CONTRIBUTED PAPER

Adaptive Networks

In this paper, the author surveys the field of adaptive networks and studies how collaboration among agents can lead to superior adaptation and learning performance over graphs.

By Ali H. Sayed, *Fellow IEEE*

ABSTRACT | This paper surveys recent advances related to adaptation, learning, and optimization over networks. Various distributed strategies are discussed that enable a collection of networked agents to interact locally in response to streaming data and to continually learn and adapt to track drifts in the data and models. Under reasonable technical conditions on the data, the adaptive networks are shown to be mean square stable in the slow adaptation regime, and their mean square error performance and convergence rate are characterized in terms of the network topology and data statistical moments. Classical results for single-agent adaptation and learning are recovered as special cases. The performance results presented in this work are useful in comparing network topologies against each other, and in comparing adaptive networks against centralized or batch implementations. The presentation is complemented with various examples linking together results from various domains.

KEYWORDS | Adaptation; big data; centralized strategies; consensus strategies; diffusion of information; diffusion strategies; distributed processing; incremental strategies; learning; multiagent networks; noncooperative strategies; optimization; stochastic-gradient methods

I. INTRODUCTION: COGNITION AND LEARNING

Nature is laden with examples where complex and sophisticated patterns of behavior emanate from limited interactions among simple elements, and from the aggregation and processing of decentralized pieces of information collected by dispersed agents over a graph. Examples abound in the realm of biological networks, where remarkable patterns of coordinated behavior manifest themselves, for example, in the form of fish schooling, bird formations, and bee swarming [2]. While each individual agent in these networks is incapable of complex decision making on its own, it is the continuous coordination and sharing of information among neighboring agents that lead to effective multiagent formations.

Network science is the field that deals with issues related to the aggregation, processing, and diffusion of information over graphs linking a multitude of agents. While the interactions over such graphs can be studied and characterized from the perspective of cluster formations, degrees of connectivity, and small-world effects [3]–[5], it is the possibility of having agents interact dynamically with each other, and influence each other's behavior, that opens up a plethora of notable questions. For example, a better understanding of how local interactions influence the global pattern of behavior at the network level can lead to a broader understanding of how localized interactions in the social sciences, life sciences, and system sciences influence the evolution of the respective networks. For a long time, system theory has focused on studying standalone dynamic systems with great success. However, rapid advances in the biological sciences, animal behavior studies, and in the neuroscience of the brain are revealing the striking power of coordination among networked units (e.g., [2] and [6]–[8]). These discoveries are motivating greater efforts toward a deeper examination of information processing over graphs in several disciplines including signal processing, machine learning, optimization, and control (see, e.g., [1], [9], and the references therein).

This paper surveys the field of adaptive networks and how collaboration among agents can lead to superior adaptation and learning performance over graphs. Adaptive networks consist of a collection of agents with learning abilities. The agents interact with each other on a local level and diffuse information across the network to solve inference and optimization tasks in a decentralized manner. Such networks are scalable, robust to node and

Manuscript received December 14, 2013; revised February 5, 2014; accepted February 6, 2014. Date of current version March 25, 2014. This work was supported in part by the National Science Foundation (NSF) under Grant CCF-1011918. This work is a condensed version that integrates select material and examples from the 400-page manuscript [1] into an overview of results on adaptive networks.

The author is with the Electrical Engineering Department, University of California Los Angeles, Los Angeles, CA 90095-1594 USA (e-mail: sayed@ee.ucla.edu).

Digital Object Identifier: 10.1109/JPROC.2014.2306253

link failures, and are particularly suitable for learning from big data sets by tapping into the power of collaboration among distributed agents. The networks are also endowed with cognitive abilities due to the sensing abilities of their agents, their interactions with their neighbors, and the embedded feedback mechanisms for acquiring and refining information. Each agent is not only capable of sensing data and experiencing the environment directly, but it also receives information through interactions with its neighbors and processes and analyzes this information to drive its learning process.

In order to highlight the main features of adaptive networks, we organize the presentation of the article into three main components. Section II reviews fundamental results on adaptation and learning by *single* standalone agents. The emphasis is on stochastic-gradient constructions. A general formulation is considered that allows us to extract classical results for adaptive filtering as special cases. The level of generality considered in this section is meant to bring forth commonalities that exist among several domains relating to adaptation, learning, and optimization.

Section III covers centralized solutions. The objective is to explain the gain in performance that results from aggregating the data from the agents and processing it centrally at a fusion center. The centralized performance is used as a frame of reference for assessing various implementations. While centralized solutions can be powerful, they nevertheless suffer from a number of limitations. First, in real-time applications where agents collect data continuously, the repeated exchange of information back and forth between the agents and the fusion center can be costly, especially when these exchanges occur over wireless links or require nontrivial routing resources. Second, in some sensitive applications, agents may be reluctant to share their data with remote centers for various reasons, including privacy and secrecy considerations. More importantly perhaps, centralized solutions have a critical point of failure: if the central processor fails, then this solution method collapses altogether.

For these reasons, we cover in Sections IV–VIII several distributed strategies (incremental, consensus, and diffusion), and study their dynamics, stability, and performance metrics. In the distributed mode of operation, agents are connected by a topology, and they are permitted to share information only with their immediate neighbors. The study of the behavior of such networked agents is more challenging than in the single-agent and centralized modes of operation due to the coupling among interacting agents and due to the fact that the networks are generally sparsely connected. The presentation in the paper clarifies the effect of network topology on performance and leads to results that enable the designer to compare various strategies against each other and against the centralized solution.

As indicated in [1], there are many good reasons for the peaked interest in distributed solutions, especially in this day and age when the word "network" has become commonplace whether one is referring to social networks, power networks, transportation networks, biological networks, or other networks. Some of these reasons have to do with the benefits of cooperation over networks in terms of improved performance and improved robustness and resilience to failure. Other reasons deal with privacy and secrecy considerations where agents may not be comfortable sharing their data with remote fusion centers. In other situations, the data may already be available in dispersed locations, as happens with cloud computing. One may also be interested in learning and extracting information through data mining from large data sets. Decentralized learning procedures offer an attractive approach to dealing with such data sets. Decentralized mechanisms can also serve as important enablers for the design of robotic swarms, which can assist in the exploration of disaster areas.

Motivated by these observations, we devote reasonable effort toward clarifying the limits of performance of distributed solutions by relying on statistical analysis tools. While several of the algorithms can be motivated in alternative ways, we opt to present them by using a common stochastic-gradient approximation framework. Whenever necessary, derivations are provided to complement the discussion with references to the pertinent literature for longer arguments that are omitted for space considerations. The results are illustrated by examples dealing with applications involving distributed estimation, learning, optimization, and adaptation. For other applications in the areas of intrusion detection, online dictionary learning, target localization, spectrum sensing, sparse data recovery, and biological networks, the reader may refer to [1], [10]–[14], and the references therein. Section IX comments on additional topics, such as gossip and asynchronous strategies, constrained optimization, sparsity constraints, noisy information exchanges, and least squares and Kalman-type strategies.

II. SINGLE-AGENT ADAPTATION AND LEARNING

We begin our treatment by reviewing stochastic-gradient algorithms, with emphasis on their application to the problems of adaptation and learning by standalone agents. We will be using the term "learning" to refer broadly to the ability of an agent to extract information about some unknown parameter from streaming data, such as estimating the parameter itself or learning about some of its features. We will be using the term "adaptation" to refer broadly to the ability of the learning algorithm to track drifts in the parameter, which are usually reflected in changes in the statistical properties of the observed data. The two attributes of learning and adaptation will be embedded simultaneously into the algorithms discussed in this work. We will also be using the term "streaming data" regularly because we are interested in algorithms that perform continuous learning and adaptation and that, therefore, are able to improve their performance in response to continuous streams of data arriving at the agent(s). This is in contrast to offline algorithms, where the data are first aggregated before being processed for extraction of information. The presentation in this section summarizes some classical results on stochastic-gradient algorithms for adaptation and learning, and provides some additional insights that are useful for our later study of the more demanding scenario of adaptation and learning by a collection of networked agents.

A. Risk and Loss Functions

Thus, let $J(w) : \mathbb{R}^{M\times 1} \mapsto \mathbb{R}$ denote a real-valued (cost or utility or risk) function of a real-valued vector argument $w \in \mathbb{R}^{M\times 1}$. The variable w can be complex valued, and many of the results in this work can be extended to the complex domain as well. However, some important technical differences arise when dealing with complex arguments. These differences are beyond the scope of this paper, and they are addressed in [1], [15], and [16], along with other relevant topics. It is sufficient for our purposes here to convey the main ideas by limiting the presentation to real arguments without much loss in generality.

We denote the gradient vectors of $J(w)$ relative to w and $w^\top$ by the following row and column vectors, respectively, where the first expression is also referred to as the Jacobian of $J(w)$ relative to w:

$$\nabla_w J(w) \overset{\Delta}{=} \left[\frac{\partial J(w)}{\partial w_1}\ \frac{\partial J(w)}{\partial w_2}\ \ldots\ \frac{\partial J(w)}{\partial w_M}\right] \tag{1a}$$

$$\nabla_{w^\top} J(w) \overset{\Delta}{=} [\nabla_w J(w)]^\top. \tag{1b}$$

These definitions are in terms of the partial derivatives of $J(w)$ relative to the individual entries of $w = \mathrm{col}\{w_1, w_2, \ldots, w_M\}$, where the notation $\mathrm{col}\{\cdot\}$ refers to a column vector that is formed by stacking the arguments of $\mathrm{col}\{\cdot\}$ on top of each other. Likewise, the Hessian matrix of $J(w)$ with respect to w is defined as the following $M \times M$ symmetric matrix:

$$\nabla_w^2 J(w) \overset{\Delta}{=} \nabla_{w^\top}[\nabla_w J(w)] = \nabla_w[\nabla_{w^\top} J(w)] \tag{1c}$$

which is constructed from two successive gradient operations. It is common in adaptation and learning applications for the risk function $J(w)$ to be constructed as the expectation of some loss function, $Q(w;\boldsymbol{x})$, where the boldface variable $\boldsymbol{x}$ is used to denote some random data, say

$$J(w) = \mathbb{E}\, Q(w;\boldsymbol{x}) \tag{2}$$

and the expectation is evaluated over the distribution of $\boldsymbol{x}$.

Example 1 (Mean Square Error Costs): Let $\boldsymbol{d}$ denote a zero-mean scalar random variable with variance $\sigma_d^2 = \mathbb{E}\boldsymbol{d}^2$, and let $\boldsymbol{u}$ denote a zero-mean $1 \times M$ random vector with covariance matrix $R_u = \mathbb{E}\boldsymbol{u}^\top\boldsymbol{u} > 0$. The combined quantities $\{\boldsymbol{d},\boldsymbol{u}\}$ represent the random variable $\boldsymbol{x}$ referred to in (2). The cross-covariance vector is denoted by $r_{du} = \mathbb{E}\boldsymbol{d}\boldsymbol{u}^\top$. We formulate the problem of estimating $\boldsymbol{d}$ from $\boldsymbol{u}$ in the linear least mean squares sense or, equivalently, the problem of seeking the vector w^o that minimizes the quadratic cost function

$$\begin{aligned} J(w) &\overset{\Delta}{=} \mathbb{E}(\boldsymbol{d} - \boldsymbol{u}w)^2 \\ &= \sigma_d^2 - (r_{du})^\top w - w^\top r_{du} + w^\top R_u w. \end{aligned} \tag{3a}$$

This cost corresponds to the following choice for the loss function:

$$Q(w;\boldsymbol{x}) = (\boldsymbol{d} - \boldsymbol{u}w)^2. \tag{3b}$$

Such quadratic costs are widely used in estimation and adaptation problems [17]–[21]. They are also widely used as quadratic risk functions in machine learning applications [22], [23]. The gradient vector and Hessian matrix of $J(w)$ are easily seen to be

$$\nabla_w J(w) = 2(R_u w - r_{du})^\top, \quad \nabla_w^2 J(w) = 2R_u. \tag{3c}$$

♦

Example 2 (Logistic or Log-Loss Risks): Let $\boldsymbol{\gamma}$ denote a binary random variable that assumes the values ±1, and let $\boldsymbol{h}$ denote an $M \times 1$ random (feature) vector with $R_h = \mathbb{E}\boldsymbol{h}\boldsymbol{h}^\top$. The combined quantities $\{\boldsymbol{\gamma},\boldsymbol{h}\}$ represent the random variable $\boldsymbol{x}$ referred to in (2). In the context of machine learning and pattern classification problems [22]–[24], the variable $\boldsymbol{\gamma}$ designates the class that feature vector $\boldsymbol{h}$ belongs to. In these problems, one seeks the vector w^o that minimizes the regularized logistic risk function

$$J(w) \overset{\Delta}{=} \frac{\rho}{2}\|w\|^2 + \mathbb{E}\left\{\ln\left[1 + e^{-\boldsymbol{\gamma}\boldsymbol{h}^\top w}\right]\right\} \tag{4a}$$

where $\rho > 0$ is some regularization parameter, $\ln(\cdot)$ is the natural logarithm function, and $\|w\|^2 = w^\top w$. The risk equation (4a) corresponds to the following choice for the loss function:

$$Q(w;\boldsymbol{x}) \overset{\Delta}{=} \frac{\rho}{2}\|w\|^2 + \ln\left[1 + e^{-\boldsymbol{\gamma}\boldsymbol{h}^\top w}\right]. \tag{4b}$$

Once w^o is recovered, its value can be used to classify new feature vectors, say, $\{\boldsymbol{h}_\ell\}$, into classes +1 or −1. This can be achieved, for example, by assigning feature vectors with $\boldsymbol{h}_\ell^\top w^o \geq 0$ to one class and feature vectors with $\boldsymbol{h}_\ell^\top w^o < 0$

to another class. It can be easily verified that for the above $J(w)$:

$$\nabla_w J(w) = \rho w^\top - \mathbb{E}\left\{\gamma \boldsymbol{h}^\top \cdot \frac{e^{-\gamma \boldsymbol{h}^\top w}}{1+e^{-\gamma \boldsymbol{h}^\top w}}\right\} \tag{4c}$$

$$\nabla_w^2 J(w) = \rho I_M + \mathbb{E}\left\{\boldsymbol{h}\boldsymbol{h}^\top \cdot \frac{e^{-\gamma \boldsymbol{h}^\top w}}{\left(1+e^{-\gamma \boldsymbol{h}^\top w}\right)^2}\right\}. \tag{4d}$$

♦

B. Conditions on Cost Function

Stochastic-gradient algorithms are powerful iterative procedures for solving optimization problems of the form

$$\min_w J(w). \tag{5}$$

While the analysis that follows can be pursued under more relaxed conditions (see, e.g., the treatments in [25]–[28]), it is sufficient for our purposes to require $J(w)$ to be strongly convex and twice differentiable with respect to w. The cost function $J(w)$ is said to be ν-strongly convex if, and only if, its Hessian matrix is sufficiently bounded away from zero [26], [29]–[31]

$$J(w) \text{ is } \nu\text{-strongly convex} \Longleftrightarrow \nabla_w^2 J(w) \geq \nu I_M > 0 \tag{6a}$$

for all w and for some scalar $\nu > 0$, where I_M denotes the identity matrix of size $M \times M$ and the notation $A > 0$ signifies that matrix A is positive definite. Strong convexity is a useful condition in the context of adaptation and learning from streaming data because it helps guard against ill-conditioning in the algorithms; it also helps ensure that $J(w)$ has a *unique* global minimum, say, at location w^o; there will be no other minima, maxima, or saddle points. In addition, it is well known that strong convexity endows stochastic-gradient algorithms with geometric convergence rates in the order of $O(\alpha^i)$, for some $0 \leq \alpha < 1$ and where i is the iteration index [26], [27]. For comparison purposes, when the function $J(w)$ is convex but not necessarily strongly convex, then convexity is equivalent to the following condition:

$$J(w) \text{ is convex} \Longleftrightarrow \nabla_w^2 J(w) \geq 0 \tag{6b}$$

for all w. In this case, there can now be multiple global minima. Moreover, the convergence of stochastic-gradient algorithms will occur at the slower rate of $O(1/i)$ [26], [27].

In many problems of interest in adaptation and learning, the cost function $J(w)$ is either already strongly convex or can be made strongly convex by means of regularization. For example, it is common in machine learning problems [22], [23] and in adaptation and estimation problems [19], [21] to incorporate regularization factors into the cost functions; these factors help ensure strong convexity automatically. For instance, the mean square error (MSE) cost (3a) is strongly convex whenever $R_u > 0$. If R_u happens to be singular, then the following regularized cost will be strongly convex:

$$J(w) \overset{\Delta}{=} \frac{\rho}{2}\|w\|^2 + \mathbb{E}(\boldsymbol{d} - \boldsymbol{u}w)^2 \tag{7}$$

where $\rho > 0$ is a regularization parameter similar to (4a).

Besides strong convexity, we will additionally assume that the gradient vector of $J(w)$ is δ-Lipschitz, namely, there exists $\delta > 0$ such that

$$\|\nabla_w J(w_2) - \nabla_w J(w_1)\| \leq \delta \|w_2 - w_1\| \tag{8}$$

for all w_1, w_2. It can be verified that for twice-differentiable costs, conditions (6a) and (8) combined are equivalent to

$$0 < \nu I_M \leq \nabla_w^2 J(w) \leq \delta I_M. \tag{9}$$

For example, it is clear that the Hessian matrices in (3c) and (4d) satisfy this property since

$$2\lambda_{\min}(R_u) I_M \leq \nabla_w^2 J(w) \leq 2\lambda_{\max}(R_u) I_M \tag{10a}$$

in the first case and

$$\rho I_M \leq \nabla_w^2 J(w) \leq (\rho + \lambda_{\max}(R_h)) I_M \tag{10b}$$

in the second case, where the notation $\lambda_{\min}(R)$ and $\lambda_{\max}(R)$ refers to the smallest and largest eigenvalues of the symmetric matrix argument R, respectively. In summary, we will be assuming the following conditions [1], [15], [32], [33].

Assumption II.1 (Conditions on Cost Function): The cost function $J(w)$ is twice differentiable and satisfies (9) for some positive parameters $\nu \leq \delta$. Condition (9) is equivalent to requiring $J(w)$ to be ν-strongly convex and for its gradient vector to be δ-Lipschitz as in (6a) and (8), respectively. ♦

C. Stochastic-Gradient Approximation

The traditional gradient–descent algorithm for solving (5) is given by

$$w_i = w_{i-1} - \mu \nabla_{w^\top} J(w_{i-1}), \qquad i \geq 0 \tag{11}$$

where $i \geq 0$ is an iteration index and $\mu > 0$ is a small step-size parameter. Starting from some initial condition w_{-1}, the iterates $\{w_i\}$ correspond to successive estimates for the minimizer w^o. In order to run recursion (11), we need to have access to the true gradient vector. This information is generally unavailable in most instances involving learning from data. For example, when cost functions are defined as the expectations of certain loss functions as in (2), the statistical distribution of the data $\boldsymbol{x}$ may not be known beforehand. In that case, the exact form of $J(w)$ will not be known since the expectation of $Q(w;\boldsymbol{x})$ cannot be computed. In such situations, it is customary to replace the true gradient vector $\nabla_{w^\top} J(w_{i-1})$ by an instantaneous approximation for it, and which we will denote by $\widehat{\nabla_{w^\top}} J(\boldsymbol{w}_{i-1})$. Doing so leads to the following *stochastic-gradient* recursion in lieu of (11):

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} - \mu \widehat{\nabla_{w^\top} J}(\boldsymbol{w}_{i-1}), \qquad i \geq 0. \qquad (12)$$

We use the boldface notation $\boldsymbol{w}_i$ for the iterates in (12) to highlight the fact that these iterates are now randomly perturbed versions of the values $\{w_i\}$ generated by the original recursion (11). The random perturbations arise from the use of the approximate gradient vector. The boldface notation is, therefore, meant to emphasize the random nature of the iterates in (12). We illustrate construction (12) by considering a scenario from classical adaptive filter theory [17]–[19], where the gradient vector is approximated directly from data realizations. The construction will reveal why stochastic-gradient implementations of the form (12), using approximate rather than exact gradient information, become naturally endowed with the ability to respond to *streaming* data.

Example 3 (LMS Adaptation): Let $\boldsymbol{d}(i)$ denote a streaming sequence of zero-mean random variables with variance $\sigma_d^2 = \mathbb{E}\boldsymbol{d}^2(i)$. Let $\boldsymbol{u}_i$ denote a streaming sequence of $1 \times M$ independent zero-mean random vectors with covariance matrix $R_u = \mathbb{E}\boldsymbol{u}_i^\top \boldsymbol{u}_i > 0$. Both processes $\{\boldsymbol{d}(i), \boldsymbol{u}_i\}$ are assumed to be jointly wide-sense stationary. The cross-covariance vector between $\boldsymbol{d}(i)$ and $\boldsymbol{u}_i$ is denoted by $r_{du} = \mathbb{E}\boldsymbol{d}(i)\boldsymbol{u}_i^\top$. The data $\{\boldsymbol{d}(i), \boldsymbol{u}_i\}$ are assumed to be related via a linear regression model of the form

$$\boldsymbol{d}(i) = \boldsymbol{u}_i w^o + \boldsymbol{v}(i) \qquad (13a)$$

for some unknown parameter vector w^o, and where $\boldsymbol{v}(i)$ is a zero-mean white-noise process with power $\sigma_v^2 = \mathbb{E}\boldsymbol{v}^2(i)$ and assumed independent of $\boldsymbol{u}_j$ for all i, j. Observe that we are using parentheses to represent the time dependency of a scalar variable, such as writing $\boldsymbol{d}(i)$, and subscripts to represent the time dependency of a vector variable, such as writing $\boldsymbol{u}_i$. This convention will be used throughout the paper. In a manner similar to Example 1, we again pose the problem of estimating w^o by minimizing the MSE cost

$$J(w) = \mathbb{E}(\boldsymbol{d}(i) - \boldsymbol{u}_i w)^2 \equiv \mathbb{E}Q(w;\boldsymbol{x}_i) \qquad (13b)$$

where now the quantities $\{\boldsymbol{d}(i), \boldsymbol{u}_i\}$ represent the random data $\boldsymbol{x}_i$ in the definition of the loss function $Q(w;\boldsymbol{x}_i)$. Using (11), the gradient–descent recursion in this case will take the form

$$w_i = w_{i-1} - 2\mu[R_u w_{i-1} - r_{du}], \qquad i \geq 0. \qquad (13c)$$

The main difficulty in running this recursion is that it requires knowledge of the moments $\{r_{du}, R_u\}$. This information is rarely available beforehand; the adaptive agent senses instead realizations $\{\boldsymbol{d}(i), \boldsymbol{u}_i\}$ whose statistical distributions have moments $\{r_{du}, R_u\}$. The agent can use these realizations to approximate the moments and the true gradient vector. There are many constructions that can be used for this purpose, with different constructions leading to different adaptive algorithms [17]–[20]. It is sufficient to illustrate the construction by focusing on one of the most popular adaptive algorithms, which results from using the data $\{\boldsymbol{d}(i), \boldsymbol{u}_i\}$ to compute *instantaneous* approximations for the unavailable moments as follows:

$$r_{du} \approx \boldsymbol{d}(i)\boldsymbol{u}_i^\top, \quad R_u \approx \boldsymbol{u}_i^\top \boldsymbol{u}_i. \qquad (13d)$$

By doing so, the true gradient vector is approximated by

$$\widehat{\nabla_{w^\top} J}(w) = 2\left[\boldsymbol{u}_i^\top \boldsymbol{u}_i w - \boldsymbol{u}_i^\top \boldsymbol{d}(i)\right] = \nabla_{w^\top} Q(w;\boldsymbol{x}_i). \qquad (13e)$$

Observe that this construction amounts to replacing the true gradient vector $\nabla_{w^\top} J(w)$ by the gradient vector of the loss function itself (which, equivalently, amounts to dropping the expectation operator). Substituting (13e) into (13c) leads to the well-known least mean squares (LMS) algorithm [17]–[19]

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} + 2\mu \boldsymbol{u}_i^\top [\boldsymbol{d}(i) - \boldsymbol{u}_i \boldsymbol{w}_{i-1}], \qquad i \geq 0. \qquad (13f)$$

The LMS algorithm is therefore a stochastic-gradient algorithm. By relying directly on the instantaneous data $\{\boldsymbol{d}(i), \boldsymbol{u}_i\}$, the algorithm is infused with useful tracking abilities. This is because drifts in the model w^o from (13a) will be reflected in the data $\{\boldsymbol{d}(i), \boldsymbol{u}_i\}$, which are used directly in the update (13f). ♦

The idea of using sample realizations to approximate actual expectations, as was the case with step (13e), is at the core of what is known as *stochastic approximation theory*. According to [19] and [28], the pioneering work in the field of stochastic approximation is that of [34], which is a variation of a scheme developed about two decades earlier in [35]. The work by [34] dealt primarily with *scalar* weights w and was extended later by [36] and [37] to weight *vectors*; see [38]. During the 1950s, stochastic approximation theory did not receive much attention in the engineering community until the landmark work by

[39], which developed the real form of the LMS algorithm [see (13f)]. The algorithm has since then found remarkable success in a wide range of applications.

If desired, it is also possible to employ iteration-dependent step-size sequences $\mu(i)$ in (12) instead of the constant step-size μ, and to require $\mu(i)$ to satisfy

$$\sum_{i=0}^{\infty} \mu^2(i) < \infty, \quad \sum_{i=0}^{\infty} \mu(i) = \infty. \tag{14}$$

Under some technical conditions, it is well known that such step-size sequences ensure the convergence of $\boldsymbol{w}_i$ toward w^o almost surely as $i \to \infty$ [26]–[28]. However, conditions (14) force the step-size sequence to decay to zero, which is problematic for applications requiring continuous adaptation and learning from streaming data. This is because, in such applications, it is not unusual for the location of the minimizer w^o to drift with time. With $\mu(i)$ decaying toward zero, the stochastic-gradient algorithm (12) will stop updating and will not be able to track drifts in the solution. For this reason, we will focus on constant step sizes from this point onward since we are interested in solutions with tracking abilities.

Now, the use of an approximate gradient vector in (12) introduces perturbations relative to the operation of the original recursion (11). We refer to the perturbation as gradient noise and define it as the difference

$$\boldsymbol{s}_i(\boldsymbol{w}_{i-1}) \overset{\Delta}{=} \widehat{\nabla_{w^\top} J}(\boldsymbol{w}_{i-1}) - \nabla_{w^\top} J(\boldsymbol{w}_{i-1}). \tag{15}$$

The presence of this perturbation prevents the stochastic iterate $\boldsymbol{w}_i$ from converging almost surely to the minimizer w^o when constant step sizes are used. Some deterioration in performance will occur, and the iterate $\boldsymbol{w}_i$ will instead fluctuate close to w^o. We will assess the size of these fluctuations by measuring their steady-state mean square value [also called mean square deviation (MSD)]. It will turn out that the MSD is small and in the order of $O(\mu)$; see (21c). It will also turn out that stochastic-gradient algorithms converge toward their MSD levels at a geometric rate. In this way, we will be able to conclude that adaptation with small constant step sizes can still lead to reliable performance in the presence of gradient noise, which is a reassuring result. We will also be able to conclude that adaptation with constant step sizes is useful even for stationary environments. This is because it is generally sufficient in practice to attain an iterate $\boldsymbol{w}_i$ within some fidelity level from w^o in a *finite* number of iterations. As long as the MSD level is satisfactory, a stochastic-gradient algorithm will be able to attain satisfactory fidelity within a reasonable time frame. In comparison, although diminishing step sizes ensure almost-sure convergence of $\boldsymbol{w}_i$ to w^o, they nevertheless disable tracking and can only guarantee slower than geometric rates of convergence (see, e.g., [1], [26], and [27]).

The next example from [32] illustrates the nature of the gradient noise process (15) in the context of MSE adaptation.

Example 4 (Gradient Noise): It is clear from the expressions in Example 3 that the corresponding gradient noise process is

$$\boldsymbol{s}_i(\boldsymbol{w}_{i-1}) = 2\left(R_u - \boldsymbol{u}_i^\top \boldsymbol{u}_i\right) \widetilde{\boldsymbol{w}}_{i-1} - 2\boldsymbol{u}_i^\top \boldsymbol{v}(i) \tag{16a}$$

where we introduced the error vector $\widetilde{\boldsymbol{w}}_i = w^o - \boldsymbol{w}_i$. Let the symbol $\boldsymbol{\mathcal{F}}_{i-1}$ represent the collection of all possible random events generated by the past iterates $\{\boldsymbol{w}_j\}$ up to time $j \leq i-1$ (more formally, $\boldsymbol{\mathcal{F}}_{i-1}$ is the filtration generated by the random process $\boldsymbol{w}_j$ for $j \leq i-1$)

$$\boldsymbol{\mathcal{F}}_{i-1} \overset{\Delta}{=} \text{filtration}\{\boldsymbol{w}_{-1}, \boldsymbol{w}_o, \boldsymbol{w}_1, \ldots, \boldsymbol{w}_{i-1}\}. \tag{16b}$$

It follows from the conditions on the random processes $\{\boldsymbol{u}_i, \boldsymbol{v}(i)\}$ in Example 3 that

$$\mathbb{E}[\boldsymbol{s}_i(\boldsymbol{w}_{i-1})|\boldsymbol{\mathcal{F}}_{i-1}] = 0 \tag{16c}$$

$$\mathbb{E}\left[\|\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2|\boldsymbol{\mathcal{F}}_{i-1}\right] \leq 4c\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + 4\sigma_v^2\ \mathrm{Tr}(R_u) \tag{16d}$$

for some constant $c \geq 0$. If we take expectations of both sides of (16d), we further conclude that the variance of the gradient noise $\mathbb{E}\|\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2$ is bounded by the combination of two factors. The first factor depends on the quality of the iterate $\mathbb{E}\|\widetilde{\boldsymbol{w}}_{i-1}\|^2$, while the second factor depends on σ_v^2. Therefore, even if the adaptive agent is able to approach w^o with great fidelity so that $\mathbb{E}\|\widetilde{\boldsymbol{w}}_{i-1}\|^2$ is small, the size of the gradient noise will still depend on σ_v^2. ♦

D. Conditions on Gradient Noise Process

In order to examine the convergence and performance properties of the stochastic-gradient recursion (12), it is necessary to introduce some assumptions on the stochastic nature of the gradient noise process $\boldsymbol{s}_i(\cdot)$. The conditions that we introduce in the rest of the paper are similar to conditions used earlier in the optimization literature, e.g., in [26, pp. 95–102] and [40, p. 635]; they are also motivated by the conditions we observed in the MSE case in Example 4. Following the developments in [15], [32], and [33], we let

$$R_{s,i}(\boldsymbol{w}_{i-1}) \overset{\Delta}{=} \mathbb{E}\left[\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\boldsymbol{s}_i^\top(\boldsymbol{w}_{i-1})|\boldsymbol{\mathcal{F}}_{i-1}\right] \tag{17a}$$

denote the conditional second-order moment of the gradient noise process, which generally depends on i. We assume that, in the limit, this covariance matrix tends to a constant value when evaluated at w^o and denote it by

$$R_s \overset{\Delta}{=} \lim_{i\to\infty} \mathbb{E}\left[\boldsymbol{s}_i(w^o)\boldsymbol{s}_i^\top(w^o)|\boldsymbol{\mathcal{F}}_{i-1}\right]. \tag{17b}$$

We sometimes refer to the term $\boldsymbol{s}_i(w^o)$ as the *absolute* noise component. For example, comparing with example (16a) for MSE costs, we have

$$\boldsymbol{s}_i(w^o) = -2\boldsymbol{u}_i^\top \boldsymbol{v}(i) \tag{18a}$$

$$R_s = 4\sigma_v^2 R_u. \tag{18b}$$

Assumption II.2 (Conditions on Gradient Noise): It is assumed that the first- and second-order conditional moments of the gradient noise process satisfies (17b) and

$$\mathbb{E}[\boldsymbol{s}_i(\boldsymbol{w}_{i-1})|\boldsymbol{\mathcal{F}}_{i-1}] = 0 \tag{19a}$$

$$\mathbb{E}\left[\|\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2|\boldsymbol{\mathcal{F}}_{i-1}\right] \leq \beta^2\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + \sigma_s^2 \tag{19b}$$

almost surely, for some nonnegative scalars β^2 and σ_s^2. ♦

Condition (19a) ensures that the construction of the approximate gradient vector is unbiased. It follows from conditions (19a)–(19b) that the gradient noise process itself satisfies:

$$\mathbb{E}\boldsymbol{s}_i(\boldsymbol{w}_{i-1}) = 0 \tag{20a}$$

$$\mathbb{E}\|\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2 \leq \beta^2\mathbb{E}\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + \sigma_s^2. \tag{20b}$$

It is straightforward to verify that the gradient noise process (16a) in the MSE case satisfies conditions (19a)–(19e). Note in particular from (16d) that we can make the identifications $\sigma_s^2 \rightarrow 4\sigma_v^2\mathrm{Tr}(R_u)$ and $\beta^2 \rightarrow 4c$.

E. MSE Stability

We now examine the convergence of the stochastic-gradient recursion (12). In the statement below, the notation $a = O(\mu)$ means $a \leq b\mu$ for some constant b that is independent of μ.

Lemma (MSE Stability): Assume the conditions under Assumptions II.1 and II.2 on the cost function and the gradient noise process hold. Let $\mu_o = 2\nu/(\delta^2 + \beta^2)$. For any $\mu < \mu_o$, it holds that $\mathbb{E}\|\widetilde{\boldsymbol{w}}_i\|^2$ converges exponentially (i.e., at a geometric rate) according to the recursion

$$\mathbb{E}\|\widetilde{\boldsymbol{w}}_i\|^2 \leq \alpha\mathbb{E}\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + \mu^2\sigma_s^2 \tag{21a}$$

where the scalar α satisfies $0 \leq \alpha < 1$ and is given by

$$\alpha = 1 - 2\nu\mu + (\delta^2 + \beta^2)\mu^2 = 1 - 2\nu\mu + O(\mu^2). \tag{21b}$$

It follows from (21a) that, for sufficiently small step-sizes:

$$\limsup_{i\rightarrow\infty} \mathbb{E}\|\widetilde{\boldsymbol{w}}_i\|^2 = O(\mu). \tag{21c}$$

Proof: While the result can be established in other ways, we employ instead an argument that is convenient for more general stochastic-gradient implementations, such as the ones that we will encounter later in the context of networked agents [15], [32], [33], [41]. We subtract w^o from both sides of (12) to get

$$\widetilde{\boldsymbol{w}}_i = \widetilde{\boldsymbol{w}}_{i-1} + \mu\nabla_{w^\top}J(\boldsymbol{w}_{i-1}) + \mu\boldsymbol{s}_i(\boldsymbol{w}_{i-1}). \tag{22a}$$

We now appeal to the mean value theorem [26], [48] to write

$$\begin{aligned}\nabla_{w^\top}J(\boldsymbol{w}_{i-1}) &= -\left(\int_0^1 \nabla_w^2 J(w_o - t\widetilde{\boldsymbol{w}}_{i-1})dt\right)\widetilde{\boldsymbol{w}}_{i-1}\\ &\overset{\Delta}{=} -\boldsymbol{H}_{i-1}\widetilde{\boldsymbol{w}}_{i-1}\end{aligned} \tag{22b}$$

where we are introducing the *symmetric* and *random* time-variant matrix $\boldsymbol{H}_{i-1}$ to represent the integral expression. Substituting into (22a), we get

$$\widetilde{\boldsymbol{w}}_i = (I_M - \mu\boldsymbol{H}_{i-1})\widetilde{\boldsymbol{w}}_{i-1} + \mu\boldsymbol{s}_i(\boldsymbol{w}_{i-1}) \tag{22c}$$

so that from Assumption II.2

$$\begin{aligned}\mathbb{E}\left[\|\widetilde{\boldsymbol{w}}_i\|^2|\boldsymbol{\mathcal{F}}_{i-1}\right] &\leq \|I_M - \mu\boldsymbol{H}_{i-1}\|^2\|\widetilde{\boldsymbol{w}}_{i-1}\|^2\\ &\quad + \mu^2\mathbb{E}\left[\|\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2|\boldsymbol{\mathcal{F}}_{i-1}\right]\\ &\leq \alpha\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + \mu^2\sigma_s^2\end{aligned} \tag{22d}$$

where the second inequality follows from (20b) and from using (9) to note that

$$\begin{aligned}\|I_M - \mu\boldsymbol{H}_{i-1}\|^2 &= [\rho(I_M - \mu\boldsymbol{H}_{i-1})]^2\\ &\leq \max\{(1-\mu\delta)^2, (1-\mu\nu)^2\}\\ &\leq 1 - 2\mu\nu + \mu^2\delta^2\end{aligned} \tag{22e}$$

since $\nu \leq \delta$. In the first line above, the notation $\rho(A)$ denotes the spectral radius of its matrix argument (i.e., $\rho(A) = \max_k|\lambda_k(A)|$ in terms of the largest magnitude eigenvalue of A). Taking expectations of both sides of (22d) we obtain (21a). The bound $\mu < \mu_o$ on the step size ensures that $0 \leq \alpha < 1$. Iterating recursion (21a) gives

$$\mathbb{E}\|\widetilde{\boldsymbol{w}}_i\|^2 \leq \alpha^{i+1}\mathbb{E}\|\widetilde{\boldsymbol{w}}_{-1}\|^2 + \frac{\mu^2\sigma_s^2}{1-\alpha} \tag{22f}$$

which proves that $\mathbb{E}\|\widetilde{\boldsymbol{w}}_i\|^2$ converges exponentially to a region that is upper bounded by $\mu^2\sigma_s^2/(1-\alpha)$. It can be verified that this bound does not exceed $\mu\sigma_s^2/\nu$, which is $O(\mu)$, for any $\mu < \mu_o/2$. ■

F. MSE Performance

We conclude from (21c) that the MSE can be made as small as desired by using small step sizes. In this section, we derive a closed-form expression for the asymptotic MSE, which is more frequently called MSD, and is defined as

$$\text{MSD} \overset{\Delta}{=} \lim_{i\to\infty} \mathbb{E}\|\widetilde{\boldsymbol{w}}_i\|^2. \tag{23}$$

This is a useful step to pursue because, once performance expressions are available, it becomes possible to carry out meaningful comparisons among different configurations for adaptation and learning (such as noncooperative, centralized, and distributed implementations). It also becomes possible to quantify how performance depends on the algorithm and system parameters (such as step size, network topology, and cooperation policy); these parameters can then be optimized for enhanced performance. We explain below how an expression for the MSD can be obtained by following the energy conservation technique of [19], [20], [42], and [43]. For that purpose, we need to introduce two smoothness conditions.

Assumption II.3 (Smoothness Conditions): In addition to Assumptions II.1 and II.2, we assume that the Hessian matrix of the cost function and the noise covariance matrix defined by (17a) are locally Lipschitz continuous in a small neighborhood around $w = w^o$

$$\left\|\nabla_w^2 J(w^o + \delta w) - \nabla_w^2 J(w^o)\right\| \le \tau\|\delta w\| \tag{24a}$$

$$\|R_{s,i}(w^o + \delta w) - R_{s,i}(w^o)\| \le \tau_2\|\delta w\|^{\kappa} \tag{24b}$$

for small perturbations $\|\delta w\| \le r$ and for some $\tau, \tau_2 \ge 0$ and $1 \le \kappa \le 2$. ♦

The range of values for κ can be enlarged [47], e.g., to $\kappa \in (0, 4]$; it is sufficient for our purposes to continue with $\kappa \in [1, 2]$. Using (9), it can be verified that condition (24) translates into a global Lipschitz property relative to the minimizer w^o, i.e., it will also hold that [16]

$$\left\|\nabla_w^2 J(w) - \nabla_w^2 J(w^o)\right\| \le \tau'\|w - w^o\| \tag{25}$$

for all w and for some $\tau' \ge 0$. For example, both conditions (24a)–(24b) are readily satisfied by mean-square-error costs. Using property (25), we can now motivate a useful (long-term) model for the evolution of the error vector $\widetilde{\boldsymbol{w}}_i$ after sufficient iterations, i.e., for $i \gg 1$. Indeed, let us reconsider recursion (22c) and introduce the deviation

$$\widetilde{\boldsymbol{H}}_{i-1} \overset{\Delta}{=} H - \boldsymbol{H}_{i-1} \tag{26a}$$

where the constant (symmetric and positive definite) matrix H is defined as

$$H \overset{\Delta}{=} \nabla_w^2 J(w^o). \tag{26b}$$

Substituting (26a) into (22c) gives

$$\widetilde{\boldsymbol{w}}_i = (I_M - \mu H)\widetilde{\boldsymbol{w}}_{i-1} + \mu \boldsymbol{s}_i(\boldsymbol{w}_{i-1}) + \mu \boldsymbol{c}_{i-1} \tag{26c}$$

in terms of the perturbation term $\boldsymbol{c}_{i-1} \overset{\Delta}{=} \widetilde{\boldsymbol{H}}_{i-1}\widetilde{\boldsymbol{w}}_{i-1}$. Using (25), the norm of this term is easily bounded by

$$\|\boldsymbol{c}_{i-1}\| \le \frac{\tau'}{2}\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 \tag{27a}$$

so that using (21c), we conclude that

$$\limsup_{i\to\infty} \mathbb{E}\|\boldsymbol{c}_{i-1}\| = O(\mu). \tag{27b}$$

We can deduce from this result that $\|\boldsymbol{c}_{i-1}\| = O(\mu)$ asymptotically with *high probability* [16]. To see this, let $r_c = m\mu$, for any constant integer $m \ge 1$. Now, calling upon Markov's inequality [44]–[46], we conclude from (27b) that for $i \gg 1$

$$\begin{aligned} \text{Prob}(\|\boldsymbol{c}_{i-1}\|<r_c) &= 1 - \text{Prob}(\|\boldsymbol{c}_{i-1}\| \ge r_c) \\ &\ge 1 - \frac{\mathbb{E}\|\boldsymbol{c}_{i-1}\|}{r_c} \\ &\overset{(27b)}{\ge} 1 - O(1/m). \end{aligned} \tag{27c}$$

This result shows that the probability of having $\|\boldsymbol{c}_{i-1}\|$ bounded by r_c can be made arbitrarily close to one by selecting a large enough value for m. Once the value for m has been fixed to meet a desired confidence level, then $r_c = O(\mu)$. This analysis, along with recursion (26c), motivate us to assess the mean square performance of the error recursion (22c) by considering instead the following long-term model, which holds with high probability after sufficient iterations $i \gg 1$:

$$\widetilde{\boldsymbol{w}}_i = (I_M - \mu H)\widetilde{\boldsymbol{w}}_{i-1} + \mu \boldsymbol{s}_i(\boldsymbol{w}_{i-1}) + O(\mu^2). \tag{28}$$

Working with iteration (28) is helpful because its dynamics is driven by the constant matrix H as opposed to the random matrix $\boldsymbol{H}_{i-1}$ in the original error recursion (22c). If desired, it can be shown that, under some technical conditions on the fourth-order moment of the gradient noise process, the MSD expression that will result from using (28) is within $O(\mu^{3/2})$ of the actual MSD expression for the original recursion (22c); see [1], [16], and [47] for a formal proof of this fact. Therefore, it is sufficient to rely on the long-term model (28) to obtain performance expressions that are accurate to first order in μ. Fig. 1 provides a block-diagram representation for (28). We will compare this diagram later to the representation shown in Fig. 6 for the case of adaptation and learning over networks.

Before explaining how model (28) can be used to assess the MSD, we remark that there is a second useful metric for evaluating the performance of stochastic-gradient

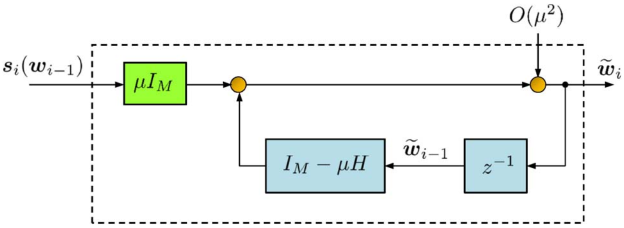

Fig. 1. ***Block-diagram representation of the long-term recursion (28) for single-agent adaptation and learning.***

algorithms. This metric relates to the mean excess cost; which is also called the excess risk (ER) in the machine learning literature [22], [23] and the excess mean square error (EMSE) in the adaptive filtering literature [17]–[19]. We denote it by the letters ER and define it as the average fluctuation of the cost function around its minimum value

$$\text{ER} \overset{\Delta}{=} \lim_{i\to\infty} \mathbb{E}\{J(\boldsymbol{w}_{i-1}) - J(w^o)\}. \tag{29a}$$

Using the smoothness condition (24a), and the mean value theorem [26], [48] again, it can be verified that [1], [16], [47]

$$\text{ER} \overset{\Delta}{=} \lim_{i\to\infty} \mathbb{E}\|\widetilde{\boldsymbol{w}}_{i-1}\|^2_{\frac{1}{2}H} + O\left(\mu^{\frac{3}{2}}\right). \tag{29b}$$

Lemma II.2 (MSE Performance): Assume the conditions under Assumptions II.1, II.2, and II.3 on the cost function and the gradient noise process hold. Assume further that the step size is sufficiently small to ensure mean square stability, as already ascertained by Lemma II.1. Then, the MSD and ER metrics for the stochastic-gradient algorithm (12) are well approximated to first order in μ by

$$\text{MSD} = \frac{\mu}{2}\text{Tr}(H^{-1}R_s) + O\left(\mu^{3/2}\right) \tag{30a}$$

$$\text{ER} = \frac{\mu}{4}\text{Tr}(R_s) + O\left(\mu^{3/2}\right) \tag{30b}$$

where R_s and H are defined by (19b) and (26b).

Proof: We introduce the eigendecomposition $H = U\Lambda U^\top$, where U is orthogonal and Λ is diagonal with positive entries. We can then rewrite (28) in terms of transformed quantities

$$\overline{\boldsymbol{w}}_i = (I - \mu\Lambda)\overline{\boldsymbol{w}}_{i-1} + \mu\overline{\boldsymbol{s}}_i(\boldsymbol{w}_{i-1}) + O(\mu^2) \tag{31a}$$

where $\overline{\boldsymbol{w}}_i = U^\top\widetilde{\boldsymbol{w}}_i$ and $\overline{\boldsymbol{s}}_i(\boldsymbol{w}_{i-1}) = U^\top\boldsymbol{s}_i(\boldsymbol{w}_{i-1})$. Let Σ denote an arbitrary $M \times M$ diagonal matrix with positive entries that we are free to choose. Then, equating the weighted squared norms of both sides of (31a) and taking expectations gives for $i \gg 1$

$$\mathbb{E}\|\overline{\boldsymbol{w}}_i\|^2_\Sigma = \mathbb{E}\|\overline{\boldsymbol{w}}_{i-1}\|^2_{\Sigma'} + \mu^2\mathbb{E}\|\overline{\boldsymbol{s}}_i(\boldsymbol{w}_{i-1})\|^2_\Sigma + O\left(\mu^{5/2}\right) \tag{31b}$$

where

$$\Sigma' \overset{\Delta}{=} (I - \mu\Lambda)\Sigma(I - \mu\Lambda) = \Sigma - 2\mu\Lambda\Sigma + O(\mu^2). \tag{31c}$$

From (17b), (19b), (21c), and (24b) we obtain

$$\lim_{i\to\infty} \mathbb{E}\|\overline{\boldsymbol{s}}_i(\boldsymbol{w}_{i-1})\|^2_\Sigma = \text{Tr}(U\Sigma U^\top R_s) + O(\mu^{\kappa/2}). \tag{31d}$$

Therefore, substituting into (31b) gives for $i \to \infty$

$$\lim_{i\to\infty} \mathbb{E}\|\overline{\boldsymbol{w}}_i\|^2_{2\Lambda\Sigma} = \mu\text{Tr}(U\Sigma U^\top R_s) + O\left(\mu^{3/2}\right). \tag{31e}$$

Since we are free to choose Σ, we let $\Sigma = (1/2)\Lambda^{-1}$ and arrive at (30a) since $\|\overline{\boldsymbol{w}}_i\|^2 = \|\widetilde{\boldsymbol{w}}_i\|^2$ and $U\Sigma U^\top = (1/2)H^{-1}$. On the other hand, selecting $\Sigma = (1/4)I_M$ leads to (30b). ■

The examples that follow show how expressions (30a) and (30b) can be used to recover classical results for MSE adaptation and learning.

Example 5 (Performance of LMS Adaptation): We reconsider the LMS recursion (13f). We know from Example 4 and (18a) that this situation corresponds to $H = 2R_u$ and $R_s = 4\sigma_v^2 R_u$. Substituting into (30a) and (30b) leads to the following well-known expressions for the MSD and EMSE of the LMS filter (see [17]–[19] and [49]–[54]):

$$\text{MSD} \approx \mu M\sigma_v^2 = O(\mu) \tag{32a}$$

$$\text{EMSE} \approx \mu\sigma_v^2\text{Tr}(R_u) = O(\mu) \tag{32b}$$

where here, and elsewhere, we will be using the symbol $\approx$ to indicate that we are ignoring higher order terms in μ. ♦

Example 6 (Performance of Online Learners): Consider a standalone learner receiving a streaming sequence of independent data vectors $\{\boldsymbol{x}_i, i \geq 0\}$ that arise from some fixed probability distribution $\mathcal{X}$. The goal is to learn vector w^o that optimizes some ν-strongly convex risk function $J(w)$ defined in terms of a loss function [55], [56]

$$w^o \overset{\Delta}{=} \arg\min_w J(w) = \arg\min_w \mathbb{E}Q(w; \boldsymbol{x}_i). \tag{33a}$$

The learner seeks w^o by running the stochastic-gradient algorithm

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} - \mu\nabla_{w^\top}Q(\boldsymbol{w}_{i-1}; \boldsymbol{x}_i), \qquad i \geq 0 \tag{33b}$$

so that the gradient noise vector is given by

$$\boldsymbol{s}_i(\boldsymbol{w}_{i-1}) = \nabla_{w^\top}Q(\boldsymbol{w}_{i-1}; \boldsymbol{x}_i) - \nabla_{w^\top}J(\boldsymbol{w}_{i-1}). \tag{33c}$$

Since $\nabla_w J(w^o) = 0$, and since the distribution of $\boldsymbol{x}_i$ is stationary, it follows that the covariance matrix of $\boldsymbol{s}_i(w^o)$ is constant and given by $R_s = \mathbb{E}\nabla_{w^\top} Q(w^o;\boldsymbol{x}_i)\nabla_w Q(w^o;\boldsymbol{x}_i)$. The ER measure that will result from this stochastic implementation is then given by (30b) so that $\text{ER} = O(\mu)$. ♦

III. CENTRALIZED ADAPTATION AND LEARNING

The discussion in Section II establishes the mean square stability of *standalone* adaptive agents for small step sizes (Lemma II.1), and provides expressions for their MSD and ER metrics (Lemma II.2). We now examine two situations involving a multitude of similar agents. In the first scenario, each agent senses data and analyzes it independently of the other agents. We refer to this mode of operation as noncooperative processing. In the second scenario, the agents transmit the collected data for processing at a fusion center. We refer to this mode of operation as centralized or batch processing. We motivate the discussion by considering first the case of MSE costs. Subsequently, we extend the results to more general costs.

A. Noncooperative MSE Processing

Thus, consider *separate* agents, labeled $k = 1, 2, \ldots, N$. Each agent k receives streaming data $\{\boldsymbol{d}_k(i), \boldsymbol{u}_{k,i},\ i \geq 0\}$, where we are using the subscript k to index the data at agent k. We assume that the data at each agent satisfy the same statistical properties as in Example 3, and the same linear regression model (13a) with a common w^o, albeit with noise $\boldsymbol{v}_k(i)$. We denote the statistical moments of the data at agent k by $R_{u,k} = \mathbb{E}\boldsymbol{u}_{k,i}^\top \boldsymbol{u}_{k,i} > 0$ and $\sigma_{v,k}^2 = \mathbb{E}\boldsymbol{v}_k^2(i)$. We further assume in this motivating example that the $R_{u,k}$ are uniform across the agents so that $R_{u,k} \equiv R_u$ for all $k = 1, 2, \ldots, N$. In this way, the MSE cost $J_k(w) = \mathbb{E}(\boldsymbol{d}_k(i) - \boldsymbol{u}_{k,i}\boldsymbol{w})^2$, which is associated with agent k, will satisfy a condition similar to (9) with the corresponding parameters $\{\nu, \delta\}$ given by [cf. (10a)]

$$\nu = 2\lambda_{\min}(R_u), \qquad \delta = 2\lambda_{\max}(R_u). \tag{34a}$$

Now, if each agent runs the LMS learning rule (13f) to estimate w^o on its own, then, according to (32a), each agent k will attain an individual MSD level that is given by

$$\text{MSD}_{\text{ncop},k} \approx \mu M \sigma_{v,k}^2, \qquad k = 1, 2, \ldots, N. \tag{34b}$$

Moreover, according to (21b), agent k will converge toward this level at a rate dictated by

$$\alpha_{\text{ncop},k} \approx 1 - 4\mu\lambda_{\min}(R_u). \tag{34c}$$

The subscript "ncop" is used in (34b) and (34c) to indicate that these expressions are for the noncooperative mode of operation. It is seen from (34b) that agents with noisier data (i.e., larger $\sigma_{v,k}^2$) will perform worse and have larger MSD levels than agents with cleaner data. In other words, whenever adaptive agents act individually, the quality of their solution will be as good as the quality of their noisy data. This is a sensible conclusion. We are going to show later in Section VII that cooperation among the agents, whereby agents share information with their neighbors, can help enhance their individual performance levels. The analysis will also show that both types of agents can benefit from cooperation: agents with bad data and agents with good data; this is because all data carry information about w^o. However, for these conclusions to hold, it is necessary for cooperation to be carried out in proper ways; see (68f).

B. Centralized MSE Processing

Let us now contrast the above noncooperative solution with a centralized implementation whereby, at every iteration i, the N agents transmit their raw data $\{\boldsymbol{d}_k(i), \boldsymbol{u}_{k,i}\}$ to a fusion center for processing. One could also consider situations where agents transmit processed data, e.g., as happens with useful techniques for combining adaptive filter outputs [57]. Once the fusion center receives the raw data, we assume it runs a stochastic-gradient update of the form

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} + \mu\left(\frac{1}{N}\sum_{k=1}^{N} 2\boldsymbol{u}_{k,i}^\top\left(\boldsymbol{d}_k(i) - \boldsymbol{u}_{k,i}\boldsymbol{w}_{i-1}\right)\right) \tag{35a}$$

where the term multiplying μ can be seen to correspond to a sample average of several approximate gradient vectors. The analysis in the rest of the paper will show that the MSD performance that results from this implementation is given by [using (40c) with $H_k = 2R_u$ and $R_{s,k} = 4\sigma_{v,k}^2 R_u$]

$$\text{MSD}_{\text{cent}} \approx \mu M \frac{1}{N}\left(\frac{1}{N}\sum_{k=1}^{N}\sigma_{v,k}^2\right). \tag{35b}$$

Moreover, using (39b), this centralized solution will converge toward the above MSD level at the same rate as the noncooperative solution

$$\alpha_{\text{cent}} \approx 1 - 4\mu\lambda_{\min}(R_u). \tag{35c}$$

Observe from (35b) that the MSD level attained by the centralized solution is proportional to $1/N$ times the *average* noise power across all agents. This scaled average noise power can be larger than some of the individual noise variances and smaller than the remaining noise variances. For example, consider a situation with $N = 2$ agents, $\sigma_{v,2}^2 = 5\sigma_v^2$ and $\sigma_{v,1}^2 = \sigma_v^2$. Then

$$\frac{1}{N}\left(\frac{1}{N}\sum_{k=1}^{N}\sigma_{v,k}^2\right) = \frac{1}{2}\left(\frac{1}{2}\left(\sigma_{v,1}^2 + \sigma_{v,2}^2\right)\right) = 1.5\sigma_v^2 \tag{36}$$

which is larger than $\sigma_{v,1}^2$ and smaller than $\sigma_{v,2}^2$. In this case, the centralized solution performs better than noncooperative agent 2 (i.e., leads to a smaller MSD) but worse than noncooperative agent 1. This example shows that it does not generally hold that centralized stochastic-gradient implementations outperform all individual noncooperative agents [58].

C. Stochastic-Gradient Centralized Solution

Sections II-A and B focused on MSE adaptation. We next extend the conclusions to more general costs. Thus, consider a collection of N agents, each with an individual convex cost function, $J_k(w)$. The objective is to determine the unique minimizer w^o of the aggregate cost

$$J^{\text{glob}}(w) \triangleq \sum_{k=1}^{N} J_k(w). \tag{37}$$

It is now the above aggregate cost $J^{\text{glob}}(w)$ that will be required to satisfy the conditions of Assumptions II.1 and II.3 relative to some parameters $\{\nu_c, \delta_c, \tau_c\}$, with the subscript "$c$" used to indicate that these factors correspond to the centralized implementation. Under these conditions, the cost $J^{\text{glob}}(w)$ will have a unique minimizer, which we continue to denote by w^o. We will not be requiring each individual cost $J_k(w)$ to be strongly convex any longer. It is sufficient for at least one of these costs to be strongly convex while the remaining costs can be simply convex; this condition ensures the strong convexity of $J^{\text{glob}}(w)$. Moreover, minimizers of the individual costs $\{J_k(w)\}$ need not coincide with each other or with w^o; we will write w_k^o to refer to a minimizer of $J_k(w)$.

There are many centralized solutions that can be used to determine the unique minimizer w^o of (37), with some solution techniques being more powerful than other techniques. Nevertheless, we will focus on centralized implementations of the stochastic-gradient type. The reason we consider the *same* class of stochastic-gradient algorithms for noncooperative, centralized, and distributed solutions in this paper is to enable a *meaningful* comparison among the various implementations. Thus, we consider a centralized strategy of the following form:

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} - \frac{\mu}{N}\sum_{k=1}^{N} \widehat{\nabla_{w^\top} J_k}(\boldsymbol{w}_{i-1}), \qquad i \geq 0 \tag{38a}$$

where the sum multiplying μ/N is seen to be an approximation for the true gradient vector of $J^{\text{glob}}(w)$; the scaling of μ by N in (38a) is meant to ensure similar convergence rates for the noncooperative and centralized solutions; see (40b). We can express the gradient noise that corresponds to implementation (38a) in terms of the following individual gradient noise processes:

$$\boldsymbol{s}_{k,i}(\boldsymbol{w}_{i-1}) \triangleq \widehat{\nabla_{w^\top} J_k}(\boldsymbol{w}_{i-1}) - \nabla_{w^\top} J_k(\boldsymbol{w}_{i-1}) \tag{38b}$$

for $k = 1, 2, \ldots, N$. Using these gradient terms, it is straightforward to verify that the overall gradient noise corresponding to (38a) is given by

$$\boldsymbol{s}_i(\boldsymbol{w}_{i-1}) = \sum_{k=1}^{N} \boldsymbol{s}_{k,i}(\boldsymbol{w}_{i-1}). \tag{38c}$$

Since the centralized iteration (38a) has the form of a stochastic-gradient recursion, we can infer its MSE behavior from Lemmas II.1 and II.2 if the individual noise processes in (38b) satisfy conditions similar to Assumption II.2 with parameters $\{\beta_k^2, \sigma_{s,k}^2, R_{s,k}\}$ and when, additionally, the gradient noise components across the agents are uncorrelated with each other:

$$\mathbb{E}\left[\boldsymbol{s}_{k,i}(\boldsymbol{w}_{i-1})\boldsymbol{s}_{\ell,i}^\top(\boldsymbol{w}_{i-1})|\boldsymbol{\mathcal{F}}_{i-1}\right] = 0, \qquad \text{all } k \neq \ell. \tag{38d}$$

Let

$$\beta_c^2 \triangleq \sum_{k=1}^{N} \beta_k^2, \qquad \sigma_s^2 \triangleq \sum_{k=1}^{N} \sigma_{s,k}^2. \tag{38e}$$

The following result follows from Lemmas II.1 and II.2 [1].

Lemma III.1 (Convergence of Centralized Solution): Assume the aggregate cost (37) satisfies the conditions under Assumption II.1 for some parameters $0 < \nu_c \leq \delta_c$. Assume also that the individual gradient noise processes defined by (38b) satisfy the conditions under Assumption II.2 for some parameters $\{\beta_k^2, \sigma_{s,k}^2, R_{s,k}\}$, in addition to the orthogonality condition (38d). Let $\mu_o = 2\nu_c/\ (\delta_c^2 + \beta_c^2)$. For any μ such that $\mu/N < \mu_o$, it holds that

$$\mathbb{E}\|\widetilde{\boldsymbol{w}}_i\|^2 \leq \alpha \mathbb{E}\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + \left(\frac{\mu}{N}\right)^2 \sigma_s^2 \tag{39a}$$

where the scalar α satisfies $0 \leq \alpha < 1$ and is given by

$$\alpha = 1 - 2\nu_c\left(\frac{\mu}{N}\right) + \left(\delta_c^2 + \beta_c^2\right)\left(\frac{\mu}{N}\right)^2. \tag{39b}$$

It follows from (39a) that for sufficiently small step sizes

$$\limsup_{i\to\infty}\ \mathbb{E}\|\widetilde{\boldsymbol{w}}_i\|^2 = O(\mu). \tag{39c}$$

Moreover, under smoothness conditions similar to (24a) for $J^{\text{glob}}(w)$ for some parameter $\tau_c \geq 0$, and similar to

(24b) for the individual gradient noise covariance matrices, it holds for small μ that:

$$\mathrm{MSD}_{\mathrm{cent}} = \frac{\mu}{2N}\mathrm{Tr}\left[\left(\sum_{k=1}^{N} H_k\right)^{-1}\left(\sum_{k=1}^{N} R_{s,k}\right)\right] + O\left(\mu^{3/2}\right) \quad \text{(39d)}$$

where $H_k = \nabla_w^2\, J_k(w^o)$. ♦

D. Comparison With Noncooperative Processing

We can now compare the performance of the centralized solution (38a) to that of noncooperative processing where agents act independently of each other and run the recursion

$$\boldsymbol{w}_{k,i} = \boldsymbol{w}_{k,i-1} - \mu\widehat{\nabla_{w^\top} J_k}(\boldsymbol{w}_{k,i-1}), \qquad i \geq 0. \quad \text{(40a)}$$

This comparison is *meaningful* only when all agents share the same minimizer, i.e., when $w_k^o = w^o$ for $k = 1, 2, \ldots, N$, so that we can compare how well the individual agents are able to recover the same w^o as the centralized solution. For this reason, we need to reintroduce in this section only the requirement that all individual costs $\{J_k(w)\}$ are ν-strongly convex with a uniform parameter ν. Since $J^{\mathrm{glob}}(w)$ is the aggregate sum of the individual costs, then we can set the lower bound ν_c for the Hessian of $J^{\mathrm{glob}}(w)$ at $\nu_c = N\nu$. From expressions (21b) and (39b), we then conclude that, for a sufficiently small μ, the convergence rates of the noncooperative and centralized solutions will be similar

$$\alpha_{\mathrm{cent}} \approx 1 - 2\nu_c\left(\frac{\mu}{N}\right) = 1 - 2\nu\mu \approx \alpha_{\mathrm{ncop},k}. \quad \text{(40b)}$$

Moreover, we observe from (30a) that the average MSD level across N noncooperative agents is given by

$$\mathrm{MSD}_{\mathrm{ncop,av}} = \frac{\mu}{2N}\mathrm{Tr}\left[\sum_{k=1}^{N} H_k^{-1} R_{s,k}\right] + O\left(\mu^{3/2}\right) \quad \text{(40c)}$$

so that comparing with (39d), some simple algebra allows us to conclude that, for small step sizes and to first order in μ

$$\mathrm{MSD}_{\mathrm{cent}} \leq \mathrm{MSD}_{\mathrm{ncop,av}}. \quad \text{(40d)}$$

That is, while the centralized solution need not outperform every individual noncooperative agent in general, its performance outperforms the average performance across all noncooperative agents. The next example illustrates the above result by considering the scenario where all agents have the same Hessian matrices at $w = w^o$, namely, $H_k \equiv H$ for $k = 1, 2, \ldots, N$. This situation occurs, for example, when the individual costs are identical across the agents, say, $J_k(w) \equiv J(w)$, as is common in machine learning applications. This situation also occurs for the MSE costs we considered earlier in this section when the regression covariance matrices $\{R_{u,k}\}$ are uniform across all agents, i.e., $R_{u,k} \equiv R_u$ for $k = 1, 2, \ldots, N$. In these cases, where the Hessian matrices H_k are uniform, the example below establishes that the centralized solution actually improves over the average MSD performance of the noncooperative solution by a factor of N.

Example 7 (N-Fold Improvement in Performance): Consider a collection of N agents whose individual cost functions $J_k(w)$ are ν-strongly convex and are minimized at the same location $w = w^o$. The costs are also assumed to have identical Hessian matrices at $w = w^o$, i.e., $H_k \equiv H$. Then, using (39d), the MSD of the centralized implementation is given by

$$\mathrm{MSD}_{\mathrm{cent}} \approx \frac{1}{N}\left(\frac{\mu}{2N}\sum_{k=1}^{N}\mathrm{Tr}(H^{-1}R_{s,k})\right) \overset{(40c)}{\approx} \frac{1}{N}\mathrm{MSD}_{\mathrm{ncop,av}}. \quad \text{(41)}$$

♦

Example 8 (Fully Connected Networks): In preparation for the discussion on networked agents, it is useful to describe one extreme situation where a collection of N agents are fully connected to each other; see Fig. 2. In this case, each agent is able to access the data from all other agents and, therefore, each agent can run a centralized implementation of the same form as (38a), namely

$$\boldsymbol{w}_{k,i} = \boldsymbol{w}_{k,i-1} - \frac{\mu}{N}\sum_{\ell=1}^{N}\widehat{\nabla_{w^\top} J_\ell}(\boldsymbol{w}_{k,i-1}), \qquad i \geq 0. \quad \text{(42)}$$

When this happens, each agent will attain the same performance level as that of the centralized solution. Two observations are in place. First, note from (42) that the information that agent k is receiving from all other agents is their gradient vector approximations. Obviously, other pieces of information could be shared among the agents, such as their iterates $\{\boldsymbol{w}_{\ell,i-1}\}$. Second, note that the rightmost term multiplying μ in (42) corresponds to a convex combination of the approximate gradients from the various agents, with the combination coefficients being uniform and all equal to $1/N$. In general, there is no need for these combination weights to be identical. Even more importantly, agents do not need to have access to information from all other agents in the network. We are going to see in the rest of the paper that interactions with a limited number of neighbors are sufficient for the agents to

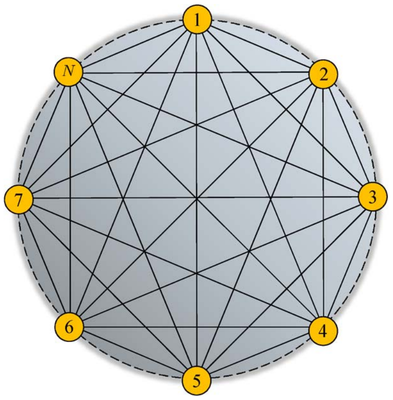

Fig. 2. *Example of a fully connected network, where each agent can access information from all other agents.*

attain performance that is comparable to that of the centralized solution.

Fig. 3 shows a sample selection of connected topologies for five agents. The leftmost panel corresponds to the noncooperative case and the rightmost panel corresponds to the fully connected case. The panels in between illustrate some other topologies. In the coming sections, we are going to present results that allow us to answer useful questions about such networked agents such as the following. 1) Which topology has best performance in terms of mean square error and convergence rate? 2) Given any connected topology, can it be made to approach the performance of the centralized solution? 3) Which aspects of the topology influence performance? 4) Which aspects of the combination weights (policy) influence performance? 5) Can different topologies deliver similar performance levels? 6) Is cooperation always beneficial? 7) If the individual agents are able to solve the inference task individually in a stable manner, does it follow that the connected network will remain stable regardless of the topology and regardless of the cooperation strategy? ♦

IV. MULTIAGENT NETWORKS

In the next section, we will describe several distributed strategies that rely solely on localized interactions among neighboring agents and that are resilient to failure. In preparation for that discussion, we first describe the network model.

A. Strongly Connected Networks

Fig. 4 shows an example of a network consisting of N connected agents, labeled $k = 1, 2, \ldots, N$. Following the presentation from [1] and [11], the network is represented by a graph consisting of N vertices (representing the agents) and a set of edges connecting the agents to each other. An edge that connects an agent to itself is called a self-loop. The neighborhood of an agent k is denoted by $\mathcal{N}_k$ and it consists of all agents that are connected to k by an edge. Any two neighboring agents k and ℓ have the ability to share information over the edge connecting them.

We assume an undirected graph so that if agent k is a neighbor of agent ℓ, then agent ℓ is also a neighbor of agent k. We assign a pair of nonnegative scaling weights $\{a_{k\ell}, a_{\ell k}\}$ to the edge connecting k and ℓ. The scalar $a_{\ell k}$ is used by agent k to scale the data it receives from agent ℓ; this scaling can be interpreted as a measure of the confidence level that agent k assigns to its interaction with agent ℓ. Likewise, $a_{k\ell}$ is used by agent ℓ to scale the data it receives from agent k. The weights $\{a_{k\ell}, a_{\ell k}\}$ can be different so that the exchange of information between the neighboring agents $\{k, \ell\}$ need not be symmetrical. One or both weights can also be zero.

A network is said to be connected if paths with nonzero scaling weights can be found linking any two distinct agents in *both* directions, either directly when they are neighbors or by passing through intermediate agents when they are not neighbors. In this way, information can flow in both directions between any two agents in the network,

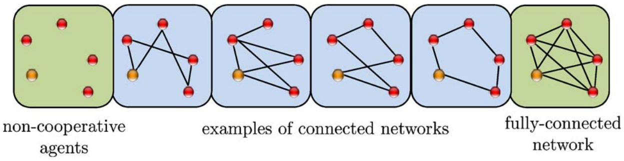

Fig. 3. *Examples of connected networks, with the leftmost panel representing a collection of noncooperative agents.*

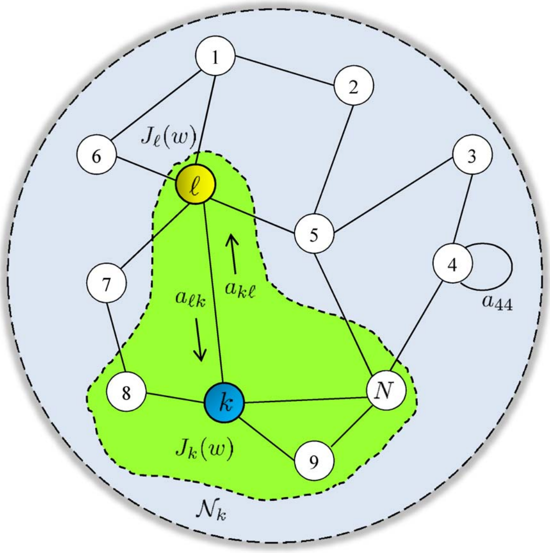

Fig. 4. *Agents that are linked by edges can share information. The neighborhood of agent k is marked by the highlighted area.*

although the forward path from an agent k to some other agent ℓ need not be the same as the backward path from ℓ to k. A strongly connected network is a connected network with at least one nontrivial self-loop, meaning that $a_{kk} > 0$ for some agent k.

The strong connectivity of a network translates into a useful property on the combination weights. Assume we collect the coefficients $\{a_{\ell k}\}$ into an $N \times N$ matrix $A = [a_{\ell k}]$, such that the entries on the kth column of A contain the coefficients used by agent k to scale data arriving from its neighbors $\ell \in \mathcal{N}_k$; we set $a_{\ell k} = 0$ if $\ell \notin \mathcal{N}_k$. We refer to A as the *combination* matrix or policy. It turns out that combination matrices that correspond to strongly connected networks are *primitive*—an $N \times N$ matrix A with nonnegative entries is said to be primitive if there exists some finite integer $n_o > 0$ such that all entries of A^{n_o} are strictly positive [1], [11], [59].

B. Distributed Optimization

Network cooperation can be exploited to solve adaptation, learning, and optimization problems in a decentralized manner in response to streaming data. To explain how cooperation can be achieved, we start by associating with each agent k a twice-differentiable cost function $J_k(w) : \mathbb{R}^{M\times 1} \mapsto \mathbb{R}$. The objective of the network of agents is still to seek the unique minimizer of the aggregate cost function $J^{\text{glob}}(w)$, defined by (37). Now, however, we seek a *distributed* (as opposed to a centralized) solution. In a distributed implementation, each agent k can only rely on its own data and on data from its neighbors. We continue to assume that $J^{\text{glob}}(w)$ satisfies the conditions of Assumptions II.1 and II.3 with parameters $\{\nu_d, \delta_d, \tau_d\}$, with the subscript "$d$" now used to indicate that these parameters are related to the distributed implementation. Under these conditions, the cost $J^{\text{glob}}(w)$ will have a unique minimizer, which we continue to denote by w^o. We will not be requiring the individual costs $J_k(w)$ to be strongly convex. As mentioned earlier, it is sufficient to assume that at least one of these costs is strongly convex while the remaining costs are simply convex; this condition ensures that $J^{\text{glob}}(w)$ will be strongly convex. We also assume that each $J_k(w)$ satisfies the smoothness condition of Assumption II.3 with parameter $\tau_{k,d}$.

The individual costs $\{J_k(w)\}$ can be distinct across the agents or they can all be identical, i.e., $J_k(w) \equiv J(w)$ for $k = 1, 2, \ldots, N$; in the latter situation, the problem of minimizing (37) would correspond to the case in which the agents work together to optimize the same cost function. Moreover, when they exist, the minimizers of the individual costs $\{J_k(w)\}$ need not coincide with each other or with w^o; we will again write w_k^o to refer to a minimizer of $J_k(w)$. There are important situations in practice where all minimizers $\{w_k^o\}$ coincide with each other. For instance, examples abound where agents need to work cooperatively to attain a common objective such as tracking a target, locating a food source, or evading a predator (see, e.g., [10], [60], and [61]). This scenario is also common in machine learning problems [22], [23], [62]–[64] when data samples at the various agents are generated by a common distribution parameterized by some vector w^o. One such situation is illustrated in the next example.

Example 9 (MSE Networks): Consider the same setting of Example 3 except that we now have N agents observing streaming data $\{\boldsymbol{d}_k(i), \boldsymbol{u}_{k,i}\}$ that satisfy the regression model (13a) with regression covariance matrices $R_{u,k} = \mathbb{E}\boldsymbol{u}_{k,i}^{\top}\boldsymbol{u}_{k,i} > 0$ and with the same unknown w^o, i.e.,

$$\boldsymbol{d}_k(i) = \boldsymbol{u}_{k,i} w^o + \boldsymbol{v}_k(i). \tag{43a}$$

The individual MSE costs are defined by $J_k(w) = \mathbb{E}(\boldsymbol{d}_k(i) - \boldsymbol{u}_{k,i}w)^2$ and are strongly convex in this case, with the minimizer of each $J_k(w)$ occurring at

$$w_k^o = R_{u,k}^{-1} r_{du,k}, \qquad k = 1, 2, \ldots, N. \tag{43b}$$

If we multiply both sides of (43a) by $\boldsymbol{u}_{k,i}^{\top}$ from the left, and take expectations, we find that w^o satisfies $r_{du,k} = R_{u,k}w^o$. This relation shows that the unknown w^o from (43a) satisfies the same expression as w_k^o in (43b), for any $k = 1, 2, \ldots, N$, so that we must have $w^o = w_k^o$. Therefore, this example amounts to a situation where all costs $\{J_k(w)\}$ attain their minima at the same location, w^o.

We will use the network model of this example to illustrate other results in the paper. For ease of reference, we will refer to strongly connected networks with agents receiving data according to model (43a) and seeking to estimate w^o by adopting the MSE costs $J_k(w)$ defined above, as *MSE networks*. We assume for these networks that the measurement noise process $\boldsymbol{v}_k(i)$ is temporally white and independent over space so that $\mathbb{E}\boldsymbol{v}_k(i)\boldsymbol{v}_\ell(j) = \sigma_{v,k}^2 \delta_{k,\ell}\delta_{i,j}$ in terms of the Kronecker delta sequence $\delta_{m,n}$. Likewise, we assume that the regression data $\boldsymbol{u}_{k,i}$ is temporally white and independent over space so that $\mathbb{E}\boldsymbol{u}_{k,i}^{\top}\boldsymbol{u}_{\ell,j} = R_{u,k}\delta_{k,\ell}\delta_{i,j}$. Moreover, the measurement noise $\boldsymbol{v}_k(i)$ and the regression data $\boldsymbol{u}_{\ell,j}$ are independent of each other for all k, ℓ, i, j. These statistical conditions help facilitate the analysis of such networks. ♦

V. MULTIAGENT ADAPTATION AND LEARNING

There are several distributed strategies that can be used to seek the minimizer of (37). In this section, we describe three prominent strategies that are based on incremental (e.g., [68]–[80]), consensus (e.g., [11], [81], [83]–[88], and [90]–[96]), and diffusion (e.g., [10], [11], [32], and [97]–[99]) techniques. We motivate the algorithms by employing stochastic-approximation arguments.

A. Incremental Strategy

We refer back to the centralized algorithm (38a) and use it to motivate the incremental strategy as follows. Starting from a connected network, we first determine a *cyclic* trajectory that visits all agents in the network in succession, one after the other. To facilitate the description of this construction, once a cycle has been identified, we renumber the agents along the trajectory from 1 to N. Then, at each iteration i, the centralized update (38a) can be split into N consecutive *incremental* steps, with each step performed locally at one of the agents

$$\begin{cases} \boldsymbol{w}_{1,i} = \boxed{\boldsymbol{w}_{i-1}} - \frac{\mu}{N}\widehat{\nabla_{w^\top} J_1}(\boldsymbol{w}_{i-1}) \\ \boldsymbol{w}_{2,i} = \boldsymbol{w}_{1,i} - \frac{\mu}{N}\widehat{\nabla_{w^\top} J_2}(\boldsymbol{w}_{i-1}) \\ \quad\vdots = \vdots \\ \boxed{\boldsymbol{w}_i} = \boldsymbol{w}_{N-1,i} - \frac{\mu}{N}\widehat{\nabla_{w^\top} J_N}(\boldsymbol{w}_{i-1}). \end{cases} \tag{44a}$$

In this implementation, information is passed from one agent to the next over the cyclic path until all agents are visited and the process is then repeated. Each agent k receives an intermediate variable, denoted by $\boldsymbol{w}_{k-1,i}$, from its predecessor agent $k-1$, incrementally adds one term from the gradient sum in (38a) to this variable, and computes $\boldsymbol{w}_{k,i}$

$$\boldsymbol{w}_{k,i} = \boldsymbol{w}_{k-1,i} - \frac{\mu}{N}\widehat{\nabla_{w^\top} J_k}(\boldsymbol{w}_{i-1}). \tag{44b}$$

At the end of the cycle of N steps in (44a), the iterate $\boldsymbol{w}_{N,i}$ that is computed by agent N coincides with the desired iterate $\boldsymbol{w}_i$ that would have resulted from (38a).

Implementation (44a) is still *not* distributed since it requires all agents to have access to the *global* iterate $\boldsymbol{w}_{i-1}$ to evaluate the gradient approximations. At this point, we can resort to a useful incremental substitution, which has been widely studied in the literature (see, e.g., [68]–[74] and [76]–[80]). According to this substitution, each agent k replaces the unavailable global variable $\boldsymbol{w}_{i-1}$ in (44b) by the incremental variable it receives from its predecessor, and which we denoted by $\boldsymbol{w}_{k-1,i}$. In this way, the update operation (44b) at each agent is replaced by

$$\boldsymbol{w}_{k,i} = \boldsymbol{w}_{k-1,i} - \frac{\mu}{N}\widehat{\nabla_{w^\top} J_k}(\boldsymbol{w}_{k-1,i}), \qquad i \geq 0 \tag{44c}$$

using the fictitious boundary condition $\boldsymbol{w}_{o,i} = \boldsymbol{w}_{i-1}$ and setting $\boldsymbol{w}_i = \boldsymbol{w}_{N,i}$ at the end of the cycle. The resulting incremental strategy (44c) is now fully distributed.

Example 10 (Incremental LMS Networks): For the MSE network of Example 9, the incremental strategy (44c) reduces to

$$\boldsymbol{w}_{k,i} = \boldsymbol{w}_{k-1,i} + \frac{2\mu}{N}\boldsymbol{u}_{k,i}^\top[\boldsymbol{d}_k(i) - \boldsymbol{u}_{k,i}\boldsymbol{w}_{k-1,i}], \qquad i \geq 0. \tag{45}$$

For sufficiently small step sizes, the performance of this incremental LMS strategy is comparable to the performance of the centralized LMS strategy (35a); see, e.g., [79, eqs. (54) and (55)], as well as the results in [58] and [76]–[78]. More specifically, for the case $R_{u,k} \equiv R_u$, the MSD performance and the convergence rate can be well approximated by the same expressions (35b) and (35c). ♦

The incremental solution (44c) suffers from a number of drawbacks for real-time adaptation and learning over networks. First, the incremental strategy is sensitive to agent or link failures. Second, determining a cyclic path that visits all agents is generally an NP-hard problem [100]. Third, cooperation between agents is limited with each agent receiving data from one preceding agent and sharing data with one successor agent. Fourth, for every iteration i, it is necessary to perform N incremental steps and to visit all agents in order to update $\boldsymbol{w}_{i-1}$ to $\boldsymbol{w}_i$ before the next cycle begins.

We next motivate two other distributed strategies based on consensus and diffusion techniques that do not suffer from these limitations. These techniques take advantage of the following flexibility. First, there is no reason why agents should only receive information from one neighbor at a time and pass information to only one other neighbor. Second, there is also no reason why the global variable $\boldsymbol{w}_{i-1}$ in (44b) cannot be replaced by some other choice, other than $\boldsymbol{w}_{k-1,i}$, to attain decentralization. Third, there is no reason why agents cannot adapt and learn simultaneously with other agents rather than wait for each cycle to complete.

B. Consensus Strategy

The iterate $\boldsymbol{w}_{k-1,i}$ appears twice on the right-hand side of the incremental update (44c). The first $\boldsymbol{w}_{k-1,i}$ represents the information that agent k receives from its preceding agent in the incremental implementation. In the consensus strategy, this term is replaced by a convex combination of the iterates that are available at the neighbors of agent k; see the first term on the right-hand side of (46a). With regards to the second $\boldsymbol{w}_{k-1,i}$ on the right-hand side of (44c), it will now be replaced by $\boldsymbol{w}_{k,i-1}$; this quantity is the iterate that is already available at agent k. In this manner, the consensus iteration at each agent k is then given by

$$\boldsymbol{w}_{k,i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k}\boldsymbol{w}_{\ell,i-1} - \mu_k\widehat{\nabla_{w^\top} J_k}(\boldsymbol{w}_{k,i-1}) \tag{46a}$$

where we are further replacing the step size μ/N in the incremental implementation (44c) by μ_k in the consensus implementation (46a) and allowing it to be agent dependent for generality (since, as we are going to see, each agent can now run its update simultaneously with the other agents). Furthermore, the combination coefficients $\{a_{\ell k}\}$ that appear in (46a) are nonnegative scalars that satisfy the following conditions for each agent $k = 1, 2, \ldots, N$:

$$a_{\ell k} \geq 0, \quad \sum_{\ell=1}^{N} a_{\ell k} = 1, \quad \text{and} \quad a_{\ell k} = 0, \quad \text{if } \ell \notin \mathcal{N}_k. \tag{46b}$$

Condition (46b) implies that the combination matrix $A = [a_{\ell k}]$ satisfies $A^\top \mathbb{1} = \mathbb{1}$, where $\mathbb{1}$ denotes the vector with all entries equal to one. We say that A is left stochastic. One useful property of left-stochastic matrices is that their spectral radius is equal to one, $\rho(A) = 1$ [11], [59], [101]–[103]. An equivalent representation that is useful for later analysis is to rewrite the consensus iteration (46a), as shown in the following listing, where the intermediate iterate that results from the neighborhood combination is denoted by $\boldsymbol{\psi}_{k,i-1}$. Observe that the gradient vector in the consensus implementation (46c) is evaluated at $\boldsymbol{w}_{k,i-1}$ and not $\boldsymbol{\psi}_{k,i-1}$.

Consensus Strategy for Distributed Adaptation

for each time instant $i \geq 0$:
each agent $k = 1, 2, \ldots, N$ performs the update

$$\begin{cases} \boldsymbol{\psi}_{k,i-1} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \boldsymbol{w}_{\ell,i-1} \\ \boldsymbol{w}_{k,i} = \boldsymbol{\psi}_{k,i-1} - \mu_k \widehat{\nabla_{w^\top} J}_k(\boldsymbol{w}_{k,i-1}) \end{cases} \tag{46c}$$

end

The consensus update (46a) can also be motivated by starting from the noncooperative step (40a) and replacing the first iterate $\boldsymbol{w}_{k,i-1}$ by the same neighborhood convex combination used in (46a). Note further that, in the consensus implementation, the information that is used by agent k from its neighbors consists of the iterates $\{\boldsymbol{w}_{\ell,i-1}\}$, and these iterates are *already* available for use from the previous iteration $i-1$. As such, there is *no* need to cycle through the agents. At every iteration i, all agents in the network can run their consensus update (46a) or (46c) *simultaneously*. Accordingly, there is no need any longer to select beforehand a cyclic trajectory or to renumber the agents, as was the case with the incremental strategy.

Example 11 (Consensus LMS Networks): For the MSE network of Example 9, the above consensus strategy reduces to

$$\begin{cases} \boldsymbol{\psi}_{k,i-1} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \boldsymbol{w}_{\ell,i-1} \\ \boldsymbol{w}_{k,i} = \boldsymbol{\psi}_{k,i-1} + 2\mu_k \boldsymbol{u}_{k,i}^\top \left[\boldsymbol{d}_k(i) - \boldsymbol{u}_{k,i} \boldsymbol{w}_{k,i-1}\right]. \end{cases} \tag{47}$$

♦

C. Diffusion Strategies

If we compare the consensus implementation (46a) with the incremental implementation (44c), we observe that consensus replaces the two instances of $\boldsymbol{w}_{k-1,i}$ on the right-hand side of (44c) by two different substitutions, namely, by $\boldsymbol{\psi}_{k,i-1}$ and $\boldsymbol{w}_{k,i-1}$. This *asymmetry* in the consensus construction will be shown in Examples 22 and 23 to be problematic when the strategy is used for adaptation and learning over networks. This is because the asymmetry can cause an unstable growth in the state of the network [99]. Diffusion strategies remove the asymmetry problem.

Combine-Then-Adapt (CTA) Diffusion: In the CTA formulation of the diffusion strategy, the *same* iterate $\boldsymbol{\psi}_{k,i-1}$ is used to replace the two instances of $\boldsymbol{w}_{k-1,i}$ on the right-hand side of the incremental implementation (44c), thus leading to description (48a) where the gradient vector is evaluated at $\boldsymbol{\psi}_{k,i-1}$ as well. The reason for the name "combine-then-adapt" is that the first step in (48a) involves a combination step, while the second step involves an adaptation step. The reason for the qualification "diffusion" is that the use of $\boldsymbol{\psi}_{k,i-1}$ to evaluate the gradient vector allows information to diffuse more thoroughly through the network. This is because information is not only being diffused through the aggregation of the neighborhood iterates, but also through the evaluation of the gradient vector at the aggregate state value.

Diffusion Strategy for Distributed Adaptation (CTA)

for each instant $i \geq 0$:
each agent $k = 1, 2, \ldots, N$ performs the update

$$\begin{cases} \boldsymbol{\psi}_{k,i-1} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \boldsymbol{w}_{\ell,i-1} \\ \boldsymbol{w}_{k,i} = \boldsymbol{\psi}_{k,i-1} - \mu_k \widehat{\nabla_{w^\top} J}_k(\boldsymbol{\psi}_{k,i-1}) \end{cases} \tag{48a}$$

end

Adapt-Then-Combine (ATC) Diffusion: A similar implementation can be obtained by switching the order of the combination and adaptation steps in (48a), as shown in the listing (48b). The structure of the CTA and ATC strategies are fundamentally identical: the difference lies in which variable we choose to correspond to the updated iterate $\boldsymbol{w}_{k,i}$. In ATC, we choose the result of the *combination* step to be $\boldsymbol{w}_{k,i}$, whereas in CTA, we choose the result of the *adaptation* step to be $\boldsymbol{w}_{k,i}$.

Diffusion Strategy for Distributed Adaptation (ATC)

```
for each instant $i \geq 0$:
    each agent $k = 1, 2, \ldots, N$ performs the update
        $\begin{cases} \boldsymbol{\psi}_{k,i} = \boldsymbol{w}_{k,i-1} - \mu_k \widehat{\nabla_{w^\top} J_k}(\boldsymbol{w}_{k,i-1}) \\ \boldsymbol{w}_{k,i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \boldsymbol{\psi}_{\ell,i} \end{cases}$    (48b)
end
```

Example 12 (Diffusion LMS Networks): For the MSE network of Example 9, the ATC and CTA diffusion strategies reduce to

$$\begin{cases} \boldsymbol{\psi}_{k,i-1} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \boldsymbol{w}_{\ell,i-1} \quad \text{(CTA diffusion)} \\ \boldsymbol{w}_{k,i} = \boldsymbol{\psi}_{k,i-1} + 2\mu_k \boldsymbol{u}_{k,i}^\top \left[\boldsymbol{d}_k(i) - \boldsymbol{u}_{k,i} \boldsymbol{\psi}_{k,i-1}\right] \end{cases} \tag{49a}$$

and

$$\begin{cases} \boldsymbol{\psi}_{k,i} = \boldsymbol{w}_{k,i-1} + 2\mu_k \boldsymbol{u}_{k,i}^\top \left[\boldsymbol{d}_k(i) - \boldsymbol{u}_{k,i} \boldsymbol{w}_{k,i-1}\right] \\ \boldsymbol{w}_{k,i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \boldsymbol{\psi}_{\ell,i} \quad \text{(ATC diffusion).} \end{cases} \tag{49b}$$

♦

Example 13 (Diffusion Logistic Regression): We revisit the pattern classification problem from Example 2, where we consider a collection of N networked agents cooperating with each other to solve the logistic regression problem. Each agent receives streaming data $\{\boldsymbol{\gamma}_k(i), \boldsymbol{h}_{k,i}\}$, where the variable $\boldsymbol{\gamma}_k(i)$ assumes the values ± 1 and designates the class that the feature vector $\boldsymbol{h}_{k,i}$ belongs to. The objective is to use the training data to determine the vector w^o that minimizes the cost

$$J(w) \triangleq \frac{\rho}{2}\|w\|^2 + \mathbb{E}\left\{\ln\left[1 + e^{-\boldsymbol{\gamma}_k(i)\boldsymbol{h}_{k,i}^\top w}\right]\right\} \tag{50}$$

under the assumption of joint wide-sense stationarity over the random data. It is straightforward to verify that the ATC diffusion strategy (48b) reduces to the following form in this case:

$$\begin{cases} \boldsymbol{\psi}_{k,i} = (1 - \rho\mu_k)\boldsymbol{w}_{k,i-1} + \mu_k \left(\frac{\boldsymbol{\gamma}_k(i)}{1+e^{\boldsymbol{\gamma}_k(i)\boldsymbol{h}_{k,i}^\top \boldsymbol{w}_{k,i-1}}}\right)\boldsymbol{h}_{k,i} \\ \boldsymbol{w}_{k,i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \boldsymbol{\psi}_{\ell,i}. \end{cases} \tag{51}$$

♦

Diffusion Strategies With Enlarged Cooperation: Other forms of diffusion cooperation are possible by allowing for enlarged exchange of information among the agents, such as exchanging gradient vector approximations *in addition* to the iterates. For example, the following ATC form (similarly for CTA) employs an additional set of combination coefficients $\{c_{\ell k}\}$ to aggregate gradient information [11], [32], [98]:

$$\begin{cases} \boldsymbol{\psi}_{k,i} = \boldsymbol{w}_{k,i-1} - \mu_k \sum_{\ell \in \mathcal{N}_k} c_{\ell k} \widehat{\nabla_{w^\top} J_\ell}(\boldsymbol{w}_{k,i-1}) \\ \boldsymbol{w}_{k,i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \boldsymbol{\psi}_{\ell,i} \quad \text{(ATC diffusion)} \end{cases} \tag{52a}$$

where $\{c_{\ell k}\}$ are nonnegative scalars that satisfy

$$c_{\ell k} \geq 0, \quad \sum_{k=1}^{N} c_{\ell k} = 1, \quad \text{and} \quad c_{\ell k} = 0 \text{ if } \ell \notin \mathcal{N}_k. \tag{52b}$$

If we collect the entries $\{c_{\ell k}\}$ into an $N \times N$ matrix C, so that the ℓth row of C is formed of $\{c_{\ell k},\ k = 1, 2, \ldots, N\}$, then $C\mathbb{1} = \mathbb{1}$ and we say that C is a *right-stochastic* matrix. Observe that (52a) is equivalent to associating with each agent k the weighted neighborhood cost function

$$J_k'(w) \triangleq \sum_{\ell \in \mathcal{N}_k} c_{\ell k} J_\ell(w) \tag{53}$$

and then applying (48b). Our discussion in the rest of the paper focuses on the case $C = I_N$.

D. Discussion and Related Literature

There has been extensive work on consensus techniques in the literature, starting with the foundational results of [104] and [105], which were of a different nature and did not respond to streaming data arriving continuously at the agents, as is the case, for instance, with the continuous arrival of data $\{\boldsymbol{d}_k(i), \boldsymbol{u}_{k,i}\}$ in Examples 11 and 12. The original consensus formulation deals instead with the problem of computing averages over graphs. This can be explained as follows [83], [84], [104], [105]. Consider a collection of measurements denoted by $\{w_\ell,\ \ell = 1, 2, \ldots, N\}$ available at the vertices of a connected graph with N agents. The objective is to devise a distributed algorithm that enables every agent to determine the *average* value

$$\overline{w} \triangleq \frac{(w_1 + w_2 + \cdots + w_N)}{N} \tag{54a}$$

by interacting solely with its neighbors. When this occurs, we say that the agents have reached consensus (or agreement) about $\overline{w}$. We select an $N \times N$ *doubly stochastic* combination matrix $A = [a_{\ell k}]$; a doubly stochastic matrix is one that has nonnegative elements and satisfies $A^\top \mathbb{1} = \mathbb{1}$

and $A\mathbb{1} = \mathbb{1}$. We assume that the second largest magnitude eigenvalue of A satisfies $|\lambda_2(A)| < 1$. Using the combination coefficients $\{a_{\ell k}\}$, each agent k then iterates *repeatedly* on the data of its neighbors

$$\begin{aligned} w_{k,i} &= \sum_{\ell \in \mathcal{N}_k} a_{\ell k} w_{\ell,i-1}, \qquad i \geq 0, \\ k &= 1, 2, \ldots, N \end{aligned} \tag{54b}$$

starting from the boundary conditions $w_{\ell,-1} = w_\ell$ for all $\ell \in \mathcal{N}_k$. Superscript i continues to denote the iteration index. Every agent k in the network performs the same calculation, which amounts to combining repeatedly, and in a convex manner, the state values of its neighbors. It can then be shown that (see [104], [105], and [11, App. E])

$$\lim_{i \to \infty} w_{k,i} = \overline{w}, \qquad k = 1, 2, \ldots, N. \tag{54c}$$

In this way, through the localized iterative process (54b), the agents are able to converge to the global average value $\overline{w}$.

Motivated by this elegant result, several works in the literature (e.g., [27], [81], [82], [84]–[89], and [106]–[109]) proposed useful extensions of the consensus construction (54b) to minimize aggregate costs of the form (37) or to solve distributed estimation problems of the least squares or Kalman filtering type. Some of the earlier extensions involved the use of *two* separate time scales: one faster time scale for performing multiple consensus iterations similar to (54b) over the states of the neighbors, and a second slower time scale for performing gradient vector updates or for updating the estimators by using the result of the consensus iterations (e.g., [81], [87], [89], and [106]–[109]). Such two time-scale implementations are a hindrance for real-time adaptation from streaming data. And the separate time scales turn out to be unnecessary, as already indicated by the single time-scale consensus and diffusion algorithms described by (46a), (49a), and (49b).

Building upon a useful procedure for distributed optimization from [84, eq. (2.1)] and [27, eq. (7.1)], more recent works suggested single time-scale implementations for consensus strategies by using an implementation similar to (46a); see, e.g., [86, eq. (3)], [88, eq. (9)], [90, eq. (19)], [96, eq. (3)], and [123, p. 1029]. These references, however, employ decaying step sizes, $\mu_k(i) \to 0$, to ensure that the iterates $\{\boldsymbol{w}_{k,i}\}$ across all agents will converge almost surely to the same value (thus, reaching agreement or consensus). As noted before, when diminishing step sizes are used, adaptation is turned off over time, which is prejudicial for learning purposes. For this reason, we are setting the step sizes to constant values in (46a) in order to endow the consensus iteration with continuous adaptation and learning abilities. Nevertheless, some care is needed for consensus implementations with constant step sizes. That is because, as explained in Examples 22 and 23, and as alluded to earlier, instability can occur due to the inherent asymmetry in the dynamics of the consensus iteration. The same examples show how diffusion strategies resolve the instability problem.

One of the main motivations for the introduction and study of cooperative strategies of the diffusion type [see (49a) and (49b)] has been to show that *single* time-scale-distributed learning from streaming data is possible, and that it can be achieved under *constant* step-size adaptation [15], [33], [41], [97], [98], [110]–[113], [115]; these strategies also allow A to be left stochastic and, therefore, permit larger modes of cooperation than doubly stochastic policies. The CTA diffusion strategy (49a) was first introduced for MSE estimation problems in [97] and [110]–[112]. The ATC diffusion structure (49b), with adaptation preceding combination, appeared in the work [114] on adaptive distributed least squares schemes and also in the works [98] and [115]–[117] on distributed MSE and state–space estimation methods. The CTA structure (48a) with an iteration-dependent step size that decays to zero, $\mu(i) \to 0$, was adopted in [118]–[120] to solve distributed optimization problems that require all agents to reach agreement. The ATC form (49b), also with an iteration-dependent sequence $\mu(i)$ that decays to zero, was further employed in [121] and [122] to ensure almost-sure convergence and agreement among agents.

There has also been works on applying the alternating direction method of multipliers (ADMM) [124] to the design of consensus-type algorithms [125], [126]. To enforce agreement among the agents, the latter references impose the requirement that the iterates at the agents should match each other and, consequently, obtain implementations that necessitate the fine tuning of additional parameters and whose performance is sensitive to these parameters. One of the advantages of the consensus (46a) and diffusion strategies (48a) and (48b) studied in this paper is that, as the discussion will reveal, they naturally lead to an equalization effect across the agents without added complexity; see the explanation following (62).

The distributed strategies described so far in this paper are well suited for cooperative networks where agents interact with each other to optimize an aggregate cost function. There are, of course, situations in which agents may behave in a selfish manner. In these cases, agents would participate in the collaborative process and share information with their neighbors only if the cooperation is deemed beneficial to them (e.g., [66] and [67]). We do not study this situation in the current paper and focus instead on cooperative networks.

VI. NETWORK LIMIT POINT

Now that we have described several strategies for distributed adaptation, learning, and optimization, including consensus and diffusion strategies, we remark that the study of the convergence and performance of these algorithms is more challenging than the study of the noncooperative and centralized implementations from the previous sections. This is because the agents are now coupled by a network topology and they influence each other's behavior. Some interesting patterns of behavior arise as a result of the coupling among the agents. For example, one interesting result established in [33], [41], and [127] is the following. Recall that the original motivation for deriving the consensus and diffusion strategies has been to seek the unique minimizer w^o of the aggregate cost (37). This minimizer is obviously the unique solution to

$$\nabla_w J^{\text{glob}}(w^o) = 0 \Longleftrightarrow \sum_{k=1}^{N} \nabla_w J_k(w^o) = 0. \tag{55}$$

It turns out though that the iterates $\{\boldsymbol{w}_{k,i}\}$ at the various agents in the distributed consensus or diffusion solution will converge toward another limit point, which we will denote by $w^\star$, and whose value is dependent on the network topology. We identify $w^\star$ in the rest of the paper and explain when $w^\star = w^o$.

A. Perron Eigenvector

Since we are assuming strongly connected networks, then the left-stochastic combination matrix A will be primitive. It then follows from the Perron–Frobenius theorem [59] that:

1) matrix A will have a *single* eigenvalue at one;
2) all other eigenvalues of A will be strictly inside the unit circle so that $\rho(A) = 1$;
3) with proper sign scaling, all entries of the right eigenvector of A corresponding to the single eigenvalue at one will be *positive*. Let p denote this right eigenvector with its entries $\{p_k\}$ normalized to add up to one, i.e.,

$$Ap = p, \quad \mathbb{1}^\top p = 1, \quad p_k > 0, \quad k = 1, 2, \ldots, N \tag{56}$$

We refer to p as the Perron eigenvector of A.

B. Weighted Aggregate Cost

Following [41] and [127], we next introduce the scalars

$$q_k \overset{\Delta}{=} \mu_k p_k > 0, \qquad k = 1, 2, \ldots, N \tag{57a}$$

and the *weighted* aggregate cost; compare with (37)

$$J^{\text{glob},\star}(w) \overset{\Delta}{=} \sum_{k=1}^{N} q_k J_k(w). \tag{57b}$$

Since all the $J_k(w)$ are convex in w, then the strong convexity of $J^{\text{glob}}(w)$ guarantees the strong convexity of $J^{\text{glob},\star}(w)$. It follows that $J^{\text{glob},\star}(w)$ will have a unique global minimum, which we denote by $w^\star$, and it satisfies

$$\nabla_w J^{\text{glob},\star}(w^\star) = 0 \Longleftrightarrow \sum_{k=1}^{N} q_k \nabla_w J_k(w^\star) = 0. \tag{57c}$$

In general, the minimizers $\{w^o, w^\star\}$ of $J^{\text{glob}}(w)$ and $J^{\text{glob},\star}(w)$, respectively, are different. However, they will coincide in some important cases such as the following.

1) When the $\{q_k\}$ are equal to each other. This situation occurs, for example, when $\mu_k \equiv \mu$ across all agents and A is doubly stochastic (for which $p = \mathbb{1}/N$).
2) When the individual costs $J_k(w)$ are all minimized at the *same* location.

The arguments in the rest of the paper will establish that the location $w^\star$ serves as the limit point for the iterates at the various agents in the MSE sense. Specifically, if we now define the errors relative to $w^\star$, say, as

$$\tilde{\boldsymbol{w}}_{k,i} \overset{\Delta}{=} w^\star - \boldsymbol{w}_{k,i}, \qquad k = 1, 2, \ldots, N \tag{58}$$

then we will be arguing in (99) that

$$\limsup_{i\to\infty} \mathbb{E}\|\tilde{\boldsymbol{w}}_{k,i}\|^2 = O(\mu_{\max}) \tag{59}$$

where $\mu_{\max}$ is the maximum step size across all agents. In this way, by calling upon Markov's inequality again and using an argument similar to (27c), we would be able to conclude that each $\boldsymbol{w}_{k,i}$ approaches $w^\star$ asymptotically with high probability for sufficiently small step sizes. It is explained in [41] that the vector $w^\star$ can be interpreted as corresponding to a Pareto optimal solution for the collection of convex functions $\{J_k(w)\}$ [29], [128], [129].

VII. NETWORK PERFORMANCE

In order to assess the benefit of cooperation over networks, it is important to examine how close and how fast the agents get to the limit point $w^\star$. We explain the main steps involved in the MSE stability and performance analysis of the distributed strategies in Section VIII. Here, we call upon the results from that section to highlight several useful aspects of network behavior.

A. MSD Performance

We denote the MSD performance of the individual agents and the average MSD performance across the network by

$$\mathrm{MSD}_{\mathrm{dist},k} \overset{\Delta}{=} \lim_{i\to\infty} \mathbb{E}\|\widetilde{\boldsymbol{w}}_{k,i}\|^2 \tag{60a}$$

$$\mathrm{MSD}_{\mathrm{dist,av}} \overset{\Delta}{=} \frac{1}{N}\sum_{k=1}^{N} \mathrm{MSD}_{\mathrm{dist},k} \tag{60b}$$

where the error vectors are measured relative to the limit point $w^\star$, as defined by (58). We further denote the gradient noise process at the individual agents by

$$\boldsymbol{s}_{k,i}(w) \overset{\Delta}{=} \widehat{\nabla_{w^\top} J}_k(w) - \nabla_{w^\top} J_k(w) \tag{61a}$$

and define

$$H_k \overset{\Delta}{=} \nabla_w^2 J_k(w^\star) \tag{61b}$$

$$R_{s,k} \overset{\Delta}{=} \lim_{i\to\infty} \mathbb{E}\left[\boldsymbol{s}_{k,i}(w^\star)\boldsymbol{s}_{k,i}^\top(w^\star)|\boldsymbol{\mathcal{F}}_{i-1}\right] \tag{61c}$$

where $\boldsymbol{\mathcal{F}}_{i-1}$ now represents the collection of all random events generated by the iterates from across all agents, $\{\boldsymbol{w}_{k,j},\ k=1,2,\ldots,N\}$, up to time $i-1$; see (91) and (92b). Observe that $\{H_k, R_{s,k}\}$ are defined relative to the limit point $w^\star$ just like $\widetilde{\boldsymbol{w}}_{k,i}$; compare with the earlier expressions at the end of Lemma III.1. It is established in (112), under some assumptions on the gradient noise processes (61a), that for strongly connected networks and for sufficiently small step sizes [15], [33], [127], [130]

$$\begin{aligned}\mathrm{MSD}_{\mathrm{dist},k} &\doteq \mathrm{MSD}_{\mathrm{dist,av}} \\ &= \frac{1}{2}\mathrm{Tr}\left[\left(\sum_{k=1}^{N}\mu_k p_k H_k\right)^{-1}\left(\sum_{k=1}^{N}\mu_k^2 p_k^2 R_{s,k}\right)\right] + O\left(\mu_{\max}^{3/2}\right)\end{aligned} \tag{62}$$

where the notation $a \doteq b$ means that a and b agree to first order in $\mu_{\max}$. Observe from (62) the interesting conclusion that the consensus and diffusion strategies are able to *equalize* the MSD performance across all agents for sufficiently small step sizes. It is also instructive to compare expression (62) with (39d) in the centralized case. Observe how cooperation among the agents leads to the appearance of the scaling coefficients $\{p_k\}$ defined by (56); these factors are determined by A.

Example 14 (Performance of MSE Networks): We continue with the setting of Example 9, which deals with MSE networks. In that case, all individual costs are minimized at the same location w^o so that the minimizers w^o and $w^\star$ coincide with each other. We assume the agents employ uniform step sizes and have uniform regression covariance matrices, i.e., $\mu_k \equiv \mu$ and $R_{u,k} \equiv R_u$ for $k=1,2,\ldots,N$. It follows that in the current setting: $H_k = 2R_u \equiv H$ and $R_{s,k} = 4\sigma_{v,k}^2 R_u$. Substituting into (62), we conclude that the MSD performance of the consensus or diffusion strategies from Examples 11 and 12 are well approximated by

$$\mathrm{MSD}_{\mathrm{dist},k} \doteq \mathrm{MSD}_{\mathrm{dist,av}} \approx \mu M\left(\sum_{k=1}^{N} p_k^2\sigma_{v,k}^2\right). \tag{63a}$$

If the combination matrix A happens to be doubly stochastic, then $p = \mathbb{1}/N$. Substituting $p_k = 1/N$ into (63a) gives

$$\mathrm{MSD}_{\mathrm{dist},k} \doteq \mathrm{MSD}_{\mathrm{dist,av}} \approx \mu M\frac{1}{N^2}\left(\sum_{k=1}^{N}\sigma_{v,k}^2\right) \tag{63b}$$

which agrees with the centralized performance (35b). In other words, the distributed strategies are able to match the performance of the centralized solution for doubly stochastic combination policies.

Another situation of interest is when the combination weights $\{a_{\ell k}\}$ are selected according to the averaging rule $a_{\ell k} = 1/n_k$ for $\ell \in \mathcal{N}_k$ and zero otherwise, where $n_k = |\mathcal{N}_k|$ denotes the size of the neighborhood of agent k (or its degree) [131]. In this case, the matrix A is left stochastic and the entries of its Perron eigenvector are given by $p_k = n_k/\sum_{m=1}^{N} n_m$. Then, (63a) gives

$$\mathrm{MSD}_{\mathrm{dist,av}} \approx \mu M\left(\sum_{k=1}^{N} n_k\right)^{-2}\left(\sum_{k=1}^{N} n_k^2\sigma_{v,k}^2\right) \tag{63c}$$

which would reduce to (63b) when the degrees of all agents are uniform, i.e., $n_k \equiv n$. ♦

Example 15 (Is Cooperation Always Beneficial?): We continue with the discussion from Example 14 over MSE networks. If each agent in the network were to estimate w^o on its own in a noncooperative manner by running its individual LMS learning rule, then we know from (32a) that each agent will attain the MSD level shown below, along with the average performance across all N agents

$$\mathrm{MSD}_{\mathrm{ncop}k} \approx \mu M\sigma_{v,k}^2 \tag{64a}$$

$$\mathrm{MSD}_{\mathrm{ncop,av}} \approx \mu M\left(\frac{1}{N}\sum_{k=1}^{N}\sigma_{v,k}^2\right). \tag{64b}$$

Now assume A is doubly stochastic. Comparing (63a) with (64b), it is obvious that

$$\mathrm{MSD}_{\mathrm{dist,av}} \approx \frac{1}{N}\mathrm{MSD}_{\mathrm{ncop,av}} \tag{64c}$$

which shows that, for MSE networks, the consensus and diffusion strategies outperform the average performance of the noncooperative strategy by a factor of N. But how do the performance metrics of an agent compare to each other in the distributed and noncooperative modes of operation? From (63a) and (64a), we observe that if the noise variance is uniform across all agents, i.e., $\sigma_{v,k}^2 \equiv \sigma_v^2$, then the MSD of each individual agent in the distributed solution will be smaller by the same factor N than their noncooperative performance. However, when the noise profile varies across the agents, then the performance metrics of an individual agent in the distributed and noncooperative solutions cannot be compared directly: one can be larger than the other depending on the noise profile. For example, for $N = 2$, $\sigma_{v,1}^2 = 1$, and $\sigma_{v,2}^2 = 9$, agent 1 will not benefit from cooperation while agent 2 will. ♦

B. Excess Risk Performance

The ER measure for distributed strategies is generally of interest when the cost functions $J_k(w)$ are identical, i.e., when $J_k(w) \equiv J(w)$ for $k = 1, 2, \ldots, N$, in which case the minimizers $\{w^o, w^\star\}$ will coincide with each other. The ER for agent k and the average ER for the network are then defined as

$$\mathrm{ER}_k \overset{\Delta}{=} \lim_{i\to\infty} \mathbb{E}\left\{J(\boldsymbol{w}_{k,i-1}) - J(w^\star)\right\} \tag{65a}$$

$$\mathrm{ER}_{\mathrm{dist,av}} \overset{\Delta}{=} \frac{1}{N}\sum_{k=1}^{N} \mathrm{ER}_k. \tag{65b}$$

It is explained later, following the proof of Lemma VIII.2, that for strongly connected networks, and for sufficiently small step sizes, it holds [16], [33], [127], [130]

$$\begin{aligned}\mathrm{ER}_{\mathrm{dist},k} &\doteq \mathrm{ER}_{\mathrm{dist,av}}\\ &= \frac{1}{4}\left(\sum_{k=1}^{N}\mu_k p_k\right)^{-1} \mathrm{Tr}\left(\sum_{k=1}^{N}\mu_k^2 p_k^2 R_{s,k}\right) + O\left(\mu_{\max}^{3/2}\right).\end{aligned} \tag{66}$$

Example 16 (Performance of Online Diffusion Learner): We revisit Example 6 and consider now a collection of N learners cooperating to minimize a strongly convex function $J(w)$ over a strongly connected network, namely

$$w^o \overset{\Delta}{=} \arg\min_w J(w) = \arg\min_w \mathbb{E}Q(w;\boldsymbol{x}_{k,i}). \tag{67a}$$

Each learner k receives a streaming sequence of temporally and spatially independent data vectors $\{\boldsymbol{x}_{k,i}, i \geq 0\}$ that arise from some common fixed distribution $\mathcal{X}$. We assume the agents run a consensus or diffusion strategy, say, the ATC diffusion strategy (48b)

$$\begin{cases}\boldsymbol{\psi}_{k,i} = \boldsymbol{w}_{k,i-1} - \mu_k \nabla_{w^\top} Q(\boldsymbol{w}_{k,i-1};\boldsymbol{x}_{k,i})\\ \boldsymbol{w}_{k,i} = \sum\limits_{\ell\in\mathcal{N}_k} a_{\ell k}\boldsymbol{\psi}_{\ell,i}.\end{cases} \tag{67b}$$

Then, the absolute gradient noise covariance matrix is given by $R_s = \mathbb{E}\nabla_{w^\top} Q(w^o;\boldsymbol{x}_{k,i})\nabla_w Q(w^o;\boldsymbol{x}_{k,i})$. Substituting into (66), we conclude that the ER of the diffusion solution (and of consensus as well) is given by

$$\mathrm{ER}_{\mathrm{dist,av}} \approx \frac{1}{4}\left(\sum_{k=1}^{N}\mu_k p_k\right)^{-1}\left(\sum_{k=1}^{N}\mu_k^2 p_k^2\right)\mathrm{Tr}(R_s). \tag{67c}$$

We illustrate this result numerically by means of a simulation in Fig. 5 for a network consisting of $N = 10$ identical agents (similar simulations can be carried out for other examples in the paper but it is sufficient to consider one simulation to avoid redundancy). We select as combination policy the Metropolis rule, which is described further ahead in (70a) and for which $p_k = 1/N$. The figure shows the evolution of the average ER metric across the agents for three implementations: the diffusion strategy (51) from Example 13, the related consensus strategy that would result from (46a), and the noncooperative strategy (33b) from Example 6. The data used for the simulation originate from the alpha data set [132]; we only use the first 25 features for illustration purposes so that $M = 25$. We also use $\rho = 10$ and a small uniform step size $\mu_k = \mu = 0.0033$ in one simulation (left) and a five times larger step size in the second simulation (right). The learning curves in the figure are averaged over 125 experiments. To generate the curves for this example, the optimal w^o and the gradient noise covariance matrix R_s are estimated offline by applying a batch algorithm to all data points. The curves in the middle plot show how the ER measure tends toward the steady-state levels predicted by theory in the slow adaptation regime. The curves in the rightmost plot show how diffusion strategies have superior performance at larger step sizes. This fact is not inconsistent with expression (66); this is because the expression shows that diffusion and consensus strategies attain similar steady-state performance levels to first order in $\mu_{\max}$; differences in performance generally occur at higher order terms in $\mu_{\max}$ [1], [99]. ♦

C. Left-Stochastic Combination Policies

Example 15 focused on MSE networks with quadratic costs. For more general costs, doubly stochastic combination policies can be verified to similarly enhance network MSD performance over the average performance of noncooperative agents. However, the performance of some individual agents in the networked solution may still be degraded relative to their performance in the

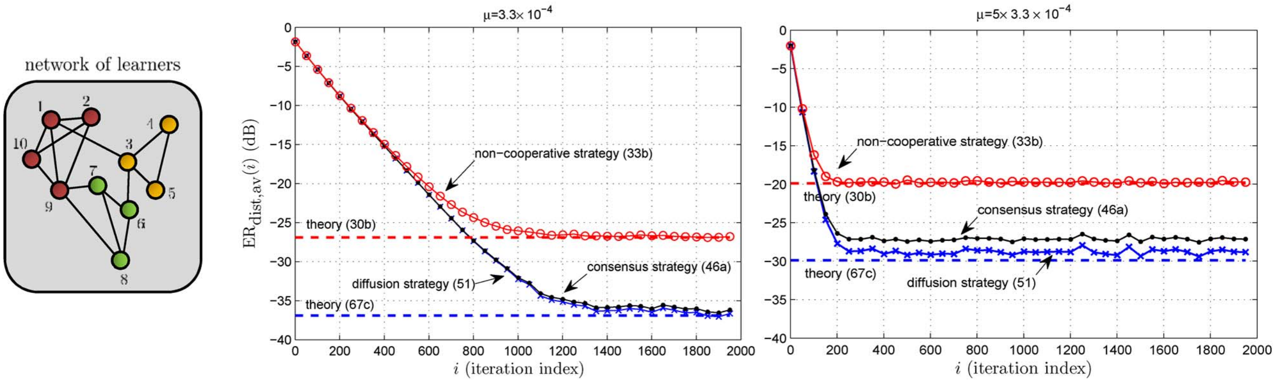

Fig. 5. *Illustration of the evolution of the ER performance for three strategies: noncooperative, consensus, and diffusion. The horizontal dashed lines in both figures indicate the steady-state levels predicted by theory in the slow adaptation regime.*

noncooperative scenario. One useful question to consider is whether it is possible to select combination matrices A that ensure that distributed (consensus or diffusion) networks will outperform the noncooperative strategy both in terms of the overall average performance *and* the individual agent performance. The choice of A will generally need to be left stochastic.

We address this question for the special case of uniform step sizes $\mu_k \equiv \mu$ and uniform Hessian matrices $H_k \equiv H$. We then seek an optimal policy that minimizes the network MSD metric, namely

$$A^o \triangleq \arg\min_{A\in\mathbb{A}} \operatorname{Tr}\left(\sum_{k=1}^{N} p_k^2 H^{-1} R_{s,k}\right)$$
$$\text{subject to} \quad Ap = p, \mathbb{1}^\top p = 1, \qquad p_k > 0 \tag{68a}$$

where $\mathbb{A}$ denotes the set of left-stochastic primitive matrices. We introduce the nonnegative scalars

$$\theta_k^2 \triangleq \operatorname{Tr}(H^{-1}R_{s,k}), \qquad k = 1, 2, \ldots, N. \tag{68b}$$

Interpreting every $A \in \mathbb{A}$ as the probability transition matrix of an irreducible aperiodic Markov chain [44], [133], and using a construction procedure developed in [134] and [135], it was argued in [11] and [58] that one choice for A^o is the following policy, which we will refer to as the Hastings rule:

$$a_{\ell k}^o = \begin{cases} \dfrac{\theta_k^2}{\max\left\{n_k\theta_k^2, n_\ell\theta_\ell^2\right\}}, & \ell \in \mathcal{N}_k \setminus \{k\} \\ 1 - \displaystyle\sum_{m\in\mathcal{N}_k\setminus\{k\}} a_{mk}^o, & \ell = k. \end{cases} \tag{68c}$$

The entries of the corresponding Perron eigenvector are

$$p_k^o = \left(\frac{1}{\theta_k^2}\right)\left(\sum_{\ell=1}^{N} 1/\theta_\ell^2\right)^{-1}, \qquad k = 1, 2, \ldots, N. \tag{68d}$$

Substituting into (68a), we find that the resulting (equalized) optimal performance is

$$\text{MSD}_{\text{dist,k}}^o \doteq \text{MSD}_{\text{dist,av}}^o \approx \frac{\mu}{2}\left(\sum_{k=1}^{N} 1/\theta_k^2\right)^{-1} \tag{68e}$$
$$\leq \frac{\mu}{2}\theta_k^2$$
$$\overset{(30a)}{\approx} \text{MSD}_{\text{ncop},k} \tag{68f}$$

so that the individual agent performance in the optimized distributed network is improved across all agents relative to the noncooperative case.

Example 17 (Optimal Policy for Distributed Networks): Let us continue with the setting of Example 14, which deals with MSE networks. We assumed uniform step sizes and uniform regression covariance matrices. In this setting, we have $\theta_k^2 = 2M\sigma_{v,k}^2$, and the Hastings rule (68c) reduces to

$$a_{\ell k}^o = \frac{\sigma_{v,k}^2}{\max\left\{n_k\sigma_{v,k}^2, n_\ell\sigma_{v,\ell}^2\right\}}, \qquad \ell \in \mathcal{N}_k \setminus \{k\} \tag{69a}$$

with the resulting optimal MSD level given by

$$\text{MSD}_{\text{dist,k}}^o \doteq \text{MSD}_{\text{dist,av}}^o \approx \mu M\left(\sum_{k=1}^{N} 1/\sigma_{v,k}^2\right)^{-1}. \tag{69b}$$

♦

Example 18 (Optimal Policy for Online Learning): We revisit Example 16 for which $\theta_k^2 = \operatorname{Tr}(H^{-1}R_s) \equiv \theta^2$, where

$H = \nabla_w^2 J(w^o)$. Substituting into (68c), we find that the Hastings rule reduces to the Metropolis rule (which is doubly stochastic) [81], [134], [136]

$$a_{\ell k}^o = \begin{cases} \frac{1}{\max\{n_k, n_\ell\}}, & \ell \in \mathcal{N}_k \setminus \{k\} \\ 1 - \sum\limits_{m \in \mathcal{N}_k \setminus \{k\}} a_{mk}^o, & \ell = k. \end{cases} \tag{70a}$$

The optimal MSD value is given by

$$\text{MSD}_{\text{dist,k}}^o \doteq \text{MSD}_{\text{dist,av}}^o \approx \frac{\mu}{4N} \text{Tr}(H^{-1} R_s). \tag{70b}$$

♦

D. Comparison With Centralized Solution

We know from (39d) that the performance of the centralized solution under the uniform condition $H_k \equiv H$ is given by

$$\text{MSD}_{\text{cent}} \approx \frac{\mu}{2N^2} \sum_{k=1}^{N} \theta_k^2. \tag{71a}$$

Comparing with the optimal distributed solution (68e), we conclude that

$$\text{MSD}_{\text{dist,av}}^o \le \text{MSD}_{\text{cent}} \tag{71b}$$

so that the optimized distributed network running the consensus strategy (46a) or the diffusion strategies (48a) or (48b) with the Hasting rule (68c) *outperforms* the centralized solution (38a). This conclusion is not surprising because the optimized combination coefficients (68c) for the distributed implementations exploit knowledge about the gradient noise factors, $\{\theta_\ell^2\}$. This information is not used by the centralized algorithm (38a). However, we can modify the centralized implementation by employing a weighted combination of the approximate gradient vectors using (68d) as follows:

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} - \mu \sum_{k=1}^{N} p_k^o \widehat{\nabla_{w^\top} J}_k(\boldsymbol{w}_{i-1}), \qquad i \ge 0. \tag{72}$$

Then, the MSD performance of this algorithm can be verified to match (68e) [1], [137], [138].

E. Static Versus Adaptive Combination Policies

The combination weights $\{a_{\ell k}\}$ used by the consensus (46a) and diffusion strategies (48a) and (48b) influence the performance of the distributed solution in a direct manner. Their influence is reflected by the entries $\{p_k\}$ of the Perron eigenvector, defined by (56), and which appear in the MSD and ER performance expressions (62) and (66), respectively. Several rules have been proposed in the literature for the selection of the $\{a_{\ell k}\}$, such as the (left stochastic) averaging rule encountered in Example 14, and the (symmetric and doubly stochastic) Metropolis rule encountered in Example 18. However, these and other similar rules, are seen to construct the $\{a_{\ell k}\}$ exclusively from the degrees (or the extent of connectivity) of the agents.

It is explained in [1] and [11] that, while such selections may be appropriate in some applications, they can nevertheless lead to degraded performance in the context of adaptation and learning over networks [139]. This is because these weighting schemes do no take into account the gradient noise profile across the network. Since some agents can be noisier than others, it is insufficient to rely exclusively on the degrees of the agents. It is also important to consider the amount of noise that is present at the agents, and to assign more or less weight to neighbors depending on their noise conditions. The Hastings rule is one example of a combination policy that takes the noise factors $\{\theta_k^2\}$ into account. One difficulty in employing this strategy is that the factors $\{\theta_\ell^2\}$ are generally unavailable since their values depend on the unknown noise moments $\{R_{s,\ell}\}$. It is, therefore, useful to devise noise-aware schemes that enable the agents to estimate and track the factors $\{\theta_\ell^2\}$, as well as adjust the combination weights $\{a_{\ell k}\}$ in an adaptive manner (see [11], [58], [140], and [141] for examples of such schemes).

VIII. NETWORK STABILITY AND PERFORMANCE

In this section, we establish the performance and stability results that were alluded to in the earlier sections in (59) and in (62) and (66) for the distributed (consensus and diffusion) strategies. We carry out the analysis in a *unified* manner for both classes of algorithms by following the MSE formulation and energy conservation arguments of [1], [11], [15], [16], [33], and [47].

A. MSE Networks: Consensus Versus Diffusion

We motivate the analysis by presenting first illustrative examples from [11] dealing with MSE networks, which involve quadratic costs that share a common minimizer. Following the examples, we extend the framework to more general costs.

Example 19 (Error Dynamics Over MSE Networks): We consider the MSE network of Example 9, which involves quadratic costs with a common minimizer w^o. The update equations for the noncooperative, consensus, and diffusion strategies are given by (13f), (47), and (49a), and (49b). We can group these strategies into a single unifying description by considering the following structure in terms of three sets of combination coefficients $\{a_{o,\ell k}, a_{1,\ell k}, a_{2,\ell k}\}$

$$\begin{cases} \boldsymbol{\phi}_{k,i-1} = \sum\limits_{\ell\in\mathcal{N}_k} a_{1,\ell k}\boldsymbol{w}_{\ell,i-1} \\ \boldsymbol{\psi}_{k,i} = \sum\limits_{\ell\in\mathcal{N}_k} a_{o,\ell k}\boldsymbol{\phi}_{\ell,i-1} + 2\mu_k\boldsymbol{u}_{k,i}^{\mathsf{T}}\left[\boldsymbol{d}_k(i) - \boldsymbol{u}_{k,i}\boldsymbol{\phi}_{k,i-1}\right] \\ \boldsymbol{w}_{k,i} = \sum\limits_{\ell\in\mathcal{N}_k} a_{2,\ell k}\boldsymbol{\psi}_{\ell,i}. \end{cases} \tag{73}$$

In (73), the quantities $\{\boldsymbol{\phi}_{k,i-1}, \boldsymbol{\psi}_{k,i}\}$ denote $M \times 1$ intermediate variables, while the nonnegative entries of the $N \times N$ matrices $A_o = [a_{o,\ell k}]$, $A_1 = [a_{1,\ell k}]$, and $A_2 = [a_{2,\ell k}]$ are assumed to satisfy the same conditions (46b) and, hence, the matrices $\{A_o, A_1, A_2\}$ are left stochastic. Any of the combination weights $\{a_{o,\ell k}, a_{1,\ell k}, a_{2,\ell k}\}$ is zero whenever $\ell \notin \mathcal{N}_k$. Different choices for $\{A_o, A_1, A_2\}$ correspond to different strategies, as the following examples reveal and where we are introducing the matrix product $P = A_1A_oA_2$:

$$\text{noncooperative: } A_1 = A_o = A_2 = I_N \longrightarrow P = I_N \tag{74a}$$
$$\text{consensus: } A_o = A,\ A_1 = I_N = A_2 \longrightarrow P = A \tag{74b}$$
$$\text{CTA diffusion: } A_1 = A,\ A_2 = I_N = A_o \longrightarrow P = A \tag{74c}$$
$$\text{ATC diffusion: } A_2 = A,\ A_1 = I_N = A_o \longrightarrow P = A. \tag{74d}$$

We associate with each agent k the following three errors:

$$\widetilde{\boldsymbol{w}}_{k,i} \overset{\Delta}{=} w^o - \boldsymbol{w}_{k,i} \tag{75a}$$
$$\widetilde{\boldsymbol{\psi}}_{k,i} \overset{\Delta}{=} w^o - \boldsymbol{\psi}_{k,i} \tag{75b}$$
$$\widetilde{\boldsymbol{\phi}}_{k,i-1} \overset{\Delta}{=} w^o - \boldsymbol{\phi}_{k,i-1} \tag{75c}$$

which measure the deviations from the global minimizer w^o. Subtracting w^o from both sides of the equations in (73), we get

$$\begin{cases} \widetilde{\boldsymbol{\phi}}_{k,i-1} = \sum\limits_{\ell\in\mathcal{N}_k} a_{1,\ell k}\widetilde{\boldsymbol{w}}_{\ell,i-1} \\ \widetilde{\boldsymbol{\psi}}_{k,i} = \sum\limits_{\ell\in\mathcal{N}_k} a_{o,\ell k}\widetilde{\boldsymbol{\phi}}_{\ell,i-1} - 2\mu_k\boldsymbol{u}_{k,i}^{\mathsf{T}}\boldsymbol{u}_{k,i}\widetilde{\boldsymbol{\phi}}_{k,i-1} - 2\mu_k\boldsymbol{u}_{k,i}^{\mathsf{T}}\boldsymbol{v}_k(i) \\ \widetilde{\boldsymbol{w}}_{k,i} = \sum\limits_{\ell\in\mathcal{N}_k} a_{2,\ell k}\widetilde{\boldsymbol{\psi}}_{\ell,i}. \end{cases} \tag{75d}$$

In a manner similar to (18a) and (18b), the gradient noise process at each agent k is given by

$$\boldsymbol{s}_{k,i}(\boldsymbol{\phi}_{k,i-1}) = 2\left(R_{u,k} - \boldsymbol{u}_{k,i}^{\mathsf{T}}\boldsymbol{u}_{k,i}\right)\widetilde{\boldsymbol{\phi}}_{k,i-1} - 2\boldsymbol{u}_{k,i}^{\mathsf{T}}\boldsymbol{v}_k(i). \tag{75e}$$

In order to examine the evolution of the error dynamics across the network, we collect the error vectors from all agents into $N \times 1$ block error vectors (whose individual entries are of size $M \times 1$ each)

$$\widetilde{\boldsymbol{w}}_i \overset{\Delta}{=} \begin{bmatrix} \widetilde{\boldsymbol{w}}_{1,i} \\ \widetilde{\boldsymbol{w}}_{2,i} \\ \vdots \\ \widetilde{\boldsymbol{w}}_{N,i} \end{bmatrix}, \quad \widetilde{\boldsymbol{\psi}}_i \overset{\Delta}{=} \begin{bmatrix} \widetilde{\boldsymbol{\psi}}_{1,i} \\ \widetilde{\boldsymbol{\psi}}_{2,i} \\ \vdots \\ \widetilde{\boldsymbol{\psi}}_{N,i} \end{bmatrix}, \quad \widetilde{\boldsymbol{\phi}}_{i-1} \overset{\Delta}{=} \begin{bmatrix} \widetilde{\boldsymbol{\phi}}_{1,i-1} \\ \widetilde{\boldsymbol{\phi}}_{2,i-1} \\ \vdots \\ \widetilde{\boldsymbol{\phi}}_{N,i-1} \end{bmatrix}. \tag{75f}$$

Motivated by the last term in the second equation in (75d), and by the gradient noise terms (75e), we also introduce the following $N \times 1$ column vectors whose entries are of size $M \times 1$ each:

$$\boldsymbol{z}_i \overset{\Delta}{=} \begin{bmatrix} 2\boldsymbol{u}_{1,i}^{\mathsf{T}}\boldsymbol{v}_1(i) \\ 2\boldsymbol{u}_{2,i}^{\mathsf{T}}\boldsymbol{v}_2(i) \\ \vdots \\ 2\boldsymbol{u}_{N,i}^{\mathsf{T}}\boldsymbol{v}_N(i) \end{bmatrix}, \quad \boldsymbol{s}_i \overset{\Delta}{=} \begin{bmatrix} \boldsymbol{s}_{1,i}(\boldsymbol{\phi}_{1,i-1}) \\ \boldsymbol{s}_{2,i}(\boldsymbol{\phi}_{2,i-1}) \\ \vdots \\ \boldsymbol{s}_{N,i}(\boldsymbol{\phi}_{N,i-1}) \end{bmatrix}. \tag{75g}$$

We further introduce the Kronecker products

$$\mathcal{A}_o \overset{\Delta}{=} A_o \otimes I_M, \quad \mathcal{A}_1 \overset{\Delta}{=} A_1 \otimes I_M, \quad \mathcal{A}_2 \overset{\Delta}{=} A_2 \otimes I_M \tag{76a}$$

and the following $N \times N$ *block* diagonal matrices, whose individual entries are of size $M \times M$ each:

$$\mathcal{M} \overset{\Delta}{=} \text{diag}\{\mu_1I_M, \mu_2I_M, \ldots, \mu_NI_M\} \tag{76b}$$
$$\boldsymbol{\mathcal{R}}_i \overset{\Delta}{=} \text{diag}\left\{2\boldsymbol{u}_{1,i}^{\mathsf{T}}\boldsymbol{u}_{1,i}, 2\boldsymbol{u}_{2,i}^{\mathsf{T}}\boldsymbol{u}_{2,i}, \ldots, 2\boldsymbol{u}_{N,i}^{\mathsf{T}}\boldsymbol{u}_{N,i}\right\}. \tag{76c}$$

From (75d), we can then easily conclude that the block network variables satisfy the recursions

$$\begin{cases} \widetilde{\boldsymbol{\phi}}_{i-1} = \mathcal{A}_1^{\mathsf{T}}\widetilde{\boldsymbol{w}}_{i-1} \\ \widetilde{\boldsymbol{\psi}}_i = \left[\mathcal{A}_o^{\mathsf{T}} - \mathcal{M}\boldsymbol{\mathcal{R}}_i\right]\widetilde{\boldsymbol{\phi}}_{i-1} - \mathcal{M}\boldsymbol{z}_i \\ \widetilde{\boldsymbol{w}}_i = \mathcal{A}_2^{\mathsf{T}}\widetilde{\boldsymbol{\psi}}_i \end{cases} \tag{77a}$$

so that the network weight error vector $\widetilde{\boldsymbol{w}}_i$ evolves according to:

$$\widetilde{\boldsymbol{w}}_i = \mathcal{A}_2^{\mathsf{T}}\left(\mathcal{A}_o^{\mathsf{T}} - \mathcal{M}\boldsymbol{\mathcal{R}}_i\right)\mathcal{A}_1^{\mathsf{T}}\widetilde{\boldsymbol{w}}_{i-1} - \mathcal{A}_2^{\mathsf{T}}\mathcal{M}\boldsymbol{z}_i. \tag{77b}$$

For comparison purposes, if each agent operates individually and uses the noncooperative strategy (13f), then the weight error vector would instead evolve according to the following recursion:

$$\widetilde{\boldsymbol{w}}_i = (I_{MN} - \mathcal{M}\boldsymbol{\mathcal{R}}_i)\widetilde{\boldsymbol{w}}_{i-1} - \mathcal{M}\boldsymbol{z}_i, \qquad i \geq 0 \tag{77c}$$

where the matrices $\{\mathcal{A}_o, \mathcal{A}_1, \mathcal{A}_2\}$ do not appear any longer and where the coefficient matrix $(I_{MN} - \mathcal{M}\boldsymbol{\mathcal{R}}_i)$ becomes block diagonal. For later reference, it is straightforward to verify that recursion (77b) can be equivalently rewritten in the following form in terms of the gradient noise vector $\boldsymbol{s}_i$, defined by (77g):

$$\widetilde{\boldsymbol{w}}_i = \mathcal{B}\widetilde{\boldsymbol{w}}_{i-1} + \mathcal{A}_2^{\top}\mathcal{M}\boldsymbol{s}_i \tag{78}$$

where we introduced the constant matrices

$$\mathcal{B} \overset{\Delta}{=} \mathcal{A}_2^{\top}\left(\mathcal{A}_o^{\top} - \mathcal{M}\mathcal{R}\right)\mathcal{A}_1^{\top} \tag{79a}$$

$$\mathcal{R} \overset{\Delta}{=} \mathbb{E}\boldsymbol{\mathcal{R}}_i = \text{diag}\{2R_{u,1},\ 2R_{u,2},\ \ldots,\ 2R_{u,N}\}. \tag{79b}$$

♦

Example 20 (Mean Error Behavior): We continue with the setting of Example 19. In MSE analysis, we are interested in examining how the quantities $\mathbb{E}\widetilde{\boldsymbol{w}}_i$ and $\mathbb{E}\|\widetilde{\boldsymbol{w}}_i\|^2$ evolve over time. If we refer back to the data model described in Example 9, where the regression data $\{\boldsymbol{u}_{k,i}\}$ were assumed to be temporally white and independent over space, then the stochastic matrix $\boldsymbol{\mathcal{R}}_i$ appearing in (77b)–(79b) is seen to be statistically independent of $\widetilde{\boldsymbol{w}}_{i-1}$. Therefore, taking expectations of both sides of these recursions, and invoking the fact that $\boldsymbol{u}_{k,i}$ and $\boldsymbol{v}_k(i)$ are also independent of each other and have zero means (so that $\mathbb{E}\boldsymbol{z}_i = 0$), we conclude that the mean error vectors evolve according to the following recursions:

$$\mathbb{E}\widetilde{\boldsymbol{w}}_i = \mathcal{B}(\mathbb{E}\widetilde{\boldsymbol{w}}_{i-1}) \quad \text{(distributed)} \tag{80a}$$

$$\mathbb{E}\widetilde{\boldsymbol{w}}_i = (I_{MN} - \mathcal{M}\mathcal{R})(\mathbb{E}\widetilde{\boldsymbol{w}}_{i-1}) \quad \text{(noncooperative).} \tag{80b}$$

The matrix $\mathcal{B}$ controls the dynamics of the mean weight/error vector for the distributed strategies. Observe, in particular, that $\mathcal{B}$ reduces to the following forms for the various strategies [noncooperative (13f), consensus (47), CTA diffusion (49a), and ATC diffusion (49b)]:

$$\mathcal{B}_{\text{ncop}} = I_{MN} - \mathcal{M}\mathcal{R} \tag{81a}$$

$$\mathcal{B}_{\text{cons}} = \mathcal{A}^{\top} - \mathcal{M}\mathcal{R} \tag{81b}$$

$$\mathcal{B}_{\text{atc}} = \mathcal{A}^{\top}(I_{MN} - \mathcal{M}\mathcal{R}) \tag{81c}$$

$$\mathcal{B}_{\text{cta}} = (I_{MN} - \mathcal{M}\mathcal{R})\mathcal{A}^{\top} \tag{81d}$$

where $\mathcal{A} = A \otimes I_M$. ♦

Example 21 (MSE Networks With Uniform Agents): The results of Example 20 simplify when all agents employ the same step size $\mu_k \equiv \mu$ and observe regression data with the same covariance matrix $R_{u,k} \equiv R_u$ [11], [99]. In this case, we can express $\mathcal{M}$ and $\mathcal{R}$ from (76b) and (79b) in Kronecker product form as follows:

$$\mathcal{M} = \mu I_N \otimes I_M, \quad \mathcal{R} = I_N \otimes 2R_u \tag{82}$$

so that expressions (81a)–(81d) reduce to

$$\mathcal{B}_{\text{ncop}} = I_N \otimes (I_M - 2\mu R_u) \tag{83a}$$

$$\mathcal{B}_{\text{cons}} = A^{\top} \otimes I_M - 2\mu(I_M \otimes R_u) \tag{83b}$$

$$\mathcal{B}_{\text{atc}} = A^{\top} \otimes (I_M - 2\mu R_u) \tag{83c}$$

$$\mathcal{B}_{\text{cta}} = A^{\top} \otimes (I_M - 2\mu R_u). \tag{83d}$$

Observe that $\mathcal{B}_{\text{atc}} = \mathcal{B}_{\text{cta}}$, so we denote these matrices by $\mathcal{B}_{\text{diff}}$. Using properties of the eigenvalues of Kronecker products of matrices, it can be easily verified that the MN eigenvalues of the above $\mathcal{B}$ matrices are given by the following expressions in terms of the eigenvalues of the component matrices $\{A, R_u\}$ for $k = 1, 2, \ldots N$ and $m = 1, 2, \ldots, M$:

$$\lambda(\mathcal{B}_{\text{ncop}}) = 1 - 2\mu\lambda_m(R_u) \tag{84a}$$

$$\lambda(\mathcal{B}_{\text{cons}}) = \lambda_k(A) - 2\mu\lambda_m(R_u) \tag{84b}$$

$$\lambda(\mathcal{B}_{\text{diff}}) = \lambda_k(A)[1 - 2\mu\lambda_m(R_u)]. \tag{84c}$$

♦

Example 22 (Potential Instability in Consensus Networks): Consensus strategies can become unstable when used for adaptation purposes [99]. This undesirable effect is already reflected in expressions (84a)–(84c). In particular, observe that the eigenvalues of A appear multiplying $(1 - 2\mu\lambda_m(R_u))$ in expression (84c) for diffusion. As such, and since $\rho(A) = 1$ for any left-stochastic matrix, we conclude for this case of uniform agents that $\rho(\mathcal{B}_{\text{diff}}) = \rho(\mathcal{B}_{\text{ncop}})$. It follows that, regardless of the choice of the combination policy A, the diffusion strategies will be stable in the mean (i.e., $\mathbb{E}\widetilde{\boldsymbol{w}}_i$ will converge asymptotically to zero) whenever the individual noncooperative agents are stable in the mean

$$\text{individual agents stable} \Longrightarrow \text{diffusion networks stable.} \tag{85a}$$

The same conclusion is not true for consensus networks; the individual agents can be stable and yet the consensus network can become unstable. This is because $\lambda_k(A)$

appears as an additive (rather than multiplicative) term in (84b) (see [10], [99] for examples)

$$\text{individual agents stable} \not\Rightarrow \text{consensus networks stable.} \tag{85b}$$

The fact that the combination matrix $\mathcal{A}^\top$ appears in an additive form in (40a) is the result of the asymmetry that was mentioned earlier in the update equation for the consensus strategy. In contrast, the update equations for the diffusion strategies lead to $\mathcal{A}^\top$ appearing in a multiplicative form in (81c) and (81d). ♦

B. Unified Description of Distributed Strategies

The results in the previous section focus on MSE networks, which deal with MSE cost functions. We can similarly capture the noncooperative strategy from (40a), the consensus strategy from (46c), and the diffusion strategies from (48a) and (48b) for more general cost functions by using a unifying description as follows [33], [47]:

$$\begin{cases} \boldsymbol{\phi}_{k,i-1} = \sum\limits_{\ell\in\mathcal{N}_k} a_{1,\ell k}\boldsymbol{w}_{\ell,i-1} \\ \boldsymbol{\psi}_{k,i} = \sum\limits_{\ell\in\mathcal{N}_k} a_{o,\ell k}\boldsymbol{\phi}_{\ell,i-1} - \mu_k\widehat{\nabla_{w^\top}J}_k(\boldsymbol{\phi}_{k,i-1}) \\ \boldsymbol{w}_{k,i} = \sum\limits_{\ell\in\mathcal{N}_k} a_{2,\ell k}\boldsymbol{\psi}_{\ell,i}. \end{cases} \tag{86}$$

We again let $P = A_1A_oA_2$ and assume it is a *primitive* matrix. For example, this condition is automatically guaranteed if the combination matrix A in the selections (74b)–(74d) is primitive, which in turn is guaranteed for strongly connected networks. The objective is to analyze how close the weight iterates $\{\boldsymbol{w}_{k,i}\}$ of the distributed strategy (86) get to the limit point $w^\star$, defined by (57c). Since we are now dealing with a general description that involves three combination matrices $\{A_o, A_1, A_2\}$, the definition of the scaling factors q_k from (57a) needs to be adjusted [33]. First, the right eigenvector p from (56) is now defined as the Perron eigenvector of the matrix P itself, namely

$$Pp = p, \quad \mathbb{1}^\top p = 1, \quad p_k > 0, \quad k = 1,2,\ldots,N. \tag{87a}$$

Second, the coefficients $\{q_k\}$ from (57a) are now defined as the entries of the vector

$$q \overset{\Delta}{=} \text{diag}\{\mu_1,\mu_2,\ldots,\mu_N\}A_2p. \tag{87b}$$

Obviously, the above two expressions for p and q reduce to (56) and (57a) for the selections (74b)–(74d), which is the situation we considered earlier in Section VI. We also represent the step sizes as scaled multiples of the same factor $\mu_{\max}$, namely, $\mu_k = \gamma_k\mu_{\max}$, where $0<\gamma_k \leq 1$. In this way, it becomes clear that all step sizes become smaller as $\mu_{\max}$ is reduced in size.

The reference vector $w^\star$ is still chosen as the unique global minimizer $w^\star$ of the same weighted aggregate function (57b). Accordingly, we associate the following error vectors with the distributed strategy (86):

$$\widetilde{\boldsymbol{w}}_{k,i} \overset{\Delta}{=} w^\star - \boldsymbol{w}_{k,i} \tag{88a}$$

$$\widetilde{\boldsymbol{\psi}}_{k,i} \overset{\Delta}{=} w^\star - \boldsymbol{\psi}_{k,i} \tag{88b}$$

$$\widetilde{\boldsymbol{\phi}}_{k,i-1} \overset{\Delta}{=} w^\star - \boldsymbol{\phi}_{k,i-1}. \tag{88c}$$

In a manner similar to Assumption II.1, we assume that the original aggregate cost function $J^{\text{glob}}(w)$, defined by (37), satisfies the following condition in the distributed case.

Assumption VII.1 (Conditions on Cost Function): The aggregate cost $J^{\text{glob}}(w)$ is twice differentiable and satisfies a condition similar to (9) for some positive parameters $\nu_d \leq \delta_d$. In particular, all individual costs $J_k(w)$ are assumed to be convex with at least one of them being strongly convex. ♦

C. Gradient Noise Model

With each agent k, we again associate a gradient noise vector and, additionally, introduce a mismatch (or bias) vector

$$\boldsymbol{s}_{k,i}(\boldsymbol{\phi}_{k,i-1}) \overset{\Delta}{=} \widehat{\nabla_{w^\top}J}_k(\boldsymbol{\phi}_{k,i-1}) - \nabla_{w^\top}J_k(\boldsymbol{\phi}_{k,i-1}) \tag{89a}$$

$$b_k \overset{\Delta}{=} -\nabla_{w^\top}J_k(w^\star). \tag{89b}$$

In the special case when all individual costs $J_k(w)$ have the same minimizer at $w_k^o \equiv w^o$ (which is the situation considered in Example 19), then $w^\star = w^o$ and the vector b_k will be identically zero. In general, though, the vector b_k is nonzero. We again introduce the limiting covariance matrix:

$$R_{s,k} \overset{\Delta}{=} \lim_{i\to\infty}\mathbb{E}\left[\boldsymbol{s}_{k,i}(w^\star)\boldsymbol{s}_{k,i}^\top(w^\star)|\boldsymbol{\mathcal{F}}_{i-1}\right] \tag{90}$$

where $\boldsymbol{\mathcal{F}}_{i-1}$ represents the collection of all random events generated by the processes $\{\boldsymbol{w}_{k,j}\}$ at all agents $k = 1,2,\ldots,N$ up to time $i-1$

$$\boldsymbol{\mathcal{F}}_{i-1} \overset{\Delta}{=} \text{filtration}\{\boldsymbol{w}_{k,-1},\boldsymbol{w}_{k,0},\boldsymbol{w}_{k,1},\ldots,\boldsymbol{w}_{k,i-1},\ \text{all } k\}. \tag{91}$$

Similarly to Assumption II.2, we assume that the gradient noise processes across the agents satisfy the following conditions.

Assumption VII.2 (Conditions on Gradient Noise): It is assumed that the first- and second-order conditional moments of the gradient noise components satisfy (90) and

$$\mathbb{E}\left[\boldsymbol{s}_{k,i}(\boldsymbol{\phi}_{k,i-1})|\boldsymbol{\mathcal{F}}_{i-1}\right] = 0 \tag{92a}$$

$$\mathbb{E}\left[\boldsymbol{s}_{k,i}(\boldsymbol{\phi}_{k,i-1})\boldsymbol{s}_{\ell,i}^{\top}(\boldsymbol{\phi}_{\ell,i-1})|\boldsymbol{\mathcal{F}}_{i-1}\right]=0, \quad \text{any } k\neq\ell \tag{92b}$$

$$\mathbb{E}\left[\|\boldsymbol{s}_{k,i}(\boldsymbol{\phi}_{k,i-1})\|^2|\boldsymbol{\mathcal{F}}_{i-1}\right] \leq \beta_k^2\|\widetilde{\boldsymbol{\phi}}_{k,i-1}\|^2 + \sigma_{s,k}^2 \tag{92c}$$

almost surely, and for some nonnegative scalars β_k^2 and $\sigma_{s,k}^2$.

♦

It follows from the above conditions that

$$\mathbb{E}\left\|\boldsymbol{s}_{k,i}(\boldsymbol{\phi}_{k,i-1})\right\|^2 \leq \beta_k^2\mathbb{E}\|\widetilde{\boldsymbol{\phi}}_{k,i-1}\|^2 + \sigma_{s,k}^2. \tag{93}$$

D. Network Error Dynamics

We collect the error vectors, gradient noises, and mismatch vectors from across all agents into $N \times 1$ *block* vectors, whose individual entries are of size $M \times 1$ each

$$\widetilde{\boldsymbol{w}}_i \overset{\Delta}{=} \begin{bmatrix} \widetilde{\boldsymbol{w}}_{1,i} \\ \widetilde{\boldsymbol{w}}_{2,i} \\ \vdots \\ \widetilde{\boldsymbol{w}}_{N,i} \end{bmatrix}, \quad \boldsymbol{s}_i \overset{\Delta}{=} \begin{bmatrix} \boldsymbol{s}_{1,i} \\ \boldsymbol{s}_{2,i} \\ \vdots \\ \boldsymbol{s}_{N,i} \end{bmatrix}, \quad b \overset{\Delta}{=} \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{bmatrix} \tag{94}$$

where we are dropping the argument $\boldsymbol{\phi}_{k,i-1}$ from $\boldsymbol{s}_{k,i}(\cdot)$ for compactness of notation. Likewise, we introduce the following $N \times N$ *block* diagonal matrices, whose individual entries are of size $M \times M$ each:

$$\mathcal{M} \overset{\Delta}{=} \text{diag}\{\mu_1 I_M, \mu_2 I_M, \ldots, \mu_N I_M\} \tag{95a}$$

$$\boldsymbol{\mathcal{H}}_{i-1} \overset{\Delta}{=} \text{diag}\{\boldsymbol{H}_{1,i-1}, \boldsymbol{H}_{2,i-1}, \ldots, \boldsymbol{H}_{N,i-1}\} \tag{95b}$$

where

$$\boldsymbol{H}_{k,i-1} \overset{\Delta}{=} \int_0^1 \nabla_w^2 J_k(w^\star - t\widetilde{\boldsymbol{\phi}}_{k,i-1})dt. \tag{95c}$$

Now, in a manner similar to (22b), we can appeal to the mean value theorem [26], [48] to note that

$$\nabla_{w^\top} J_k(\boldsymbol{\phi}_{k,i-1}) = -b_k - \boldsymbol{H}_{k,i-1}\widetilde{\boldsymbol{\phi}}_{k,i-1} \tag{96}$$

so that the approximate gradient vector can be expressed as

$$\widehat{\nabla_{w^\top} J}_k(\boldsymbol{\phi}_{k,i-1}) = -b_k - \boldsymbol{H}_{k,i-1}\widetilde{\boldsymbol{\phi}}_{k,i-1} + \boldsymbol{s}_{k,i}(\boldsymbol{\phi}_{k,i-1}). \tag{97}$$

Subtracting $w^\star$ from both sides of (86), and using (97), we find that the network error vector evolves according to the following stochastic recursion:

$$\widetilde{\boldsymbol{w}}_i = \boldsymbol{\mathcal{B}}_{i-1}\widetilde{\boldsymbol{w}}_{i-1} + \mathcal{A}_2^\top \mathcal{M}\boldsymbol{s}_i - \mathcal{A}_2^\top \mathcal{M} b \tag{98a}$$

where

$$\boldsymbol{\mathcal{B}}_{i-1} \overset{\Delta}{=} \mathcal{A}_2^\top\left(\mathcal{A}_o^\top - \mathcal{M}\boldsymbol{\mathcal{H}}_{i-1}\right)\mathcal{A}_1^\top. \tag{98b}$$

Recursion (98a) describes the evolution of the network error vector for general convex costs $J_k(w)$ in a manner similar to recursion (78) in the MSE case. However, recursion (98a) is more challenging to deal with because of the randomness of $\boldsymbol{\mathcal{B}}_{i-1}$ and the presence of the bias term arising from b. The reason why recursion (78) involves a constant coefficient matrix $\mathcal{B}$ is because that example deals with MSE networks where the individual costs $J_k(w)$ are quadratic in w and, therefore, their Hessian matrices are constant and independent of w. In that case, each matrix $\boldsymbol{H}_{k,i-1}$ in (95c) will evaluate to $2R_{u,k}$ and the matrix $\boldsymbol{\mathcal{H}}_{i-1}$ in (98b) will coincide with the matrix $\mathcal{R}$ defined by (79b).

E. MSE Stability

The next statement ascertains that sufficiently small step sizes exist that guarantee the MSE stability of the network learning process [33].

Lemma VIII.1 (MSE Network Stability): Consider a network of N interacting agents running the distributed strategy (86) with a primitive matrix $P = A_1 A_o A_2$. Assume the conditions under Assumptions VIII.1 and VIII.2 hold. Then, there exists $\mu_o > 0$ such that for all $\mu_{\max} < \mu_o$

$$\limsup_{i\to\infty} \mathbb{E}\|\widetilde{\boldsymbol{w}}_{k,i}\|^2 = O(\mu_{\max}). \tag{99}$$

Proof: The argument is demanding, and we only provide a sketch of the proof. The details appear in [1] and [33]. We first note that it follows from (57c) and (87b) that $(q^\top \otimes I_M)b = 0$. Next, since P is left stochastic and primitive, it admits a Jordan canonical decomposition of the form $P = VJV^{-1}$ where [59, p. 128]

$$J = \begin{bmatrix} 1 & 0 \\ 0 & J_\epsilon \end{bmatrix} \quad V = [p \quad V_R] \quad V^{-1} = \begin{bmatrix} \mathbb{1}^\top \\ V_L^\top \end{bmatrix}. \tag{100}$$

Matrix J_ϵ consists of Jordan blocks with the unit entries above the diagonal replaced by ϵ, where ϵ is a positive number that can be chosen arbitrarily small. Moreover, all eigenvalues of J_ϵ have magnitude smaller than one and p is the Perron eigenvector of P. We introduce the transformed weight error vector

$$(V^\top \otimes I_M)\widetilde{\boldsymbol{w}}_i = \begin{bmatrix} (p^\top \otimes I_M)\widetilde{\boldsymbol{w}}_i \\ (V_R^\top \otimes I_M)\widetilde{\boldsymbol{w}}_i \end{bmatrix} \overset{\Delta}{=} \begin{bmatrix} \bar{\boldsymbol{w}}_i \\ \check{\boldsymbol{w}}_i \end{bmatrix}. \tag{101}$$

Starting from (98a), using $(q^\top \otimes I_M)b = 0$ and the above transformation, some extended algebra will then show that the mean square norms of the transformed error vectors evolve according to the following dynamics for sufficiently small step sizes:

$$\begin{bmatrix} \mathbb{E}\|\bar{\boldsymbol{w}}_i\|^2 \\ \mathbb{E}\|\check{\boldsymbol{w}}_i\|^2 \end{bmatrix} \preceq \Gamma \cdot \begin{bmatrix} \mathbb{E}\|\bar{\boldsymbol{w}}_{i-1}\|^2 \\ \mathbb{E}\|\check{\boldsymbol{w}}_{i-1}\|^2 \end{bmatrix} + \begin{bmatrix} O(\mu_{\max}^2) \\ O(\mu_{\max}^2) \end{bmatrix} \tag{102}$$

where the notation $\preceq$ denotes element-wise inequality, and where the entries of the 2×2 matrix Γ are in the order of

$$\Gamma = \begin{bmatrix} 1 - O(\mu_{\max}) & O(\mu_{\max}) \\ O(\mu_{\max}^2) & \rho(J_\epsilon) + \epsilon + O(\mu_{\max}^2) \end{bmatrix}. \tag{103}$$

Now, since the spectral radius of a matrix is bounded by any of its norms [59], we can use the maximum column sum norm of Γ to conclude that $\rho(\Gamma) < 1$ for sufficiently small $\mu_{\max}$ and ϵ. Subsequently, if we iterate (102), we get

$$\begin{aligned} \limsup_{i\to\infty} \mathbb{E}\|\bar{\boldsymbol{w}}_i\|^2 &= O(\mu_{\max}) \\ \limsup_{i\to\infty} \mathbb{E}\|\check{\boldsymbol{w}}_i\|^2 &= O(\mu_{\max}^2) \end{aligned} \tag{104}$$

from which (99) follows. ■

F. MSE Performance

Result (99) shows that the mean square deviation of the network is in the order of $O(\mu_{\max})$. Therefore, sufficiently small step sizes lead to sufficiently small MSEs. As was the case with the discussion following (21c) in the single-agent case, we can also seek a closed-form expression for the MSD performance of the network and its agents. We again rely on the energy conservation technique of [16], [19], [20], [42], [43], and [47]. For that purpose, we first introduce the analog of Assumption II.3 for the network case.

Assumption VIII.3 (Smoothness Conditions): The Hessian matrix of the individual cost functions $J_k(w)$, and the noise covariance matrices defined for each agent in a manner similar to (17a) and denoted by $R_{s,k,i}$, are assumed to be locally Lipschitz continuous in a small neighborhood around $w = w^\star$:

$$\left\|\nabla_w^2 J_k(w^\star + \delta w) - \nabla_w^2 J_k(w^\star)\right\| \le \tau_{k,d}\|\delta w\| \tag{105a}$$

$$\|R_{s,k,i}(w^\star + \delta w) - R_{s,k,i}(w^\star)\| \le \tau_{k,2}\|\delta w\|^\kappa \tag{105b}$$

for small perturbations $\|\delta w\| \le r_d$ and for some $\tau_{k,d}, \tau_{k,2} \ge 0$ and $1 \le \kappa \le 2$. ♦

Following the same argument that was used in the single-agent case to motivate (28), we can again argue that the asymptotic mean square performance of the error recursion (98a) can be assessed by considering the following long-term model, which holds with high probability after sufficient iterations $i \gg 1$:

$$\widetilde{\boldsymbol{w}}_i = \mathcal{B}\widetilde{\boldsymbol{w}}_{i-1} + \mathcal{A}_2^\top \mathcal{M}\boldsymbol{s}_i - \mathcal{A}_2^\top \mathcal{M} b + O(\mu_{\max}^2) \tag{106a}$$

where the constant matrix $\mathcal{B}$ is defined by

$$\mathcal{B} \overset{\Delta}{=} \mathcal{A}_2^\top(\mathcal{A}_o^\top - \mathcal{M}\mathcal{H})\mathcal{A}_1^\top \tag{106b}$$

$$\mathcal{H} \overset{\Delta}{=} \mathrm{diag}\{H_1, H_2, \ldots, H_N\} \tag{106c}$$

$$H_k \overset{\Delta}{=} \nabla_w^2 J_k(w^\star). \tag{106d}$$

Using the fact that $P = A_1 A_o A_2$ is primitive, it can be proven using eigenvalue perturbation analysis that the spectral radius of $\mathcal{B}$ is given by [1], [15], [32], [33]

$$\begin{aligned} \rho(\mathcal{B}) &= 1 - \lambda_{\min}\left(\sum_{k=1}^N q_k H_k\right) + O\left(\mu_{\max}^{(N+1)/N}\right) \\ &= 1 - O(\mu_{\max}) \end{aligned} \tag{106e}$$

so that $\mathcal{B}$ is stable for small step sizes. Comparing (98a) and (106a), we see that the stochastic matrix $\boldsymbol{\mathcal{B}}_{i-1}$ in (98a) is now replaced by the constant matrix $\mathcal{B}$ in (106a). Therefore, working with iteration (106a) is more convenient than working with the original recursion (98a). As was the case with the single-agent scenario, if desired, it can be shown that, under some technical conditions on the fourth-order moments of the gradient noise processes, $\{\boldsymbol{s}_{k,i}(\boldsymbol{\phi}_{k,i-1})\}$, the MSD expression that would result from using (106a) is within $O(\mu_{\max}^{3/2})$ of the actual MSD expression for the original recursion (98a) [47]. We, therefore, continue with (106a).

To begin with, we first note from (106a) that, under expectation, and for $i \gg 1$

$$\mathbb{E}\widetilde{\boldsymbol{w}}_i = \mathcal{B}(\mathbb{E}\widetilde{\boldsymbol{w}}_{i-1}) - \mathcal{A}_2^\top \mathcal{M} b + O(\mu_{\max}^2) \tag{106f}$$

and, hence

$$(I - \mathcal{B})(\lim_{i\to\infty} \mathbb{E}\widetilde{\boldsymbol{w}}_i) = -\mathcal{A}_2^\top \mathcal{M} b + O(\mu_{\max}^2). \tag{107}$$

It is seen that the size of the bias is dependent on the value of b, whose entries are constructed from the gradient information $\nabla_k J_k(w^\star)$ at all agents. Using the fact that $w^\star$

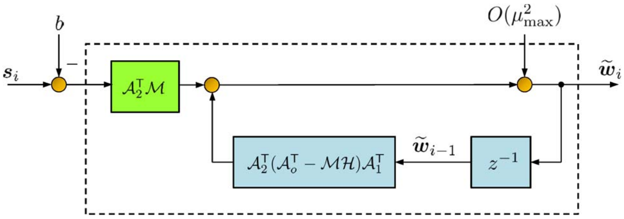
b
$O(\mu_{\max}^2)$
s_i
$-$
$\mathcal{A}_2^\mathsf{T}\mathcal{M}$
$\widetilde{w}_i$
$\mathcal{A}_2^\mathsf{T}(\mathcal{A}_o^\mathsf{T}-\mathcal{MH})\mathcal{A}_1^\mathsf{T}$
$\widetilde{w}_{i-1}$
z^{-1}

Fig. 6. ***Block diagram representation for the long-term network recursion (106a).***

satisfies (57c), it can be verified, by multiplying both sides of the last equation by $(I-\mathcal{B})^{-1}$ that [41, App. B]

$$\lim_{i\to\infty}\mathbb{E}\widetilde{w}_i = O(\mu_{\max}). \tag{108}$$

Fig. 6 provides a block-diagram representation for recursion (106a). Comparing with the earlier block diagram in Fig. 1, we notice the effect of the network connectivity, which is reflected through the appearance of the combination policies $\{\mathcal{A}_o,\mathcal{A}_1,\mathcal{A}_2\}$ and the bias factor b.

To evaluate the performance of the individual agents and the network, we introduce the centered variable

$$\widetilde{w}_i^c \overset{\Delta}{=} \widetilde{w}_i - \mathbb{E}\widetilde{w}_i. \tag{109a}$$

Subtracting (106f) from (106a), we find that this centered variable satisfies the following recursion for $i \gg 1$:

$$\widetilde{w}_i^c = \mathcal{B}\widetilde{w}_{i-1}^c + \mathcal{A}_2^\top\mathcal{M}s_i + O\left(\mu_{\max}^2\right) \tag{109b}$$

where the constant driving term $\mathcal{A}_2^\top\mathcal{M}b$ has been removed. Although we are interested in evaluating the asymptotic value of $\mathbb{E}\|\widetilde{w}_i\|^2$, we can still rely on the centered variable $\widetilde{w}_i^c$ for this purpose since it follows from (108) and (109a) that

$$\mathrm{MSD}_{\mathrm{dist,av}} = \lim_{i\to\infty}\left(\frac{1}{N}\mathbb{E}\|\widetilde{w}_i^c\|^2\right) + O\left(\mu_{\max}^2\right). \tag{110}$$

Let Σ denote an $N\times N$ block Hermitian nonnegative–definite matrix that we are free to choose, with $M\times M$ block entries. Equating the weighted square norms on both sides of (109b), and taking expectations, we obtain that the following variance relation holds for $i\gg 1$ [1], [16], [19], [47]:

$$\mathbb{E}\left\|\widetilde{w}_i^c\right\|_\Sigma^2 = \mathbb{E}\left\|\widetilde{w}_{i-1}^c\right\|_{\mathcal{B}^\top\Sigma\mathcal{B}}^2 + \mathrm{Tr}(\Sigma\mathcal{Y}) + \Sigma\cdot O\left(\mu_{\max}^{5/2}\right) \tag{111a}$$

where

$$\mathcal{Y} \overset{\Delta}{=} \mathcal{A}_2^\top\mathcal{MSMA}_2 \tag{111b}$$

$$\mathcal{S} \overset{\Delta}{=} \mathrm{diag}\{R_{s,1},R_{s,2},\ldots,R_{s,N}\}. \tag{111c}$$

Lemma VIII.2 (Network MSD Performance): Consider a network of N interacting agents running the distributed strategy (86) with a primitive matrix $P=A_1A_oA_2$. Assume the conditions under Assumptions VIII.1, VIII.2, and VIII. 3 hold. Assume further that the step-size parameter is sufficiently small to ensure mean square stability, as already ascertained by Lemma VIII.1. Then

$$\mathrm{MSD}_{\mathrm{dist},k} \doteq \mathrm{MSD}_{\mathrm{dist,av}}$$
$$= \frac{1}{2}\mathrm{Tr}\left[\left(\sum_{k=1}^N q_kH_k\right)^{-1}\left(\sum_{k=1}^N q_k^2R_{s,k}\right)\right] + O\left(\mu_{\max}^{3/2}\right) \tag{112}$$

where $\{q_k\}$ denotes the entries of the vector q defined by (87b) in terms of the Perron vector p defined by (87a). Moreover, for large enough i, the convergence rate of the MSE, $\mathbb{E}\|\widetilde{w}_i\|^2$, toward its steady-state value is well approximated by the scalar $\alpha_{\mathrm{dist}}\in(0,1)$ given by

$$\alpha_{\mathrm{dist}} = 1-2\lambda_{\min}\left(\sum_{k=1}^N q_kH_k\right) + O\left(\mu_{\max}^{(N+1)/N}\right). \tag{113}$$

Proof: The argument requires some effort and we again provide a sketch of the main steps. The details appear in [1], [16], and [47]. Since $\mathcal{B}$ is stable for sufficiently small step sizes, by successively setting Σ in (111a) equal to the choices $\{I,\mathcal{B}^\top\mathcal{B},(\mathcal{B}^2)^\top\mathcal{B}^2,(\mathcal{B}^3)^\top\mathcal{B}^3,\ldots\}$, and by using (110), we get

$$\mathrm{MSD}_{\mathrm{dist,av}} = \frac{1}{N}\sum_{n=0}^{\infty}\mathrm{Tr}\left[\mathcal{B}^n\mathcal{Y}(\mathcal{B}^\top)^n\right] + O\left(\mu_{\max}^{3/2}\right). \tag{114a}$$

We can rewrite the series on the right-hand side more compactly by exploiting properties of the block Kronecker product operation $\otimes_b$ and the block vectorization operation bvec$(\cdot)$ [16], [142]; we note that the operation bvec(X) first vectorizes the $M\times M$ block entries of X and then stacks these columns on top of each other. Let $\mathcal{F}=\mathcal{B}^\top\otimes_b\mathcal{B}^\top$. Using the properties

$$\mathrm{bvec}(UCW) = (W^\top\otimes_b U)\mathrm{bvec}(C) \tag{114b}$$

$$\mathrm{Tr}(CW) = \left[\mathrm{bvec}(W^\top)\right]^\top\mathrm{bvec}(C) \tag{114c}$$

for matrices $\{U, W, C\}$ of compatible dimensions, we can readily establish the identity

$$\sum_{n=0}^{\infty}\mathrm{Tr}\left[\mathcal{B}^n\mathcal{Y}(\mathcal{B}^\top)^n\right]=\left[\mathrm{bvec}(\mathcal{Y}^\top)\right]^\top(I-\mathcal{F})^{-1}\mathrm{bvec}(I_{MN}). \tag{114d}$$

Using the fact that P is primitive, the following low-rank approximation can be derived [1], [16], [32], [58]:

$$(I-\mathcal{F})^{-1} = \left[(p\otimes p)(\mathbb{1}\otimes\mathbb{1})^\top\right]\otimes Z^{-1}+O(1) \tag{114e}$$

where

$$Z \overset{\Delta}{=} \sum_{k=1}^{N} q_k\left[(I_M\otimes H_k)+(H_k^\top\otimes I_M)\right]. \tag{114f}$$

Substituting (114e) into (114d), we arrive, after some algebra, at expression (112) for $\mathrm{MSD}_{\mathrm{dist,av}}$ [1], [16], [127]; similarly for $\mathrm{MSD}_{\mathrm{dist,k}}$. Finally, with regards to the convergence rate, we observe from (114a) that it is determined by $[\rho(\mathcal{B})]^2$, in terms of the square of the spectral radius of $\mathcal{B}$. The result then follows from (106e). ■

G. Excess Risk Performance

We can evaluate the ER metric (66) in a similar manner to the MSD. This computation is of interest when the individual cost functions are uniform across all agents $J_k(w)\equiv J(w)$. In this case, the same argument used in the proof of Lemma VIII.2 will show that [1], [16], [47]

$$\begin{aligned}\mathrm{ER}_{\mathrm{dist},k} &\doteq \mathrm{ER}_{\mathrm{dist,av}}\\ &= \frac{1}{4}\left(\sum_{k=1}^{N} q_k\right)^{-1}\mathrm{Tr}\left(\sum_{k=1}^{N} q_k^2 R_{s,k}\right)+O\left(\mu_{\max}^{3/2}\right)\end{aligned} \tag{115a}$$

with the convergence rate given by

$$\alpha_{\mathrm{dist}} = 1-2\left(\sum_{k=1}^{N} q_k\right)\lambda_{\min}(H)+O\left(\mu_{\max}^{(N+1)/N}\right) \tag{115b}$$

where $H=\nabla_w^2\, J(w^o)$.

Example 23 (Stabilizing Effect of Diffusion Networks): We can now revisit the conclusion of Example 22, which focused on MSE networks, and extend it to more general costs. We refer to the mean recursion (106f) and note from (106b) that the $\mathcal{B}$ matrices for the various strategies can be expressed in the following forms in terms of the $\mathcal{B}$ matrix for the noncooperative strategy:

$$\begin{aligned}
\mathcal{B}_{\mathrm{ncop}} &= I_{MN}-\mathcal{M}\mathcal{H} && \text{(noncooperation)} && \text{(116a)}\\
\mathcal{B}_{\mathrm{cons}} &= \mathcal{B}_{\mathrm{ncop}}+\left(\mathcal{A}^\top-I_{MN}\right) && \text{(consensus)} && \text{(116b)}\\
\mathcal{B}_{\mathrm{atc}} &= \mathcal{A}^\top\mathcal{B}_{\mathrm{ncop}} && \text{(ATC diffusion)} && \text{(116c)}\\
\mathcal{B}_{\mathrm{cta}} &= \mathcal{B}_{\mathrm{ncop}}\mathcal{A}^\top && \text{(CTA diffusion)} && \text{(116d)}
\end{aligned}$$

where $\mathcal{A}=A\otimes I_M$ and A is the left-stochastic combination matrix used by consensus or diffusion. Observe that the coefficient matrices $\{\mathcal{B}_{\mathrm{atc}}, \mathcal{B}_{\mathrm{cta}}\}$ for the diffusion strategies are expressed in terms of $\mathcal{B}_{\mathrm{ncop}}$ in a *multiplicative* manner, while $\mathcal{B}_{\mathrm{cons}}$ is related to $\mathcal{B}_{\mathrm{ncop}}$ in an *additive* manner. These structures have an important implication on mean stability in view of the following matrix result.

Let $\mathcal{X}_1$ and $\mathcal{X}_2$ be any left-stochastic matrices with blocks of size $M\times M$, and let $\mathcal{D}$ be any symmetric block-diagonal positive–definite matrix also with blocks of size $M\times M$. Then, it holds that $\rho(\mathcal{X}_2^\top\mathcal{D}\mathcal{X}_1^\top)\le\rho(\mathcal{D})$ [11, App. D]. That is, multiplication of $\mathcal{D}$ by left-stochastic transformations generally reduces the spectral radius. This result can be used to establish the mean stability of the diffusion networks whenever the noncooperative strategy is mean stable and regardless of the combination policy, A. Indeed, note that $\mathcal{B}_{\mathrm{ncop}}$ has a symmetric block-diagonal structure similar to $\mathcal{D}$ and that it is stable for any $\mu_{\max}<2/\rho(\mathcal{H})$. Matrix $\mathcal{A}$ in (116c) and (116d) plays the role of $\mathcal{X}_1$ or $\mathcal{X}_2$. Therefore, it follows that it will also hold that $\rho(\mathcal{B}_{\mathrm{atc}})<1$ and $\rho(\mathcal{B}_{\mathrm{cta}})<1$ for any $\mathcal{A}$. The same conclusion does not generally hold for $\mathcal{B}_{\mathrm{cons}}$ [99]. Note further that since $\rho(\mathcal{B}_{\mathrm{atc}})\le\rho(\mathcal{B}_{\mathrm{ncop}})$ and $\rho(\mathcal{B}_{\mathrm{cta}})\le\rho(\mathcal{B}_{\mathrm{ncop}})$, it follows that diffusion strategies have a stabilizing effect. ♦

IX. CONCLUDING REMARKS

This work provides an overview of strategies for adaptation, learning, and optimization over networks including noncooperative, centralized, incremental, consensus, and diffusion strategies. Particular attention is given to the constant step-size case in order to examine solutions that are able to adapt and learn continuously from streaming data. There are, of course, several other aspects of distributed strategies that are not covered in this work. Following [1] and [11], we comment briefly on some of these aspects with supporting references.

A. Noisy Exchanges of Information

We ignored in our presentation the effect of noise during the exchange of information among neighboring agents. To model noisy links, one can introduce an additive noise component into the steps involving the exchange of iterates. For example, in the diffusion LMS network of

Example 12, when some noise, say, $\boldsymbol{v}_{\ell k,i}^{(\psi)}$, interferes with the transmission of $\boldsymbol{\psi}_{\ell,i}$ from agent ℓ to agent k, then agent k ends up receiving the perturbed iterate

$$\boldsymbol{\psi}_{\ell k,i} = \boldsymbol{\psi}_{\ell,i} + \boldsymbol{v}_{\ell k,i}^{(\psi)} \tag{117a}$$

instead of $\boldsymbol{\psi}_{\ell,i}$. In this case, the actual ATC diffusion implementation will become

$$\boldsymbol{\psi}_{k,i} = \boldsymbol{\psi}_{k,i-1} + 2\mu_k \boldsymbol{u}_{k,i}^{\top}\left[\boldsymbol{d}_k(i) - \boldsymbol{u}_{k,i}\boldsymbol{\psi}_{k,i-1}\right] \tag{117b}$$

$$\boldsymbol{w}_{k,i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k}\boldsymbol{\psi}_{\ell k,i} \tag{117c}$$

with $\boldsymbol{\psi}_{\ell k,i}$ now appearing in the combination step (117c). Studying the degradation in performance that results from these noisy exchanges can be pursued by extending the mean square analysis of Section VIII. Readers can refer to [11], [140], [141], and [143]–[145] for results on diffusion strategies and to [146] and [147] for results on consensus strategies.

B. Gossip and Asynchronous Strategies

It is also possible to train networks whereby agents are not required to continually interact with all their neighbors at every iteration. Instead, agents may select a subset of their neighbors (or even a single neighbor). Criteria can be developed for determining which neighbors to select. One simple strategy is to pick one neighbor at a time randomly, which is the case with useful gossip implementations for distributed processing [88], [90], [148]–[152].

Moreover, an implicit assumption made in our presentation has been that all agents act synchronously. One can also study asynchronous implementations that are subject to random events such as random data arrival times, random agent failures, random link failures, random topology changes, etc. There exist several studies in the literature on the performance of consensus and gossip-type strategies in response to asynchronous events or changing topologies [84], [88], [119], [146]–[155]. There are also studies in the context of diffusion strategies [15], [150], [156]. In the works [15], [16], and [137], a fairly detailed analysis of asynchronous network behavior is carried out using techniques similar to what we presented for the synchronous case in this paper. For example, the ATC diffusion update (48b) in an asynchronous environment becomes

$$\begin{cases} \boldsymbol{\psi}_{k,i} = \boldsymbol{w}_{k,i-1} - \boldsymbol{\mu}_k(i)\widehat{\nabla_{w^{\top}} J_k}(\boldsymbol{w}_{k,i-1}) \\ \boldsymbol{w}_{k,i} = \sum\limits_{\ell \in \boldsymbol{\mathcal{N}}_{k,i}} \boldsymbol{a}_{\ell k}(i)\boldsymbol{\psi}_{\ell,i} \end{cases} \tag{118}$$

where the $\{\boldsymbol{\mu}_k(i), \boldsymbol{a}_{\ell k}(i)\}$ are now *time-varying* and *random* step sizes and combination coefficients, and $\boldsymbol{\mathcal{N}}_{k,i}$ denotes the *random* neighborhood of agent k at time i. The underlying network is, therefore, randomly varying. Two of the main results established in [15], [16], and [137] are that, under some independence conditions on the random events, the asynchronous network continues to be mean square stable for sufficiently small step sizes. Moreover, its convergence rate and MSD performance compare well to those of the synchronous network that is constructed by employing the average values for the step sizes and the average values for the combination coefficients, namely

$$\alpha_{\text{async}} = \alpha_{\text{sync}} + O\left(\mu_{\max}^{(N^2+1)/N^2}\right) \tag{119a}$$

$$\text{MSD}_{\text{async}} = \text{MSD}_{\text{sync}} + O(\mu_{\max}). \tag{119b}$$

In other words, the convergence rate remains largely unaffected by asynchronous events at the expense of a small deterioration in MSD performance. These results explain the remarkable robustness and resilience of cooperative networks in the face of random failures at multiple levels: agents, links, data, and topology.

C. Distributed Estimation with Sparsity Constraints

We may also consider distributed strategies that enforce sparsity constraints on the solution vector (e.g., [157]–[160]). For example, in the context of the MSE networks of Example 9, we may consider instead individual costs of the form

$$J_k(w) = \mathbb{E}(\boldsymbol{d}_k(i) - \boldsymbol{u}_{k,i}w)^2 + \gamma f(w) \tag{120}$$

where $f(w)$ is some real-valued convex function weighted by some scalar parameter $\gamma > 0$. The role of $f(w)$ is to ensure that the solution vector is sparse [161]–[163]. One ATC diffusion strategy for solving such problems takes the form [157]:

$$\boldsymbol{e}_k(i) = \boldsymbol{d}_k(i) - \boldsymbol{u}_{k,i}\boldsymbol{w}_{k,i-1} \tag{121a}$$

$$\boldsymbol{\psi}_{k,i} = \boldsymbol{w}_{k,i-1} + 2\mu_k \boldsymbol{u}_{k,i}^{\top}\boldsymbol{e}_k(i) - \mu_k\gamma\, \partial f(\boldsymbol{w}_{k,i-1}) \tag{121b}$$

$$\boldsymbol{w}_{k,i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k}\boldsymbol{\psi}_{\ell,i} \tag{121c}$$

where $\partial f(w)$ denotes a subgradient vector for $f(w)$ relative to w. Various possibilities exist for the selection of $f(w)$ and its subgradient vector. One choice is

$$\partial f(w) = \left[\frac{\text{sign}(w_1)}{\epsilon + |w_1|}\ \frac{\text{sign}(w_2)}{\epsilon + |w_2|}\ \cdots\ \frac{\text{sign}(w_M)}{\epsilon + |w_M|}\right] \tag{122}$$

where $\text{sign}(x) = \pm 1$, except when $x = 0$ for which we set its sign to zero. The choice (122) has the advantage of selectively shrinking those components of the iterate $\boldsymbol{w}_{k,i-1}$ whose magnitudes are comparable to ϵ with little effect on components whose magnitudes are much larger than ϵ [162], [164], [165]. Greedy techniques can also be used to develop useful sparsity-aware diffusion strategies, as shown in [159].

D. Distributed Constrained Optimization

Distributed strategies can also be developed for the solution of *constrained* convex optimization problems of the form

$$\min_{w} \sum_{k=1}^{N} J_k(w)$$
$$\text{subject to } w \in \mathbb{W}_1 \cap \mathbb{W}_2 \cap \ldots \cap \mathbb{W}_N \qquad (123a)$$

where each $J_k(w)$ is convex and each $\mathbb{W}_k$ is a convex set of points w that satisfy a collection of affine equality constraints and convex inequality constraints, say, as

$$\mathbb{W}_k \overset{\Delta}{=} \left\{ w : \begin{array}{ll} h_{k,m}(w) = 0, & m = 1, 2, \ldots, U_k \\ g_{k,n}(w) \leq 0, & n = 1, 2, \ldots, L_k. \end{array} \right. \qquad (123b)$$

The key challenge in solving such problems in a distributed manner is that each agent k should only be aware of its cost function $J_k(w)$ and its $L_k + U_k$ total constraints. For this reason, some available solution methods are in effect nondistributed because they require each agent to know all constraints from across the network [118]. If the feasible set and the constraints happen to be agent independent, then such solution methods become distributed.

More generally, when solving constrained optimization problems of the form (123a) in a distributed manner, it is customary to rely on the use of useful projection steps in order to ensure that the successive iterates that are computed by the agents satisfy the convex constraints; see, e.g., [119], [120], and [166]–[169]. An insightful overview of the use of projection methods in optimization problems is given in [167]. We illustrate the main idea of projection methods by considering one example from [168]. The example below does not solve (123a) and (123b) but deals instead with a different type of constraints. Specifically, assume for the case of the MSE network from Example 9 that we are interested in minimizing the same aggregate MSE cost under the additional constraint that $|\boldsymbol{d}_k(i) - \boldsymbol{u}_{k,i} w^o| \leq \epsilon_k$ for all agents k and $i \geq 0$. This formulation corresponds to a situation in which it is known beforehand that the measurement noise $\boldsymbol{v}_k(i)$ is bounded by some ϵ_k. Reference [168] starts from the CTA algorithm (49a) and seeks a viable solution w^o by incorporating a projection step between the combination step and the adaptation step, namely

$$\boldsymbol{\psi}_{k,i-1} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \boldsymbol{w}_{\ell,i-1} \qquad (124a)$$
$$\boldsymbol{\phi}_{k,i-1} = \mathcal{P}'_{k,i}[\boldsymbol{\psi}_{k,i-1}] \qquad (124b)$$
$$\boldsymbol{w}_{k,i} = \boldsymbol{\phi}_{k,i-1} - 2\mu_k \left\{ \boldsymbol{\phi}_{k,i-1} - \sum_{j=0}^{L-1} f_{kj} \mathcal{P}_{k,i-j}[\boldsymbol{\phi}_{k,i-1}] \right\} \qquad (124c)$$

where, for each k, the nonnegative coefficients $\{f_{k,j}\}$ add up to one. Moreover, the notation $\phi = \mathcal{P}_{k,i}[\psi]$ refers to projecting ψ onto the hyperslab $P_{k,i}$ that consists of all $M \times 1$ vectors z satisfying the constraint (similarly for the projection $\mathcal{P}'_{k,i}$)

$$P_{k,i} \overset{\Delta}{=} \left\{ z \text{ such that } \left| \boldsymbol{d}_k(i) - \boldsymbol{u}_{k,i} z \right| \leq \epsilon_k \right\} \qquad (125a)$$
$$P'_{k,i} \overset{\Delta}{=} \left\{ z \text{ such that } \left| \boldsymbol{d}_k(i) - \boldsymbol{u}_{k,i} z \right| \leq \epsilon'_k \right\} \qquad (125b)$$

where $\epsilon'_k > \epsilon_k > 0$ are tolerance parameters. For generic values $\{d, u, \epsilon\}$, where d is a scalar and u is a row vector, the projection operator is given by [167], [172]

$$\mathcal{P}[\psi] = \psi + \begin{cases} \frac{u^\top}{\|u\|^2}[d - \epsilon - u\psi], & \text{if } d - \epsilon > u\psi \\ \frac{u^\top}{\|u\|^2}[d + \epsilon - u\psi], & \text{if } d + \epsilon < u\psi \\ 0, & \text{if } |d - u\psi| \leq \epsilon. \end{cases} \qquad (126)$$

Returning to (123a) and (123b), solution techniques that rely on the use of similar projection operations require the constraint conditions to be relatively simple in order for the distributed algorithm to be able to compute the necessary projections analytically (such as projecting onto the nonnegative orthant) [119], [120], [166]. For example, the following form of the diffusion CTA strategy (48a) with projections is used in [120]:

$$\boldsymbol{\psi}_{k,i-1} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \boldsymbol{w}_{\ell,i-1} \qquad (127a)$$
$$\boldsymbol{\zeta}_{k,i} = \boldsymbol{\psi}_{k,i-1} - \mu(i) \nabla_{w^\top} J_k(\boldsymbol{\psi}_{k,i-1}) \qquad (127b)$$
$$\boldsymbol{w}_{k,i} = \mathcal{P}_{\mathbb{W}_k}[\boldsymbol{\zeta}_{k,i}]. \qquad (127c)$$

In this construction, the main motivation is to solve a static optimization problem (in lieu of adaptation and learning). Thus, note that the actual gradient vector is employed in (127b) along with a decaying step-size sequence. Moreover, the notation $\mathcal{P}_{\mathbb{W}_k}[\cdot]$ denotes projection onto the set $\mathbb{W}_k$; each of these sets is required to consist of "simple constraints" so that the projections can be carried out analytically. Motivated by these considerations, the work in [170] and [171] develops distributed strategies that circumvent projection steps. The solution relies on the use of suitably chosen penalty functions and replaces the projection step by a stochastic approximation update that runs simultaneously with the optimization step. One form of this diffusion solution can be described as follows. We select convex and twice-differentiable functions $\delta^{\mathrm{IP}}(x)$ and $\delta^{\mathrm{EP}}(x)$ that satisfy the properties

$$\begin{aligned} \delta^{\mathrm{IP}}(x) &= \begin{cases} 0, & x \le 0 \\ > 0, & x > 0 \end{cases} \\ \delta^{\mathrm{EP}}(x) &= \begin{cases} 0, & x = 0 \\ > 0, & x \neq 0. \end{cases} \end{aligned} \qquad (128a)$$

For example, the following continuous, convex, and twice-differentiable functions satisfy these conditions for small ρ:

$$\delta^{\mathrm{IP}}(x) = \max\left\{0, \frac{x^3}{\sqrt{x^2+\rho^2}}\right\}, \qquad \delta^{\mathrm{EP}}(x) = x^2. \qquad (128b)$$

Using the functions $\{\delta^{\mathrm{IP}}(x), \delta^{\mathrm{EP}}(x)\}$, we associate with each agent k the following penalty function, which takes into account all the constraints at the agent:

$$p_k(w) \triangleq \sum_{n=1}^{L_k} \delta^{\mathrm{IP}}\left(g_{k,n}(w)\right) + \sum_{m=1}^{U_k} \delta^{\mathrm{EP}}\left(h_{k,m}(w)\right). \qquad (129)$$

The penalized ATC diffusion form for solving (123a) then takes the following form for any $0 < \theta < 1$ [170], [171]:

$$\begin{aligned} \boldsymbol{\zeta}_{k,i} &= \boldsymbol{w}_{k,i-1} - \mu \widehat{\nabla_{w^\top} J_k}(\boldsymbol{w}_{k,i-1}) & (130a)\\ \boldsymbol{\psi}_{k,i} &= \boldsymbol{\zeta}_{k,i} - \mu^{1-\theta} \nabla_{w^\top} p_k(\boldsymbol{\zeta}_{k,i}) & (130b)\\ \boldsymbol{w}_{k,i} &= \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \boldsymbol{\psi}_{\ell,i}. & (130c) \end{aligned}$$

E. Diffusion Recursive Least Squares

We can also approach recursive least squares (RLS) problems in a distributed manner. For example, consider a collection of N agents observing deterministic streaming data $\{d_k(i), u_{k,i}\}$, which are assumed to be related via $d_k(i) = u_{k,i} w^o + v_k(i)$. Here, $u_{k,i}$ is a $1 \times M$ regression vector and w^o is the $M \times 1$ unknown vector to be estimated in a least squares sense by minimizing the regularized global cost

$$\min_w \; \lambda^{i+1}\delta \cdot \|w\|^2 + \sum_{j=0}^{i} \lambda^{i-j} \left(\sum_{k=1}^{N} \left(d_k(j) - u_{k,j} w\right)^2 \right) \qquad (131)$$

where $\delta > 0$ and $0 \ll \lambda \le 1$ is an exponential forgetting factor whose value is usually close to one. Consensus-type strategies for solving related least squares problems appear in [91], [109], and [126]. Diffusion-type strategies for addressing problems of the type (131) are developed in [114], [117], and [173], and they take the form listed below, where symbol $\leftarrow$ denotes an assignment. Scalars $\{a_{\ell k}, c_{\ell k}\}$ are nonnegative combination coefficients such that $A = [a_{\ell k}]$ is left stochastic and $C = [c_{\ell k}]$ is right stochastic. In the listing, $w_{k,i}$ denotes the estimate for w^o that is computed by agent k at time i. For every agent k, we start with the initial conditions $w_{k,-1} = 0$ and $P_{k,-1} = \delta^{-1} I_M$, where $P_{k,-1}$ is $M \times M$. The works [11] and [117] explain how this diffusion-based solution compares to the consensus-based solution of [91].

Diffusion RLS Strategy (ATC)

```
step 1 (initialization by agent $k$)
    $\psi_{k,i} \leftarrow w_{k,i-1}$
    $P_{k,i} \leftarrow \lambda^{-1} P_{k,i-1}$
step 2 (adaptation)
    Update $\{\psi_{k,i}, P_{k,i}\}$ by iterating over $\ell \in \mathcal{N}_k$:
        $\psi_{k,i} \leftarrow \psi_{k,i} + \frac{c_{\ell k} P_{k,i} u_{\ell,i}^\top}{1 + c_{\ell k} u_{\ell,i} P_{k,i} u_{\ell,i}^\top} [d_{\ell,i} - u_{\ell,i}\psi_{k,i}]$
        $P_{k,i} \leftarrow P_{k,i} - \frac{c_{\ell k} P_{k,i} u_{\ell,i}^\top u_{\ell,i} P_{k,i}}{1 + c_{\ell k} u_{\ell,i} P_{k,i} u_{\ell,i}^\top}$        (132)
    end
step 3 (combination)
    $w_{k,i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \psi_{\ell,i}$
```

F. Distributed State–Space Estimation

Distributed strategies can also be applied to the solution of state–space filtering and smoothing problems [82], [91], [109], [116], [174]–[179]. Thus, consider a network consisting of N agents observing the state vector $\boldsymbol{x}_i$, of size $n \times 1$ of some linear state–space model. At every time i, every agent k collects a measurement vector $\boldsymbol{y}_{k,i}$ of size $p \times 1$, which is related to the state vector as follows:

$$\begin{aligned} \boldsymbol{x}_{i+1} &= F_i \boldsymbol{x}_i + G_i \boldsymbol{n}_i & (133)\\ \boldsymbol{y}_{k,i} &= H_{k,i} \boldsymbol{x}_i + \boldsymbol{v}_{k,i}, \qquad k = 1, 2, \ldots, N. & (134) \end{aligned}$$

The processes $\boldsymbol{n}_i$ and $\boldsymbol{v}_{k,i}$ denote noises of sizes $n \times 1$ and $p \times 1$, respectively, and they are assumed to be zero mean, uncorrelated, and white, with covariance matrices denoted by $Q_i \ge 0$ and $R_{k,i} > 0$, respectively [21]. The initial state vector $\boldsymbol{x}_o$ is also assumed to be zero mean with covariance matrix $\mathbb{E}\boldsymbol{x}_o\boldsymbol{x}_o^\top = \Pi_o > 0$, and is uncorrelated with $\boldsymbol{n}_i$ and

$\boldsymbol{v}_{k,i}$ for all i and k. The parameter matrices $\{F_i, G_i, H_{k,i}, Q_i, R_{k,i}, \Pi_o\}$ are assumed to be known by node k. Let $\widehat{\boldsymbol{x}}_{k,i|j}$ denote a local estimator for $\boldsymbol{x}_i$ that is computed by agent k at time i based on local observations and on neighborhood data up to time j. The two listings below from [11] show two equivalent forms of a diffusion strategy proposed in [116], [176], and [177] to evaluate *approximate* predicted and filtered versions of these local estimators in a distributed manner.

Time and Measurement Form of Diffusion Kalman Filter

```
step 1 (initialization by agent k)
    $\psi_{k,i} \leftarrow \widehat{\boldsymbol{x}}_{k,i|i-1}$
    $P_{k,i} \leftarrow P_{k,i|i-1}$
step 2 (adaptation)
    Update $\{\psi_{k,i}, P_{k,i}\}$ by incorporating $\{\boldsymbol{y}_{\ell,i}\}$ from
    neighbors by iterating over $\ell \in \mathcal{N}_k$:
        $R_e \leftarrow R_{\ell,i} + H_{\ell,i} P_{k,i} H_{\ell,i}^\top$
        $\psi_{k,i} \leftarrow \psi_{k,i} + P_{k,i} H_{\ell,i}^\top R_e^{-1} [\boldsymbol{y}_{\ell,i} - H_{\ell,i}\psi_{k,i}]$
        $P_{k,i} \leftarrow P_{k,i} - P_{k,i} H_{\ell,i}^\top R_e^{-1} H_{\ell,i} P_{k,i}$
    end
step 3 (combination)
    $\widehat{\boldsymbol{x}}_{k,i|i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \psi_{\ell,i}$
    $P_{k,i|i} = P_{k,i}$
    $\widehat{\boldsymbol{x}}_{k,i+1|i} = F_i \widehat{\boldsymbol{x}}_{k,i|i}$
    $P_{k,i+1|i} = F_i P_{k,i|i} F_i^\top + G_i Q_i G_i^\top$
```

Information Form of the Diffusion Kalman Filter

```
step 1 (adaptation)
    $S_{k,i} = \sum_{\ell \in \mathcal{N}_k} H_{\ell,i}^\top R_{\ell,i}^{-1} H_{\ell,i}$
    $\boldsymbol{q}_{k,i} = \sum_{\ell \in \mathcal{N}_k} H_{\ell,i}^\top R_{\ell,i}^{-1} \boldsymbol{y}_{\ell,i}$
    $P_{k,i|i}^{-1} = P_{k,i|i-1}^{-1} + S_{k,i}$
    $\psi_{k,i} = \widehat{\boldsymbol{x}}_{k,i|i-1} + P_{k,i|i}[\boldsymbol{q}_{k,i} - S_{k,i}\widehat{\boldsymbol{x}}_{k,i|i-1}]$
step 2: (combination)
    $\widehat{\boldsymbol{x}}_{k,i|i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \psi_{\ell,i}$
    $\widehat{\boldsymbol{x}}_{k,i+1|i} = F_i \widehat{\boldsymbol{x}}_{k,i|i}$
    $P_{k,i+1|i} = F_i P_{k,i|i} F_i^\top + G_i Q_i G_i^\top$
```

The algorithms start with $\widehat{\boldsymbol{x}}_{k,0|-1} = 0$ and $P_{k,0|-1} = \Pi_o$, where $P_{k,0|-1}$ is $M \times M$. Step 1 in the second listing corresponding to the information form is similar to the update used in the consensus-based distributed Kalman filter derived in [175]. The main difference is that the weights $\{a_{\ell k}\}$ in [175] are all defined in terms of a single parameter $\epsilon > 0$ and set uniformly to $a_{\ell k} = \epsilon$ for $\ell \in \mathcal{N}_k \setminus \{k\}$ and $a_{kk} = 1 - (n_k - 1)\epsilon$, whereas the $\{a_{\ell k}\}$ are chosen more arbitrarily in the diffusion setting. ■

Acknowledgment

The author would like to thank his current and former Ph.D. students J. Chen, X. Zhao, Z. Towfic, S.-Y. Tu, C.-K. Yu, C. G. Lopes, F. S. Cattivelli, V. H. Nascimento, and R. Merched for their insightful contributions to the field and for thoughtful feedback on earlier drafts of this work.

REFERENCES

[1] A. H. Sayed, *Adaptation, Learning, and Optimization Over Networks*. Delft, The Netherlands: NOW Publishers, 2014, under review.

[2] S. Camazine, J. L. Deneubourg, N. R. Franks, J. Sneyd, G. Theraulaz, and E. Bonabeau, *Self-Organization in Biological Systems*. Princeton, NJ, USA: Princeton Univ. Press, 2003.

[3] M. E. J. Newman, "The structure and function of complex networks," *SIAM Rev.*, vol. 45, no. 2, pp. 167–256, 2003.

[4] M. E. J. Newman, *Networks: An Introduction*. Oxford, U.K.: Oxford Univ. Press, 2010.

[5] T. G. Lewis, *Network Science: Theory and Applications*. Hoboken, NJ, USA: Wiley, 2009.

[6] C. W. Reynolds, "Flocks, herds, and schools: A distributed behavior model," in *Proc. ACM Annu. Conf. Comput. Graphics Interactive Tech.*, 1987, pp. 25–34.

[7] I. D. Couzin, "Collective cognition in animal groups," *Trends Cogn. Sci.*, vol. 13, pp. 36–43, Jan. 2009.

[8] O. Sporns, *Networks of the Brain*. Cambridge, MA, USA: MIT Press, 2010.

[9] S. Haykin, *Cognitive Dynamic Systems*. Cambridge, U.K.: Cambridge Univ. Press, 2012.

[10] A. H. Sayed, S.-Y. Tu, J. Chen, X. Zhao, and Z. Towfic, "Diffusion strategies for adaptation and learning over networks," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 155–171, May 2013.

[11] A. H. Sayed, "Diffusion adaptation over networks," in *E-Reference Signal Processing*, vol. 3, R. Chellapa and S. Theodoridis, Eds. New York, NY, USA: Academic, May 2012, pp. 323–454, 2014.

[12] P. Di Lorenzo, S. Barbarossa, and A. H. Sayed, "Bio-inspired decentralized radio access based on swarming mechanisms over adaptive networks," *IEEE Trans. Signal Process.*, vol. 61, no. 12, pp. 3183–3197, Jun. 2013.

[13] P. Chainais and C. Richard, "Learning a common dictionary over a sensor network," in *Proc. IEEE 5th Int. Workshop Comput. Adv. Multi-Sensor Adaptive Process.*, Dec. 2013, pp. 133–136.

[14] J. Chen, A. H. Sayed, and Z. Towfic, "Online dictionary learning over distributed models," in *Proc. Int. Conf. Acoust. Speech Signal Process.*, Florence, Italy, May 2014, pp. 3109–3112.

[15] X. Zhao and A. H. Sayed, "Asynchronous adaptation and learning over networks—Part I: Modeling and stability analysis," Dec. 2013. [Online]. Available: arXiv:1312.5434 [cs.SY]

[16] X. Zhao and A. H. Sayed, "Asynchronous adaptation and learning over networks—Part II: Performance analysis," Dec. 2013. [Online]. Available: arXiv:1312.5438 [cs.SY].

[17] S. Haykin, *Adaptive Filter Theory*. Englewood Cliffs, NJ, USA: Prentice-Hall, 2002.

[18] B. Widrow and S. D. Stearns, *Adaptive Signal Processing*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1985.

[19] A. H. Sayed, *Adaptive Filters*. Hoboken, NJ, USA: Wiley, 2008.

[20] A. H. Sayed, *Fundamentals of Adaptive Filtering*. Hoboken, NJ, USA: Wiley, 2003.

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

[22] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer-Verlag, 2007.

[23] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*, 4th ed. New York, NY, USA: Academic, 2008.

[24] D. W. Hosmer and S. Lemeshow, *Applied Logistic Regression*, 2nd ed. Hoboken, NJ, USA: Wiley, 2000.

[25] B. T. Poljak and Y. Z. Tsypkin, "Pseudogradient adaptation and training algorithms," *Autom. Remote Control*, vol. 12, pp. 83–94, 1973.

[26] B. Poljak, *Introduction to Optimization*, Optimization Software, New York, NY, USA, 1987.

[27] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods, 1st ed.* Singapore: Athena Scientific, 1997.

[28] Y. Z. Tsypkin, *Adaptation and Learning in Automatic Systems*. New York, NY, USA: Academic, 1971.

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

[30] D. Bertsekas, *Convex Analysis and Optimization*. Singapore: Athena Scientific, 2003.

[31] Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course*. Norwell, MA, USA: Kluwer, 2004.

[32] 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.

[33] J. Chen and A. H. Sayed, "On the learning behavior of adaptive networks—Part I: Transient analysis," Dec. 2013. [Online]. Available: arXiv:1312.7581 [cs.MA].

[34] H. Robbins and S. Monro, "A stochastic approximation method," *Ann. Math. Stat.*, vol. 22, pp. 400–407, 1951.

[35] R. von Mises and H. Pollaczek-Geiringer, "Praktische verfahren der gleichungs-auflösung," *Z. Agnew. Math. Mech.*, vol. 9, 1929, pp. 152–154.

[36] J. R. Blum, "Multidimensional stochastic approximation methods," *Ann. Math. Stat.*, vol. 25, pp. 737–744, 1954.

[37] L. Schmetterer, "Stochastic approximation," in *Proc. Berkeley Symp. Math. Stat. Probab.*, 1961, pp. 587–609.

[38] G. B. Wetherhill, *Sequential Methods in Statistics*. London, U.K.: Methuen, 1966.

[39] B. Widrow and M. E. Hoff, Jr., "Adaptive switching circuits *IRE WESCON Conv. Rec.*, 1960, pp. 96–104.

[40] 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.

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

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

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

[44] A. Papoulis and S. U. Pilla, *Probability, Random Variables, and Stochastic Processes*. New York, NY, USA: McGraw-Hill, 2002.

[45] R. Durret, *Probability Theory and Examples*, 2nd ed. Independence, KY, USA: Duxbury Press, 1996.

[46] R. M. Dudley, *Real Analysis and Probability*, 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[47] J. Chen and A. H. Sayed, "On the learning behavior of adaptive networks—Part II: Performance analysis," Dec. 2013. [Online]. Available: arXiv:1312.7580 [cs.MA].

[48] W. Rudin, *Principles of Mathematical Analysis*. New York, NY, USA: McGraw-Hill, 1976.

[49] B. Widrow, J. M. McCool, M. G. Larimore, and C. R. Johnson, Jr., "Stationary and nonstationary learning characteristics of the LMS adaptive filter," *Proc. IEEE*, vol. 64, no. 8, pp. 1151–1162, Aug. 1976.

[50] L. Horowitz and K. Senne, "Performance advantage of complex LMS for controlling narrow-band adaptive arrays," *IEEE Trans. Acoust. Speech Signal Process.*, vol. ASSP-29, no. 3, pp. 722–736, Jun. 1981.

[51] S. Jones, R. C. III, and W. Reed, "Analysis of error-gradient adaptive linear estimators for a class of stationary dependent processes," *IEEE Trans. Inf. Theory*, vol. IT-28, no. 2, pp. 318–329, Mar. 1982.

[52] W. A. Gardner, "Learning characterisitcs of stochastic-gradient-descent algorithms: A general study, analysis, and critique," *Signal Process.*, vol. 6, no. 2, pp. 113–133, Apr. 1984.

[53] A. Feuer and E. Weinstein, "Convergence analysis of LMS filters with uncorrelated Gaussian data," *IEEE Trans. Acoust. Speech Signal Process.*, vol. ASSP-33, no. 1, pp. 222–230, Feb. 1985.

[54] J. B. Foley and F. M. Boland, "A note on the convergence analysis of LMS adaptive filters with Gaussian data," *IEEE Trans. Acoust. Speech Signal Process.*, vol. ASSP-36, no. 7, pp. 1087–1089, Jul. 1988.

[55] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer-Verlag, 2000.

[56] Z. Towfic, J. Chen, and A. H. Sayed, "On the generalization ability of distributed online learners," in *Proc. IEEE Workshop Mach. Learn. Signal Process.*, Santander, Spain, Sep. 2012, DOI: 10.1109/MLSP.2012.6349778.

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

[58] 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.

[59] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 2003.

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

[61] F. 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.

[62] O. Dekel, R. Gilad-Bachrach, O. Shamir, and L. Xiao, "Optimal distributed online prediction," in *Proc. Int. Conf. Mach. Learn.*, Bellevue, WA, USA, Jun. 2011, pp. 713–720.

[63] A. Agarwal and J. Duchi, "Distributed delayed stochastic optimization," in *Proc. Neural Inf. Process. Syst.*, Granada, Spain, Dec. 2011, pp. 873–881.

[64] J. B. Predd, S. B. Kulkarni, and H. V. Poor, "Distributed learning in wireless sensor networks," *IEEE Signal Process. Mag.*, vol. 23, no. 4, pp. 56–69, Jul. 2006.

[65] Z. J. Towfic, J. Chen, and A. H. Sayed, "Collaborative learning of mixture models using diffusion adaptation," in *Proc. IEEE Workshop Mach. Learn. Signal Process.*, Beijing, China, Sep. 2011, DOI: 10.1109/MLSP.2011.6064578.

[66] C.-K. Yu, M. van der Schaar, and A. H. Sayed, "Reputation design for adaptive networks with selfish agents," in *Proc. IEEE Workshop Signal Process. Adv. Wireless Commun.*, Darmstadt, Germany, Jun. 2013, pp. 160–164.

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

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

[69] D. P. Bertsekas, *Nonlinear Programming*, 2nd ed. Belmont, MA, USA: Athena Scientific, 1999.

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

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

[72] E. S. Helou and A. R. De Pierro, "Incremental subgradients for constrained convex optimization: A unified framework and new methods," *SIAM J. Optim.*, vol. 20, pp. 1547–1572, 2009.

[73] B. Johansson, M. Rabi, and M. Johansson, "A randomized incremental subgradient method for distributed optimization in networked systems," *SIAM J. Optim.*, vol. 20, pp. 1157–1170, 2009.

[74] D. Blatt, A. O. Hero, and H. Gauchman, "A convergent incremental gradient method with a constant step size," *SIAM J. Optim.*, vol. 18, pp. 29–51, 2008.

[75] A. H. Sayed and C. Lopes, "Distributed recursive least-squares strategies over adaptive networks," in *Proc. 40th Asilomar Conf. Signals Syst. Comput.*, Pacific Grove, CA, USA, Oct.–Nov. 2006, pp. 233–237.

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

[77] A. H. Sayed and F. Cattivelli, "Distributed adaptive learning mechanisms," in *Handbook on Array Processing and Sensor Networks*, S. Haykin and K. J. Ray Liu, Eds. Hoboken, NJ, USA: Wiley, 2009, pp. 695–722.

[78] L. Li, C. G. Lopes, J. Chambers, and A. H. Sayed, "Distributed estimation over an adaptive incremental network based on the affine projection algorithm," *IEEE Trans. Signal Process.*, vol. 58, no. 1, pp. 151–164, Jan. 2010.

[79] F. Cattivelli and A. H. Sayed, "Analysis of spatial and incremental LMS processing for distributed estimation," *IEEE Trans. Signal Process.*, vol. 59, no. 4, pp. 1465–1480, Apr. 2011.

[80] J. B. Predd, S. R. Kulkarni, and H. V. Poor, "A collaborative training algorithm for distributed learning," *IEEE Trans. Inf. Theory*, vol. 55, no. 4, pp. 1856–1871, Apr. 2009.

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

[82] P. Alriksson and A. Rantzer, "Distributed Kalman filtering using weighted averaging," in *Proc. 17th Int. Symp. Math. Theory Netw. Syst.*, Kyoto, Japan, 2006, pp. 1–6.

[83] J. Tsitsiklis and M. Athans, "Convergence and asymptotic agreement in distributed

decision problems," *IEEE Trans. Autom. Control*, vol. AC-29, no. 1, pp. 42–50, Jan. 1984.

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

[85] A. Nedic and A. Ozdaglar, "Cooperative distributed multi-agent optimization," in *Convex Optimization in Signal Processing and Communications*, Y. Eldar and D. Palomar, Eds. Cambridge, U.K.: Cambridge Univ. Press, 2010, pp. 340–386.

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

[87] B. Johansson, T. Keviczky, M. Johansson, and K. Johansson, "Subgradient methods and consensus algorithms for solving convex optimization problems," in *Proc. IEEE Conf. Decision Control*, Cancun, Mexico, Dec. 2008, pp. 4185–4190.

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

[89] 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, Jun. 2012.

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

[91] L. Xiao, S. Boyd, and S. Lall, "A space-time diffusion scheme peer-to-peer least-squares-estimation," in *Proc. Inf. Process. Sensor Netw.*, Nashville, TN, USA, Apr. 2006, pp. 168–176.

[92] W. Ren and R. W. Beard, "Consensus seeking in multi-agent systems under dynamically changing interaction topologies," *IEEE Trans. Autom. Control*, vol. 50, no. 5, pp. 655–661, May 2005.

[93] R. Olfati-Saber and J. Shamma, "Consensus filters for sensor networks and distributed sensor fusion," in *Proc. 44th IEEE Conf. Decision Control*, Seville, Spain, Dec. 2005, pp. 6698–6703.

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

[95] S. Sardellitti, M. Giona, and S. Barbarossa, "Fast distributed average consensus algorithms based on advection-diffusion processes," *IEEE Trans. Signal Process.*, vol. 58, no. 2, pp. 826–842, Feb. 2010.

[96] P. Braca, S. Marano, and V. Matta, "Running consensus in wireless sensor networks," in *Proc. 11th Int. Conf. Inf. Fusion*, Cologne, Germany, Jun. 2008, pp. 1–6.

[97] 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.

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

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

[100] R. M. Karp, "Reducibility among combinational problems," in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, Eds. New York, NY, USA: Plenum, 1972, pp. 85–104.

[101] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed. Baltimore, MD, USA: The John Hopkins Univ. Press, 1996.

[102] A. Berman and R. J. Plemmons, *Nonnegative Matrices in the Mathematical Sciences*. Philadelphia, PA, USA: SIAM, 1994.

[103] S. U. Pillai, T. Suel, and S. Cha, "The Perron-Frobenius theorem: Some of its applications," *IEEE Signal Process. Mag.*, vol. 22, no. 2, pp. 62–75, Mar. 2005.

[104] M. H. DeGroot, "Reaching a consensus," *J. Amer. Stat. Assoc.*, vol. 69, no. 345, pp. 118–121, 1974.

[105] R. L. Berger, "A necessary and sufficient condition for reaching a consensus using DeGroot's method," *J. Amer. Stat. Assoc.*, vol. 76, no. 374, pp. 415–418, Jun. 1981.

[106] R. Olfati-Saber, "Distributed Kalman filter with embedded consensus filters," in *Proc. Joint IEEE Conf. Decision Control/Eur. Control Conf.*, Seville, Spain, Dec. 2005, pp. 8179–8184.

[107] A. Das and M. Mesbahi, "Distributed linear parameter estimation in sensor networks based on Laplacian dynamics consensus algorithm," in *Proc. 3rd Annu. IEEE Commun. Soc. Conf. Sensor Ad Hoc Commun. Netw.*, Reston, VA, USA, Sep. 2006, vol. 2, pp. 440–449.

[108] R. Carli, A. Chiuso, L. Schenato, and S. Zampieri, "Distributed Kalman filtering using consensus strategies," *IEEE J. Sel. Areas Commun.*, vol. 26, no. 4, pp. 622–633, Sep. 2008.

[109] U. A. Khan and J. M. F. Moura, "Distributing the Kalman filter for large-scale systems," *IEEE Trans. Signal Process.*, vol. 56, no. 10, pp. 4919–4935, Oct. 2008.

[110] C. G. Lopes and A. H. Sayed, "Distributed processing over adaptive networks," in *Proc. Adaptive Sensor Array Process. Workshop*, MIT Lincoln Laboratory, Cambridge, MA, USA, Jun. 2006, pp. 1–5.

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

[112] C. G. Lopes and A. H. Sayed, "Diffusion least-mean-squares over adaptive networks," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, Honolulu, HI, USA, Apr. 2007, vol. 3, pp. 917–920.

[113] C. G. Lopes and A. H. Sayed, "Steady-state performance of adaptive diffusion least-mean squares," in *Proc. IEEE Workshop Stat. Signal Process.*, Madison, WI, USA, Aug. 2007, pp. 136–140.

[114] F. S. Cattivelli, C. G. Lopes, and A. H. Sayed, "A diffusion RLS scheme for distributed estimation over adaptive networks," in *Proc. IEEE Workshop Signal Process. Adv. Wireless Commun.*, Helsinki, Finland, Jun. 2007, DOI: 10.1109/SPAWC.2007.4401393.

[115] F. S. Cattivelli and A. H. Sayed, "Diffusion LMS algorithms with information exchange," in *Proc. Asilomar Conf. Signals Syst. Comput.*, Pacific Grove, CA, USA, Nov. 2008, pp. 251–255.

[116] F. S. Cattivelli and A. H. Sayed, "Diffusion mechanisms for fixed-point distributed Kalman smoothing," in *Proc. EUSIPCO*, Lausanne, Switzerland, Aug. 2008, pp. 1–4.

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

[118] 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.

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

[120] S. Lee and A. Nedic, "Distributed random projection algorithm for convex optimization," *IEEE J. Sel. Top. Signal Process.*, vol. 7, no. 2, pp. 221–229, Apr. 2013.

[121] P. Bianchi, G. Fort, and W. Hachem, "Performance of a distributed stochastic approximation algorithm," *IEEE Trans. Inf. Theory*, vol. 59, no. 11, pp. 7405–7418, Nov. 2013.

[122] S. S. Stankovic, M. S. Stankovic, and D. S. Stipanovic, "Decentralized parameter estimation by consensus based stochastic approximation," *IEEE Trans. Autom. Control*, vol. 56, no. 3, pp. 531–543, Mar. 2011.

[123] A. Speranzon, C. Fischione, and K. H. Johansson, "Distributed and collaborative estimation over wireless sensor networks," in *Proc. IEEE Conf. Decision Control*, San Diego, CA, USA, Dec. 2006, pp. 1025–1030.

[124] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2010.

[125] I. D. Schizas, G. Mateos, and G. B. Giannakis, "Distributed LMS for consensus-based in-network adaptive processing," *IEEE Trans. Signal Process.*, vol. 57, no. 6, pp. 2365–2382, Jun. 2009.

[126] G. Mateos, Gonzalo, I. D. Schizas, and G. B. Giannakis, "Distributed recursive least-squares for consensus-based in-network adaptive estimation," *IEEE Trans. Signal Process.*, vol. 57, no. 11, pp. 4583–4599, Nov. 2009.

[127] J. Chen and A. H. Sayed, "On the limiting behavior of distributed optimization strategies," in *Proc. 50th Annu. Allerton Conf. Commun. Control Comput.*, Monticello, IL, USA, Oct. 2012, pp. 1535–1542.

[128] L. A. Zadeh, "Optimality and non-scalar-valued performance criteria," *IEEE Trans. Autom. Control*, vol. AC-8, no. 1, pp. 59–60, Jan. 1963.

[129] M. D. Intriligator, *Mathematical Optimization and Economic Theory*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1971.

[130] A. H. Sayed, S.-Y. Tu, and J. Chen, "Online learning and adaptation over networks: More information is not necessarily better," in *Proc. Inf. Theory Appl. Workshop (ITA)*, San Diego, CA, USA, Feb. 2013, DOI: 10.1109/ITA.2013.6502975.

[131] V. D. Blondel, J. M. Hendrickx, A. Olshevsky, and J. N. Tsitsiklis, "Convergence in multiagent coordination, consensus, and flocking," in *Proc. Joint IEEE Conf. Decision Control/Eur. Control Conf.*, Seville, Spain, Dec. 2005, pp. 2996–3000.

[132] Pascal Large Scale Learning Challenge. [Online]. Available: http://largescale.ml.tu-berlin.de/instructions/

[133] S. Meyn and R. L. Tweedie, *Markov Chains and Stochastic Stability,* 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, 2009.

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

[135] 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.

[136] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equations of state calculations by fast computing machines," *J. Chem. Phys.*, vol. 21, no. 6, pp. 1087–1092, 1953.

[137] X. Zhao and A. H. Sayed, "Asynchronous adaptation and learning over networks—Part III: Comparison analysis," Dec. 2013. [Online]. Available: arXiv:1312.5439 [cs.SY].

[138] X. Zhao and A. H. Sayed, "Attaining optimal batch performance via distributed processing over networks," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, Vancouver, BC, Canada, May 2013, pp. 5214–5218.

[139] N. Takahashi, I. Yamada, and A. H. Sayed, "Diffusion least-mean-squares with adaptive combiners: Formulation and performance analysis," *IEEE Trans. Signal Process.*, vol. 58, no. 9, pp. 4795–4810, Sep. 2010.

[140] S.-Y. Tu and A. H. Sayed, "Adaptive networks with noisy links," in *Proc. IEEE Global Telecommun. Conf.*, Houston, TX, USA, Dec. 2011, DOI: 10.1109/GLOCOM.2011.6134038.

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

[142] 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.

[143] R. Abdolee and B. Champagne, "Diffusion LMS algorithms for sensor networks over non-ideal inter-sensor wireless channels," in *Proc. IEEE Int. Conf. Distrib. Comput. Sensor Syst.*, Barcelona, Spain, Jun. 2011, DOI: 10.1109/DCOSS.2011.5982212.

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

[145] X. Zhao and A. H. Sayed, "Combination weights for diffusion strategies with imperfect information exchange," in *Proc. IEEE Int. Conf. Commun.*, Ottawa, ON, Canada, Jun. 2012, pp. 648–652.

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

[147] G. Mateos, I. D. Schizas, and G. B. Giannakis, "Performance analysis of the consensus-based distributed LMS algorithm," *EURASIP J. Adv. Signal Process.*, pp. 1–19, 2009, DOI: 10.1155/2009/981030.

[148] T. C. Aysal, M. E. Yildiz, A. D. Sarwate, and A. Scaglione, "Broadcast gossip algorithms for consensus," *IEEE Trans. Signal Process.*, vol. 57, no. 7, pp. 2748–2761, Jul. 2009.

[149] D. Shah, "Gossip algorithms," *Found. Trends Netw.*, vol. 3, pp. 1–125, 2009.

[150] C. Lopes and A. H. Sayed, "Diffusion adaptive networks with changing topologies," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, Las Vegas, NV, USA, Apr. 2008, pp. 3285–3288.

[151] O. L. Rortveit, J. H. Husoy, and A. H. Sayed, "Diffusion LMS with communications constraints," in *Proc. 44th Asilomar Conf. Signals Syst. Comput.*, Pacific Grove, CA, USA, Nov. 2010, pp. 1645–1649.

[152] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2508–2530, Jun. 2006.

[153] S. Kar and J. M. F. Moura, "Distributed consensus algorithms in sensor networks: Quantized data and random link failures," *IEEE Trans. Signal Process.*, vol. 58, no. 3, pp. 1383–1400, Mar. 2010.

[154] S. Kar and J. M. F. Moura, "Sensor networks with random links: Topology design for distributed consensus," *IEEE Trans. Signal Process.*, vol. 56, no. 7, pp. 3315–3326, Jul. 2008.

[155] D. Jakovetic, J. Xavier, and J. M. F. Moura, "Cooperative convex optimization in networked systems: Augmented Lagranian algorithms with directed Gossip communication," *IEEE Trans. Signal Process.*, vol. 59, no. 8, pp. 3889–3902, Aug. 2011.

[156] N. Takahashi and I. Yamada, "Link probability control for probabilistic diffusion least-mean squares over resource-constrained networks," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, Dallas, TX, USA, Mar. 2010, pp. 3518–3521.

[157] P. Di Lorenzo and A. H. Sayed, "Sparse distributed learning based on diffusion adaptation," *IEEE Trans. Signal Process.*, vol. 61, no. 6, pp. 1419–1433, Mar. 2013.

[158] S. Chouvardas, K. Slavakis, Y. Kopsinis, and S. Theodoridis, "A sparsity-promoting adaptive algorithm for distributed learning," *IEEE Trans. Signal Process.*, vol. 60, no. 10, pp. 5412–5425, Oct. 2012.

[159] S. Chouvardas, G. Mileounis, N. Kalouptsidis, and S. Theodoridis, "A greedy sparsity-promoting LMS for distributed adaptive learning in diffusion networks," in *Proc. Int. Conf. Acoust. Speech Signal Process.*, VancouverBCCanada, 2013, pp. 5415–5419.

[160] Y. Liu, C. Li, and Z. Zhang, "Diffusion sparse least-mean squares over networks," *IEEE Trans. Signal Process.*, vol. 60, no. 8, pp. 4480–4485, Aug. 2012.

[161] R. Tibshirani, "Regression shrinkage and selection via the LASSO," *J. Roy. Stat. Soc. B*, vol. 58, pp. 267–288, 1996.

[162] E. J. Candes, M. Wakin, and S. Boyd, "Enhancing sparsity by reweighted ℓ_1 minimization," *J. Fourier Anal. Appl.*, vol. 14, pp. 877–905, 2007.

[163] R. Baraniuk, "Compressive sensing," *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 21–30, Mar. 2007.

[164] Y. Chen, Y. Gu, and A. O. Hero, "Sparse LMS for system identification," in *Proc. Int. Conf. Acoust. Speech Signal Process.*, Taipei, Taiwan, May 2009, pp. 3125–3128.

[165] Y. Kopsinis, K. Slavakis, and S. Theodoridis, "Online sparse system identification and signal reconstruction using projections onto weighted balls," *IEEE Trans. Signal Process.*, vol. 59, no. 3, pp. 936–952, Mar. 2010.

[166] F. Yan, S. Sundaram, S. V. N. Vishwanathan, and Y. Qi, "Distributed autonomous online learning: Regrets and intrinsic privacy-preserving properties," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 11, pp. 2483–2493, Nov. 2013.

[167] S. Theodoridis, K. Slavakis, and I. Yamada, "Adaptive learning in a world of projections: A unifying framework for linear and nonlinear classification and regression tasks," *IEEE Signal Process. Mag.*, vol. 28, no. 1, pp. 97–123, Jan. 2011.

[168] 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.

[169] R. Cavalcante, I. Yamada, and B. Mulgrew, "An adaptive projected subgradient approach to learning in diffusion networks," *IEEE Trans. Signal Process.*, vol. 57, no. 7, pp. 2762–2774, Jul. 2009.

[170] Z. Towfic and A. H. Sayed, "Adaptive stochastic convex optimization over networks," in *Proc. 51st Annu. Allerton Conf. Commun. Control Comput.*, Monticello, IL, USA, Oct. 2013, pp. 1272–1277.

[171] Z. Towfic and A. H. Sayed, "Adaptive penalty-based distributed stochastic convex optimization," Dec. 2013. [Online]. Available: arXiv:1312.4415 [math.OC].

[172] K. Slavakis, Y. Kopsinis, and S. Theodoridis, "Adaptive algorithm for sparse system identification using projections onto weighted ℓ_1 balls," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, Dallas, TX, USA, Mar. 2010, pp. 3742–3745.

[173] A. Bertrand, M. Moonen, and A. H. Sayed, "Diffusion bias-compensated RLS estimation over adaptive networks," *IEEE Trans. Signal Process.*, vol. 59, no. 11, pp. 5212–5224, Nov. 2011.

[174] R. Olfati-Saber, "Kalman-consensus filter: Optimality, stability, and performance," in *Proc. IEEE Conf. Decision Control*, Shangai, China, 2009, pp. 7036–7042.

[175] R. Olfati-Saber, "Distributed Kalman filtering for sensor networks," in *Proc. 46th IEEE Conf. Decision Control*, New Orleans, LA, USA, Dec. 2007, pp. 5492–5498.

[176] F. S. Cattivelli, C. G. Lopes, and A. H. Sayed, "Diffusion strategies for distributed Kalman filtering: Formulation and performance analysis," in *Proc. IAPR Workshop Cogn. Inf. Process.*, Santorini, Greece, Jun. 2008, pp. 36–41.

[177] F. Cattivelli and A. H. Sayed, "Diffusion strategies for distributed Kalman filtering and smoothing," *IEEE Trans. Autom. Control*, vol. 55, no. 9, pp. 2069–2084, Sep. 2010.

[178] F. Cattivelli and A. H. Sayed, "Diffusion distributed Kalman filtering with adaptive weights," in *Proc. Asilomar Conf. Signals Syst. Comput.*, Pacific Grove, CA, USA, Nov. 2009, pp. 908–912.

[179] O. Hlinka, O. Sluciak, F. Hlawatsch, and P. M. Djuric, "Likelihood consensus and its application to distributed particle filtering," *IEEE Trans. Signal Process.*, vol. 60, no. 8, pp. 4334–4349, Aug. 2012.

ABOUT THE AUTHOR

Ali H. Sayed (Fellow, IEEE) received his Ph.D. degree in electrical engineering from Stanford University, in 1992.

He is a Professor and former chairman of electrical engineering at the University of California Los Angeles (UCLA), Los Angeles, CA, USA, where he directs the UCLA Adaptive Systems Laboratory. His research activities involve several areas including adaptation and learning, network science, information processing theories, distributed processing, statistical data analysis, and biologically inspired designs.

Prof. Sayed is also a Fellow of the American Association for the Advancement of Science (AAAS). His work has been recognized with several awards, including the 2014 Athanasios Papoulis Award from the European Association for Signal Processing, the 2013 Meritorious Service Award and the 2012 Technical Achievement Award from the IEEE Signal Processing Society, the 2005 Terman Award from the American Society for Engineering Education, the 2003 Kuwait Prize, and the 1996 Donald G. Fink Prize from IEEE. He served as a Distinguished Lecturer of the IEEE Signal Processing Society in 2005 and delivered recent plenary lectures at the 2013 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP) and the 2013 European Signal Processing Conference (EUSIPCO). His articles received best paper awards from the IEEE Signal Processing Society in 2002, 2005, and 2012. He has been active in serving the signal processing community in various roles. Among other activities, he served as Editor-in-Chief of the IEEE Transactions on Signal Processing (2003–2005), Editor-in-Chief of the *EURASIP Journal on Advances in Signal Processing* (2006–2007), General Chairman of ICASSP (Las, Vegas, NV, USA, 2008), and Vice-President of Publications of the IEEE Signal Processing Society (2009–2011). He also served as member of the Board of Governors (2007–2011), Awards Board (2005), Publications Board (2003–2005), Conference Board (2007–2011), and Technical Directions Board (2008–2009) of the same Society.