Diffusion LMS for Multitask Problems With Local Linear Equality Constraints

Roula Nassif, Cédric Richard, *Senior Member, IEEE*, André Ferrari, *Member, IEEE*, and Ali H. Sayed, *Fellow, IEEE*

***Abstract*—We consider distributed multitask learning problems over a network of agents where each agent is interested in estimating its own parameter vector, also called task, and where the tasks at neighboring agents are related according to a set of linear equality constraints. Each agent possesses its own convex cost function of its parameter vector and a set of linear equality constraints involving its own parameter vector and the parameter vectors of its neighboring agents. We propose an adaptive stochastic algorithm based on the projection gradient method and diffusion strategies in order to allow the network to optimize the individual costs subject to all constraints. Although the derivation is carried out for linear equality constraints, the technique can be applied to other forms of convex constraints. We conduct a detailed mean-square-error analysis of the proposed algorithm and derive closed-form expressions to predict its learning behavior. We provide simulations to illustrate the theoretical findings. Finally, the algorithm is employed for solving two problems in a distributed manner: A minimum-cost flow problem over a network and a space–time varying field reconstruction problem.**

***Index Terms*—Multitask networks, distributed estimation, local linear equality constraints, projection approach, diffusion adaptation.**

I. Introduction

SINGLE-TASK distributed optimization over networks allows to minimize the aggregate sum of convex cost functions, each available at an agent, subject to convex constraints that are also distributed across the agents. Each learner seeks to estimate the minimizer through local computations and communications among neighboring agents without the need to know any of the constraints or costs besides their own. Several useful strategies have been proposed to solve constrained and unconstrained versions of this problem in a fully decentralized manner [1]–[13]. Diffusion strategies [3], [8]–[12] are attractive since they are scalable, robust, and enable continuous learning and adaptation in response to drifts in the location of the minimizer due to changes in the costs or in the constraints.

Multitask distributed learning over networks is particularly well-suited for applications where several parameter vectors need to be estimated simultaneously from successive noisy measurements using in-network processing [8], [14]–[33]. The network is decomposed into clusters of agents and each cluster estimates its own parameter vector [22]. Distributed strategies for solving multitask problems have been addressed in two main ways. In a first scenario, agents do not know the cluster they belong to and no prior information on possible relationships between tasks is assumed. In this case, all agents cooperate with each other as dictated by the network topology. It is shown in [8] that, in this case, the diffusion iterates will converge to a Pareto optimal solution corresponding to a multi-objective optimization problem. To avoid cooperation with neighbors seeking different objectives, automatic clustering techniques based on diffusion strategies have also been proposed [27]–[29]. In a second scenario, it is assumed that agents know which cluster they belong to. Multitask diffusion strategies are then derived by exploiting prior information about relationships among the tasks. For example, one way to model and exploit relationships among tasks is to formulate convex optimization problems with appropriate co-regularizers between the agents [16], [22]–[25]. While [16] deals with deterministic optimization problems, [22]–[25] are concerned with adaptive estimation problems. In [15], distributed algorithms are derived to estimate node-specific signals that lie in a common latent signal subspace in the presence of node-specific linear equality constraints. Several useful works consider stochastic [17]–[20] and deterministic [21] multitask estimation problems with overlapping parameter vectors. They assume that each agent is interested in estimating its own parameter vector, and that the local parameter vectors at neighboring agents have some entries that are equal. Unsupervised strategies are also considered in [30], [31] to address multitask overlapping problems. In [26], a diffusion algorithm is derived to solve multitask estimation problem where the parameter space is decomposed into two orthogonal subspaces, with one of the subspaces being common to all agents.

In some applications, it happens that the optimum parameter vectors to be estimated at neighboring agents are related according to a set of constraints. This observation motivates us to consider in this work multitask estimation problems subject to

Manuscript received October 12, 2016; revised March 9, 2017, May 22, 2017, and June 2, 2017; accepted June 5, 2017. Date of publication June 29, 2017; date of current version July 24, 2017. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Wee Peng Tay. The work of C. Richard and A. Ferrari was supported by in part by ANR and in part by DGA under Grant ANR-13-ASTR-0030 (ODISSEE Project). The work of A. H. Sayed was supported in part by NSF under Grants CIF-1524250 and ECCS-1407712. *(Corresponding author: Cédric Richard.)*

R. Nassif, C. Richard, and A. Ferrari are with the Université de Nice Sophia-Antipolis, Nice 06000, France (e-mail: roula.nassif@oca.eu; cedric.richard@unice.fr; andre.ferrari@unice.fr).

A. H. Sayed is with the Department of Electrical Engineering, University of California, Los Angeles, CA 90095 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.2017.2721930

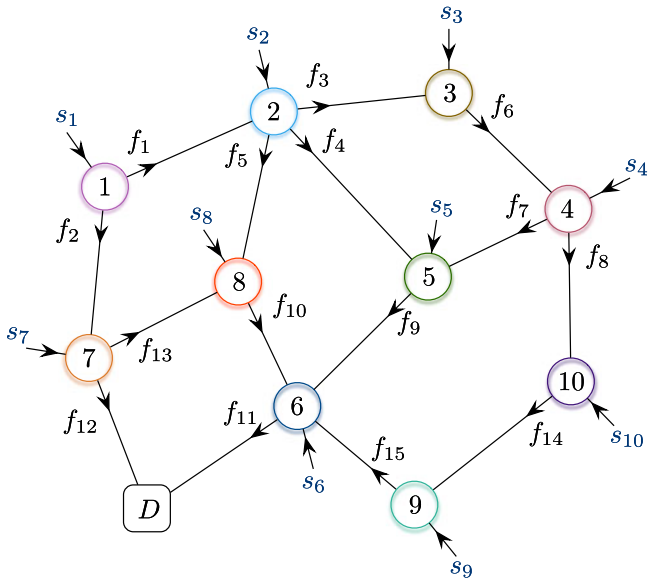


Fig. 1. Flow network topology with 10 nodes, 1 destination sink D, and 15 communication links.

linear equality constraints of the form:

$$\underset{\boldsymbol{w}_1,\ldots,\boldsymbol{w}_N}{\text{minimize}} \; J^{\text{glob}}(\boldsymbol{w}_1,\ldots,\boldsymbol{w}_N) \triangleq \sum_{k=1}^{N} J_k(\boldsymbol{w}_k), \tag{1a}$$

$$\text{subject to} \sum_{\ell\in\mathcal{I}_p} \boldsymbol{D}_{p\ell}\,\boldsymbol{w}_\ell + \boldsymbol{b}_p = \boldsymbol{0},\; p=1,\ldots,P. \tag{1b}$$

Each agent k in the network seeks to estimate its own $M_k \times 1$ parameter vector $\boldsymbol{w}_k$, and has knowledge of its cost function $J_k(\cdot)$ and the set of linear equality constraints that agent k is involved in. Each constraint is indexed by p, and defined by the $L_p \times M_\ell$ matrices $\boldsymbol{D}_{p\ell}$, the $L_p \times 1$ vector $\boldsymbol{b}_p$, and the set $\mathcal{I}_p$ of agent indices involved in this constraint. Note that, by properly selecting the matrices $\boldsymbol{D}_{p\ell}$ and setting the vectors $\boldsymbol{b}_p$ to $\boldsymbol{0}$ in (1), the *single-task* estimation problem [2]–[4] and the *multi-task overlapping* estimation problem [17]–[21] can be recast as problem (1).

Assumption: In the current work, it is assumed that each agent k in $\mathcal{I}_p$ can collect estimates from all agents in $\mathcal{I}_p$ in order to satisfy the p-th constraint, i.e., $\mathcal{I}_p \subseteq \mathcal{N}_k$ for all $k \in \mathcal{I}_p$ where $\mathcal{N}_k$ denotes the neighborhood of agent k. This assumption is reasonable in many applications, for instance, in remote monitoring of physical phenomena involving discretization of spatial differential equations [34], and in network monitoring involving conservation laws at each junction [35].

For illustration purposes, consider a minimum-cost flow problem over the network shown in Fig. 1. This network consists of 10 nodes, 1 destination sink D, and 15 communication links. With each link j, we associate a directed arc and we let f_j denote the flow or traffic on this link, with $f_j > 0$ meaning that the flow is in the direction of the arc, and $f_j < 0$ otherwise. At each node k, an external source flow s_k enters and flows through the network to the destination sink. The flow must satisfy a conservation equation, which states that at each node k, the sum of flows entering the node, plus the external source s_k, is equal to the sum of flows leaving node k. Given the external sources s_k and the network topology, a number of studies have been devoted to finding the optimal flows $f_j^\star$ that minimize a total flow transmission cost and satisfy the conservation equations [35]–[37]. Problems of this type arise in applications such as electrical networks, telecommunication networks, pipeline networks [35]. In some of these applications, it happens that node k has only access to noisy measurements $s_k(i)$ of the external source at each time instant i. For example, in electrical networks, the agents may not be able to collect the exact values of the current sources (or the current demands). Denoting by $\boldsymbol{w}_k$ the $M_k \times 1$ vector containing the flows f_j entering and leaving node k, we are interested in distributed online learning settings where each node k seeks to estimate $\boldsymbol{w}_k$ from noisy measurements $s_k(i)$ by relying only on local computations and communications with its neighbors. This problem can be recast in the form (1a)–(1b) and addressed with the multitask strategy proposed in this paper. This example will be considered further in the numerical experiments section.

We shall propose a primal adaptive technique (based on propagating and estimating the primal variable) for solving problem (1) in a distributed manner. The technique relies on combining diffusion adaptation with a stochastic gradient projection step, and on the use of constant step-sizes to enable continuous adaptation and learning from streaming data. Since we are learning from streaming data, the dual function cannot be computed exactly and the use of primal-dual methods may result in stability problems as already shown in [12]. For this reason, we focus on primal techniques. Our current work is able to cope with the following two scenarios: 1) multitask problems with prior information on linear relationships between tasks, and 2) constrained multitask problems with distributed information access. We analyze the behavior of our algorithm in the mean and mean-square-error sense (w.r.t. the minimizers of the local costs and w.r.t. the solution of the constrained multitask problem) and we derive expressions to predict its transient and steady-state behavior. Some simulation results show that, for small constant step-sizes, the expected distance between the estimates at each agent and the optimal value can be made arbitrarily small.

Notation: Normal font letters denote scalars, boldface lowercase letters denote column vectors, and boldface uppercase letters denote matrices. We use the symbol $(\cdot)^\top$ to denote matrix transpose, the symbol $(\cdot)^{-1}$ to denote matrix inverse, the symbol $(\cdot)^\dagger$ to denote the pseudo-inverse of a full row-rank matrix, and the symbol $\text{tr}(\cdot)$ to denote the trace operator. The symbol $\text{diag}\{\cdot\}$ forms a matrix from block arguments by placing each block immediately below and to the right of its predecessor. The operator $\text{col}\{\cdot\}$ stacks the column vector entries on top of each other. The symbols $\otimes$ and $\otimes_b$ denote the Kronecker product and the block Kronecker product, respectively. The symbol $\text{vec}(\cdot)$ refers to the standard vectorization operator that stacks the columns of a matrix on top of each other and the symbol $\text{bvec}(\cdot)$ refers to the block vectorization operation that vectorizes each block of a matrix and stacks the vectors on top of each other. The identity matrix of size $N \times N$ is denoted by $\boldsymbol{I}_N$. The $N \times 1$ vector of ones is denoted by $\mathbb{1}_{N\times 1}$. For a $P \times N$ block

matrix $\boldsymbol{\mathcal{A}}$, the $1 \times N$ k-th block row is denoted by $[\boldsymbol{\mathcal{A}}]_{k,\bullet}$ and the $P \times 1$ k-th block column is denoted by $[\boldsymbol{\mathcal{A}}]_{\bullet,k}$. The notation $\mathrm{P}_{\Omega}(\boldsymbol{w})$ denotes the projection of $\boldsymbol{w}$ onto the manifold Ω.

II. Problem Formulation and Centralized Solution

A. Problem Formulation and Assumptions

Consider a network of N agents, labeled $k = 1, \ldots, N$. At each time instant i, each agent k has access to a zero-mean real-valued observation $d_k(i)$, and a zero-mean real-valued $M_k \times 1$ regression vector $\boldsymbol{x}_k(i)$, with positive covariance matrix $\boldsymbol{R}_{x,k} = \mathbb{E}\{\boldsymbol{x}_k(i)\boldsymbol{x}_k^{\top}(i)\} > 0$. We assume the data to be related via the linear data model:

$$d_k(i) = \boldsymbol{x}_k^{\top}(i)\boldsymbol{w}_k^o + z_k(i), \tag{2}$$

where $\boldsymbol{w}_k^o$ is an $M_k \times 1$ unknown parameter vector, and $z_k(i)$ is a zero-mean measurement noise of variance $\sigma_{z,k}^2$, independent of $\boldsymbol{x}_\ell(j)$ for all ℓ and j, and independent of $z_\ell(j)$ for $\ell \neq k$ or $i \neq j$. We let $\boldsymbol{r}_{dx,k} \triangleq \mathbb{E}\{d_k(i)\boldsymbol{x}_k(i)\}$ and $\sigma_{d,k}^2 \triangleq \mathbb{E}|d_k(i)|^2$.

Let $\boldsymbol{w}_k$ denote some generic $M_k \times 1$ vector that is associated with agent k. The objective of agent k is to find an estimate for $\boldsymbol{w}_k^o$, and we associate with this agent the mean-square-error criterion:

$$J_k(\boldsymbol{w}_k) = \mathbb{E}\,|d_k(i) - \boldsymbol{x}_k^{\top}(i)\boldsymbol{w}_k|^2, \tag{3}$$

which is strongly convex, second-order differentiable, and minimized at $\boldsymbol{w}_k^o$. In addition, P linear equality constraints of the form (1b) are imposed on the parameter vectors $\{\boldsymbol{w}_k\}$ at each time instant i. Let us collect the parameter vectors $\{\boldsymbol{w}_k\}$ and $\{\boldsymbol{w}_k^o\}$ from across the network into $N \times 1$ block column vectors $\boldsymbol{w}$ and $\boldsymbol{w}^o$, respectively:

$$\boldsymbol{w} \triangleq \mathrm{col}\{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_N\}, \qquad \boldsymbol{w}^o \triangleq \mathrm{col}\{\boldsymbol{w}_1^o, \ldots, \boldsymbol{w}_N^o\}, \tag{4}$$

and let us write the P linear equality constraints in (1b) more compactly as:

$$\boldsymbol{\mathcal{D}}\boldsymbol{w} + \boldsymbol{b} = \boldsymbol{0}, \tag{5}$$

where $\boldsymbol{\mathcal{D}}$ is a $P \times N$ block matrix, with each block $\boldsymbol{D}_{p\ell}$ having dimensions $L_p \times M_\ell$, and $\boldsymbol{b}$ is a $P \times 1$ block column vector where each block $\boldsymbol{b}_p$ has dimensions $L_p \times 1$. Combining (5) and (3), the network optimization problem becomes:

$$\begin{aligned} \underset{\boldsymbol{w}}{\text{minimize}} \quad & \sum_{k=1}^{N} \mathbb{E}\,|d_k(i) - \boldsymbol{x}_k^{\top}(i)\boldsymbol{w}_k|^2, \\ \text{subject to} \quad & \boldsymbol{\mathcal{D}}\boldsymbol{w} + \boldsymbol{b} = \boldsymbol{0}, \end{aligned} \tag{6}$$

where each agent k is in charge of estimating the k-th subvector $\boldsymbol{w}_k$ of $\boldsymbol{w}$. Since the mean-square-error criterion in (6) is separable, we shall assume without loss of generality that each parameter vector $\boldsymbol{w}_k$ is involved in at least one constraint so that cooperation is justified. We shall also assume that $\boldsymbol{\mathcal{D}}$ is full row-rank to ensure that equation $\boldsymbol{\mathcal{D}}\boldsymbol{w} + \boldsymbol{b} = \boldsymbol{0}$ has at least one solution. We also introduce an assumption on the availability of the constraints. Let $\mathcal{I}_p$ be the set of agent indices involved in the p-th constraint. We shall assume that every agent k in $\mathcal{I}_p$ is aware of the p-th constraint, and that the network topology permits this agent to collect estimates from all agents in $\mathcal{I}_p$, that is, $\mathcal{I}_p \subseteq \mathcal{N}_k$, so it can apply this constraint to its own estimate. This assumption is reasonable in many applications, for instance, in remote monitoring of physical phenomena [34], and in network distribution system monitoring (as described in the introduction) [35]. These examples will be considered in numerical experiments section.

Before proceeding, note that problem (6) can be recast as a quadratic program (QP) [36], and any algorithm that solves QPs can solve it. We are interested instead in distributed adaptive solutions that can operate in real-time on streaming data. As we will see later, the traditional constrained LMS algorithm [38] can solve (6) in a centralized manner. In this centralized solution, each agent at each iteration sends its data to a fusion center, which in turn processes the data and sends the estimates back to the agents. The entire matrix $\boldsymbol{\mathcal{D}}$ and the entire vector $\boldsymbol{b}$ then need to be available at the fusion center. While centralized solutions can be powerful, decentralized solutions are more attractive since they are more robust and respect the privacy policy of each agent [9], [39], [40].

B. Centralized Solution

Let us first describe the centralized solution. We assume that the agents transmit the collected data $\{d_k(i), \boldsymbol{x}_k(i)\}$ to a fusion center for processing. Problem (6) can be written equivalently as:

$$\begin{aligned} \underset{\boldsymbol{w}}{\text{minimize}} \quad & \boldsymbol{w}^{\top}\boldsymbol{\mathcal{R}}_x\boldsymbol{w} - 2\boldsymbol{r}_{dx}^{\top}\boldsymbol{w} + \boldsymbol{r}_d^{\top}\mathbb{1}_{N\times 1}, \\ \text{subject to} \quad & \boldsymbol{\mathcal{D}}\boldsymbol{w} + \boldsymbol{b} = \boldsymbol{0}, \end{aligned} \tag{7}$$

where the $N \times N$ block diagonal matrix $\boldsymbol{\mathcal{R}}_x$, the $N \times 1$ block column vector $\boldsymbol{r}_{dx}$, and the $N \times 1$ column vector $\boldsymbol{r}_d$ are given by:

$$\boldsymbol{\mathcal{R}}_x \triangleq \mathrm{diag}\{\boldsymbol{R}_{x,1}, \ldots, \boldsymbol{R}_{x,N}\}, \tag{8}$$

$$\boldsymbol{r}_{dx} \triangleq \mathrm{col}\{\boldsymbol{r}_{dx,1}, \ldots, \boldsymbol{r}_{dx,N}\}, \tag{9}$$

$$\boldsymbol{r}_d \triangleq \mathrm{col}\{\sigma_{d,1}^2, \ldots, \sigma_{d,N}^2\}. \tag{10}$$

Since $\boldsymbol{\mathcal{R}}_x$ is positive definite, problem (7) is a positive definite quadratic program with equality constraints. It has a unique global minimum given by:

$$\boldsymbol{w}^{\star} = \boldsymbol{w}^o - \boldsymbol{\mathcal{R}}_x^{-1}\boldsymbol{\mathcal{D}}^{\top}(\boldsymbol{\mathcal{D}}\boldsymbol{\mathcal{R}}_x^{-1}\boldsymbol{\mathcal{D}}^{\top})^{-1}(\boldsymbol{\mathcal{D}}\boldsymbol{w}^o + \boldsymbol{b}). \tag{11}$$

Let Ω denote the linear manifold:

$$\Omega \triangleq \{\boldsymbol{w}\colon\ \boldsymbol{\mathcal{D}}\boldsymbol{w} + \boldsymbol{b} = \boldsymbol{0}\}. \tag{12}$$

If $\boldsymbol{w}^o \in \Omega$, the optimum $\boldsymbol{w}^{\star}$ coincides with $\boldsymbol{w}^o$. In this case, the constrained optimization problem (6) can be thought as estimating the unknown parameter vectors $\boldsymbol{w}_k^o$ given prior information about relationships between tasks of the form (1b). Exploiting such prior information may improve the estimation as we will see in the experiments. Let M denote the dimension of the network parameter vector $\boldsymbol{w}$, i.e., $M = \sum_{k=1}^{N} M_k$. The projection of any vector $\boldsymbol{y} \in \mathbb{R}^M$ onto Ω is given by:

$$\mathrm{P}_{\Omega}(\boldsymbol{y}) = \boldsymbol{\mathcal{P}}\boldsymbol{y} - \boldsymbol{f}, \tag{13}$$

where

$$\boldsymbol{\mathcal{P}} \triangleq \boldsymbol{I}_M - \boldsymbol{\mathcal{D}}^{\dagger}\boldsymbol{\mathcal{D}}, \qquad \boldsymbol{f} \triangleq \boldsymbol{\mathcal{D}}^{\dagger}\boldsymbol{b}. \tag{14}$$

Let $\boldsymbol{w}(i)$ denotes the estimate of $\boldsymbol{w}^{\star}$ at iteration i. In order to solve (7) iteratively, the gradient projection method [41] can be applied on top of a gradient-descent iteration:

$$\boldsymbol{w}(i+1) = \mathrm{P}_{\Omega}\big(\boldsymbol{w}(i) + \mu[\boldsymbol{r}_{dx} - \boldsymbol{\mathcal{R}}_x \boldsymbol{w}(i)]\big), \qquad i \geq 0. \tag{15}$$

In order to run recursion (15), we need to have access to the second-order moments $\{\boldsymbol{R}_{x,k}, \boldsymbol{r}_{dx,k}\}$. Since these moments are rarely available beforehand, the agents use their instantaneous data $\{d_k(i), \boldsymbol{x}_k(i)\}$ to approximate the second-order moments, namely, $\boldsymbol{R}_{x,k} \approx \boldsymbol{x}_k(i)\boldsymbol{x}_k^{\top}(i)$ and $\boldsymbol{r}_{dx,k} \approx d_k(i)\boldsymbol{x}_k(i)$. Doing so and replacing $\mathrm{P}_{\Omega}(\cdot)$ by (13), we obtain the following stochastic-gradient algorithm in lieu of (15):

$$\boldsymbol{w}(i+1) = \boldsymbol{\mathcal{P}} \cdot \mathrm{col}\big\{\boldsymbol{w}_k(i) + \mu \boldsymbol{x}_k(i)[d_k(i) - \boldsymbol{x}_k^{\top}(i)\boldsymbol{w}_k(i)]\big\}_{k=1}^{N} - \boldsymbol{f}. \tag{16}$$

Collecting the regression vectors into the $M \times N$ matrix $\boldsymbol{X}(i) \triangleq \mathrm{diag}\{\boldsymbol{x}_k(i)\}_{k=1}^{N}$ and the observations into the $N \times 1$ vector $\boldsymbol{d}(i) \triangleq \mathrm{col}\{d_k(i)\}_{k=1}^{N}$, algorithm (16) becomes the Constrained Least-Mean-Squares (CLMS) algorithm:

$$\boldsymbol{w}(i+1) = \boldsymbol{\mathcal{P}}\big(\boldsymbol{w}(i) + \mu \boldsymbol{X}(i)[\boldsymbol{d}(i) - \boldsymbol{X}^{\top}(i)\boldsymbol{w}(i)]\big) - \boldsymbol{f}. \tag{17}$$

This procedure was originally proposed in [38] as an online linearly constrained minimum variance (LCMV) filter for solving mean-square-error estimation problems subject to linear constraints; the motivation there was not concerned with multi-task problems. In this section, we showed that the centralized multitask constrained problem reduces to a similar problem, for which algorithm (17) can be applied. The performance of such stand-alone centralized solutions was studied in [38], [42], [43].

III. Problem Reformulation and Distributed Solution

A. Problem Reformulation

We move on to develop a distributed solution with a continuous adaptation mechanism. First, note that several works for solving problems of the form (6) with possible distributed information access already exist in the literature [4], [6], [7], [10], [13], [21], [33], [44], [45]. However, except for [21], [33], these other works solve single-task estimation problems where the entire network is employed to estimate the minimizer of (6). Furthermore, compared to [6], [21], [33], [44], [45], we shall assume stochastic errors in the evaluation of the gradients of local cost functions.

To proceed with the analysis, one of the challenges we now face is that any given agent k may be involved in several constraints. Our strategy is to transform (6) into an equivalent optimization problem exhibiting structure amenable to distributed optimization with separable constraints. Let j_k denote the number of constraints that agent k is involved in. We expand each node k into a cluster $\mathcal{C}_k$ of j_k *virtual* sub-nodes, namely, $\mathcal{C}_k \triangleq \{k_m\}_{m=1}^{j_k}$. Each one of these sub-nodes is involved in a single constraint. Let $\boldsymbol{w}_{k_m}$ denote the $M_k \times 1$ auxiliary vector associated with sub-node k_m. In order to ensure that agent k satisfies simultaneously all the constraints at convergence, we will allow all sub-nodes at agent k to run diffusion learning to reach agreement on their estimates $\{\boldsymbol{w}_{k_m}\}$ asymptotically. We denote by $\mathcal{I}_{e,p}$ the set of sub-nodes which are involved in the p-th constraint.

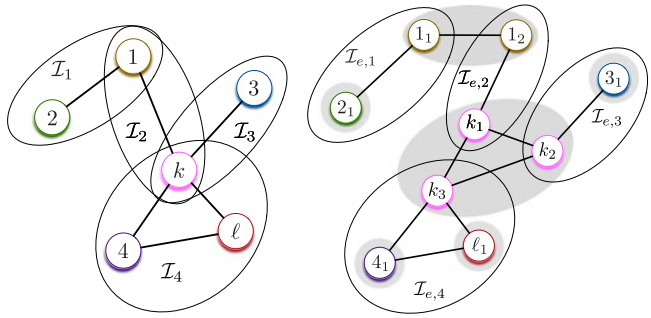


Fig. 2. (*Left*) Network topology with constraints identified by the subsets of nodes $\mathcal{I}_1, \mathcal{I}_2, \mathcal{I}_3$, and $\mathcal{I}_4$. (*Right*) Network topology model with fully-connected clusters shown in grey color and constraints now identified by the subsets of sub-nodes $\mathcal{I}_{e,1}, \mathcal{I}_{e,2}, \mathcal{I}_{e,3}$, and $\mathcal{I}_{e,4}$. All sub-nodes in this model are involved in a single constraint. Diffusion learning is run in clusters with more than one sub-node to reach agreement on local estimates while satisfying their respective constraints.

In order to clarify the presentation, an illustrative example is provided in Fig. 2. On the left of this panel is the original network topology with $N = 6$ agents and $P = 3$ constraints. On the right is the network topology model with clusters of sub-nodes shown in grey color. Observe that $\mathcal{I}_2 = \{1, k\}$, $\mathcal{I}_3 = \{3, k\}$ and $\mathcal{I}_4 = \{4, k, \ell\}$, which means that agent k is involved in constraints 2, 3, and 4. Agent k is thus expanded into a cluster $\mathcal{C}_k = \{k_1, k_2, k_3\}$ of 3 sub-nodes. Sub-nodes k_1, k_2, and k_3 are assigned to constraints 2, 3, and 4, respectively. Each other agent, say ℓ, involved in a single constraint is renamed ℓ_1 and assigned to a single-node cluster $\mathcal{C}_\ell = \{\ell_1\}$ for consistency of notation. This leads to the sets $\mathcal{I}_{e,2} = \{1_2, k_1\}$, $\mathcal{I}_{e,3} = \{3_1, k_2\}$ and $\mathcal{I}_{e,4} = \{4_1, \ell_1, k_3\}$ where all sub-nodes are involved in a single constraint. All sub-nodes k_m in cluster $\mathcal{C}_k$ can share data since they refer to the same agent k. In the sequel, we shall propose a general algorithm for strongly-connected clusters (see (35) below) and show how the designer can simplify the algorithm by choosing fully-connected clusters (see (40) below).

Accordingly, we can now reformulate problem (6). We start by collecting the vectors $\boldsymbol{w}_{k_m}$ into the $N_e \times 1$ network block column vector:

$$\boldsymbol{w}_e \triangleq \mathrm{col}\Big\{\mathrm{col}\big\{\boldsymbol{w}_{k_m}\big\}_{m=1}^{j_k}\Big\}_{k=1}^{N}, \tag{18}$$

where $N_e \triangleq \sum_{k=1}^{N} j_k$. Throughout this work, a subscript "$_e$" below a symbol indicates an extended version associated with sub-nodes (auxiliary variables). For example, while the symbol N represents the number of nodes, the symbol N_e represents the number of sub-nodes. Likewise, the vector $\boldsymbol{w}_e$ in (18) corresponds to the extended version of the vector $\boldsymbol{w}$ in (4). We introduce for each agent k a set of j_k coefficients $\{c_{k_m}\}$ that

satisfy two conditions:

$$c_{k_m} > 0, \text{ for } m = 1, \ldots, j_k, \quad \text{and} \quad \sum_{m=1}^{j_k} c_{k_m} = 1. \tag{19}$$

The coefficients $\{c_{k_m}\}$ are free parameters that are chosen by the user. A natural choice is $c_{k_m} = \frac{1}{j_k}$ for all m. The global cost in (1 a) can be written as:

$$J^{\text{glob}}(\boldsymbol{w}_1, \ldots, \boldsymbol{w}_N) \triangleq \sum_{k=1}^{N} J_k(\boldsymbol{w}_k) = \sum_{k=1}^{N} \sum_{m=1}^{j_k} c_{k_m} J_k(\boldsymbol{w}_k). \tag{20}$$

We reformulate problem (1) in the following equivalent form by introducing the auxiliary variables $\{\boldsymbol{w}_{k_m}\}$:

$$\underset{\boldsymbol{w}_e}{\text{minimize}} \quad \sum_{k=1}^{N} \sum_{m=1}^{j_k} c_{k_m} J_k(\boldsymbol{w}_{k_m}) \tag{21a}$$

$$\text{subject to} \quad \sum_{\ell_n \in \mathcal{I}_{e,p}} \boldsymbol{D}_{p\ell_n} \boldsymbol{w}_{\ell_n} + \boldsymbol{b}_p = \boldsymbol{0}, \quad p = 1, \ldots, P, \tag{21b}$$

$$\boldsymbol{w}_{k_1} = \ldots = \boldsymbol{w}_{k_{j_k}}, \quad k = 1, \ldots, N. \tag{21c}$$

In the following, we shall address the equality constraints (21c) with a diffusion algorithm within each cluster of sub-nodes with the objective of reaching an agreement within each cluster (all sub-nodes converge to the same estimate). Since the diffusion strategy in a single-task network allows the agents to converge to the same limit point asymptotically for sufficiently small constant step-sizes when the network is strongly connected [9], we allow the sub-nodes in cluster $\mathcal{C}_k$ to be connected such that the resultant cluster $\mathcal{C}_k$ is strongly connected. This does not lead to a change in the network topology since each sub-node in a cluster refers to the same agent. We refer to the *virtual* set of neighboring sub-nodes of k_m in $\mathcal{C}_k$ by $\mathcal{N}_{k_m} \cap \mathcal{C}_k$.

The cost function in (21a) can be written as:

$$\sum_{k=1}^{N} \sum_{m=1}^{j_k} c_{k_m} J_k(\boldsymbol{w}_{k_m}) = \boldsymbol{w}_e^{\top} \boldsymbol{\mathcal{R}}_{x,e} \boldsymbol{w}_e - 2\boldsymbol{r}_{dx,e}^{\top} \boldsymbol{w}_e + \boldsymbol{r}_{d,e}^{\top} \mathbb{1}_{N_e \times 1}, \tag{22}$$

where the $N_e \times N_e$ block diagonal matrix $\boldsymbol{\mathcal{R}}_{x,e}$, the $N_e \times 1$ block column vector $\boldsymbol{r}_{dx,e}$, and the $N_e \times 1$ column vector $\boldsymbol{r}_{d,e}$ are given by:

$$\boldsymbol{\mathcal{R}}_{x,e} \triangleq \text{diag}\{\boldsymbol{C}_k \otimes \boldsymbol{R}_{x,k}\}_{k=1}^{N}, \tag{23}$$

$$\boldsymbol{r}_{dx,e} \triangleq \text{col}\{\boldsymbol{c}_k \otimes \boldsymbol{r}_{dx,k}\}_{k=1}^{N}, \tag{24}$$

$$\boldsymbol{r}_{d,e} \triangleq \text{col}\{\sigma_{d,k}^2 \boldsymbol{c}_k\}_{k=1}^{N}, \tag{25}$$

with $\boldsymbol{C}_k \triangleq \text{diag}\{c_{k_m}\}_{m=1}^{j_k}$ and $\boldsymbol{c}_k \triangleq \text{col}\{c_{k_m}\}_{m=1}^{j_k}$.

The equality constraints in (21b)–(21c) can be written more compactly as:

$$\boldsymbol{\mathcal{D}}_e' \boldsymbol{w}_e + \boldsymbol{b}' = \boldsymbol{0} \tag{26}$$

with

$$\boldsymbol{\mathcal{D}}_e' = \begin{bmatrix} \boldsymbol{\mathcal{D}}_e \\ \boldsymbol{\mathcal{H}} \end{bmatrix}, \quad \boldsymbol{b}' = \begin{bmatrix} \boldsymbol{b} \\ \boldsymbol{0} \end{bmatrix}, \tag{27}$$

where $\boldsymbol{\mathcal{D}}_e$ is a $P \times N_e$ block matrix constructed according to (21b) which can be viewed as an expanded form of the $P \times N$ block matrix $\boldsymbol{\mathcal{D}}$, and $\boldsymbol{\mathcal{H}}$ is a $\sum_{k=1}^{N}(j_k - 1) \times N_e$ block matrix constructed according to (21c).

Using similar arguments as in Section II-B, we find that the solution of (21) is given by:

$$\boldsymbol{w}_e^{\star} = \boldsymbol{w}_e^{o} - \boldsymbol{\mathcal{R}}_{x,e}^{-1} \boldsymbol{\mathcal{D}}_e'^{\top} (\boldsymbol{\mathcal{D}}_e' \boldsymbol{\mathcal{R}}_{x,e}^{-1} \boldsymbol{\mathcal{D}}_e'^{\top})^{-1} (\boldsymbol{\mathcal{D}}_e' \boldsymbol{w}_e^{o} + \boldsymbol{b}'), \tag{28}$$

where the $N_e \times 1$ block column vector $\boldsymbol{w}_e^{o}$ is given by:

$$\boldsymbol{w}_e^{o} \triangleq \text{col}\{\mathbb{1}_{j_k \times 1} \otimes \boldsymbol{w}_k^{o}\}_{k=1}^{N}. \tag{29}$$

Let $\boldsymbol{w}_k^{\star}$ denote the k-th block of $\boldsymbol{w}^{\star}$ in (11). The optimum vector $\boldsymbol{w}_e^{\star}$ can be written alternatively as:

$$\boldsymbol{w}_e^{\star} = \text{col}\{\mathbb{1}_{j_k \times 1} \otimes \boldsymbol{w}_k^{\star}\}_{k=1}^{N}. \tag{30}$$

B. Distributed Solution

To solve problem (21) with distributed information access, we propose an iterative algorithm based on diffusion strategies and gradient-projection principle. First, we present the algorithm when the second order moments of the observations are assumed to be known by each sub-node. Although cluster $\mathcal{C}_k$ and agent k refer to the same entity, we shall use the notion of cluster and sub-nodes in order to simplify the presentation.

Let $\boldsymbol{w}_{e,p}$ denote the $i_p \times 1$ block column vector given by $\boldsymbol{w}_{e,p} = \text{col}\{\boldsymbol{w}_{\ell_n}\}_{\ell_n \in \mathcal{I}_{e,p}}$ where i_p is the number of nodes involved in the p-th constraint. Also, note that i_p is the cardinality of $\mathcal{I}_p$ and $\mathcal{I}_{e,p}$. Let Ω_p denote the linear manifold corresponding to the p-th constraint in (21b), namely, $\Omega_p \triangleq \{\boldsymbol{\mathcal{D}}_p \boldsymbol{w}_{e,p} + \boldsymbol{b}_p = \boldsymbol{0}\}$ where $\boldsymbol{\mathcal{D}}_p$ is a $1 \times i_p$ block matrix. Let $\boldsymbol{w}_{k_m}(i)$ be the estimate of $\boldsymbol{w}_k^{\star}$ at sub-node k_m and time instant i. We assume that $k_m \in \mathcal{I}_{e,p}$. Following the same line of reasoning as [11] in the single-task case, and extending the argument to our multitask problem, we arrive at the following diffusion algorithm consisting of three steps:

$$\boldsymbol{\psi}_{k_m}(i+1) = \boldsymbol{w}_{k_m}(i) + \mu\, c_{k_m} [\boldsymbol{r}_{dx,k} - \boldsymbol{R}_{x,k} \boldsymbol{w}_{k_m}(i)] \tag{31a}$$

$$\boldsymbol{\phi}_{k_m}(i+1) = \sum_{k_n \in \mathcal{N}_{k_m} \cap \mathcal{C}_k} a_{k_n,k_m} \boldsymbol{\psi}_{k_n}(i+1) \tag{31b}$$

$$\boldsymbol{w}_{k_m}(i+1) = \left[\text{P}_{\Omega_p}\left(\text{col}\{\boldsymbol{\phi}_{\ell_n}(i+1)\}_{\ell_n \in \mathcal{I}_{e,p}}\right)\right]_{k_m} \tag{31c}$$

where $\mu > 0$ is a constant step-size parameter, $[\boldsymbol{x}]_{k_m}$ is the block of $\boldsymbol{x}$ corresponding to sub-node k_m, and $\boldsymbol{w}_{k_m}(0) = \boldsymbol{w}_k(0)$ for all m. In the first step (31a), also called adaptation step, each sub-node k_m in the network adapts its estimate $\boldsymbol{w}_{k_m}(i)$ via gradient descent on $c_{k_m} J_k(\cdot)$. This step results in the intermediate estimate $\boldsymbol{\psi}_{k_m}(i+1)$.

In the combination step (31b), each sub-node k_m combines its estimate $\boldsymbol{\psi}_{k_m}(i+1)$ with the estimates $\boldsymbol{\psi}_{k_n}(i+1)$ of its intra-cluster neighbors $\mathcal{N}_{k_m} \cap \mathcal{C}_k$. This step results in the intermediate estimate $\boldsymbol{\phi}_{k_m}(i+1)$. The nonnegative coefficients $\{a_{k_n,k_m}\}$ are

chosen to satisfy the following conditions:

$$a_{k_n,k_m} \geq 0, \quad \sum_{k_m \in \mathcal{N}_{k_n} \cap \mathcal{C}_k} a_{k_n,k_m} = 1, \quad \sum_{k_n \in \mathcal{N}_{k_m} \cap \mathcal{C}_k} a_{k_n,k_m} = 1,$$
$$\text{and } a_{k_n,k_m} = 0 \text{ if } k_n \notin \mathcal{N}_{k_m} \cap \mathcal{C}_k. \tag{32}$$

Collecting these coefficients into a $j_k \times j_k$ matrix $\boldsymbol{A}_k$ for each cluster $\mathcal{C}_k$, it follows that $\boldsymbol{A}_k$ is doubly stochastic.

Let M_p denote the dimension of the vector $\boldsymbol{w}_{e,p}$, i.e., $M_p = \sum_{\ell_n \in \mathcal{I}_{e,p}} M_\ell$. Before describing the third step, we recall that the projection of any point $\boldsymbol{y}$ onto Ω_p has the form:

$$\mathrm{P}_{\Omega_p}(\boldsymbol{y}) = \boldsymbol{\mathcal{P}}_p \boldsymbol{y} - \boldsymbol{f}_p \tag{33}$$

where

$$\boldsymbol{\mathcal{P}}_p \triangleq \boldsymbol{I}_{M_p} - \boldsymbol{\mathcal{D}}_p^\dagger \boldsymbol{\mathcal{D}}_p \qquad \text{and} \qquad \boldsymbol{f}_p \triangleq \boldsymbol{\mathcal{D}}_p^\dagger \boldsymbol{b}_p. \tag{34}$$

To evaluate the block $\big[\mathrm{P}_{\Omega_p}(\boldsymbol{y})\big]_{k_m}$, even if sub-node k_m is only in charge of estimating $\boldsymbol{w}_{k_m}$, it needs the entire vector $\boldsymbol{y}$, the $M_k \times M_p$ matrix $[\boldsymbol{\mathcal{P}}_p]_{k_m,\bullet}$, and the $M_k \times 1$ vector $[\boldsymbol{f}_p]_{k_m}$. In the projection step (31c), each sub-node $k_m \in \mathcal{I}_{e,p}$ collects the intermediate estimates $\boldsymbol{\phi}_{\ell_n}(i+1)$ from all sub-nodes $\ell_n \in \mathcal{I}_{e,p}$ and combines them according to (31c). This step results in the estimate $\boldsymbol{w}_{k_m}(i+1)$ of $\boldsymbol{w}_k^\star$ at sub-node k_m and iteration $i+1$.

The adaptation step (31a) requires knowledge of the second-order moments of data. Proceeding as in the centralized case, and replacing the moments by instantaneous approximations, we obtain algorithm (35) for solving (21) in a distributed way:

$$\boldsymbol{\psi}_{k_m}(i+1) = \boldsymbol{w}_{k_m}(i) + \mu\, c_{k_m}\, \boldsymbol{x}_k(i)[d_k(i) - \boldsymbol{x}_k^\top(i)\boldsymbol{w}_{k_m}(i)], \tag{35a}$$

$$\boldsymbol{\phi}_{k_m}(i+1) = [\boldsymbol{\mathcal{P}}_p]_{k_m,\bullet} \cdot \mathrm{col}\big\{\boldsymbol{\psi}_{\ell_n}(i+1)\big\}_{\ell_n \in \mathcal{I}_{e,p}} - [\boldsymbol{f}_p]_{k_m}, \tag{35b}$$

$$\boldsymbol{w}_{k_m}(i+1) = \sum_{k_n \in \mathcal{N}_{k_m} \cap \mathcal{C}_k} a_{k_n,k_m}\, \boldsymbol{\phi}_{k_n}(i+1). \tag{35c}$$

Compared to (31), observe in (35) that each sub-node k_m projects its intermediate estimate before combining it. We recommend this permutation since it allows, with the appropriate parameter settings described below, to reduce the algorithm complexity without compromising its convergence, as confirmed in the sequel. Consider any agent k. By setting factors c_{k_m} to $\frac{1}{j_k}$ for all $m = 1, \ldots, j_k$, and combining the intermediate estimate $\boldsymbol{\phi}_{k_m}(i+1)$ at each sub-node k_m with the estimates of all other sub-nodes available at node k using uniform combination coefficients, i.e., $\mathcal{N}_{k_m} \cap \mathcal{C}_k = \mathcal{C}_k$ and $a_{k_n,k_m} = \frac{1}{j_k}$ for $n = 1, \ldots, j_k$, (35a) and (35c) reduce to:

$$\boldsymbol{\psi}_{k_m}(i+1) = \boldsymbol{\psi}_k(i+1), \quad \text{for } m = 1, \ldots, j_k, \tag{36}$$

$$\boldsymbol{w}_{k_m}(i+1) = \boldsymbol{w}_k(i+1), \quad \text{for } m = 1, \ldots, j_k, \tag{37}$$

where $\boldsymbol{\psi}_k(i+1)$ and $\boldsymbol{w}_k(i+1)$ are given by:

$$\boldsymbol{\psi}_k(i+1) = \boldsymbol{w}_k(i) + \frac{\mu}{j_k}\boldsymbol{x}_k(i)\big[d_k(i) - \boldsymbol{x}_k^\top(i)\boldsymbol{w}_k(i)\big], \tag{38}$$

$$\boldsymbol{w}_k(i+1) = \frac{1}{j_k}\sum_{m=1}^{j_k} \boldsymbol{\phi}_{k_m}(i+1). \tag{39}$$

In this case, at each agent k, the algorithm (35) becomes:

$$\boldsymbol{\psi}_k(i+1) = \boldsymbol{w}_k(i) + \frac{\mu}{j_k}\boldsymbol{x}_k(i)\big[d_k(i) - \boldsymbol{x}_k^\top(i)\boldsymbol{w}_k(i)\big] \tag{40a}$$

$$\boldsymbol{\phi}_{k_m}(i+1) = [\boldsymbol{\mathcal{P}}_p]_{k_m,\bullet} \cdot \mathrm{col}\big\{\boldsymbol{\psi}_\ell(i+1)\big\}_{\ell \in \mathcal{I}_p} - [\boldsymbol{f}_p]_{k_m}, \quad k_m \in \mathcal{I}_{e,p}, \quad m = 1, \ldots, j_k, \tag{40b}$$

$$\boldsymbol{w}_k(i+1) = \frac{1}{j_k}\sum_{m=1}^{j_k} \boldsymbol{\phi}_{k_m}(i+1). \tag{40c}$$

Instead of maintaining and updating j_k coefficient vectors $\boldsymbol{\psi}_{k_m}(i+1)$, agent k maintains and updates only one parameter vector $\boldsymbol{\psi}_k(i+1)$. Then, it transmits the vector $\boldsymbol{\psi}_k(i+1)$ to its neighbors, receives $\{\boldsymbol{\psi}_\ell(i+1)\}$ from its neighborhood, and generates j_k parameter vectors $\boldsymbol{\phi}_{k_m}(i+1)$ by projecting onto its constraints. Finally, it combines these vectors to obtain $\boldsymbol{w}_k(i+1)$, i.e., the estimate of $\boldsymbol{w}_k^\star$ at iteration $i+1$. Therefore, with this setting, the computational and communication complexity of our distributed algorithm is significantly reduced.

IV. Performance Analysis Relative to $\boldsymbol{w}_e^o$

A. Network Error Vector Recursion

We shall first study the stochastic behavior of algorithm (35) with respect to the optimal parameter vector $\boldsymbol{w}_e^o$. We introduce the error vector $\widetilde{\boldsymbol{w}}_{k_m}(i) \triangleq \boldsymbol{w}_k^o - \boldsymbol{w}_{k_m}(i)$ and the intermediate error vectors $\widetilde{\boldsymbol{\psi}}_{k_m}(i) \triangleq \boldsymbol{w}_k^o - \boldsymbol{\psi}_{k_m}(i)$ and $\widetilde{\boldsymbol{\phi}}_{k_m}(i) \triangleq \boldsymbol{w}_k^o - \boldsymbol{\phi}_{k_m}(i)$. We further introduce the $N_e \times 1$ block network error vector:

$$\widetilde{\boldsymbol{w}}_e(i) \triangleq \mathrm{col}\Big\{\mathrm{col}\big\{\widetilde{\boldsymbol{w}}_{k_m}(i)\big\}_{m=1}^{j_k}\Big\}_{k=1}^{N}. \tag{41}$$

Let M_e denote the length of the network error vector $\widetilde{\boldsymbol{w}}_e(i)$, that is, $M_e \triangleq \sum_{k=1}^N j_k M_k$. Using the linear model (2), the estimation error in the adaptation step (35a) can be written as:

$$d_k(i) - \boldsymbol{x}_k^\top(i)\boldsymbol{w}_{k_m}(i) = \boldsymbol{x}_k^\top(i)\widetilde{\boldsymbol{w}}_{k_m}(i) + z_k(i). \tag{42}$$

Subtracting $\boldsymbol{w}_k^o$ from both sides of the adaptation step (35a), using (42), and collecting the error vectors $\widetilde{\boldsymbol{\psi}}_{k_m}(i)$ into the $N_e \times 1$ block vector $\widetilde{\boldsymbol{\psi}}_e(i) \triangleq \mathrm{col}\{\mathrm{col}\{\widetilde{\boldsymbol{\psi}}_{k_m}(i)\}_{m=1}^{j_k}\}_{k=1}^N$, we obtain:

$$\widetilde{\boldsymbol{\psi}}_e(i+1) = \big[\boldsymbol{I}_{M_e} - \mu\boldsymbol{\mathcal{R}}_{x,e}(i)\big]\widetilde{\boldsymbol{w}}_e(i) - \mu\,\boldsymbol{p}_{zx,e}(i), \tag{43}$$

where

$$\boldsymbol{\mathcal{R}}_{x,e}(i) \triangleq \mathrm{diag}\Big\{\boldsymbol{C}_k \otimes \boldsymbol{x}_k(i)\boldsymbol{x}_k^\top(i)\Big\}_{k=1}^N, \tag{44}$$

$$\boldsymbol{p}_{zx,e}(i) \triangleq \mathrm{col}\Big\{\boldsymbol{c}_k \otimes \boldsymbol{x}_k(i)z_k(i)\Big\}_{k=1}^N. \tag{45}$$

Projecting $\boldsymbol{\psi}_e(i+1)$ onto the sets Ω_p in (33), we obtain from (35b):

$$\boldsymbol{\phi}_e(i+1) = \boldsymbol{\mathcal{P}}_e\boldsymbol{\psi}_e(i+1) - \boldsymbol{f}_e, \tag{46}$$

where

$$\boldsymbol{\psi}_e(i) \triangleq \mathrm{col}\Big\{\mathrm{col}\big\{\boldsymbol{\psi}_{k_m}(i)\big\}_{m=1}^{j_k}\Big\}_{k=1}^{N}, \tag{47}$$

$$\boldsymbol{\phi}_e(i) \triangleq \mathrm{col}\Big\{\mathrm{col}\big\{\boldsymbol{\phi}_{k_m}(i)\big\}_{m=1}^{j_k}\Big\}_{k=1}^{N}, \tag{48}$$

$\boldsymbol{\mathcal{P}}_e$ is an $M_e \times M_e$ orthogonal projection matrix, and $\boldsymbol{f}_e$ is an $M_e \times 1$ vector given by (see [46, Appendix A]):

$$\boldsymbol{\mathcal{P}}_e \triangleq \boldsymbol{I}_{M_e} - \boldsymbol{\mathcal{D}}_e^{\dagger}\boldsymbol{\mathcal{D}}_e = \boldsymbol{I}_{M_e} - \boldsymbol{\mathcal{D}}_e^{\top}(\boldsymbol{\mathcal{D}}_e\boldsymbol{\mathcal{D}}_e^{\top})^{-1}\boldsymbol{\mathcal{D}}_e \tag{49}$$

$$\boldsymbol{f}_e \triangleq \boldsymbol{\mathcal{D}}_e^{\dagger}\boldsymbol{b} = \boldsymbol{\mathcal{D}}_e^{\top}(\boldsymbol{\mathcal{D}}_e\boldsymbol{\mathcal{D}}_e^{\top})^{-1}\boldsymbol{b}. \tag{50}$$

Subtracting $\boldsymbol{w}_e^o$ in (29) from both sides of recursion (46), we obtain:

$$\begin{aligned}\widetilde{\boldsymbol{\phi}}_e(i+1) &\triangleq \mathrm{col}\Big\{\mathrm{col}\big\{\widetilde{\boldsymbol{\phi}}_{k_m}(i+1)\big\}_{m=1}^{j_k}\Big\}_{k=1}^{N}\\ &= \boldsymbol{\mathcal{P}}_e\widetilde{\boldsymbol{\psi}}_e(i+1) + \big(\boldsymbol{I}_{M_e} - \boldsymbol{\mathcal{P}}_e\big)\boldsymbol{w}_e^o + \boldsymbol{f}_e. \end{aligned}\tag{51}$$

Subtracting $\boldsymbol{w}_k^o$ from both sides of the combination step (35c) and using (32), we obtain that the network error vector for the diffusion strategy (35) evolves according to the following recursion:

$$\boxed{\begin{aligned}\widetilde{\boldsymbol{w}}_e(i+1) &= \boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{P}}_e\big[\boldsymbol{I}_{M_e} - \mu\boldsymbol{\mathcal{R}}_{x,e}(i)\big]\widetilde{\boldsymbol{w}}_e(i)\\ &\quad - \mu\boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{P}}_e\boldsymbol{p}_{zx,e}(i) + \boldsymbol{\mathcal{A}}^{\top}(\boldsymbol{I}_{M_e} - \boldsymbol{\mathcal{P}}_e)\boldsymbol{w}_e^o + \boldsymbol{\mathcal{A}}^{\top}\boldsymbol{f}_e,\end{aligned}}\tag{52}$$

where $\boldsymbol{\mathcal{A}} \triangleq \mathrm{diag}\{\boldsymbol{A}_k \otimes \boldsymbol{I}_{M_k}\}_{k=1}^{N}$. Before proceeding, let us introduce the following assumption on the regression data.

Assumption 1: (Independent regressors) The regression vectors $\boldsymbol{x}_k(i)$ arise from a zero-mean random process that is temporally white and spatially independent.

Under this assumption, $\boldsymbol{x}_k(i)$ is independent of $\boldsymbol{w}_{\ell_m}(j)$ for $i \geq j$ and for all ℓ_m. This assumption is commonly used in the adaptive filtering literature since it helps simplify the analysis, and the performance results obtained under this assumption match well the actual performance of stand-alone filters for sufficiently small step-sizes [43].

B. Mean Behavior Analysis

Recursion (52) can be rewritten in a more compact form:

$$\widetilde{\boldsymbol{w}}_e(i+1) = \boldsymbol{\mathcal{B}}(i)\widetilde{\boldsymbol{w}}_e(i) - \mu\boldsymbol{g}(i) + \boldsymbol{r}, \tag{53}$$

where we introduced the following notations:

$$\boldsymbol{\mathcal{B}}(i) \triangleq \boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{P}}_e\big[\boldsymbol{I}_{M_e} - \mu\boldsymbol{\mathcal{R}}_{x,e}(i)\big], \tag{54}$$

$$\boldsymbol{g}(i) \triangleq \boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{P}}_e\boldsymbol{p}_{zx,e}(i), \tag{55}$$

$$\boldsymbol{r} \triangleq \boldsymbol{\mathcal{A}}^{\top}(\boldsymbol{I}_{M_e} - \boldsymbol{\mathcal{P}}_e)\boldsymbol{w}_e^o + \boldsymbol{\mathcal{A}}^{\top}\boldsymbol{f}_e. \tag{56}$$

Taking the expectation of both sides of recursion (53), using Assumption 1, and $\mathbb{E}\,\boldsymbol{g}(i) = \boldsymbol{0}$, we find that the mean error vector evolves according to the recursion:

$$\mathbb{E}\,\widetilde{\boldsymbol{w}}_e(i+1) = \boldsymbol{\mathcal{B}}\,\mathbb{E}\,\widetilde{\boldsymbol{w}}_e(i) + \boldsymbol{r}, \tag{57}$$

where

$$\boldsymbol{\mathcal{B}} \triangleq \mathbb{E}\,\boldsymbol{\mathcal{B}}(i) = \boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{P}}_e\big(\boldsymbol{I}_{M_e} - \mu\boldsymbol{\mathcal{R}}_{x,e}\big), \tag{58}$$

with $\boldsymbol{\mathcal{R}}_{x,e} = \mathbb{E}\boldsymbol{\mathcal{R}}_{x,e}(i)$ given in (23)[1]. Recursion (57) converges as $i \to \infty$ if the matrix $\boldsymbol{\mathcal{B}}$ is stable. If we let $i \to \infty$ on both sides of (57), we find that the asymptotic mean bias is given by:

$$\mathbb{E}\,\widetilde{\boldsymbol{w}}_e(\infty) = \lim_{i\to\infty}\mathbb{E}\,\widetilde{\boldsymbol{w}}_e(i) = (\boldsymbol{I}_{M_e} - \boldsymbol{\mathcal{B}})^{-1}\boldsymbol{r}. \tag{59}$$

It is known that any induced matrix norm is lower bounded by the spectral radius of the matrix. We can thus write in terms of the 2-induced matrix norm:

$$\rho(\boldsymbol{\mathcal{B}}) \leq \|\boldsymbol{\mathcal{A}}^{\top}\|_2 \cdot \|\boldsymbol{\mathcal{P}}_e\|_2 \cdot \|\boldsymbol{I}_{M_e} - \mu\boldsymbol{\mathcal{R}}_{x,e}\|_2, \tag{60}$$

where we used the sub-multiplicative property of the 2-induced norm. Since $\boldsymbol{\mathcal{P}}_e$ is an orthogonal projection matrix and $\boldsymbol{\mathcal{A}}^{\top}$ is a doubly-stochastic matrix, their 2-induced norms are equal to one. Since the matrix $\boldsymbol{I}_{M_e} - \mu\boldsymbol{\mathcal{R}}_{x,e}$ is a symmetric block diagonal matrix, its 2-induced norm agrees with its spectral radius:

$$\begin{aligned}\|\boldsymbol{I}_{M_e} - \mu\boldsymbol{\mathcal{R}}_{x,e}\|_2 &= \rho(\boldsymbol{I}_{M_e} - \mu\boldsymbol{\mathcal{R}}_{x,e})\\ &= \max_{1\leq k\leq N}\max_{1\leq m\leq j_k}\rho(\boldsymbol{I}_{M_k} - \mu\, c_{k_m}\boldsymbol{R}_{x,k}).\end{aligned}\tag{61}$$

Thus, the stability of $\boldsymbol{\mathcal{B}}$ is ensured by choosing μ such that:

$$0 < \mu < \frac{2}{c_{k,\max}\cdot\lambda_{\max}(\boldsymbol{R}_{x,k})}, \quad \forall k = 1,\ldots,N. \tag{62}$$

where $c_{k,\max} \triangleq \max_{1\leq m\leq j_k} c_{k_m}$. We observe that when $\boldsymbol{w}_e^{\star} = \boldsymbol{w}_e^o$, i.e., perfect model scenario where $\boldsymbol{w}^o$ satisfies the linear equality constraints, the bias reduces to $\boldsymbol{0}$.

C. Mean-Square-Error Behavior Analysis

To perform the mean-square-error analysis, we shall use the block Kronecker product operator [47] and the block vectorization operator bvec$(\cdot)$. As explained in [9], these block operators preserve the locality of the blocks in the original matrix arguments. To analyze the convergence in mean-square-error sense, we consider the variance of the weight error vector $\widetilde{\boldsymbol{w}}_e(i)$, weighted by any positive-definite matrix $\boldsymbol{\Sigma}$, that is, $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_e(i)\|_{\boldsymbol{\Sigma}}^2$, where $\|\widetilde{\boldsymbol{w}}_e(i)\|_{\boldsymbol{\Sigma}}^2 \triangleq \widetilde{\boldsymbol{w}}_e^{\top}(i)\boldsymbol{\Sigma}\widetilde{\boldsymbol{w}}_e(i)$. The freedom in selecting $\boldsymbol{\Sigma}$ allows us to extract various types of information about the network and the sub-nodes. From (53) and Assumption 1, we obtain:

$$\begin{aligned}\mathbb{E}\{\|\widetilde{\boldsymbol{w}}_e(i+1)\|_{\boldsymbol{\Sigma}}^2\} &= \mathbb{E}\{\|\widetilde{\boldsymbol{w}}_e(i)\|_{\boldsymbol{\Sigma}'}^2\} + \mu^2\mathbb{E}\{\|\boldsymbol{g}(i)\|_{\boldsymbol{\Sigma}}^2\}\\ &\quad + \|\boldsymbol{r}\|_{\boldsymbol{\Sigma}}^2 + 2\boldsymbol{r}^{\top}\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}\mathbb{E}\,\widetilde{\boldsymbol{w}}_e(i),\end{aligned}\tag{63}$$

where matrix $\boldsymbol{\Sigma}'$ is given by:

$$\boldsymbol{\Sigma}' \triangleq \mathbb{E}\{\boldsymbol{\mathcal{B}}^{\top}(i)\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}(i)\}. \tag{64}$$

Let $\boldsymbol{\sigma}$ denotes the $M_e^2 \times 1$ vector representation of $\boldsymbol{\Sigma}$ that is obtained by the block vectorization operator, namely, $\boldsymbol{\sigma} \triangleq \mathrm{bvec}(\boldsymbol{\Sigma})$. In the sequel, it will be more convenient to work with $\boldsymbol{\sigma}$ than with $\boldsymbol{\Sigma}$ itself. We will use the notations $\|\boldsymbol{x}\|_{\boldsymbol{\Sigma}}^2$ and $\|\boldsymbol{x}\|_{\boldsymbol{\sigma}}^2$ to denote the same quantity $\boldsymbol{x}^{\top}\boldsymbol{\Sigma}\boldsymbol{x}$. Let $\boldsymbol{\sigma}' = \mathrm{bvec}(\boldsymbol{\Sigma}')$. Using

[1] If $\boldsymbol{U}(i)$ is a random matrix, its expected value $\mathbb{E}\,\boldsymbol{U}(i)$ is denoted by $\boldsymbol{U}$.

the property $\text{bvec}(\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{W}) = (\boldsymbol{W}^{\top} \otimes_b \boldsymbol{U})\boldsymbol{\sigma}$, the vector $\boldsymbol{\sigma}'$ can be related to $\boldsymbol{\sigma}$:

$$\boldsymbol{\sigma}' = \boldsymbol{\mathcal{F}}\boldsymbol{\sigma}, \tag{65}$$

where $\boldsymbol{\mathcal{F}}$ is an $M_e^2 \times M_e^2$ matrix given by:

$$\boldsymbol{\mathcal{F}} \triangleq \mathbb{E}\{\boldsymbol{\mathcal{B}}^{\top}(i) \otimes_b \boldsymbol{\mathcal{B}}^{\top}(i)\}. \tag{66}$$

The evaluation of the matrix $\boldsymbol{\mathcal{F}}$ requires knowledge of the fourth-order moments of the regression vectors. In practice, when $\mathbb{E}\{\boldsymbol{\mathcal{R}}_{x,e}(i) \otimes_b \boldsymbol{\mathcal{R}}_{x,e}(i)\}$ can be computed, as in the case of zero-mean Gaussian regressors (see [46, Appendix B]), the matrix $\boldsymbol{\mathcal{F}}$ can be calculated in closed form and its stability can be checked for a given μ.[2]

The second term on the RHS of relation (63) can be written as:

$$\begin{aligned}\mu^2\mathbb{E}\{\|\boldsymbol{g}(i)\|_{\boldsymbol{\Sigma}}^2\} &= \mu^2\mathbb{E}\{\boldsymbol{g}^{\top}(i)\boldsymbol{\Sigma}\boldsymbol{g}(i)\} \\ &= \mu^2\text{tr}(\boldsymbol{\Sigma}\boldsymbol{\mathcal{G}}) = \mu^2\big[\text{bvec}(\boldsymbol{\mathcal{G}}^{\top})\big]^{\top}\boldsymbol{\sigma},\end{aligned} \tag{67}$$

where $\boldsymbol{\mathcal{G}}$ is the $M_e \times M_e$ matrix given by:

$$\begin{aligned}\boldsymbol{\mathcal{G}} &\triangleq \mathbb{E}\{\boldsymbol{g}(i)\boldsymbol{g}^{\top}(i)\} \\ &= \boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{P}}_e\text{diag}\big\{\boldsymbol{c}_k\boldsymbol{c}_k^{\top} \otimes \sigma_{z,k}^2\boldsymbol{R}_{x,k}\big\}_{k=1}^{N}\boldsymbol{\mathcal{P}}_e\boldsymbol{\mathcal{A}}.\end{aligned} \tag{68}$$

Similarly, the third term on the RHS of relation (63) can be written as:

$$\|\boldsymbol{r}\|_{\boldsymbol{\Sigma}}^2 = \big[\text{bvec}(\boldsymbol{r}\boldsymbol{r}^{\top})\big]^{\top}\boldsymbol{\sigma}, \tag{69}$$

and the fourth term can be written as:

$$\begin{aligned}2\boldsymbol{r}^{\top}\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}\mathbb{E}\,\widetilde{\boldsymbol{w}}_e(i) &= 2\text{tr}(\boldsymbol{r}^{\top}\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}\mathbb{E}\,\widetilde{\boldsymbol{w}}_e(i)) \\ &= 2\big[\text{bvec}(\boldsymbol{r}\mathbb{E}\{\widetilde{\boldsymbol{w}}_e^{\top}(i)\}\boldsymbol{\mathcal{B}}^{\top})\big]^{\top}\boldsymbol{\sigma}.\end{aligned} \tag{70}$$

Let us define the $M_e \times M_e$ time dependent matrix $\boldsymbol{\mathcal{Y}}(i)$ given by:

$$\boldsymbol{\mathcal{Y}}(i) \triangleq \mu^2\boldsymbol{\mathcal{G}}^{\top} + \boldsymbol{r}\boldsymbol{r}^{\top} + 2\boldsymbol{r}\mathbb{E}\{\widetilde{\boldsymbol{w}}_e(i)\}^{\top}\boldsymbol{\mathcal{B}}^{\top}. \tag{71}$$

Then, the variance relation (63) can be expressed as:

$$\mathbb{E}\{\|\widetilde{\boldsymbol{w}}_e(i+1)\|_{\boldsymbol{\sigma}}^2\} = \mathbb{E}\{\|\widetilde{\boldsymbol{w}}_e(i)\|_{\boldsymbol{\mathcal{F}}\boldsymbol{\sigma}}^2\} + \big[\text{bvec}(\boldsymbol{\mathcal{Y}}(i))\big]^{\top}\boldsymbol{\sigma}. \tag{72}$$

Provided that $\boldsymbol{\mathcal{F}}$ is stable, recursion (72) is stable. Since $\boldsymbol{\mathcal{G}}$, $\boldsymbol{r}$, $\boldsymbol{\mathcal{B}}$, $\boldsymbol{\sigma}$, and μ are constant and finite terms, the boundedness of $\big[\text{bvec}(\boldsymbol{\mathcal{Y}}(i))\big]^{\top}\boldsymbol{\sigma}$ depends on $\mathbb{E}\,\widetilde{\boldsymbol{w}}_e(i)$ being bounded. We know from (57) that $\mathbb{E}\,\widetilde{\boldsymbol{w}}_e(i)$ is bounded if the step-size μ is chosen according to condition (62) because (57) is a Bounded-Input Bounded-Output (BIBO) stable recursion with a bounded driving term $\boldsymbol{r}$. It follows that $\big[\text{bvec}(\boldsymbol{\mathcal{Y}}(i))\big]^{\top}\boldsymbol{\sigma}$ is uniformly bounded. As a result, the algorithm is mean-square-error stable, i.e., $\mathbb{E}\{\|\widetilde{\boldsymbol{w}}_e(i+1)\|_{\boldsymbol{\sigma}}^2\}$ converges to a bounded value as $i \to \infty$, if μ is chosen such that $\boldsymbol{\mathcal{F}}$ in (66) is stable in addition to condition (62) that ensures mean stability. As explained above, step-sizes that ensure stability in the mean and that are sufficiently small will also ensure stability in the mean-square.

Following similar arguments as in [22], [23], [26] and doing the required adjustments, we find that the weighted variance $\mathbb{E}\{\|\widetilde{\boldsymbol{w}}_e(i+1)\|_{\boldsymbol{\sigma}}^2\}$ evolves according to the following recursion:

$$\begin{aligned}\mathbb{E}\{\|\widetilde{\boldsymbol{w}}_e(i+1)\|_{\boldsymbol{\sigma}}^2\} &= \mathbb{E}\{\|\widetilde{\boldsymbol{w}}_e(i)\|_{\boldsymbol{\sigma}}^2\} \\ &+ \big[\text{bvec}(\mathbb{E}\{\widetilde{\boldsymbol{w}}_e(0)\widetilde{\boldsymbol{w}}_e^{\top}(0)\})\big]^{\top}(\boldsymbol{\mathcal{F}} - \boldsymbol{I}_{M_e^2})\boldsymbol{\mathcal{F}}^i\boldsymbol{\sigma} \\ &+ \big[\text{bvec}(\boldsymbol{\mathcal{Y}}(i))\big]^{\top}\boldsymbol{\sigma} + \boldsymbol{\Gamma}(i)\boldsymbol{\sigma},\end{aligned} \tag{73}$$

where $\widetilde{\boldsymbol{w}}_e(0)$ is the initial condition and $\boldsymbol{\Gamma}(i+1)$ is a $1 \times M_e^2$ vector that can be evaluated from $\boldsymbol{\Gamma}(i)$ according to:

$$\boldsymbol{\Gamma}(i+1) = \boldsymbol{\Gamma}(i)\boldsymbol{\mathcal{F}} + \big[\text{bvec}(\boldsymbol{\mathcal{Y}}(i))\big]^{\top}(\boldsymbol{\mathcal{F}} - \boldsymbol{I}_{M_e^2}), \tag{74}$$

with $\boldsymbol{\Gamma}(0) = \boldsymbol{0}$.

The steady-state network performance with metric $\boldsymbol{\sigma}_{ss}$ is defined as:

$$\zeta^{\star} = \lim_{i\to\infty}\mathbb{E}\|\widetilde{\boldsymbol{w}}_e(i)\|_{\boldsymbol{\sigma}_{ss}}^2. \tag{75}$$

If the matrix $\boldsymbol{\mathcal{F}}$ is stable, from the recursive expression (72), we obtain as $i \to \infty$:

$$\lim_{i\to\infty}\mathbb{E}\{\|\widetilde{\boldsymbol{w}}_e(i)\|_{(\boldsymbol{I}_{M_e^2} - \boldsymbol{\mathcal{F}})\boldsymbol{\sigma}}^2\} = [\text{bvec}(\boldsymbol{\mathcal{Y}}(\infty))]^{\top}\boldsymbol{\sigma}, \tag{76}$$

where, from (71) and (59), we have:

$$\boldsymbol{\mathcal{Y}}(\infty) = \mu^2\boldsymbol{\mathcal{G}}^{\top} + \boldsymbol{r}\boldsymbol{r}^{\top} + 2\boldsymbol{r}\mathbb{E}\{\widetilde{\boldsymbol{w}}_e(\infty)\}^{\top}\boldsymbol{\mathcal{B}}^{\top}. \tag{77}$$

To obtain (75), we replace $\boldsymbol{\sigma}$ in (76) by $(\boldsymbol{I}_{M_e^2} - \boldsymbol{\mathcal{F}})^{-1}\boldsymbol{\sigma}_{ss}$. The theoretical findings (57), (59), (73), and (76) allow us to predict the behavior in the mean and in the mean-square-error sense of the stochastic algorithm (35) w.r.t. the parameter vector $\boldsymbol{w}_e^o$. Note that, the network MSD w.r.t. $\boldsymbol{w}_e^o$ given by:

$$\text{MSD}_{\text{net}}(i) \triangleq \frac{1}{N}\sum_{k=1}^{N}\left(\frac{1}{j_k}\sum_{m=1}^{j_k}\mathbb{E}\|\widetilde{\boldsymbol{w}}_{k_m}(i)\|^2\right), \tag{78}$$

can be obtained by setting $\boldsymbol{\Sigma} = \frac{1}{N}\,\text{diag}\{\frac{1}{j_k}\boldsymbol{I}_{j_k\cdot M_k}\}_{k=1}^{N}$.

V. Performance Analysis Relative to $\boldsymbol{w}_e^{\star}$

We shall now study the convergence behavior of algorithm (35) toward the solution $\boldsymbol{w}_e^{\star}$ of the optimization problem with constraints (21). To this end, we introduce for each sub-node k_m the weight error vector:

$$\widetilde{\boldsymbol{w}}'_{k_m}(i) \triangleq \boldsymbol{w}_k^{\star} - \boldsymbol{w}_{k_m}(i), \tag{79}$$

and the $N_e \times 1$ network block error vector:

$$\widetilde{\boldsymbol{w}}'_e(i) \triangleq \text{col}\Big\{\text{col}\big\{\widetilde{\boldsymbol{w}}'_{k_m}(i)\big\}_{m=1}^{j_k}\Big\}_{k=1}^{N} \tag{80}$$

We note that the behavior of algorithm (35) with respect to $\boldsymbol{w}_e^{\star}$ can be deduced from its behavior with respect to $\boldsymbol{w}_e^o$ using the following relation:

$$\widetilde{\boldsymbol{w}}'_e(i+1) = \widetilde{\boldsymbol{w}}_e(i+1) - \boldsymbol{w}_e^{\delta}. \tag{81}$$

where $\boldsymbol{w}_e^{\delta} \triangleq \boldsymbol{w}_e^o - \boldsymbol{w}_e^{\star}$. Using (81) with (52), the fact that $\boldsymbol{w}_e^{\star}$ verifies the constraints $\{\boldsymbol{\mathcal{D}}_e\boldsymbol{w}_e + \boldsymbol{b} = \boldsymbol{0}\}$, namely,

$$\boldsymbol{\mathcal{P}}_e\boldsymbol{w}_e^{\star} - \boldsymbol{f}_e = \boldsymbol{w}_e^{\star}, \tag{82}$$

[2] When $\mathbb{E}\{\boldsymbol{\mathcal{R}}_{x,e}(i) \otimes_b \boldsymbol{\mathcal{R}}_{x,e}(i)\}$ cannot be evaluated, a common alternative is to use the approximation $\boldsymbol{\mathcal{F}} \approx \boldsymbol{\mathcal{B}}^{\top} \otimes_b \boldsymbol{\mathcal{B}}^{\top}$ for sufficiently small step-sizes (see [9], [11]). In this case, we have $\rho(\boldsymbol{\mathcal{F}}) \approx \rho(\boldsymbol{\mathcal{B}}^{\top} \otimes_b \boldsymbol{\mathcal{B}}^{\top}) = \rho(\boldsymbol{\mathcal{B}})^2$. As long as this approximation is reasonable, the stability of $\boldsymbol{\mathcal{F}}$ is ensured if $\rho(\boldsymbol{\mathcal{B}}) < 1$, i.e., if the step-size is chosen according to condition (62).

and the fact that $\boldsymbol{\mathcal{A}}^\top \mathbb{1} = \mathbb{1}$, we obtain that $\widetilde{\boldsymbol{w}}'_e(i+1)$ evolves according to the following recursion:

$$\begin{aligned}\widetilde{\boldsymbol{w}}'_e(i+1) &= \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{P}}_e \big[\boldsymbol{I}_{M_e} - \mu \boldsymbol{\mathcal{R}}_{x,e}(i)\big] \widetilde{\boldsymbol{w}}'_e(i) \\ &\quad - \mu \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{P}}_e \boldsymbol{p}_{zx,e}(i) - \mu \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{P}}_e \boldsymbol{\mathcal{R}}_{x,e}(i) \boldsymbol{w}_e^\delta.\end{aligned} \tag{83}$$

Taking the expectation of both sides of recursion (83), using Assumption 1, and $\mathbb{E}\, \boldsymbol{p}_{zx,e}(i) = \boldsymbol{0}$, the mean error vector evolves according to:

$$\mathbb{E}\, \widetilde{\boldsymbol{w}}'_e(i+1) = \boldsymbol{\mathcal{B}}\, \mathbb{E}\, \widetilde{\boldsymbol{w}}'_e(i) - \mu \boldsymbol{r}', \tag{84}$$

where $\boldsymbol{\mathcal{B}}$ is given by (58) and

$$\boldsymbol{r}' \triangleq \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{P}}_e \boldsymbol{\mathcal{R}}_{x,e} \boldsymbol{w}_e^\delta. \tag{85}$$

Using arguments similar to Section IV-B, we find that the multitask diffusion algorithm (35) is stable in the mean if the step-size is chosen such that the matrix $\boldsymbol{\mathcal{B}}$ is stable. The asymptotic mean bias is given by:

$$\lim_{i\to\infty} \mathbb{E}\, \widetilde{\boldsymbol{w}}'_e(i) = -\mu [\boldsymbol{I}_{M_e} - \boldsymbol{\mathcal{B}}]^{-1} \boldsymbol{r}'. \tag{86}$$

Note that the bias depends on the step-size μ and the vector $\boldsymbol{w}_e^\delta = \boldsymbol{w}_e^o - \boldsymbol{w}_e^\star$. In the next section, we shall illustrate with simulation results that $\lim_{i\to\infty} \|\mathbb{E}\, \widetilde{\boldsymbol{w}}'_e(i)\|^2$ is on the order of μ^2. The bias (86) is $\boldsymbol{0}$ in two cases: 1) in the perfect model scenario where $\boldsymbol{w}_k^o$ satisfy the constraints ($\boldsymbol{w}_e^\delta = \boldsymbol{0}$); 2) if each agent is involved in at most one constraint ($\boldsymbol{\mathcal{D}}_e = \boldsymbol{\mathcal{D}}'_e = \boldsymbol{\mathcal{D}}$). In this second case, consider (85) and observe that $\boldsymbol{\mathcal{A}} = \boldsymbol{I}_{M_e}$. Replacing $\boldsymbol{w}_e^\delta$ by its expression obtained from (28), and $\boldsymbol{\mathcal{P}}_e$ by (49), yields $\boldsymbol{r}' = \boldsymbol{0}$.

To obtain the behavior of algorithm (35) toward $\boldsymbol{w}_e^\star$ in the mean-square sense, we use (81) to write:

$$\begin{aligned}\mathbb{E}\{\|\widetilde{\boldsymbol{w}}'_e(i+1)\|^2_{\boldsymbol{\Sigma}}\} &= \mathbb{E}\{\|\widetilde{\boldsymbol{w}}_e(i+1)\|^2_{\boldsymbol{\Sigma}}\} \\ &\quad - 2\,\mathbb{E}\{\widetilde{\boldsymbol{w}}_e^\top(i+1)\}\boldsymbol{\Sigma}\boldsymbol{w}_e^\delta + \|\boldsymbol{w}_e^\delta\|^2_{\boldsymbol{\Sigma}}.\end{aligned} \tag{87}$$

The transient and steady-state behaviors of $\mathbb{E}\{\|\widetilde{\boldsymbol{w}}'_e(i)\|^2_{\boldsymbol{\Sigma}}\}$ can be derived from the models derived for $\widetilde{\boldsymbol{w}}_e(i)$ in the mean and mean-square sense. We shall show with simulation results that the steady-state $\lim_{i\to\infty} \mathbb{E}\, \|\widetilde{\boldsymbol{w}}'_e(i)\|^2$ is on the order of μ. We observed experimentally that modeling the behavior of $\mathbb{E}\{\|\widetilde{\boldsymbol{w}}'_e(i)\|^2_{\boldsymbol{\Sigma}}\}$ accurately needs the exact expression of $\boldsymbol{\mathcal{F}}$. For zero-mean real valued regressors with $M_k = M_0\ \forall\ k$, the evaluation of $\boldsymbol{\mathcal{F}}$ leads to (see [46, Appendix B]):

$$\begin{aligned}\boldsymbol{\mathcal{F}} &= \boldsymbol{\mathcal{B}}^\top \otimes_b \boldsymbol{\mathcal{B}}^\top \\ &\quad + \mu^2 \sum_{k=1}^{N} \Big[\big(\mathcal{S}_k^\top (\boldsymbol{I}_{N_e} \otimes \boldsymbol{R}_{x,k}) \otimes_b \mathcal{S}_k (\boldsymbol{I}_{N_e} \otimes \boldsymbol{R}_{x,k})\big) \\ &\qquad + \big(\mathcal{S}_k^\top \otimes_b (\mathcal{S}_k (\boldsymbol{I}_{N_e} \otimes \boldsymbol{R}_{x,k}))\big) \\ &\quad \big(\boldsymbol{I}_{N_e^2} \otimes \mathrm{vec}(\boldsymbol{I}_{M_0}) \otimes [\mathrm{vec}(\boldsymbol{R}_{x,k})]^\top\big) \Big] (\boldsymbol{\mathcal{P}}_e \boldsymbol{\mathcal{A}} \otimes_b \boldsymbol{\mathcal{P}}_e \boldsymbol{\mathcal{A}}),\end{aligned} \tag{88}$$

where $\mathcal{S}_k$ is the $N \times N$ block diagonal matrix whose (k,k)-th block is equal to $\boldsymbol{C}_k \otimes \boldsymbol{I}_{M_0}$.

VI. Simulation Results

Throughout this section, the factors c_{k_m} were set to $\frac{1}{j_k}$, and $\mathcal{N}_{k_m} \cap \mathcal{C}_k = \mathcal{C}_k$ for all m. We run algorithm (35) with uniform combination coefficients $a_{k_n,k_m} = \frac{1}{j_k}$ for all n.

A. Theoretical Model Validation

We considered a network consisting of 15 agents with the topology shown in Fig. 3. The regression vectors $\boldsymbol{x}_k(i)$ were 2×1 zero-mean Gaussian distributed with covariance matrices $\boldsymbol{R}_{x,k} = \sigma^2_{x,k} \boldsymbol{I}_2$. The noises $z_k(i)$ were zero-mean i.i.d. Gaussian random variables, independent of any other signal with variances $\sigma^2_{z,k}$. The variances $\sigma^2_{x,k}$ and $\sigma^2_{z,k}$ are shown in Fig. 3. We randomly sampled 9 linear equality constraints of the form:

$$\sum_{\ell \in \mathcal{I}_p} d_{p\ell} \boldsymbol{w}_\ell = b_p \cdot \mathbb{1}_{2\times 1}, \tag{89}$$

where the scalars $d_{p\ell}$ and b_p were randomly chosen from the set $\{-3,-2,-1,1,2,3\}$. We used a constant step-size $\mu = 0.025$ for all agents. The results were averaged over 200 Monte-Carlo runs.

First, we considered the case of a perfect model scenario where the observation parameter vector $\boldsymbol{w}^o$ satisfies the equality constraints, i.e., $\boldsymbol{w}^\star = \boldsymbol{w}^o$. In Fig. 4 (left), we compare three algorithms: the non-cooperative LMS algorithm (obtained from (17) by setting $\boldsymbol{\mathcal{P}} = \boldsymbol{I}_M$ and $\boldsymbol{f} = \boldsymbol{0}$), the centralized CLMS algorithm (17) which assumes that the constraints are available at the fusion center, and algorithm (35). For each algorithm, we report the theoretical transient MSD, the theoretical steady-state MSD, and the simulated MSD. We observe that the simulation results match well the actual performance. Furthermore, the network MSD is improved by promoting relationships between tasks. Finally, our algorithm performs well compared to the centralized solution.

Next, we perturbed the optimum parameter vector $\boldsymbol{w}^o$ as follows:

$$\boldsymbol{w}^o_{\text{pert}} = \boldsymbol{w}^o + \boldsymbol{u}^o, \tag{90}$$

so $\boldsymbol{w}^o_{\text{pert}}$ does not satisfy the constraints (89). The entries of $\boldsymbol{u}^o$ were sampled from Gaussian distribution $\mathcal{N}(0,\sigma^2)$. We evaluated algorithm (35) on 6 different setups characterized by $\sigma \in \{0, 0.01, 0.05, 0.1, 0.2, 0.5, 1\}$. The theoretical and simulated learning curves with respect to $\boldsymbol{w}_e^o$ and $\boldsymbol{w}_e^\star$ are reported in Fig. 4. Observe that the performance with respect to $\boldsymbol{w}_e^o$ highly deteriorates when σ increases. However, even for the largest values of $\sigma = 1$, algorithm (35) still performs well with respect to the solution $\boldsymbol{w}_e^\star$ of the optimization problem with constraints.

For comparison purposes, we illustrate in Fig. 5 the theoretical and simulated learning curves with respect to $\boldsymbol{w}^\star$ for the settings where $\sigma = 0.5$ (left) and $\sigma = 1$ (right) of the centralized CLMS algorithm (17), algorithm (35) where the sub-nodes "project-then-combine", and the stochastic version (obtained by replacing the moments by instantaneous approximations) of algorithm (31) where the sub-nodes "combine-then-project" (Appendix C in [46] explains how the performance of this algorithm can be obtained). Observe that both algorithms (35) and the stochastic version of (31) have approximately the same

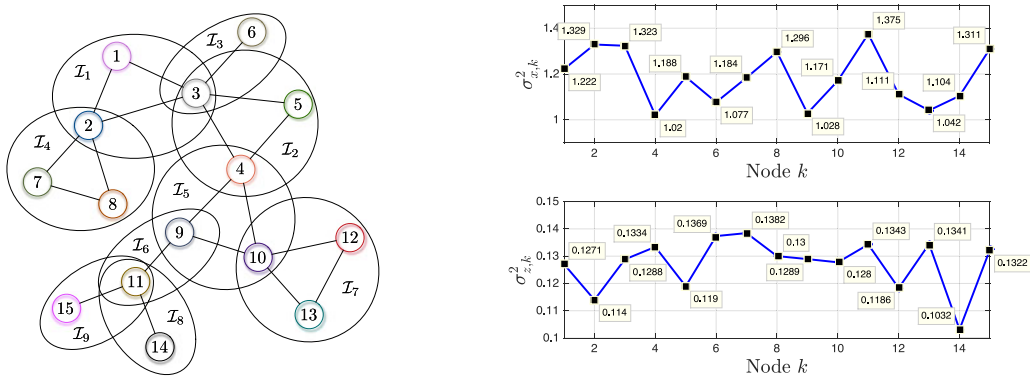


Fig. 3. Experimental setup. (*Left*) Network topology with constraints. (*Right*) Regression and noise variances.

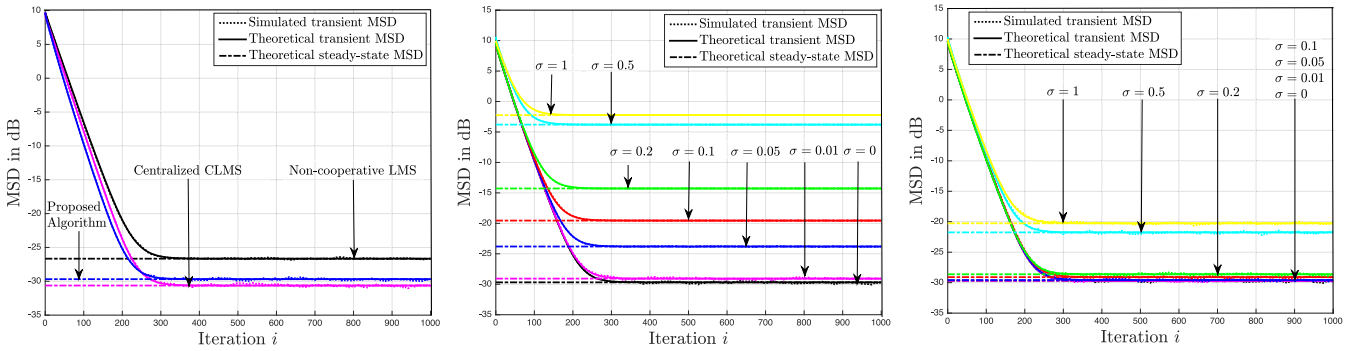


Fig. 4. (*Left*) MSD comparison of the non-cooperative LMS, the centralized CLMS, and our multitask algorithm for the perfect model scenario. Learning curves of algorithm (35) with respect to $\boldsymbol{w}_e^o$ (*middle*) and $\boldsymbol{w}_e^\star$ (*right*) for 6 different values of σ.

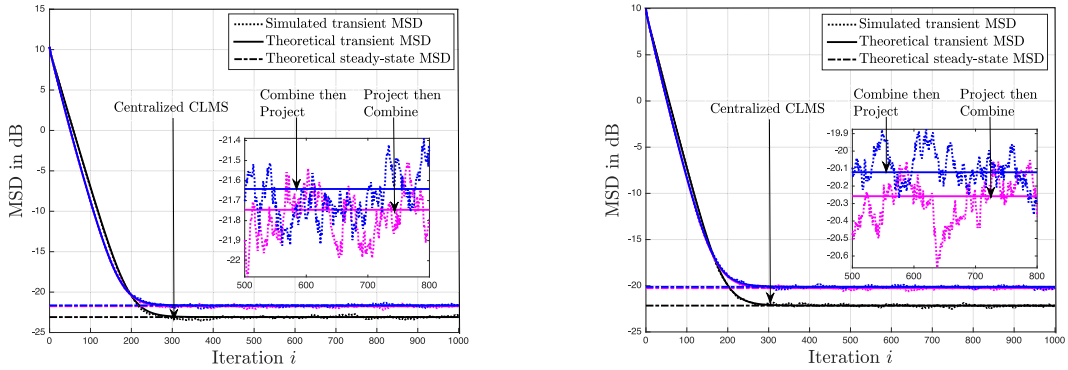


Fig. 5. Network MSD comparison with respect to $\boldsymbol{w}^\star$, $\boldsymbol{w}_e^\star$ of the centralized CLMS (17), algorithm (35), and stochastic version of algorithm (31) for $\sigma = 0.5$ (*left*) and $\sigma = 1$ (*right*).

performance. However, with the settings considered in this section, algorithm (35) is less complex than algorithm (31) as explained in Section III-B. Furthermore, we observe that the larger the vector $\boldsymbol{w}_e^\delta$ is, the larger the performance gap between the centralized solution and the distributed solutions is. This is due to the bias (86) induced in the distributed solution which does not exist in the centralized CLMS algorithm (see [46, Appendix C]). In order to characterize the constraints violation at the sub-nodes for the setting where $\sigma = 0.5$, we evaluate the steady-state quantity $\|\boldsymbol{\mathcal{D}}_e'\boldsymbol{w}_e(\infty) + \boldsymbol{b}'\|^2$ where $\boldsymbol{\mathcal{D}}_e'$ and $\boldsymbol{b}'$ are given by (27) and $\boldsymbol{w}_e(\infty) \triangleq \lim_{i\to\infty} \boldsymbol{w}_e(i)$. When the sub-nodes project first and then combine, we obtain $\|\boldsymbol{\mathcal{D}}_e'\boldsymbol{w}_e(\infty) + \boldsymbol{b}'\|^2 = \|\boldsymbol{\mathcal{D}}_e\boldsymbol{w}_e(\infty) + \boldsymbol{b}\|^2 = 0.0264$. On the contrary, when the sub-nodes combine first and then project, we obtain $\|\boldsymbol{\mathcal{D}}_e'\boldsymbol{w}_e(\infty) + \boldsymbol{b}'\|^2 = \|\boldsymbol{\mathcal{H}}\boldsymbol{w}_e(\infty)\|^2 = 0.0072$. Thus, at the expense of a higher computational complexity, the constraints violation, measured by $\|\boldsymbol{\mathcal{D}}_e'\boldsymbol{w}_e(\infty) + \boldsymbol{b}'\|^2$, is smaller when the projection step is performed after the combination step.

In order to characterize the influence of the step-size μ on the performance of algorithm (35), Fig. 6 (left) reports the theoretical steady-state MSD with respect to $\boldsymbol{w}_e^\star$ for different values of μ. We observe that the network MSD increases 10 dB per

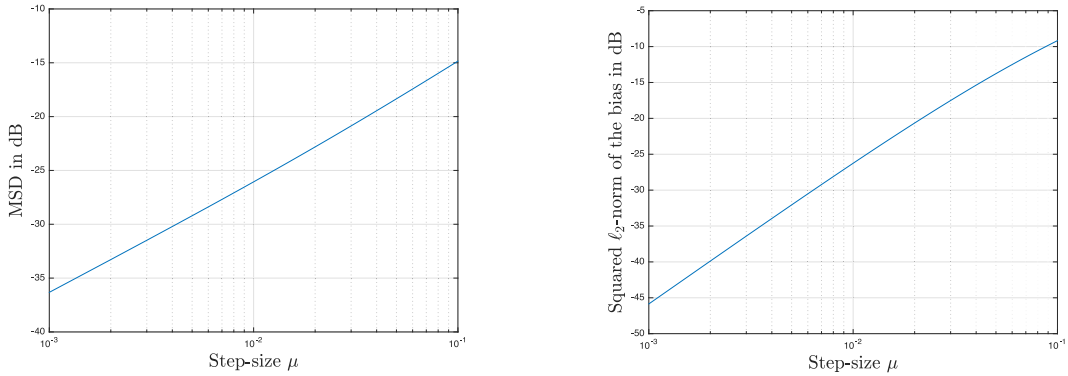


Fig. 6. Influence of the step-size μ on the performance of the algorithm. (*Left*) Network steady-state MSD for different values of μ. (*Right*) Squared norm of the bias, i.e, $\lim_{i\to\infty} \|\mathbb{E}\,\widetilde{\boldsymbol{w}}'(i)\|^2$, for different values of μ.

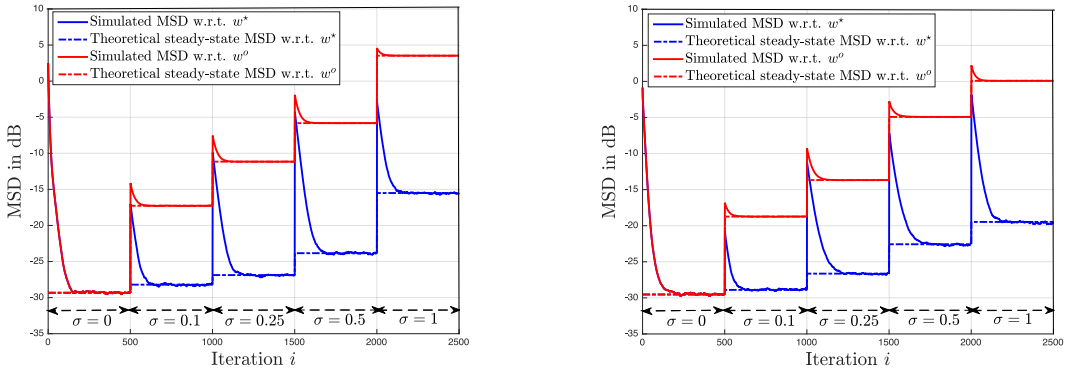


Fig. 7. Tracking ability of the algorithm for two sets of linear equality constraints. (*Left*) $\sigma_D^2 = 0.01$. (*Right*) $\sigma_D^2 = 1$.

decade (when the step-size goes from μ_1 to $10\mu_1$). This means that the steady-state MSD is on the order of μ. Fig. 6 (right) reports the squared norm of the bias (86) for different values of μ. We note that it increases approximately 20 dB per decade. This shows that, as expected, this quantity is on the order of μ^2.

Next, we considered the case of non-diagonal matrices $\boldsymbol{D}_{p\ell}$ defined as:

$$\boldsymbol{D}_{p\ell} = d_{p\ell}\boldsymbol{I}_2 + \boldsymbol{\Delta}_{p\ell} \tag{91}$$

Parameters $d_{p\ell}$ were randomly selected as in (89). The entries of the 2×2 matrix $\boldsymbol{\Delta}_{p\ell}$ were sampled from Gaussian distribution $\mathcal{N}(0, \sigma_D^2)$. As shown in Fig. 7, the variance σ_D^2 was set to 0.01 (left) and 1 (right). To test the tracking ability of algorithm (35), we also perturbed the parameter vector $\boldsymbol{w}^o$ as in (90) by increasing σ^2 every 500 iterations. In both cases, i.e., $\sigma_D^2 = 0.01$ and $\sigma_D^2 = 1$, $\boldsymbol{w}^o$ in (90) was set to satisfy the equality constraints defined by $\boldsymbol{D}_{p\ell}$. We observe that the theoretical models match well the actual performance whatever the constraints are. Furthermore, algorithm (35) adapts its response to drifts in the location of $\boldsymbol{w}^\star$ when $\boldsymbol{w}^o$ changes over time.

B. Optimal Network Flow

As briefly discussed in the Introduction, we shall now consider the minimum-cost flow problem over the network with topology shown in Fig. 1. We are interested in online distributed learning where each node k seeks to estimate the entering and leaving flows f_j from noisy measurement $s_k(i)$ of the external source, by relying only on local computations and communications with its neighbors.

Let M_k be the number of flows to be estimated at node k. We denote by $\boldsymbol{w}_k$ the $M_k \times 1$ parameter vector containing the flows f_j entering and leaving node k, negatively and positively signed, respectively. For instance, for nodes 1 and 2, we have:

$$\boldsymbol{w}_1 \triangleq [f_1\ f_2]^\top \quad \boldsymbol{w}_2 \triangleq [-f_1\ f_3\ f_4\ f_5]^\top \tag{92}$$

From the flow conservation principle, the noisy measurement $s_k(i)$ can be related to $\boldsymbol{w}_k(i)$ as follows:

$$s_k(i) = \mathbf{1}_{M_k\times 1}^\top \boldsymbol{w}_k + z_k(i), \tag{93}$$

with $z_k(i)$ a zero-mean measurement noise, and $\mathbf{1}_{M_k\times 1}$ an $M_k \times 1$ vector of ones. We consider the bi-objective problem consisting of minimizing $\mathbb{E}\,|z_k(i)|^2$ and the cost network flow. We shall assume that the cost for flow through an arc is quadratic in the flow, as in applications such as electrical network monitoring and urban traffic control [35], [37]. We formulate the

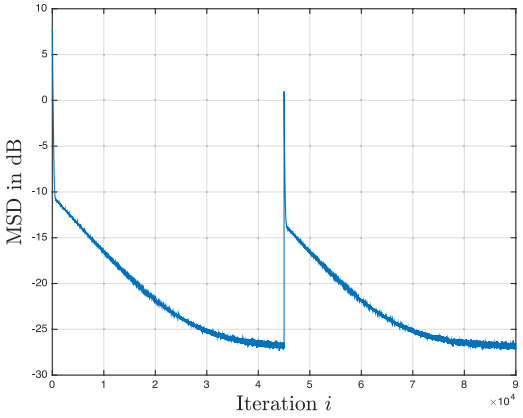


Fig. 8. MSD performance and tracking ability of algorithm (35) for the minimum cost network flow problem.

estimation problem as follows:

$$\underset{\boldsymbol{w}_1,\ldots,\boldsymbol{w}_N}{\text{minimize}} \sum_{k=1}^{N} \Big(\mathbb{E}\,|s_k(i) - \mathbf{1}_{M_k\times 1}^{\top}\boldsymbol{w}_k|^2 + \frac{\eta}{2}\|\boldsymbol{w}_k\|^2 \Big),$$
$$\text{subject to } [\boldsymbol{w}_k]_{f(k,\ell)} + [\boldsymbol{w}_\ell]_{f(\ell,k)} = 0,\ \ell \in \mathcal{N}_k, \text{ for all } k, \tag{94}$$

where $[\boldsymbol{w}_p]_{f(p,q)}$ returns the flow entry in $\boldsymbol{w}_p$ that node p has in common with node q, and η is a tuning parameter to trade off between both objectives.

For each agent k, the external flow s_k and the variance $\sigma_{z,k}^2$ of the Gaussian noise $z_k(i)$ were randomly generated from the uniform distributions $\mathcal{U}(0,3)$ and $\mathcal{U}(0.1,0.14)$, respectively. In order to solve the multitask problem (94) in a fully distributed manner, we applied algorithm (35) by modifying the adaptation step according to:

$$\boldsymbol{\psi}_{k_m}(i+1) = \boldsymbol{w}_{k_m}(i) + \mu\, c_{k_m}\, \mathbf{1}_{M_k\times 1}[s_k(i) - \mathbf{1}_{M_k\times 1}^{\top}\boldsymbol{w}_{k_m}(i)] - \frac{\mu}{2} c_{km}\,\eta\,\boldsymbol{w}_{k_m}(i), \tag{95}$$

and setting $\mu = 0.2$ and $\eta = 0.002$. Note that equation (95) leads to a leaky-LMS version of the proposed algorithm. It is well-known that the leaky-LMS algorithm introduces a bias compared to the LMS, but improves its robustness against the so-called weight-drift problem of the LMS algorithm [43]. In order to test the tracking ability of the algorithm, the external flow s_k at each node k was re-generated from $\mathcal{U}(0,3)$ after 45000 iterations. The MSD learning curve with respect to the solution of problem (94) is reported in Fig. 8. This result was obtained by averaging over 150 Monte-Carlo runs. This figure shows that our strategy was able to solve the minimum-cost flow problem in a fully distributed manner. The estimated flows over the network for both settings considered in the tracking experiment are showed in Fig. 9 (left and middle). Note that the direction of the estimated flow between nodes 3 and 4 is reversed. The true and estimated flows are reported in Fig. 9 (right) for both settings.

C. Numerical Solution of a Two-Dimensional Process

Consider now the problem of estimating a two-dimensional process driven by a partial differential equation (PDE) with a sensor network. To see how our distributed algorithm can be tuned to address this issue, we shall focus on the Poisson's PDE defined by:

$$\frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2} = g(x,y), \qquad (x,y) \in [0,1]^2, \tag{96}$$

with $g : [0,1]^2 \to \mathbb{R}$ an input function, and on a two dimensional network of $(n-2)^2$ sensor nodes and $4(n-1)$ boundary points equally spaced over the unit square $(x,y) \in [0,1]^2$ with $\Delta_x = \Delta_y = \Delta = \frac{1}{n-1}$, as illustrated in Fig. 10 (a).

We introduce the grid point $(x_k, y_\ell) \triangleq (k\Delta, \ell\Delta)$ and the sampled values at this point $f_{k,\ell} \triangleq f(k\Delta,\ell\Delta)$ and $g_{k,\ell} \triangleq g(k\Delta,\ell\Delta)$ with $0 \le k,\ell \le n-1$. We use the central difference approximation for the second derivative [34]:

$$\frac{\partial^2 f(k\Delta,\ell\Delta)}{\partial x^2} \approx \frac{1}{\Delta^2}\Big(f_{k+1,\ell} - 2f_{k,\ell} + f_{k-1,\ell}\Big) \tag{97}$$

$$\frac{\partial^2 f(k\Delta,\ell\Delta)}{\partial y^2} \approx \frac{1}{\Delta^2}\Big(f_{k,\ell+1} - 2f_{k,\ell} + f_{k,\ell-1}\Big) \tag{98}$$

which leads to:

$$\frac{1}{\Delta^2}\Big(-4f_{k,\ell} + f_{k-1,\ell} + f_{k,\ell-1} + f_{k,\ell+1} + f_{k+1,\ell}\Big) = g_{k,\ell}. \tag{99}$$

In this experiment, we shall consider the unknown physical process f and the input function g given by:

$$f(x,y) = (1-x^2)(2y^3 - 3y^2 + 1), \tag{100}$$

$$g(x,y) = -2(2y^3 - 3y^2 + 1) + 6(1-x^2)(2y-1), \tag{101}$$

for $(x,y) \in [0,1]^2$ with boundary conditions $f(0,y) = 2y^3 - 3y^2 + 1$, $f(x,0) = 1 - x^2$, and $f(1,y) = f(x,1) = 0$. These functions are illustrated in Fig. 10 (b), (c).

The objective is to estimate $f(x,y)$ at the interior grid points (x_k, y_ℓ) with $0 < k,\ell < n-1$, given noisy measurements $g_{k\ell}(i) = g_{k\ell} + z_{k\ell}(i)$ of $g(x,y)$ collected by the sensors located at these interior grid points. The noise process $z_{k\ell}(i)$ is assumed to be zero mean, temporally white, and spatially independent. The values of $f(x,y)$ at the boundary points are known a priori as they correspond to boundary conditions. We denote by $f_{k\ell}^o$ the value at (x_k, y_ℓ) of the function $f(x,y)$ that satisfies (96), and by $f_{k\ell}$ the estimated value of $f_{k\ell}^o$. To each node (k,ℓ) we associate an $M_{k\ell}\times 1$ parameter vector $\boldsymbol{w}_{k\ell}$ to estimate, an $M_{k\ell}\times 1$ regression vector $\boldsymbol{x}_{k\ell}$ and a scalar $v_{k\ell}^o$, defined in Table I depending on the node location on the grid.

Given the values of $f(x,y)$ at the boundary points, and according to (99), the linear regression model can be written as follows:

$$g_{k\ell}(i) = \boldsymbol{x}_{k\ell}^{\top}\boldsymbol{w}_{k\ell} + v_{k\ell}^o + z_{k\ell}(i). \tag{102}$$

As can be seen in Table I, equality constraints of the form (1b) need to be imposed on the parameter vectors of neighboring sensor nodes in order to achieve equality between common

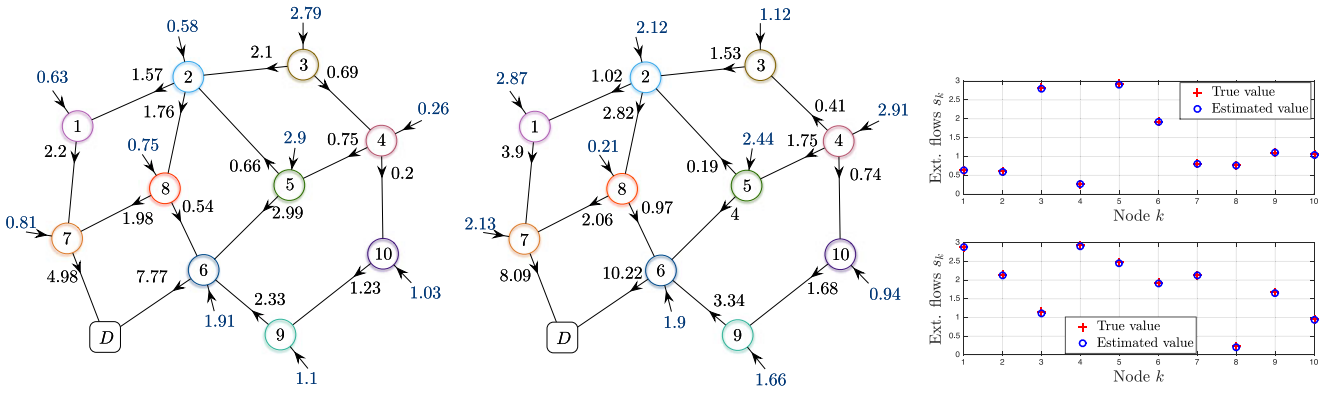
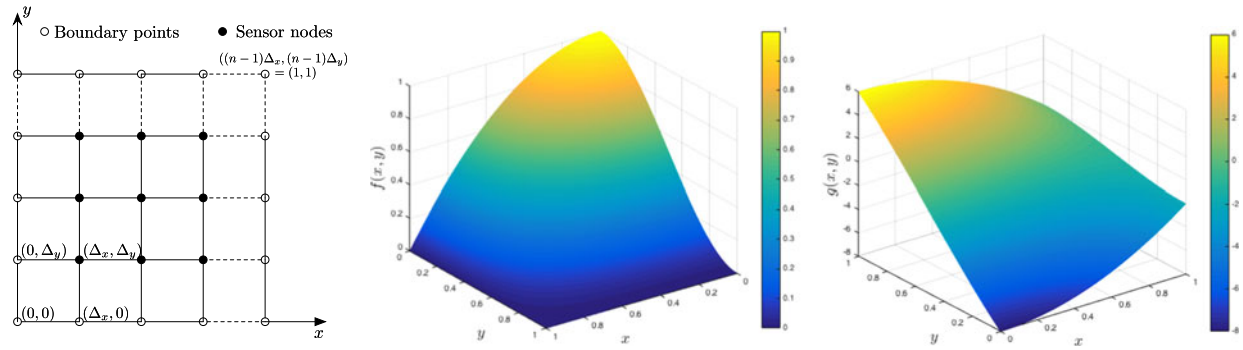


Fig. 9. Estimated network flows. (*Left*) First experiment. (*Middle*) Second experiment. A rounding to 2 decimal places is adopted when visualizing the estimated flows. (*Right*) Comparison of the true and estimated flows s_k (top: first experiment, bottom: second experiment).



(a) An $n \times n$ grid network for the solution of Poisson's equation

(b) $f(x,y) = (1-x^2)(2y^3 - 3y^2 + 1)$

(c) $g(x,y) = -2(2y^3 - 3y^2 + 1) + 6(1 - x^2)(2y - 1)$

Fig. 10. Network topology, function $f(x,y)$ to estimate over the interior grid points, and input function $g(x,y)$.

TABLE I
PARAMETER VECTOR $\boldsymbol{w}_{k\ell}$ (FIRST ROW OF EACH CELL), REGRESSION VECTOR $\Delta^2 \boldsymbol{x}_{k\ell}$ (SECOND ROW OF EACH CELL), AND SCALAR VALUE $\Delta^2 v^o_{k\ell}$ (LAST ROW OF EACH CELL) AT EACH NODE (k,ℓ)

ℓ \ k	1	$2,\ldots,n-3$	$n-2$
1	$[f_{k,\ell}, f_{k,\ell+1}, f_{k+1,\ell}]^\top$ $[-4,1,1]^\top$ $f^o_{1,0} + f^o_{0,1}$	$[f_{k,\ell}, f_{k-1,\ell}, f_{k,\ell+1}, f_{k+1,\ell}]^\top$ $[-4,1,1,1]^\top$ $f^o_{k,0}$	$[f_{k,\ell}, f_{k-1,\ell}, f_{k,\ell+1}]^\top$ $[-4,1,1]^\top$ $f^o_{n-2,0} + f^o_{n-1,1}$
2 $\vdots$ $n-3$	$[f_{k,\ell}, f_{k,\ell-1}, f_{k,\ell+1}, f_{k+1,\ell}]^\top$ $[-4,1,1,1]^\top$ $f^o_{0,\ell}$	$[f_{k,\ell}, f_{k-1,\ell}, f_{k,\ell-1}, f_{k,\ell+1}, f_{k+1,\ell}]^\top$ $[-4,1,1,1,1]^\top$ 0	$[f_{k,\ell}, f_{k-1,\ell}, f_{k,\ell-1}, f_{k,\ell+1}]^\top$ $[-4,1,1,1]^\top$ $f^o_{n-1,\ell}$
$n-2$	$[f_{k,\ell}, f_{k,\ell-1}, f_{k+1,\ell}]^\top$ $[-4,1,1]^\top$ $f^o_{0,n-2} + f^o_{1,n-1}$	$[f_{k,\ell}, f_{k-1,\ell}, f_{k,\ell-1}, f_{k+1,\ell}]^\top$ $[-4,1,1,1]^\top$ $f^o_{k,n-1}$	$[f_{k,\ell}, f_{k-1,\ell}, f_{k,\ell-1}]^\top$ $[-4,1,1]^\top$ $f^o_{n-2,n-1} + f^o_{n-1,n-2}$

entries. For instance, let us consider neighboring nodes (k,ℓ) and $(k+1,\ell)$ with $2 \le k \le n-4$ and $2 \le \ell \le n-3$. Since these nodes are jointly estimating $f_{k,\ell}$ and $f_{k+1,\ell}$, the following equality constraint is required:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \boldsymbol{w}_{k\ell} + \begin{bmatrix} 0 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 \end{bmatrix} \boldsymbol{w}_{(k+1)\ell} = \boldsymbol{0}. \tag{103}$$

Algorithm (35) can be used to address this problem by replacing the adaptation step (35a) by:

$$\boldsymbol{\psi}_{k\ell_m}(i+1) = \boldsymbol{w}_{k\ell_m}(i) + \mu\, c_{k\ell_m}\, \boldsymbol{x}_{k\ell} \big[g_{k\ell}(i) - \boldsymbol{x}^\top_{k\ell} \boldsymbol{w}_{k\ell_m}(i) - v^o_{k\ell} \big], \tag{104}$$

where $\boldsymbol{w}_{k\ell_m}(i)$ denotes the estimate of $\boldsymbol{w}_{k\ell}$ at the m-th subnode of (k,ℓ). The noises $z_{k,\ell}(i)$ were zero-mean i.i.d. Gaussian

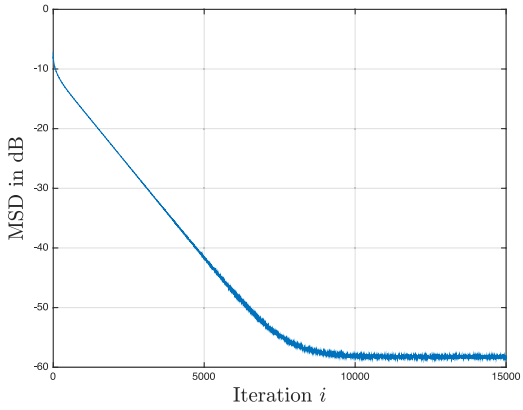


Fig. 11. Network MSD performance for $n = 9$.

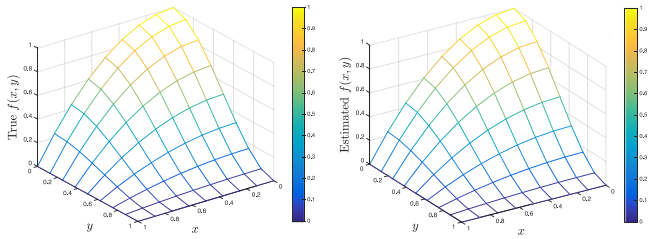


Fig. 12. Poisson process $f(x, y)$ over the network grid. (*Left*) True process. (*Right*) Estimated process.

distributed with variances $\sigma^2_{z,k\ell}$ randomly generated from the uniform distribution $\mathcal{U}(0.1, 0.14)$. We used a constant step-size $\mu = 7 \cdot 10^{-5}$ for all nodes. Fig. 11 shows the network MSD learning curves for $n = 9$. The simulated curves were obtained by averaging over 100 independent runs. Fig. 12 shows the true (left) and estimated (right) process after convergence of our algorithm.

VII. Conclusion and Perspectives

In this work, we proposed a multitask LMS algorithm for solving problems that require the simultaneous estimation of multiple parameter vectors that are related locally via linear constraints. Our primal technique was based on the stochastic gradient projection algorithm with constant step-sizes. The behavior of the algorithm in the mean and mean-square-error sense was studied. We checked with simulations that the agents are able to reach the optimal solution with good precision. In future work, we shall extend our approach to other types of constraints and also consider other constraints distribution over networks.

References

[1] 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.
[2] A. Nedic and D. P. Bertsekas, "Incremental subgradient methods for non-differentiable optimization," *SIAM J. Optim.*, vol. 12, no. 1, pp. 109–138, Jul. 2001.
[3] 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.
[4] S. S. Ram, A. Nedić, and V. V. Veeravalli, "Distributed stochastic subgradient projection algorithms for convex optimization," *J. Optim. Theory Appl.*, vol. 147, no. 3, pp. 516–545, 2010.
[5] K. Srivastava and A. Nedić, "Distributed asynchronous constrained stochastic optimization," *IEEE J. Sel. Topics Signal Process.*, vol. 5, no. 4, pp. 772–790, Aug. 2011.
[6] J. F. C. Mota, J. M. F. Xavier, P. M. Q. Aguiar, and M. Püschel, "D-ADMM: A communication-efficient distributed algorithm for separable optimization," *IEEE Trans. Signal Process.*, vol. 61, no. 10, pp. 2718–2723, May 2013.
[7] S. Lee and A. Nedić, "Distributed random projection algorithm for convex optimization," *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 2, pp. 221–229, Apr. 2013.
[8] 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.
[9] A. H. Sayed, "Adaptation, learning, and optimization over networks," *Found. Trends Mach. Learn.*, vol. 7, no. 4/5, pp. 311–801, 2014.
[10] Z. J. Towfic and A. H. Sayed, "Adaptive penalty-based distributed stochastic convex optimization," *IEEE Trans. Signal Process.*, vol. 62, no. 15, pp. 3924–3938, Aug. 2014.
[11] A. H. Sayed, "Diffusion adaptation over networks," in *Academic Press Library in Signal Processing*, vol. 3. Amsterdam, The Netherlands: Elsevier, 2014, pp. 322–454.
[12] Z. J. Towfic and A. H. Sayed, "Stability and performance limits of adaptive primal-dual networks," *IEEE Trans. Signal Process.*, vol. 63, no. 11, pp. 2888–2903, Jun. 2015.
[13] H. Zhang, W. Shi, A. Mokhtari, A. Ribeiro, and Q. Ling, "Decentralized constrained consensus optimization with primal dual splitting projection," in *Proc. IEEE Global Conf. Signal Inf. Process.*, Washington, DC, USA, Dec. 2016, pp. 565–569.
[14] 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.
[15] A. Bertrand and M. Moonen, "Distributed node-specific LCMV beamforming in wireless sensor networks," *IEEE Trans. Signal Process.*, vol. 60, no. 1, pp. 233–246, Jan. 2012.
[16] C. Eksin and A. Ribeiro, "Distributed network optimization with heuristic rational agents," *IEEE Trans. Signal Process.*, vol. 60, no. 10, pp. 5396–5411, Oct. 2012.
[17] V. Kekatos and G. B. Giannakis, "Distributed robust power system state estimation," *IEEE Trans. Power Syst.*, vol. 28, no. 2, pp. 1617–1626, May 2013.
[18] N. Bogdanović, J. Plata-Chaves, and K. Berberidis, "Distributed incremental-based LMS for node-specific parameter estimation over adaptive networks," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, May 2013, pp. 5425–5429.
[19] N. Bogdanović, J. Plata-Chaves, and K. Berberidis, "Distributed incremental-based LMS for node-specific adaptive parameter estimation," *IEEE Trans. Signal Process.*, vol. 62, no. 20, pp. 5382–5397, Oct. 2014.
[20] J. Plata-Chaves, N. Bogdanovic, and K. Berberidis, "Distributed diffusion-based LMS for node-specific adaptive parameter estimation," *IEEE Trans. Signal Process.*, vol. 63, no. 13, pp. 3448–3460, Jul. 2015.
[21] J. F. C. Mota, J. M. F. Xavier, P. M. Q. Aguiar, and M. Püschel, "Distributed optimization with local domains: Applications in MPC and network flows," *IEEE Trans. Autom. Control*, vol. 60, no. 7, pp. 2004–2009, Jul. 2015.
[22] J. Chen, C. Richard, and A. H. Sayed, "Multitask diffusion adaptation over networks," *IEEE Trans. Signal Process.*, vol. 62, no. 16, pp. 4129–4144, Aug. 2014.
[23] R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed, "Multitask diffusion adaptation over asynchronous networks," *IEEE Trans. Signal Process.*, vol. 64, no. 11, pp. 2835–2850, Jun. 2016.
[24] R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed, "Proximal multitask learning over networks with sparsity-inducing coregularization," *IEEE Trans. Signal Process.*, vol. 64, no. 23, pp. 6329–6344, Dec. 2016.
[25] R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed, "Multitask diffusion LMS with sparsity-based regularization," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, Brisbane, Qld, Australia, Apr. 2015, pp. 3516–3520.
[26] 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.
[27] X. Zhao and A. H. Sayed, "Clustering via diffusion adaptation over networks," in *Proc. Int. Workshop Cogn. Inf. Process.*, Parador de Baiona, Spain, May 2012, pp. 1–6.

[28] J. Chen, C. Richard, and A. H. Sayed, "Diffusion LMS over multitask networks," *IEEE Trans. Signal Process.*, vol. 63, no. 11, pp. 2733–2748, Jun. 2015.
[29] X. Zhao and A. H. Sayed, "Distributed clustering and learning over networks," *IEEE Trans. Signal Process.*, vol. 63, no. 13, pp. 3285–3300, Jul. 2015.
[30] J. Chen, S. K. Ting, C. Richard, and A. H. Sayed, "Group diffusion LMS," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, Shanghai, China, Mar. 2016, pp. 4925–4929.
[31] J. Plata-Chaves, M. H. Bahari, M. Moonen, and A. Bertrand, "Unsupervised diffusion-based LMS for node-specific parameter estimation over wireless sensor networks," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, Shanghai, China, Mar. 2016, pp. 4159–4163.
[32] 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.
[33] J. Chen, Z. J. Towfic, and A. H. Sayed, "Dictionary learning over distributed models," *IEEE Trans. Signal Process.*, vol. 63, no. 4, pp. 1001–1016, Feb. 2015.
[34] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*. Englewood Cliffs, NY, USA: Prentice-Hall, 1989.
[35] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Upper Saddle River, NJ, USA: Prentice-Hall, 1993.
[36] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge, U.K.: Cambridge Univ. Press, 2004.
[37] J. A. Ventura, "Computational development of a Lagrangian dual approach for quadratic networks," *Networks*, vol. 21, no. 4, pp. 469–485, 1991.
[38] O. L. Frost III, "An algorithm for linearly constrained adaptive array processing," *Proc. IEEE*, vol. 60, no. 8, pp. 926–935, Aug. 1972.
[39] D. Estrin, L. Girod, G. Pottie, and M. Srivastava, "Instrumenting the world with wireless sensor networks," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, vol. 4, 2001, pp. 2033–2036.
[40] I. K. Harrane, R. Flamary, and C. Richard, "Toward privacy-preserving diffusion strategies for adaptation and learning over networks," in *Proc. 24th Eur. Signal Process. Conf.*, Budapest, Hungary, Aug. 2016, pp. 1513–1517.
[41] D. P. Bertsekas, *Nonlinear Programming*. Belmont, MA, USA: Athena Scientific, 1999.
[42] R. Arablouei, K. Doğançay, and S. Werner, "On the mean-square performance of the constrained LMS algorithm," *Signal Process.*, vol. 117, pp. 192–197, Dec. 2015.
[43] A. H. Sayed, *Adaptive Filters.* New York, NY, USA: Wiley, 2008.
[44] J. F. C. Mota, J. M. F. Xavier, P. M. Q. Aguiar, and M. Puschel, "Distributed basis pursuit," *IEEE Trans. Signal Process.*, vol. 60, no. 4, pp. 1942–1956, Apr. 2012.
[45] K. Yuan, Q. Ling, and W. Yin, "On the convergence of decentralized gradient descent," *SIAM J. Optim.*, vol. 26, no. 3, pp. 1835–1854, 2016.
[46] R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed, "Diffusion LMS for multitask problems with local linear equality constraints," arXiv: 1610.02943v2, Jun. 2017.
[47] R. H. Koning, H. Neudecker, and T. Wansbeek, "Block Kronecker products and the vecb operator," *Linear Algebra Appl.*, vol. 149, pp. 165–184, Apr. 1991.

Roula Nassif received the Bachelor's degree in electrical engineering from the Lebanese University, Beirut, Lebanon, in 2013, the M.S. degrees in industrial control and intelligent systems for transport from the Lebanese University and from Compiègne University of Technology, Compiègne, France, in 2013, and the Ph.D. degree in 2016 from the University of Côte d'Azur (UCA), Nice, France. She is currently a Researcher and a Teaching Assistant at UCA. Her current research interests include adaptation and learning over networks.

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, Compiègne, France. He is currently a Full Professor with the University of Nice Sophia Antipolis, Nice, France. He was a junior member of the Institut Universitaire de France in 2010–2015.

He is the author of more than 250 papers. His current research interests include statistical signal processing and machine learning. He was the General Co-Chair of the IEEE SSP'11 Workshop that was held in Nice, France. He was the Technical Co-Chair of EUSIPCO'15 that was held in Nice, France, and of the IEEE CAMSAP'15 Workshop that was held in Cancun, Mexico. Since 2015, 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. He is an Associate Editor of *Signal Processing* Elsevier since 2009. He is a member of the IEEE Machine Learning for Signal Processing Technical Committee, and served as a member of the IEEE Signal Processing Theory and Methods Technical Committee in 2009–2014.

André Ferrari (SM'91–M'93) received the Ingénieur degree from École Centrale de Lyon, Lyon, France, in 1988 and the M.Sc. and the Ph.D. degrees from the University of Nice Sophia Antipolis (UNS), Nice, France, in 1989 and 1992, respectively, all in electrical and computer engineering.

He is currently a Professor at UNS. He is a member of the Joseph-Louis Lagrange Laboratory (CNRS, OCA), where his research activity is centered around statistical signal processing and modeling, with a particular interest in applications to astrophysics.

Ali H. Sayed (S'90–M'92–SM'99–F'01) is currently a Professor and Former Chairman of electrical engineering at the University of California, Los Angeles, CA, USA, where he directs the UCLA Adaptive Systems Laboratory. He is an author of more than 480 scholarly publications and six books, his research involves several areas including adaptation and learning, statistical signal processing, distributed processing, network and data sciences, and biologically-inspired designs. He has received several awards including the 2015 Education Award from the IEEE Signal Processing Society, 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. Also, the 2005 Terman Award from the American Society for Engineering Education, the 2003 Kuwait Prize, and the 1996 IEEE Donald G. Fink Prize. He was the Distinguished Lecturer for the IEEE Signal Processing Society in 2005 and as an 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 (2002, 2005, 2012, and 2014). He is a Fellow of the American Association for the Advancement of Science. He is recognized as a Highly Cited Researcher by Thomson Reuters. He is serving as President-Elect of the IEEE Signal Processing Society.