Diffusion LMS Over Multitask Networks

Jie Chen, *Member, IEEE*, Cédric Richard, *Senior Member, IEEE*, and Ali H. Sayed, *Fellow, IEEE*

***Abstract*—The diffusion LMS algorithm has been extensively studied in recent years. This efficient strategy allows to address distributed optimization problems over networks in the case where nodes have to collaboratively estimate a single parameter vector. Nevertheless, there are several problems in practice that are multitask-oriented in the sense that the optimum parameter vector may not be the same for every node. This brings up the issue of studying the performance of the diffusion LMS algorithm when it is run, either intentionally or unintentionally, in a multitask environment. In this paper, we conduct a theoretical analysis on the stochastic behavior of diffusion LMS in the case where the single-task hypothesis is violated. We analyze the competing factors that influence the performance of diffusion LMS in the multitask environment, and which allow the algorithm to continue to deliver performance superior to non-cooperative strategies in some useful circumstances. We also propose an unsupervised clustering strategy that allows each node to select, via adaptive adjustments of combination weights, the neighboring nodes with which it can collaborate to estimate a common parameter vector. Simulations are presented to illustrate the theoretical results, and to demonstrate the efficiency of the proposed clustering strategy.**

***Index Terms*—Adaptive clustering, collaborative processing, diffusion strategy, distributed optimization, multitask learning, stochastic performance.**

I. Introduction

DISTRIBUTED adaptive estimation is an attractive and challenging problem that allows a collection of interconnected nodes to perform preassigned tasks from streaming measurements, such as parameter estimation. Although centralized strategies may benefit from information collected throughout a network, in most cases, distributed strategies are more robust to solve inference problems in a collaborative and autonomous manner [2].

Most recent efforts in the study of distributed estimation problems have focused on scenarios where the network is employed to collectively estimate a single parameter vector. Several strategies have been proposed for this purpose for sequential data processing over networks, including consensus strategies [3]–[10], incremental strategies [11]–[15], and diffusion strategies [16], [17]. Diffusion strategies are particularly attractive due to their enhanced adaptation performance and wider stability ranges when constant step-sizes are used to enable continuous learning [18]. For this reason, we focus on this class of strategies in the remainder of the article. These strategies estimate a common parameter vector by minimizing, in a distributed manner, a global criterion that aggregates neighborhood cost functions. Nodes exchange information locally and cooperate only with their neighbors, without the need for sharing and requiring any global information. The resulting networks benefit from the temporal and spatial diversity of the data and end up being endowed with powerful learning and tracking abilities [18], [19]. The performance of the corresponding adaptive networks have been extensively studied in the literature, under favorable and unfavorable conditions such as model non-stationarities and imperfect communication [20], [21]. This framework has also been extended by considering more general cost functions and data models [19], [22]–[24], by incorporating additional regularizers [25]–[27], or by expanding its use to other scenarios [28]–[31].

The working hypothesis for these earlier studies on diffusion LMS strategies is that the nodes cooperate with each other to estimate a single parameter vector. We shall refer to problems of this type as *single-task* problems. However, many problems of interest happen to be *multitask*-oriented in the sense that there are multiple optimum parameter vectors to be inferred simultaneously and in a collaborative manner. The multitask learning problem is relevant in several machine learning formulations [32]–[34]. In the distributed estimation context, which is the focus of this work, there exist many applications where either agents are subject to data measurements arising from different models, or they are sensing data that varies over the spatial domain. Only a handful of works have considered problem formulations that deal multitask scenarios. A brief summary follows.

For instance, if different groups of agents within a network happen to be tracking different moving targets, then all agents within the same cluster would be interested in estimating the same parameter vector (say, the vector that describes the location of their target). If the targets are moving in formation, then their location vectors would be related to each other and, therefore, cooperation among clusters would be beneficial. In a second example [35], we consider agents engaged in cooperative spectrum sensing over cognitive radio networks. These networks involve two types of users: primary users and secondary users. Secondary users are allowed to detect and occupy temporarily unused spectral bands provided that they do not cause interference to primary users. Therefore, secondary users need to estimate the aggregated spectrum transmitted by all primary

Manuscript received April 28, 2014; revised August 23, 2014 and November 29, 2014; accepted February 12, 2015. Date of publication March 13, 2015; date of current version April 20, 2015. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Taraq Al-Naffouri. The work of C. Richard was partly supported by the Agence Nationale pour la Recherche, France (ODISSEE project, ANR-13-ASTR-0030). The work of A. H. Sayed was supported in part by NSF grants CCF-1011918 and ECCS-1407712. A short and preliminary version of this work appears in the Proceedings of the International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), St. Martin, France, December 2013.

J. Chen and C. Richard are with the Université of Nice Sophia-Antipolis, CNRS, 06108 Nice, France (e-mail: dr.jie.chen@ieee.org; cedric.richard@unice.fr).

A. H. Sayed is with the department of electrical engineering, University of California, Los Angeles, CA 9005-1594 USA (e-mail: 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.2015.2412918

users, as well as local interference profiles. This multitask estimation problem requires cooperation between nodes because noncooperative strategies would lead to local spectral profiles that are subject to hidden node effects [35]. In another example, a network may be deployed to estimate the spatially-varying temperature profile over a certain region, where the parameters that determine how the temperature varies across the agents may be space-dependent [36]. In another example, the works [37], [38] describe a multitask estimation algorithm over a fully connected broadcasting network. These works assume that the node-specific parameter vectors to estimate lie in a common latent signal subspace and exploit this property to compress information and reduce communication costs. Another scenario is described in [39], [40], where incremental and diffusion strategies are used to solve a distributed estimation problem with nodes that simultaneously estimate local and global parameters. In [41], the parameter space is decomposed into two orthogonal subspaces with one of them being common to all nodes.

In all these previous examples, it is assumed beforehand that the nodes have some prior knowledge about clustering or about the parameter space, such as which agents belong to a particular cluster or how the parameter space is subdivided. The aim of the current work is different and does not assume prior information about the tasks or clusters. It then becomes critical to assess the performance limits of diffusion strategies when used, knowingly or unknowingly, in an multitask environment. Due to inaccurate modeling, or minor differences between tasks neglected intentionally, there may be situations in which the diffusion LMS algorithm is applied to multitask scenarios. When these situations occur, the distributed implementation will lead to biased results that may be acceptable depending on the application at hand. This biased solution may still be beneficial compared to purely non-cooperative strategies provided that the local optimums are sufficiently close to each other.

These observations motivate us to examine the performance of the diffusion LMS strategy when it is run, either intentionally or unintentionally, in a multitask environment. In this respect, we shall analyze the performance of the diffusion LMS in terms of its mean weight deviation and mean-square error in the case when the single-task hypothesis is violated. We shall also identify and analyze the competing factors that influence the performance of diffusion LMS in the multitask environment, and which allow this algorithm to continue to deliver performance superior to non-cooperative strategies in some useful circumstances. We shall also propose an unsupervised clustering strategy that allows each node to select, via adaptive adjustments of combination weights, the neighboring nodes with which it should collaborate to improve its estimation accuracy. In the related work [35], we formulated the multitask problem directly over networks with connected clusters of nodes. In that work, the clusters are assumed to be known beforehand and no clustering is proposed. We then derived extended diffusion strategies that enable adaptation and learning under these conditions. In the current work, on the other hand, the clusters are not assumed to be known. It then becomes necessary to examine how this lack of information influences performance. It also becomes necessary to endow the nodes with the ability to identify and form appropriate clusters to enhance performance. One clustering strategy was proposed in the earlier work [42]; its performance is dependent on the initial conditions used by the nodes to launch their adaptation rules. In this work, we propose a more robust clustering strategy.

Notation: Boldface small letters $\boldsymbol{x}$ denote vectors. All vectors are column vectors. Boldface capital letters $\boldsymbol{X}$ denote matrices. The superscript $(\cdot)^\top$ represents the transpose of a matrix or a vector. Matrix trace is denoted by trace $\{\cdot\}$, Kronecker product is denoted by $\otimes$, and expectation is denoted by $\mathbb{E}\{\cdot\}$. Identity matrix of size $N \times N$ is denoted by $\boldsymbol{I}_N$, and the all-one vector of length N is denoted by $\boldsymbol{1}_N$. We denote by $\mathcal{N}_k$ the set of node indices in the neighborhood of node k, including k itself, and $|\mathcal{N}_k|$ its cardinality. The operator $\mathrm{col}\{\cdot\}$ stacks its vector arguments on the top of each other to generate a connected vector. The other symbols will be defined in the context where they are used.

II. Multitask Problems and Diffusion LMS

A. Modeling Assumptions and Pareto Solution

We consider a connected network composed of N nodes. The problem is to estimate $L \times 1$ unknown vectors $\boldsymbol{w}_k^\star$ at each node k from collected measurements. Node k has access to temporal wide-sense stationary measurement sequences $\{d_k(n), \boldsymbol{x}_k(n)\}$, with $d_k(n)$ denoting a scalar zero-mean reference signal, and $\boldsymbol{x}_k(n)$ an $L \times 1$ regression vector with a positive definite covariance matrix $\boldsymbol{R}_{x,k} = \mathbb{E}\{\boldsymbol{x}_k(n)\boldsymbol{x}_k^\top(n)\} > 0$. The data at node k are assumed to be related via the linear regression model:

$$d_k(n) = \boldsymbol{x}_k^\top(n)\boldsymbol{w}_k^\star + z_k(n) \tag{1}$$

where $z_k(n)$ is a zero-mean i.i.d. additive noise at node k and time n. Noise $z_k(n)$ is assumed to be independent of any other signals and has variance $\sigma_{z,k}^2$. Let $J_k(\boldsymbol{w})$ denote the mean-square-error cost at node k, namely,

$$J_k(\boldsymbol{w}) = \mathbb{E}\left\{|d_k(n) - \boldsymbol{x}_k^\top(n)\boldsymbol{w}|^2\right\}. \tag{2}$$

It is clear from (1) that each $J_k(\boldsymbol{w})$ is minimized at $\boldsymbol{w}_k^\star$. Depending on whether the minima of all the $J_k(\boldsymbol{w})$ are achieved at the same location or not, referred to as tasks, the distributed learning problem can be single-task or multitask oriented [35].

In a single-task network, all nodes have to estimate the same parameter vector $\boldsymbol{w}^\star$. That is, in this case we have that

$$\boldsymbol{w}_k^\star = \boldsymbol{w}^\star, \quad \forall k \in \{1, \ldots, N\}. \tag{3}$$

Diffusion LMS strategies for the distributed estimation of $\boldsymbol{w}^\star$ under this scenario were derived in [2], [16], [17], [43] by seeking the minimizer of the following aggregate cost function:

$$J^{\mathrm{glob}}(\boldsymbol{w}) = \sum_{k=1}^{N} J_k(\boldsymbol{w}) \tag{4}$$

in a cooperative manner in order to improve estimation accuracy. In a multitask network, on the other hand, each node needs to determine its own parameter vector $\boldsymbol{w}_k^\star$. In [35], we assume that the parameter vectors at two connected nodes k and ℓ may satisfy certain similarity properties, such as being close to each other in some Euclidean norm sense. Nodes can also be interested in simultaneously estimating some parameters of local interest as well as parameters of global interest [40], [41]. Cooperation between these nodes can therefore be beneficial to

infer $\boldsymbol{w}_k^\star$ and $\boldsymbol{w}_\ell^\star$. A possible way to exploit and model relationships among tasks is to formulate optimization problems with appropriate co-regularizers [35]. An alternative is to build on the principle that the node hypothesis spaces partially overlap [40], [41]. These formulations, however, require some prior knowledge about how tasks are related to each other. In this work, we do not assume the availability of any prior information; in particular, nodes do not know which other nodes share similar objectives. Now since each cost function $J_k(\boldsymbol{w})$ may not be minimized at the same location, the minimizer of the aggregate cost (4) can be shown to correspond to a Pareto optimum solution for the multi-objective optimization problem [22], [44], [47]. Diffusion LMS thus leads to a compromise for the entire network, and we would like to examine how its performance is affected when used in a multitask scenario.

B. Diffusion LMS

The diffusion LMS algorithm was originally designed for minimizing the cost function (4) in an adaptive and distributed manner [16], [17], [43], [45]. Let $\boldsymbol{w}_k(n)$ denote the estimate of the minimizer of (4) at node k and time instant n. The general structure of the algorithm consists of the following steps:

$$\boldsymbol{\phi}_k(n) = \sum_{\ell\in\mathcal{N}_k} a_{1,\ell k}\boldsymbol{w}_\ell(n) \tag{5}$$

$$\boldsymbol{\psi}_k(n+1) = \boldsymbol{\phi}_k(n) + \mu_k \sum_{\ell\in\mathcal{N}_k} c_{\ell k}\boldsymbol{x}_\ell(n)\big[d_\ell(n) - \boldsymbol{x}_\ell^\top(n)\boldsymbol{\phi}_k(n)\big] \tag{6}$$

$$\boldsymbol{w}_k(n+1) = \sum_{\ell\in\mathcal{N}_k} a_{2,\ell k}\boldsymbol{\psi}_\ell(n+1). \tag{7}$$

The non-negative coefficients $a_{1,\ell k}$, $a_{2,\ell k}$ and $c_{\ell k}$ are the (ℓ,k)-th entries of two left-stochastic matrices, $\boldsymbol{A}_1$ and $\boldsymbol{A}_2$, and a right-stochastic matrix $\boldsymbol{C}$, that is,

$$\boldsymbol{A}_1^\top\mathbf{1}_N = \mathbf{1}_N, \boldsymbol{A}_2^\top\mathbf{1}_N = \mathbf{1}_N, \boldsymbol{C}\mathbf{1}_N = \mathbf{1}_N \tag{8}$$

and satisfy

$$a_{1,\ell k} = 0, a_{2,\ell k} = 0, c_{\ell k} = 0 \quad \text{if} \quad \ell \notin \mathcal{N}_k. \tag{9}$$

Several adaptive strategies can be obtained as special cases of (5)–(7) through appropriate selections of $\boldsymbol{A}_1$, $\boldsymbol{A}_2$ and $\boldsymbol{C}$. For instance, setting $\boldsymbol{A}_1 = \boldsymbol{I}_N$ yields the so-called adapt-then-combine (ATC) diffusion LMS. Setting $\boldsymbol{A}_2 = \boldsymbol{I}_N$ leads to the combine-then-adapt (CTA) diffusion LMS. By setting $\boldsymbol{A}_1 = \boldsymbol{A}_2 = \boldsymbol{C} = \boldsymbol{I}_N$, the algorithm degenerates to non-cooperative LMS that will be considered in the sequel for comparison purposes.

When applying ATC diffusion LMS without gradient information exchange, that is, with $\boldsymbol{C} = \boldsymbol{I}_N$, the agents converge toward the Pareto optimum with a bias of the order $\mathcal{O}(\mu_{\max})$, where $\mu_{\max}$ denotes the largest step-size parameter across all nodes [22]. In this paper, rather than focusing on this convergence point that can be perceived as a compromise, we shall study analytically how diffusion LMS (5)–(7) behaves in a multitask environment in relation to the optimum vectors $\boldsymbol{w}_k^\star$. Moreover, in order to generalize the analysis, we shall consider drifting optimums around a fixed value $\boldsymbol{w}_k^\star$, namely,

$$\boldsymbol{w}_k^\star(n) = \boldsymbol{w}_k^\star + \boldsymbol{\epsilon}_k(n) \tag{10}$$

where $\boldsymbol{\epsilon}_k(n)$ is a zero-mean random perturbation independent of any other signal, with zero-mean and covariance matrix $\sigma_{\epsilon,k}^2\boldsymbol{I}_L$. Under (10), model (1) is replaced by

$$d_k(n) = \boldsymbol{x}_k^\top(n)\boldsymbol{w}_k^\star(n) + z_k(n). \tag{11}$$

III. Performance Analysis of Diffusion LMS for Multitask Networks

We collect the information from across the network into block vectors and matrices. In particular, we denote by $\boldsymbol{w}(n)$, $\boldsymbol{w}^\star$ and $\boldsymbol{w}^\star(n)$ the block weight estimate vector (12), the block optimum mean weight vector (13), and the instantaneous block optimum weight vector (14), all of size $LN \times 1$, that is,

$$\boldsymbol{w}(n) = \operatorname{col}\{\boldsymbol{w}_1(n), \ldots, \boldsymbol{w}_N(n)\} \tag{12}$$

$$\boldsymbol{w}^\star = \operatorname{col}\{\boldsymbol{w}_1^\star, \ldots, \boldsymbol{w}_N^\star\} \tag{13}$$

$$\boldsymbol{w}^\star(n) = \operatorname{col}\{\boldsymbol{w}_1^\star(n), \ldots, \boldsymbol{w}_N^\star(n)\}. \tag{14}$$

The weight error vector $\boldsymbol{v}_k(n)$ for each node k at iteration n is defined by

$$\boldsymbol{v}_k(n) = \boldsymbol{w}_k(n) - \boldsymbol{w}_k^\star(n). \tag{15}$$

Let

$$\boldsymbol{v}_k^\star(n) = \boldsymbol{w}_k(n) - \boldsymbol{w}_k^\star \tag{16}$$

be the weight error vector between $\boldsymbol{w}_k(n)$ and the fixed weight vector $\boldsymbol{w}_k^\star$. The following relation holds

$$\boldsymbol{v}_k(n) = \boldsymbol{v}_k^\star(n) - \boldsymbol{\epsilon}_k(n). \tag{17}$$

This relation allows us to derive recursions with respect to $\boldsymbol{v}_k^\star(n)$, and then get back to $\boldsymbol{v}_k(n)$. The weight error vectors $\boldsymbol{v}_k(n)$ and $\boldsymbol{v}_k^\star(n)$ are also stacked on top of each other to get the block weight error vectors:

$$\boldsymbol{v}(n) = \operatorname{col}\{\boldsymbol{v}_1(n), \ldots, \boldsymbol{v}_N(n)\} \tag{18}$$

$$\boldsymbol{v}^\star(n) = \operatorname{col}\{\boldsymbol{v}_1^\star(n), \ldots, \boldsymbol{v}_N^\star(n)\}. \tag{19}$$

To perform the theoretical analysis, we introduce the following independence assumption.

Assumption 1 (Independent Regressors): The regression vectors $\boldsymbol{x}_k(n)$ arise from a zero-mean random process that is temporally (over n) stationary, white, and independent over space (over k) with $\boldsymbol{R}_{x,k} = \mathbb{E}\{\boldsymbol{x}_k(n)\boldsymbol{x}_k^\top(n)\} > 0$. ■

A direct consequence is that $\boldsymbol{x}_k(n)$ is independent of $\boldsymbol{v}_\ell(m)$ for all ℓ and $m \le n$. Although not true in general, this assumption is commonly used for analyzing adaptive constructions because it allows to simplify the derivation without constraining the conclusions. There are several results in the adaptation literature that show that performance results that are obtained under the above independence assumptions match well the actual performance of the algorithms when the step-sizes are sufficiently small. (see, e.g., [46, App. 24.A] and [47, Chs. 10–11] and the many references therein).

A. Mean Weight Behavior Analysis

Subtracting optimum vectors $\boldsymbol{w}_k^\star$ from both sides of the first step of diffusion LMS, namely (5), gives

$$\boldsymbol{\phi}_k(n) - \boldsymbol{w}_k^\star = \sum_{\ell\in\mathcal{N}_k} a_{1,\ell k}\boldsymbol{w}_\ell(n) - \boldsymbol{w}_k^\star. \tag{20}$$

Defining $\boldsymbol{\mathcal{A}}_1 = \boldsymbol{A}_1 \otimes \boldsymbol{I}_L$ and using $\boldsymbol{w}_k^\star = \boldsymbol{w}_k(n) - \boldsymbol{v}_k^\star(n)$, expression (20) can be expressed in block-based form:

$$\boldsymbol{\phi}(n) - \boldsymbol{w}^\star = \boldsymbol{\mathcal{A}}_1^\top \boldsymbol{v}^\star(n) + (\boldsymbol{\mathcal{A}}_1^\top - \boldsymbol{I}_{NL})\boldsymbol{w}^\star. \tag{21}$$

Note that the term $(\boldsymbol{\mathcal{A}}_1^\top - \boldsymbol{I}_{NL})\boldsymbol{w}^\star$, which does not appear for single-task networks, is inherited from the multitask context[1]. The estimation error in the second step (6) of diffusion LMS can be rewritten as

$$\begin{aligned} &d_\ell(n) - \boldsymbol{x}_\ell^\top(n)\boldsymbol{\phi}_k(n) \\ &= z_\ell(n) - \boldsymbol{x}_\ell^\top(n)(\boldsymbol{\phi}_k(n) - \boldsymbol{w}_\ell^\star) + \boldsymbol{x}_\ell^\top(n)\boldsymbol{\epsilon}_\ell(n). \end{aligned} \tag{22}$$

For single-task networks, $(\boldsymbol{\phi}_k(n) - \boldsymbol{w}_\ell^\star)$ in the above expression reduces to $(\boldsymbol{\phi}_k(n) - \boldsymbol{w}_k^\star)$ since $\boldsymbol{w}_k^\star = \boldsymbol{w}_\ell^\star$ for all k, ℓ. In the multitask context, we can establish the following relationship:

$$\begin{aligned} \boldsymbol{\phi}_k(n) - \boldsymbol{w}_\ell^\star &= (\boldsymbol{\phi}_k(n) - \boldsymbol{w}_k^\star) + (\boldsymbol{w}_k^\star - \boldsymbol{w}_\ell^\star) \\ &= (\boldsymbol{\phi}_k(n) - \boldsymbol{w}_k^\star) + \boldsymbol{u}_{k\ell}^\star \end{aligned} \tag{23}$$

where $\boldsymbol{u}_{k\ell}^\star$ is the difference between the fixed weight vectors $\boldsymbol{w}_k^\star$ and $\boldsymbol{w}_\ell^\star$. Incorporating this expression into (22) yields:

$$\begin{aligned} &d_\ell(n) - \boldsymbol{x}_\ell^\top(n)\boldsymbol{\phi}_k(n) \\ &= z_\ell(n) - \boldsymbol{x}_\ell^\top(n)(\boldsymbol{\phi}_k(n) - \boldsymbol{w}_k^\star) - \boldsymbol{x}_\ell^\top(n)\boldsymbol{u}_{k\ell}^\star + \boldsymbol{x}_\ell^\top(n)\boldsymbol{\epsilon}_\ell(n). \end{aligned} \tag{24}$$

Subtracting $\boldsymbol{w}_k^\star$ from both sides of (6) and using the above relation, we have

$$\begin{aligned} &\boldsymbol{\psi}_k(n+1) - \boldsymbol{w}_k^\star \\ &= (\boldsymbol{\phi}_k(n) - \boldsymbol{w}_k^\star) - \mu_k \sum_{\ell\in\mathcal{N}_k} c_{\ell k}\boldsymbol{x}_\ell(n)\boldsymbol{x}_\ell^\top(n)(\boldsymbol{\phi}_k(n) - \boldsymbol{w}_k^\star) \\ &+\mu_k \sum_{\ell\in\mathcal{N}_k} c_{\ell k}\,\boldsymbol{x}_\ell(n)z_\ell(n) - \mu_k \sum_{\ell\in\mathcal{N}_k} c_{\ell k}\,\boldsymbol{x}_\ell(n)\boldsymbol{x}_\ell^\top(n)\,\boldsymbol{u}_{k\ell}^\star \\ &+\mu_k \sum_{\ell\in\mathcal{N}_k} c_{\ell k}\,\boldsymbol{x}_\ell(n)\boldsymbol{x}_\ell^\top(n)\,\boldsymbol{\epsilon}_\ell(n). \end{aligned} \tag{25}$$

Let us introduce the following $N \times N$ block diagonal matrices with blocks of size $L \times L$:

$$\boldsymbol{U} = \operatorname{diag}\{\mu_1\boldsymbol{I}_L, \ldots, \mu_N\boldsymbol{I}_L\} \tag{26}$$

$$\boldsymbol{H}(n) = \operatorname{diag}\Big\{\sum_{\ell\in\mathcal{N}_k} c_{\ell k}\,\boldsymbol{x}_\ell(n)\boldsymbol{x}_\ell^\top(n)\Big\}_{k=1}^N, \tag{27}$$

and the following vectors of length NL:

$$\boldsymbol{h}_u(n) = \operatorname{col}\Big\{\sum_{\ell\in\mathcal{N}_k} c_{\ell k}\,\boldsymbol{x}_\ell(n)\boldsymbol{x}_\ell^\top(n)\boldsymbol{u}_{k\ell}^\star\Big\}_{k=1}^N \tag{28}$$

$$\boldsymbol{h}_\epsilon(n) = \operatorname{col}\Big\{\sum_{\ell\in\mathcal{N}_k} c_{\ell k}\,\boldsymbol{x}_\ell(n)\boldsymbol{x}_\ell^\top(n)\boldsymbol{\epsilon}_\ell(n)\Big\}_{k=1}^N \tag{29}$$

$$\boldsymbol{p}_{zx}(n) = \operatorname{col}\Big\{\sum_{\ell\in\mathcal{N}_k} c_{\ell k}\boldsymbol{x}_\ell(n)z_\ell(n)\Big\}_{k=1}^N. \tag{30}$$

[1]In the single-task context, and since all nodes share the same optimum, say $\boldsymbol{w}_1^\star$, we can write $\boldsymbol{w}^\star = \operatorname{col}\{\boldsymbol{w}_1^\star, \ldots, \boldsymbol{w}_1^\star\}$. Consequently, $\boldsymbol{\mathcal{A}}_1^\top\boldsymbol{w}^\star = (\boldsymbol{A}_1 \otimes \boldsymbol{I}_L)(\boldsymbol{1}_N \otimes \boldsymbol{w}_1^\star) = (\boldsymbol{A}_1\boldsymbol{1}_N) \otimes \boldsymbol{w}_1^\star = \boldsymbol{w}^\star$, where the last step is due to the fact that $\boldsymbol{A}_1$ is left-stochastic. This result leads to $(\boldsymbol{\mathcal{A}}_1^\top - \boldsymbol{I}_{NL})\boldsymbol{w}^\star = \boldsymbol{0}_{NL}$.

Using the above notation and (21) and (25), the block weight error vector $\boldsymbol{\psi}(n+1)$ can be expressed as

$$\begin{aligned} &\boldsymbol{\psi}(n+1) - \boldsymbol{w}^\star = \\ &(\boldsymbol{I}_{NL} - \boldsymbol{U}\boldsymbol{H}(n))\big(\boldsymbol{\mathcal{A}}_1^\top\boldsymbol{v}^\star(n) + (\boldsymbol{\mathcal{A}}_1^\top - \boldsymbol{I}_{NL})\boldsymbol{w}^\star\big) + \\ &\boldsymbol{U}\boldsymbol{p}_{zx}(n) - \boldsymbol{U}(\boldsymbol{h}_u(n) - \boldsymbol{h}_\epsilon(n)). \end{aligned} \tag{31}$$

Let $\boldsymbol{\mathcal{A}}_2 = \boldsymbol{A}_2 \otimes \boldsymbol{I}_L$. The combination step (7) of diffusion LMS leads to

$$\boldsymbol{w}(n+1) = \boldsymbol{\mathcal{A}}_2^\top\boldsymbol{\psi}(n+1). \tag{32}$$

Subtracting $\boldsymbol{w}^\star$ from both sides of (32), we get

$$\boldsymbol{v}^\star(n+1) = \boldsymbol{\mathcal{A}}_2^\top(\boldsymbol{\psi}(n+1) - \boldsymbol{w}^\star) + (\boldsymbol{\mathcal{A}}_2^\top - \boldsymbol{I}_{NL})\boldsymbol{w}^\star. \tag{33}$$

Again, note that the term $(\boldsymbol{\mathcal{A}}_2^\top - \boldsymbol{I}_{NL})\,\boldsymbol{w}^\star$ does not appear for single-task networks and is inherited from the multitask context. Combining (31) and (33), we obtain the update relation for $\boldsymbol{v}^\star(n)$ in a single expression as follows:

$$\begin{aligned} &\boldsymbol{v}^\star(n+1) = \\ &\boldsymbol{\mathcal{A}}_2^\top(\boldsymbol{I}_{NL} - \boldsymbol{U}\boldsymbol{H}(n))\boldsymbol{\mathcal{A}}_1^\top\boldsymbol{v}^\star(n) + \\ &\boldsymbol{\mathcal{A}}_2^\top\boldsymbol{U}\boldsymbol{p}_{zx}(n) - \boldsymbol{\mathcal{A}}_2^\top\,\boldsymbol{U}(\boldsymbol{h}_u(n) - \boldsymbol{h}_\epsilon(n)) + \\ &\Big(\boldsymbol{\mathcal{A}}_2^\top(\boldsymbol{I}_{NL}(n) - \boldsymbol{U}\boldsymbol{H}(n))(\boldsymbol{\mathcal{A}}_1^\top - \boldsymbol{I}_{NL}) + (\boldsymbol{\mathcal{A}}_2^\top - \boldsymbol{I}_{NL})\Big)\,\boldsymbol{w}^\star. \end{aligned} \tag{34}$$

In order to make the presentation clearer, we use the following notation for terms in expression (34):

$$\boldsymbol{B}(n) = \boldsymbol{\mathcal{A}}_2^\top(\boldsymbol{I}_{NL} - \boldsymbol{U}\boldsymbol{H}(n))\boldsymbol{\mathcal{A}}_1^\top \tag{35}$$

$$\boldsymbol{g}(n) = \boldsymbol{\mathcal{A}}_2^\top\boldsymbol{U}\boldsymbol{p}_{zx}(n) \tag{36}$$

$$\begin{aligned} \boldsymbol{r}(n) = &\underbrace{\boldsymbol{\mathcal{A}}_2^\top\boldsymbol{U}\boldsymbol{h}_u(n)}_{\boldsymbol{r}_u(n)} - \underbrace{\boldsymbol{\mathcal{A}}_2^\top\boldsymbol{U}\boldsymbol{h}_\epsilon(n)}_{\boldsymbol{r}_\epsilon(n)} \\ &- \underbrace{\Big(\boldsymbol{\mathcal{A}}_2^\top(\boldsymbol{I}_{NL} - \boldsymbol{U}\boldsymbol{H}(n))(\boldsymbol{\mathcal{A}}_1^\top - \boldsymbol{I}_{NL}) + (\boldsymbol{\mathcal{A}}_2^\top - \boldsymbol{I}_{NL})\Big)\,\boldsymbol{w}^\star}_{\boldsymbol{r}_w(n)}. \end{aligned} \tag{37}$$

Then, recursion (34) can be rewritten as

$$\boldsymbol{v}^\star(n+1) = \boldsymbol{B}(n)\boldsymbol{v}^\star(n) + \boldsymbol{g}(n) - \boldsymbol{r}(n). \tag{38}$$

The non-zero driving term $\boldsymbol{r}(n)$, arising from the multitask scenario and the random perturbations $\boldsymbol{\epsilon}_k(n)$, introduces a further level of complexity in the algorithm analysis, especially in the mean-square error behavior one. This analysis reduces to the traditional analysis of diffusion LMS by setting $\boldsymbol{r}(n) = 0$. Let $\boldsymbol{H}$ be the expected value of $\boldsymbol{H}(n)$ given by

$$\boldsymbol{H} = \operatorname{diag}\{\boldsymbol{R}_1, \ldots, \boldsymbol{R}_N\} \tag{39}$$

in terms of the neighborhood covariance matrices:

$$\boldsymbol{R}_k = \sum_{\ell\in\mathcal{N}_k} c_{\ell k}\boldsymbol{R}_{x,\ell}. \tag{40}$$

Let $\boldsymbol{h}_u$ be the expected value $\mathbb{E}\{\boldsymbol{h}_u(n)\}$, that is,

$$\boldsymbol{h}_u = \operatorname{col}\Big\{\sum_{\ell\in\mathcal{N}_1} c_{\ell 1}\boldsymbol{R}_{x,\ell}\boldsymbol{u}_{1\ell}^\star, \ldots, \sum_{\ell\in\mathcal{N}_N} c_{\ell N}\boldsymbol{R}_{x,\ell}\boldsymbol{u}_{N\ell}^\star\Big\}. \tag{41}$$

The independence assumption (Assumption 1), and the statistical properties of noise $z_k(n)$ and perturbations $\boldsymbol{\epsilon}_k(n)$, lead us to the following expected values for $\boldsymbol{B}(n)$, $\boldsymbol{g}(n)$ and $\boldsymbol{r}(n)$:

$$\boldsymbol{B} = \boldsymbol{\mathcal{A}}_2^\top \left(\boldsymbol{I}_{NL} - \boldsymbol{U}\boldsymbol{H}\right)\boldsymbol{\mathcal{A}}_1^\top \tag{42}$$

$$\boldsymbol{g} = 0 \tag{43}$$

$$\boldsymbol{r} = \underbrace{\boldsymbol{\mathcal{A}}_2^\top \boldsymbol{U}\boldsymbol{h}_u}_{\boldsymbol{r}_u} - \underbrace{\left(\boldsymbol{\mathcal{A}}_2^\top(\boldsymbol{I}_{NL} - \boldsymbol{U}\boldsymbol{H})(\boldsymbol{\mathcal{A}}_1^\top - \boldsymbol{I}_{NL}) + (\boldsymbol{\mathcal{A}}_2^\top - \boldsymbol{I}_{NL})\right)\boldsymbol{w}^\star}_{\boldsymbol{r}_w} \tag{44}$$

where $\boldsymbol{r}_u, \boldsymbol{r}_\epsilon$ and $\boldsymbol{r}_w$ denote the expected values of $\boldsymbol{r}_u(n), \boldsymbol{r}_\epsilon(n)$ and $\boldsymbol{r}_w(n)$, respectively. Note that the expected value $\boldsymbol{r}$ is expressed as $\boldsymbol{r} = \boldsymbol{r}_u - \boldsymbol{r}_w$ because $\boldsymbol{r}_\epsilon = 0$. Taking the expectation of both sides of (34), and observing that $\boldsymbol{B}(n)$ and $\boldsymbol{v}^\star(n)$ are independent under Assumption 1, we get

$$\mathbb{E}\{\boldsymbol{v}^\star(n+1)\} = \boldsymbol{B}\mathbb{E}\{\boldsymbol{v}^\star(n)\} - \boldsymbol{r}_u + \boldsymbol{r}_w. \tag{45}$$

Moreover, (17) tells us that

$$\mathbb{E}\{\boldsymbol{v}(n+1)\} = \mathbb{E}\{\boldsymbol{v}^\star(n+1)\}. \tag{46}$$

Theorem 1 (Stability in the Mean): Assume data model (1) and Assumption 1 hold. Then, for any initial condition, the diffusion LMS strategy (5)–(7) applied to multitask networks asymptotically converges in the mean if the step-sizes are chosen to satisfy

$$0 < \mu_k < \frac{2}{\lambda_{\max}\{\boldsymbol{R}_k\}}, \qquad k = 1, \ldots, N \tag{47}$$

where $\lambda_{\max}\{\cdot\}$ denotes the maximum eigenvalue of its matrix argument. In that case, it follows from (45) that the asymptotic mean bias is given by

$$\mathbb{E}\{\boldsymbol{v}(\infty)\} = \left(\boldsymbol{I}_{NL} - \boldsymbol{B}\right)^{-1}(-\boldsymbol{r}_u + \boldsymbol{r}_w). \tag{48}$$

Proof: Since the last two terms on the RHS of (45) are constant, the convergence of this recursion requires that the matrix $\boldsymbol{B}$ be stable. As shown in [43], because $\boldsymbol{\mathcal{A}}_1$ and $\boldsymbol{\mathcal{A}}_2$ are left-stochastic, the spectral norm of $\boldsymbol{\mathcal{A}}_2^\top(\boldsymbol{I}_{NL} - \boldsymbol{U}\boldsymbol{H})\boldsymbol{\mathcal{A}}_1^\top$ is upper bounded by the spectral norm of $\boldsymbol{I}_{NL} - \boldsymbol{U}\boldsymbol{H}$. The former is thus stable if the latter is stable. This yields condition (47) considering that $\boldsymbol{H}$ is a block diagonal matrix of the form (39). ■

Inspecting expression (48), we observe that the bias of diffusion LMS originates from the multiple local optimums $\boldsymbol{w}_k^\star(n)$ and information exchange among neighbors. This means that, even though the algorithm converges toward the Pareto optimum over multitask networks [22], the bias (48) can be large if the distance between the $\boldsymbol{w}_k^\star(n)$ is large, and if nodes cooperate to estimate them.

B. Mean-Square Error Behavior Analysis

By Assumption 1, (38), and definition (36) of $\boldsymbol{g}(n)$ where $z_k(n)$ is a zero-mean noise independent of any other signal, the mean-square of the weight error vector $\boldsymbol{v}^\star(n+1)$ weighted by any positive semi-definite matrix $\boldsymbol{\Sigma}$ satisfies the following relation:

$$\begin{aligned}\mathbb{E}\{\|\boldsymbol{v}^\star(n+1)\|_{\boldsymbol{\Sigma}}^2\} = \\ \mathbb{E}\{\|\boldsymbol{v}^\star(n)\|_{\boldsymbol{\Sigma}'}^2\} + \operatorname{trace}\left\{\boldsymbol{\Sigma}\mathbb{E}\{\boldsymbol{g}(n)\boldsymbol{g}^\top(n)\}\right\} + \\ \mathbb{E}\{\|\boldsymbol{r}(n)\|_{\boldsymbol{\Sigma}}^2\} - 2\mathbb{E}\{\boldsymbol{r}^\top(n)\boldsymbol{\Sigma}\boldsymbol{B}(n)\boldsymbol{v}^\star(n)\}\end{aligned} \tag{49}$$

with $\|\boldsymbol{x}\|_{\boldsymbol{\Sigma}}^2 = \boldsymbol{x}^\top\boldsymbol{\Sigma}\boldsymbol{x}$, and $\boldsymbol{\Sigma}' = \mathbb{E}\{\boldsymbol{B}^\top(n)\boldsymbol{\Sigma}\boldsymbol{B}(n)\}$. The freedom in selecting $\boldsymbol{\Sigma}$ will allow us to derive several performance metrics. Let

$$\begin{aligned}\boldsymbol{G} &= \mathbb{E}\{\boldsymbol{g}(n)\boldsymbol{g}^\top(n)\} \\ &= \boldsymbol{\mathcal{A}}_2^\top\boldsymbol{U}\boldsymbol{\mathcal{C}}^\top\operatorname{diag}\{\sigma_{z,1}^2\boldsymbol{R}_{x,1}, \ldots, \sigma_{z,N}^2\boldsymbol{R}_{x,N}\}\boldsymbol{\mathcal{C}}\boldsymbol{U}\boldsymbol{\mathcal{A}}_2\end{aligned} \tag{50}$$

where $\boldsymbol{\mathcal{C}} = \boldsymbol{C} \otimes \boldsymbol{I}_L$. For the sake of clarity, let us introduce

$$\boldsymbol{f}(\boldsymbol{r}(n), \boldsymbol{\Sigma}, \boldsymbol{v}^\star(n)) = \|\boldsymbol{r}(n)\|_{\boldsymbol{\Sigma}}^2 - 2\boldsymbol{r}^\top(n)\boldsymbol{\Sigma}\boldsymbol{B}(n)\boldsymbol{v}^\star(n). \tag{51}$$

Relation (49) can then be written as

$$\begin{aligned}\mathbb{E}\{\|\boldsymbol{v}^\star(n+1)\|_{\boldsymbol{\Sigma}}^2\} = \mathbb{E}\{\|\boldsymbol{v}^\star(n)\|_{\boldsymbol{\Sigma}'}^2\} + \operatorname{trace}\{\boldsymbol{\Sigma}\boldsymbol{G}\} \\ + \mathbb{E}\{\boldsymbol{f}(\boldsymbol{r}(n), \boldsymbol{\Sigma}, \boldsymbol{v}^\star(n))\}.\end{aligned} \tag{52}$$

Let $\operatorname{vec}\{\cdot\}$ denote the operator that stacks the columns of a matrix on top of each other. Vectorizing both matrices $\boldsymbol{\Sigma}$ and $\boldsymbol{\Sigma}'$ by $\boldsymbol{\sigma} = \operatorname{vec}\{\boldsymbol{\Sigma}\}$ and $\boldsymbol{\sigma}' = \operatorname{vec}\{\boldsymbol{\Sigma}'\}$, it can be checked that

$$\boldsymbol{\sigma}' = \boldsymbol{K}\,\boldsymbol{\sigma} \tag{53}$$

where $\boldsymbol{K}$ is the $(NL)^2 \times (NL)^2$ matrix given by

$$\boldsymbol{K} = \mathbb{E}\{\boldsymbol{B}^\top(n) \otimes \boldsymbol{B}^\top(n)\}. \tag{54}$$

We can rewrite $\boldsymbol{K}$ as $\boldsymbol{B}^\top \otimes \boldsymbol{B}^\top + \mathcal{O}(\mu_{\max}^2)$, with an error term that depends on the square of the (maximum) step-size entry (see [43, Section 6.5] and [47, Ch. 10]). It is sufficient for the exposition in this work to focus on the case of sufficiently small step-sizes where terms involving higher powers of the step-sizes can be ignored. Therefore, we continue our discussion by letting

$$\boldsymbol{K} = \boldsymbol{B}^\top \otimes \boldsymbol{B}^\top. \tag{55}$$

Let us now examine the term $\mathbb{E}\{\boldsymbol{f}(\boldsymbol{r}(n), \boldsymbol{\Sigma}, \boldsymbol{v}^\star(n))\}$. Consider first the weighted norm $\mathbb{E}\{\|\boldsymbol{r}(n)\|_{\boldsymbol{\Sigma}}^2\}$:

$$\begin{aligned}\mathbb{E}\{\|\boldsymbol{r}(n)\|_{\boldsymbol{\Sigma}}^2\} = \\ \mathbb{E}\{\boldsymbol{r}_u^\top(n)\boldsymbol{\Sigma}\boldsymbol{r}_u(n)\} + \mathbb{E}\{\boldsymbol{r}_\epsilon^\top(n)\boldsymbol{\Sigma}\boldsymbol{r}_\epsilon(n)\} + \\ \mathbb{E}\{\boldsymbol{r}_w^\top(n)\boldsymbol{\Sigma}\boldsymbol{r}_w(n)\} - 2\mathbb{E}\{\boldsymbol{r}_u^\top(n)\boldsymbol{\Sigma}\boldsymbol{r}_w(n)\}.\end{aligned} \tag{56}$$

We note that the stochastic components in $\boldsymbol{r}_u(n)$, $\boldsymbol{r}_\epsilon(n)$ and $\boldsymbol{r}_w(n)$ depend on the square of the step-sizes. We can write

$$\mathbb{E}\{\|\boldsymbol{r}(n)\|_{\boldsymbol{\Sigma}}^2\} = \|\boldsymbol{r}\|_{\boldsymbol{\Sigma}}^2 + \mathcal{O}(\mu_{\max}^2). \tag{57}$$

Likewise, we can write

$$\begin{aligned}\mathbb{E}\{\boldsymbol{r}^\top(n)\boldsymbol{\Sigma}\boldsymbol{B}(n)\boldsymbol{v}^\star(n)\} \\ = \boldsymbol{r}^\top\boldsymbol{\Sigma}\boldsymbol{B}\mathbb{E}\{\boldsymbol{v}^\star(n)\} + \mathcal{O}(\mu_{\max}^2).\end{aligned} \tag{58}$$

By ignoring the higher-order terms for small step-sizes, we can continue the presentation by considering:

$$\mathbb{E}\{\boldsymbol{f}(\boldsymbol{r}(n), \boldsymbol{\Sigma}, \boldsymbol{v}^\star(n))\} = \boldsymbol{f}(\boldsymbol{r}, \boldsymbol{\Sigma}, \mathbb{E}\{\boldsymbol{v}^\star(n)\}). \tag{59}$$

In this way, relation (52) can be approximated as follows:

$$\mathbb{E}\{\|\boldsymbol{v}^\star(n+1)\|^2_{\boldsymbol{\sigma}}\} = \mathbb{E}\{\|\boldsymbol{v}^\star(n)\|^2_{\boldsymbol{K}\boldsymbol{\sigma}}\} + \mathrm{vec}(\boldsymbol{G}^\top)^\top\boldsymbol{\sigma} + \boldsymbol{f}(\boldsymbol{r},\boldsymbol{\sigma},\mathbb{E}\{\boldsymbol{v}^\star(n)\}) \quad (60)$$

where we are using the notation $\|\cdot\|^2_{\boldsymbol{\Sigma}}$ and $\|\cdot\|^2_{\boldsymbol{\sigma}}$ interchangeably to refer to the same square weighted norm using $\boldsymbol{\Sigma}$ or its vector representation.

Theorem 2 (Mean-Square Stability): Assume model (1) and Assumption 1 hold. Assume that the step-sizes $\{\mu_k\}$ are sufficiently small such that condition (47) is satisfied and approximations (55) and (59) are justified by ignoring higher-order powers of $\{\mu_k\}$. Then, the diffusion LMS strategy (5)–(7) applied over multitask networks is mean-square stable if the matrix $\boldsymbol{K}$ is stable. Under approximation (55), the stability of $\boldsymbol{K}$ is guaranteed for sufficiently small step-sizes that also satisfy (47).

Proof: Iterating (60) starting from $n = 0$, we find that

$$\mathbb{E}\{\|\boldsymbol{v}^\star(n+1)\|^2_{\boldsymbol{\sigma}}\} = \|\boldsymbol{v}^\star(0)\|^2_{\boldsymbol{K}^{n+1}\boldsymbol{\sigma}} + \mathrm{vec}(\boldsymbol{G}^\top)^\top\sum_{i=0}^{n}\boldsymbol{K}^i\boldsymbol{\sigma} + \sum_{i=0}^{n}\boldsymbol{f}(\boldsymbol{r},\boldsymbol{K}^i\boldsymbol{\sigma},\mathbb{E}\{\boldsymbol{v}^\star(n-i)\}) \quad (61)$$

with the initial condition $\boldsymbol{v}^\star(0) = \boldsymbol{w}(0) - \boldsymbol{w}^\star$. Provided that matrix $\boldsymbol{K}$ is stable, the terms on the RHS of (61) converge either to zero, or to bounded values. The algorithm is then mean-square stable for sufficiently small step-sizes. ■

Corollary 1 (Transient MSD): Consider sufficiently small step-sizes μ_k that ensure mean and mean-square stability, and let $\boldsymbol{\Sigma} = \frac{1}{N}\boldsymbol{I}_{NL}$. Then, the mean-square deviation (MSD) learning curve of the diffusion LMS algorithm in a multitask environment, defined by $\zeta(n) = \mathbb{E}\{\|\boldsymbol{v}(n)\|^2\}/N$, evolves according to the following recursion for $n \geq 0$

$$\zeta(n) = \zeta^\star(n) + \frac{L}{N}\sum_{k=1}^{N}\sigma^2_{\epsilon,k} \quad (62)$$

where $\zeta^\star(n)$ is evaluated as follows

$$\begin{aligned}\zeta^\star(n+1) = \zeta^\star(n) + \Big(&\big(\mathrm{vec}\{\boldsymbol{G}^\top\}\big)^\top\boldsymbol{K}^n\boldsymbol{\sigma}_I + \|\boldsymbol{r}\|^2_{\boldsymbol{K}^n\boldsymbol{\sigma}_I} - \\ &\|\boldsymbol{v}^\star(0)\|^2_{(\boldsymbol{I}_{(NL)^2}-\boldsymbol{K})\boldsymbol{K}^n\boldsymbol{\sigma}_I} - \\ &2\big(\boldsymbol{\Gamma}(n) + (\boldsymbol{B}\mathbb{E}\{\boldsymbol{v}^\star(n)\})^\top\otimes\boldsymbol{r}^\top\big)\boldsymbol{\sigma}_I\Big) \quad (63)\end{aligned}$$

$$\boldsymbol{\Gamma}(n+1) = \boldsymbol{\Gamma}(n)\boldsymbol{K} + (\boldsymbol{B}\mathbb{E}\{\boldsymbol{v}^\star(n)\})^\top\otimes\boldsymbol{r}^\top)(\boldsymbol{K} - \boldsymbol{I}_{(NL)^2}) \quad (64)$$

with

$$\boldsymbol{\sigma}_I = \mathrm{vec}\{\frac{1}{N}\boldsymbol{I}_{NL}\}, \zeta^\star(0) = \frac{1}{N}\|\boldsymbol{v}^\star(0)\|^2, \boldsymbol{\Gamma}(0) = \boldsymbol{0}_{1\times(NL)^2}.$$

Proof: Comparing (61) at instants $n+1$ and n, we can relate $\mathbb{E}\{\|\boldsymbol{v}(n+1)\|^2_{\boldsymbol{\sigma}}\}$ to $\mathbb{E}\{\|\boldsymbol{v}(n)\|^2_{\boldsymbol{\sigma}}\}$:

$$\begin{aligned}\mathbb{E}\{\|\boldsymbol{v}^\star(n+1)\|^2_{\boldsymbol{\sigma}}\} = &\\ &\mathbb{E}\{\|\boldsymbol{v}^\star(n)\|^2_{\boldsymbol{\sigma}}\} - \mathbb{E}\{\|\boldsymbol{v}^\star(0)\|^2_{(\boldsymbol{I}_{(NL)^2}-\boldsymbol{K})\boldsymbol{K}^n\boldsymbol{\sigma}}\} + \\ &\mu^2\mathrm{vec}(\boldsymbol{G}^\top)^\top\boldsymbol{K}^n\boldsymbol{\sigma} + \mu^2\eta^2\|\boldsymbol{r}\|^2_{\boldsymbol{K}^n\boldsymbol{\sigma}} - \\ &2\mu\eta\left((\boldsymbol{B}\mathbb{E}\{\boldsymbol{v}^\star(n)\}\otimes\boldsymbol{r})^\top + \boldsymbol{\Gamma}(n)\right)\boldsymbol{\sigma} \quad (65)\end{aligned}$$

where

$$\boldsymbol{\Gamma}(n) = \sum_{i=1}^{n}(\boldsymbol{B}\mathbb{E}\{\boldsymbol{v}^\star(n-i)\}\otimes\boldsymbol{r})^\top\boldsymbol{K}^i + \sum_{i=0}^{n-1}(\boldsymbol{B}\mathbb{E}\{\boldsymbol{v}^\star(n-i-1)\}\otimes\boldsymbol{r})^\top\boldsymbol{K}^i. \quad (66)$$

We can then rewrite (65)–(66) as (63)–(64). ■

Corollary 2 (Steady-state MSD): If the step-sizes are sufficiently small to ensure mean and mean-square-error convergences, then the steady-state MSD for diffusion LMS in a multitask environment is given by

$$\begin{aligned}\mathrm{MSD}^{\mathrm{network}} = &\frac{1}{N}\big(\mathrm{vec}\{\boldsymbol{G}^\top\}\big)^\top(\boldsymbol{I}_{(NL)^2}-\boldsymbol{K})^{-1}\mathrm{vec}\{\boldsymbol{I}_{NL}\} \\ &+\boldsymbol{f}\Big(\boldsymbol{r},\frac{1}{N}(\boldsymbol{I}_{(NL)^2}-\boldsymbol{K})^{-1}\mathrm{vec}\{\boldsymbol{I}_{NL}\},\mathbb{E}\{\boldsymbol{v}^\star(\infty)\}\Big) \\ &+\frac{L}{N}\sum_{k=1}^{N}\sigma^2_{\epsilon,k} \quad (67)\end{aligned}$$

with $\mathbb{E}\{\boldsymbol{v}(\infty)\}$ determined by (48).

Proof: The steady-state MSD is given by the limit

$$\begin{aligned}\mathrm{MSD}^{\mathrm{network}} &= \lim_{n\to\infty}\frac{1}{N}\mathbb{E}\{\|\boldsymbol{v}(n)\|^2\} \\ &= \lim_{n\to\infty}\frac{1}{N}\mathbb{E}\{\|\boldsymbol{v}^\star(n)\|^2\} + \frac{L}{N}\sum_{k=1}^{N}\sigma^2_{\epsilon,k}. \quad (68)\end{aligned}$$

Recursion (60) with $n \to \infty$ yields

$$\lim_{n\to\infty}\mathbb{E}\{\|\boldsymbol{v}^\star(n)\|^2_{(\boldsymbol{I}_{(NL)^2}-\boldsymbol{K})\boldsymbol{\sigma}}\} = \big(\mathrm{vec}\{\boldsymbol{G}^\top\}\big)^\top\boldsymbol{\sigma} + \boldsymbol{f}(\boldsymbol{r},\boldsymbol{\sigma},\mathbb{E}\{\boldsymbol{v}^\star(\infty)\}). \quad (69)$$

In order to use (69) in (68), we select $\boldsymbol{\sigma}$ to satisfy:

$$(\boldsymbol{I}_{(NL)^2} - \boldsymbol{K})\,\boldsymbol{\sigma} = \frac{1}{N}\mathrm{vec}\{\boldsymbol{I}_{NL}\}. \quad (70)$$

This leads to expression (67). ■

The transient and steady-state MSD for any single node k can be obtained by setting $\boldsymbol{\Sigma} = \mathrm{diag}\{\boldsymbol{O}_N,\ldots,\boldsymbol{I}_L,\ldots,\boldsymbol{O}_N\}$ in Corollaries 1 and 2, with the identity matrix $\boldsymbol{I}_L$ at the k-th diagonal block and the all-zero matrix $\boldsymbol{O}_N$ at the others.

The steady-state MSD can be expressed in an alternative form, which will facilitate the performance analysis. Since $\boldsymbol{K}$ is stable when the network is mean-square stable, we can write

$$(\boldsymbol{I}_{(NL)^2} - \boldsymbol{K})^{-1} = \sum_{i=0}^{\infty}\boldsymbol{K}^i = \sum_{i=0}^{\infty}(\boldsymbol{B}^\top\otimes\boldsymbol{B}^\top)^i. \quad (71)$$

Consider now the following formula involving the trace of a product of matrices and the Kronecker product [48]

$$\mathrm{trace}\{\boldsymbol{X}_1^\top\boldsymbol{X}_2\boldsymbol{X}_3\boldsymbol{X}_4^\top\} = \mathrm{vec}\{\boldsymbol{X}_1\}^\top(\boldsymbol{X}_4\otimes\boldsymbol{X}_2)\mathrm{vec}\{\boldsymbol{X}_3\} \quad (72)$$

where $\boldsymbol{X}_1$ to $\boldsymbol{X}_4$ denote matrices with compatible sizes. Using expansion (71) with (72), the first term on the RHS of (67) can be expressed as follows

$$\frac{1}{N}\big(\mathrm{vec}\{\boldsymbol{G}^\top\}\big)^\top(\boldsymbol{I}_{(NL)^2}-\boldsymbol{K})^{-1}\mathrm{vec}\{\boldsymbol{I}_{NL}\}= \frac{1}{N}\sum_{j=0}^{\infty}\mathrm{trace}\{\boldsymbol{B}^j\boldsymbol{G}\boldsymbol{B}^{j\top}\}. \quad (73)$$

Similarly, the second term on the RHS of (67) can be written as

$$\begin{aligned}&\boldsymbol{f}\Big(\boldsymbol{r},\frac{1}{N}(\boldsymbol{I}_{(NL)^2}-\boldsymbol{K})^{-1}\mathrm{vec}\{\boldsymbol{I}_{NL}\},\mathbb{E}\{\boldsymbol{v}^\star(\infty)\}\Big)\\ &=\mathrm{vec}\{\boldsymbol{r}\boldsymbol{r}^\top\}^\top\frac{1}{N}(\boldsymbol{I}_{(NL)^2}-\boldsymbol{K})^{-1}\mathrm{vec}\{\boldsymbol{I}_{NL}\}\\ &-2\mathrm{vec}\{(\boldsymbol{B}\mathbb{E}\{\boldsymbol{v}^\star(\infty)\}\boldsymbol{r}^\top)^\top\}\frac{1}{N}(\boldsymbol{I}_{(NL)^2}-\boldsymbol{K})^{-1}\mathrm{vec}\{\boldsymbol{I}_{NL}\}\\ &=\frac{1}{N}\sum_{j=0}^{\infty}\mathrm{trace}\{\boldsymbol{B}^j[\boldsymbol{r}\boldsymbol{r}^\top-2\boldsymbol{B}\mathbb{E}\{\boldsymbol{v}^\star(\infty)\}\boldsymbol{r}^\top]\boldsymbol{B}^{j\top}\}\\ &\overset{(48)}{=}\frac{1}{N}\sum_{j=0}^{\infty}\mathrm{trace}\{\boldsymbol{B}^j[\boldsymbol{I}_{NL}+2\boldsymbol{B}(\boldsymbol{I}-\boldsymbol{B})^{-1}]\boldsymbol{r}\boldsymbol{r}^\top\boldsymbol{B}^{j\top}\}. \quad (74)\end{aligned}$$

Finally, we can express the steady-state MSD (67) as

$$\mathrm{MSD}^{\mathrm{network}}=\frac{L}{N}\sum_{k=1}^{N}\sigma_{\epsilon,k}^2 +\frac{1}{N}\sum_{j=0}^{\infty}\mathrm{trace}\Big\{\boldsymbol{B}^j(\boldsymbol{G}+(\boldsymbol{I}_{NL}+2\boldsymbol{B}(\boldsymbol{I}_{NL}-\boldsymbol{B})^{-1})\boldsymbol{r}\boldsymbol{r}^\top)\boldsymbol{B}^{j\top}\Big\}. \quad (75)$$

In the sequel, this formulation will allow us to compare the performance of different algorithms.

C. Performance Comparison With Non-Cooperative LMS

We shall now compare the performance of the ATC and CTA diffusion LMS algorithms with the non-cooperative LMS strategy when applied to a multitask network. We consider the case of uniform step-sizes, $\mu_k=\mu$, for a meaningful comparison. Diffusion LMS degenerates to non-cooperative LMS by setting

$$\boldsymbol{C}=\boldsymbol{I}_N,\quad \boldsymbol{A}_1=\boldsymbol{I}_N,\quad \boldsymbol{A}_2=\boldsymbol{I}_N \quad (76)$$

from which the performance of the latter can be easily derived. In this case, matrices $\boldsymbol{B}$ and $\boldsymbol{G}$ reduce to

$$\boldsymbol{B}_{\mathrm{lms}}=\boldsymbol{I}_{NL}-\mu\boldsymbol{H} \quad (77)$$

$$\boldsymbol{G}_{\mathrm{lms}}=\mu^2\,\mathrm{diag}\{\sigma_{z,1}^2\boldsymbol{R}_{x,1},\ldots,\sigma_{z,N}^2\boldsymbol{R}_{x,N}\} \quad (78)$$

where we use the subscript LMS for clarity. Note that in this case we have $\boldsymbol{r}_w(n)=0$. In addition, since $\mathcal{N}_k=\{k\}$ and $\boldsymbol{u}_{kk}^\star=0$, we have $\boldsymbol{r}_u(n)=0$. This implies that $\boldsymbol{r}=0$. The steady-state MSD for non-cooperative LMS is then given by:

$$\mathrm{MSD}_{\mathrm{lms}}^{\mathrm{network}}=\frac{1}{N}\sum_{j=0}^{\infty}\mathrm{trace}\left(\boldsymbol{B}_{\mathrm{lms}}^j\boldsymbol{G}_{\mathrm{lms}}\boldsymbol{B}_{\mathrm{lms}}^{j^\top}\right)+\frac{L}{N}\sum_{k=1}^{N}\sigma_{\epsilon,k}^2. \quad (79)$$

It is useful to note that the matrices $\boldsymbol{B}$ and $\boldsymbol{G}$ for diffusion LMS can be expressed in terms of $\boldsymbol{B}_{\mathrm{lms}}$ and $\boldsymbol{G}_{\mathrm{lms}}$:

$$\begin{aligned}\boldsymbol{B}&=\boldsymbol{\mathcal{A}}_2^\top(\boldsymbol{I}_{NL}-\mu\boldsymbol{H})\boldsymbol{\mathcal{A}}_1^\top=\boldsymbol{\mathcal{A}}_2^\top\boldsymbol{B}_{\mathrm{lms}}\boldsymbol{\mathcal{A}}_1^\top\\ \boldsymbol{G}&=\mu^2\boldsymbol{\mathcal{A}}_2^\top\boldsymbol{\mathcal{C}}^\top\mathrm{diag}\{\sigma_{z,1}^2\boldsymbol{R}_{x,1},\ldots,\sigma_{z,N}^2\boldsymbol{R}_{x,N}\}\boldsymbol{\mathcal{C}}\boldsymbol{\mathcal{A}}_2\\ &=\boldsymbol{\mathcal{A}}_2^\top\boldsymbol{\mathcal{C}}^\top\boldsymbol{G}_{\mathrm{lms}}\boldsymbol{\mathcal{C}}\boldsymbol{\mathcal{A}}_2 \quad (80)\end{aligned}$$

with $\boldsymbol{A}_1=\boldsymbol{I}_N$ for the ATC diffusion strategy, and $\boldsymbol{A}_2=\boldsymbol{I}_N$ for the CTA diffusion strategy. Using the series expansions for $\mathrm{MSD}^{\mathrm{network}}$ and $\mathrm{MSD}_{\mathrm{lms}}^{\mathrm{network}}$, the difference between the MSDs for non-cooperative LMS and diffusion LMS is given by

$$\begin{aligned}&\mathrm{MSD}_{\mathrm{lms}}^{\mathrm{network}}-\mathrm{MSD}^{\mathrm{network}}=\\ &\frac{1}{N}\sum_{j=0}^{\infty}\mathrm{trace}\left\{\boldsymbol{B}_{\mathrm{lms}}^j\boldsymbol{G}_{\mathrm{lms}}\boldsymbol{B}_{\mathrm{lms}}^{j^\top}-\boldsymbol{B}^j\boldsymbol{G}\boldsymbol{B}^{j^\top}\right\}-\\ &\frac{1}{N}\sum_{j=0}^{\infty}\mathrm{trace}\left\{\boldsymbol{B}^j(\boldsymbol{I}_{NL}+2\boldsymbol{B}(\boldsymbol{I}_{NL}-\boldsymbol{B})^{-1})\boldsymbol{r}\boldsymbol{r}^\top\boldsymbol{B}^{j^\top}\right\}. \quad (81)\end{aligned}$$

Note that the first term is the difference in performance between the non-cooperative LMS strategy and the cooperative diffusion strategy. It was first analyzed in [43] and, because it is not specific to the multitask context, it is denoted by $\Delta\mathrm{MSD}^{\mathrm{network}}$. Only the second term, which depends on $\boldsymbol{r}$, is specific to the multitask scenario. Thus, it is denoted by $\Delta\mathrm{MSD}_{\mathrm{multi}}^{\mathrm{network}}(\boldsymbol{r})$. Therefore,

$$\mathrm{MSD}_{\mathrm{lms}}^{\mathrm{network}}-\mathrm{MSD}^{\mathrm{network}}=\Delta\mathrm{MSD}^{\mathrm{network}}-\Delta\mathrm{MSD}_{\mathrm{multi}}^{\mathrm{network}}(\boldsymbol{r}). \quad (82)$$

In order to obtain analytical results that allow some understanding of the algorithm behavior, we further assume that the matrices $\boldsymbol{C}$, $\boldsymbol{A}_1$ and $\boldsymbol{A}_2$ in the diffusion implementation are doubly stochastic, and the regression covariance matrices are uniform across the agents, that is, $\boldsymbol{R}_{x,k}=\boldsymbol{R}_x$. With these assumptions, it was shown in [43, Sec. 7] that the first term $\Delta\mathrm{MSD}^{\mathrm{network}}$ on the RHS of (81) is always nonnegative, namely,

$$\mathrm{trace}\left\{\boldsymbol{B}_{\mathrm{lms}}^j\boldsymbol{G}_{\mathrm{lms}}\boldsymbol{B}_{\mathrm{lms}}^{j^\top}-\boldsymbol{B}^j\boldsymbol{G}\boldsymbol{B}^{j^\top}\right\}\geq 0. \quad (83)$$

We need to check under which conditions the second term $\Delta\mathrm{MSD}_{\mathrm{multi}}^{\mathrm{network}}(\boldsymbol{r})$ on the RHS of equation (81) is nonnegative so that it can be viewed as a degradation factor caused by the multitask scenario. Introduce the symmetric matrix $\boldsymbol{Z}=\boldsymbol{I}_{NL}+2\boldsymbol{B}(\boldsymbol{I}_{NL}-\boldsymbol{B})^{-1}=2(\boldsymbol{I}_{NL}-\boldsymbol{B})^{-1}-\boldsymbol{I}_{NL}$. We find that

$$\Delta\mathrm{MSD}_{\mathrm{multi}}^{\mathrm{network}}(\boldsymbol{r})=\frac{1}{N}\boldsymbol{r}^\top(\boldsymbol{I}_{NL}-\boldsymbol{B}^\top\boldsymbol{B})^{-1}\boldsymbol{Z}\boldsymbol{r}. \quad (84)$$

We conclude that expression (84) is non-negative for all $\boldsymbol{r}$ if, and only if, $(\boldsymbol{I}_{NL}-\boldsymbol{B}^\top\boldsymbol{B})^{-1}\boldsymbol{Z}$ is a symmetric positive semidefinite matrix. Now, we show that this condition is met for a large class of information exchange protocols. Assume, for instance, that either $\boldsymbol{A}_1$ or $\boldsymbol{A}_2$ is symmetric, depending on whether the focus is on the CTA or ATC strategy. Recalling conditions $\boldsymbol{R}_{x,k}=\boldsymbol{R}_x$ and $\mu_k=\mu$ for uniform data profile, it then holds that $\boldsymbol{B}$ is a symmetric matrix. It can be further verified that $(\boldsymbol{I}_{NL}-\boldsymbol{B}^2)^{-1}$ and $\boldsymbol{Z}$ are positive definite when $\boldsymbol{B}$ is stable. Now, we

verify that the product $(\boldsymbol{I}_{NL}-\boldsymbol{B}^2)^{-1}\boldsymbol{Z}$ is a symmetric positive semidefinite matrix.

Lemma 1 (Positivity of a Matrix Product): [49, Fact 8.10.11] Given two symmetric positive semidefinite matrices $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ with compatible sizes. Then, $\boldsymbol{X}_1\boldsymbol{X}_2$ is symmetric positive semidefinite if, and only if, $\boldsymbol{X}_1\boldsymbol{X}_2$ is normal, that is, if it satisfies: $(\boldsymbol{X}_1\boldsymbol{X}_2)(\boldsymbol{X}_1\boldsymbol{X}_2)^\top=(\boldsymbol{X}_1\boldsymbol{X}_2)^\top(\boldsymbol{X}_1\boldsymbol{X}_2)$. ■

By setting $\boldsymbol{X}_1=(\boldsymbol{I}_{NL}-\boldsymbol{B}^2)^{-1}$ and $\boldsymbol{X}_2=\boldsymbol{Z}$, observe that $\boldsymbol{X}_1\boldsymbol{X}_2$ is symmetric. It then holds that $(\boldsymbol{I}_{NL}-\boldsymbol{B}^2)^{-1}\boldsymbol{Z}$ is normal. By Lemma 1, $(\boldsymbol{I}_{NL}-\boldsymbol{B}^2)^{-1}\boldsymbol{Z}$ is a symmetric positive semidefinite matrix, which means that $\Delta\text{MSD}^{\text{network}}_{\text{multi}}(\boldsymbol{r})$ is nonnegative under the conditions specified above. It follows that this term can be viewed as a degradation factor caused by the cooperation of nodes performing different estimation tasks, which can be expressed as $\frac{1}{N}\|\boldsymbol{r}\|^2_{(\boldsymbol{I}_{NL}-\boldsymbol{B}^2)^{-1}\boldsymbol{Z}}$. We summarize the results in the following statement.

Theorem 3 (Non-Cooperative vs. Cooperative Strategies): Consider the same setting of Theorems 1 and 2, with the additional requirement that the conditions for a uniform data profile hold. The adaptive ATC or CTA diffusion strategies outperform the non-cooperative strategy if, and only if,

$$\Delta\text{MSD}^{\text{network}}-\Delta\text{MSD}^{\text{network}}_{\text{multi}}(\boldsymbol{r})\geq 0. \tag{85}$$

Given doubly stochastic $\boldsymbol{A}$ and $\boldsymbol{C}$, the gain $\Delta\text{MSD}^{\text{network}}$ in performance between the cooperative diffusion strategy and the non-cooperative LMS strategy, which is independent of $\boldsymbol{r}$, is nonnegative. Furthermore, by assuming that $\boldsymbol{A}$ is symmetric, then the degradation in performance $\Delta\text{MSD}^{\text{network}}_{\text{multi}}(\boldsymbol{r})$ caused by the multitask environment is positive. It is given by

$$\Delta\text{MSD}^{\text{network}}_{\text{multi}}(\boldsymbol{r})=\frac{1}{N}\|\boldsymbol{r}\|^2_{(\boldsymbol{I}_{NL}-\boldsymbol{B}^2)^{-1}\boldsymbol{Z}} \tag{86}$$

where $\boldsymbol{Z}=2(\boldsymbol{I}_{NL}-\boldsymbol{B})^{-1}-\boldsymbol{I}_{NL}$. ■

Although condition (85) allows to determine whether using the diffusion LMS is beneficial for multitask learning compared to the non-cooperative LMS strategy, it cannot be easily exploited to estimate appropriate combination coefficients because of its complexity and the need to handle dynamic problems. The aim of the next section is to derive an efficient strategy to estimate these coefficients.

IV. Node Clustering via Combination Matrix Selection

We now derive a clustering strategy where each node k can adjust the combination weights $a_{\ell k}$ in an online manner, for $\ell\in\mathcal{N}_k$, in order to adapt to multitask environments. It is sufficient to focus on the adapt-then-combine diffusion LMS defined by steps (6) and (7). For ease of presentation, the corresponding algorithm is summarized below:

$$\begin{cases}\boldsymbol{\psi}_k(n+1)=\boldsymbol{w}_k(n)+\mu_k\sum\limits_{\ell\in\mathcal{N}_k}c_{\ell k}\boldsymbol{x}_\ell(n)\big[d_\ell(n)-\boldsymbol{x}_\ell^\top(n)\boldsymbol{w}_k(n)\big]\\ \boldsymbol{w}_k(n+1)=\sum\limits_{\ell\in\mathcal{N}_k}a_{\ell k}\boldsymbol{\psi}_\ell(n+1)\end{cases} \tag{87}$$

where $a_{\ell k}$ is used instead of $a_{2,\ell k}$. As shown in the previous section, running (87) in a multitask environment leads to biased results. We now discuss how to cluster nodes in order to reduce this effect.

A. Clustering via Matrix $\boldsymbol{A}$ Adjustments

Following [42], we suggest to adjust matrix $\boldsymbol{A}$ in an online manner via MSD optimization. At each instant n, the instantaneous MSD at node k is given by

$$\mathbb{E}\{\|\boldsymbol{v}_k^\star(n+1)\|^2\}=E\Big\{\|\boldsymbol{w}_k^\star-\sum_{\ell\in\mathcal{N}_k}a_{\ell k}\boldsymbol{\psi}_\ell(n+1)\|^2\Big\}. \tag{88}$$

Computation of this quantity requires the knowledge of $\boldsymbol{w}_k^\star$. Because the matrix $\boldsymbol{A}$ is assumed left-stochastic, this expression can be rewritten as

$$\mathbb{E}\{\|\boldsymbol{v}_k^\star(n+1)\|^2\}=\sum_{\ell\in\mathcal{N}_k}\sum_{p\in\mathcal{N}_k}a_{\ell k}a_{pk}E\left\{[\boldsymbol{w}_k^\star-\boldsymbol{\psi}_\ell(n+1)]^\top[\boldsymbol{w}_k^\star-\boldsymbol{\psi}_p(n+1)]\right\}. \tag{89}$$

Let $\boldsymbol{\Psi}_k$ be the matrix at each node k with (ℓ,p)-th entry defined as

$$[\boldsymbol{\Psi}_k]_{\ell p}=\begin{cases}\mathbb{E}\big\{[\boldsymbol{w}_k^\star-\boldsymbol{\psi}_\ell(n+1)]^\top & [\boldsymbol{w}_k^\star-\boldsymbol{\psi}_p(n+1)]\big\},\\ & \ell,p\in\mathcal{N}_k\\ 0, & \text{otherwise}\end{cases}. \tag{90}$$

Let $\boldsymbol{a}_k=[a_{1k},\ldots,a_{Nk}]^\top$. Minimizing (89) for node k at time n, subject to left-stochasticity of $\boldsymbol{A}$ and $a_{\ell k}=0$ for $\ell\notin\mathcal{N}_k$, can be formulated as follows:

$$\begin{aligned}\min_{\boldsymbol{a}_k}\quad&\boldsymbol{a}_k^\top\boldsymbol{\Psi}_k\boldsymbol{a}_k\\ \text{subject to}\quad&\boldsymbol{1}_N^\top\boldsymbol{a}_k=1,\quad a_{\ell k}\geq 0,\\ &a_{\ell k}=0\text{ if }\ell\notin\mathcal{N}_k.\end{aligned} \tag{91}$$

Generally, it is not possible to solve this problem at each node k since $\boldsymbol{w}_k^\star$ and $\boldsymbol{\Psi}_k$ are unknown. We suggest to use an approximation for $\boldsymbol{w}_k^\star$, to approximate matrix $\boldsymbol{\Psi}_k$ by an instantaneous value, and to drop its off-diagonal entries in order to make the problem tractable and have a closed-form solution (see (93)). The resulting problem is as follows:

$$\begin{aligned}\min_{\boldsymbol{a}_k}\quad&\sum_{\ell=1}^N a_{\ell k}^2\|\widehat{\boldsymbol{w}}_k^\star-\boldsymbol{\psi}_\ell(n+1))\|^2\\ \text{subject to}\quad&\boldsymbol{1}_N^\top\boldsymbol{a}_k=1,\quad a_{\ell k}\geq 0,\\ &a_{\ell k}=0\text{ if }\ell\notin\mathcal{N}_k\end{aligned} \tag{92}$$

with $\widehat{\boldsymbol{w}}_k^\star$ some approximation for $\boldsymbol{w}_k^\star$. The objective function shown above has the natural interpretation of penalizing the combination weight $a_{\ell k}$ assigned by node ℓ to node k if the local estimate at node ℓ is far from the objective at node k. The solution to this problem is given by[2]

$$a_{\ell k}(n+1)=\frac{\|\widehat{\boldsymbol{w}}_k^\star-\boldsymbol{\psi}_\ell(n+1))\|^{-2}}{\sum_{j\in\mathcal{N}_k}\|\widehat{\boldsymbol{w}}_k^\star-\boldsymbol{\psi}_j(n+1))\|^{-2}},\text{for }\ell\in\mathcal{N}_k. \tag{93}$$

[2]To achieve this result, discard the non-negativity constraint first, and write the Lagrangian function with respect to the equality constraint only. The solution to this simplified problem is given by (93). Observe that it satisfies the non-negativity constraint $a_{\ell k}\geq 0$. Consequently, (93) is also the solution to problem (92).

Let us now construct an approximation for $\boldsymbol{w}_k^\star$ to be used in (93). In order to reduce the MSD bias that results from the cooperation of nodes performing distinct estimation tasks, one strategy is to use the local one-step approximation:

$$\widehat{\boldsymbol{w}}_k^\star(n+1) = \boldsymbol{\psi}_k(n+1) - \mu_k \nabla J_k(\boldsymbol{w})\big|_{\boldsymbol{w}=\boldsymbol{\psi}_k(n+1)}. \quad (94)$$

Since the true gradient of $J_k(\boldsymbol{w})$ at $\boldsymbol{\psi}_k(n+1)$ is not available in an adaptive implementation, we can approximate it by using the instantaneous value $\boldsymbol{q}_k(n) \triangleq e_k(n)\boldsymbol{x}_k(n)$ with $e_k(n) = [d_k(n) - \boldsymbol{x}_k^\top(n)\boldsymbol{\psi}_k(n+1)]$. This yields the following approximation:

$$\widehat{\boldsymbol{w}}_k^\star(n+1) = \boldsymbol{\psi}_k(n+1) + \mu_k \boldsymbol{q}_k(n). \quad (95)$$

Substituting this expression into (93), we get the combination rule

$$a_{\ell k}(n+1) = \frac{\|\boldsymbol{\psi}_k(n+1) + \mu_k \boldsymbol{q}_k(n) - \boldsymbol{\psi}_\ell(n+1)\|^{-2}}{\sum_{j\in\mathcal{N}_k} \|\boldsymbol{\psi}_k(n+1) + \mu_k\, \boldsymbol{q}_k(n) - \boldsymbol{\psi}_j(n+1)\|^{-2}}, \quad \text{for } \ell \in \mathcal{N}_k. \quad (96)$$

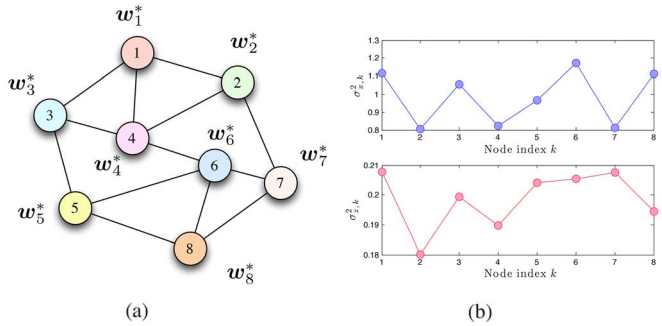
This rule admits a useful interpretation. On the one hand, as mentioned above, it relies on the local estimate (94) in order to reduce the MSD bias effect caused by the cooperation of neighboring nodes estimating distinct parameter vectors. On the other hand, consider the inverse of the numerator of rule (96):

$$\begin{aligned}&\|\boldsymbol{\psi}_k(n+1) + \mu_k \boldsymbol{q}_k(n) - \boldsymbol{\psi}_\ell(n+1)\|^2 \\ &\quad = \|\boldsymbol{\psi}_\ell(n+1) - \boldsymbol{\psi}_k(n+1)\|^2 \\ &\quad + 2[\boldsymbol{\psi}_\ell(n+1) - \boldsymbol{\psi}_k(n+1)]^\top [-\mu_k \boldsymbol{q}_k(n)] + \mu_k^2 \|\boldsymbol{q}_k(n)\|^2. \quad (97)\end{aligned}$$

The first term $\|\boldsymbol{\psi}_\ell(n+1) - \boldsymbol{\psi}_k(n+1)\|^2$ on the RHS accounts for the distance of the current estimates between nodes k and ℓ; this term tends to decrease the combination weight $a_{\ell k}(n+1)$ if this distance is large, and to limit information exchange. Now, consider the first-order Taylor series expansion of $J_k(\boldsymbol{w})$ at $\boldsymbol{\psi}_k(n+1)$:

$$J_k(\boldsymbol{\psi}) \approx J_k(\boldsymbol{\psi}_k(n+1)) - [\boldsymbol{\psi} - \boldsymbol{\psi}_k(n+1)]^\top \boldsymbol{q}_k(n). \quad (98)$$

The second term $[\boldsymbol{\psi}_\ell(n+1) - \boldsymbol{\psi}_k(n+1)]^\top [-\mu_k \boldsymbol{q}_k(n)]$ on the RHS of (97) is proportional to $J_k(\boldsymbol{\psi}_\ell(n+1)) - J_k(\boldsymbol{\psi}_k(n+1))$. This term also tends to decrease the combination weight $a_{\ell k}(n+1)$ if $J_k(\boldsymbol{\psi}_\ell(n+1)) > J_k(\boldsymbol{\psi}_k(n+1))$. Indeed, in this case, it is not recommended to promote the combination of models $\boldsymbol{\psi}_k(n+1)$ and $\boldsymbol{\psi}_\ell(n+1)$ because the latter induces an increase of the cost function. Finally, $\mu_k^2 \|\boldsymbol{q}_k(n)\|^2$ is the same for all $\ell \in \mathcal{N}_k$. To summarize this discussion, the combination rule (96) considers the closeness of the local estimate to the neighboring estimates, and the local slope of the cost function, to adjust the combination weights. This tends to promote information exchange between nodes that estimate the same optimum parameter vector, and thus to reduce the MSD bias and improve the estimation accuracy.



(a) (b)

Fig. 1. (a) Network studied in Section V-A, with 8 nodes. (b) Input signal and noise variances for each sensor node. (a) Network topology. (b) Input variances (top) and noise variances (bottom).

Algorithm 1: ATC Diffusion LMS With Adaptive Clustering for Multitask Problems

Initialization: Set $\boldsymbol{C}(0) = \boldsymbol{I}_N$ and $\boldsymbol{A}(0) = \boldsymbol{I}_N$.

Set $\boldsymbol{w}_k(0) = 0$ for all $k = 1, \ldots, N$.

Algorithm: At each time instant $n \geq 1$, and for each node k, update $\boldsymbol{\psi}_k(n+1)$:

$$\boldsymbol{\psi}_k(n+1) = \boldsymbol{w}_k(n) + \mu_k \sum_{\ell\in\mathcal{N}_k} c_{\ell k}(n) \big[d_\ell(n) - \boldsymbol{x}_\ell^\top(n)\boldsymbol{w}_k(n)\big]\boldsymbol{x}_\ell(n). \quad (100)$$

Update the combination coefficients:

$$\boldsymbol{q}_k(n) = [d_k(n) - \boldsymbol{x}_k^\top(n)\boldsymbol{\psi}_k(n+1)]\boldsymbol{x}_k(n).$$

Optional: normalize $\boldsymbol{q}_k(n)$, i.e., use $\boldsymbol{q}_k(n)/(\|\boldsymbol{q}_k(n)\| + \xi)$

$$a_{\ell k}(n+1) = \frac{\|\boldsymbol{\psi}_k(n+1) + \mu_k \boldsymbol{q}_k(n) - \boldsymbol{\psi}_\ell(n+1))\|^{-2}}{\sum_{j\in\mathcal{N}_k} \|\boldsymbol{\psi}_k(n+1) + \mu_k \boldsymbol{q}_k(n) - \boldsymbol{\psi}_j(n+1))\|^{-2}}. \quad (101)$$

Optional: $c_{k\ell}(n+1) = a_{\ell k}(n+1)$

Combine weights:

$$\boldsymbol{w}_k(n+1) = \sum_{\ell\in\mathcal{N}_k} a_{\ell k}(n+1)\boldsymbol{\psi}_k(n+1). \quad (102)$$

Algorithm

The flexibility of multitask networks may be exploited by considering distinct cost functions for each node. This raises the issue of sharing information via the exchange matrix $\boldsymbol{C}$, which can be simply set to the identity. However, the time-variant combination matrix $\boldsymbol{A}(n)$ determined by (96) describes how each agent combines the parameter vectors transmitted by its neighbors as a function of the estimated contrast between tasks. An additional way to exploit this information is that each agent uses the reciprocity principle defined by

$$\boldsymbol{C}(n+1) = \boldsymbol{A}^\top(n+1). \quad (99)$$

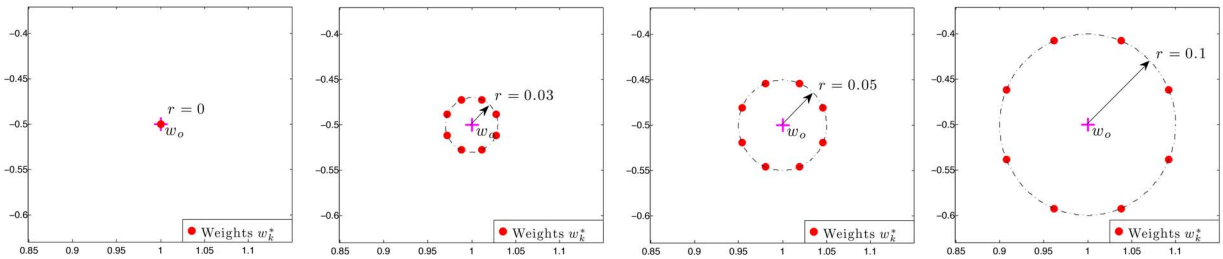


Fig. 2. Node coefficients centered at $\boldsymbol{w}_o = [1, -0.5]^\top$ with $r = 0$, $r = 0.03$, $r = 0.05$ and $r = 0.1$ (from left to right).

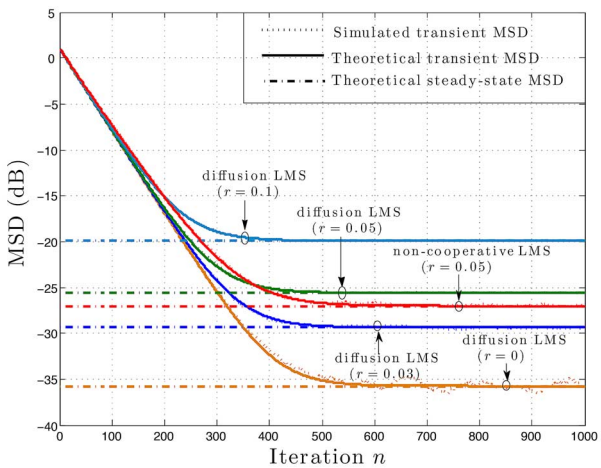


Fig. 3. Network MSD behavior for the deterministic case. Theoretical MSD curves were obtained by Corollary 1 and steady-state MSD values were obtained by Corollary 2. Simulated and theoretical transient MSD curves are perfectly superimposed.

The rationale underlying this principle is that the magnitude of $a_{\ell k}$ reflects the similarity of the estimation tasks performed by nodes k and ℓ, as it is perceived by node k. It is reasonable that node ℓ should use this information, and scale the local cost function accordingly. The smaller $a_{\ell k}$ is, the smaller $c_{k\ell}$ should be because nodes k and ℓ do not address the same estimation problem. Other strategies, in the spirit of (96), may be considered to estimate the coefficients $c_{k\ell}$. Moreover, we found that using the normalized gradient $\boldsymbol{q}_k(n)/(\|\boldsymbol{q}_k(n)\| + \xi)$, with ξ a small positive number to avoid division by zero, prevents premature convergence due to over-corrections. The ATC diffusion algorithm with adaptive clustering defined by time-variant combination matrices $\boldsymbol{A}(n)$ and $\boldsymbol{C}(n)$ is summarized in Algorithm 1. Considering that no prior information on clusters is available, we suggest to initialize the combination matrices $\boldsymbol{A}(0)$ and $\boldsymbol{C}(0)$ with $\boldsymbol{I}_N$. During simulations, we did not experience convergence issues with other initial settings, provided that $\boldsymbol{A}(0)$ and $\boldsymbol{C}(0)$ are left-stochastic and right-stochastic, respectively. Further analysis can help guide more informed choices for the combination policies.

V. Simulations

In this section, we report simulation results that validate the algorithm and the theoretical results. The ATC diffusion LMS algorithm is considered. All nodes were initialized with zero parameter vectors $\boldsymbol{w}_k(0)$. All simulated curves were obtained by averaging over 100 runs, since this gave sufficiently smooth curves to check consistency with theoretical results[3].

A. Model Validation

For the validation, we consider a network consisting of 8 nodes with interconnections shown in Fig. 1(a). The parameter vectors to be estimated are of length $L = 2$. The optimum mean vectors are uniformly distributed on a circle of radius r centered at $\boldsymbol{w}_o$, that is,

$$\begin{aligned} \boldsymbol{w}_k^\star &= \boldsymbol{w}_o + r \begin{pmatrix} \cos\theta_k \\ \sin\theta_k \end{pmatrix} \\ \theta_k &= 2\pi(k-1)/N + \pi/8. \end{aligned} \tag{103}$$

The regression inputs $\boldsymbol{x}_k(n)$ were zero-mean 2×1 random vectors governed by a Gaussian distribution with covariance matrices $\boldsymbol{R}_{x,k} = \sigma_{x,k}^2 \boldsymbol{I}_L$. The background noises $z_k(n)$ were i.i.d. zero-mean Gaussian random variables, and independent of any other signal. The variances $\sigma_{x,k}^2$ and $\sigma_{z,k}^2$ are depicted in Fig. 1(b). We considered the ATC diffusion LMS with measurement diffusion governed by a uniform matrix $\boldsymbol{C}$ such that $c_{\ell k} = |\mathcal{N}_\ell|^{-1}$ for all $k \in \mathcal{N}_\ell$. The combination matrix $\boldsymbol{A}$ simply averaged the estimates from the neighbors, namely, $a_{\ell k} = |\mathcal{N}_k|^{-1}$ for $\ell \in \mathcal{N}_k$. For all nodes, the step-sizes were set to $\mu_k = 0.01$.

1) Stationary Optimums: We first check the convergence analysis with stationary parameter vectors, that is, $\sigma_{\epsilon,k}^2 = 0$ for all nodes. Four groups of coefficient vectors, centered at $\boldsymbol{w}_o = [1, -0.5]^\top$ with $r = 0$, $r = 0.03$, $r = 0.05$ and $r = 0.1$ were considered, as illustrated in Fig. 2. Note that the case $r = 0$ corresponds to the single-task network where $\boldsymbol{w}_k^\star = \boldsymbol{w}_o$ for each node. Running ATC diffusion LMS with these four settings, we obtained the MSD curves shown in Fig. 3. Observe that the theoretical and simulated transient MSD curves are accurately superimposed. The non-cooperative LMS algorithm was also considered. Since the average steady-state MSD of the non-cooperative LMS algorithm over all nodes is approximately given by [45], [46]:

$$\mathrm{MSD}_{\mathrm{lms}}^{\mathrm{network}} = \frac{1}{N} \sum_{k=1}^{N} \frac{\mu_k \sigma_{z,k}^2 L}{2}, \tag{104}$$

then the MSD behavior with the different settings is almost the same, provided the other parameters remain unchanged. Consequently, the theoretical MSD curve for the non-cooperative LMS algorithm is only provided for $r = 0.05$. It can be observed that diffusion LMS can still be advantageous over non-cooperative LMS if the differences between local optimum weight

[3]Matlab source code is available at http://www.jie-chen.com.

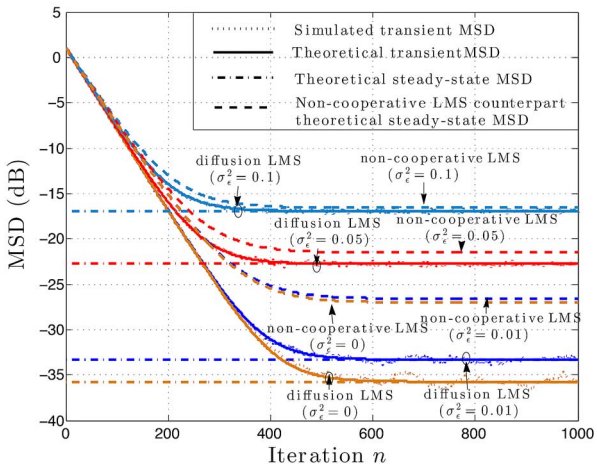


Fig. 4. Network MSD behavior for the perturbation-only case. Theoretical MSD curves were obtained by Corollary 1 and steady-state MSD values were obtained by Corollary 2. Note that simulated and theoretical transient MSD curves are superimposed.

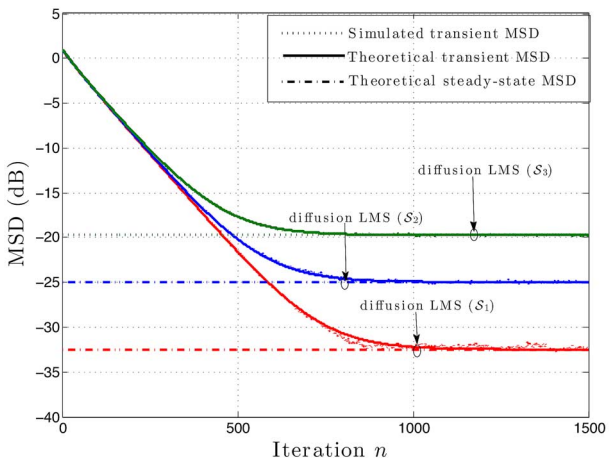


Fig. 5. Network MSD behavior for correlated inputs. Theoretical MSD curves were obtained by Corollary 1 and steady-state MSD values were obtained by Corollary 2. Simulated curves and theoretical curves are accurately superimposed.

vectors are sufficiently small, $r = 0$ and $r = 0.03$ in this simulation. However, when the contrast between the tasks increases, diffusion LMS provides lower performance than non-cooperative LMS due to the bias introduced by the algorithm, $r = 0.05$ and $r = 0.1$ in this simulation.

2) Randomly Perturbed Optimums: We now consider the network described previously with $r = 0$ so that the differences between the optimum weight vectors $\boldsymbol{w}_k^\star(n)$ arise from the random perturbations $\boldsymbol{\epsilon}_k(n)$. Keeping all the other parameters unchanged, the variance of these perturbations was successively set to $\sigma_\epsilon^2 = 0$, 0.01, 0.05 and 0.1 for all the agents. MSD curves for diffusion LMS and non-cooperative LMS are provided in Fig. 4. It can be observed that diffusion LMS always outperformed its non-cooperative counterpart. This experiment shows the advantage provided by cooperation. The relative performance gain becomes smaller as σ_ϵ^2 increases because weight lags caused by random perturbations dominate the estimation error.

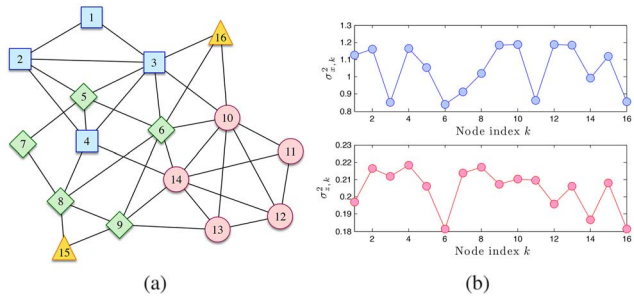


Fig. 6. Network topology in Section V-B1 and associated input variances and noise variances. (a) Network topology (b) Input variances (top) and noise variances (bottom).

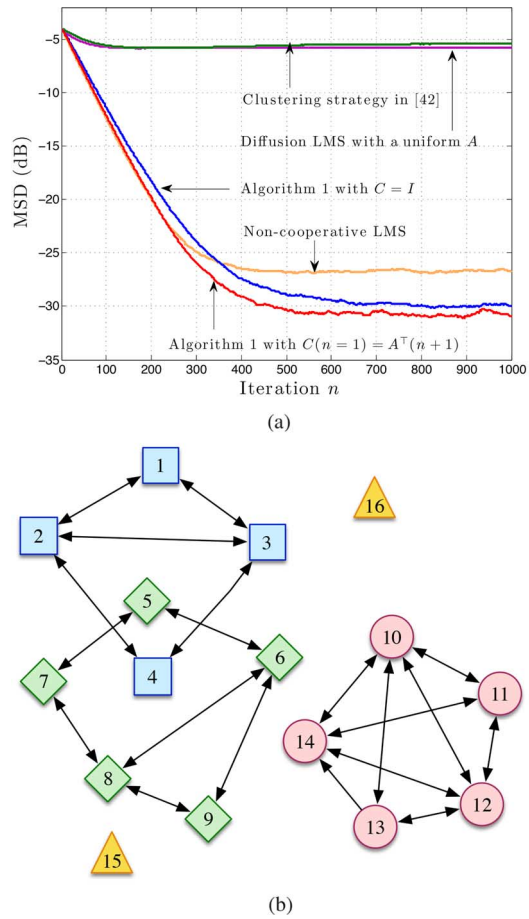


Fig. 7. Network MSD comparison in a stationary multitask environment, and estimated cluster structure by the proposed algorithm (averaged over the last 100 instants in one realization). The weight iterates for [42] were initialized at the same value. If random well-separated initial conditions are used across the nodes, then the performance of [42] becomes similar to that of the non-cooperative solution in the above plot. (a) MSD behavior. (b) Estimated cluster structure.

3) Correlated in Time Inputs: This simulation example illustrates the accuracy of models (62)–(67) for inputs correlated in time. We considered regression vectors

$$\boldsymbol{x}_k(n) = [x_k(n) \;\; x_k(n-1)]^\top \tag{105}$$

with a first-order AR model given by

$$x_k(n) = 0.5x_k(n-1) + \sqrt{(1-0.5^2)\sigma_{x,k}^2}\, w_k(n). \tag{106}$$

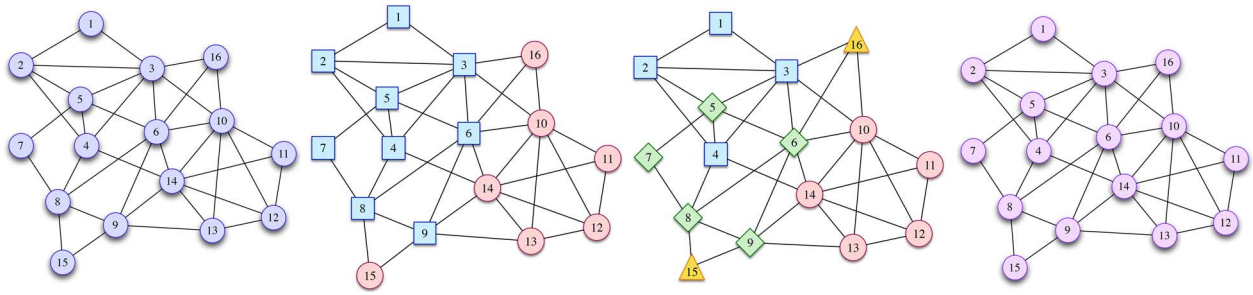


Fig. 8. Evolution of cluster structures of the network (1 cluster → 2 clusters → 4 clusters → 1 cluster).

The parameters $\sigma_{x,k}^2$ were set as in Fig. 1(b). The noise $w_k(n)$ was i.i.d. and drawn from a zero-mean Gaussian distribution with variance $\sigma_w^2 = 1$, so that

$$\boldsymbol{R}_{x,k} = \sigma_{x,k}^2 \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}.$$

The diffusion LMS algorithm was tested in the following experimental settings:

$$\begin{aligned} \mathcal{S}_1 &: \{r = 0.01, \sigma_\epsilon = 0.010\} \\ \mathcal{S}_2 &: \{r = 0.05, \sigma_\epsilon = 0.015\} \\ \mathcal{S}_3 &: \{r = 0.10, \sigma_\epsilon = 0.015\}. \end{aligned} \tag{107}$$

Although Assumption 1 is not valid, observe in Fig. 5 that the theoretical and simulated transient MSD curves are superimposed. This illustrates the accuracy of the analysis when the step-sizes are sufficiently small.

B. Adaptive Clustering in Multitask Networks

We shall now illustrate the performance of diffusion LMS with adaptive clustering in a multitask environment. Our approach is compared with the strategy introduced in [42]. For the latter, as suggested in [42], the so-called smoothing factor γ was set to 0.1. A stationary problem is first considered. Next, a dynamic problem with time-varying clusters is introduced in order to confirm the reliability of our approach.

1) Stationary Environment: Consider the network of 16 agents depicted in Fig. 6(a). The regression inputs $\boldsymbol{x}_k(n)$ were zero-mean 2×1 random vectors governed by a Gaussian distribution with covariance matrices $\boldsymbol{R}_{x,k} = \sigma_{x,k}^2 \boldsymbol{I}_L$. The background noises $z_k(n)$ were i.i.d. zero-mean Gaussian random variables, independent of any other signals. The variances $\sigma_{x,k}^2$ and $\sigma_{z,k}^2$ are depicted in Fig. 6(b). The scenario under study is a multitask problem with a cluster structure. Nodes 1 to 4 belong to the first cluster. Nodes 5 to 9 are in the second cluster. Nodes 10 to 14 compose the third cluster, and nodes 15 and 16 are in the fourth cluster. The parameter vectors to be estimated are as follows:

$$\boldsymbol{w}_k^\star = \begin{cases} [0.5 \; -0.4]^\top & k = 1, \ldots, 4 \quad \text{Cluster 1} \\ [0.6 \; -0.2]^\top & k = 5, \ldots, 9 \quad \text{Cluster 2} \\ [0.3 \; -0.3]^\top & k = 10, \ldots, 14 \quad \text{Cluster 3} \\ [-0.8 \; 0.5]^\top & k = 15, 16 \quad \text{Cluster 4.} \end{cases} \tag{108}$$

Note that the distances between the optimum parameter vectors for clusters 1, 2 and 3 are much smaller than those with respect to cluster 4, which acts as an outlier. The following algorithms were considered for estimating the four optimum parameter vectors: 1) diffusion LMS with a uniform combination matrix $\boldsymbol{A}$, 2) non-cooperative LMS, 3) diffusion LMS with the clustering strategy introduced in [42], 4) diffusion LMS with our clustering strategy, with $\boldsymbol{C} = \boldsymbol{I}$ and $\boldsymbol{C}(n) = \boldsymbol{A}^\top(n)$. The step-size was set to $\mu = 0.01$ for all nodes.

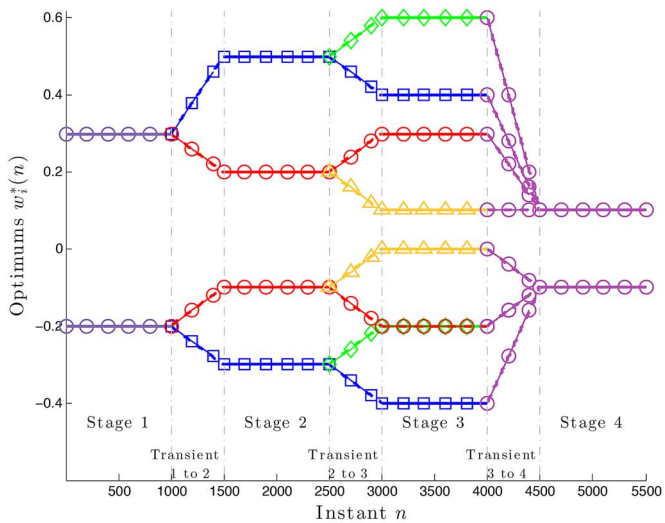


Fig. 9. Evolution of clusters over time. Colors and markers are consistent with those of clusters in Fig. 8. Dashed lines represent optimums during transient episodes.

Fig. 7(a) illustrates the MSD convergence behavior for these algorithms. Due to large bias of the estimated weights, diffusion LMS with a uniform combination matrix had large MSD. Non-cooperative LMS performed much better as it provides unbiased estimates. The proposed algorithm with $\boldsymbol{C} = \boldsymbol{I}$ achieved better performance, and $\boldsymbol{C}(n) = \boldsymbol{A}^\top(n)$ led to additional performance gain due to information exchange. Finally, in order to provide a straightforward but visually-meaningful clustering result, we averaged the combination matrix $\boldsymbol{A}$ over the last 100 iterations of a single realization, and we considered that $a_{\ell k} > 0.05$ represents a one-way connection from ℓ to k. The estimated relationships between nodes provided in Fig. 7(b) perfectly match the ground truth configuration.

2) Non-Stationary Environment: Consider now a more complex environment where clusters vary over time. Four stationary stages and three transient episodes were modeled in this experiment. Properties of input signals and noise were the same as those in the stationary case considered above. From instant $n = 1$ to 1000, the network consisted of one cluster with a unique optimum parameter vector to estimate. From $n = 1501$ to 2500, nodes were split into two clusters with two different optimums. From $n = 3001$ to 4000, nodes were split again to give

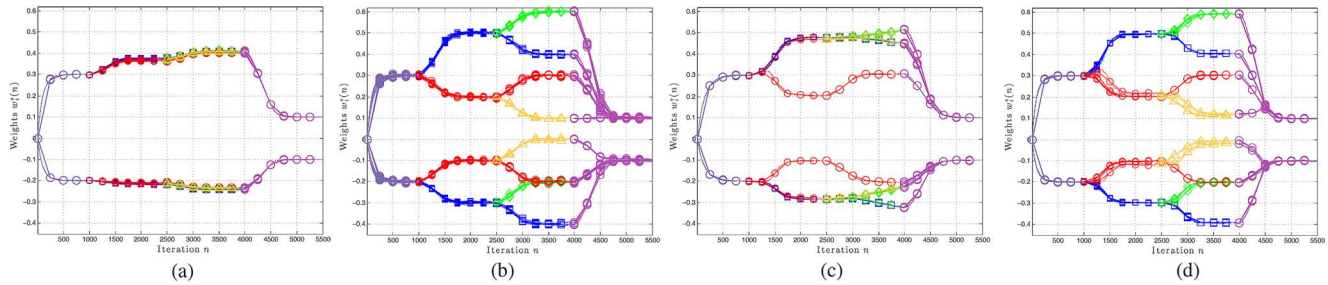


Fig. 10. Mean weight behavior of various algorithms in the non-stationary environment. Colors and markers are consistent with cluster colors in Fig. 6(a). (a) Diffusion LMS with uniform $\boldsymbol{A}$, $\boldsymbol{C}$. (b) Non-cooperative LMS. (c) Algorithm in [42]. (d) Proposed with $\boldsymbol{C}(n) = \boldsymbol{A}^{\top}(n)$.

four clusters. Finally, from instant $n = 4501$, nodes were aggregated into one cluster with another unique parameter vector to estimate. Transient episodes were designed with linear interpolation between each steady-state stage over a period of 500 time samples. Taking, for example, the first component of the weight vector of node 1 over the time interval 1 to 2500, the time-variant optimum $w^{\star}_{1,1}(n)$ is expressed by

$$w^{\star}_{1,1}(n) = \begin{cases} 0.3, & n = 1, \ldots, 1000 \\ 0.3 + \frac{0.5-0.3}{500}(n-1000), & n = 1001, \ldots, 1500 \\ 0.5, & n = 1501, \ldots, 2500. \end{cases} \tag{109}$$

Cluster structures and optimum parameter vectors are illustrated in Figs. 8 and 9, respectively.

The same four algorithms as before were considered for comparison. Fig. 10 shows their mean weight behavior. Conventional diffusion LMS with a uniform combination matrix made all nodes converge to the same Pareto optimum during all phases. The non-cooperative LMS estimated the optimum weight vectors without bias. The algorithm presented in [42], which generally performs well for well-separated tasks and well-separated initial random weights, did not perform well with this setting. Our algorithm showed the same convergence behavior for $\boldsymbol{C} = \boldsymbol{I}$ and $\boldsymbol{C}(n) = \boldsymbol{A}^{\top}(n)$. Only the case $\boldsymbol{C}(n) = \boldsymbol{A}^{\top}(n)$ is presented here due to space limitation. It can be observed that, for each node, the parameter vector converged properly in accordance to the original cluster structures represented in Fig. 8. MSD learning curves are shown in Fig. 11. Transient stages can be clearly observed on both weight behavior and MSD behavior curves. Diffusion LMS enforced the weight vectors estimated by each agent to converge to the same solution at each stage. As a consequence, the MSD learning curve shows poor performance due to large bias. Non-cooperative LMS converged without bias towards the optimum parameter vectors. The algorithm introduced by [42] showed some ability to conduct clustering but did not provide satisfactory results during transient episodes. During stages 1 and 4, it worked as well as diffusion LMS. However, during stages 2 and 3, it only performed slightly better than diffusion LMS. The proposed algorithm was able to track the system dynamic with correct clustering and appropriate convergence in the mean-square sense.

3) Large Network and High-dimensional Regressors: For the sake of simplicity, previous experiments were conducted with relatively small networks and low-dimensional optimum parameter vectors. A network consisting of two clusters with 50

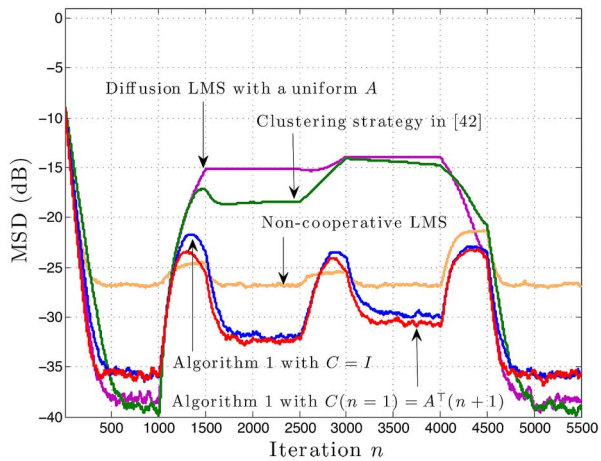


Fig. 11. Network MSD behavior comparison in the time variant multitask environment.

nodes in each cluster was randomly deployed in a given area, with physical connections defined by the connectivity matrix in Fig. 12(b). The optimum parameter vectors were set as follows: $\boldsymbol{w}^{\star}_{k} = \boldsymbol{1}_{50}$ for $k = 1, \ldots, 50$, and $\boldsymbol{w}^{\star}_{k} = -\boldsymbol{1}_{50}$ for $k = 51, \ldots, 100$. The regression inputs $\boldsymbol{x}_k(n)$ were zero-mean 50×1 random vectors governed by a Gaussian distribution with covariance matrices $\boldsymbol{R}_{x,k} = \sigma^2_{x,k}\boldsymbol{I}_L$. The background noises $z_k(n)$ were i.i.d. zero-mean Gaussian random variables, and independent of any other signal. The variances $\sigma^2_{x,k}$ and $\sigma^2_{z,k}$ were uniformly sampled in $[0.8, 1.2]$ and $[0.018, 0.022]$, respectively. For all nodes, the step-sizes were set to $\mu_k = 0.01$. The same four algorithms as before were considered. Our algorithm was used with the normalized gradient $\boldsymbol{q}_k(n)/(\|\boldsymbol{q}_k(n)\| + \xi)$ and $\xi = 0.01$. MSD learning curves are shown in Fig. 12(a), and the connectivity matrix determined by our algorithm is represented in Fig. 12(c). It can be observed that the performance of our algorithm is better than that of other methods.

C. Collaborative Target Tracking Over Sensor Networks

Consider now a target tracking problem to illustrate our adaptive clustering strategy with diffusion LMS. We focused on a scenario involving four targets, numbered from $i = 1$ to 4, moving according to the state-transition equation

$$\boldsymbol{x}_i(n+1) = \boldsymbol{T}^{\star}_i \boldsymbol{x}_i(n) + \boldsymbol{z}_i(n) \quad \text{for } i = 1, \ldots, 4, \quad n = 0, \ldots, 100 \tag{110}$$

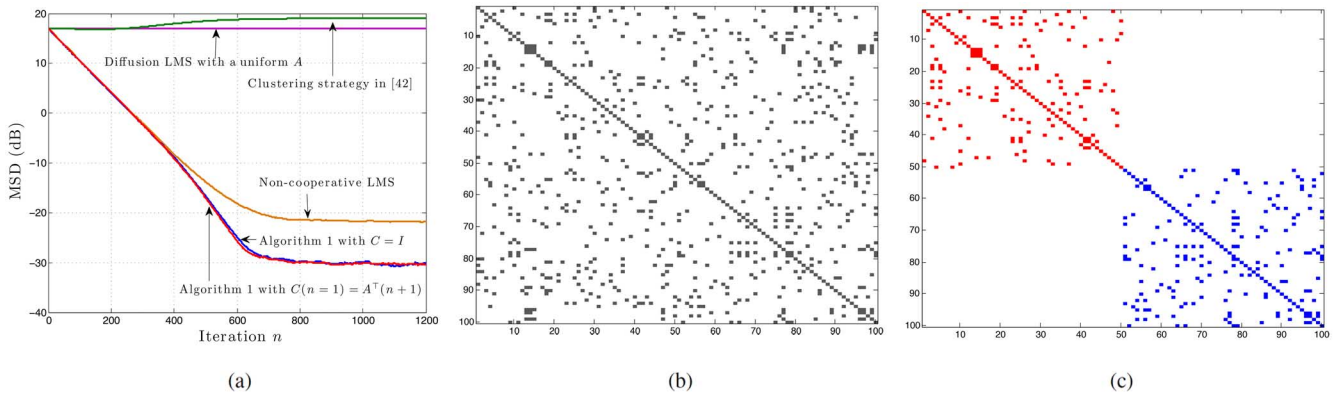


Fig. 12. Simulation results on a large network ($N = 100$) with high-dimensional regressors ($L = 50$). (a) Comparison of MSD learning curves. (b) Initial connectivity matrix, where gray elements represent physical connections. (c) Connectivity matrix resulting from our clustering strategy. (a) MSD learning curves. (b) Network physical connection matrix. (c) Connection selected by the algorithm.

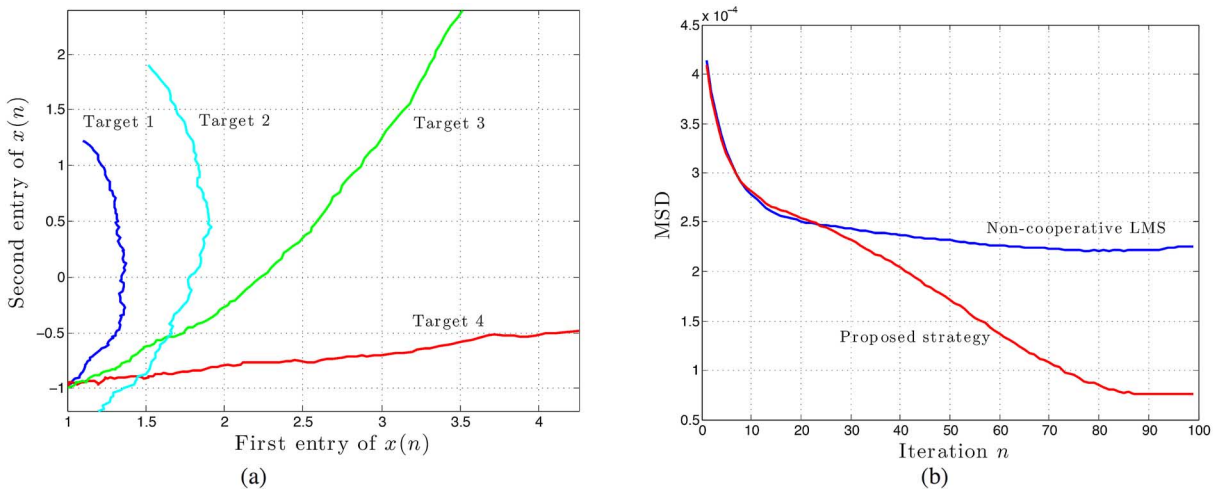


Fig. 13. (a) Trajectories of four targets with initial coordinates $\boldsymbol{x}_1(0) = \boldsymbol{x}_2(0) = \boldsymbol{x}_3(0) = [1, -1]^\top$, $\boldsymbol{x}_4(0) = [1.2, -1.2]^\top$, from $n = 0$ to 100. (b) Comparison of MSD learning curves for the estimate $\boldsymbol{T}_k$. (a) Target trajectories. (b) MSD learning curves.

where $\boldsymbol{x}_i(n)$ is the 2-dimensional coordinates for target i at instant n. Matrices $\boldsymbol{T}_i$ are 2×2 state-transition matrices that were set to

$$\boldsymbol{T}_1^\star = \begin{pmatrix} 1.0019 & -0.0129 \\ 0.0187 & 1.0034 \end{pmatrix}, \boldsymbol{T}_2^\star = \begin{pmatrix} 1.0149 & -0.0014 \\ 0.0033 & 1.0034 \end{pmatrix},$$
$$\boldsymbol{T}_3^\star = \begin{pmatrix} 1.0128 & -0.0041 \\ 0.0156 & 1.0086 \end{pmatrix}, \boldsymbol{T}_4^\star = \boldsymbol{T}_1^\star \tag{111}$$

and $\boldsymbol{z}_i(n)$ is the modeling error with i.i.d. zero-mean Gaussian distribution with covariance matrix $\sigma_z^2 \boldsymbol{I}$. The standard deviation was set to $\sigma_z = 0.01$. The initial coordinates for the four targets were

$$\boldsymbol{x}_1(0) = \boldsymbol{x}_2(0) = \boldsymbol{x}_3(0) = [1 \ \ -1]^\top, \quad \boldsymbol{x}_4(0) = [1.2 \ \ -1.2]^\top. \tag{112}$$

Fig. 13(a) shows the trajectories of the four targets from instant $n = 0$ to 100. A network with $N = 100$ nodes was randomly deployed in a given area, with physical connections defined by the connectivity matrix in Fig. 14(a).

We supposed that each node was able to track only one target during the experiment, with noisy observations $\tilde{\boldsymbol{x}}_k(n)$:

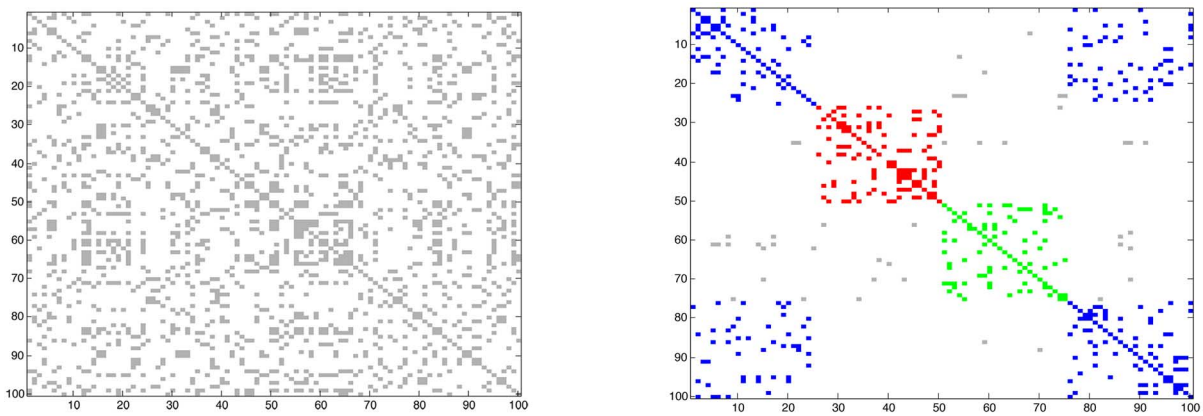
$$\tilde{\boldsymbol{x}}_k(n) = \boldsymbol{x}_k(n) + \boldsymbol{u}_k(n) \qquad \text{for } k = 1, \ldots, N \tag{113}$$

with $\boldsymbol{u}_k(n)$ an i.i.d. zero-mean Gaussian observation noise with covariance matrix $\sigma_u^2 \boldsymbol{I}$ and standard deviation $\sigma_u = 0.01$. For ease of presentation, we assumed that nodes 1–25 tracked target 1, nodes 26–50 tracked target 2, nodes 51–75 tracked target 3, and nodes 76–100 tracked target 4.

Considering $\tilde{\boldsymbol{x}}(n)$ as input data and $\tilde{\boldsymbol{x}}(n+1)$ as the desired output data for the learning algorithm, each node was aimed to track a target or, equivalently, to estimate its transition matrix given input-output noisy data. Without cooperation, this task can be performed by each node k by minimizing the following cost function with respect to matrix $\boldsymbol{T}_k$:

$$J_k(\boldsymbol{T}_k) = \mathbb{E}\|\tilde{\boldsymbol{x}}_k(n+1) - \boldsymbol{T}_k\, \tilde{\boldsymbol{x}}_k(n)\|^2 \qquad \text{for } k = 1, \ldots, N. \tag{114}$$

Collaboration among nodes may be beneficial as several nodes are conducting the same task, including nodes that track the same target and nodes that track distinct targets with the same state-transition matrix. Clearly, diffusion LMS with a uniform combination matrix is not suitable within this context since neighboring nodes may not have the same task to conduct. This problem requires adaptive clustering to automatically aggregate nodes that perform a similar task.



(a) (b)

Fig. 14. (a) Initial connectivity matrix, where gray elements represent physical connections. (b) Connectivity matrix resulting from our clustering strategy. Blue elements correspond to the connections used to estimate the transition matrices $\boldsymbol{T}_1^\star = \boldsymbol{T}_4^\star$. Red and green elements correspond to the connections used to estimate the transition matrices $\boldsymbol{T}_2^\star$ and $\boldsymbol{T}_3^\star$, respectively. Gray elements can be considered as false connections because they involve nodes that do not estimate the same transition matrix. (a) Network physical connection matrix. (b) Connections selected collaboratively by the algorithm.

Algorithm 1 was run with $\boldsymbol{C} = \boldsymbol{I}_N$ and was initialized with $\boldsymbol{T}_k(0) = \boldsymbol{I}_2$. The step-size μ was set equal to $\mu = 0.05$. Fig. 13(b) shows the MSD learning curves of transition matrices estimated by non-cooperative LMS, and diffusion LMS with adaptive clustering strategy. The performance gain can be clearly seen from these figures. Fig. 14(b) shows the connectivity matrix determined by the clustering strategy at iteration $n = 100$. Gray elements in this figure represent the combination weights $a_{\ell k}$ that are larger than 0.05. It can be seen that connections are distributed in 4 blocks on the diagonal, each one corresponding to a target, and 2 other blocks (upper-right and lower-left ones) where nodes track two distinct targets with the same state-transition matrix.

VI. Conclusion and Perspectives

Many practical problems of interest happen to be multitask-oriented in the sense that there are multiple optimum parameter vectors to be inferred simultaneously. In this paper, we studied the performance of the single-task diffusion LMS algorithm when it is run in a multitask environment. Accurate mean weight behavior model and mean square deviation model were derived. Next, we proposed an unsupervised clustering strategy that allows each node to select the neighboring nodes with which it can collaborate to address a given task. Simulations were presented to demonstrate the efficiency of the proposed clustering strategy.

References

[1] J. Chen and C. Richard, "Performance analysis of diffusion LMS in multitask networks," in *Proc. IEEE Int. Workshop Comput. Adv. Multi-Sensor Adapt. Process. (CAMSAP)*, Saint Martin, France, Dec. 2013, pp. 137–140.

[2] A. H. Sayed, "Adaptive networks," *Proc. IEEE*, vol. 102, no. 4, pp. 460–497, Apr. 2014.

[3] J. Tsitsiklis and M. Athans, "Convergence and asymptotic agreement in distributed decision problems," *IEEE Trans. Autom. Control*, vol. 29, no. 1, pp. 42–50, Jan. 1984.

[4] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Syst. Control lett.*, vol. 53, no. 9, pp. 65–78, Sep. 2004.

[5] P. Braca, S. Marano, and V. Matta, "Enforcing consensus while monitoring the environment in wireless sensor networks," *IEEE Trans. Signal Process.*, vol. 56, no. 7, pp. 3375–3380, Jul. 2008.

[6] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Trans. Autom. Control*, vol. 54, no. 1, pp. 48–61, Jan. 2009.

[7] S. Kar and J. M. F. Moura, "Distributed consensus algorithms in sensor networks: link failures and channel noise," *IEEE Trans. Signal Process.*, vol. 57, no. 1, pp. 355–369, Jan. 2009.

[8] P. Braca, S. Marano, V. Matta, and P. Willett, "Asymptotic optimality of running consensus in testing binary hypotheses," *IEEE Trans. Signal Process.*, vol. 58, no. 2, pp. 814–825, Feb. 2010.

[9] A. G. Dimakis, S. Kar, J. M. F. Moura, M. G. Rabbat, and A. Scaglione, "Gossip algorithms for distributed signal processing," *Proc. IEEE*, vol. 98, no. 11, pp. 1847–1864, Nov. 2010.

[10] K. Srivastava and A. Nedic, "Distributed asynchronous constrained stochastic optimization," *IEEE J. Sel. Topics Signal Process.*, vol. 5, no. 4, pp. 772–790, Aug. 2011.

[11] D. P. Bertsekas, "A new class of incremental gradient methods for least squares problems," *SIAM J. Optim.*, vol. 7, no. 4, pp. 913–926, Nov. 1997.

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

[13] M. G. Rabbat and R. D. Nowak, "Quantized incremental algorithms for distributed optimization," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 4, pp. 798–808, Apr. 2005.

[14] D. Blatt, A. O. Hero, and H. Gauchman, "A convergent incremental gradient method with constant step size," *SIAM J. Optim.*, vol. 18, no. 1, pp. 29–51, Feb. 2007.

[15] 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.

[16] 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.

[17] F. S. Cattivelli and A. H. Sayed, "Diffusion LMS strategies for distributed estimation," *IEEE Trans. Signal Process.*, vol. 58, no. 3, pp. 1035–1048, Mar. 2010.

[18] S.-Y. Tu and A. H. Sayed, "Diffusion strategies outperform consensus strategies for distributed estimation over adaptive netowrks," *IEEE Trans. Signal Process.*, vol. 60, no. 12, pp. 6217–6234, Dec. 2012.

[19] J. Chen and A. H. Sayed, "Diffusion adaptation strategies for distributed optimization and learning over networks," *IEEE Trans. Signal Process.*, vol. 60, no. 8, pp. 4289–4305, Aug. 2012.

[20] A. Khalili, M. A. Tinati, A. Rastegarnia, and J. A. Chambers, "Steady-state analysis of diffusion LMS adaptive networks with noisy links," *IEEE Trans. Signal Process.*, vol. 60, no. 2, pp. 974–979, Feb. 2012.

[21] X. Zhao, S.-Y. Tu, and A. H. Sayed, "Diffusion adaptation over networks under imperfect information exchange and non-stationary data," *IEEE Trans. Signal Process.*, vol. 60, no. 7, pp. 3460–3475, Jul. 2012.

[22] J. Chen and A. H. Sayed, "Distributed Pareto optimization via diffusion strategies," *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 2, pp. 205–220, Apr. 2013.

[23] O. N. Gharehshiran, V. Krishnamurthy, and G. Yin, "Distributed energy-aware diffusion least mean squares: game-theoretic learning," *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 5, pp. 1–16, Oct. 2013.

[24] S. Chouvardas, K. Slavakis, and S. Theodoridis, "Adaptive robust distributed learning in diffusion sensor networks," *IEEE Trans. Signal Process.*, vol. 59, no. 10, pp. 4692–4707, Oct. 2011.
[25] Y. Liu, C. Li, and Z. Zhang, "Diffusion sparse least-mean squares over networks," *IEEE Trans. Signal Process.*, vol. 60, no. 8, pp. 4480–4485, Aug. 2012.
[26] S. Chouvardas, K. Slavakis, Y. Kopsinis, and S. Theodoridis, "A sparsity-promoting adaptive algorithm for distributed learning," *IEEE Trans. Signal Process.*, vol. 60, no. 10, pp. 5412–5425, Oct. 2012.
[27] P. Di Lorenzo and A. H. Sayed, "Sparse distributed learning based on diffusion adaptation," *IEEE Trans. Signal Process.*, vol. 61, no. 6, pp. 1419–1433, Mar. 2013.
[28] P. Chainais and C. Richard, "Distributed dictionary learning over a sensor network," in *Proc. Conf. sur l'Apprentissage Automatique (CAP)*, Lille, France, Jul. 2013, pp. 1–6.
[29] P. Chainais and C. Richard, "Learning a common dictionary over a sensor network," in *Proc. IEEE Int. Workshop Comput. Adv. Multi-Sensor Adapt. Process. (CAMSAP)*, Saint Martin, France, Dec. 2013, pp. 1–4.
[30] J. Predd, S. Kulkarni, and H. Vincent Poor, "Distributed learning in wireless sensor networks," *IEEE Signal Process. Mag.*, vol. 23, no. 4, pp. 59–69, Jul. 2006.
[31] P. Honeine, M. Essoloh, C. Richard, and H. Snoussi, "Distributed regression in sensor networks with a reduced-order kernel model," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, New Orleans, LA, USA, 2008, pp. 1–5.
[32] J. Chen, L. Tang, J. Liu, and J. Ye, "A convex formulation for leaning shared structures from muliple tasks," in *Proc. Annu. Int. Conf. Mach. Learning (ICML)*, Montreal, Canada, Jun. 2009, pp. 137–144.
[33] O. Chapelle, P. Shivaswmy, K. Q. Vadrevu, S. Weinberger, Y. Zhang, and B. Tseng, "Multi-task learning for boosting with application to web search ranking," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, Washington, DC, USA, Jul. 2010, pp. 1189–1198.
[34] J. Zhou, L. Yuan, J. Liu, and J. Ye, "A multi-task learning formulation for predicting disease progression," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, San Diego, CA, USA, Aug. 2011, pp. 814–822.
[35] J. Chen, C. Richard, and A. H. Sayed, "Multitask diffusion LMS over networks," *IEEE Trans. Signal Process.*, vol. 62, no. 16, pp. 4129–4144, Aug. 2014.
[36] R. Abdolee, B. Champagne, and A. H. Sayed, "Estimation of space-time varying parameters using a diffusion LMS algorithm," *IEEE Trans. Signal Process.*, vol. 62, no. 2, pp. 403–418, Jan. 2014.
[37] A. Bertrand and M. Moonen, "Distributed adaptive node-specific signal estimation in fully connected sensor networks—Part I: sequential node updating," *IEEE Trans. Signal Process.*, vol. 58, no. 10, pp. 5277–5291, Oct. 2010.
[38] A. Bertrand and M. Moonen, "Distributed adaptive estimation of node-specific signals in wireless sensor networks with a tree topology," *IEEE Trans. Signal Process.*, vol. 59, no. 5, pp. 2196–2210, May 2011.
[39] N. Bogdanovic, J. Plata-Chaves, and K. Berberidis, "Distributed incremental-based LMS for node-specific parameter estimation over adaptive networks," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Vancouver, Canada, May 2013, pp. 5425–5429.
[40] N. Bogdanovic, J. Plata-Chaves, and K. Berberidis, "Distributed diffusion-based LMS for node-specific parameter estimation over adaptive networks," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Florence, Italy, May 2014, pp. 7223–7227.
[41] J. Chen, C. Richard, A. O. Hero, and A. H. Sayed, "Diffusion LMS for multitask problems with overlapping hypothesis subspaces," in *Proc. IEEE Int. Workshop Mach. Learn. Signal Process.*, Reims, France, Sep. 2014, pp. 1–6.
[42] X. Zhao and A. H. Sayed, "Clustering via diffusion adaptation over networks," in *Proc. Int. Workshop Cognit. Inf. Process. (CIP)*, Parador de Baiona, Spain, May 2012, pp. 1–6.
[43] A. H. Sayed, "Diffusion adaptation over networks," in *Academic Press Libraray in Signal Processing*, R. Chellapa and S. Theodoridis, Eds. New York, NY, USA: Elsevier, 2014, pp. 322–454.
[44] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.
[45] A. H. Sayed, S.-Y. Tu, J. Chen, X. Zhao, and Z. Towfic, "Diffusion strategies for adaptation and learning over networks," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 155–171, May 2013.
[46] A. H. Sayed, *Adaptive Filters*. NJ, USA: Wiley, 2008.
[47] A. H. Sayed, "Adaptation, learning, and optimization over networks," in *Foundations and Trends in Mach. Learning*. Boston, MA, USA: NOW Publishers, 2014, vol. 7, pp. 311–801.
[48] K. M. Abadir and J. R. Magnus, *Matrix algebra*. Cambridge, U.K.: Cambridge Univer. Press, 2005.
[49] D. S. Bernstein, *Matrix Mathematics: Theory, Facts, and Formulas*. Princeton, NJ, USA: Princeton Univ. Press, 2009.

Jie Chen (S'12–M'14) was born in Xi'an, China, in 1984. He received the B.S. degree in information and telecommunication engineering in 2006 from the Xi'an Jiaotong University, Xi'an, and the Dipl.-Ing. and the M.S. degrees in information and telecommunication engineering in 2009 from the University of Technology of Troyes (UTT), Troyes, France, and from the Xi'an Jiaotong University, respectively. In 2013, he received the Ph.D. degree in systems optimization and security from the UTT.

From April 2013 to March 2014, he was a Postdoctoral Research Fellow at the Cote d'Azur Observatory, University of Nice Sophia Antipolis, Nice, France. Since April 2014, he has been a Postdoctoral Research Fellow with the Department of Electrical Engineering and Computer Science at the University of Michigan, Ann Arbor, USA.

His current research interests include adaptive signal processing, supervised and unsupervised learning, distributed optimization, hyperspectral image analysis, and bio-signal processing.

Cédric Richard (S'98–M'01–SM'07) received the Dipl.-Ing. and the M.S. degrees in 1994, and the Ph.D. degree in 1998, from Compiègne University of Technology, France, all in electrical and computer engineering. From 1999 to 2003, he was an Associate Professor at Troyes University of Technology, France, and a Full Professor from 2003 to 2009. Since 2009, he is a Full Professor at the University of Nice Sophia Antipolis, France. He is a junior member of the Institut Universitaire de France since 2010.

His current research interests include statistical signal processing and machine learning. Cédric Richard is the author of over 230 papers. He was the General Co-Chair of the IEEE SSP Workshop that was held in Nice, France, in 2011. He is the Technical Co-Chair of EUSIPCO 2015 to be held in Nice, France, and of the IEEE CAMSAP Workshop 2015 to be held in Cancun, Mexico. He serves as a Senior Area Editor of the IEEE Transactions on Signal Processing and as an Associate Editor of the IEEE Transactions on Signal and Information Processing over Networks since 2015. He is also an Associate Editor of Signal Processing Elsevier since 2009. Cédric Richard is member of the Machine Learning for Signal Processing (MLSP TC) Technical Committee, and served as member of the Signal Processing Theory and Methods (SPTM TC) Technical Committee in 2009–2014.

Ali H. Sayed (S'90–M'92–SM'99–F'01) is professor and former chairman of electrical engineering at the University of California, Los Angeles, where he directs the UCLA Adaptive Systems Laboratory. An author of over 450 scholarly publications and six books, his research involves several areas including adaptation and learning, statistical signal processing, distributed processing, network science, and biologically-inspired designs. His work has been recognized with several awards including the 2014 Athanasios Papoulis Award from the European Association for Signal Processing, the 2013 Meritorious Service Award and the 2012 Technical Achievement Award from the IEEE Signal Processing Society, the 2005 Terman Award from the American Society for Engineering Education, the 2003 Kuwait Prize, and the 1996 IEEE Donald G. Fink Prize. He served as Distinguished Lecturer for the IEEE Signal Processing Society in 2005 and as Editor-in Chief of the IEEE Transactions on Signal Processing (2003–2005). His articles received several Best Paper Awards from the IEEE Signal Processing Society in 2002, 2005, 2012, and 2014. He is a Fellow of the American Association for the Advancement of Science (AAAS). He is recognized as a Highly Cited Researcher by Thomson Reuters.