Fiftieth Annual Allerton Conference
Allerton House, UIUC, Illinois, USA
October 1 - 5, 2012

On the Limiting Behavior of Distributed Optimization Strategies

Jianshu Chen and Ali H. Sayed

Abstract— **Motivated by recent developments in the context of adaptation over networks, this work establishes useful results about the limiting global behavior of diffusion and consensus strategies for the solution of distributed optimization problems. It is known that the choice of combination policies has a direct bearing on the convergence and performance of distribued solutions. This article reveals what aspects of the combination policies determine the nature of the Pareto-optimal solution and how close the distributed solution gets to it. The results suggest useful constructive procedures to control the convergence behavior of distributed strategies and to design effective combination procedures.**

I. INTRODUCTION

In multi-agent systems, agents interact with each other to solve a problem of common interest, such as an optimization problem in a distributed manner. Such networks of interacting agents are useful in solving distributed estimation, learning and decision making problems [1]–[18]. They are also useful in modeling biological networks and bio-inspired cognition [19]–[21]. Two useful strategies that can used to guide the interactions of the agents are consensus strategies [3]–[6] and diffusion strategies [8]–[16]. Both classes involve self-learning and social-learning steps. During self-learning, each agent updates its state using its local data. During social learning, each agent aggregates information from its neighbors. A useful feature that results from these localized interactions is that the network ends up exhibiting global patterns of behavior. For example, in biological networks, fish schools move together towards food or away from predators [21]. Likewise, in distributed estimation and learning, each agent is able to attain the performance of centralized solutions by relying solely on local cooperation [4], [10], [14].

In this article, we consider a general class of distributed strategies and study the resulting global behavior by addressing four important questions: (i) where does the distributed algorithm converge to? (ii) when does it converge? (iii) how fast does it converge? and (iv) how close does it converge to the intended point? An interesting conclusion that will follow from our analysis is that the performance of the multi-agent system is largely dependent on the right-eigenvector of the combination matrix corresponding to the eigenvalue at one. This result reveals the manner by which the network topology influences performance in a compact and interesting way.

Most prior studies on distributed optimization and estimation tend to focus on the performance and convergence of the algorithms under diminishing step-size conditions [2]–[6], [10], [17], [18], or on convergence under deterministic conditions on the data [6]. In this paper, we instead examine the global behavior of the distributed strategies from a mean-square-error perspective at *constant* step-sizes. This is because constant step-sizes are necessary for continuous adaptation, learning, and tracking, which in turn enable the algorihtms to perform well even under data that exhibit statistical variations and measurement noise.

Notation. All vectors are column vectors. We use boldface letters to denote random quantities (such as $\boldsymbol{u}_{k,i}$) and regular font to denote their realizations or deterministic variables (such as $u_{k,i}$). We use $\mathrm{diag}\{x_1,\ldots,x_N\}$ to denote a (block) diagonal matrix consisting of diagonal entries (blocks) $x_1,\ldots,x_N$, and use $\mathrm{col}\{x_1,\ldots,x_N\}$ to denote a column vector formed by stacking $x_1,\ldots,x_N$ on top of each other. The notation $x \preceq y$ means each entry of the vector x is less than or equal to the corresponding entry of the vector y. We use the tilde notation to represent the error with respect to the limit point: $\tilde{w} = w^o - w$. The notation $x = \mathrm{vec}(X)$ denotes the vectorization operation that stacks the columns of a matrix X on top of each other to form a vector x, and $X = \mathrm{vec}^{-1}(x)$ is its inverse operation. The operators ∇_w and ∇_{w^T} denote the column and row gradient vectors with respect to w. When ∇_{w^T} is applied to a column vector s, it generates a matrix.

II. PROBLEM FORMULATION

A. Distributed Strategies: Consensus and Diffusion

We consider a network of N agents that are connected according to a certain topology — see Fig. 1. Each agent k implements a distributed algorithm of the following form:

$$\boldsymbol{\phi}_{k,i-1} = \sum_{l=1}^{N} a_{1,lk}\boldsymbol{w}_{l,i-1} \tag{1}$$

$$\boldsymbol{\psi}_{k,i} = \sum_{l=1}^{N} a_{0,lk}\boldsymbol{\phi}_{l,i-1} - \mu_k \hat{\boldsymbol{s}}_{k,i}(\boldsymbol{\phi}_{k,i-1}) \tag{2}$$

$$\boldsymbol{w}_{k,i} = \sum_{l=1}^{N} a_{2,lk}\boldsymbol{\psi}_{l,i} \tag{3}$$

where $\boldsymbol{w}_{k,i} \in \mathbb{R}^M$ is the state of the agent k at time i, usually an estimate for the solution of some optimization problem, $\boldsymbol{\phi}_{k,i-1} \in \mathbb{R}^M$ and $\boldsymbol{\psi}_{k,i} \in \mathbb{R}^M$ are intermediate variables generated at node k before updating to $\boldsymbol{w}_{k,i}$, μ_k is a nonnagative constant step-size parameter used by node k, and $\hat{\boldsymbol{s}}_{k,i}(\cdot)$ is an $M \times 1$ update vector function at node k. In deterministic optimization problems, the update vectors

This work was supported in part by NSF grants CCF-1011918 and CCF-0942936.

The authors are with Department of Electrical Engineering, University of California, Los Angeles, CA 90095. Email: {jshchen, sayed}@ee.ucla.edu.

TABLE I

DIFFERENT CHOICES FOR A_1, A_0 AND A_2 CORRESPOND TO DIFFERENT DISTRIBUTED STRATEGIES.

Distributed Strategeis	A_1	A_0	A_2	$A_1A_0A_2$
Consensus	I	A	I	A
ATC diffusion	I	I	A	A
CTA diffusion	A	I	I	A

$\hat{\boldsymbol{s}}_{k,i}(\cdot)$ can be the gradients or Newton steps associated with the cost functions [6]. On the other hand, in stocastic approximation problems, such as adaptation, learning and estimation problems [3]–[5], [7]–[18], the update vectors are usually computed from realizations of data samples that arrive sequentially at the nodes. In the stochastic setting, the quantities appearing in (1)–(3) become random and we use boldface letters to highlight their stochastic nature. In Example 1 below, we will illustrate the choices for $\hat{\boldsymbol{s}}_{k,i}(w)$ in different contexts.

The combination coefficients $a_{1,lk}$, $a_{0,lk}$ and $a_{2,lk}$ in (1)–(3) are nonnegative weights that each node k assigns to the information arriving from node l; these coefficients are required to satisfy:

$$\sum_{l=1}^{N} a_{1,lk} = 1, \quad \sum_{l=1}^{N} a_{0,lk} = 1, \quad \sum_{l=1}^{N} a_{2,lk} = 1 \tag{4}$$

$$a_{1,lk} \geq 0, \qquad a_{0,lk} \geq 0, \qquad a_{2,lk} \geq 0 \tag{5}$$

$$a_{1,lk} = a_{2,lk} = a_{0,lk} = 0, \qquad \text{if } l \notin \mathcal{N}_k \tag{6}$$

Observe from (6) that the combination coefficients are zero if $l \notin \mathcal{N}_k$, where $\mathcal{N}_k$ denotes the set of neighbors of node k (including node k itself). Therefore, each summation in (1)–(3) is confined within the neighborhood of node k. In algorithm (1)–(3), each node k first combines the states $\{\boldsymbol{w}_{l,i-1}\}$ from its neighbors and updates $\boldsymbol{w}_{k,i-1}$ to the intermediate variable $\boldsymbol{\phi}_{k,i-1}$. Then, the $\{\boldsymbol{\phi}_{l,i-1}\}$ from the neighbors are aggregated and updated to $\boldsymbol{\psi}_{k,i}$ by $\hat{\boldsymbol{s}}_{k,i}(\boldsymbol{\phi}_{k,i-1})$. Finally, the intermediate estimators $\{\boldsymbol{\psi}_{l,i}\}$ from the neighbors are combined to generate the new state $\boldsymbol{w}_{k,i}$ at node k.

The general distributed strategy (1)–(3) can be specialized into various algorithms. We let A_1, A_0 and A_2 denote the $N \times N$ matrices that collect the coefficients $\{a_{1,lk}\}$, $\{a_{0,lk}\}$ and $\{a_{2,lk}\}$. Then, condition (4) is equivalent to

$$A_1^T \mathbb{1} = \mathbb{1}, \quad A_0^T \mathbb{1} = \mathbb{1}, \quad A_2^T \mathbb{1} = \mathbb{1} \tag{7}$$

which means that the matrices $\{A_0, A_1, A_2\}$ are left-stochastic. Different choices for A_1, A_0 and A_2 correspond to different distributed strategies, as summarized in Table I. Specifically, the consensus [3]–[6] and diffusion (ATC and CTA) [8]–[16] algorithms are given by the following iterations:

$$\text{Consensus}: \boldsymbol{w}_{k,i} = \sum_{l \in \mathcal{N}_k} a_{0,lk} \boldsymbol{w}_{l,i-1} - \mu_k \hat{\boldsymbol{s}}_{k,i}(\boldsymbol{w}_{k,i-1}) \tag{8}$$

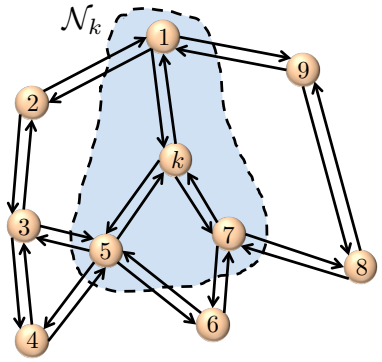

Fig. 1. A network representing a multi-agent system. The set of all agents that can communicate with node k (including node k itself) is denoted as $\mathcal{N}_k$.

$$\text{ATCdiffusion}: \begin{cases} \boldsymbol{\psi}_{k,i} = \boldsymbol{w}_{k,i-1} - \mu_k \hat{\boldsymbol{s}}_{k,i}(\boldsymbol{w}_{k,i-1}) \\ \boldsymbol{w}_{k,i} = \sum_{l \in \mathcal{N}_k} a_{2,lk} \boldsymbol{\psi}_{l,i} \end{cases} \tag{9}$$

$$\text{CTAdiffusion}: \begin{cases} \boldsymbol{\phi}_{k,i-1} = \sum_{l \in \mathcal{N}_k} a_{1,lk} \boldsymbol{w}_{l,i-1} \\ \boldsymbol{w}_{k,i} = \boldsymbol{\phi}_{k,i-1} - \mu_k \hat{\boldsymbol{s}}_{k,i}(\boldsymbol{\phi}_{k,i-1}) \end{cases} \tag{10}$$

Therefore, the convex combination steps appear in different locations in the consensus and diffusion implementations. However, in our analysis, we will proceed with the general form (1)–(3) to study all three schemes within a unifying framework.

We observe that there are two types of learning processes involved in the dynamics of agent k: (i) self-learning in (2) from locally sensed data and (ii) social learning in (1) and (3) from neighbors. All nodes implement the same self- and social learning structure. As a result, the learning dynamics of all nodes in the network are coupled; knowledge exploited from local data at node k will be propagated to its neighbors and from there to their neighbors in a diffusive learning process. It is expected that some global performance pattern will emerge from these localized interactions in the multi-agent system. In this work, we are interested in addressing the following questions:

- Limit point: where does each state $\boldsymbol{w}_{k,i}$ converge to?
- Stability: under which condition does convergence occur?
- Learning rate: how fast does convergence occur?
- Performance: how close is $\boldsymbol{w}_{k,i}$ to the limit point?

The answers to these questions provide useful insights about how to tune the algorithm parameters in order to reach desired performance levels.

Example 1: The distributed algorithm (1)–(3) can be applied to optimize global costs of the following form [15]:

$$J^{\text{glob}}(w) = \sum_{k=1}^{N} J_k(w) \tag{11}$$

or to find Pareto-optimal solutions to multi-objective optimization problems:

$$\min_{w} \{J_1(w), \ldots, J_N(w)\} \tag{12}$$

where $J_k(w)$ is an individual cost associated with each agent k. Optimization problems like (11)–(12) arise in various applications — see [2]–[18]. Depending on the context, the update vector $\hat{\boldsymbol{s}}_{k,i}(\cdot)$ may be chosen in different ways:

- In deterministic optimization problems, the expressions for $\{J_k(w)\}$ are known and the update vector $\hat{\boldsymbol{s}}_{k,i}(\cdot)$ at node k can be chosen as the deterministic gradient (column) vector $\nabla_w J_k(\cdot)$.
- In distributed estimation and learning, the individual cost function at each node k is usually given as the expected value of some loss function $Q_k(\cdot,\cdot)$, i.e., $J_k(w) = \mathbb{E}\{Q_k(w, \boldsymbol{x}_{k,i})\}$ [10], [15], where the expectation is with respect to the randomness in data samples $\{\boldsymbol{x}_{k,i}\}$ collected at node k at time i. The exact expression for $\nabla_w J_k(w)$ is usually unknown since the probability distribution of the data is not known beforehand. In this case, the update vector $\hat{\boldsymbol{s}}_{k,i}(\cdot)$ is chosen as an instantaneous approximation for the true gradient vector, namely, $\widehat{\nabla_w J_k}(\cdot) = \nabla_w Q_k(\cdot, \boldsymbol{x}_{k,i})$. Note that the update vector $\hat{\boldsymbol{s}}_{k,i}(w)$ is now evaluated from the random data sample $\boldsymbol{x}_{k,i}$ collected at agent k at time i. Therefore, it is also random and time dependent.

The update vectors may not necessarily be the gradients of the cost functions or their stochastic approximations. They may also take other forms for different reasons:

- In [4], a certain gain matrix K is multiplied to the left of the stochastic gradient vector $\widehat{\nabla_w J_k}(\cdot)$ to make the estimator asymptotically efficient for a linear observation model.
- In gradient temporal-difference (GTD) learning [22] and its distributed version [9], the cost function $J_k(w)$ is chosen to be the mean-square projected Bellman error (MSPBE), which can be expressed as a product of three expectation terms. As a result, the instantaneous approximation of its gradient is implemented with the help of an auxiliary recursion, and the equivalent update vector becomes non-gradient type. ■

III. MODELING ASSUMPTIONS

In this section, we list the assumptions and definitions that are used in the analysis.

Assumption 1 (Standard network): The $N \times N$ matrix product $A = A_2 A_0 A_1$ is a primitive left-stochastic matrix, i.e., $A^T \mathbb{1} = \mathbb{1}$ and there exists a finite integer j_o such that all entries of A^{j_o} are strictly positive. ■

Let $A = [a_{lk}]$ denote the entries of A. Assumption 1 is readily satisfied if the network is connected and there is at least one $a_{kk} > 0$ for some node k. It then follows from the Peron-Frobenius Theorem [23] that the matrix $A_1 A_0 A_2$ has an eigenvalue one of multiplicity one and all other eigenvalues are strictly less than one. Obviously, $\mathbb{1}^T$ is a left eigenvector of $A_1 A_0 A_2$ corresponding to the eigenvalue at one. Let θ denote the right eigenvector corresponding to the eigenvalue at one and whose entries are normalized to add up to one, i.e., $\mathbb{1}^T \theta = 1$. Then, the Peron-Frobenius Theorem further ensures that all entries of θ are positive. Note that, unlike [3]–[6], [17], [18], we do note require the matrix $A_1 A_0 A_2$ to be doubly-stochastic (in which case θ would be $\mathbb{1}$). Instead, we will study the performance of the algorithms in the context of general left-stochastic matrices $\{A_1, A_0, A_2\}$ and we will examine the influence of θ on both the limit point and performance.

Definition 1 (Step-sizes): Without loss of generality, we express the step-size at each node k as $\mu_k = \mu\beta_k$, where μ is a positive scalar, and $\beta_k \geq 0$. ■

Definition 2 (Useful vectors): Let π and p be the following $N \times 1$ vectors:

$$\pi \triangleq A_2 \theta \tag{13}$$

$$p \triangleq \mathrm{col}\{\pi_1 \beta_1, \ldots, \pi_N \beta_N\} \tag{14}$$

where π_k is the kth entry of the vector π. ■

The vector p will play a critical role in the performance of the distributed strategy (1)–(3). Furthermore, we introduce the following assumptions on the update vectors $\hat{\boldsymbol{s}}_{k,i}(\cdot)$ in (1)–(3).

Assumption 2 (Update vector: Randomness): There exist an $M \times 1$ deterministic vector function $s_k(w)$ such that for all $\boldsymbol{w} \in \mathcal{F}_{i-1}$:

$$\mathbb{E}\,\{\hat{\boldsymbol{s}}_{k,i}(\boldsymbol{w}) | \mathcal{F}_{i-1}\} = s_k(\boldsymbol{w}) \tag{15}$$

for all i, k, where $\mathcal{F}_{i-1}$ denotes the past history of estimators $\{\boldsymbol{w}_{k,j}\}$ for $j \leq i-1$ and all k. Furthermore, there exist $\alpha_k \geq 0$ and $\sigma_{v,k}^2 \geq 0$ such that for all i, k and $\boldsymbol{w} \in \mathcal{F}_{i-1}$:

$$\mathbb{E}\left\{\|\hat{\boldsymbol{s}}_{k,i}(\boldsymbol{w}) - s_k(\boldsymbol{w})\|^2\right\} \leq \alpha_k \cdot \mathbb{E}\|s_k(\boldsymbol{w})\|^2 + \sigma_{v,k}^2 \tag{16}$$

■

The above assumption requires the variance of the random update vector $\hat{\boldsymbol{s}}_{k,i}(\boldsymbol{w})$ to be bounded by the variance of $s_k(\boldsymbol{w})$ due to the randomness in $\boldsymbol{w}$. It is a generalized version of Assumption 2 from [15], [16] and the assumptions from [17], [24], [25], where $\hat{\boldsymbol{s}}_{k,i}(\boldsymbol{w})$ was instead taken as the stochastic gradient:

$$\widehat{\nabla_w J_k}(\boldsymbol{w}) = \nabla_w J_k(\boldsymbol{w}) + \boldsymbol{v}_{k,i}(\boldsymbol{w}) \tag{17}$$

In this case, $s_k(\boldsymbol{w}) = \nabla_w J_k(\boldsymbol{w})$, and (15)–(16) become the following conditions on the gradient noise $\boldsymbol{v}_{k,i}(\boldsymbol{w})$:

$$\mathbb{E}\,\{\hat{\boldsymbol{v}}_{k,i}(\boldsymbol{w}) | \mathcal{F}_{i-1}\} = 0 \tag{18}$$

$$\mathbb{E}\left\{\|\hat{\boldsymbol{v}}_{k,i}(\boldsymbol{w})\|^2\right\} \leq \alpha_k \cdot \mathbb{E}\|\nabla_w J_k(\boldsymbol{w})\|^2 + \sigma_{v,k}^2 \tag{19}$$

In Example 2 of [15], we illustrate why these conditions are necessary in modeling stochastic approximation algorithms. Assumption 2 given by (15)–(16) is more general because we allow the update vector $\hat{\boldsymbol{s}}_{k,i}(\cdot)$ to be of other forms (e.g., [4], [9]).

Assumption 3 (Update vector: Lipschitz): There exist a nonnegative λ_U such that for all $x, y \in \mathbb{R}^M$ and all k:

$$\|s_k(x) - s_k(y)\| \leq \lambda_U \cdot \|x - y\| \tag{20}$$

where the subscript U in λ_U means upper bound. ■

Assumption 4 (Update vector: Strong monotonicity): Let p_k denote the kth entry of the vector p defined in (14). There exists $\lambda_L > 0$ such that for all $x, y \in \mathbb{R}^M$:

$$(x-y)^T \cdot \sum_{k=1}^{N} p_k \Big[s_k(x) - s_k(y) \Big] \geq \lambda_L \|x - y\|^2 \tag{21}$$

where the subscript L in λ_L means lower bound. ■

In the context of distributed optimization with stochastic gradient, i.e., $s_k(w) = \nabla_w J_k(w)$ and $\hat{\boldsymbol{s}}_{k,i}(w) = \widehat{\nabla_w J_k}(w) = \nabla_w J_k(w) + \boldsymbol{v}_{k,i}(w)$, the above Assumptions 3–4 are equivalent to requiring

$$\nabla_w^2 J_k(w) \leq \lambda_U I_M \tag{22}$$

$$\sum_{k=1}^{N} p_k \nabla_w^2 J_k(w) \geq \lambda_L I_M > 0 \tag{23}$$

On the other hand, in [15], [16], we assumed $\nabla_w^2 J_k(w) \geq \lambda_{\min,k} I_M$ (see Assumption 1 in [15], [16] with the combination matrix $C = I$). This was meant to require each individual cost function $J_k(w)$ to be strongly convex, which is a stronger condition than (23). Condition (23) is equivalent to requiring a certain weighted sum of the individual cost functions $\{J_k(w)\}$ to be strongly convex:

$$J^{\text{glob}}(w) = \sum_{k=1}^{N} p_k J_k(w) \tag{24}$$

Such a relaxation of the assumptions introduces some challenges into the analysis, as we explain in Sec. IV-B.

IV. LIMITING BEHAVIOR OF DISTRIBUTED STRATEGIES

In this section, we study the global behavior that emerges from the local interactions in the distributed strategy (1)–(3). First, we establish the existence of a unique limit point w^o under the assumptions of Sec. III. Second, we show that under certain conditions on the *constant* step-sizes, the state vector $\boldsymbol{w}_{k,i}$ at each node k converges to the same limit point w^o with certain steady-state MSE performance. We also evaluate the convergence rate and steady-state mean-square-error at small step-sizes, and show that each agent achieves approximately the same performance.

A. Limit Point

To study the limiting global behavior of (1)–(3), the first step is to identify the potential limit point w^o of the algorithm if it converges. We also need to show the existence and uniqueness of such a vector.

Theorem 1 (Limit point): Given Assumptions 3–4, there exists a unique $M \times 1$ vector w^o such that

$$\sum_{k=1}^{N} p_k s_k(w^o) = 0 \tag{25}$$

where p_k is the kth entry of the vector p defined in (14).

Proof: Omitted for brevity. ■

Example 2: In the special case that $s_k(w) = \nabla_w J_k(w)$, where $J_k(w)$ is the individual cost function asssociated with agent k, the above equation (25) becomes:

$$\sum_{k=1}^{N} p_k \nabla_w J_k(w^o) = 0 \tag{26}$$

which means the vector w^o is the minimizer of the following global cost function:

$$J^{\text{glob}}(w) = \sum_{k=1}^{N} p_k J_k(w) \tag{27}$$

We will see that minimizing the above $J^{\text{glob}}(w)$ is equivalent to finding a Pareto-optimal solution to the multi-objective optimization problem in (12) — see Sec. V further ahead. ■

B. Error Recursion

We are going to show next that the vector w^o defined above is actually the limit point of the distributed algorithms (1)–(3); the state vector $\boldsymbol{w}_{k,i}$ at each node k converges to w^o at a certain rate and with a certain steady-state MSE. First, note from (1)–(3) that the recursion at each node k is coupled with the recursions at its neighbors. Therefore, it is necessary to study the evolution of the states $\{\boldsymbol{w}_{k,i}\}$ over the entire network. Introduce the following global quantities

$$\boldsymbol{\phi}_i \triangleq \text{col}\{\boldsymbol{\phi}_{1,i}, \ldots, \boldsymbol{\phi}_{N,i}\} \tag{28}$$

$$\boldsymbol{\psi}_i \triangleq \text{col}\{\boldsymbol{\psi}_{1,i}, \ldots, \boldsymbol{\psi}_{N,i}\} \tag{29}$$

$$\boldsymbol{w}_i \triangleq \text{col}\{\boldsymbol{w}_{1,i}, \ldots, \boldsymbol{w}_{N,i}\} \tag{30}$$

$$\Omega_0 \triangleq \text{diag}\{\beta_1, \ldots, \beta_N\} \tag{31}$$

$$\mathcal{M}_0 \triangleq \Omega_0 \otimes I_M \tag{32}$$

$$\mathcal{A}_1 \triangleq A_1 \otimes I_M \tag{33}$$

$$\mathcal{A}_0 \triangleq A_0 \otimes I_M \tag{34}$$

$$\mathcal{A}_2 \triangleq A_2 \otimes I_M \tag{35}$$

$$\hat{\boldsymbol{s}}_i(\boldsymbol{\phi}_{i-1}) \triangleq \text{col}\{\hat{\boldsymbol{s}}_{1,i}(\boldsymbol{\phi}_{1,i-1}), \ldots, \hat{\boldsymbol{s}}_{N,i}(\boldsymbol{\phi}_{N,i-1})\} \tag{36}$$

Then, recursions (1)–(3) can be expressed as

$$\boldsymbol{\phi}_{i-1} = \mathcal{A}_1^T \boldsymbol{w}_{i-1} \tag{37}$$

$$\boldsymbol{\psi}_i = \mathcal{A}_0^T \boldsymbol{\phi}_{i-1} - \mu \mathcal{M}_0 \hat{\boldsymbol{s}}_i(\boldsymbol{\phi}_{i-1}) \tag{38}$$

$$\boldsymbol{w}_i = \mathcal{A}_2^T \boldsymbol{\psi}_i \tag{39}$$

which leads to

$$\boldsymbol{w}_i = \mathcal{A}_2^T \mathcal{A}_0^T \mathcal{A}_1^T \boldsymbol{w}_{i-1} - \mu \mathcal{A}_2^T \mathcal{M}_0 \hat{\boldsymbol{s}}_i(\mathcal{A}_1^T \boldsymbol{w}_{i-1}) \tag{40}$$

To measure how close each $\boldsymbol{w}_{k,i}$ is to w^o, introduce a global error vector of the following form

$$\begin{aligned}\tilde{\boldsymbol{w}}_i &\triangleq \mathbb{1} \otimes w^o - \boldsymbol{w}_i \\ &= \mathrm{col}\{\tilde{\boldsymbol{w}}_{1,i}, \ldots, \tilde{\boldsymbol{w}}_{N,i}\}\end{aligned} \tag{41}$$

We analyze the mean-square performance of the distributed algoirthm by studying the evolution of the error covariance matrix

$$\mathcal{P}_i \triangleq \mathbb{E}\left\{\tilde{\boldsymbol{w}}_i \tilde{\boldsymbol{w}}_i^T\right\} \tag{42}$$

from which we can evaluate any weighted mean-square-error by using the following relation:

$$\mathbb{E}\|\tilde{\boldsymbol{w}}_i\|_\Sigma^2 = \mathbb{E}\left\{\tilde{\boldsymbol{w}}_i^T \Sigma \tilde{\boldsymbol{w}}_i\right\} = \mathrm{Tr}(\mathcal{P}_i \Sigma) \tag{43}$$

where Σ is an arbitrary positive semidefinite weighting matrix. Specifically, we are going to study how $\mathcal{P}_i$ evolves over time, under what condition it converges, how fast it converges, and the expression of its limit value when it converges:

$$\mathcal{P}_\infty \triangleq \lim_{i \to \infty} \mathcal{P}_i \tag{44}$$

However, this is a challenging task because of the coupling and nonlinear natures of the recursions (1)–(3). Furthermore, since we further relaxed the assumptions (especially Assumption 4) relative to previous results [15], [16], the analysis becomes more demanding. Nevertheless, we are still able to analyze the mean-square-error performance by introducing a useful transformation. Specifically, we analyze the performance in two steps: (i) we first study the evolution of the error covariance in a transformed domain and establish bounds on the mean-square-error, and (ii) we then evaluate the convergence rate and $\mathcal{P}_\infty$ for small step-sizes.

First, we introduce the transform that we are going to use in the analysis. To begin with, we introduce the Jordan canonical decomposition [26]:

$$A_2^T A_0^T A_1^T = U D U^{-1} \tag{45}$$

By Assumption 1, $A_2^T A_0^T A_1^T$ is a primitive right-stochastic matrix. Therefore, the matrices U, D and U^{-1} can be expressed in the following block forms:

$$U = \begin{bmatrix}\mathbb{1} & U_L\end{bmatrix},\ D = \begin{bmatrix}1 & 0 \\ 0 & D_{N-1}\end{bmatrix},\ U^{-1} = \begin{bmatrix}\theta^T \\ U_R^T\end{bmatrix} \tag{46}$$

where D_{N-1} is the $(N-1)\times(N-1)$ matrix with all Jordan blocks that have eigenvalue strictly less than one. It follows that

$$\mathcal{A}_2^T \mathcal{A}_0^T \mathcal{A}_1^T = \mathcal{U} \mathcal{D} \mathcal{U}^{-1} \tag{47}$$

where $\mathcal{U} \triangleq U \otimes I_M$, $\mathcal{D} \triangleq D \otimes I_M$, and $\mathcal{U}^{-1} \triangleq U^{-1} \otimes I_M$. Let

$$\boldsymbol{w}_i' \triangleq \mathcal{U}^{-1} \boldsymbol{w}_i \tag{48}$$

We can express $\boldsymbol{w}_i'$ as a block vector that is formed by stacking N vectors of size $M \times 1$ on top of each other:

$$\boldsymbol{w}_i' = \mathrm{col}\{\boldsymbol{w}_{c,i}, \boldsymbol{e}_{1,i}, \ldots, \boldsymbol{e}_{N-1,i}\} \tag{49}$$

Then, the transform relation (48) implies that the vector $\boldsymbol{w}_{c,i}$ is a weighted average of the state vectors $\{\boldsymbol{w}_{k,i}\}$ at all agents, or equivalently, $\boldsymbol{w}_{c,i}$ is the "centroid" of $\{\boldsymbol{w}_{k,i}\}$:

$$\boldsymbol{w}_{c,i} = \sum_{k=1}^N \theta_k \boldsymbol{w}_{k,i} \tag{50}$$

Furthermore, from (48), we obtain

$$\boldsymbol{w}_i = \mathcal{U} \boldsymbol{w}_i' = \mathbb{1} \otimes \boldsymbol{w}_{c,i} + (U_L \otimes I_M) \begin{bmatrix}\boldsymbol{e}_{1,i} \\ \vdots \\ \boldsymbol{e}_{N-1,i}\end{bmatrix} \tag{51}$$

We observe that the transform (48) actually decomposes the state vector $\boldsymbol{w}_i$ into a centroid component $\boldsymbol{w}_{c,i}$ and perturbation terms $\{\boldsymbol{e}_{n,i}\}$; the state vector $\boldsymbol{w}_{k,i}$ at each agent k can be expressed as $\boldsymbol{w}_{c,i}$ plus a perturbation term:

$$\boldsymbol{w}_{k,i} = \boldsymbol{w}_{c,i} + \sum_{n=1}^{N-1} [U_L]_{kn} \cdot \boldsymbol{e}_{n,i} \tag{52}$$

where $[U_L]_{kn}$ denotes the (k,n)-th entry of the matrix U_L. This observation allows us to study the evolution of the error vector $\tilde{\boldsymbol{w}}_{k,i}$ by studying the evolution of $\tilde{\boldsymbol{w}}_{c,i}$ and $\{\boldsymbol{e}_{1,i}, \ldots, \boldsymbol{e}_{N-1,i}\}$ because of the following relation:

$$\tilde{\boldsymbol{w}}_{k,i} = \tilde{\boldsymbol{w}}_{c,i} - \sum_{n=1}^{N-1} [U_L]_{kn} \cdot \boldsymbol{e}_{n,i} \tag{53}$$

Next, we use this observation to (i) establish a bound on the following mean-square-error vector:

$$\mathcal{W}_i' \triangleq \mathrm{col}\left\{\mathbb{E}\|\tilde{\boldsymbol{w}}_{c,i}\|^2, \mathbb{E}\|\boldsymbol{e}_{1,i}\|^2, \ldots, \mathbb{E}\|\boldsymbol{e}_{N-1,i}\|^2\right\} \tag{54}$$

and to (ii) evaluate the evolution of the error covariance matrix $\mathcal{P}_i$ defined in (42) at small step-sizes.

C. Mean-Square-Error Performance

We establish in the following theorem a bound on $\mathcal{W}_i'$ and stability conditions on the step-sizes. The theorem shows that when the step-size parameter μ is small enough, the state vector $\boldsymbol{w}_{k,i}$ at each node k would converge to the same limit point w^o defined in (25) with a certain steady-state mean-square-error. It also provides bounds on how fast and how close it converges to the limit point w^o.

Theorem 2 (Bound on mean-square-error): The following non-asymptotic bound on $\mathcal{W}_i'$ holds for all $i \geq 0$:

$$\mathcal{W}_i' \preceq \Gamma^i \left[\mathcal{W}_0' - \mathcal{W}_\infty^{\mathrm{ub}'}\right] + \mathcal{W}_\infty^{\mathrm{ub}'} \tag{55}$$

if the matrix Γ is stable, where

$$\mathcal{W}_{\infty}^{\mathrm{ub}'} \triangleq \mu^2\sigma^2(I_N-\Gamma)^{-1}\mathbb{1} \tag{56}$$

$$\Gamma \triangleq \begin{bmatrix} 1-\mu\lambda_L+\frac{\mu^2}{2}\|p\|_1^2\lambda_U^2 & \mu h_c\mathbb{1}^T \\ 0 & \Gamma_0 \end{bmatrix} + \mu^2\alpha\mathbb{1}\mathbb{1}^T \tag{57}$$

$$\Gamma_0 \triangleq \begin{bmatrix} d_2 & \frac{4}{1-d_2} & & \\ & \ddots & \ddots & \\ & & \ddots & \frac{4}{1-d_2} \\ & & & d_2 \end{bmatrix} \tag{58}$$

Moreover, d_2 is the magnitude of the second-largest eigenvalue of the matrix $A_1A_0A_2$, and the parameters σ^2, h_c and α are constant numbers that are determined by A_1, A_2, U, Ω_0, α_k, λ_U, $\sigma_{v,k}^2$, and p (we omit their definitions for brevity). Furthermore, a sufficient condition that guarantees the stability of the matrix Γ is that

$$0<\mu<\min\left\{\frac{\lambda_L}{\frac{1}{2}\|p\|_1^2\lambda_U^2+\alpha\tau},\ \frac{\frac{1-d_2}{2}}{h_c\frac{(1-d_2)^2}{8}+\lambda_L}\right\} \tag{59}$$

where

$$\tau \triangleq \left(\frac{(1-d_2)^2}{8}\right)^{N-1}\left[1-\frac{(1-d_2)^2}{8}\right] \tag{60}$$

Under condition (59), the spectral radius of the matrix Γ is upper bounded by

$$\rho(\Gamma) \le 1-\mu\lambda_L+\mu^2\left(\frac{1}{2}\|p\|_1^2\lambda_U^2+\alpha\tau\right) \tag{61}$$

Proof: Omitted for brevity. ■

Note from (55) that, as $i\to\infty$, each entry of $\mathcal{W}_i'$ is upper bounded by the corresponding entry of the vector $\mathcal{W}_{\infty}^{\mathrm{ub}'}$:

$$\limsup_{i\to\infty}\mathcal{W}_i' \preceq \mathcal{W}_{\infty}^{\mathrm{ub}'} \tag{62}$$

An important implication of the above theorem is that each entry of $\mathcal{W}_i'$ (i.e., $\mathbb{E}\|\tilde{\boldsymbol{w}}_{c,i}\|^2, \mathbb{E}\|\boldsymbol{e}_{1,i}\|^2, \ldots, \mathbb{E}\|\boldsymbol{e}_{N-1,i}\|^2$) can be made arbitrarily small for a sufficiently small step-size μ. To see this, we evaluate the expression of $\mathcal{W}_{\infty}^{\mathrm{ub}'}$ by substituting (57) into (56) to obtain

$$\begin{aligned}\mathcal{W}_{\infty}^{\mathrm{ub}'} &= \frac{\sigma^2}{1-\alpha\left[\frac{\mu(1+\mu h_c\mathbb{1}^T(I-\Gamma_0)^{-1}\mathbb{1})}{\lambda_L-\mu\frac{1}{2}\|p\|_1^2\lambda_U^2}+\mu^2\mathbb{1}^T(I-\Gamma_0)^{-1}\mathbb{1}\right]} \\ &\quad\times\begin{bmatrix}\mu\frac{1+\mu h_c\mathbb{1}^T(I-\Gamma_0)^{-1}\mathbb{1}}{\lambda_L-\mu\frac{1}{2}\|p\|_1^2\lambda_U^2} \\ \mu^2(I-\Gamma_0)^{-1}\mathbb{1}\end{bmatrix} \\ &\approx \begin{bmatrix}\mu\frac{\sigma^2}{\lambda_L} \\ \mu^2(I-\Gamma_0)^{-1}\mathbb{1}\end{bmatrix}\end{aligned} \tag{63}$$

Expression (63) implies that

$$\limsup_{i\to\infty}\mathbb{E}\|\tilde{\boldsymbol{w}}_{c,i}\|^2 \le O(\mu) \tag{64}$$

$$\limsup_{i\to\infty}\mathbb{E}\|\boldsymbol{e}_{n,i}\|^2 \le O(\mu^2) \tag{65}$$

We see that the mean-square-error between the centroid point and the limit point is on the order of $O(\mu)$ at steady-state, while the mean-square values of the perturbation terms are on the order of $O(\mu^2)$. Therefore, from (53) and using Jensen's inequality, we obtain[1]

$$\begin{aligned}\mathbb{E}\|\tilde{\boldsymbol{w}}_{k,i}\|^2 &= \mathbb{E}\left\|\tilde{\boldsymbol{w}}_{c,i}-\sum_{n=1}^{N-1}[U_L]_{kn}\cdot\boldsymbol{e}_{n,i}\right\|^2 \\ &\le 2\mathbb{E}\|\tilde{\boldsymbol{w}}_{c,i}\|^2+2\mathbb{E}\left\|\sum_{n=1}^{N-1}[U_L]_{kn}\cdot\boldsymbol{e}_{n,i}\right\|^2 \\ &\le 2\mathbb{E}\|\tilde{\boldsymbol{w}}_{c,i}\|^2+C\cdot\sum_{n=1}^{N}\mathbb{E}\|\boldsymbol{e}_{n,i}\|^2\end{aligned} \tag{66}$$

where C is some constant number. This means that, as $i\to\infty$, we have

$$\limsup_{i\to\infty}\mathbb{E}\|\tilde{\boldsymbol{w}}_{k,i}\|^2 \le O(\mu)+O(\mu^2)=O(\mu) \tag{67}$$

Therefore, the mean-square-error between $\boldsymbol{w}_{k,i}$ at each node k and the limit point w^o can be made arbitrarily small for sufficiently small step-size μ.

Theorems 1 and 2 together establish the fact that, for sufficiently small constant step-sizes, the state vector $\boldsymbol{w}_{k,i}$ generated by the distributed algorithm (1)–(3) at each node k converges to the same limit point w^o defined in (25). And the steady-state mean-square-error can be made arbitrarily small for small step-sizes. Inequalities (61) and (63) also provide estimates for the convergence rate and steady-state mean-square-error. However, for small step-sizes, we are able to evaluate the approximate values (rather than bounds) of the convergence rate and steady-state error covariance matrix $\mathcal{P}_{\infty}$. The main results are summarized in the following theorem.

Theorem 3 (Convergence rate and error covariance): For sufficiently small step-sizes and after long enough time ($i\ge i_0$ for some large enough i_0), the error covariance matrix $\mathcal{P}_i$ evolves approximately according to the following relation:

$$\mathrm{vec}(\mathcal{P}_i) = (\mathcal{B}\otimes\mathcal{B})^{i-i_0}\left[\mathrm{vec}(\mathcal{P}_{i_0})-\mathrm{vec}(\mathcal{P}_{\infty})\right]+\mathrm{vec}(\mathcal{P}_{\infty}) \tag{68}$$

where

$$\mathcal{B} \triangleq \mathcal{A}_2^T[\mathcal{A}_0^T-\mu\mathcal{M}_0\mathcal{R}_{\infty}]\mathcal{A}_1^T \tag{69}$$

$$\mathcal{R}_{\infty} \triangleq \mathrm{diag}\left\{\nabla_{w^T}s_1(w^o),\ldots,\nabla_{w^T}s_N(w^o)\right\} \tag{70}$$

$$\mathrm{vec}(\mathcal{P}_{\infty}) \triangleq (I-\mathcal{B}\otimes\mathcal{B})^{-1}\mathrm{vec}(\mathcal{Y}) \tag{71}$$

$$\mathcal{Y} \triangleq \mathcal{A}_2^T\mathcal{M}_0\mathcal{R}_s\mathcal{M}_0\mathcal{A}_2 \tag{72}$$

$$\begin{aligned}\mathcal{R}_s \triangleq\ &\mathbb{E}\big\{[\hat{\boldsymbol{s}}_i(\boldsymbol{\phi}_{i-1})-s(\boldsymbol{\phi}_{i-1})] \\ &\times[\hat{\boldsymbol{s}}_i(\boldsymbol{\phi}_{i-1})-s(\boldsymbol{\phi}_{i-1})]^T\big\}\big|_{\boldsymbol{\phi}_{i-1}=\mathbb{1}\otimes w^o}\end{aligned} \tag{73}$$

$$\begin{aligned}s(\boldsymbol{\phi}_{i-1}) &\triangleq \mathbb{E}\left\{\hat{\boldsymbol{s}}_i(\boldsymbol{\phi}_{i-1})\middle|\mathcal{F}_{i-1}\right\} \\ &= \mathrm{col}\left\{s_1(\boldsymbol{\phi}_{i-1}),\ldots s_N(\boldsymbol{\phi}_{i-1})\right\}\end{aligned} \tag{74}$$

When the step-size parameter μ is small, the convergence rate and the steady-state weighted mean-square-error at each

[1] In the second inequality, we used the following relation: $\|x+y\|^2 \le \|x\|^2+2\|x\|\cdot\|y\|+\|y\|^2 \le 2\|x\|^2+2\|y\|^2$.

node k is given by

$$r \triangleq [\rho(\mathcal{B})]^2 \approx 1 - 2\mu\lambda_{\min}(R_c) \tag{75}$$

$$\begin{aligned}\mathrm{MSE}_k^{\Sigma} &\triangleq \lim_{i\to\infty} \mathbb{E}\|\tilde{\boldsymbol{w}}_{k,i}\|_{\Sigma}^2 \\ &= \mu \cdot \mathrm{Tr}\left\{(p^T \otimes I_M)\mathcal{R}_s(p \otimes I_M)Z\right\} + O(\mu^2)\end{aligned} \tag{76}$$

where the $M \times M$ matrix R_c is defined by

$$R_c = \sum_{k=1}^{N} p_k \nabla_{w^T} s_k(w^o) \tag{77}$$

and the $M \times M$ matrix Z is the solution to the following Lyapunov equation:

$$R_c^T Z + Z R_c = \Sigma \tag{78}$$

which is given by

$$Z = \mathrm{vec}^{-1}\left\{(I_M \otimes R_c^T + R_c^T \otimes I_M)^{-1}\mathrm{vec}(\Sigma)\right\} \tag{79}$$

Proof: Omitted for brevity. ■

D. Global Behavior: Centralized Performance

We observe from (76) that the weighted mean-square-error at each node k is the same across all agents in the entire network for small step-sizes. This is an important "equalization" effect observed in diffusion adaptation strategies [8]–[16]. The next theorem will further reveal that such performance is close to the centralized strategy that collects all the data from the agents and processes it using the following recursion:

$$\boldsymbol{w}_{\mathrm{cent},i} = \boldsymbol{w}_{\mathrm{cent},i-1} - \mu \sum_{k=1}^{N} p_k \hat{\boldsymbol{s}}_{k,i}(\boldsymbol{w}_{\mathrm{cent},i-1}) \tag{80}$$

Theorem 4 (Centralized Performance): Suppose Assumptions 2–4 hold and the step-size μ satisfies the following condition

$$0 < \mu < \frac{2\lambda_L}{\|p\|_1^2 \lambda_U^2 (1 + 2\alpha)} \tag{81}$$

where $\alpha \triangleq \max_{1\le k\le N} \alpha_k$. Then the centralized strategy (80) converges to the same limit point w^o defined in (25). Furthermore, for sufficiently small step-sizes, the convergence rate and steady-state mean-square-error are the same as (75)–(76).

Proof: Omitted for brevity. ■

Theorems 1–4 answer the four questions we posed in Sec. II-A about distributed processing. They show that, via local interactions and learning, the distributed strategies (1)–(3) lead to the same global behavior as that of a centralized strategy. Specifically, for sufficiently small step-sizes, the state vector $\boldsymbol{w}_{k,i}$ at each node k converges to the same limit point w^o as the centralized strategy (80) with the same convergence rate and steady-state mean-square-error (up to the first order term of μ) as (80). We may note that it was shown in [27] that, in distributed LMS estimation, the diffusion strategies (9)–(10) outperform the cosensus strategy (8) in the high-order term $O(\mu^2)$.

V. APPLICATION TO DISTRIBUTED OPTIMIZATION

In this section, we illustrate the results of Sec. IV in the context of distributed Pareto optimization. Each agent k is associated with an individual convex cost function $J_k(w)$, $k = 1, \ldots, N$. We would like to find a Pareto-optimal solution to the following multi-objective optimization problem:

$$\min_{w} \{J_1(w), \ldots, J_N(w)\} \tag{82}$$

We choose the update vectors to be stochastic gradients:

$$s_k(w) = \nabla_w J_k(w), \qquad \hat{\boldsymbol{s}}_{k,i}(w) = \widehat{\nabla_w J_k}(w) \tag{83}$$

Without loss of generality, we assume $\mu_k = \mu$ (i.e., $\beta_k = 1$). Then, by definition (14), the vector $p = \pi = A_2\theta$, where θ is the right eigenvector of the matrix $A_1A_0A_2$ of eigenvalue one. For any of the three distributed strategies in Table I, the vector θ is the right eigenvector of the matrix A of eigenvalue one, and $A_2\theta = \theta$ (because A_2 is either A or I). In these three cases, the vector p is the right eigenvector of the matrix A of eigenvalue one. In the following discussion, we are going to show how this right eigenvector p influences the limit point. Furthermore, we also derive a simplified mean-square-error expression from (76) to quantify how close each $\boldsymbol{w}_{k,i}$ is to the Pareto-optimal point.

A. Moving along the Pareto-optimal Tradeoff Curve

By Theorems 1–2, the distributed strategy (1)–(3) converges to the limit point w^o defined by (25). Substituting $s_k(w) = \nabla_w J_k(w)$ into (25), we obtain

$$\sum_{k=1}^{N} p_k \nabla_w J_k(w^o) = 0 \tag{84}$$

In other words, w^o is the minimizer of the following global cost function:

$$J^{\mathrm{glob}}(w) = \sum_{k=1}^{N} p_k J_k(w) \tag{85}$$

It is shown in [28, pp.178–180] that the minimizer of (85) is a Pareto-optimal solution for the multi-objective optimization problem (82). And different choices of the vector p lead to different Pareto-optimal points on the tradeoff curve. Therefore, in order to converge to a certain Pareto-optimal point corresponding to a given set of positive coefficients $\{p_k\}$, we need to design a left-stochastic matrix A so that its right eigenvector of eigenvalue one is p. It was shown in [14] that one way to construct such a matrix A is by applying a procedure due to Hasting [29], [30] to set the (l,k)-th entry of A to be:

$$a_{lk} = \begin{cases} \dfrac{p_k^{-1}}{\max\left\{|\mathcal{N}_k| \cdot p_k^{-1}, |\mathcal{N}_l| \cdot p_l^{-1}\right\}}, & l \in \mathcal{N}_k \backslash \{k\} \\ 1 - \displaystyle\sum_{l \in \mathcal{N}_k \backslash \{k\}} a_{lk}, & l = k \end{cases} \tag{86}$$

where $|\mathcal{N}_k|$ denotes the cardinality of the set $\mathcal{N}_k$. The above combination matrix can be constructed in a decentralized manner, where each node only requires information from its own neighbors.

B. Mean-Square-Error Performance

When stochastic gradients are used, we can further simplify the mean-square-error expression in (76). To see this, we substitute $s_k(w) = \nabla_w J_k(w)$ into (77) and obtain

$$R_c = \sum_{k=1}^{N} p_k \nabla_w^2 J_k(w^o) \tag{87}$$

Now the matrix R_c is the weighted sum of the Hessian matrices of the individual costs $\{J_k(w)\}$ and is therefore symmetric. Then, the Lyapunov equation (78) becomes

$$R_c Z + Z R_c = \Sigma \tag{88}$$

As a result, we have simple solutions to the Lyapunov equation (78) for the following two choices of Σ:

1) When $\Sigma = I_M$, we have $Z = \frac{1}{2} R_c^{-1}$ and

$$\lim_{i\to\infty} \mathbb{E}\|\tilde{\boldsymbol{w}}_{k,i}\|^2 = \frac{\mu}{2} \cdot \mathrm{Tr}\left\{(p^T \otimes I_M)\mathcal{R}_s(p \otimes I_M)R_c^{-1}\right\} + O(\mu^2) \tag{89}$$

2) When $\Sigma = \frac{1}{2} R_c$, we have $Z = \frac{1}{4} I_M$ and

$$\lim_{i\to\infty} \mathbb{E}\|\tilde{\boldsymbol{w}}_{k,i}\|_{R_c}^2 = \frac{\mu}{4} \cdot \mathrm{Tr}\left\{(p^T \otimes I_M)\mathcal{R}_s(p \otimes I_M)\right\} + O(\mu^2) \tag{90}$$

VI. CONCLUDING REMARKS

In this paper, we studied the limiting behavior of a class of distributed strategies, namely, diffusion and consensus strategies. We showed how and in what manner the choice of combination policies has a direct bearing on the convergence and performance of the distributed solutions. Specificially, we showed that the right eigenvector of the combinaton matrix corresponding to the eigenvalue at one influences the limit point, the convergence rate and the steady-state mean-square-error (MSE) performance. A key observation is that, for sufficiently small step-sizes, the distributed strategies approach the performance of the centralized strategy in both convergence rate and steady-state MSE.

REFERENCES

[1] S. Barbarossa and G. Scutari, "Bio-inspired sensor network design," *IEEE Signal Process. Mag.*, vol. 24, no. 3, pp. 26–35, 2007.

[2] D.P. Bertsekas and J.N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*, Athena Scientific, Belmont, 1997.

[3] J. N. Tsitsiklis, D. P. Bertsekas, and M. Athans, "Distributed asynchronous deterministic and stochastic gradient optimization algorithms," *IEEE Trans. Autom. Control*, vol. 31, no. 9, pp. 803–812, 1986.

[4] S. Kar and J. M. F. Moura, "Convergence rate analysis of distributed gossip (linear parameter) estimation: Fundamental limits and tradeoffs," *IEEE J. Sel. Topics. Signal Process.*, vol. 5, no. 4, pp. 674–690, Aug. 2011.

[5] S. Kar, J. M. F. Moura, and K. Ramanan, "Distributed parameter estimation in sensor networks: Nonlinear observation models and imperfect communication," *IEEE Trans. Inf. Theory*, vol. 58, no. 6, pp. 3575–3605, June 2012.

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

[7] S. Theodoridis, K. Slavakis, and I. Yamada, "Adaptive learning in a world of projections," *IEEE Signal Process. Mag.*, vol. 28, no. 1, pp. 97–123, Jan. 2011.

[8] S. Chouvardas, K. Slavakis, and S. Theodoridis, "Adaptive robust distributed learning in diffusion sensor networks," *IEEE Trans. Signal Process.*, vol. 59, no. 10, pp. 4692–4707, Oct. 2011.

[9] S. V. Macua, P. Belanovic, and S. Zazo, "Diffusion gradient temporal difference for cooperative reinforcement learning with linear function approximation," in *Proc. IEEE International Workshop on Cognitive Information Process. (CIP)*, Parador de Baiona, Spain, May 2012, pp. 1–6.

[10] Z. J. Towfic, J. Chen, and A. H. Sayed, "On the generalization ability of online learners," in *Proc. IEEE Workshop on Machine Learning for Signal Processing (MLSP)*, Santander, Spain, Sep. 2012, pp. 1–6.

[11] A. H. Sayed, "Diffusion adaptation over networks," *to appear in* E-Reference Signal Processing*, R. Chellapa and S. Theodoridis, editors, Elsevier, 2013 [Also available online as arXiv:1205.4220v1]*, May 2012.

[12] 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, July 2008.

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

[14] X. Zhao and A. H. Sayed, "Performance limits for distributed estimation over LMS adaptive networks," *IEEE Trans. Signal Process.*, vol. 60, no. 10, pp. 5107–5124, Oct. 2012.

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

[16] J. Chen and A. H. Sayed, "Distributed Pareto-optimal solutions via diffusion adaptation," in *Proc. IEEE Workshop on Statistical Signal Process. (SSP)*, Ann Arbor, MI, Aug. 2012, pp. 1–4.

[17] S. S. Ram, A. Nedic, and V. V. Veeravalli, "Distributed stochastic subgradient projection algorithms for convex optimization," *J. Optim. Theory Appl.*, vol. 147, no. 3, pp. 516–545, 2010.

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

[19] P. Di Lorenzo and S. Barbarossa, "A bio-inspired swarming algorithm for decentralized access in cognitive radio," *IEEE Trans. Signal Process.*, vol. 59, no. 12, pp. 6160–6174, Dec. 2011.

[20] F. S. Cattivelli and A. H. Sayed, "Modeling bird flight formations using diffusion adaptation," *IEEE Trans. Signal Process.*, vol. 59, no. 5, pp. 2038–2051, May 2011.

[21] S.-Y. Tu and A. H. Sayed, "Mobile adaptive networks," *IEEE J. Sel. Topics. Signal Process.*, vol. 5, no. 4, pp. 649–664, Aug. 2011.

[22] R. S. Sutton, H. R. Maei, D. Precup, S. Bhatnagar, D. Silver, C. Szepesvári, and E. Wiewiora, "Fast gradient-descent methods for temporal-difference learning with linear function approximation," in *Proc. ACM International Conf. on Machine Learning (ICML)*, Montreal, Canada, Jun. 2009, pp. 993–1000.

[23] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, 1990.

[24] D. P. Bertsekas and J. N. Tsitsiklis, "Gradient convergence in gradient methods with errors," *SIAM J. Optim.*, vol. 10, no. 3, pp. 627–642, 2000.

[25] B. Polyak, *Introduction to Optimization*, Optimization Software, NY, 1987.

[26] A. J. Laub, *Matrix Analysis for Scientists and Engineers*, SIAM, PA, 2005.

[27] S.-Y. Tu and A. H. Sayed, "Diffusion networks outperform consensus networks," in *Proc. IEEE Statistical Signal Processing Workshop*, Ann Arbor, MI, Aug. 2012, pp. 1–4. [See also http://arxiv.org/abs/1205.3993, May 2012.]

[28] S. P. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.

[29] W. K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, vol. 57, no. 1, pp. 97–109, Apr. 1970.

[30] S. Boyd, P. Diaconis, and L. Xiao, "Fastest mixing markov chain on a graph," *SIAM Rev.*, vol. 46, no. 4, pp. 667–689, Dec. 2004.