# Diffusion LMS Strategies for Distributed Estimation

Federico S. Cattivelli, *Student Member, IEEE*, and Ali H. Sayed, *Fellow, IEEE*

***Abstract*—We consider the problem of distributed estimation, where a set of nodes is required to collectively estimate some parameter of interest from noisy measurements. The problem is useful in several contexts including wireless and sensor networks, where scalability, robustness, and low power consumption are desirable features. Diffusion cooperation schemes have been shown to provide good performance, robustness to node and link failure, and are amenable to distributed implementations. In this work we focus on diffusion-based adaptive solutions of the LMS type. We motivate and propose new versions of the diffusion LMS algorithm that outperform previous solutions. We provide performance and convergence analysis of the proposed algorithms, together with simulation results comparing with existing techniques. We also discuss optimization schemes to design the diffusion LMS weights.**

***Index Terms*—Adaptive networks, diffusion LMS, diffusion networks, distributed estimation, energy conservation.**

## I. Introduction

We study the problem of distributed estimation, where a set of nodes is required to collectively estimate some parameter of interest from noisy measurements by relying solely on in-network processing. Distributed estimation algorithms are useful in several contexts, including wireless and sensor networks, where scalability, robustness, and low power consumption are desirable. The availability of low-power sensors and processors is generating demand for in-network processing solutions, with applications ranging from precision agriculture to environmental monitoring and transportation.

Thus, consider a set of $N$ nodes distributed over some geographic 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 $w^o$.

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 being non-robust to failure by the fusion center. Moreover, in the context of wireless sensor networks, centralizing all measurements in a single node lacks scalability, and may require large amounts of energy and communication resources [1]. On the other hand, in distributed implementations, every node in the network communicates with a subset of the nodes, and processing is distributed among all nodes in the network. We say that two nodes are connected if they can communicate directly with each other. A node is always connected to itself. 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 $k$. The number of nodes connected to note $k$ is called the degree of node $k$, and is denoted by $n_k$.

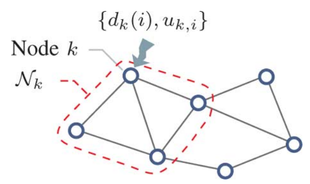


Fig. 1. At time $i$, every node $k$ takes a measurement $\{d_k(i), u_{k,i}\}$.

In-network distributed estimation algorithms have been proposed in the context of distributed adaptive filtering [2]–[4]. These include incremental LMS [3], [5], [6], incremental RLS [3], diffusion LMS [3], [7], and diffusion RLS [8], [9]. Diffusion Kalman filtering and smoothing algorithms were also proposed [10], [11], [32]. Distributed estimation algorithms based on incremental [3], [5], [12] and consensus strategies [13]–[15] have also been proposed. The work [14] proposes a distributed LMS algorithm based on consensus techniques that relies on node hierarchy to reduce communications. The work [15] is related to one of our diffusion algorithms from [8] and is discussed later in Section III-B.

The purpose of this work is to motivate and develop distributed strategies that are able to update their estimates in *real-time* through local interactions and by relying on a single time-scale (namely, the observation time-scale). In this way, the resulting network becomes an adaptive entity on its own right. In comparison to the prior work on diffusion LMS in [7], the current paper develops a formulation for deriving diffusion filters and introduces a more general class of diffusion strategies of which [7] is a special case. We subsequently study the performance of this class of filters and optimize its parameters. Simulation results illustrate the theoretical findings and reveal the enhanced learning abilities of the proposed filters.

This work is organized as follows. In Section II, we formulate the estimation problem and present the global solution. In Section III, we motivate and derive a family of diffusion LMS algorithms for distributed estimation. The algorithms are analyzed in Section IV. In Section V, we discuss different choices of weighting matrices for the diffusion algorithms, including optimization techniques. We conclude our work with simulations of our results in Section VI.

Manuscript received May 20, 2009; accepted September 06, 2009. First published October 06, 2009; current version published February 10, 2010. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Isao Yamada. This work was supported in part by the National Science Foundation under awards ECS-0601266, ECS-0725441, and CCF-094936 . Some preliminary parts of this work appeared in the *Proceedings of Asilomar Conference on Signals, Systems and Computers,* Pacific Grove, CA, October 2008.

The authors are with the Department of Electrical Engineering, University of California, Los Angeles, CA 90095 USA (e-mail: fcattiv@ee.ucla.edu; sayed@ee.ucla.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSP.2009.2033729

## 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, the optimal solution is given by [16], [17]

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

where $R_{u,k} = \mathrm{E}\boldsymbol{u}_{k,i}^*\boldsymbol{u}_{k,i}$ is assumed positive-definite (i.e., $R_{u,k} > 0$) and $R_{du,k} = \mathrm{E}\boldsymbol{d}_k(i)\boldsymbol{u}_{k,i}^*$, and where the operator $^*$ denotes complex conjugate-transposition. Observe that we are allowing the second-order moments $\{R_{u,k}, R_{du,k}\}$ to vary across the nodes.

### B. Local Optimization

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

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

where $\mathbb{1}$ denotes the $N \times 1$ vector with unit entries. In other words, $c_{l,k}$ is zero when nodes $l$ and $k$ are not connected. Moreover, the rows and columns of $C$ add up to one. 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 of node $k$. The local optimal solution is therefore

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

Thus, the local estimate, $w_k^{\text{loc}}$, is based only on covariance data $\{R_{u,l}, R_{du,l}\}_{l\in\mathcal{N}_k}$ that are available to node $k$. In comparison, the global solution, $w^o$ in (2), requires access to the covariance data $\{R_{u,l}, R_{du,l}\}_{l=1,\ldots,N}$ across the entire network. Let

$$\Gamma_k \triangleq \sum_{l \in \mathcal{N}_k} c_{l,k}R_{u,l}.$$

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

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

where mmse is a constant term that does not depend on $w$, and the notation $\|a\|_\Sigma^2 = a^*\Sigma a$ represents a weighted vector norm for any Hermitian $\Sigma > 0$. An interesting question is how to relate the local solutions (5) at all nodes to the global solution (2). Note that because of (3), we can express 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) over $w$ is equivalent to minimizing the following cost function, for any $k \in \{1,\ldots,N\}$:

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

We therefore have an alternative representation of the global cost (1) in terms of the local estimates $\{w_k^{\text{loc}}\}$ across the network.

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

### C. Steepest-Descent Global Solution

To begin with, consider minimizing the cost function (1) using a traditional iterative steepest-descent solution [16], say,

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

where $\mu > 0$ is a step-size parameter and $w_i$ is an estimate for $w^o$ at iteration $i$. Moreover, $\nabla_w J^{\text{glob}}$ denotes the complex gradient of $J^{\text{glob}}(w)$ with respect to $w$, which is given by[1]

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

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

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

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

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

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

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

Algorithm (13) is not distributed: it requires access to data $\{d_k(i), u_{k,i}\}$ across the entire network. It serves as an example of an algorithm that can be run by a fusion center once all $N$ nodes transmit their data to it. In the next section we will consider distributed strategies, which is the main motivation for this work.

[1]A factor of 2 multiplies (10) when the data are real. This factor can be absorbed into the step-size $\mu$ in (10).

## III. Diffusion Adaptive Solutions

As indicated above, the global LMS solution (13) is not distributed. If every node were to run (13), then every node will need to have access to global information (namely, the measurements and regressors of every other node) in order to compute the new estimate $w_i$. One fully distributed solution based on diffusion strategies was proposed in [3], [7] 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 [3], [7] as a special case. In addition, the formulation provides a useful way to motivate and derive diffusion filters by optimizing the cost (8) via incremental strategies.

### A. MSE Minimization

Thus, refer to the equivalent global cost (8). Minimizing this cost at every node $k$ still requires the nodes to have access to global information, namely the local estimates, $w_l^{\text{loc}}$, and the matrices $\Gamma_l$, at the other nodes in the network. In order to facilitate distributed implementations, we now explain how the cost (8) motivates useful distributed algorithms.

To begin with, we replace the covariance matrices $\Gamma_l$ in (8) with constant-diagonal weighting matrices of the form $\Gamma_l = b_{l,k} I_M$, where $b_{l,k}$ is a set of non-negative real coefficients that give different weights to different neighbors, and $I_M$ is the $M \times M$ identity matrix. In particular, we are interested in choices of coefficients such that

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

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

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

Observe that while (15) is an approximation for the global cost (8), it is nevertheless more general than the local cost (4). That is, we used the relation (7) between global and local costs $\{J^{\text{glob}}, J_k^{\text{loc}}\}$ to motivate the approximation (15). Later we will examine the performance of the diffusion algorithms that result from (15) and how close their weight estimates get to $w^o$.

Taking the gradient of (15) we obtain[2]

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

[2]A factor of 2 multiplies (16) when the data are real. This factor can be absorbed into the step-sizes $\mu_k$ and $\nu_k$ in (17).

Thus, we can use (15) to obtain a recursion for the estimate of $w$ at node $k$, denoted by $w_{k,i}$, as we did in the steepest-descent case, say,

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

for some positive step-sizes $\{\mu_k, \nu_k\}$. However, note that the gradient vector in (16) is a sum of two terms, namely

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

Incremental solutions are useful for minimizing sums of convex functions as in (15) (see [5], [6], [12], [18]); they are based on the principle of iterating sequentially over each term, in some pre-defined order. For example, we can accomplish the update (17) in two steps by generating an intermediate estimate $\psi_{k,i}$ as follows:

$$\psi_{k,i} = w_{k,i-1} + \mu_k \sum_{l \in \mathcal{N}_k} c_{l,k}(R_{du,l} - R_{u,l} w_{k,i-1}) \tag{18}$$

$$w_{k,i} = \psi_{k,i} + \nu_k \sum_{l \in \mathcal{N}_k/\{k\}} b_{l,k}(\psi_l - w_{k,i-1}). \tag{19}$$

We now replace $\psi_l$ in (19) by the intermediate estimate that is available at node $l$ at time $i$, namely, $\psi_{l,i}$. We also replace $w_{k,i-1}$ in (19) by the intermediate estimate $\psi_{k,i}$ (as argued in [4], such substitutions lead to enhanced performance since, intuitively, $\psi_{k,i}$ contains more information than $w_{k,i-1}$). This leads to

$$\begin{aligned} \psi_{k,i} &= w_{k,i-1} + \mu_k \sum_{l \in \mathcal{N}_k} c_{l,k}(R_{du,l} - R_{u,l} w_{k,i-1}) \\ w_{k,i} &= \psi_{k,i} + \nu_k \sum_{l \in \mathcal{N}_k/\{k\}} b_{l,k}(\psi_{l,i} - \psi_{k,i}). \end{aligned} \tag{20}$$

Note from the second equation that

$$w_{k,i} = (1 - \nu_k + \nu_k b_{k,k})\psi_{k,i} + \nu_k \sum_{l \in \mathcal{N}_k/\{k\}} b_{l,k} \psi_{l,i} \tag{21}$$

so that if we introduce the coefficients

$$a_{k,k} = 1 - \nu_k + \nu_k b_{k,k} \text{ and } a_{l,k} = \nu_k b_{l,k} \text{ for } l \neq k$$

we obtain

$$\boxed{\begin{aligned} \psi_{k,i} &= w_{k,i-1} + \mu_k \textstyle\sum_{l \in \mathcal{N}_k} c_{l,k}(R_{du,l} - R_{u,l} w_{k,i-1}) \\ w_{k,i} &= \textstyle\sum_{l \in \mathcal{N}_k} a_{l,k} \psi_{l,i} \end{aligned}} \tag{22}$$

where the weighting coefficients $\{a_{l,k}, c_{l,k}\}$ are real, non-negative, and satisfy:

$$\boxed{c_{l,k} = a_{l,k} = 0 \text{ if } l \notin \mathcal{N}_k, \mathbb{1}^T C = \mathbb{1}^T, C\mathbb{1} = \mathbb{1}, \mathbb{1}^T A = \mathbb{1}^T} \tag{23}$$

where $A$ is the $N \times N$ matrix with individual entries $\{a_{l,k}\}$.

### B. Diffusion LMS

Using the instantaneous approximations (12) in (22), we obtain the Adapt-then-Combine (ATC) diffusion LMS algorithm.

**ATC Diffusion LMS**

Start with $\{w_{l,-1}=0\}$ for all $l$. Given non-negative real coefficients $\{c_{l,k},a_{l,k}\}$ satisfying (23), for each time $i\geq 0$ and for each node $k$, repeat:

$$\begin{cases}\psi_{k,i}=w_{k,i-1}+\mu_k\sum_{l\in\mathcal{N}_k}c_{l,k}u_{l,i}^*(d_l(i)-u_{l,i}w_{k,i-1}) \\ \qquad\qquad\qquad\qquad\text{(incremental step)} \\ w_{k,i}=\sum_{l\in\mathcal{N}_k}a_{l,k}\psi_{l,i}\quad\text{(diffusion step)}\end{cases}\tag{24}$$

The ATC diffusion LMS algorithm (24) consists of an incremental update followed by a diffusion update of the form:

$$w_{k,i}=\sum_{l\in\mathcal{N}_k}a_{l,k}\psi_{l,i}$$

which represents a convex combination of estimates from LMS filters fed by spatially distinct data $\{d_k(i),u_{k,i}\}$. In the incremental step in (24), the coefficients $\{c_{l,k}\}$ determine which nodes $l\in\mathcal{N}_k$ should share their measurements $\{d_l(i),u_{l,i}\}$ with node $k$. On the other hand, the coefficients $\{a_{l,k}\}$ in the diffusion step in (24) determine which nodes $l\in\mathcal{N}_k$ should share their intermediate estimates $\{\psi_{l,i}\}$ with node $k$. This is a more general convex combination than the form employed before in the context of conventional adaptive filtering [19]–[21]. We note that when measurements are not exchanged (i.e., when $C=I$), the ATC algorithm (24) becomes similar to the one studied in [15], where noisy links are also considered and analyzed. We further note that this particular ATC mode of cooperation with $C=I$ was originally proposed and studied in [8], [9] in the context of least-squares adaptive networks.

If we reverse the order by which we perform the incremental update (18)–(19), we get

$$\psi_{k,i-1}=w_{k,i-1}+\nu_k\sum_{l\in\mathcal{N}_k/\{k\}}b_{l,k}(\psi_l-w_{k,i-1})\tag{25}$$

$$w_{k,i}=\psi_{k,i-1}+\mu_k\sum_{l\in\mathcal{N}_k}c_{l,k}(R_{du,l}-R_{u,l}w_{k,i-1}).\tag{26}$$

We now replace $\psi_l$ in (25) by $w_{l,i-1}$ and $w_{k,i-1}$ in (26) by $\psi_{k,i-1}$. This leads to

$$\psi_{k,i-1}=\sum_{l\in\mathcal{N}_k}a_{l,k}w_{l,i-1}$$
$$w_{k,i}=\psi_{k,i-1}+\mu_k\sum_{l\in\mathcal{N}_k}c_{l,k}(R_{du,l}-R_{u,l}\psi_{k,i-1}).$$

Using instantaneous approximations for $\{R_{du,l},R_{u,l}\}$ now leads to the Combine-then-Adapt (CTA) diffusion LMS algorithm.

**CTA Diffusion LMS**

Start with $\{w_{l,-1}=0\}$ for all $l$. Given non-negative real coefficients $\{c_{l,k},a_{l,k}\}$ satisfying (23), for each time $i\geq 0$ and for each node $k$, repeat:

$$\begin{cases}\psi_{k,i-1}=\sum_{l\in\mathcal{N}_k}a_{l,k}w_{l,i-1}\qquad\text{(diffusion step)} \\ w_{k,i}=\psi_{k,i-1}+\mu_k\sum_{l\in\mathcal{N}_k}c_{l,k}u_{l,i}^*(d_l(i)-u_{l,i}\psi_{k,i-1}) \\ \qquad\qquad\qquad\qquad\text{(incremental step)}\end{cases}\tag{27}$$

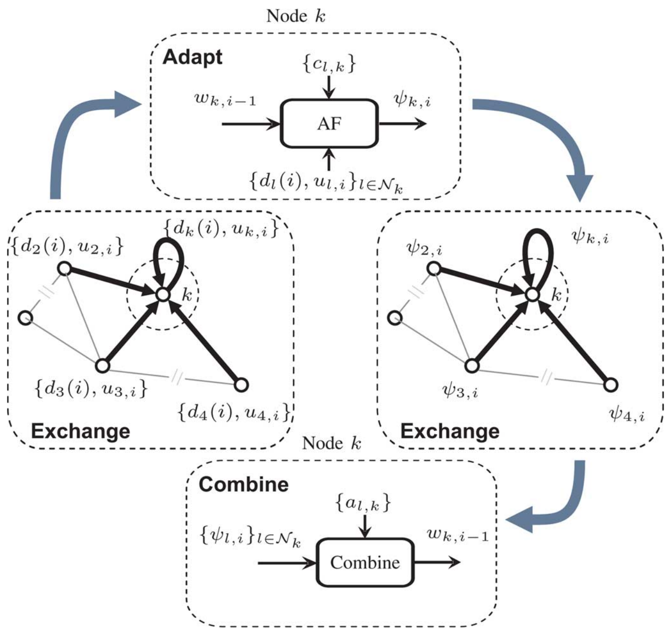


Fig. 2. ATC diffusion strategy.

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 ($C\neq I$). Note that in the former case, the CTA diffusion LMS algorithm (27) reduces to the original diffusion LMS algorithm [7]:

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

## C. Structure

In general, at each iteration $i$, every node $k$ performs a procedure consisting of up to four steps, as summarized in Table I. For example, the ATC algorithm without measurement exchange ($C=I$), consists of three steps. First, every node adapts its current weight 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 $w_{k,i}$. Figs. 2 and 3 show schematically the cooperation strategies for the ATC and CTA algorithms, respectively, for the general case where measurements are shared ($C\neq I$).

When $C=I$, the ATC algorithm has the same processing and communication complexity as the CTA algorithm. In Section VI we will see by simulation that the ATC version outperforms the CTA version, and therefore also outperforms diffusion LMS [7], without penalty.

## IV. Performance Analysis

In this section, we analyze the diffusion LMS algorithms in their ATC (24) and CTA (27) forms. In what follows we view the estimates $w_{k,i}$ as realizations of random processes $\boldsymbol{w}_{k,i}$, and

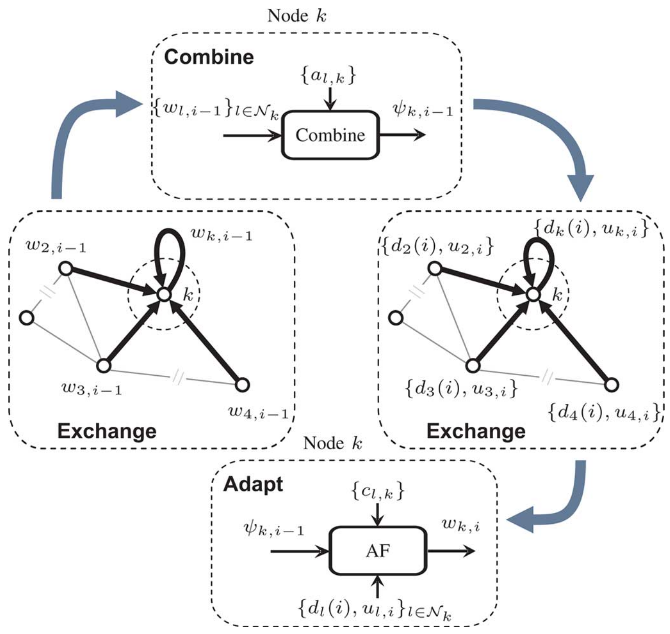


Fig. 3. CTA diffusion strategy.

TABLE I
STEPS FOR ATC AND CTA ALGORITHMS WITH AND WITHOUT MEASUREMENT SHARING

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

analyze the performance of the algorithms in terms of their expected behavior. Instead of analyzing each algorithm separately, we formulate a general algorithmic form that includes the ATC and CTA algorithms as special cases. Subsequently, we 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[3]

$$\begin{cases} \boldsymbol{\phi}_{k,i-1} = \sum_{l=1}^{N} p_{1,l,k}\boldsymbol{w}_{l,i-1} & \text{(diffusion I)} \\ \boldsymbol{\psi}_{k,i} = \boldsymbol{\phi}_{k,i-1} + \mu_k \sum_{l=1}^{N} s_{l,k}\boldsymbol{u}_{l,i}^*[\boldsymbol{d}_l(i) - \boldsymbol{u}_{l,i}\boldsymbol{\phi}_{k,i-1}] & \\ & \text{(incremental)} \\ \boldsymbol{w}_{k,i} = \sum_{l=1}^{N} p_{2,l,k}\boldsymbol{\psi}_{l,i} & \text{(diffusion II)} \end{cases} \tag{28}$$

where the coefficients $\{p_{1,l,k}\}$, $\{s_{l,k}\}$ and $\{p_{2,l,k}\}$ are generic non-negative real coefficients corresponding to the $\{l,k\}$ entries of matrices $P_1$, $S$ and $P_2$, respectively, and satisfy

$$\mathbb{1}^T P_1 = \mathbb{1}^T, \mathbb{1}^T S = \mathbb{1}^T, \mathbb{1}^T P_2 = \mathbb{1}^T. \tag{29}$$

[3]We are now using boldface symbols to denote random quantities.

TABLE II
DIFFERENT CHOICES OF MATRICES $P_1 S$ AND $P_2$ RESULT IN DIFFERENT LMS ALGORITHMS

| Algorithm | $P_1$ | $P_2$ | $S$ |
|---|---|---|---|
| No cooperation LMS | $I$ | $I$ | $I$ |
| Global LMS (13) | $I$ | $I$ | $\frac{1}{N}\mathbb{1}\mathbb{1}^T$ |
| ATC diffusion LMS (24) | $I$ | $A$ | $C$ |
| CTA diffusion LMS (27) | $A$ | $I$ | $C$ |
| Diffusion LMS [7] | $A$ | $I$ | $I$ |

Equation (28) can be specialized to the ATC diffusion LMS algorithm (24) by choosing $P_1 = I$, $S = C$ and $P_2 = A$, to the CTA diffusion LMS algorithm (27) by choosing $P_1 = A$, $S = C$ and $P_2 = I$, and to the diffusion LMS algorithm from [7] by choosing $P_1 = A$ and $P_2 = S = I$. Equation (28) can also be specialized to the global LMS algorithm (13) by choosing $P_1 = P_2 = I$ and $S = \mathbb{1}\mathbb{1}^T/N$, and also to the case where nodes do not cooperate and run LMS recursions individually, by selecting $P_1 = S = P_2 = I$. Table II summarizes the choices of the matrices $P_1$, $S$ and $P_2$ required to obtain different LMS algorithms. Notice that a constraint of the form $S\mathbb{1} = \mathbb{1}$ is not required at this point, since it has no implications on the subsequent analysis.

## A. Modeling Assumptions

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

$$\boldsymbol{d}_k(i) = \boldsymbol{u}_{k,i}w^o + \boldsymbol{v}_k(i) \tag{30}$$

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 (30) are customary in the adaptive filtering literature [16] 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).

Using (28), we define the error quantities

$$\tilde{\boldsymbol{w}}_{k,i} = w^o - \boldsymbol{w}_{k,i}\tilde{\boldsymbol{\psi}}_{k,i} = w^o - \boldsymbol{\psi}_{k,i}\tilde{\boldsymbol{\phi}}_{k,i-1} = w^o - \boldsymbol{\phi}_{k,i-1}$$

and the global vectors:

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

We also introduce the diagonal matrix

$$\mathcal{M} = \text{diag}\{\mu_1 I_M, \ldots, \mu_N I_M\} \tag{31}$$

and the extended weighting matrices

$$\mathcal{P}_1 = P_1 \otimes I_M \mathcal{P}_2 = P_2 \otimes I_M \mathcal{S} = S \otimes I_M \tag{32}$$

where $\otimes$ denotes the Kronecker product operation. We further introduce the following matrices:

$$\boldsymbol{\mathcal{D}}_i = \text{diag}\left\{\sum_{l=1}^{N} s_{l,1}\boldsymbol{u}_{l,i}^*\boldsymbol{u}_{l,i}, \ldots, \sum_{l=1}^{N} s_{l,N}\boldsymbol{u}_{l,i}^*\boldsymbol{u}_{l,i}\right\}$$
$$\boldsymbol{\mathcal{G}}_i = \mathcal{S}^T \text{col}\{\boldsymbol{u}_{1,i}^*\boldsymbol{v}_1(i), \ldots, \boldsymbol{u}_{N,i}^*\boldsymbol{v}_N(i)\}.$$

Then we have

$$\begin{aligned}\tilde{\boldsymbol{\phi}}_{i-1} &= \mathcal{P}_1^T \tilde{\boldsymbol{w}}_{i-1}\\ \tilde{\boldsymbol{\psi}}_i &= \tilde{\boldsymbol{\phi}}_{i-1} - \mathcal{M}[\boldsymbol{\mathcal{D}}_i \tilde{\boldsymbol{\phi}}_{i-1} + \boldsymbol{\mathcal{G}}_i]\\ \tilde{\boldsymbol{w}}_i &= \mathcal{P}_2^T \tilde{\boldsymbol{\psi}}_i\end{aligned}$$

or, equivalently,

$$\tilde{\boldsymbol{w}}_i = \mathcal{P}_2^T[I - \mathcal{M}\boldsymbol{\mathcal{D}}_i]\mathcal{P}_1^T \tilde{\boldsymbol{w}}_{i-1} - \mathcal{P}_2^T \mathcal{M}\boldsymbol{\mathcal{G}}_i. \tag{33}$$

Moreover, let

$$\begin{aligned}\mathcal{D} &\triangleq \mathrm{E}\boldsymbol{\mathcal{D}}_i\\ &= \mathrm{diag}\left\{\sum_{l=1}^N s_{l,1}R_{u,l}, \ldots, \sum_{l=1}^N s_{l,N}R_{u,l}\right\} &(34)\\ \mathcal{G} &\triangleq \mathrm{E}[\boldsymbol{\mathcal{G}}_i \boldsymbol{\mathcal{G}}_i^*]\\ &= \mathcal{S}^T \mathrm{diag}\{\sigma_{v,1}^2 R_{u,1}, \ldots, \sigma_{v,N}^2 R_{u,N}\}\mathcal{S}. &(35)\end{aligned}$$

We now introduce the following *independence assumption*:

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

Assumption 1 allows us to consider the regressors $\{\boldsymbol{u}_{k,i}\}$ independent of $\{\boldsymbol{w}_{l,j}\}$ for all $l$ and for $j \le i-1$. Though not true in general, this type of assumption is customary in the context of adaptive filters, as it simplifies the analysis. Some results in the literature indicate that the performance analysis obtained using this assumption is reasonably close to the actual performance for sufficiently small step-sizes [16], [17].

## B. Mean Stability

From Assumption 1, $\boldsymbol{\mathcal{D}}_i$ is independent of $\tilde{\boldsymbol{w}}_{i-1}$, which depends on regressors up to time $i-1$. Taking expectation of (33) yields

$$\mathrm{E}\tilde{\boldsymbol{w}}_i = \mathcal{P}_2^T[I - \mathcal{M}\mathcal{D}]\mathcal{P}_1^T \mathrm{E}\tilde{\boldsymbol{w}}_{i-1}. \tag{36}$$

We say that a square matrix $X$ is stable if it satisfies $X^i \to 0$ as $i \to \infty$. A known result in linear algebra states that a matrix is stable if, and only if, all its eigenvalues lie inside the unit circle. We need the following lemma to proceed.

*Lemma 1:* Let $P_1$, $P_2$, and $X$ denote arbitrary $M \times M$ matrices, where $P_1$ and $P_2$ have real, non-negative entries, with columns adding up to one, i.e., $\mathbb{1}^T P_1 = \mathbb{1}^T$, $\mathbb{1}^T P_2 = \mathbb{1}^T$. Then, the matrix $Y = P_2^T X P_1^T$ is stable for *any* choice of $P_1$ and $P_2$ if, and only if, $X$ is stable.

*Proof:* See Appendix I. □

The following theorem guarantees asymptotic unbiasedness of the general diffusion LMS filter (28). We use the notation $\lambda_{\max}(X)$ to denote the maximum eigenvalue of a Hermitian matrix $X$.

*Theorem 1:* Assume the data model (30) and Assumption 1 hold. Then the general diffusion LMS algorithm (28) is asymptotically unbiased for any initial condition and any choice of matrices $P_1$ and $P_2$ satisfying (29) if, and only if,

$$0 < \mu_k < \frac{2}{\lambda_{\max}\left(\sum_{l=1}^N s_{l,k}R_{u,l}\right)}, k = 1, \ldots, N. \tag{37}$$

*Proof:* In view of Lemma 1 and (36), we have asymptotic unbiasedness if, and only if, the matrix $I - \mathcal{M}\mathcal{D}$ is stable. Thus, we require $I - \mu_k \sum_{l\in\mathcal{N}_k} s_{l,k}R_{u,l}$ to be stable for all $k$, which, by using $\sum_{l\in\mathcal{N}_k} s_{l,k}R_{u,l} > 0$, is equivalent to $|1 - \lambda_{\max}(\mu_k \sum_{l\in\mathcal{N}_k} s_{l,k}R_{u,l})| < 1$ and (37) follows. □

Notice that the maximum eigenvalue of a Hermitian matrix $X$ is convex in the elements of $X$ [22, p. 118], and from the convexity of the coefficients $s_{l,k}$, we have from (37)

$$\lambda_{\max}\left(\sum_{l=1}^N s_{l,k}R_{u,l}\right) \le \sum_{l=1}^N s_{l,k}\lambda_{\max}(R_{u,l}) \le \max_{l=1,\ldots,N} \lambda_{\max}(R_{u,l}).$$

Thus, we note that a sufficient condition for unbiasedness is $0 < \mu_k < 2/\max_{l=1,\ldots,N}\lambda_{\max}(R_{u,l})$.

## C. Variance Relation

We now study the mean-square performance of the general diffusion filter (28). To do so, we resort to the energy conservation analysis of [16], [17], [23], [31]. Evaluating the weighted norm of $\tilde{\boldsymbol{w}}_i$ in (33) we obtain

$$\begin{aligned}\mathrm{E}\|\tilde{\boldsymbol{w}}_i\|_\Sigma^2 = \mathrm{E}\|\tilde{\boldsymbol{w}}_{i-1}\|^2_{\mathcal{P}_1(I-\boldsymbol{\mathcal{D}}_i\mathcal{M})\mathcal{P}_2\Sigma\mathcal{P}_2^T(I-\mathcal{M}\boldsymbol{\mathcal{D}}_i)\mathcal{P}_1^T}\\ +\mathrm{E}[\boldsymbol{\mathcal{G}}_i^*\mathcal{M}\mathcal{P}_2\Sigma\mathcal{P}_2^T\mathcal{M}\boldsymbol{\mathcal{G}}_i] \qquad (38)\end{aligned}$$

where $\Sigma$ is any Hermitian positive-definite matrix that we are free to choose. Using Assumption 1, we can rewrite (38) as a variance relation in the form

$$\begin{aligned}\mathrm{E}\|\tilde{\boldsymbol{w}}_i\|_\Sigma^2 &= \mathrm{E}\|\tilde{\boldsymbol{w}}_{i-1}\|_{\Sigma'}^2 + \mathrm{Tr}[\Sigma\mathcal{P}_2^T\mathcal{M}\mathcal{G}\mathcal{M}\mathcal{P}_2]\\ \Sigma' &= \mathcal{P}_1\mathcal{P}_2\Sigma\mathcal{P}_2^T\mathcal{P}_1^T - \mathcal{P}_1\mathcal{D}\mathcal{M}\mathcal{P}_2\Sigma\mathcal{P}_2^T\mathcal{P}_1^T\\ &\quad - \mathcal{P}_1\mathcal{P}_2\Sigma\mathcal{P}_2^T\mathcal{M}\mathcal{D}\mathcal{P}_1^T\\ &\quad + \mathrm{E}(\mathcal{P}_1\boldsymbol{\mathcal{D}}_i\mathcal{M}\mathcal{P}_2\Sigma\mathcal{P}_2^T\mathcal{M}\boldsymbol{\mathcal{D}}_i\mathcal{P}_1^T). \qquad (39)\end{aligned}$$

Let

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

where the vec( · ) notation stacks the columns of its matrix argument on top of each other and $\mathrm{vec}^{-1}(\cdot)$ is the inverse operation. We will also use the notation $\|\tilde{w}\|_\sigma^2$ to denote $\|\tilde{w}\|_\Sigma^2$. Using the Kronecker product property

$$\mathrm{vec}(U\Sigma V) = (V^T \otimes U)\mathrm{vec}(\Sigma) \tag{40}$$

and the fact that the expectation and vectorization operators commute, we can vectorize expression (39) for $\Sigma'$ as follows:

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

where the matrix $F$ is given by

$$\boxed{\begin{array}{c}F=(\mathcal{P}_1\otimes\mathcal{P}_1)\{I-I\otimes(\mathcal{D}\mathcal{M})-(\mathcal{D}^T\mathcal{M})\otimes I+\\ \mathrm{E}[(\mathcal{D}_i^T\mathcal{M})\otimes(\mathcal{D}_i\mathcal{M})]\}(\mathcal{P}_2\otimes\mathcal{P}_2)\end{array}}. \quad (41)$$

Using the property $\mathrm{Tr}(\Sigma X)=\mathrm{vec}(X^T)^T\sigma$ we can rewrite (39) as follows:

$$\mathrm{E}\|\tilde{\boldsymbol{w}}_i\|_\sigma^2=\mathrm{E}\|\tilde{\boldsymbol{w}}_{i-1}\|_{F\sigma}^2+[\mathrm{vec}(\mathcal{P}_2^T\mathcal{M}\mathcal{G}^T\mathcal{M}\mathcal{P}_2)]^T\sigma. \quad (42)$$

## D. Special Cases

We now specialize our results to different cases, namely the case of small step-sizes, and the case where the regressors are Gaussian.

*1) Small Step-Sizes:* Note that the step-sizes $\{\mu_k, k=1,\dots,N\}$ only influence (42) through the matrix $\mathcal{M}$. When these step-sizes are sufficiently small, the rightmost term of (41) can be neglected, since it will depend on $\{\mu_k^2\}$, while all other terms will depend at most on $\{\mu_k\}$. In this case, we get (43), shown at the bottom of the page.

*2) Gaussian Regressors:* Assume now that the regressors are circular complex-valued Gaussian zero-mean random vectors. In this case, we can evaluate the right-most term in (41) in terms of the fourth moment of a Gaussian variable. We show in Appendix II that in this case, $F$ is given by (44), shown at the bottom of the page, where $\beta=1$ if the regressors are complex, $\beta=2$ if the regressors are real, $r_m=\mathrm{vec}(R_{u_m})$, and the remaining quantities are given by

$$\mathcal{S}_m=\mathrm{diag}\{s_{m,1}I_M,\dots,s_{m,N}I_M\}=\mathrm{diag}(e_m^TS)\otimes I_M \quad (45)$$

and

$$\mathcal{L}_m=I_N\otimes\begin{bmatrix}I_N\otimes e_1\\ \vdots\\ I_N\otimes e_M\end{bmatrix}\otimes r_m^* \quad (46)$$

with $e_m$ being column vectors with a unit entry at position $m$ and zeros elsewhere.

## E. Mean-Square Performance

*1) Steady-State Performance:* In steady-state, and assuming that the matrix $I-F$ is invertible (see Section IV-E-3), we have from (42)

$$\mathrm{E}\|\tilde{\boldsymbol{w}}_\infty\|_{(I-F)\sigma}^2=[\mathrm{vec}(\mathcal{P}_2^T\mathcal{M}\mathcal{G}^T\mathcal{M}\mathcal{P}_2)]^T\sigma. \quad (47)$$

The steady-state MSD and EMSE at node $k$ are defined as [16]

$$\mathrm{MSD}_k=\mathrm{E}\|\boldsymbol{w}_{k,i}-w^o\|^2 \text{ and } \mathrm{EMSE}_k=\mathrm{E}|\boldsymbol{u}_{k,i}\tilde{\boldsymbol{w}}_{k,i-1}|^2.$$

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. We denote the vectorized version of this matrix by $m_k$, i.e.,

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

Then, choosing $\sigma=(I-F)^{-1}m_k$, the MSD becomes (48), shown at the bottom of the page.

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

$$r_k=\mathrm{vec}(\mathrm{diag}(e_k)\otimes R_{u_k})$$

and $\sigma=(I-F)^{-1}r_k$, the EMSE becomes (49), shown at the bottom of the page. The *network* MSD and EMSE are defined as the average MSD and EMSE, respectively, across all nodes in the network:

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

$$\boxed{F=(\mathcal{P}_1\otimes\mathcal{P}_1)\{I-I\otimes(\mathcal{D}\mathcal{M})-(\mathcal{D}^T\mathcal{M})\otimes I\}(\mathcal{P}_2\otimes\mathcal{P}_2)}. \quad (43)$$

$$\boxed{\begin{array}{l}F=(\mathcal{P}_1\otimes\mathcal{P}_1)\Big\{(I-\mathcal{D}^T\mathcal{M})\otimes(I-\mathcal{D}\mathcal{M})+\\ \qquad\beta\sum_{m=1}^N\left\{\mathcal{S}_m^T\otimes[\mathcal{S}_m(I_N\otimes R_{u_m})]\right\}\mathcal{L}_m(\mathcal{M}\otimes\mathcal{M})\Big\}(\mathcal{P}_2\otimes\mathcal{P}_2)\end{array}}. \quad (44)$$

$$\boxed{\mathrm{MSD}_k=\mathrm{E}\|\tilde{\boldsymbol{w}}_\infty\|_{m_k}^2=[\mathrm{vec}(\mathcal{P}_2^T\mathcal{M}\mathcal{G}^T\mathcal{M}\mathcal{P}_2)]^T(I-F)^{-1}m_k}. \quad (48)$$

$$\boxed{\mathrm{EMSE}_k=\mathrm{E}\|\tilde{\boldsymbol{w}}_\infty\|_{r_k}^2=[\mathrm{vec}(\mathcal{P}_2^T\mathcal{M}\mathcal{G}^T\mathcal{M}\mathcal{P}_2)]^T(I-F)^{-1}r_k}. \quad (49)$$

*2) Transient Analysis:* Starting from (42), we define $r \triangleq \mathrm{vec}(\mathcal{P}_2^T \mathcal{M}\mathcal{G}^T\mathcal{M}\mathcal{P}_2)$ and we get

$$\begin{aligned}\mathrm{E}\|\tilde{\boldsymbol{w}}_i\|_\sigma^2 &= \mathrm{E}\|\tilde{\boldsymbol{w}}_{i-1}\|_{F\sigma}^2 + r^T\sigma \\ &= \mathrm{E}\|\tilde{\boldsymbol{w}}_{-1}\|_{F^{i+1}\sigma}^2 + r^T\sum_{j=0}^{i} F^j\sigma \\ &= \mathrm{E}\|\tilde{\boldsymbol{w}}_{i-1}\|_\sigma^2 - \|w^o\|_{F^i(I-F)\sigma}^2 + r^T F^i\sigma \qquad (50)\end{aligned}$$

where we assumed $w_{k,-1} = 0$ for all $k$. Thus, by taking $\sigma = r_k$ or $\sigma = m_k$, we can compute, recursively, the instantaneous EMSE and MSD, respectively, for every node in the network and for every time instant. Equation (50) is also useful to compute the steady-state EMSE and MSD when $N$ and $M$ are large, and inverting the matrix $I - F$ (of size $M^2N^2 \times M^2N^2$) is not computationally tractable.

*3) Convergence Analysis:* We now provide conditions on the step-sizes for mean-square convergence. Let $L = MN$ and let $p(x)$ denote the characteristic polynomial of the $L^2 \times L^2$ matrix $F$, where

$$p(x) = \det(xI - F) = x^{L^2} + p_{L^2-1}x^{L^2-1} + \cdots + p_0.$$

Then we have

$$F^{L^2} = -p_0 - p_1F - \cdots - p_{L^2-1}F^{L^2-1}.$$

Let $q = \mathrm{vec}(I_L)$, and selecting $\sigma = F^jq$, $j = 0,\ldots,L^2-1$, we have

$$\underbrace{\begin{bmatrix}\mathrm{E}\|\tilde{\boldsymbol{w}}_i\|^2 \\ \mathrm{E}\|\tilde{\boldsymbol{w}}_i\|_{Fq}^2 \\ \vdots \\ \mathrm{E}\|\tilde{\boldsymbol{w}}_i\|_{F^{L^2-1}q}^2\end{bmatrix}}_{\mathcal{W}_i} = \underbrace{\begin{bmatrix}0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & & & \ddots & \\ 0 & 0 & 0 & \dots & 1 \\ -p_0 & -p_1 & -p_2 & \dots & -p_{L^2-1}\end{bmatrix}}_{\mathcal{F}}$$
$$\times \underbrace{\begin{bmatrix}\mathrm{E}\|\tilde{\boldsymbol{w}}_{i-1}\|^2 \\ \mathrm{E}\|\tilde{\boldsymbol{w}}_{i-1}\|_{Fq}^2 \\ \vdots \\ \mathrm{E}\|\tilde{\boldsymbol{w}}_{i-1}\|_{F^{L^2-1}q}^2\end{bmatrix}}_{\mathcal{W}_{i-1}} + \underbrace{\begin{bmatrix}r^Tq \\ r^TFq \\ \vdots \\ r^TF^{L^2-1}q\end{bmatrix}}_{\mathcal{Y}}.$$

Matrix $\mathcal{F}$ is in companion form, and it is known that its eigenvalues are the roots of $p(x)$, which are also the eigenvalues of $F$. Therefore, a necessary and sufficient condition for the convergence of $\mathcal{W}_i$ is that $F$ is a stable matrix, or equivalently, that all the eigenvalues of $F$ are inside the unit circle. In view of Lemma 1 and (41), and noting that the matrices $\mathcal{P}_2 \otimes \mathcal{P}_2$ and $\mathcal{P}_1 \otimes \mathcal{P}_1$ have real, non-negative entries and their columns add up to one, we conclude that $F$ will be stable for any matrices $P_2$ and $P_1$ if, and only if, the step-sizes $\{\mu_k\}$ are such that the following matrix is stable:

$$\bar{F} = I - I \otimes (\mathcal{D}\mathcal{M}) - (\mathcal{D}^T\mathcal{M}) \otimes I + \mathrm{E}[(\boldsymbol{\mathcal{D}}_i^T\mathcal{M}) \otimes (\boldsymbol{\mathcal{D}}_i\mathcal{M})]. \qquad (51)$$

Notice that the stability of $F$ guarantees that $I - F$ will be invertible, and therefore the steady-state expressions (48) and (49) are well defined. We conclude that in order to guarantee mean-square convergence, the step-sizes $\{\mu_k\}$ must be chosen such that $\bar{F}$ in (51) is stable. This condition can be checked in practice when the right-most term in (51) can be computed, as in the Gaussian case, or when the step-sizes are small such that this term can be neglected.

A simpler, approximate condition can be obtained for small step-sizes. In this case, we have

$$\bar{F} \approx (I - \mathcal{D}^T\mathcal{M}) \otimes (I - \mathcal{D}\mathcal{M})$$

which is stable if, and only if, $I - \mathcal{D}^T\mathcal{M}$ is stable. This condition is the same as the condition for mean stability (37) and can be easily checked.

We summarize our results in the following theorem.

*Theorem 2:* Assume the data model (30) and Assumption 1 hold. Then the general diffusion LMS algorithm (28) will converge in the mean-square sense if the step-sizes $\{\mu_k\}$ are such that the matrix $\bar{F}$ in (51) is stable. In this case, the steady-state MSD and EMSE are given by (48) and (49) respectively, where the matrix $F$ is given by (41). For the case of Gaussian regressors, $F$ simplifies to (44) and $\bar{F}$ to

$$\begin{aligned}\bar{F} = (I - \mathcal{D}^T\mathcal{M}) &\otimes (I - \mathcal{D}\mathcal{M}) \\ &+\beta\sum_{m=1}^{N}\left\{\mathcal{S}_m^T \otimes [\mathcal{S}_m(I_N \otimes R_{u_m})]\right\}\mathcal{L}_m(\mathcal{M}\otimes\mathcal{M}).\end{aligned}$$

### *F. Comparison of ATC and CTA Algorithms*

In Section VI, we will show through simulation examples that the ATC diffusion LMS algorithm (24) tends to outperform the CTA version (27). In this section we provide some insight to explain this behavior, and show that it is always true when both algorithms use the same weights, and the diffusion matrix satisfies certain conditions.

From (24) and (27), we have

$$\begin{aligned}\text{ATC}: \; w_{k,i} &= \sum_{l\in\mathcal{N}_k} a_{l,k}\left\{w_{l,i-1} + \mu_l\sum_{m\in\mathcal{N}_l} c_{m,l}u_{m,i}^*[d_m(i) - u_{m,i}w_{l,i-1}]\right\} \\ \text{CTA}: \; w_{k,i} &= \sum_{l\in\mathcal{N}_k} a_{l,k}\left\{w_{l,i-1} + \mu_k\sum_{m\in\mathcal{N}_k} c_{m,k}u_{m,i}^*[d_m(i) - u_{m,i}w_{l,i-1}]\right\}.\end{aligned}$$

The above expressions show that in order to compute the new estimate $w_{k,i}$, the CTA algorithm uses the measurements $\{d_k(i), u_{k,i}\}$ from all nodes $m$ in the neighborhood of node $k$, while the ATC version uses the measurements from all nodes $m$ in the neighborhood of nodes $l$, which are neighbors of $k$. Thus, the ATC version effectively uses data from nodes that are two hops away in every iteration, while the CTA version uses data from nodes that are one hop away. For the special case $C = I$, the CTA algorithm uses the measurements available at node $k$ only, while the ATC version uses measurements available at the neighbors of node $k$.

We now present a more formal comparison. Starting from (38), and introducing the small step-size approximation, we have

$$\mathrm{E}\|\tilde{\boldsymbol{w}}_i\|^2_\Sigma = \mathrm{E}\|\tilde{\boldsymbol{w}}_{i-1}\|^2_{\mathcal{P}_1(I-\mathcal{D}\mathcal{M})\mathcal{P}_2\Sigma\mathcal{P}_2^T\times(I-\mathcal{M}\mathcal{D})\mathcal{P}_1^T} + \mathrm{Tr}[\Sigma\mathcal{P}_2^T\mathcal{M}\mathcal{G}\mathcal{M}\mathcal{P}_2].$$

If the step-size is chosen such that $I-\mathcal{D}\mathcal{M}$ is stable, we get

$$\mathrm{MSD}^{\text{network}} = \frac{1}{N}\mathrm{E}\|\tilde{\boldsymbol{w}}_\infty\|^2 \approx \frac{1}{N}\sum_{j=0}^{\infty}\mathrm{Tr}[X^jY(X^*)^j] \quad (52)$$

where we defined

$$X \triangleq \mathcal{P}_2^T[I-\mathcal{M}\mathcal{D}]\mathcal{P}_1^T \quad Y \triangleq \mathcal{P}_2^T\mathcal{M}\mathcal{G}\mathcal{M}\mathcal{P}_2.$$

For the ATC version, we set $\mathcal{P}_1 = 0$ and $\mathcal{P}_2 = \mathcal{P}$, while for the CTA version, we set $\mathcal{P}_2 = 0$ and $\mathcal{P}_1 = \mathcal{P}$, where $\mathcal{P} = A \otimes I_N$, and the diffusion matrix $A$ is such that the spectral radius of $AA^T$ is one, i.e., $\rho(AA^T) = 1$. This condition is not satisfied for general diffusion matrices, but is satisfied by all doubly stochastic diffusion matrices ($A\mathbb{1} = \mathbb{1}$), and therefore by symmetric diffusion matrices. Let

$$H_j \triangleq [(I-\mathcal{M}\mathcal{D})\mathcal{P}^T]^j\mathcal{M}\mathcal{G}\mathcal{M}[\mathcal{P}(I-\mathcal{D}\mathcal{M})]^j.$$

Then we can write (52) as

$$\mathrm{MSD}^{\text{network}}_{\text{ATC}} = \frac{1}{N}\sum_{j=0}^{\infty}\mathrm{Tr}[\mathcal{P}^T H_j \mathcal{P}]$$

$$\mathrm{MSD}^{\text{network}}_{\text{CTA}} = \frac{1}{N}\sum_{j=0}^{\infty}\mathrm{Tr}[H_j]$$

and therefore,

$$\mathrm{MSD}^{\text{network}}_{\text{CTA}} - \mathrm{MSD}^{\text{network}}_{\text{ATC}} = \frac{1}{N}\sum_{j=0}^{\infty}\mathrm{Tr}[(I-\mathcal{P}\mathcal{P}^T)H_j] \geq 0.$$

The above inequality follows from the fact that $H_j$ is positive semi-definite, and that when $AA^T$ has unit spectral radius, so does $\mathcal{P}\mathcal{P}^T$, and the matrix $I-\mathcal{P}\mathcal{P}^T$ is also positive semi-definite. Therefore, the network MSD of the CTA algorithm is always larger than or equal to the network MSD of the ATC version, when they use the same diffusion matrix $A$ satisfying $\rho(AA^T) = 1$. When $A$ does not satisfy this condition, the above statement is not true in general, though it is usually the case that ATC outperforms CTA, as we shall see in Section VI.

In a similar fashion, we now show that under some conditions, a diffusion LMS algorithm with information exchange ($C \neq I$) will always outperform the same algorithm with no information exchange ($C = I$). Intuitively, this behavior is to be expected, since measurement exchange allows every node to have access to more data. More formally, we will now assume that every node has the same regressor covariance, $R_{u,k} = R_u$, and the same noise variance, $\sigma^2_{v,k} = \sigma^2_v$. Under these assumptions, $\mathcal{D}$ does not depend on $\mathcal{S}$, and $\mathcal{G}$ can be written as $[I_N \otimes (\sigma_v R_u^{1/2})]\mathcal{S}^T\mathcal{S}[I_N \otimes (\sigma_v R_u^{*/2})]$. Selecting $\mathcal{S} = C \otimes I_M$, where $C$ satisfies $\mathbb{1}^T C = \mathbb{1}^T$ and $C\mathbb{1} = \mathbb{1}$, we have from (52) that

$$\mathrm{MSD}^{\text{network}}_{C=I} - \mathrm{MSD}^{\text{network}}_{C\neq I} = \mathrm{Tr}[H(I-\mathcal{S}^T\mathcal{S})] \geq 0$$

where $H$ is some positive semi-definite matrix that does not depend on $\mathcal{S}$. Again, the above inequality follows from the fact that $I - \mathcal{S}^T\mathcal{S}$ is positive semi-definite. We conclude that the algorithm that uses measurement exchange will have equal or lower network MSD than the algorithm without measurement exchange, under the above assumptions.

TABLE III
POSSIBLE COMBINATION RULES. IN ALL CASES, $a_{l,k} = 0$ IF $l \notin \mathcal{N}_k$, AND $a_{k,k}$ IS CHOSEN SUCH THAT $\sum_{l=1}^N a_{l,k} = 1$ FOR ALL $k$

| **Name** | **Rule** ($l \in \mathcal{N}_k$, $l \neq k$) | **Reference** |
|---|---|---|
| Uniform | $a_{l,k} = 1/n_k$ | [25] |
| Laplacian | $a_{l,k} = 1/n_{\max}$ | [26], [27] |
| Maximum degree | $a_{l,k} = 1/N$ | [28] |
| Metropolis | $a_{l,k} = 1/\max(n_k, n_l)$ | [27] |
| Relative degree | $a_{l,k} = n_l/\left(\sum_{m\in\mathcal{N}_k} n_m\right)$ | [9] |
| Relative degree-variance | $a_{l,k} = \dfrac{n_l\sigma^2_{v,l}}{\left(\sum_{m\in\mathcal{N}_k} n_m\sigma^2_{v,l}\right)}$ | This work |

## V. CHOICE OF WEIGHTING MATRICES

It is clear from the analysis of Section IV that the performance of the diffusion LMS algorithms depends heavily on the choice of the weighting matrices $P_2$, $P_1$, and $S$ in Table II. There are several ways by which the combination weights can be selected. Some popular choices of weights from graph theory, average consensus, and diffusion adaptive networks are shown in Table III, where $n_k$ denotes the degree of node $k$. Although these rules are shown for the coefficients $\{a_{l,k}\}$, they can also be applied to the coefficients $\{c_{l,k}\}$, provided $C\mathbb{1} = \mathbb{1}$, as in the Metropolis, Laplacian or Maximum-degree rules.

One rule that has been employed previously is the relative-degree rule [9], which gives more weight to nodes that are better connected. However, if a node that is well connected has high noise variance, this rule may not be a good one as shown in [24]. We can consider a new rule denoted "relative degree-variance," where every neighbor is weighted proportionally according to its degree times its noise variance as shown in Table III. This rule assumes that the noise variance is known, or can be estimated from the data. As we shall see in Section VI-B, this rule has improved performance when some of the nodes are very noisy.

In the above rules, the combination weights $\{a_{l,k}\}$ are largely dictated by the sizes of the neighborhoods (or by the node degrees). When the neighborhoods vary with time, the degrees will also vary. However, for all practical purposes, these combination schemes are not adaptive in the sense that the schemes do not learn which nodes are more or less reliable so that the weights can be adjusted accordingly.

An adaptive combination rule along these lines can be motivated based on the analysis results of [19]. The combination weights can be adjusted adaptively so that the network can respond to node conditions and assign smaller weights to nodes that are subject to higher noise levels. This strategy was used in [7], while a newer, more powerful adaptive strategy was proposed in [24].

## A. Weight Optimization

We now consider optimal choices for the weighting matrices in the general diffusion LMS algorithm (28). Our objective in this section is to find matrices $P_2$, $P_1$ and $S$ that minimize the steady-state average network MSD, i.e.,

$$\underset{P_1, P_2, S}{\text{minimize}} \ \frac{1}{N}\mathrm{E}\|\tilde{\boldsymbol{w}}_\infty\|^2 \text{ subject to (29).}$$

To accomplish this, we will use expression (52), which is valid for small step-sizes. Equation (52) is neither convex nor quasi-convex in $P_1$ or $P_2$ in general. However, it is convex in $S$ when $P_1$ and $P_2$ are fixed, since in this case $\mathrm{Tr}[X^j Y (X^*)^j]$ is convex for every $j$. Still, we can attempt to minimize (52) in $P_1$ or $P_2$ using a general nonlinear solver, and the following approximation:

$$\mathrm{MSD}^{\text{network}} \approx \frac{1}{N}\sum_{j=0}^{J}\mathrm{Tr}[X^j Y (X^*)^j] \qquad (53)$$

where $J$ is some positive integer ($J = 10 \sim 50$ is usually sufficient in simulations). This technique of minimizing (53) yields very good results in practice as we shall see in Section VI.

The average MSD of (52) can be made convex in $P_2$ for $P_1 = I$ and fixed $S$, and convex in $P_1$ for $P_2 = I$ and fixed $S$, if we introduce the following three assumptions: $\mu_k = \mu$ for every $k$, $R_{u,k} = R_u$ for every $k$, and the weighting matrices satisfy $P_2 = P_2^T$ and $P_1 = P_1^T$. This can be shown by noticing that under these assumptions, we have

$$\begin{aligned}\mathrm{E}\|\tilde{\boldsymbol{w}}_\infty\|^2 &= \mathrm{Tr}\left[\sum_{j=0}^{\infty} X^{2j} Y\right] = \mathrm{Tr}[(I - X^2)^{-1} Y] \\ &= \frac{1}{2}\{\mathrm{Tr}[(I - X)^{-1} Y] + \mathrm{Tr}[(I + X)^{-1} Y]\}\end{aligned}$$

and that $\mathrm{Tr}(Y X^{-1})$ is convex in $X$ for $X > 0$ and $Y > 0$ [22, p. 116]. Unfortunately, the restriction that the diffusion matrices $P_2$ and $P_1$ be symmetric reduces the performance of the diffusion LMS algorithms, and it is usually the case that optimized symmetric matrices are outperformed by non-symmetric matrices.

## VI. Simulation Results

In order to illustrate the adaptive network performance, we present a simulation example in Figs. 4–6. Fig. 4 depicts the network topology with $N = 30$ nodes, together with the network statistical profile showing how the signal and noise power vary across the nodes. The regressors have size $M = 2$, 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 shows the learning curves for different diffusion LMS algorithms in terms of EMSE and MSD. The simulations use a value of $\mu = 0.05$, and the results are averaged over 100 independent experiments. For the diffusion algorithms, relative-degree weights [9] are used for the diffusion matrix. For the adaptation 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 [13] for the adaptation matrix, and denote it as $C^{\text{met}}$. This choice of relative-degree weights for the diffusion matrix and metropolis weights for the adaptation matrix is based on our previous work [9], though other choices are possible. We 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

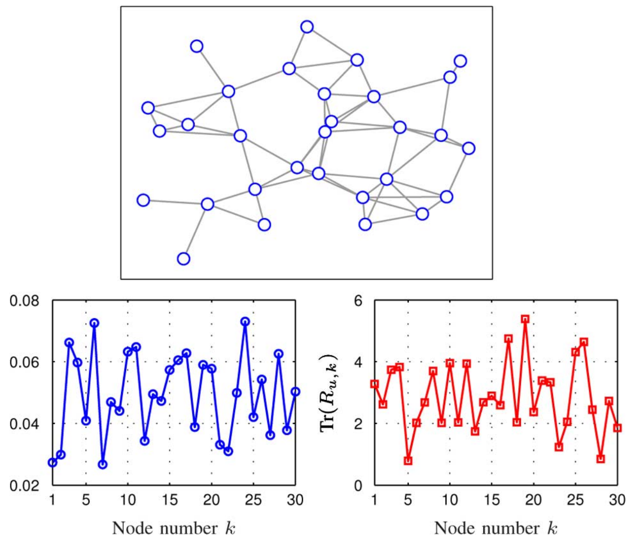


Fig. 4. Network topology (top), noise variances $\sigma_{v,k}^2$ (bottom, left) and trace of regressor covariances $\mathrm{Tr}(R_{u,k})$ (bottom, right) for $N = 30$ nodes.

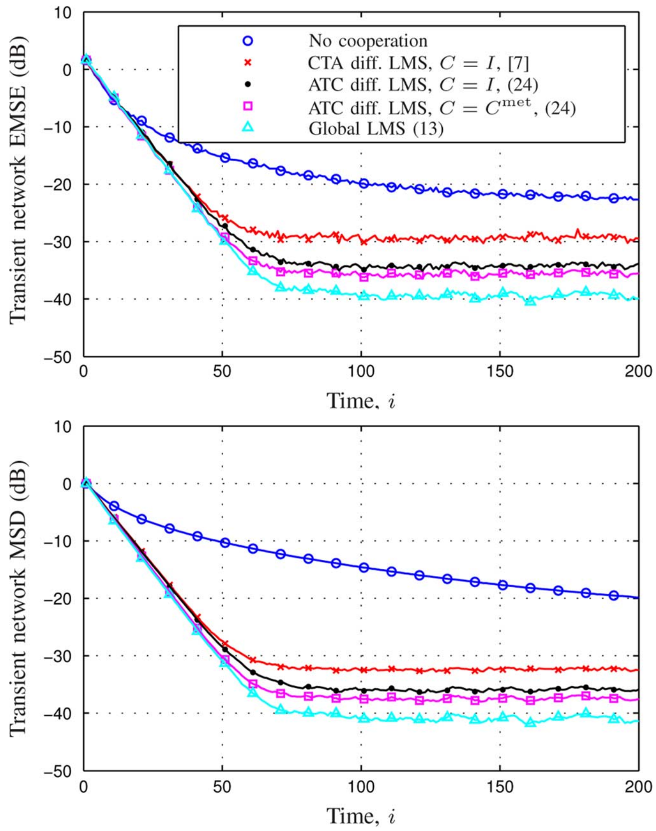


Fig. 5. Transient network EMSE (top) and MSD (bottom) for LMS without cooperation, CTA and ATC diffusion LMS, and global LMS.

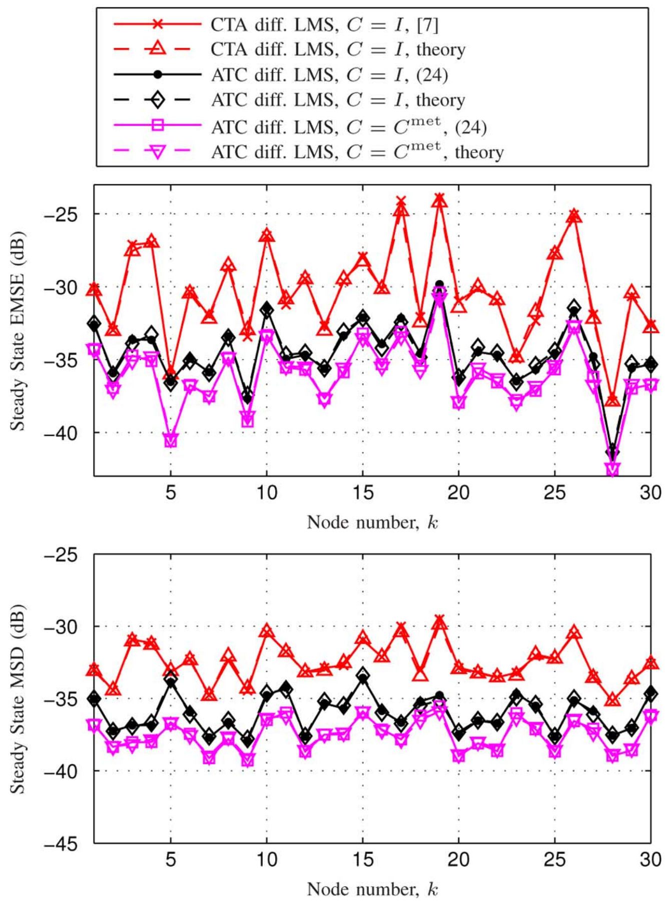


Fig. 6. Steady-state performance of different diffusion LMS algorithms, comparing simulation versus theory, using expressions (48) and (49).

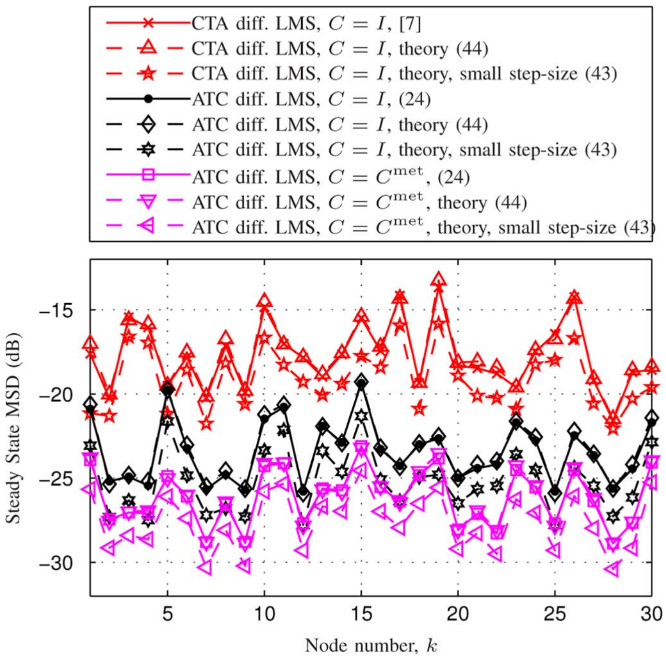


Fig. 7. Steady-state performance of different diffusion LMS algorithms, comparing simulation versus theory, using expressions (48) and (49), where $F$ is computed using (43) and (44).

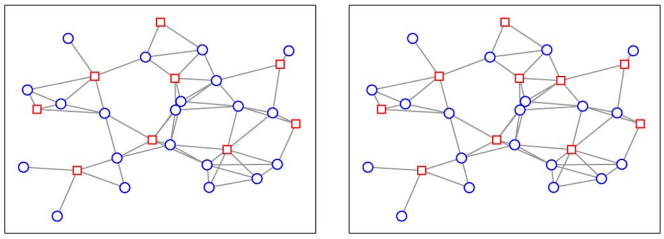


Fig. 8. Network topology, where red squares denote leader nodes (left) and bridge nodes (right).

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. In our simulations, we checked that condition (37) was met, and that the step-sizes $\mu_k$ were such that (51) is stable.

Fig. 6 shows the steady-state EMSE and MSD for a set of diffusion LMS algorithms, and compares with the theoretical results from expressions (44), (48), and (49). The steady-state values are obtained by averaging over 100 experiments and over 40 time samples after convergence. It can be observed that the simulation results match well the theoretical values.

If we compute the theoretical steady-state behavior, by using the small step-size approximation of $F$, given by (43), the difference is imperceptible for this value of the step-size. For a step-size $\mu = 0.3$, this difference becomes more prominent, as depicted in Fig. 7.

## A. Hierarchical Cooperation

We now compare our proposed algorithms with two hierarchical LMS algorithms, namely our recently proposed multi-level diffusion [29], and the consensus-based D-LMS algorithm of [14]. In contrast to the diffusion algorithms proposed in this work, where all nodes perform the same type of processing, hierarchical algorithms introduce node hierarchy, and different nodes perform different types of operations. Hierarchical algorithms may provide increased performance at the expense of reduced robustness, and higher complexity required to establish and maintain the hierarchies. The multi-level diffusion algorithm is based on the diffusion LMS algorithm (24) proposed here (see [29] for details). The algorithm designates a subset of the nodes, called *leader* nodes, which perform a special type of processing. The nodes that are not leaders run a conventional diffusion LMS algorithm such as (24). At the same time, leader nodes form a second network, where a second diffusion LMS algorithm such as (24) is ran. This results in increased speed and performance as we shall see, at the expense of requiring multi-hop communications.

In order to simulate and compare different hierarchical algorithms, we clustered the network as described in [29], and the result is shown in the left plot of Fig. 8, where the leaders are denoted by red squares. In order to compare with the consensus-based D-LMS algorithm of [14], we also clustered the nodes as described in [14]. The resulting “bridge” nodes are shown as red squares in the right plot of Fig. 8.

Fig. 9 shows the learning curves for different LMS algorithms, including the (non-hierarchical) ATC and CTA diffusion LMS algorithms with $C = I$, multi-level diffusion LMS from

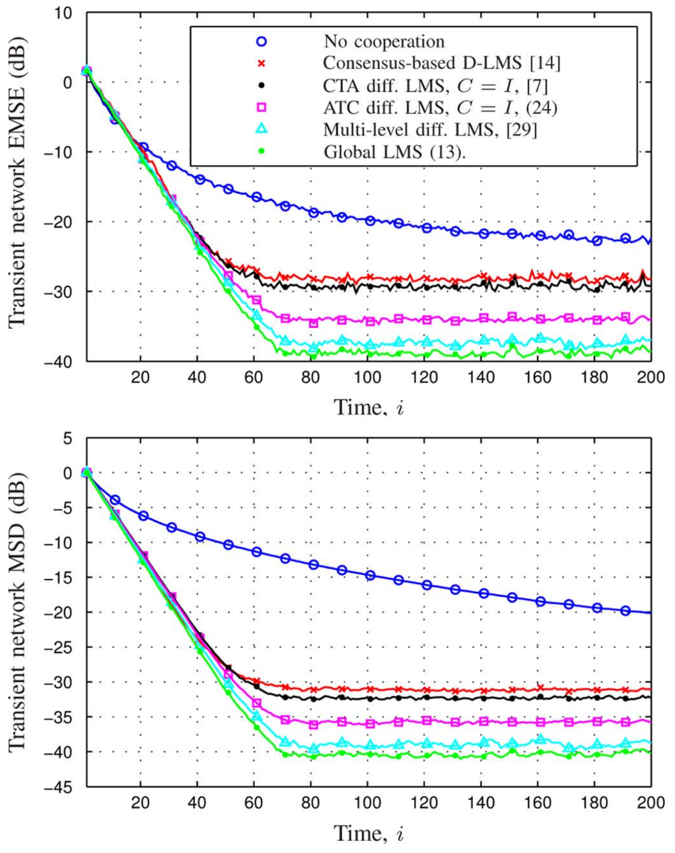


Fig. 9. Transient network EMSE (top) and MSD (bottom) for different LMS algorithms.

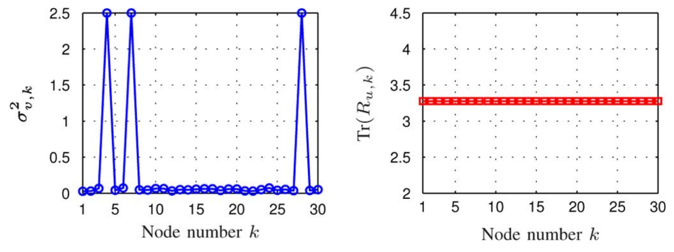


Fig. 10. Noise variances $\sigma^2_{v,k}$ (left) and trace of regressor covariances $\mathrm{Tr}(R_{u,k})$ (right) for $N = 30$ nodes, where the noise variance of three nodes is increased by a factor of 50.

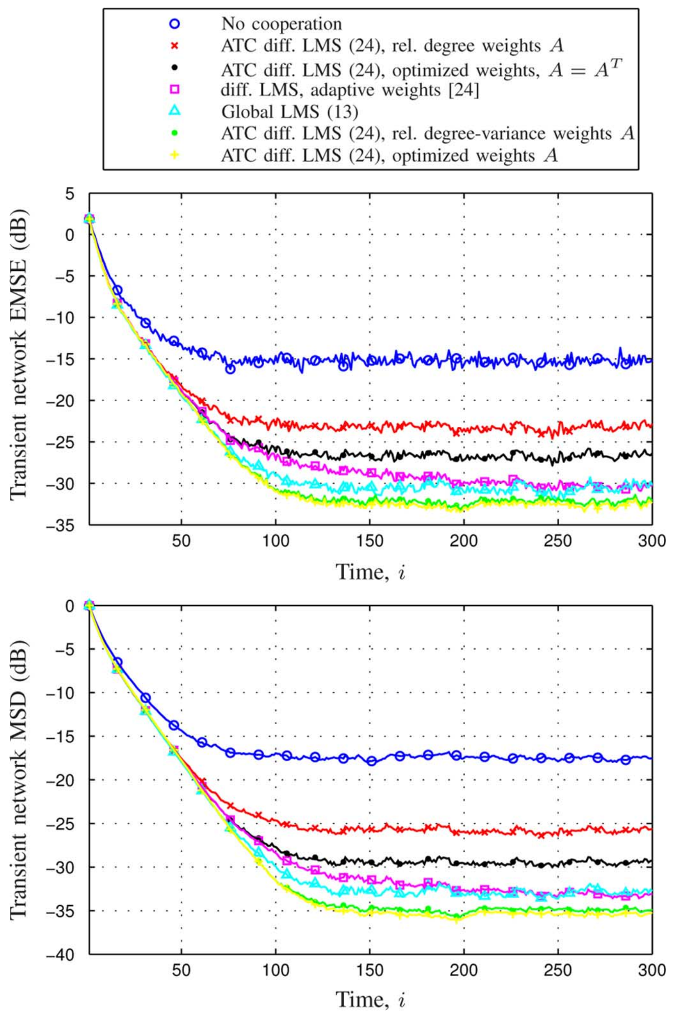


Fig. 11. Transient network EMSE (top) and MSD (bottom) for different LMS algorithms. All diffusion LMS algorithms use $C = I$ (no measurement sharing).

[29] and the consensus-based D-LMS from [14]. The step-size for all cases is $\mu = 0.05$. Since the consensus-based D-LMS algorithm [14] uses a step-size of value $2\bar{\mu}$, we used $\bar{\mu} = 0.025$ for this algorithm. The remaining parameters are $\epsilon = 1/\sqrt{2\bar{\mu}}$ and $c_k = 1/|\mathcal{B}_k|$, where $|\mathcal{B}_k|$ denotes the number of bridge nodes connected to node $k$. For multi-level diffusion, we use relative-degree weights for the diffusion step, and metropolis weights for the adaptation step. We observe that among the hierarchical algorithms, multi-level diffusion outperforms all other solutions. The performance of the consensus-based D-LMS algorithm of [14] is comparable to the performance of the CTA diffusion LMS of [7] (or (27) with $C = I$). This does not contradict the results of [14], where only MSE curves were provided, and here we are showing EMSE and MSD curves. Nonetheless, the proposed ATC version of diffusion LMS with $C = I$ outperforms [14], although D-LMS may provide savings in communications as discussed in [14].

## B. Weight Optimization

We now show a simulation where the diffusion weights $A$ are either optimized or adapted. The results obtained when nodes have similar noise and regressor statistics across space are similar to the results obtained by using an ad hoc choice for the diffusion weights (using, for instance, relative-degree weights). However, as noted in [24], weight adaptation provides a considerable improvement in performance when the noise statistics across the nodes are very different. In particular, we will consider the case where all regressors across nodes have the same covariance matrix, but the noise variances $\sigma^2_{v,k}$ of three nodes are fifty times higher than the average of the remaining nodes. The noise profile is shown in Fig. 10.

Fig. 11 shows the resulting learning curves for different algorithms, including LMS without cooperation, ATC diffusion LMS with $C = I$, the adaptive weighting technique of [24], ATC diffusion LMS with $C = I$ and $A$ optimized according to (53), ATC diffusion LMS with $C = I$ and relative-degree-variance weights from Table III, and the global LMS solution. The optimized weights were computed using the MATLAB Optimization Toolbox. We observe that the optimized solution provides the best performance, and is even better than the global steepest descent solution (13). The relative-degree-variance rule achieves a performance close to the optimal one, and the adaptive algorithm of [24] also achieves performance better than the global solution (13). The reason is that both the relative-degree-variance rule and the optimized weights assume knowledge of the noise variances at each node $\sigma_{v,k}^2$, while the adaptive algorithm of [24] learns this noise variance on the fly. In contrast, the global steepest descent solution (13) does not utilize or attempt to estimate the noise variance.

## VII. Conclusion

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 mean and mean-square analysis was presented and matched well with simulation results. Convergence of the algorithms was also analyzed. It is observed that the ATC version of diffusion LMS outperforms the original diffusion LMS from [7] and better performance can be obtained if measurements are also shared. We also showed how to formulate the weight optimization problem in such a way that it can be approximately solved using a nonlinear solver, and discussed restrictions required to formulate it as a convex problem. We showed that the optimized weights outperform other solutions, including the global steepest-descent solution, when some of the nodes are extremely noisy.

## Appendix I
## Proof of Lemma 1

Assume that $X$ is stable. A known result in matrix theory [30, p. 554], [16] states that for every square matrix $X$, and every $\epsilon > 0$, there exists a submultiplicative matrix norm[4] $\|\cdot\|_\rho$ such that $\|X\|_\rho \leq \rho(X)+\epsilon$, where $\rho(X)$ is the spectral radius of $X$. Since $X$ is stable, $\rho(X) < 1$, and we can choose $\epsilon > 0$ such that $\rho(X)+\epsilon = \lambda < 1$ and $\|X\|_\rho \leq \lambda < 1$. Then we have

$$\begin{aligned}\|Y^i\|_\rho &\leq \| \left(P_2^T\right)^i \|_\rho \| \left(P_1^T\right)^i \|_\rho \|X^i\|_\rho \\ &\leq \lambda^i \| \left(P_2^T\right)^i \|_\rho \| \left(P_1^T\right)^i \|_\rho.\end{aligned}$$

Now, since both $P_2^i$ and $P_1^i$ have non-negative entries with columns that add up to one, they are element-wise bounded by unity. This implies that their Frobenius norms are bounded, and by the equivalence of norms, so is any norm, and in particular $\|(P_2^T)^i\|_\rho$ and $\|(P_1^T)^i\|_\rho$. Thus, we have

$$\lim_{i\to\infty} \|Y^i\|_\rho = 0$$

so $Y^i$ converges to the zero matrix for large $i$. Therefore, $Y$ is stable.

Conversely, assume $X$ is not stable. Then, for the choice $P_2 = P_1 = I$, $Y$ is not stable. □

[4]A submultiplicative matrix norm satisfies $\|AB\| \leq \|A\| \cdot \|B\|$.

## Appendix II
## Derivation of (44)

In view of (45), we can rewrite $\mathcal{D}$ in (34) as

$$\mathcal{D} = \sum_m \mathcal{S}_m (I \otimes R_{u_m}).$$

Assume now that the regressors are circular complex-valued Gaussian zero-mean random vectors. Then, for any Hermitian matrix $W$ it holds [16, p. 46]

$$\mathrm{E}[\boldsymbol{u}_{k,i}^* \boldsymbol{u}_{k,i} W \boldsymbol{u}_{l,i}^* \boldsymbol{u}_{l,i}] = R_{u_k} W R_{u_l} + \beta \delta_{kl} R_{u_k} \mathrm{Tr}(W R_{u_k})$$

where $\beta = 1$ if the regressors are complex, and $\beta = 2$ if the regressors are real. Consider the matrix $K = \mathrm{E}\boldsymbol{\mathcal{D}}_i^* Q \boldsymbol{\mathcal{D}}_i$ where $Q = \mathcal{M}\mathcal{P}_2 \Sigma \mathcal{P}_2^T \mathcal{M}$. Its $\{k,l\}$ block is given by

$$\begin{aligned}K_{k,l} &= \sum_m \sum_n s_{m,k} s_{n,l} \mathrm{E}\boldsymbol{u}_{m,i}^* \boldsymbol{u}_{m,i} Q_{k,l} \boldsymbol{u}_{n,i}^* \boldsymbol{u}_{n,i} \\ &= \sum_m \sum_n s_{m,k} s_{n,l} [R_{u_m} Q_{k,l} R_{u_n} \\ &\quad + \beta \delta_{mn} R_{u_m} \mathrm{Tr}(Q_{k,l} R_{u_m})]\end{aligned}$$

and we can write

$$K = \mathcal{D}^* Q \mathcal{D} + \beta \sum_m \mathcal{S}_m (I_N \otimes R_{u_m}) Z_m \mathcal{S}_m$$

where

$$Z_m = \begin{bmatrix} I_M r_m^* q_{11} & \cdots & I_M r_m^* q_{1N} \\ \vdots & \ddots & \vdots \\ I_M r_m^* q_{N1} & \cdots & I_M r_m^* q_{NN} \end{bmatrix},$$
$$q_{k,l} = \mathrm{vec}(Q_{k,l}),\ r_m = \mathrm{vec}(R_{u_m}).$$

Taking the vector operator of the above matrix $K$, and using property (40) obtain

$$\begin{aligned}\mathrm{vec}(K) &= (\mathcal{D}^T \otimes \mathcal{D}^*)\mathrm{vec}(Q) \\ &\quad + \beta \sum_m \mathcal{S}_m^T \otimes [\mathcal{S}_m (I_N \otimes R_{u_m})] \mathrm{vec}(Z_m) \\ \mathrm{vec}(Z_m) &= \mathcal{L}_m \mathrm{vec}(Q) \qquad (54)\end{aligned}$$

where $\mathcal{L}_m$ is given by (46). Equation (44) follows.

## Acknowledgment

The authors would like to thank the anonymous reviewer for a suggestion relevant to Section IV-F.

## References

[1] D. Estrin, L. Girod, G. Pottie, and M. Srivastava, “Instrumenting the world with wireless sensor networks,” in *Proc. Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Salt Lake City, UT, May 2001, vol. 4, pp. 2033–2036.

[2] C. G. Lopes and A. H. Sayed, "Distributed processing over adaptive networks," in *Proc. Adaptive Sensor Array Processing Workshop*, Lexington, MA, Jun. 2006.

[3] A. H. Sayed and C. G. Lopes, "Adaptive processing over distributed networks," *IEICE Trans. Fund. Electron., Commun. Comput. Sci.*, vol. E90-A, no. 8, pp. 1504–1510, Aug. 2007.

[4] A. H. Sayed and F. Cattivelli, "Distributed adaptive learning mechanisms," in *Handbook on Array Processing and Sensor Networks*, S. Haykin and K. J. Ray Liu, Eds. New York: Wiley, 2009.

[5] C. G. Lopes and A. H. Sayed, "Distributed adaptive incremental strategies: Formulation and performance analysis," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Toulouse, France, May 2006, vol. 3, pp. 584–587.

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

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

[8] F. S. Cattivelli, C. G. Lopes, and A. H. Sayed, "A diffusion RLS scheme for distributed estimation over adaptive networks," in *Proc. IEEE SPAWC*, Helsinki, Finland, Jun. 2007, pp. 1–5.

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

[10] F. S. Cattivelli, C. G. Lopes, and A. H. Sayed, "Diffusion strategies for distributed Kalman filtering: Formulation and performance analysis," in *Proc. Workshop on Cognitive Inf. Process.*, Santorini, Greece, Jun. 2008.

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

[12] S. S. Ram, A. Nedic, and V. V. Veeravalli, "Stochastic incremental gradient descent for estimation in sensor networks," in *Proc. Asilomar Conf. Signals, Syst., Comput.*, Pacific Grove, CA, 2007, pp. 582–586.

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

[14] I. D. Schizas, G. Mateos, and G. B. Giannakis, "Distributed LMS for consensus-based in-network adaptive processing," *IEEE Trans. Signal Process.*, vol. 8, no. 6, pp. 2365–2381, Jun. 2009.

[15] S. S. Stankovic, M. S. Stankovic, and D. S. Stipanovic, "Decentralized parameter estimation by consensus based stochastic approximation," in *Proc. 46th Conf. Decision Control*, New Orleans, LA, Dec. 2007, pp. 1535–1540.

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

[17] A. H. Sayed, *Adaptive Filters*. New York: Wiley, 2008.

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

[19] J. Arenas-Garcia, A. Figueiras-Vidal, and A. H. Sayed, "Mean-square performance of a convex combination of two adaptive filters," *IEEE Trans. Signal Process.*, vol. 54, no. 3, pp. 1078–1090, Mar. 2006.

[20] Y. Zhang and J. A. Chambers, "Convex combination of adaptive filters for a variable tap-length LMS algorithm," *IEEE Signal Process. Lett.*, vol. 13, no. 10, pp. 628–631, Oct. 2006.

[21] M. Silva and V. H. Nascimento, "Convex combination of adaptive filters with different tracking capabilities," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Honolulu, HI, Apr. 2007, vol. 3, pp. 925–928.

[22] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[23] T. Y. Al-Naffouri and A. H. Sayed, "Transient analysis of data-normalized adaptive filters," *IEEE Trans. Signal Process.*, vol. 51, no. 3, pp. 639–652, Mar. 2003.

[24] N. Takahashi, I. Yamada, and A. H. Sayed, "Diffusion least-mean squares with adaptive combiners," in *Proc. Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Taipei, Taiwan, Apr. 2009, pp. 2845–2848.

[25] V. D. Blondel, J. M. Hendrickx, A. Olshevsky, and J. N. Tsitsiklis, "Convergence in multiagent coordination, consensus, and flocking," in *Proc. Joint 44th IEEE Conf. Decision Control Eur. Control Conf. (CDC-ECC)*, Seville, Spain, Dec. 2005, pp. 2996–3000.

[26] D. S. Scherber and H. C. Papadopoulos, "Locally constructed algorithms for distributed computations in ad hoc networks," in *Proc. Information Processing Sensor Networks (IPSN)*, Berkeley, CA, Apr. 2004, pp. 11–19.

[27] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Syst. Control Lett.*, vol. 53, no. 1, pp. 65–78, Sep. 2004.

[28] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *Proc. IPSN*, Los Angeles, CA, Apr. 2005, pp. 63–70.

[29] F. S. Cattivelli and A. H. Sayed, "Multi-level diffusion adaptive networks," in *Proc. Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Taipei, Taiwan, Apr. 2009, pp. 2789–2792.

[30] T. Kailath, A. H. Sayed, and B. Hassibi, *Linear Estimation*. Englewood Cliffs, NJ: Prentice-Hall, 2000.

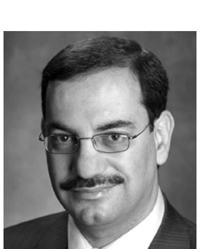
[31] N. R. Yousef and A. H. Sayed, "A unified approach to the steady-state and tracking analysis of adaptive filters," *IEEE Trans. Signal Process.*, vol. 49, no. 2, pp. 314–324, Feb. 2001.

[32] F. S. Cattivelli and A. H. Sayed, "Diffusion strategies for distributed Kalman filtering and smoothing," *IEEE Trans. Autom. Control*, 2010, to be published.

**Federico S. Cattivelli** (S'03) received the B.S. degree in electronics engineering from Universidad ORT, Uruguay, in 2004 and the M.S. degree from the University of California, Los Angeles (UCLA), in 2006. He is currently pursuing the Ph.D. degree in electrical engineering at the same university.

He is a member of the Adaptive Systems Laboratory at UCLA. His research interests are in signal processing, adaptive filtering, estimation theory, and optimization. His current research activities are focused in the areas of distributed estimation and detection over adaptive networks. Previous research activities include Doppler shift frequency estimation for space applications (in collaboration with the NASA Jet Propulsion Laboratory), and state estimation and modeling in biomedical systems (in collaboration with the UCLA School of Medicine).

Mr. Cattivelli was a recipient of the UCLA Graduate Division and UCLA Henry Samueli SEAS Fellowship in 2005 and 2006.



**Ali H. Sayed** (F'01) is Professor and Chairman of Electrical Engineering at the University of California, Los Angeles (UCLA), where he directs the Adaptive Systems Laboratory (www.ee.ucla.edu/asl).

Dr. Sayed is a Fellow of the IEEE and has served on the editorial boards of several journals, including as Editor-in-Chief of the IEEE TRANSACTIONS ON SIGNAL PROCESSING. He has published widely in the areas of adaptive systems, statistical signal processing, estimation theory, adaptive and cognitive networks, and signal processing for communications. He has authored or coauthored several books, including *Adaptive Filters* (Wiley, 2008), *Fundamentals of Adaptive Filtering* (Wiley, 2003), and *Linear Estimation* (Prentice-Hall, 2000). His work has received several recognitions, including the 1996 IEEE Donald G. Fink Award, the 2003 Kuwait Prize, the 2005 Terman Award, and the 2002 Best Paper Award and 2005 Young Author Best Paper Award from the IEEE Signal Processing Society. He has served as a 2005 Distinguished Lecturer of the same society and as General Chairman of ICASSP 2008. He is serving as the Vice-President at Publications of the IEEE Signal Processing Society.