Block Least-Mean Squares Algorithm Over Distributed Wireless Sensor Network

Hindawi Publishing Corporation
Journal of Computer Networks and Communications
Volume 2012, Article ID 601287, 13 pages
doi:10.1155/2012/601287

Research Article

Block Least Mean Squares Algorithm over Distributed Wireless Sensor Network

T. Panigrahi,[1] P. M. Pradhan,[2] G. Panda,[2] and B. Mulgrew[3]

[1] *Department of ECE, National Institute of Technology, Rourkela 769008, India*
[2] *School of Electrical Sciences, Indian Institute of Technology, Bhubaneswar 713002, India*
[3] *Institute for Digital Communication, The University of Edinburgh, Edinburgh EH899AD, UK*

Correspondence should be addressed to T. Panigrahi, panigrahit@nitrkl.ac.in

Received 31 October 2011; Accepted 1 December 2011

Academic Editor: Liansheng Tan

In a distributed parameter estimation problem, during each sampling instant, a typical sensor node communicates its estimate either by the diffusion algorithm or by the incremental algorithm. Both these conventional distributed algorithms involve significant communication overheads and, consequently, defeat the basic purpose of wireless sensor networks. In the present paper, we therefore propose two new distributed algorithms, namely, block diffusion least mean square (BDLMS) and block incremental least mean square (BILMS) by extending the concept of block adaptive filtering techniques to the distributed adaptation scenario. The performance analysis of the proposed BDLMS and BILMS algorithms has been carried out and found to have similar performances to those offered by conventional diffusion LMS and incremental LMS algorithms, respectively. The convergence analyses of the proposed algorithms obtained from the simulation study are also found to be in agreement with the theoretical analysis. The remarkable and interesting aspect of the proposed block-based algorithms is that their communication overheads per node and latencies are less than those of the conventional algorithms by a factor as high as the block size used in the algorithms.

1. Introduction

A wireless sensor network (WSN) consists of a group of sensors nodes which perform distributed sensing by coordinating themselves through wireless links. Since the nodes operate in a WSN function with limited battery power, it is important to design the networks with a minimum of communication among the nodes to estimate the required parameter vector [1, 2]. In the literature, a number of research papers have appeared which address the energy issues of sensor networks. According to the energy estimation scheme based on the 4th power loss model with Rayleigh fading [3], the transmission of 1 kb of data over a distance of 100 m, operating at 1 GHz using BPSK modulation with 10^{-6} bit-error rate, requires 3 J of energy. The same energy can be used for executing 300 M instructions in a 100 MIPS/watt general purpose processor. Therefore, it is of great importance to minimize the communication among nodes by maximizing local estimation in each sensor node.

Each node in a WSN collects noisy observations related to certain desired parameters. In the centralized solution, every node in the network transmits its data to a central fusion center (FC) for processing. This approach has the disadvantage of being nonrobust to the failure of the FC and also needs a powerful central processor. Again the problem with centralized processing is the lack of scalability and the requirement for a large communication resource [1]. If the intended application and the sensor architecture allow more local processing, then it would be more energy efficient compared to communication extensive centralized processing. Alternatively, each node in the network can function as an individual adaptive filter to estimate the parameter from the local observations and by cooperating with the neighbors. So there is a need to search for new distributed adaptive algorithms to reduce communication overhead for low-power consumption and low-latency systems for real-time operation.

The performance of distributed algorithms depends on the mode of cooperation among the nodes, for example, incremental [4, 5], diffusion [6], probabilistic diffusion [7], and diffusion with adaptive combiner [8]. To improve the robustness against the spatial variation of signal-to-noise

ratio (SNR) over the network, recently an efficient adaptive combination strategy has been proposed [8]. Also a fully distributed and adaptive implementation to make individual decisions by each node in the network is dealt with in [9].

Since in block filtering technique [10], the filter coefficients are adjusted once for each new block of data in contrast to once for each new input sample in the least mean square (LMS) algorithm, the block adaptive filter permits faster implementation while maintaining equivalent performance as that of widely used LMS adaptive filter. Therefore, the block LMS algorithms could be used at each node in order to reduce the amount of communications.

With this in mind, we present a block formulation of the existing cooperative algorithm [4, 11] based on the distributed protocols. Distinctively, in this paper, the adaptive mechanism is proposed in which the nodes of the same neighborhood communicate with each other after processing a block of data, instead of communicating the estimates to the neighbors after every sample of input data. As a result, the average bandwidth for communication among the neighboring nodes decreases by a factor equal to the block size of the algorithm. In real-time scenarios, the nodes in the sensor network follow a particular protocol for communication [12–14], where the communication time is much more than the processing time. The proposed block distributed algorithm provides an excellent balance between the message transmission delay and processing delay, by increasing the interval between two messages and by increasing the computational load on each node in the interval between two successive transmissions. The main motivation here is to propose communication-efficient block distributed LMS algorithms (both incremental and diffusion type). We analyze the performance of the proposed algorithms and compare them with existing distributed LMS algorithms.

The reminder of the paper is organized as follows. In Section 2, we present the BDLMS algorithm and its network global model. The performance analysis of BDLMS and its learning characteristics obtained from a simulation study are presented in Section 3. Performance analysis of the BILMS and its simulation results are presented in Section 4. The performance of the proposed algorithms in terms of communication cost and latency is compared with the conventional distributed adaptive algorithms in Section 5. Finally, Section 6 discusses the conclusions of the paper.

2. Block Adaptive Distributed Solution

Consider a sensor network with N number of sensor nodes randomly distributed over the region of interest. The topology of a sensor network is modeled by an undirected graph. Let $\mathcal{G}$ be an undirected graph defined by a set of nodes $\mathcal{V}$ and a set of edges $\mathcal{E}$. Nodes i and j are called neighbors if the are connected by an edgey that is, $(i, j) \in \mathcal{E}$. We also considered a loop which consists of a set of nodes $i_1, i_2, \ldots, i_N$ such that the node i_k is i_{k+1}'s neighbor, $k = 1, 2, \ldots, N$, and i_1 is i_N's neighbor. Every node in the network $i \in \mathcal{V}$ is associated with noisy output d_i to the input data vector u_i. We have assumed that the noise is independent of both input and output data; therefore, the observations are spatially and temporally independent. The neighborhood of node i is defined as the set of nodes connected to node i which is defined as $\mathcal{N}_i = j \mid (i, j) \in \mathcal{E}$ [15].

Now, the objective is to estimate an $M \times 1$ unknown vector $\mathbf{w}^\circ$ from the measurements of N nodes. In order estimate this, every node is modeled as a block adaptive linear filter where each node updates its weights using the set of errors observed in the estimated output vector, and broadcasts that to its neighbors. The estimated weight vector of the kth node at time n is denoted as $\hat{\mathbf{w}}_k(n)$. Let $u_k(n)$ be the input data of kth node at time instant n, then the input vector to the filter at time instant n is

$$\mathbf{u}_k(n) = [u_k(n), u_k(n-1), \ldots, u_k(n-M+1)]^T. \quad (1)$$

The corresponding desired output of the node for the input vector $\mathbf{u}_k(n)$ is modeled as [16, 17]

$$d_k(n) = \mathbf{u}_k^T(n)\mathbf{w}^\circ + v_k(n), \quad (2)$$

where $v_k(n)$ denotes a temporally and spatially uncorrelated white noise with variance $\sigma_{v,k}^2$.

The *block index* j is related to the time index n as

$$n = jL + i, \quad i = 0, 1, 2, \ldots, L-1, \ j = 1, 2, \ldots, \quad (3)$$

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

$$\mathbf{X}_k^j = \left[\mathbf{u}_k(jL), \mathbf{u}_k(jL+1), \ldots, \mathbf{u}_k(jL+L-1)\right]^T, \quad (4)$$

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

$$\mathbf{d}_k^j = \left[d_k(jL), d_k(jL+1), \ldots, d_k(jL+L-1)\right]^T. \quad (5)$$

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

$$\mathbf{e}_k^j = \mathbf{d}_k(j) - \mathbf{X}_k^j \hat{\mathbf{w}}_k^j, \quad (6)$$

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

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

$$\begin{aligned} \mathbf{X}_{bg}^j &= \operatorname{col}\left\{\mathbf{X}_1^j, \mathbf{X}_2^j, \ldots, \mathbf{X}_N^j\right\}, \\ \mathbf{d}_{bg}^j &= \operatorname{col}\left\{\mathbf{d}_1^j, \mathbf{d}_2^j, \ldots, \mathbf{d}_N^j\right\}. \end{aligned} \quad (7)$$

The objective is to estimate the $M \times 1$ vector $\mathbf{w}$ from the above quantities, those collected the data across N nodes. By using this global data, the block error vector for the whole network is

$$\mathbf{e}_{bg}^j = \mathbf{d}_{bg}^j - \mathbf{X}_{bg}^j \hat{\mathbf{w}}. \quad (8)$$

Now, the vector $\hat{\mathbf{w}}$ can be estimated by minimizing MSE function as

$$\min_{\hat{\mathbf{w}}} E\left\|\mathbf{d}_{bg} - \mathbf{X}_{bg}\hat{\mathbf{w}}\right\|^2. \tag{9}$$

The time index is dropped here for simple mathematical representation. Since the quantities are collected data across the network in block format; therefore, the block mean square error (BMSE) is to be minimized. The BMSE is given by [17, 18]

$$\begin{aligned}\text{BMSE} = \frac{1}{L}\Big[&E\left[\mathbf{d}_{bg}^T\mathbf{d}_{bg}\right] - E\left[\mathbf{d}_{bg}^T\mathbf{X}_{bg}\right]\hat{\mathbf{w}} - \hat{\mathbf{w}}^T E\left[\mathbf{X}_{bg}^T\mathbf{d}_{bg}\right] \\ &-\hat{\mathbf{w}}^T E\left[\mathbf{X}_{bg}^T\mathbf{X}_{bg}\right]\hat{\mathbf{w}}\Big].\end{aligned} \tag{10}$$

Let the input regression data $\mathbf{u}$ be Gaussian and defined by the correlation function $r(l) = \sigma^2\alpha^{|l|}$ in the covariance matrix, where α is the correlation index and σ^2 is the variance of the input regression data, then the relation between correlation and cross-correlation quantities among blocked and unblocked data can be denoted as [10]

$$\mathrm{R}_X^{bg} = L\mathrm{R}_U^g, \qquad \mathrm{R}_{dX}^{bg} = L\mathrm{R}_{du}^g, \qquad \mathrm{R}_d^{bg} = L\mathrm{R}_d^g, \tag{11}$$

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

Now, the block mean square error (BMSE) in (10) is reduced to

$$\begin{aligned}\text{BMSE} &= \frac{1}{L}\left[L\mathrm{R}_d^g - L\mathrm{R}_{du}^g\hat{\mathbf{w}} - L\hat{\mathbf{w}}^T\mathrm{R}_{du}^g - L\mathbf{w}^T\mathrm{R}_u^g\hat{\mathbf{w}}\right] \\ &= \mathrm{R}_d^g - \mathrm{R}_{du}^g\hat{\mathbf{w}} - \hat{\mathbf{w}}^T\mathrm{R}_{du}^g - \mathbf{w}^T\mathrm{R}_u^g\hat{\mathbf{w}} = \text{MSE}.\end{aligned} \tag{12}$$

Comparing (12) with the MSE of conventional LMS for global data [17, 19], it can be concluded that the MSE in both the cases is same. Hence, block LMS algorithm has similar properties as that of the conventional LMS algorithm. Now, (9) for blocked data can be reduced to a form similar to that of unblocked data as

$$\min_{\hat{\mathbf{w}}} E\left\|\mathbf{d}_g - \mathbf{U}_g\hat{\mathbf{w}}\right\|^2. \tag{13}$$

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

2.1. Adaptive Block Distributed Algorithms. In adaptive block LMS algorithm, each node k in the network receives the estimates from its neighboring nodes after each block of input data to adapt the local changes in the environment. Two different types of distributed LMS in WSN have been reported in literature, namely, incremental and diffusion LMS [6, 19]. These algorithms are based on conventional LMS for local learning process which in terms needs large communication resources. In order to achieve the same performance with less communication resource, the block distributed LMS is proposed here.

2.1.1. The Block Incremental LMS (BILMS) Algorithm. In an incremental mode of cooperation, information flows in a sequential manner from one node to the adjacent one in the network after processing one sample of data [4]. The communications in the incremental way of cooperation can be reduced if each node need to communicate only after processing a block of data. For any block of data j, it is assumed that node k has access to the $\hat{\mathbf{w}}_{k-1}^j$ estimates from its predecessor node, as defined by the network topology and constitution. Based on these assumptions, the proposed block incremental LMS algorithm can be stated by reducing the conventional incremental LMS algorithm ((16) in [19]) to a blocked data form as follows,

$$\begin{aligned}&\hat{\mathbf{w}}_0^j = \hat{\mathbf{w}}^{j-1}, \\ &\hat{\mathbf{w}}_k^j = \hat{\mathbf{w}}_{k-1}^j + \frac{\mu_k}{L}\sum_{q=0}^{L-1}\mathbf{u}_k(jL+q) \\ &\qquad\times\left(d_k(jL+q) - \mathbf{u}_k^T(jL+q)\hat{\mathbf{w}}_{k-1}^j\right) \\ &\qquad= \hat{\mathbf{w}}_{k-1}^j + \frac{\mu_k}{L}\mathbf{X}^{jT}\left(\mathbf{d}_k^j - \mathbf{X}_k^j\hat{\mathbf{w}}_{k-1}^j\right), \quad \text{for } k = 1, 2, \ldots N, \\ &\hat{\mathbf{w}}^j = \hat{\mathbf{w}}_N^j,\end{aligned} \tag{14}$$

where μ_k is the local step size, and L is the block size.

2.1.2. The Block Diffusion LMS (BDLMS) Algorithm. Here, each node k updated its estimate by using a simple local rule based on the average of its own estimates plus the information received from its neighbor $\mathcal{N}_k$. In this case, for every jth block of data at the kth node, the node has access to a set of estimates from its neighbors $\mathcal{N}_k$. Similar to block incremental LMS, the proposed block diffusion strategy for a set of local combiners c_{kl} and for local step size μ_k can be described as a reduced form of conventional diffusion LMS [6, 20] as

$$\begin{aligned}&\boldsymbol{\theta}_k^{j-1} = \sum_{l\in\mathcal{N}_{k,j-1}} c_{kl}\hat{\mathbf{w}}_k^{j-1}, \quad \boldsymbol{\theta}_k(-1) = 0, \\ &\hat{\mathbf{w}}_k^j = \boldsymbol{\theta}_k^{j-1} + \frac{\mu_k}{L}\sum_{q=0}^{L-1}\mathbf{u}_k(jL+q) \\ &\qquad\times\left(d_k(jL+q) - \mathbf{u}_k^T(jL+q)\boldsymbol{\theta}_k^{j-1}\right).\end{aligned} \tag{15}$$

The weight update equation can be rewritten in more compact form by using the data in block format given in (4) and (5) as

$$\widehat{\mathbf{w}}_k^j = \boldsymbol{\theta}_k^{j-1} + \frac{\mu_k}{L}\mathbf{X}_k^{j\,T}\left(\mathbf{d}_k^j - \mathbf{X}_k^j\boldsymbol{\theta}_k^{j-1}\right). \tag{16}$$

Comparing (15) with (19) in [21], it is concluded that the weight update equation is modified into block format.

3. Performance Analysis of BDLMS Algorithm

The performance of an adaptive filter is evaluated in terms of its transient and steady-state behaviors, which, respectively provide the information about how fast and how well a filter is capable to learn. Such performance analysis is usually challenging in interconnected network because each node k is influenced by local data with local statistics $\{R_{dx,k}, R_{X,k}\}$, by its neighborhood nodes through local diffusion, and by local noise with variance $\sigma_{v,k}^2$. In case of block distributed system, the analysis becomes more challenging as it has to handle data in block form. The key performance metrics used in the analysis are MSD (mean square deviation), EMSE (excess mean square error), and MSE for local and also for global networks and are defined as

$$\begin{aligned}
\eta_k^j &= E\left\|\widetilde{\mathbf{w}}_{k-1}^j\right\|^2 \quad \text{(MSD)},\\
\zeta_k^j &= E\left\|e_{a,k}^j\right\|^2 \quad \text{(EMSE)},\\
\xi_k^j &= E||e_k(j)||^2 = \zeta_k^j + \sigma_{v,k}^2 \quad \text{(MSE)},
\end{aligned} \tag{17}$$

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

$$\begin{aligned}
\widetilde{\mathbf{w}}_{k-1}^j &= \mathbf{w}^\circ - \widehat{\mathbf{w}}_{k-1}^j,\\
e_{a,k}^j &= \mathbf{u}_k^j\widetilde{\mathbf{w}}_{k-1}^j.
\end{aligned} \tag{18}$$

The algorithm described in (15) is looking like the interconnection of block adaptive filters instead of conventional LMS adaptive algorithm among all the nodes across the network. As shown in (12) that the block LMS algorithm has similar properties to those of the conventional LMS algorithm, the convergence analysis of the proposed block diffusion LMS algorithm can be carried out similar to the diffusion LMS algorithm described in [18, 21].

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

$$\widehat{\mathbf{w}}^j = \left[\widehat{\mathbf{w}}_1^j;\ldots;\widehat{\mathbf{w}}_N^j\right]. \tag{19}$$

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

$$\boldsymbol{\theta}^j = G\widehat{\mathbf{w}}^j. \tag{20}$$

Now, the input data vector at jth block is defined as

$$\mathbf{X}^j = \operatorname{diag}\left\{\mathbf{X}_1^j,\ldots,\mathbf{X}_N^j\right\}. \tag{21}$$

The desired block responses at each node k are assumed which have to obey the traditional data model used in literature [16–18], that is,

$$\mathbf{d}_k^j = \mathbf{X}_k^j\mathbf{w}^\circ + \mathbf{v}_k^j, \tag{22}$$

where $\mathbf{v}_k^j$ is the background noise vector of length L. The noise is assumed to be spatially and temporarily independent with variance $\sigma_{v,k}^2$. Using blocked desired response for single node (17), the global response for kth block can be modeled as

$$\mathbf{d}_{bg}^j = \mathbf{X}_{bg}^j\mathbf{w}_g^\circ + \mathbf{v}^j, \tag{23}$$

where $\mathbf{w}_g^\circ$ is the optimum global weight vector defined for every node and is written as $\mathbf{w}_g^\circ = [\mathbf{w}^\circ;,\ldots,;\mathbf{w}^\circ]$ and

$$\mathbf{v}^j = \left[\mathbf{v}_1^j;,\ldots,;\mathbf{v}_N^j\right](LN \times 1) \tag{24}$$

is the additive Gaussian noise for jth block index.

Using the relations defined above, the block diffusion strategy in (15) can be written in global form as

$$\widehat{\mathbf{w}}^j = \boldsymbol{\theta}^{j-1} + \frac{1}{L}\mathbf{S}\mathbf{X}^j\left(\mathbf{d}_{bg}^j - \mathbf{X}^j\boldsymbol{\theta}^{j-1}\right), \tag{25}$$

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

$$\mathbf{S} = \operatorname{diag}\{\mu_1\mathrm{I}_M, \mu_2\mathrm{I}_M,\ldots,\mu_N\mathrm{I}_M\}\ (NM \times NM). \tag{26}$$

Using (20), it can be written as

$$\widehat{\mathbf{w}}^j = G\widehat{\mathbf{w}}^{j-1} + \frac{1}{L}\mathbf{S}\mathbf{X}^j\left(\mathbf{d}_{bg}^j - \mathbf{X}^jG\widehat{\mathbf{w}}^{j-1}\right). \tag{27}$$

3.1. Mean Transient Analysis. The mean behavior of the proposed BDLMS is similar to diffusion LMS given in [18, 21]. The mean error vector signal is given as

$$E\left[\widetilde{\mathbf{w}}^j\right] = \left(\mathrm{I}_{NM} - \frac{1}{L}\mathbf{S}\mathrm{R}_X\right)GE\left[\widetilde{\mathbf{w}}^{j-1}\right], \tag{28}$$

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

$$\mathrm{R}_{X,k} = E\left[\mathbf{X}_k{}^T\mathbf{X}_k\right] = LE\left[\mathbf{U}_k^T\mathbf{U}_k\right] = L\mathrm{R}_U. \tag{29}$$

Hence, (28) can be written as

$$E\left[\widetilde{\mathbf{w}}^j\right] = (\mathrm{I}_{NM} - \mathbf{S}\mathrm{R}_U)GE\left[\widetilde{\mathbf{w}}^{j-1}\right]. \tag{30}$$

Comparing (30) with that of diffusion LMS ((35) in [21]), we can find that both block diffusion LMS and diffusion LMS yield the same characteristic equation for the convergence of mean; and it can be concluded that block diffusion protocol defined in (15) has the same stabilizing effect on the network as diffusion LMS,

3.2. Mean-Square Transient Analysis. The variance estimate is a key performance indicator in mean-square transient analysis of any adaptive system. The variance relation for block data is similar to that of conventional diffusion LMS

$$E\left\|\tilde{\mathbf{w}}^{j}\right\|_{\Sigma}^{2} = E\left\|\tilde{\mathbf{w}}^{j-1}\right\|_{\Sigma'}^{2} + \frac{1}{L^2}E\left[\mathbf{v}^{j\,T}\mathbf{X}^{j}\mathbf{S}\Sigma\mathbf{S}\mathbf{X}^{j\,T}\mathbf{v}^{j}\right], \tag{31}$$

$$\begin{aligned}\Sigma' = {} & G^T\Sigma G - \frac{1}{L}G^T\Sigma\mathbf{S}E\left[\mathbf{X}^{j\,T}\mathbf{X}^{j}\right]G \\ & - \frac{1}{L}G^T E\left[\mathbf{X}^{j\,T}\mathbf{X}^{j}\right]\mathbf{S}\Sigma G \\ & + \frac{1}{L^2}G^T E\left[\mathbf{X}^{j\,T}\mathbf{X}^{j}\right]\mathbf{S}\Sigma\mathbf{S}E\left[\mathbf{X}^{j\,T}\mathbf{X}^{j}\right]G.\end{aligned} \tag{32}$$

Using $E[\mathbf{X}^{j\,T}\mathbf{X}^{j}] = LE[\mathbf{U}^{j\,T}\mathbf{U}^{j}]$ from the definition in (32), we obtain

$$\begin{aligned}\Sigma' = {} & G^T\Sigma G - G^T\Sigma\mathbf{S}E\left[\mathbf{U}^{j\,T}\mathbf{U}^{j}\right]G - G^T E\left[\mathbf{U}(j)^{T}\mathbf{U}^{j}\right]\mathbf{S}\Sigma G \\ & + G^T E\left[\mathbf{U}^{j\,T}\mathbf{U}^{j}\right]\mathbf{S}\Sigma\mathbf{S}E\left[\mathbf{U}^{j\,T}\mathbf{U}^{j}\right]G,\end{aligned} \tag{33}$$

which is similar to (45) in [21]. Using the properties of expectation and trace [18], the second term of (31) is solved as

$$\begin{aligned}\frac{1}{L^2}E\mathbf{v}^{j\,T}\mathbf{X}^{j}\mathbf{S}\Sigma\mathbf{S}\mathbf{X}^{j\,T}\mathbf{v}^{j} &= \frac{1}{L^2}E\left[\mathrm{tr}\left[\mathbf{X}^{j}\mathbf{S}\Sigma\mathbf{S}\mathbf{X}^{j\,T}\right]\right]E\left[\mathbf{v}^{j\,T}\mathbf{v}^{j}\right] \\ &= E\left[\mathbf{n}^{j\,T}\mathbf{U}^{j}\mathbf{S}\Sigma\mathbf{S}\mathbf{U}^{j\,T}\mathbf{n}^{j}\right],\end{aligned} \tag{34}$$

where the noise variance vector $\mathbf{n}^{j}$ is not in block form, and it is assumed that the noise is stationary Gaussian. Equations (31) and (32) may therefore be written as

$$E\left\|\tilde{\mathbf{w}}^{j}\right\|_{\Sigma}^{2} = E\left\|\tilde{\mathbf{w}}^{j-1}\right\|_{\Sigma'}^{2} + \frac{1}{L^2}E\mathbf{n}^{j\,T}\mathbf{U}^{j}\mathbf{S}\Sigma\mathbf{S}\mathbf{U}^{j\,T}\mathbf{n}(j), \tag{35}$$

$$\begin{aligned}\Sigma' = {} & G^T\Sigma G - G^T\Sigma\mathbf{S}E\left(\mathbf{U}^{j\,T}\mathbf{U}^{j}\right)G - G^T E\left(\mathbf{U}^{j\,T}\mathbf{U}^{j}\right)\mathbf{S}\Sigma G \\ & + G^T E\left(\mathbf{U}^{j\,T}\mathbf{U}^{j}\right)\mathbf{S}\Sigma\mathbf{S}E\left(\mathbf{U}^{j\,T}\mathbf{U}^{j}\right)G.\end{aligned} \tag{36}$$

It may be noted that variance estimate (36) for BDLMS algorithm is exactly the same as that of DLMS [21]. In the block LMS algorithm, the local step size is chosen to be L times that of the local step size of diffusion LMS in order to have the same level of performance. As the proposed algorithm and the diffusion LMS algorithm have similar properties, the evolution of their variances is also similar. Therefore, the recursion equation of the global variances for BDLMS will be similar to (73) and (74) in [21]. Similarly, the local node performances will be similar to (89) and (91) of [21].

3.3. Learning Behavior of BDLMS Algorithm. The learning behavior of BDLMS algorithm is examined using simulations. The characteristic or variance curves are plotted for block LMS and are compared with that of DLMS. The row regressors with shift invariance input [18] are used with each regressor having data as

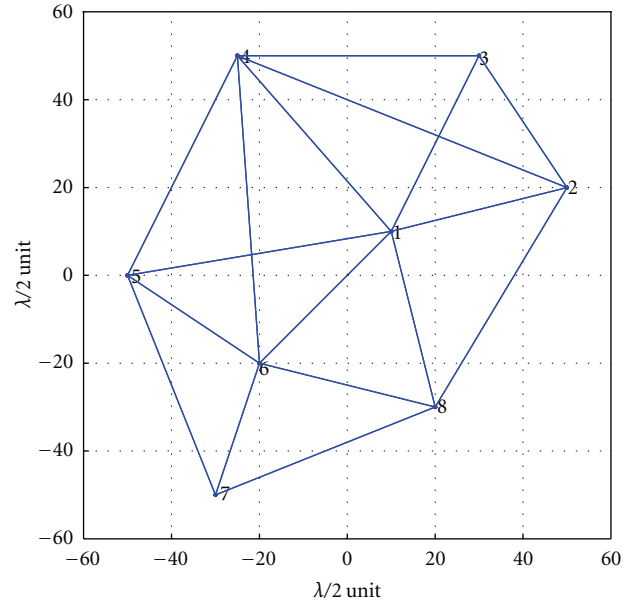

Figure 1: Network topology used for block diffusion LMS.

$$\mathbf{u}_k(i) = [u_k(i), u_k(i-1), \ldots, u_k(i-M+1)]^T. \tag{37}$$

In block LMS, the regressors for $L = 3$ and $M = 3$ are given as

$$\begin{aligned}\mathbf{X}_k(1) &= \begin{pmatrix} u_k(1) & 0 & 0 \\ u_k(2) & u_k(1) & 0 \\ u_k(3) & u_k(2) & u_k(1) \end{pmatrix}, \\ \mathbf{X}_k(2) &= \begin{pmatrix} u_k(4) & u_k(3) & u_k(2) \\ u_k(5) & u_k(4) & u_k(3) \\ u_k(6) & u_k(5) & u_k(4) \end{pmatrix}.\end{aligned} \tag{38}$$

The desired data are generated according to the model given in literature [18]. The unknown vector $\mathbf{w}^{\circ}$ is set to $[1, 1, \ldots, 1]^T/\sqrt{M}$.

The input sequence $\{\mathbf{u}_k(i)\}$ is assumed to be spatially correlated and is generated as

$$u_k(i) = a_k \cdot u_k(i-1) + b_k \cdot n_k(i), \quad i > -\infty. \tag{39}$$

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

Figure 1 shows an eight-node network topology used in the simulation study. The network settings are given in Figures 2(a) and 2(b).

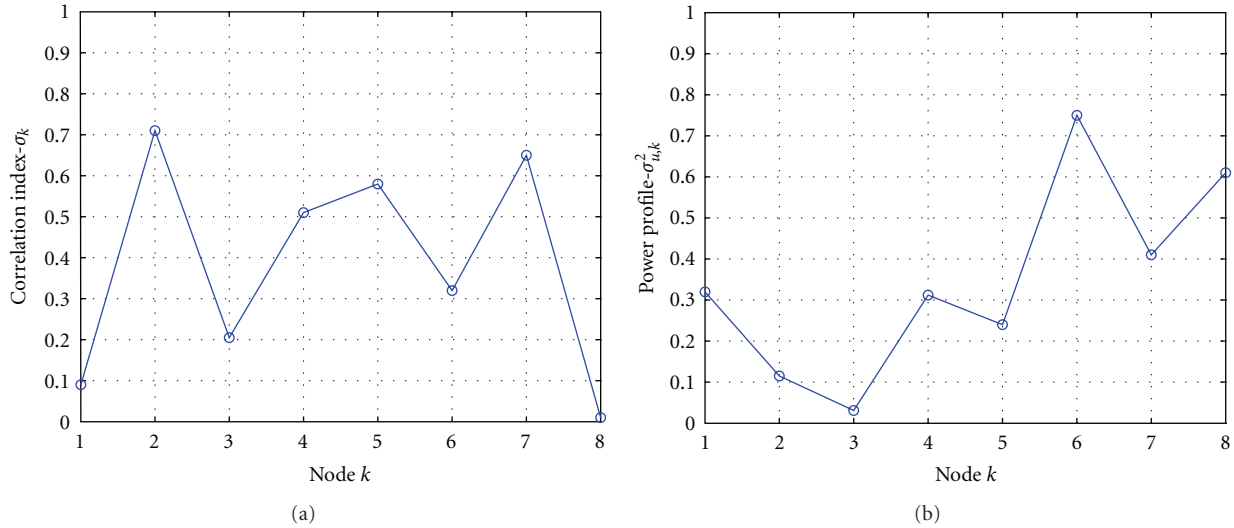

Figure 2: Network statistics used for the simulation of BDLMS. (a) Network corelation index per node. (b) Regressor power profile.

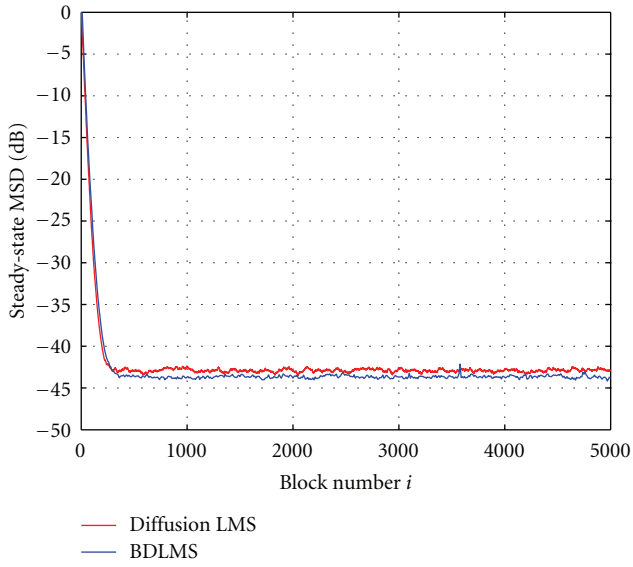

Figure 3: Global mean-square deviation (MSD) curve for diffusion and block diffusion LMS.

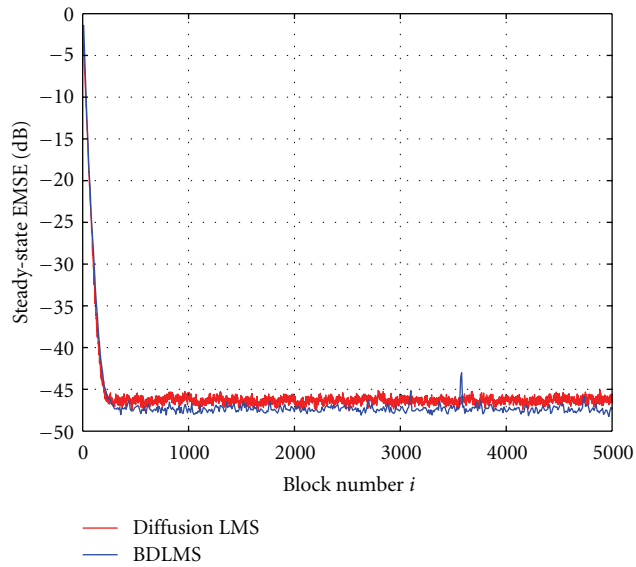

Figure 4: Global excess mean-square deviation (EMSE) curve for diffusion and block diffusion LMS.

3.4. The Simulation Conditions. The algorithm is valid for any block of length greater than one [10], while $L = M$ is the most preferable and optimum choice.

The background noise is assumed to be Gaussian white noise of variance $\sigma^2_{v,k} = 10^{-3}$, and the data used in the study is generated using $d_k(n) = \mathbf{u}_k(n)\mathbf{w}^\circ + v_k(n)$. In order to generate the performance curves, 50 independent experiments are performed and averaged. The results are obtained by averaging the last 50 samples of the corresponding learning curves. The global MSD curve is shown in Figure 3. This is obtained by averaging $E\|\widetilde{w}_k^{j-1}\|^2$ across all the nodes over 100 experiments. Similarly, the global EMSE curve obtained by averaging $E\|\mathbf{e}^{\mathbf{j}}_{a,k}\|^2$, where $\mathbf{e}^{\mathbf{j}}_{a,k} = \mathbf{x}_k^j \widetilde{w}_k^{j-1}$, across all the nodes over 100 experiments is displayed in Figure 4. The global MSE is depicted in Figure 5. It shows that in both the cases the MSE is exactly matching.

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

The global performances are the contributions of all individual nodes, and it is obtained by taking the mean performance of all the nodes. The simulation results are provided to compare with that obtained by diffusion LMS for individual node. The local MSD evolution at node 1 is given in Figure 6(a) and at node 5 is given in Figure 6(b). Similarly, the local EMSE evolution at nodes 1 and 7 is depicted in

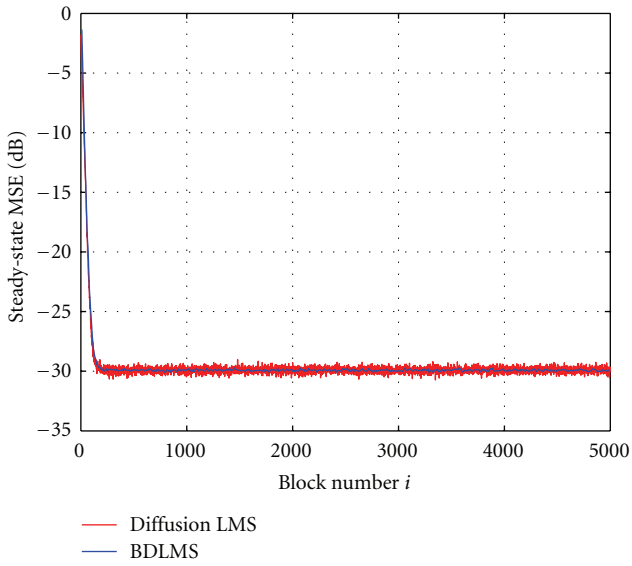

FIGURE 5: Global mean-square deviation (MSE) curve for diffusion and block diffusion LMS.

Figure 7. The convergence speed is nearly the same in both MSD and EMSE evolution, but the performance is slightly degraded in case of BDLMS. The loss of performance in case of BDLMS could be traded for the huge reduction in of communication bandwidth.

4. Performance Analysis of BILMS Algorithm

To show that the BILMS algorithm has guaranteed convergence, we may follow the steady-state performance analysis of the algorithm using the same data model as the one which is commonly used in the conventional sequential adaptive algorithms [5, 22, 23].

The weight-energy relation is derived by using the definition of weighted *a priori* and *a posteriori* error [18]

$$\left\|\widetilde{\mathbf{w}}_k^j\right\|_\Sigma^2 + \frac{\left|e_{a,k}^{j\Sigma}\right|^2}{\left\|\mathbf{X}_k^j\right\|_\Sigma^2} = \left\|\widetilde{\mathbf{w}}_{k-1}^j\right\|_\Sigma^2 + \frac{\left|e_{p,k}^{j\Sigma}\right|^2}{\left\|\mathbf{X}_k^j\right\|_\Sigma^2}. \tag{40}$$

Since (40) is similar to that of (35) in [19]. Thus, the performance of BILMS is similar to that of ILMS. The variance expression is obtained from the energy relation (40) by replacing a *posteriori error* by its equivalent expression and then averaging both the sides

$$\begin{aligned} E\left[\left\|\widetilde{\mathbf{w}}_k^j\right\|_\Sigma^2\right] &= E\left[\left\|\widetilde{\mathbf{w}}_{k-1}^j\right\|_{\Sigma'}^2\right] + \left|\frac{\mu_k}{L}\right|^2 E\left[V^{j}{}_k^T \mathbf{X}_k^j \Sigma \mathbf{X}^{j}{}_k^T V_k^j\right] \\ \Sigma' &= \Sigma - \frac{\mu_k}{L}\left(\Sigma \mathbf{X}^{j}{}_k^T \mathbf{X}_k^j + \mathbf{X}^{j}{}_k^T \mathbf{X}_k^j \Sigma\right) \\ &\quad + \left|\frac{\mu_k}{L}\right|^2 \mathbf{X}^{j}{}_k^T \mathbf{X}_k^j \Sigma \mathbf{X}^{j}{}_k^T \mathbf{X}_k^j. \end{aligned} \tag{41}$$

The variance relation in (41) is similar to the variance relation of ILMS in [19]. The performance of ILMS is studied in detail in literature. It is observed that the theoretical performance of block incremental LMS and conventional incremental LMS algorithms are similar because both have the same variance expressions. Simulation results provide the validation of this analysis.

4.1. Simulation Results of BILMS Algorithm. For the simulation study of IBLMS, we have used the regressors with shift-invariance as with the same desired data used in the case of BDLMS algorithm. The time-correlated sequences are generated at every node according to the network statistics. The same network has been chosen here for simulation study as defined for block diffusion network in Section 3.3. In incremental way of cooperation, each node receives information from its previous node, updates it by using own data, and sends the updated estimate to the next node. The ring topology used here is shown in Figure 8. We assume that the background noise to be temporarily and spatially uncorrelated additive white Gaussian noise with variance 10^{-3}. The learning curves are obtained by averaging the performance of 100 independent experiments, generated by 5,000 samples in the network. It can be observed from figures that the steady-state performances at different nodes of the network achieved by BILMS matche very closely with that of ILMS algorithm. The EMSE plots which are more sensitive to local statistics are depicted in Figures 9(a) and 9(b). A good match between BILMS and ILMS is observed from these plots. In [19], the authors have already proved the theoretical matching of steady-state nodal performance with simulation results. As the MSE roughly reflects the noise power and the plot indicates the good performance of the adaptive network, it may be inferred that the adaptive node performs well in the steady state.

The global MSD curve shown in Figure 10 is obtained by averaging $E\|\widetilde{\boldsymbol{\psi}}_k^{(j-1)}\|^2$ across all the nodes and over 50 experiments. Similarly, the global EMSE and MSE plots are displayed in Figures 11 and 12, respectively. These are obtained by averaging $E\|\mathbf{e}_{a,k}(j)\|^2$, where $\mathbf{e}_{a,k}(j) = \mathbf{x}_{k,j}\widetilde{\boldsymbol{\psi}}_k^{(j-1)}$ across all the nodes over 50 experiments.

If the weights are updated after L data points and then communicated for local diffusion, the number of communications between neighbors is reduced by L times that of ILMS where the weights are updated after processing each sample of data. Therefore, similar to BDLMS, the communication overhead in BILMS also gets reduced by L times that of ILMS algorithm.

The performance comparison between two proposed algorithms BDLMS and BILMS for the same network is shown in Figures 13–15. One can observe from Figure 13 that the MSE for BILMS algorithm is converging faster than BDLMS. Since the same noise model is used for both the algorithms, therefore after convergence, the steady-state performances are the same for both of them. But in case of MSD and EMSE performances in Figures 14 and 15, little difference is observed. It is due the different cooperation scheme used for different algorithms. However, the diffusion cooperation scheme is more adaptive to the environmental change compared to the incremental cooperation. But

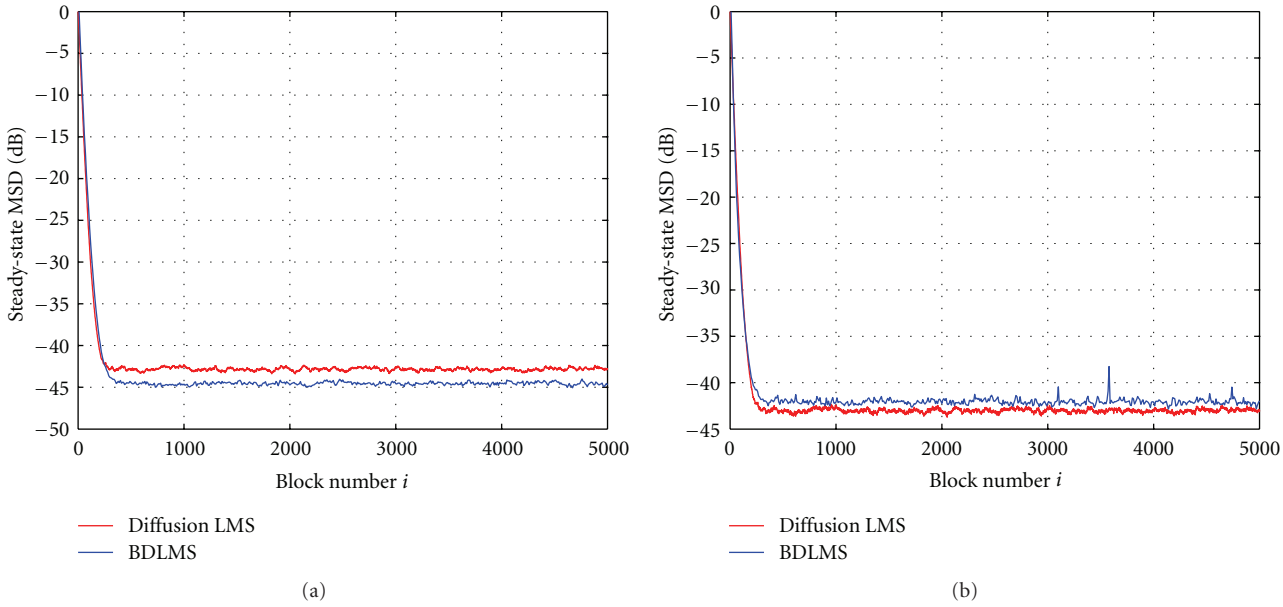

Figure 6: Local mean-square deviation (MSD) comparison between block diffusion LMS and diffusion LMS. (a) MSD curve at node 1. (b) MSD curve at node 7.

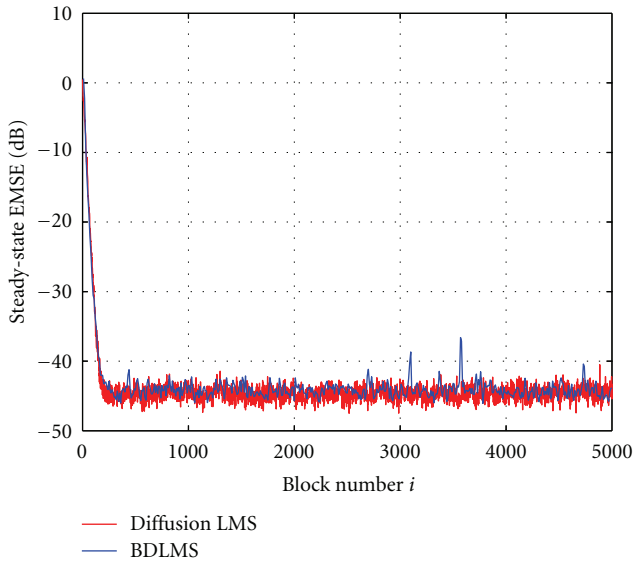

Figure 7: Local EMSE at node 7 for the same network.

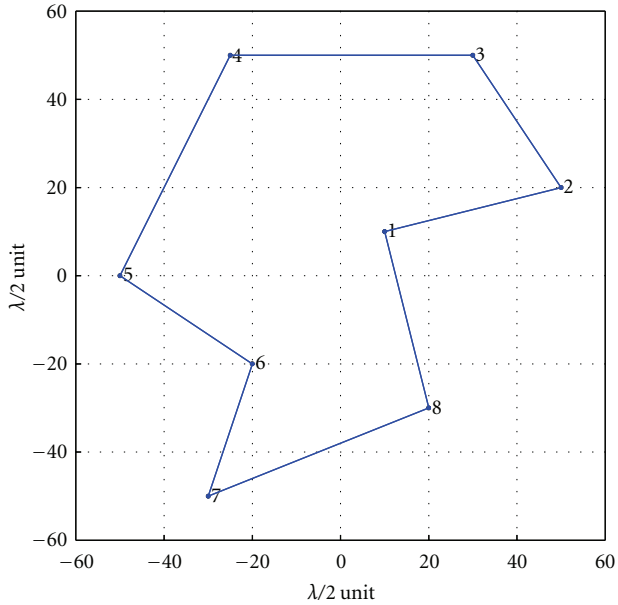

Figure 8: Network topology.

a higher number of communication overhead are required for BDLMS than BILMS algorithm.

5. Performance Comparison

In this section, we present an analysis of communication cost and latency to have a theoretical comparison of the performances of distributed LMS with block distributed LMS.

5.1. Analysis of Communication Cost. Assuming that the messages are of fixed bit width, the communication cost is modeled as the number of messages transmitted to achieve the steady-state value in the network. Let N be the number of nodes in the network, and let M be the filter length. The block length L is chosen to be the same as the filter length. Let h be the average time required for the transmission of one message, that is, for one communication between the nodes [24–26].

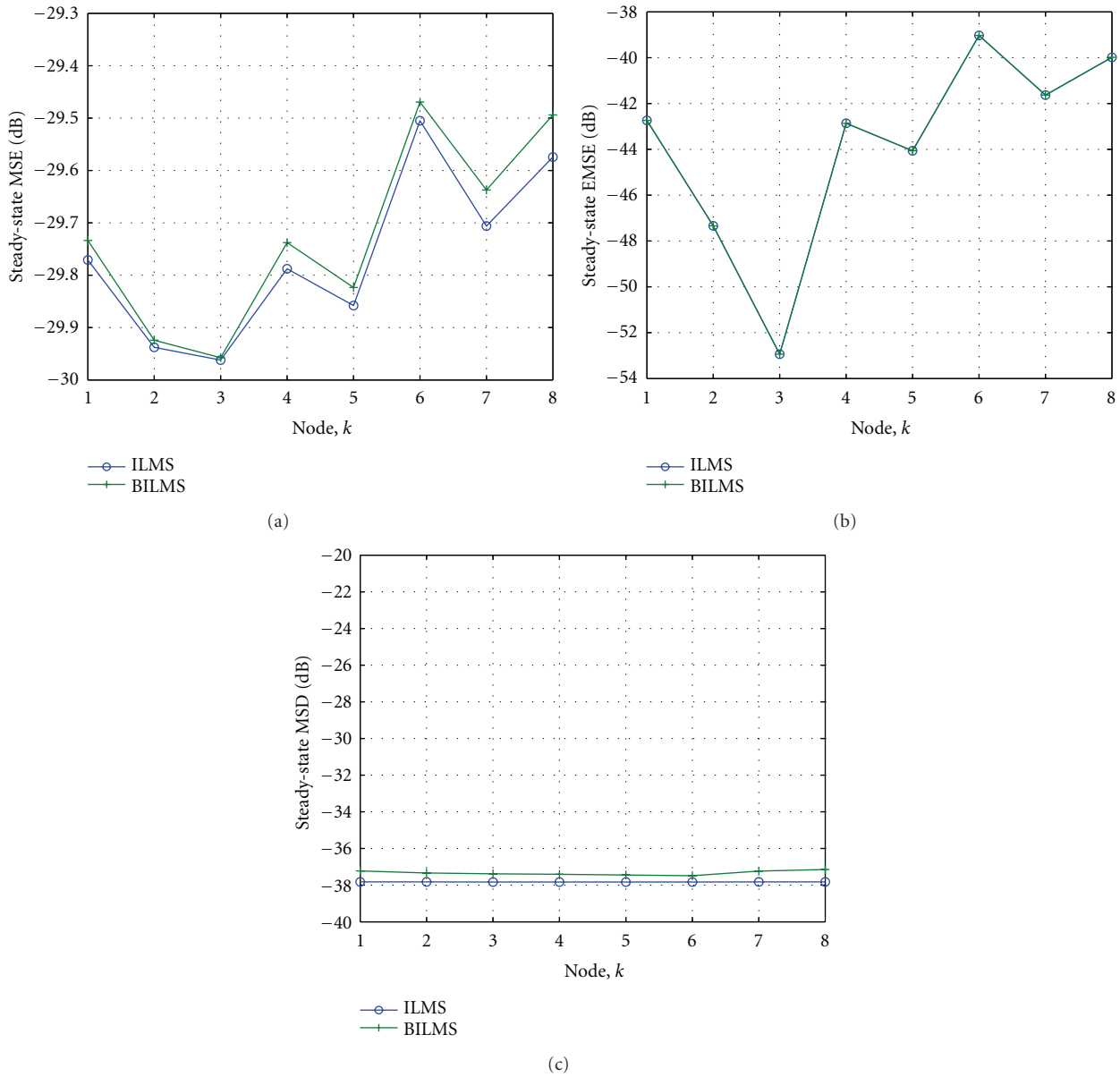

Figure 9: Network nodal performance. (a) Mean-square error (MSE) versus node. (b) excess mean-square error (EMSE) versus node. (c) Mean-square deviation (MSD) versus node.

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

$$C_{\text{ILMS}} = (N - 1)K. \tag{42}$$

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

$$C_{\text{BILMS}} = \frac{(N - 1)K}{L}. \tag{43}$$

5.1.2. DLMS and BDLMS Algorithms. The diffusion-based algorithms are communication intensive. In DLMS mode of cooperation, in each cycle, each node in the network sends its estimated information to all its connected nodes in the

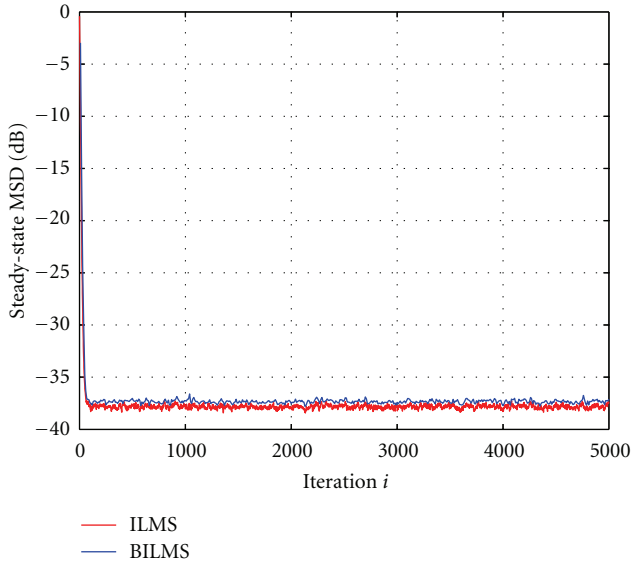

Figure 10: Global mean-square deviation (MSD) curve for incremental LMS and block incremental LMS.

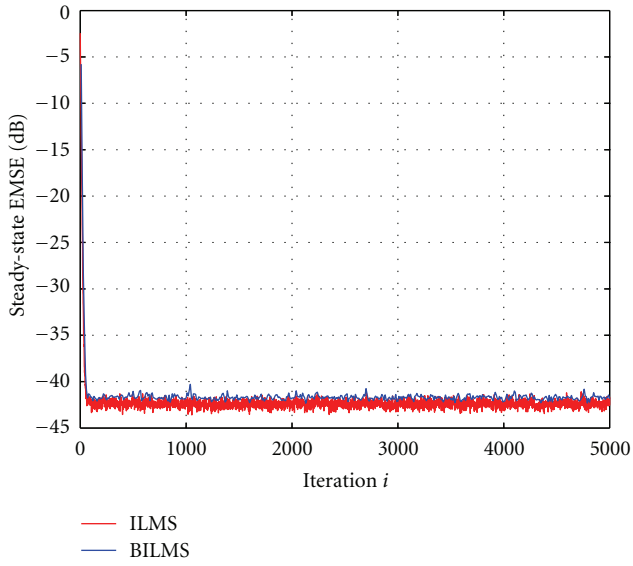

Figure 11: Global excess mean-square deviation (EMSE) curve for incremental LMS and block incremental LMS.

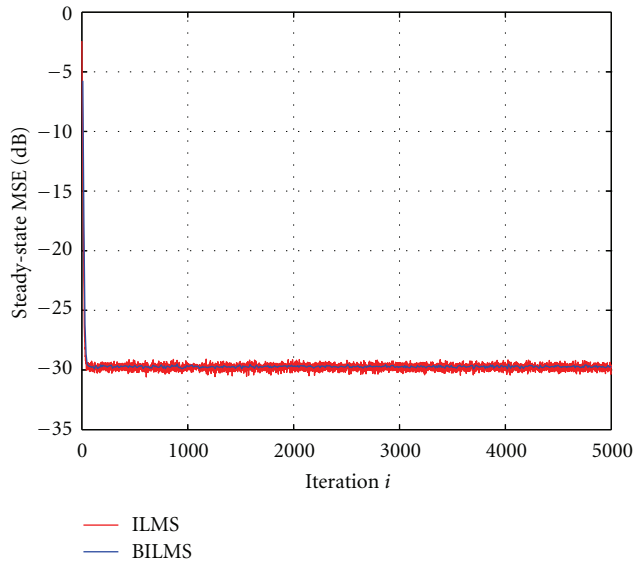

Figure 12: Global mean-square deviation (MSE) curve for incremental LMS and block incremental LMS.

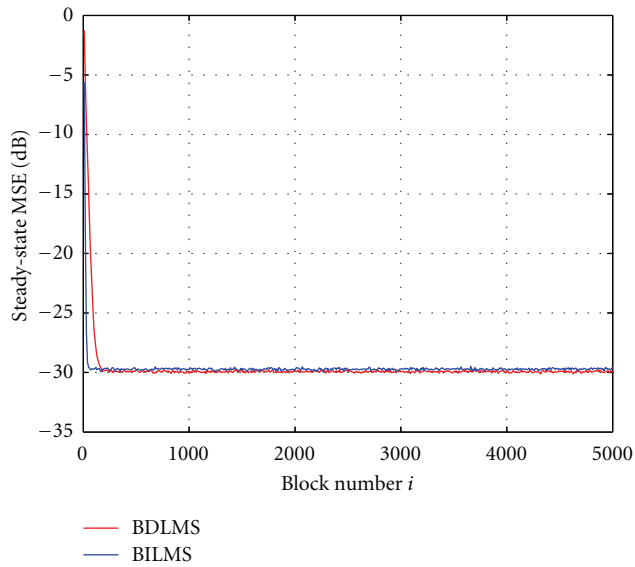

Figure 13: Global mean-square error curve for BILMS and BDLMS.

network. So the total number of messages transmitted by all the nodes in a cycle is

$$c = \sum_{i=1}^{N} n_i, \tag{44}$$

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

$$C_{\text{DLMS}} = cK. \tag{45}$$

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

$$C_{\text{BDLMS}} = \frac{cK}{L}. \tag{46}$$

5.2. Analysis of Duration for Convergence. The time interval between the arrival of input to a node and the time of reception of corresponding updates by the designated node(s) may be assumed to be comprised of two major components.

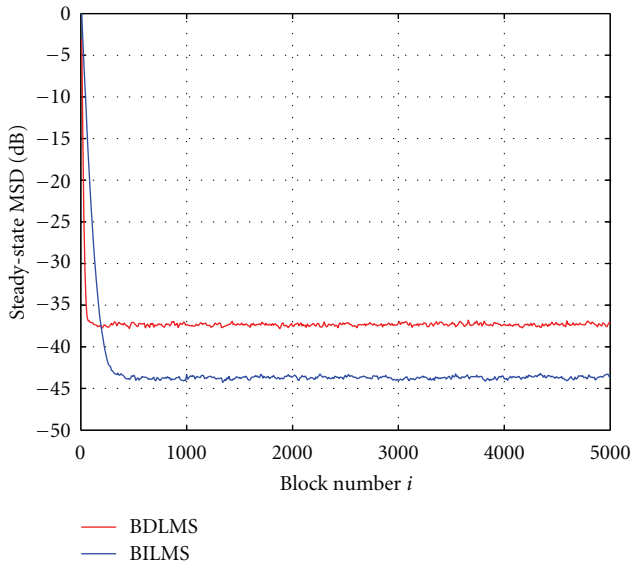

Figure 14: Global mean-square deviation curve for BILMS and BDLMS.

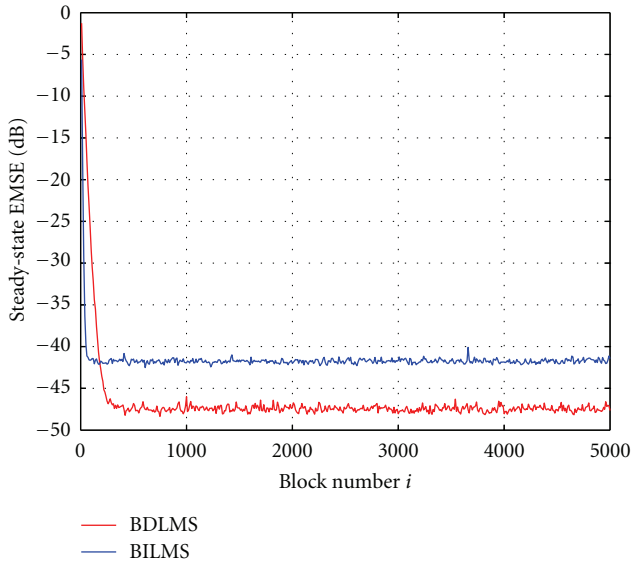

Figure 15: Global excess mean-square error curve for BILMS and BDLMS.

Those are processing delay to perform the necessary computations in a node to obtain the estimates to be updated and the communication delay involved in transferring the message to the receiver node(s). The processing delay will very much depend on the hardware architecture of the nodes to perform the computation which could be widely varying. But, without losing much of the generality of analysis, we can assume that each node has M parallel multipliers and one full adder to implement the LMS algorithm. Let T_M and T_A be the time required for executing a multiplication and an addition, respectively. Therefore, the processing delay needed for single update in LMS is

$$D = 2T_M + (M+1)T_A. \tag{47}$$

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

5.2.1. Estimation of Delays for the ILMS and BILMS Algorithms. In case of ILMS, the duration of each updating cycle by all the nodes is

$$ND + (N-1)h, \tag{48}$$

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

$$L_{\text{ILMS}} = [ND + (N-1)h]K. \tag{49}$$

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

$$L_{\text{BILMS}} = \frac{[NMD + (N-1)h]K}{L}. \tag{50}$$

For $L = M$, the above expression could be reduced to

$$L_{\text{BILMS}} = \left[ND + \frac{(N-1)h}{L}\right]K. \tag{51}$$

Comparing (51) with (49), we can find that in BILMS the processing delay remains the same as that in ILMS, but the communication overhead is reduced by L times.

5.2.2. Estimation of Delays for the DLMS and BDLMS Algorithms. Similar to ILMS, it is also assumed here that the updates of a node reaches all the connected nodes after the same average delay h. Therefore, the communication delay remains the same as that of ILMS, but in this case, it needs more processing delay to process the unbiased estimates received from the connected neighboring nodes. The total communication delay in a cycle in this case can be given by $cT_A + NT_M$, where c is the total number of messages transferred in a cycle given by (44). Now, the total duration of a cycle in diffusion LMS with the same hardware constraints is given by

$$L_{\text{DLMS}} = [cT_A + NT_M + ND + (N-1)h]K. \tag{52}$$

Table 1: Comparison of performances of distributed algorithms and block distributed algorithms.

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

Table 2: Numerical comparison of performances of sequential and block distributed adaptive algorithms.

Parameter	ILMS	BILMS	DLMS	BDLMS
Communication cost	950	95	4500	450
Duration of convergence	9.5 s	0.95 s	9.5 s	0.95 s

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

$$L_{\text{BDLMS}} = \frac{[cT_A + NT_M + NMD + (N-1)h]K}{L}. \quad (53)$$

The mathematical expressions of communication cost and latency for the distributed LMS and the block distributed LMS algorithms are summarized in Table 1. A numerical example is given in Table 2 to show the advantage of block-distributed algorithms over the sequential-distributed algorithms. The authors have simulated the hardware for 8-bit multiplication and addition in TSMC 90 nm. The multiplication and addition time are found to be $T_A = 10^{-5}$ ns, $T_M = 10^{-3}$ ns. We assume the transmission delay $h = 10^{-2}$ s. Looking at the convergence curves obtained from the simulation studies, we can say that the network attains steady state after 250-input data in DLMS and 50-input data in ILMS case. The filter length M as well as the block size L are taken to be 10 in the numerical study.

6. Conclusion

We have proposed the block implementation of the distributed LMS algorithms for WSN. The theoretical analysis and the corresponding simulation results demonstrate that the performance of the block-distributed LMS algorithms is similar to that of the sequential-distributed LMS. The remarkable achievement of the proposed algorithms is that a node requires L (block size) times of less communications compared to the conventional sequential-distributed LMS algorithms. This would be of great advantage in reducing the communication bandwidth and power consumption involved in the transmission and reception of messages across the resource-constrained nodes in a WSN. In the coming years, with continuing advances in microelectronics, we can accommodate enough computing resources in the nodes to reduce the processing delays in the nodes, but the communication bandwidth and communication delay could be the major operational bottlenecks in the WSNs. The proposed block formulation therefore would have further advantages over the sequential counterpart in the coming years.

References

[1] D. Estrin, L. Girod, G. Pottie, and M. Srivastava, "Instrumenting the world with wireless sensor networks," in *Proceedings of the IEEE Interntional Conference on Acoustics, Speech, and Signal Processing (ICASP'01)*, pp. 2033–2036, May 2001.

[2] D. Estin, R. Govindan, J. Heidemann, and S. Kumar, "Next century chanlanges: scalable cordination in sensor network," in *Proceedings of the International Conference on Mobile Computing and Networks (MobiCOM '99)*, pp. 263–270, 1999.

[3] G. J. Pottie and W. J. Kaiser, "Wireless integrated network sensors," *Communications of the ACM*, vol. 43, no. 5, pp. 51–53, 2000.

[4] M. G. Rabbat and R. D. Nowak, "Decentralized source localization and tracking [wireless sensor networks]," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, (ICASSP'04)*, vol. 3, pp. III921–III924, 2004.

[5] A. Nedić and D. P. Bertsekas, "Incremental subgradient methods for nondifferentiable optimization," *SIAM Journal on Optimization*, vol. 12, no. 1, pp. 109–138, 2001.

[6] F. S. Cattivelli and A. H. Sayed, "Diffusion LMS strategies for distributed estimation," *IEEE Transactions on Signal Processing*, vol. 58, no. 3, pp. 1035–1048, 2010.

[7] C. G. Lopes and A. H. Sayed, "Diffusion adaptive networks with changing topologies," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, (ICASSP'08)*, pp. 3285–3288, April 2008.

[8] N. Takahashi, I. Yamada, and A. H. Sayed, "Diffusion adaptive networks with changing topologies," in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing, (ICASSP '06)*, pp. 2845–2849, 2009.

[9] F. S. Cattivelli and A. H. Sayed, "Distributed detection over adaptive networks using diffusion adaptation," *IEEE Transactions on Signal Processing*, vol. 59, no. 5, pp. 1917–1932, 2011.

[10] G. A. Clark, S. K. Mitra, and S. R. Parker, "Block implementation of adaptive digital filters," *IEEE Transactions on Signal Processing*, vol. 29, no. 3, pp. 744–752, 1981.

[11] F. S. Cattivelli and A. H. Sayed, "Analysis of spatial and incremental LMS processing for distributed estimation," *IEEE Transactions on Signal Processing*, vol. 59, no. 4, pp. 1465–1480, 2011.

[12] X. Liu, Q. Huang, and Y. Zhang, "Combs, needles, haystacks: balancing push and pull for discovery in large-scale sensor networks," in *Proceedings of the Second International Conference on Embedded Networked Sensor Systems, (SenSys'04)*, pp. 122–133, November 2004.

[13] S. Madden, M. Franklin, J. Hellerstein, and W. Wong, "TAG: a tiny aggregation service for ad-hoc sensor network," in *Proceedings of the ACM Symposium on Operating System Design and Implementation (OSDI'02)*, 2002.

[14] J. Ahn and B. Krishnamachari, "Modelling search cost in wireless sensor network," Tech. Rep. CENG-2007-1, Computer Science Department, Unversity of southern California, 2007.

[15] A. Jadbabaie, J. Lin, and A. S. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *IEEE Transactions on Automatic Control*, vol. 48, no. 6, pp. 988–1001, 2003.

[16] B. Widrow and S. D. Strearns, *Adaptive Signal Processing*, Prentice Hall, Englewood Cliffs, NJ, USA, 1985.

[17] S. Haykin, *Adaptive Filter Theory*, Prentice Hall, Englewood Cliffs, NJ, USA, 2001.

[18] A. H. Sayed, *Fundamentals of Adaptive Filtering*, John Wiley and Sons, 2003.

[19] C. G. Lopes and A. H. Sayed, "Incremental adaptive strategies over distributed networks," *IEEE Transactions on Signal Processing*, vol. 55, no. 8, pp. 4064–4077, 2007.

[20] D. P. Bertsekas and J. N. Tsitsiklis, "Comments on "Coordination of groups of mobile autonomous agents using nearest neighbor rules"," *IEEE Transactions on Automatic Control*, vol. 52, no. 5, pp. 968–969, 2007.

[21] C. G. Lopes and A. H. Sayed, "Diffusion least-mean squares over adaptive networks: formulation and performance analysis," *IEEE Transactions on Signal Processing*, vol. 56, no. 7, pp. 3122–3136, 2008.

[22] A. Geary and D. P. Bertsekas, "Incremental subgradient methods for nondifferentiable optimization," in *Proceedings of the 38th IEEE Conference on Decision and Control (CDC'99)*, pp. 907–912, December 1999.

[23] D. P. Bertsekas, "A new class of incremental gradient methods for least squares problems," *SIAM Journal on Optimization*, vol. 7, no. 4, pp. 913–926, 1997.

[24] R. F. Woods, J. V. McCanny, and J. G. McWhirter, "From bit level systolic arrays to HDTV processor chips," *Journal of Signal Processing Systems*, vol. 53, no. 1-2, pp. 35–49, 2008.

[25] L.-K. Ting, R. Woods, and C. F. N. Cowan, "Virtex FPGA implementation of a pipelined adaptive LMS predictor for electronic support measures receivers," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 13, no. 1, pp. 86–95, 2005.

[26] P. Fletcher and M. Dean, "Low complexity implementation of LMS algorithm," *Electronics Letters*, vol. 38, no. 15, pp. 836–837, 2002.