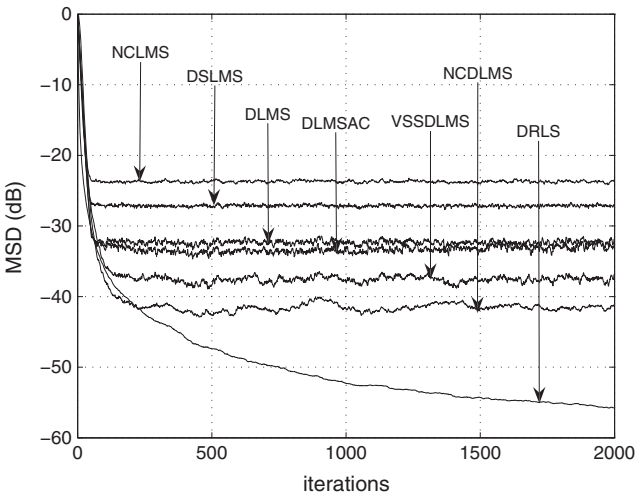

Figure 2. MSD for 20 nodes at SNR = 10 dB.

DOI: 10.1002/acs

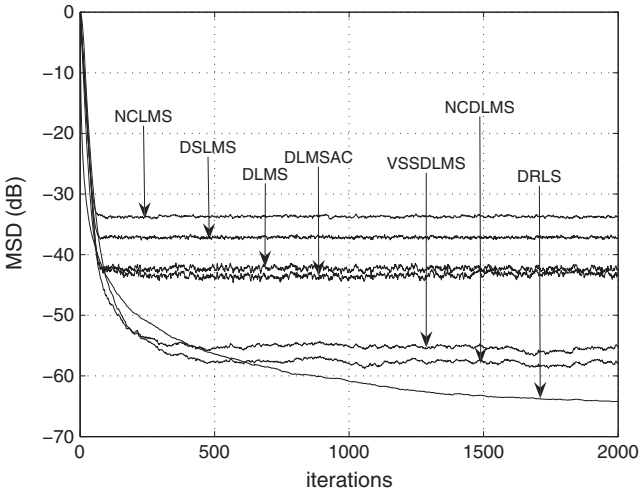

Figure 3. MSD for 20 nodes at SNR = 20 dB.

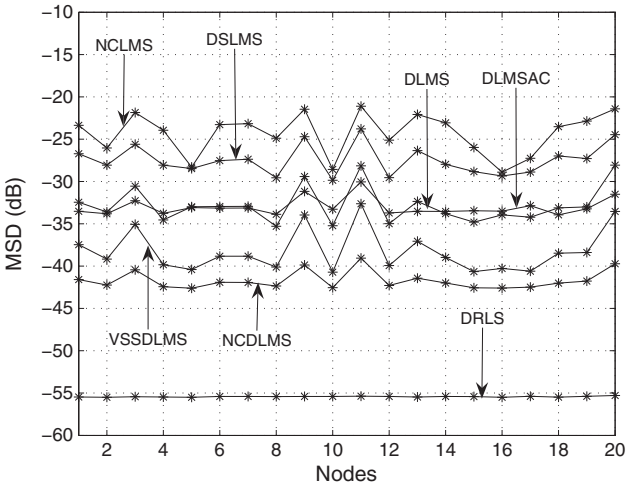

Figure 4. MSD at steady-state for 20 nodes at SNR = 10 dB.

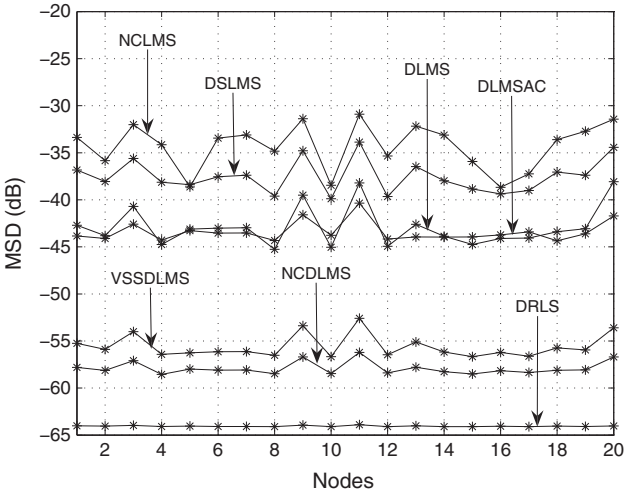

Figure 5. MSD at steady-state for 20 nodes at SNR = 20 dB.

DOI: 10.1002/acs

To test further the convergence behavior of the proposed algorithm, Figure 6 compares the faster convergence obtained by the proposed algorithm with that of the DLMS algorithm for the same steady-state error. As can be seen from this figure, the proposed algorithm performs much better than the DLMS algorithm. This is due to the mechanism embedded in the algorithm that makes it act as a variable step size algorithm and eventually performs better than the DLMS algorithm.

Next, the robustness of the proposed NCDLMS algorithm is shown when there is a mismatch in the noise variance estimate. The performance is compared with that of the VSSDLMS algorithm. Figure 7 shows the comparison for a SNR of 10 dB. As can be seen from the figure, the performance degrades as the mismatch increases, but the performance is still better than the VSSDLMS algorithm, the mismatch level for which can be taken to be approximately 45%. The ZNCDLMS algorithm performs slightly worse than the VSSDLMS algorithm, but its complexity is comparable with that of the VSSDLMS algorithm, and the performance is justified.

Next, the theoretical analysis of the proposed algorithm ((59) and (60)) as compared with the simulation results is reported in Figures 8 and 9 for perfect noise variance estimation and Figures 10 and 11 for a 50% mismatch in noise variance estimation. Here, the plots for (59) and (60) are compared with simulation plots for the proposed algorithm, averaged over 100 Monte

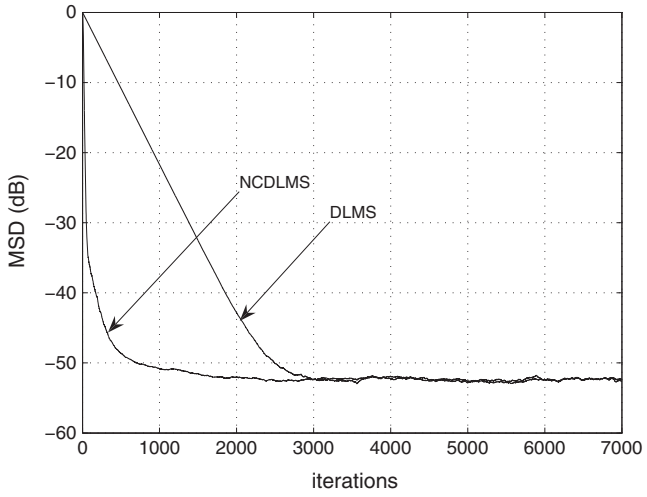

Figure 6. Convergence behavior of NCDLMS and DLMS at SNR = 20 dB.

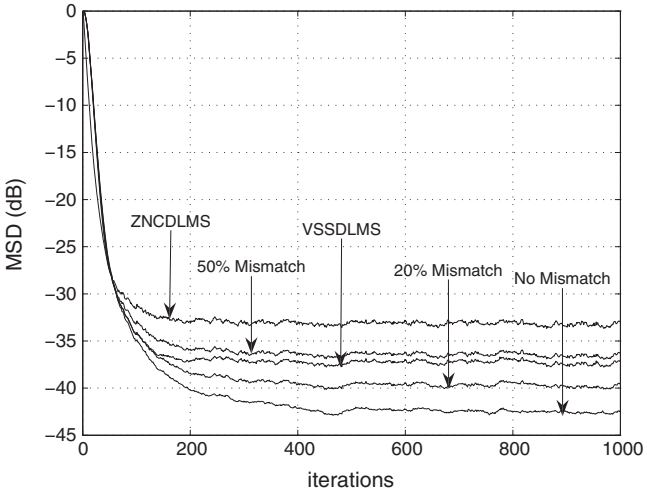

Figure 7. MSD for different mismatches at SNR = 10 dB.

DOI: 10.1002/acs

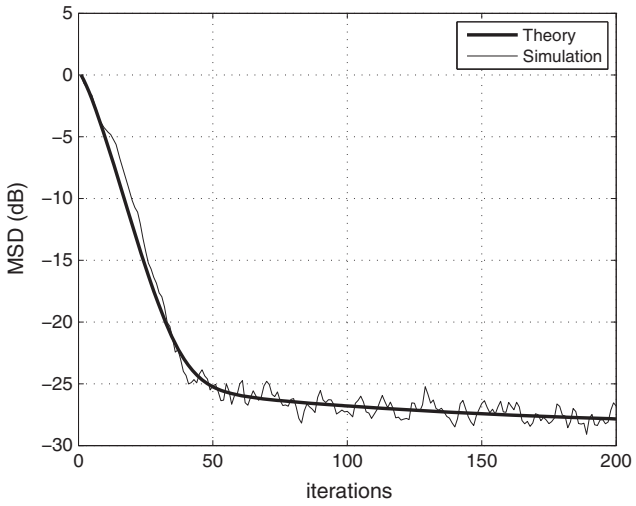

Figure 8. MSD for theory and simulation at SNR = 10 dB.

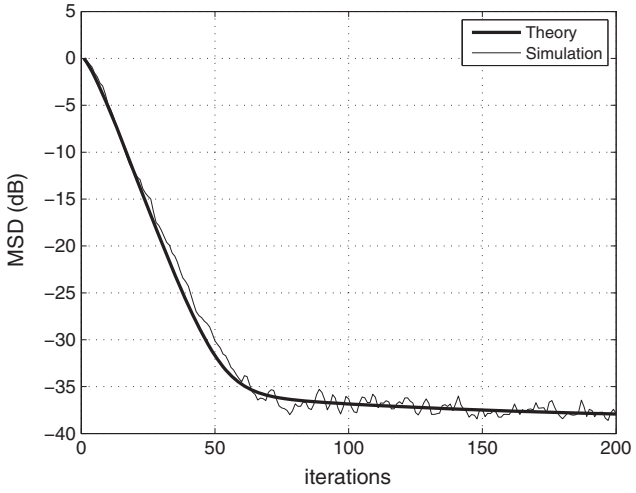

Figure 9. MSD for theory and simulation at SNR = 20 dB.

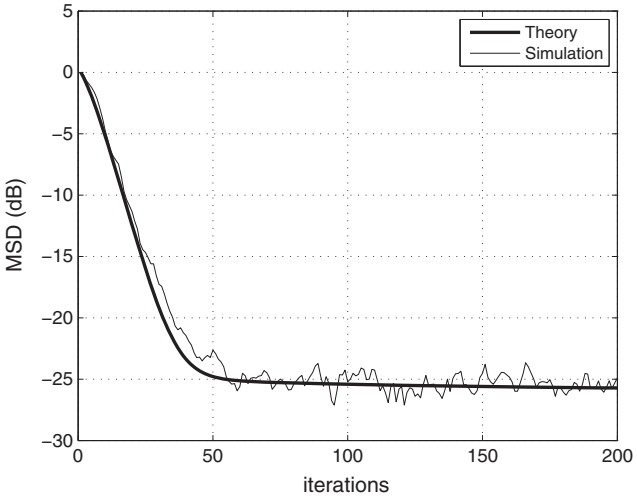

Figure 10. MSD for theory and simulation at SNR = 10 dB for 50% mismatch.

DOI: 10.1002/acs

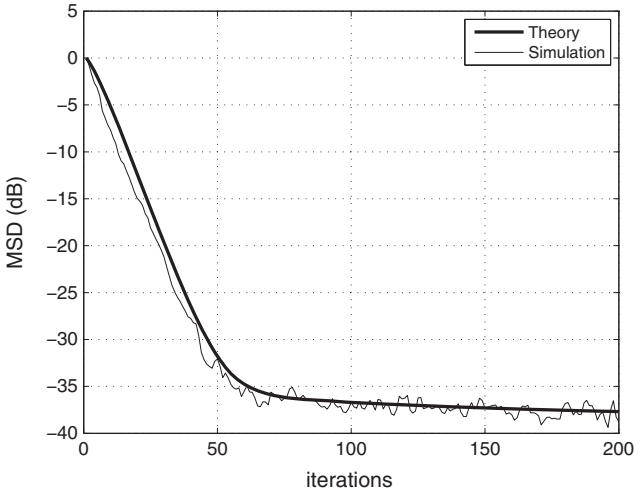

Figure 11. MSD for theory and simulation at SNR = 20 dB for 50% mismatch.

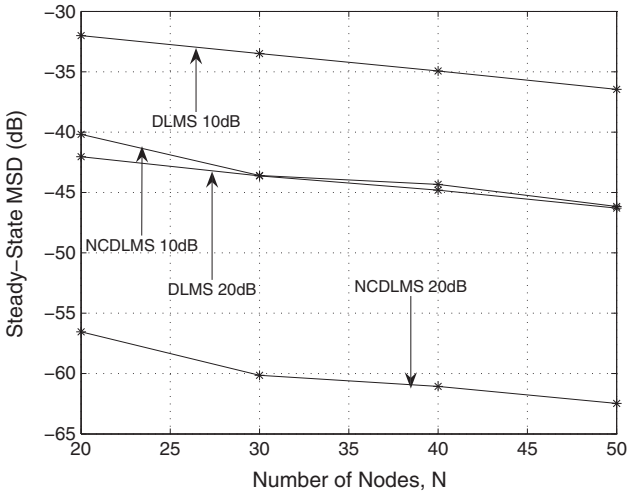

Figure 12. Steady-state MSD for varying N.

Carlo experiments. As can be seen from these figures, the simulation results are corroborating the theoretical results very well. The value for $\alpha = 0.01$, whereas $\gamma = 20$, and the results are shown for SNR of 10 and 20 dB.

The effect on the performance of the proposed algorithm when the size of the network varies is reported in Figure 12. An increase in node density improves performance. Furthermore, the performance of the proposed algorithm at SNR=10 dB is seen to be almost similar to that of the DLMS algorithm at SNR=20 dB, which shows the tremendous advantage gained through the proposed algorithm.

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 13. A network of 50 nodes is chosen so that enough nodes can be switched off 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

DOI: 10.1002/acs

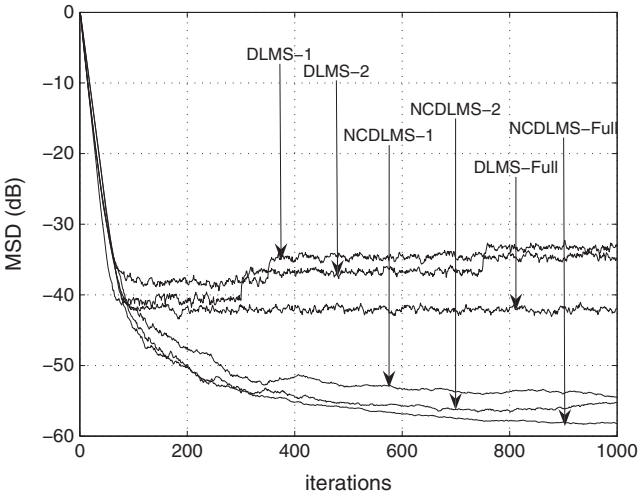

Figure 13. Robustness of algorithm at SNR = 20 dB.

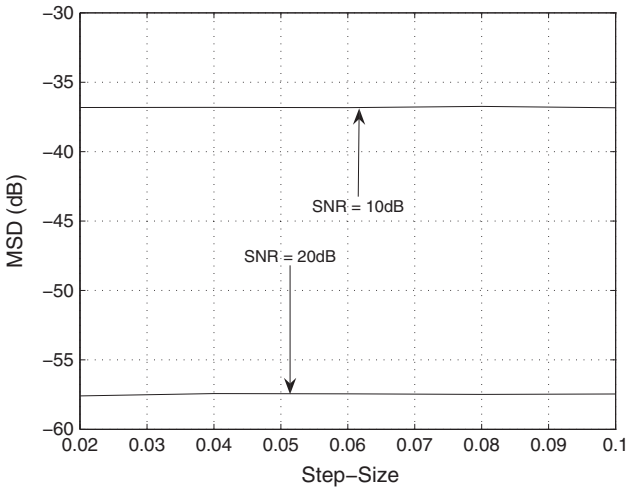

Figure 14. Steady-state MSD for different values of μ_0.

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 gets much worse when 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.

Figure 14 shows a comparison of steady-state performance for the proposed algorithm at different SNR values with varying initial step size. As can be seen, the initial value has an almost negligible effect on the steady-state performance of the algorithm, which further enhances the importance of the proposed algorithm. In comparison, the DLMS algorithm shows worse performance for a large value of the step size.

Finally, we look at the stability analysis of the algorithm. Here, the autocorrelation matrix, $\mathbf{R}_{\mathbf{u},k}$, is taken to be an identity matrix. Table I gives results for steady-state MSD for the network when the value of μ_k is varied, for $k = 3$, $\gamma_{\mathrm{NC}} = 0.1$, $\alpha_{\mathrm{NC}} = 0.01$, and SNR=20 dB. From this table, it can be seen that the simulations corroborate the theoretical finding for the steady-state MSD. Moreover, the bound in (32) holds true, that is, μ_k is chosen correctly.

DOI: 10.1002/acs

Table I. Comparison of MSD, from simulations and theory.

μ_k	**SS-MSD simulations**	**SS-MSD equation (67)**
1.9	−15.3	−15.7
1.75	−18.4	−18.6
1.5	−21.5	−21.5
1	−25.8	−26

SS-MSD, steady-state mean square deviation.

6. CONCLUSION

This work entails a detailed discussion on the newly developed NCDLMS algorithm. Here, complete derivations of the algorithm by using the ATC diffusion technique are given, where the convergence and the steady-state analyses are carried out using the assumption of Gaussian data for a closed form solution. Furthermore, the case for a mismatch in the estimation of noise variance is also considered. Extensive simulations are carried out under different scenarios to assess the performance of the proposed algorithm. It was found that the proposed algorithm performs remarkably better than the existing algorithms with similar complexity. Also, the robustness of the proposed algorithm was tested for different scenarios. The first one consists of testing it under different levels of mismatches. Here, the algorithm showed its superiority even when the ZNCDLMS algorithm is compared with the VSSDLMS algorithm of similar complexity. The second case consists of testing the performance of the proposed algorithm when random nodes become unoperational during the course of the estimation process. It is shown that in this scenario, the level of deterioration is minimal when compared with that of the DLMS algorithm. The last case considers a mismatch in the estimation of the noise variance, and here too, the simulations report great improvement in performance for the proposed algorithm. More importantly, here in this work is the corroboration of theoretical results to simulations where it was found that both agree very well. Finally, the computational complexity added to the proposed algorithm is justified by its remarkable performance.

ACKNOWLEDGEMENT

This research work is funded by King Fahd University of Petroleum and Minerals under Research Grant (FT100012).

REFERENCES

1. Estrin D, Girod L, Pottie G, Srivastava M. Instrumenting the world with wireless sensor networks. *Proceedings of the ICASSP*, Salt Lake City, UT, May 2001; 2033–2036.
2. Olfati-Saber R, Fax JA, Murray RM. Consensus and cooperation in networked multi-agent systems. *Proceesings of the IEEE* 2007; **95**:215–233.
3. Lopes CG, Sayed AH. Incremental adaptive strategies over distributed networks. *IEEE Transactions on Signal Processing* 2007; **55**:4064–4077.
4. Lopes CG, Sayed AH. Diffusion least-mean squares over adaptive networks: formulation and performance analysis. *IEEE Transactions on Signal Process* 2008; **56**(7):3122–3136.
5. Cattivelli F, Sayed AH. Diffusion LMS strategies for distributed estimation. *IEEE Transactions on Signal Process* 2010; **58**(3):1035–1048.
6. Takahashi N, Yamada I, Sayed AH. Diffusion least mean squares with adaptive combiners: formulation and performance analysis. *IEEE Transactions on Signal Process* 2010; **58**(9):4795–4810.
7. Schizas ID, Mateos G, Gannakis GB. Distributed LMS for consensus-based in-network adaptive processing. *IEEE Transactions on Signal Processing* 2009; **57**(6):2365–2381.
8. Mateos G, Schizas ID, Giannakis GB. Performance analysis of the consensus-based distributed LMS algorithm. *EURASIP Journal on Advances in Signal Processing* 2009; **2009**:19. article ID 981030.
9. Bin Saeed MO, Zerguine A, Zummo SA. Variable step size least mean square algorithms over adaptive networks. *Proceedings of ISSPA*, Kuala Lampur, Malaysia, May 2010; 381–384.

DOI: 10.1002/acs

10. Haykin S. *Adaptive Filter Theory*, (4th edn). Prentice-Hall: Englewood Cliffs, NJ, 2000.
11. Papadimitriou CH. *Computational Complexity*. Addison-Wesley, 1993.
12. Wei Y, Gelfand SB, Krogmeier JV. Noise-constrained least-mean squares algorithm. *IEEE Transactions on Signal Process* 2001; **49**(9):1961–1970.
13. Bin Saeed MO, Zerguine A, Zummo SA. Noise constrained diffusion least mean squares over adaptive networks. *Proceedings of IEEE PIMRC*, Istanbul, Turkey, September 2010; 288–292.
14. Nagumo IJ, Noda A. A learning method for system identification. *IEEE Transactions on Automatic Control* 1967; **AC-28**:282–287.
15. Moinuddin M, Zerguine A, Sheikh A. Multiple-access interference plus noise-constrained least mean square (MNCLMS) algorithm for CDMA systems. *IEEE Transactions on Circuits and Systems - Part I: Regular Papers* 2008; **55**(9):2870–2883.
16. Zerguine A, Moinuddin M, Imam SAA. A noise constrained least mean fourth (NCLMF) adaptive algorithm. *Signal Processing* 2011; **91**(1):136–149.
17. Dufflo M. *Random Iterative Models*. Springer-Verlag: Berlin, Germany, 1997.
18. Sayed AH. *Fundamentals of Adaptive Filtering*. Wiley: New York, 2003.
19. Koning R, Neudecker H, Wansbeek T. Block Kronecker products and the vecb operator. *Linear Algebra and its Applications* 1991; **149**:165–184.
20. Cattivelli FS, Lopes CG, Sayed AH. Diffusion recursive least-squares for distributed estimation over adaptive networks. *IEEE Transactions on Signal Process* 2008; **56**(5):1865–1877.

DOI: 10.1002/acs