Diffusion LMS Algorithms with Information Exchange

Federico S. Cattivelli and Ali H. Sayed

Department of Electrical Engineering
University of California, Los Angeles, CA 90095
Emails: {fcattiv, sayed}@ee.ucla.edu

***Abstract*— We consider the problem of distributed estimation, where a set of nodes are required to collectively estimate some parameter of interest. We motivate and propose new versions of the diffusion LMS algorithm, including a version that outperforms previous solutions without increasing the complexity or communications, and others that obtain even better performance by allowing additional communications. We analyze their performance and compare with simulation results.**

I. Introduction

We consider the problem of distributed estimation, where a set of nodes are required to collectively estimate some parameter of interest. Consider a set of N nodes distributed over some region (see Fig. 1). At every time instant i, every node k takes a scalar measurement $d_k(i)$ of some random process $\boldsymbol{d}_k(i)$ and a $1 \times M$ regression vector, $u_{k,i}$, corresponding to a realization of a random process $\boldsymbol{u}_{k,i}$, which is correlated with $\boldsymbol{d}_k(i)$. The objective is for every node in the network to use the data $\{d_k(i), u_{k,i}\}$ to estimate some parameter vector w.

In the centralized solution to the problem, every node in the network transmits its data $\{d_k(i), u_{k,i}\}$ to a central fusion center for processing. This approach has the disadvantage of requiring large amounts of energy and communication resources.

In distributed implementations, every node in the network communicates with a subset of the nodes. The set of nodes that are connected to node k, including k itself, is denoted by $\mathcal{N}_k$ and is called the neighborhood of node k.

Distributed estimation algorithms have been proposed in the context of distributed adaptive filtering [1], [2]. These include incremental LMS [2]-[3], incremental RLS [2], diffusion LMS [4] and diffusion RLS [5]. Diffusion Kalman filtering [6] and smoothing [7] have also been proposed. Distributed estimation algorithms based on consensus strategies have been proposed in [8], [9].

II. Problem Formulation

A. Global optimization

We seek the optimal linear estimator w^o that minimizes the following global cost function:

$$J^{\text{glob}}(w) \triangleq \sum_{k=1}^{N} \mathrm{E}\,|\boldsymbol{d}_k(i) - \boldsymbol{u}_{k,i} w|^2 \tag{1}$$

where E denotes the expectation operator. Assuming the processes $\boldsymbol{d}_k(i)$ and $\boldsymbol{u}_{k,i}$ are jointly wide sense stationary (WSS), the optimal solution is given by [10], [11]:

$$w^o = \left(\sum_{k=1}^{N} R_{u,k}\right)^{-1} \left(\sum_{k=1}^{N} R_{du,k}\right) \tag{2}$$

where $R_{u,k} = \mathrm{E}\,\boldsymbol{u}_{k,i}^* \boldsymbol{u}_{k,i}$ and $R_{du,k} = \mathrm{E}\,\boldsymbol{d}_k(i) \boldsymbol{u}_{k,i}^*$, and where the operator * denotes complex conjugate-transposition.

This material was based on work supported in part by the National Science Foundation under awards ECS-0601266 and ECS-0725441.

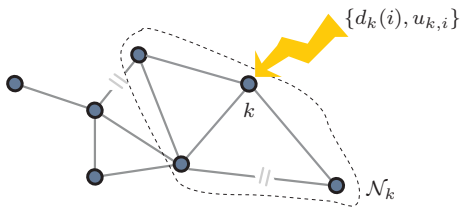

Fig. 1. Node k takes a measurement at time i.

B. Local optimization

Now consider an $N \times N$ matrix C with individual non-negative real entries $\{c_{l,k}\}$ such that

$$c_{l,k} = 0 \;\text{ if } l \notin \mathcal{N}_k \qquad C\mathbb{1} = \mathbb{1} \qquad \mathbb{1}^* C = \mathbb{1}^* \tag{3}$$

where $\mathbb{1}$ denotes the $N \times 1$ vector with unit entries. When node k has access only to the data from its neighbors $\{l \in \mathcal{N}_k\}$, it can then seek to minimize the following *local* cost function:

$$J_k^{\text{loc}}(w) = \sum_{l \in \mathcal{N}_k} c_{l,k}\, \mathrm{E}\,|\boldsymbol{d}_l(i) - \boldsymbol{u}_{l,i} w|^2 \tag{4}$$

where the coefficients $c_{l,k}$ give different weights to the data from the neighbors. The local optimal solution is therefore:

$$w_k^{\text{loc}} = \Gamma_k^{-1} \left(\sum_{l \in \mathcal{N}_k} c_{l,k} R_{du,l}\right) \quad \text{where} \quad \Gamma_k \triangleq \sum_{l \in \mathcal{N}_k} c_{l,k} R_{u,l} \tag{5}$$

A completion-of-squares argument shows that (4) can be rewritten in terms of w_k^{loc} as:

$$J_k^{\text{loc}}(w) = \|w - w_k^{\text{loc}}\|_{\Gamma_k}^2 + \text{mmse} \tag{6}$$

where mmse is a constant that does not depend on w and $\|a\|_\Sigma^2 = a^* \Sigma a$ represents a weighted vector norm for any Hermitian $\Sigma > 0$.

An interesting question is how do the local solutions (5) at all nodes relate to the global solution (2)? Because of (3), note that we can write the global cost (1) as

$$J^{\text{glob}}(w) = \sum_{l=1}^{N} J_l^{\text{loc}}(w) = J_k^{\text{loc}}(w) + \sum_{l \neq k}^{N} J_l^{\text{loc}}(w) \tag{7}$$

Thus, using (4), (6) and (7), we find that minimizing the global cost (1) is equivalent to minimizing the following cost function, for any $k \in \{1, ..., N\}$:

$$J^{\text{glob}'}(w) = \sum_{l \in \mathcal{N}_k} c_{l,k}\, \mathrm{E}\,|\boldsymbol{d}_l(i) - \boldsymbol{u}_{l,i} w|^2 + \sum_{l \neq k}^{N} \|w - w_l^{\text{loc}}\|_{\Gamma_l}^2 \tag{8}$$

In the following sections we will show that Equation (8) suggests distributed implementations of the diffusion type, and will be instrumental in the derivation of different diffusion estimation algorithms.

III. Steepest-Descent global solution

To begin with, consider minimizing the cost function (1) using the traditional iterative steepest-descent solution [10]:

$$w_i = w_{i-1} - \mu \left[\nabla_w J^{\rm glob}(w_{i-1})\right]^* \tag{9}$$

where $\mu > 0$ is a step-size parameter and w_i is an estimate for w^o at iteration i. Moreover, $\nabla_w J^{\rm glob}$ denotes the complex gradient of $J^{\rm glob}(w)$ with respect to w, which is given by

$$\left[\nabla_w J^{\rm glob}(w)\right]^* = \sum_{k=1}^{N} (R_{u,k} w - R_{du,k}) \tag{10}$$

Substituting into (9) leads to the steepest descent iteration:

$$w_i = w_{i-1} + \mu \sum_{k=1}^{N} (R_{du,k} - R_{u,k} w_{i-1}) \tag{11}$$

Now note that (11) requires knowledge of the second-order moments $\{R_{u,k}, R_{du,k}\}$. An adaptive implementation of (11) can be obtained by replacing these second-order moments by local instantaneous approximations, say of the LMS type, as follows:

$$R_{u,k} \approx u_{k,i}^* u_{k,i} \qquad R_{du,k} \approx d_k(i) u_{k,i}^* \tag{12}$$

Then a global (centralized) LMS recursion is obtained, namely:

$$w_i = w_{i-1} + \mu \sum_{k=1}^{N} u_{k,i}^* (d_k(i) - u_{k,i} w_{i-1}) \tag{13}$$

In the next section we will consider distributed strategies.

IV. Diffusion Adaptive Solutions

The steepest-descent solution (13) is not distributed, since every node in the network needs to have access to global information (namely, the measurements and regressors of every other node) in order to compute the new estimate. A fully distributed solution based on diffusion strategies was proposed in [2], [4] and is known as *diffusion* LMS.

We now propose more general variants that can accommodate higher levels of interaction and information exchange among the nodes. The formulation that follows includes the diffusion LMS algorithm of [2], [4] as a special case.

A. MSE minimization

Thus, refer to the equivalent global cost (8). Minimizing this cost at every node requires the nodes to have access to global information, namely the local estimates, $w_l^{\rm loc}$, and the matrices Γ_l, of the nodes in the network. In order to facilitate distributed implementations, we shall instead use the cost (8) to motivate alternative formulations that lead to useful distributed algorithms. To begin with, we replace Γ_l in (8) with weighting matrices of the form $\Gamma_l = \gamma_k b_{l,k} I_M$, where γ_k is some constant and $b_{l,k}$ is a set of non-negative real coefficients that give different weights to different neighbors. In particular, we are interested in choices of coefficients such that

$$b_{l,k} = 0 \ \text{ if } l \notin \mathcal{N}_k \qquad \mathbb{1}^* B = \mathbb{1}^* \tag{14}$$

where B is the $N \times N$ matrix with individual entries $b_{l,k}$. Furthermore, we replace the optimal local estimate $w_l^{\rm loc}$ in (8) with the best estimate that is available at node l, and denote it by ψ_l. In this way, each node k can proceed to minimize a cost of the form:

$$J_k^{\rm dist}(w) = \sum_{l\in\mathcal{N}_k} c_{l,k}\, \mathrm{E}\,|\boldsymbol{d}_l(i) - \boldsymbol{u}_{l,i} w|^2 + \gamma_k \sum_{l\in\mathcal{N}_k/\{k\}} b_{l,k} \|w - \psi_l\|^2 \tag{15}$$

Taking the gradient of (15) we obtain:

$$\left[\nabla_w J_k^{\rm dist}(w)\right]^* = \sum_{l\in\mathcal{N}_k} c_{l,k}(R_{u,l} w - R_{du,l}) + \gamma_k \sum_{l\in\mathcal{N}_k/\{k\}} b_{l,k}(w - \psi_l) \tag{16}$$

Thus, we can use (15) to obtain a recursion for the estimate of node k, denoted by $w_{k,i}$, as we did in the steepest-descent case. However, note that the above gradient is a sum of two terms, namely

$$\sum_{l\in\mathcal{N}_k} c_{l,k}(R_{u,l} w - R_{du,l}) \quad \text{ and } \quad \gamma_k \sum_{l\in\mathcal{N}_k/\{k\}} b_{l,k}(w - \psi_l)$$

Incremental solutions are useful for minimizing sums of convex functions [12], and are based on the principle of iterating sequentially over each sub-gradient, in some pre-defined order. An incremental LMS algorithm was proposed in [3] based on these ideas. Here we propose iterating incrementally between the two above sub-gradients, as follows:

$$\begin{cases} \psi_{k,i} &= w_{k,i-1} + \mu_k \sum_{l\in\mathcal{N}_k} c_{l,k}\,(R_{du,l} - R_{u,l} w_{k,i-1}) \\ w_{k,i} &= \psi_{k,i} + \nu_k \gamma_k \sum_{l\in\mathcal{N}_k/\{k\}} b_{l,k}(\psi_{l,i} - \psi_{k,i}) \end{cases} \tag{17}$$

where we have replaced ψ_l with the best estimate that is available at node l at time i, namely, $\psi_{l,i}$. Note that

$$w_{k,i} = (1 - \nu_k\gamma_k + \nu_k\gamma_k b_{k,k})\psi_{k,i} + \nu_k\gamma_k \sum_{l\in\mathcal{N}_k/\{k\}} b_{l,k}\psi_{l,i} \tag{18}$$

so that if we define $a_{k,k} = (1 - \nu_k\gamma_k + \nu_k\gamma_k b_{k,k})$ and $a_{l,k} = \nu_k\gamma_k b_{l,k}$ for $l \neq k$, and use the instantaneous approximations (12), we obtain the Adapt-then-Combine (ATC) diffusion LMS algorithm.

ATC Diffusion LMS: Start with $\{w_{l,-1} = 0\}$ for all l. For each time $i \geq 0$ and for each node k, repeat:

$$\begin{cases} \psi_{k,i} = w_{k,i-1} + \mu_k \sum_{l\in\mathcal{N}_k} c_{l,k} u_{l,i}^* (d_l(i) - u_{l,i} w_{k,i-1}) \\ w_{k,i} = \sum_{l\in\mathcal{N}_k} a_{l,k} \psi_{l,i} \end{cases} \tag{19}$$

Note that the coefficients $a_{l,k}$ are real, non-negative and also satisfy

$$a_{l,k} = 0 \ \text{ if } l \notin \mathcal{N}_k \qquad \mathbb{1}^* A = \mathbb{1}^* \tag{20}$$

where A is the $N \times N$ matrix with individual entries $a_{l,k}$. If we reverse the order by which we perform the incremental update, we obtain the Combine-then-Adapt (CTA) diffusion LMS algorithm.

CTA Diffusion LMS: Start with $\{\psi_{l,-1} = 0\}$ for all l. For each time $i \geq 0$ and for each node k, repeat:

$$\begin{cases} \psi_{k,i-1} = \sum_{l\in\mathcal{N}_k} a_{l,k} w_{l,i-1} \\ w_{k,i} = \psi_{k,i-1} + \mu_k \sum_{l\in\mathcal{N}_k} c_{l,k} u_{l,i}^* (d_l(i) - u_{l,i} \psi_{k,i-1}) \end{cases} \tag{21}$$

Recall that A and C denote $N \times N$ matrices with individual entries $\{a_{l,k}\}$ and $\{c_{l,k}\}$, respectively. For each algorithm, we distinguish between two cases: the case when measurements and regressors are not exchanged between the nodes (or, equivalently, $C = I$), and the case when measurements and regressors are exchanged ($C \neq I$).

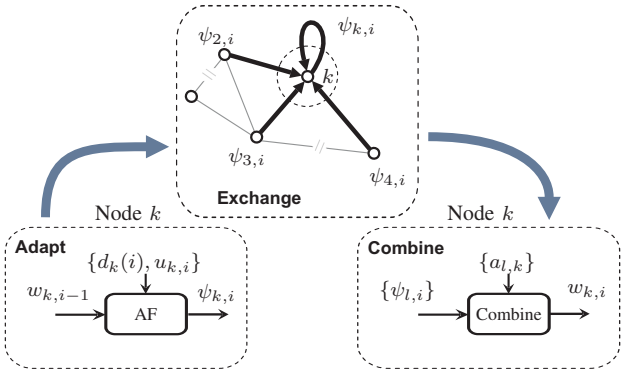

Fig. 2. ATC diffusion strategy, where measurements are not shared ($C = I$).

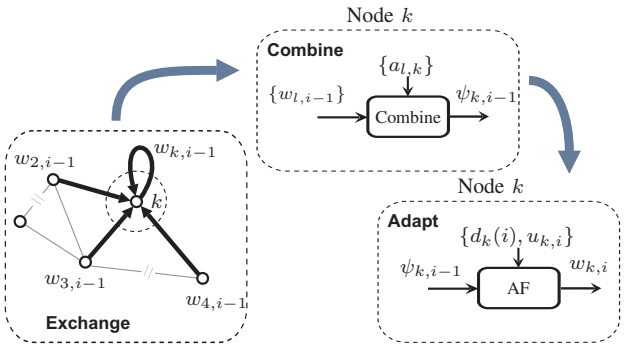

Fig. 3. CTA diffusion strategy, where measurements are not shared ($C = I$).

Note that in the former case, the CTA diffusion LMS algorithm (21) reduces to the original diffusion LMS algorithm [4]:

$$\begin{cases} \psi_{k,i-1} = \sum_{l\in\mathcal{N}_k} a_{l,k} w_{l,i-1} \\ w_{k,i} = \psi_{k,i-1} + \mu_k u_{k,i}^*(d_k(i) - u_{k,i}\psi_{k,i-1}) \end{cases}$$

At each iteration i, every node k performs a procedure consisting of up to four steps, as shown in Table 1. For example, the ATC algorithm without measurement exchange ($C = I$), consists of three steps. First, every node adapts its current estimate using its individual measurements available at time i, namely $\{d_k(i), u_{k,i}\}$, to obtain $\psi_{k,i}$. Second, all nodes exchange their pre-estimates $\psi_{k,i}$ with their neighbors. Finally, every node combines the pre-estimates to obtain the new estimate $w_{k,i}$. Figures 2 and 3 show schematically the cooperation strategies for the ATC and CTA algorithms, respectively, for the case where measurements are not shared ($C = I$).

When $C = I$, the ATC algorithm has the same processing and communication complexity as the CTA algorithm. In the next section we will see that the ATC version outperforms the CTA version in general, and therefore also outperforms diffusion LMS [4], without penalty. When $C \neq I$, both the ATC and CTA algorithms will require two exchanges per iteration, and therefore require more communications than diffusion LMS [4].

V. Performance Analysis

In this section we analyze the diffusion LMS algorithms in their ATC (19) and CTA (21) forms. In what follows we will consider the estimates $\boldsymbol{w}_{k,i}$ to be random processes, and will analyze their performance in terms of their expected behavior. Instead of analyzing each algorithm separately, we formulate a general form that includes the ATC and CTA algorithms as special cases. Subsequently, we

TABLE I

Steps for ATC and CTA algorithms with and without measurement sharing.

Step	ATC	ATC (C=I)	CTA	CTA (C=I)
1	Exchange $\{d_k(i), u_{k,i}\}$	Adapt	Exchange $w_{k,i-1}$	Exchange $w_{k,i-1}$
2	Adapt	Exchange $\psi_{k,i}$	Combine	Combine
3	Exchange $\psi_{k,i}$	Combine	Exchange $\{d_k(i), u_{k,i}\}$	Adapt
4	Combine	-	Adapt	-

derive expressions for the mean-square deviation (MSD) and excess mean-square error (EMSE) of the general form, and specialize the results to the ATC and CTA cases. Thus, consider a general LMS diffusion filter of the form:

$$\begin{cases} \boldsymbol{\phi}_{k,i} &= \sum_{l\in\mathcal{N}_k} b_{l,k}\boldsymbol{w}_{l,i-1} \\ \boldsymbol{\psi}_{k,i} &= \boldsymbol{\phi}_{k,i} + \mu_k \sum_{l\in\mathcal{N}_k} c_{l,k}\boldsymbol{u}_{l,i}^*[\boldsymbol{d}_l(i) - \boldsymbol{u}_{l,i}\boldsymbol{\phi}_{k,i}] \\ \boldsymbol{w}_{k,i} &= \sum_{l\in\mathcal{N}_k} a_{l,k}\boldsymbol{\psi}_{l,i} \end{cases} \quad (22)$$

where the coefficients $a_{l,k}$ and $b_{l,k}$ satisfy (14) and (20), respectively, and $c_{l,k}$ satisfies (3). Let A, B and C denote the matrices with $\{l,k\}$ entries given by $a_{l,k}$, $b_{l,k}$ and $c_{l,k}$, respectively. Equation (22) can be specialized to the ATC diffusion LMS algorithm (19) by choosing $B = I$, to the CTA diffusion LMS algorithm (21) by choosing $A = I$ and B as the combination matrix (i.e., $b_{l,k}$ in (22) would correspond to $a_{l,k}$ in (21)), and to the diffusion LMS algorithm from [4] by choosing $A = C = I$ and B as the combination matrix. The form in (22) allows us to analyze all these variants uniformly, and in the end we specialize the results to the different algorithms.

To proceed with the analysis, we assume a linear measurement model as follows:

$$\boldsymbol{d}_k(i) = \boldsymbol{u}_{k,i}w^o + \boldsymbol{v}_k(i) \quad (23)$$

where $\boldsymbol{v}_k(i)$ is a zero-mean random variable with variance $\sigma_{v,k}^2$, independent of $\boldsymbol{u}_{k,i}$ for all k and i, and independent of $\boldsymbol{v}_l(j)$ for $l \neq k$ or $i \neq j$. Linear models as in (23) are customary in the adaptive filtering literature [10] since they are able to capture many cases of interest. Note that w^o in the above equation is the same as the optimal solution in (2), since in this case $R_{du,k} = R_{u,k}w^o$.

Using (22), we define the error quantities $\tilde{\boldsymbol{w}}_{k,i} = w^o - \boldsymbol{w}_{k,i}$, $\tilde{\boldsymbol{\psi}}_{k,i} = w^o - \boldsymbol{\psi}_{k,i}$ and $\tilde{\boldsymbol{\phi}}_{k,i} = w^o - \boldsymbol{\phi}_{k,i}$, and the global vectors:

$$\tilde{\boldsymbol{w}}_i = \begin{bmatrix} \tilde{\boldsymbol{w}}_{1,i} \\ \vdots \\ \tilde{\boldsymbol{w}}_{N,i} \end{bmatrix}, \quad \tilde{\boldsymbol{\psi}}_i = \begin{bmatrix} \tilde{\boldsymbol{\psi}}_{1,i} \\ \vdots \\ \tilde{\boldsymbol{\psi}}_{N,i} \end{bmatrix}, \quad \tilde{\boldsymbol{\phi}}_i = \begin{bmatrix} \tilde{\boldsymbol{\phi}}_{1,i} \\ \vdots \\ \tilde{\boldsymbol{\phi}}_{N,i} \end{bmatrix}$$

We also introduce the matrix:

$$\mathcal{M} = \text{diag}\{\mu_1 I_M, \ldots, \mu_N I_M\} \quad (24)$$

and the extended weighting matrices:

$$\mathcal{A} = A \otimes I_M \qquad \mathcal{B} = B \otimes I_M \qquad \mathcal{C} = C \otimes I_M \quad (25)$$

We also introduce the following matrices:

$$\boldsymbol{D}_i = \text{diag}\left\{\sum_{l=1}^N c_{l,1}\boldsymbol{u}_{l,i}^*\boldsymbol{u}_{l,i}\ ,\ \ldots\ ,\ \sum_{l=1}^N c_{l,N}\boldsymbol{u}_{l,i}^*\boldsymbol{u}_{l,i}\right\}$$

$$\boldsymbol{G}_i = \mathcal{C}^T \mathrm{col}\left\{\boldsymbol{u}_{1,i}^* \boldsymbol{v}_1(i)\ ,\ \ldots\ ,\ \boldsymbol{u}_{N,i}^* \boldsymbol{v}_N(i)\right\}$$

Then we have

$$\begin{aligned}\tilde{\boldsymbol{\phi}}_i &= \mathcal{B}^T \tilde{\boldsymbol{w}}_{i-1} \\ \tilde{\boldsymbol{\psi}}_i &= \tilde{\boldsymbol{\phi}}_i - \mathcal{M}[\boldsymbol{D}_i \tilde{\boldsymbol{\phi}}_i + \boldsymbol{G}_i] \\ \tilde{\boldsymbol{w}}_i &= \mathcal{A}^T \tilde{\boldsymbol{\psi}}_i\end{aligned}$$

or, equivalently,

$$\tilde{\boldsymbol{w}}_i = \mathcal{A}^T[I - \mathcal{M}\boldsymbol{D}_i]\mathcal{B}^T \tilde{\boldsymbol{w}}_{i-1} - \mathcal{A}^T \mathcal{M}\boldsymbol{G}_i \tag{26}$$

Moreover, let

$$\mathcal{D} \triangleq \mathrm{E}\,\boldsymbol{D}_i = \mathrm{diag}\left\{\sum_{l=1}^{N} c_{l,1} R_{u,l}\ ,\ \ldots\ ,\ \sum_{l=1}^{N} c_{l,N} R_{u,l}\right\} \tag{27}$$

and

$$\mathcal{G} \triangleq \mathrm{E}[\boldsymbol{G}_i \boldsymbol{G}_i^*] = \mathcal{C}^T \times \mathrm{diag}\left\{\sigma_{v,1}^2 R_{u,1}\ ,\ \ldots\ ,\ \sigma_{v,N}^2 R_{u,N}\right\} \times \mathcal{C} \tag{28}$$

A. Mean-square analysis

We follow the energy conservation analysis of [10]-[11]. Evaluating the weighted norm of $\tilde{w}_i$ in (26) we obtain:

$$\begin{aligned}\mathrm{E}\,||\tilde{\boldsymbol{w}}_i||_{\Sigma}^2 &= \mathrm{E}\,||\tilde{\boldsymbol{w}}_{i-1}||^2_{\mathcal{B}(I-\boldsymbol{D}_i M)\mathcal{A}\Sigma\mathcal{A}^T(I-\mathcal{M}\boldsymbol{D}_i)\mathcal{B}^T} + \\ &\quad \mathrm{E}[\boldsymbol{G}_i^* \mathcal{M}\mathcal{A}\Sigma\mathcal{A}^T\mathcal{M}\boldsymbol{G}_i]\end{aligned} \tag{29}$$

where Σ is any Hermitian positive-definite matrix. We now introduce the *independence assumption*:

Assumption 1 (Independence): *All regressors $\boldsymbol{u}_{k,i}$ are spatially and temporally independent.*

This allows us to consider $\boldsymbol{D}_i$ independent of $\tilde{\boldsymbol{w}}_{i-1}$, which depends on the regressors up to time $i-1$. Then we can rewrite (29) as a variance relation as:

$$\begin{aligned}\mathrm{E}\,||\tilde{\boldsymbol{w}}_i||_{\Sigma}^2 &= \mathrm{E}\,||\tilde{\boldsymbol{w}}_{i-1}||^2_{\Sigma'} + \mathrm{Tr}[\Sigma\mathcal{A}^T\mathcal{M}\mathcal{G}\mathcal{M}\mathcal{A}] \\ \Sigma' &= \mathcal{B}\mathcal{A}\Sigma\mathcal{A}^T\mathcal{B}^T - \mathcal{B}\mathcal{D}\mathcal{M}\mathcal{A}\Sigma\mathcal{A}^T\mathcal{B}^T - \\ &\quad \mathcal{B}\mathcal{A}\Sigma\mathcal{A}^T\mathcal{M}\mathcal{D}\mathcal{B}^T + \mathrm{E}(\mathcal{B}\boldsymbol{D}_i\mathcal{M}\mathcal{A}\Sigma\mathcal{A}^T\mathcal{M}\boldsymbol{D}_i\mathcal{B}^T)\end{aligned} \tag{30}$$

We also assume that the step-sizes are small enough and ignore the last term in Σ' because of its dependence on μ^2. The general case is omitted due to space considerations, and will be considered in a future publication.

Assumption 2 (Small step-size): *The step-sizes $\{\mu_k\}$, $k = 1, \ldots, N$, are sufficiently small such that the rightmost term of (30) can be neglected.*

From Assumption 2 we get:

$$\Sigma' = \mathcal{B}\left(\mathcal{A}\Sigma\mathcal{A}^T - \mathcal{D}\mathcal{M}\mathcal{A}\Sigma\mathcal{A}^T - \mathcal{A}\Sigma\mathcal{A}^T\mathcal{M}\mathcal{D}\right)\mathcal{B}^T$$

Let

$$\sigma = \mathrm{vec}(\Sigma) \qquad \Sigma = \mathrm{vec}^{-1}(\sigma)$$

where the $\mathrm{vec}(\cdot)$ notation stacks the columns of the matrix argument on top of each other. We will also use the notation $\|\tilde{w}\|_{\sigma}^2$ to denote $\|\tilde{w}\|_{\Sigma}^2$. Using the Kronecker product property

$$\mathrm{vec}(P\Sigma Q) = (Q^T \otimes P)\mathrm{vec}(\Sigma)$$

we arrive at

$$\sigma' \triangleq \mathrm{vec}(\Sigma') = F\sigma$$

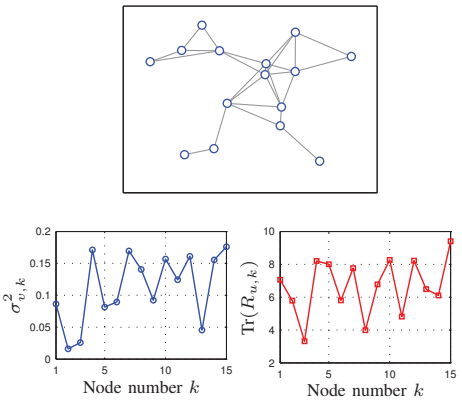

Fig. 4. Network topology (top), noise variances $\sigma_{v,k}^2$ (bottom, left) and trace of regressor covariances $\mathrm{Tr}(R_{u,k})$ (bottom, right) for $N = 15$ nodes.

where

$$\boxed{F = (\mathcal{B} \otimes \mathcal{B})\left\{\mathcal{A} \otimes \mathcal{A} - \mathcal{A} \otimes (\mathcal{D}\mathcal{M}\mathcal{A}) - (\mathcal{D}^T\mathcal{M}\mathcal{A}) \otimes \mathcal{A}\right\}} \tag{31}$$

Then, using the result that $\mathrm{Tr}(\Sigma X) = \mathrm{vec}(X^T)^T\sigma$ we arrive at

$$\mathrm{E}\,||\tilde{\boldsymbol{w}}_{\infty}||^2_{(I-F)\sigma} = [\mathrm{vec}(\mathcal{A}^T\mathcal{M}\mathcal{G}^T\mathcal{M}\mathcal{A})]^T\sigma \tag{32}$$

The MSD at node k can be obtained by weighting $\mathrm{E}\,||\tilde{\boldsymbol{w}}_{\infty}||^2$ with a block matrix that has an identity matrix at block $\{k,k\}$ and zeros elsewhere. Let us denote the vectorized version of this matrix by q_k, that is:

$$q_k = \mathrm{vec}(\mathrm{diag}(e_k) \otimes I_M)$$

Then the MSD becomes:

$$\boxed{\mathrm{MSD}_k = \mathrm{E}\,||\tilde{\boldsymbol{w}}_{\infty}||^2_{q_k} = [\mathrm{vec}(\mathcal{A}^T\mathcal{M}\mathcal{G}^T\mathcal{M}\mathcal{A})]^T(I-F)^{-1}q_k} \tag{33}$$

The EMSE at node k is obtained by weighting $\mathrm{E}\,||\tilde{\boldsymbol{w}}_{\infty}||^2$ with a block matrix that has R_{u_k} at block $\{k,k\}$ and zeros elsewhere, that is, by selecting

$$(I-F)\sigma = r_k = \mathrm{vec}(\mathrm{diag}(e_k) \otimes R_{u_k})$$

Then the EMSE becomes:

$$\boxed{\mathrm{EMSE}_k = \mathrm{E}\,||\tilde{\boldsymbol{w}}_{\infty}||^2_{r_k} = [\mathrm{vec}(\mathcal{A}^T\mathcal{M}\mathcal{G}^T\mathcal{M}\mathcal{A})]^T(I-F)^{-1}r_k} \tag{34}$$

The *network* MSD and EMSE are defined as the average MSD and EMSE, respectively, across all nodes in the network:

$$\begin{aligned}\mathrm{MSD}^{\mathrm{network}} &\triangleq \frac{1}{N}\sum_{k=1}^{N}\mathrm{MSD}_k = \frac{1}{N}\mathrm{E}\,||\tilde{\boldsymbol{w}}_{\infty}||^2 \\ \mathrm{EMSE}^{\mathrm{network}} &\triangleq \frac{1}{N}\sum_{k=1}^{N}\mathrm{EMSE}_k = \frac{1}{N}\mathrm{E}\,||\tilde{\boldsymbol{w}}_{\infty}||^2_{\mathrm{diag}\{R_{u,1},\ldots,R_{u,N}\}}\end{aligned}$$

VI. Simulations

In order to illustrate the adaptive network performance, we present a simulation example in Figs. 4-6. Fig. 4 depicts the network topology with $N = 15$ nodes, together with the network statistical profile. The regressors have size $M = 3$, are zero-mean Gaussian, independent in time and space and have covariance matrices $R_{u,k}$. The background noise power is denoted by $\sigma_{v,k}^2$.

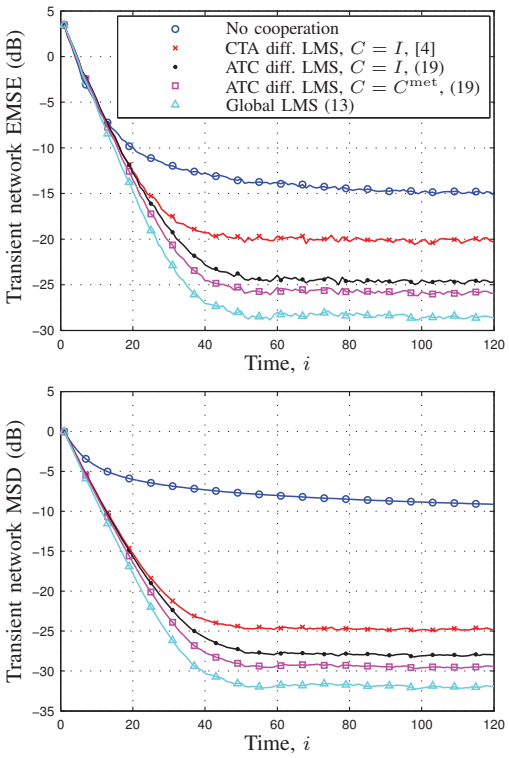

Fig. 5. Transient network EMSE (top) and MSD (bottom) for LMS without cooperation, CTA and ATC diffusion LMS, and global LMS.

Fig. 5 shows the learning curves for different diffusion LMS algorithms in terms of EMSE and MSD. The simulations use a value of $\mu = 0.08$, and the results are averaged over 200 experiments. For the diffusion algorithms, relative-degree weights [5] are used for the adaptation matrix. Relative-degree weights are defined as:

$$a_{l,k} = \begin{cases} n_l / \sum_{l \in \mathcal{N}_k} n_l & l \in \mathcal{N}_k \\ 0 & \text{otherwise} \end{cases}$$

where n_k is the degree of node k, defined as the number of neighbors of node k including itself. For the combination matrix C, we present two cases: one where the measurements are not shared ($C = I$), and a second where the measurements are shared. In the latter case, we use metropolis weights [8] as the combination matrix, and denote it as $C^{\rm met}$. This choice of relative-degree weights for the adaptation matrix and metropolis weights for the combination matrix is based on our previous work [5], though other choices are possible. We can observe that in the case where measurements are not shared ($C = I$), the ATC version of the diffusion LMS algorithm outperforms the CTA version. Note also that there is no penalty in using ATC over CTA, since both require one exchange per iteration. Further improvement can be obtained if measurements are shared between the nodes, at the expense of requiring twice as many communications.

Fig. 6 shows the steady-state EMSE and MSD for a set of diffusion LMS algorithms, and compares with the theoretical results from expressions (33) and (34). The steady-state values are obtained by averaging over 200 experiments and over 50 time samples after convergence. It can be observed that the simulation results match well the theoretical values.

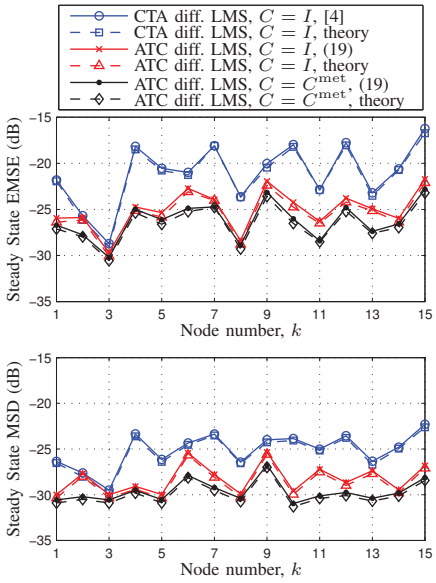

Fig. 6. Steady-state performance of different diffusion LMS algorithms, comparing simulation vs. theory, using expressions (33) and (34).

VII. Conclusions

We presented a general form of diffusion LMS algorithms, and formulated the Adapt-then-Combine and Combine-then-Adapt versions of diffusion LMS, which allow information exchange. Steady-state analysis was presented and matched well with simulation results. It is observed that ATC outperforms the original diffusion LMS from [4] and better performance can be obtained if measurements are also shared.

References

[1] A. H. Sayed and F. Cattivelli, "Distributed adaptive learning mechanisms," Handbook on Array Processing and Sensor Networks, S. Haykin and K. J. Ray Liu, editors, Wiley, NJ, 2009.

[2] A. H. Sayed and C. G. Lopes, "Adaptive processing over distributed networks," *IEICE Trans. on Fund. of Electronics, Communications and Computer Sciences*, vol. E90-A, no. 8, pp. 1504–1510, August 2007.

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

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

[5] F. S. Cattivelli, C. G. Lopes, and A. H. Sayed, "Diffusion recursive least-squares for distributed estimation over adaptive networks," *IEEE Trans. on Signal Processing*, vol. 56, no. 5, pp. 1865–1877, May 2008.

[6] F. S. Cattivelli, C. G. Lopes, and A. H. Sayed, "Diffusion strategies for distributed Kalman filtering: Formulation and performance analysis," in *Proc. Cognitive Information Processing*, Santorini, Greece, June 2008.

[7] F. S. Cattivelli and A. H. Sayed, "Diffusion mechanisms for fixed-point distributed Kalman smoothing," in *Proc. EUSIPCO*, Lausanne, Switzerland, August 2008.

[8] L. Xiao, S. Boyd, and S. Lall, "A space-time diffusion scheme for peer-to-peer least-squares estimation," in *Proc IPSN*, Nashville, TN, April 2006, pp. 168–176.

[9] I. D. Schizas, G. Mateos, and G. B. Giannakis, "Stability analysis of the consensus-based distributed LMS algorithm," in *Proc. ICASSP*, Las Vegas, NV, March 2008, pp. 3289–3292.

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

[11] A. H. Sayed, *Adaptive Filters*, Wiley, NJ, 2008.

[12] A. Nedic and D. Bertsekas, "Incremental subgradient methods for nondifferentiable optimization," *SIAM J. Optim.*, vol. 12, no. 1, pp. 109–138, 2001.