Stefan Vlaski, Soummya Kar, Ali H. Sayed, and José M.F. Moura

Networked Signal and Information Processing

Learning by multiagent systems

©SHUTTERSTOCK.COM/TRIFF

Digital Object Identifier 10.1109/MSP.2023.3267896
Date of current version: 14 July 2023

This article reviews significant advances in networked signal and information processing (SIP), which have enabled in the last 25 years extending decision making and inference, optimization, control, and learning to the increasingly ubiquitous environments of distributed agents. As these interacting agents cooperate, new collective behaviors emerge from local decisions and actions. Moreover, and significantly, theory and applications show that networked agents, through cooperation and sharing, are able to match the performance of cloud or federated solutions while offering the potential for improved privacy, increased resilience, and conserved resources. A longer version of this manuscript, with examples and illustrative applications, is available at https://arxiv.org/abs/2210.13767.

Introduction

Since its beginnings, throughout the past century and still dominant at the turn of the 21st century, the SIP prevailing paradigm has been to process signals and information by stand-alone systems or central computing units with no cooperation or interaction among them [see Figure 1(a)]. This focus has led to tremendous progress in a wide range of problems in speech and image processing, control and guidance, estimation and filtering theories, communications theory, and array processing, with enormous impacts in telecommunication and wireless, audio, medical imaging, multimedia, radar, and other application areas. In the nearly 25 years since the turn of the century, each of these areas has progressed rapidly, in large part due to increases in computational resources along with the availability of data, giving rise to a variety of advanced data-driven processing tools.

At the end of the century, we also witnessed significant technological progress, from massive layouts of fiber at the backbone; to successes in high-speed wireless and Wi-Fi deployments; to chip advances combining single, miniature inexpensive platform functionalities like sensing, storage, communication, and computing; and to breakthroughs in network protocols and software. This progress has led, for example, to the launching of hundreds, soon thousands and millions,

of inexpensive sensing devices (which we call here *agents*) that sense, compute, communicate, and are networked, ushering a paradigm shift in SIP. Initially, the agents observed data independently of one another and simply forwarded their raw data to the cloud, with no local processing in a centralized architecture (see Figure 1).

Parallel architectures soon emerged, where agents started processing their local data, transferring only their (local) inferences to a fusion center. The fusion center aggregated the locally processed data and orchestrated the computations that occurred in parallel at the individual agents. While traditionally computation and communication occurred in a synchronous fashion, synchrony requirements were relaxed, like with federated learning architectures [see Figure 1(c)]. But as a result of scenarios with abundant data available at dispersed networked locations, such as sensor networks that monitor large geographical regions, robotic swarms that collaborate over a common task, or social networks of many interacting agents, a new critical trend started to materialize. This led to decentralization and democratization of technology and, toward the middle and end of the first decade of this century, SIP moved definitively from parallel, federated, or edge architectures to a distributed, decentralized, or networked paradigm. (Note that we interpret an edge architecture as a layered or hierarchical federated architecture.) The agents sense and process their own data, and then cooperate with other agents. They transmit to and exchange information with agents in their vicinity. It marked the appearance of networked elementary processing units, with each unit collecting and processing data and sharing their information with immediate neighbors.

Distributed, decentralized, or networked architectures achieve aggregation and coordination through device-to-device or peer-to-peer interactions.

Individual agents are now capable of local inference decisions and limited actions. The coupling among the agents gives rise to powerful network structures that open up vast opportunities for the solution of more complex problems by tapping into the power of the group. Examples of such networked systems are plentiful, including instrumented critical infrastructures like water, gas, financial networks, and smart grids as well as networked mobile devices, swarms of drones, autonomous vehicles, or populations of individuals. The interconnectedness of the agents within the network allows for their cooperation to rise from local to global coherent decision and action. To study, understand, and steer the occurrence of these global behaviors adds new layers of complexity. More advanced analytical tools became necessary to combine local processing with cooperation among the agents. This ushered in the design of new processing algorithms, new methods to derive performance guarantees and assess their quality, to examine the effect of agents coupling on network stability, to endow agents with adaptation and learning abilities and with the capacity to handle privacy, and to enable such networks to contribute to multiple tasks at once.

Distributed, decentralized, or networked architectures achieve aggregation and coordination through device-to-device or peer-to-peer interactions. Computation is no longer at the cloud or like in federated or edge computing at a fusion center, but fully distributed at the device level. Synchrony requirements need not be assumed. Networked architectures may be viewed as a generalization of centralized and federated configurations, allowing us to recover federated algorithms from distributed or decentralized ones by employing a star topology.

Networked distributed processing architectures are more robust: if an edge or an agent fails, the entire network can continue to process data and deliver inference decisions. There is no need for costly communications with the cloud or a remote edge server. Furthermore, although the exchange of processed iterates might still leak some private information, recent works have demonstrated that networked architectures can be designed to offer improved privacy due to their decentralized nature. Even more importantly, distributed networked architectures can be shown to match the performance of centralized solutions.

This tutorial article surveys the recent history of networked SIP, including consensus and diffusion strategies for regression problems [1], [2], [3], [4], [5] developed in the 2000s, detection and parameter estimation over networks [6], [7], [8], [9] and their performance guarantees [8], [9], [10], [11], distributed optimization [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], learning, and adaptation [20], [21], [22]. It provides a

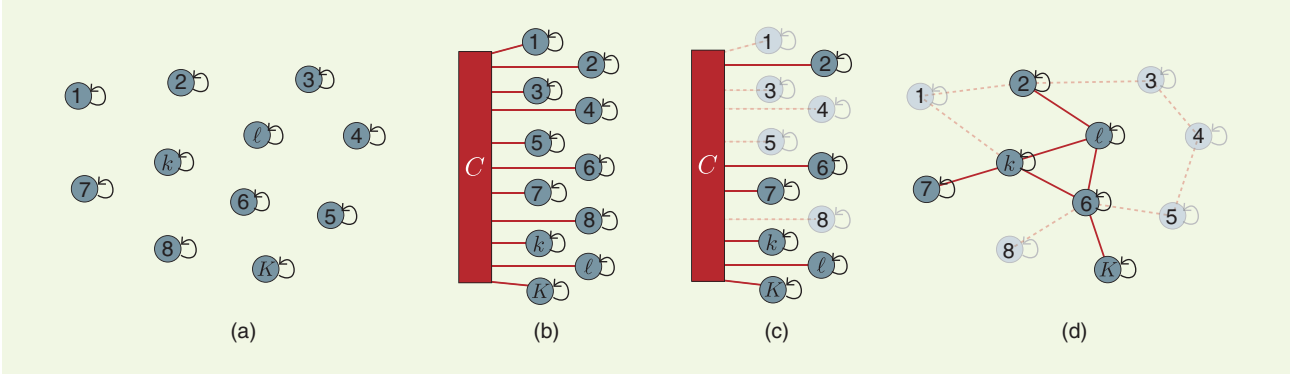


FIGURE 1. The taxonomy of networked multiagent systems: (a) noncooperative, (b) centralized or parallel, (c) federated, and (d) networked or decentralized.

comprehensive coverage of topics and references. We bridge the gap by unifying under a common umbrella more recent applications to distributed machine learning, including multitask learning [23] and nonconvex optimization [24], [25], design variants under different operating scenarios such as asynchronous algorithms [26], and connections to alternative architectures such as federated learning.

Historical remarks

There has been extensive work on distributed techniques for information and signal processing over networks. Many optimal inference problems adopt a quadratic optimization cost whose solution, under linear models and Gaussian noise, is a linear statistic of the data. With peer-to-peer communication among sensors, the question becomes how to compute the global average of the local statistics only through cooperation among the agents. Departing from centralized architectures, the solution is built on the consensus strategy for distributed averaging, with no need for a fusion center to collect the dispersed data for processing. Consensus was initially proposed by DeGroot [27] to enable a group of agents to reach agreement by pooling their information and to converge [27], [28] to an average estimate solely by interaction among neighbors. Many subsequent works were devoted to characterizing consensus' convergence behavior, the role of the graph topology, random selection of neighbors, and several other aspects. Some sample works include [5], [29], and [30], while useful overviews appear in [7] and [22] with many additional references.

Several works in the literature proposed extensions of the original consensus construction to more generally minimize aggregate cost functions, such as mean-square-error costs, or to solve distributed estimation problems of the least-squares or Kalman filtering type. These extensions involve constructions with gradient descent-type updates. Among these works, we mention [29] and [31]. Although an early version of the consensus gradient-based algorithm for distributed estimation and optimization already appears in [29], convergence results were limited to the asymptotic regime, and there was no understanding of the performance of the algorithm, its actual convergence rate, and the influence of data, topology, quantization, noise, and asynchrony on behavior. These considerations are of great significance when designing practical, data-driven systems, and they attracted the attention of the signal processing community after the turn of the century. Moreover, some of the earlier investigations on consensus implementations involved separate timescales (fast communication and consensus iterations among agents, and slow data collection), which can be a challenge for streaming or online data.

Online consensus implementations, where data are collected at every consensus step, appeared in works by the authors of [5], [8], [12], [32], [33], and others. Using decaying step-sizes, these works established the ability of the algorithms to converge. In particular, the work of Kar et al. [8] introduced the so-called consensus + innovations variant, which responds to streaming data, and established several performance measures in terms of convergence rate and the effect of topology, quantization, and noisy conditions and other factors (see also [33]). In parallel with these developments, online distributed algorithms of the diffusion type were introduced in [3], [34], and [35] to enable continuous adaptation and learning by networked agents under constant step-size conditions. The diffusion strategies modified the consensus update to incorporate symmetry, which was shown to enlarge the stability range of the network, and to enhance its performance, even under decaying step-sizes (see [36] and the overviews in [20] and [22]). The diffusion structure was used in several subsequent works for distributed optimization, such as in [14], [37], [38], and other works.

In all of these and the related works on online distributed inference, the goal is for every agent in the network to converge to an estimate of the unknown by relying exclusively on local interactions with its neighbors. Important questions that arise in this context include the following:

- *Convergence*: Do the distributed inference algorithms converge, and if so, in what sense?
- *Agreement*: Do the agents reach a consensus on their inferences?
- *Distributed versus centralized*: How good is the distributed inference solution at each agent when compared with the centralized inference obtained by a fusion center; in other words, are the distributed inference sequences consistent and asymptotically unbiased, efficient, and normal?
- *Rate of convergence*: What is the rate at which the distributed inference at each agent converges?

These questions require very different approaches than the methods used in the "consensus or averaging-only" solution from earlier works. Solutions that emerged of the consensus and diffusion type combine at each iteration 1) an aggregation step that fuses the current inference statistic at each agent with the states of their neighbors with 2) a local update driven by the new observation at the agent. This generic framework, of which there are several variations, is very different from the standard consensus wherein each time step only local averaging of the neighbors' states occurs, and no observations are processed, and from other distributed inference algorithms with multiple timescales, where between-measurement updates involve a large number of consensus steps (theoretically, an infinite number of steps). The classes of successful distributed inference algorithms that emerged add only to the identifiability condition of the centralized model that the network be connected on average. The results of these algorithms are also shown to hold under broad conditions like agents' communication channel intermittent failures, asynchronous and random communication protocols, and quantized communication (limited bandwidth), making their application realistic when 1) a large number of agents are involved (bound to fail at random times), 2) packet losses in wireless digital communications cause links to fail intermittently, 3) agents communicate asynchronously, and 4) the agents may be resource constrained and have a limited bit budget for communication. Furthermore, these distributed inference algorithms make no distributional assumptions on the agents and link failures that can be spatially

correlated. The reader may refer to the overviews in [7], [20], [22], and the many references therein.

There have of course been many other useful contributions to the theory and practice of networked information and signal processing, with many other variations and extensions. However, space limitations prevent us from elaborating further. The reader can refer to the overviews in [7] and [22]. Among such extensions, we may mention extensive works on distributed Kalman filtering in [39], [40], [41], and others. Other parts of this manuscript refer to other classes of distributed algorithms, such as constructions of the primal and primal-dual type, and the corresponding references. The presentation actually presents a unified view of a large family of distributed implementations, including consensus and diffusion, for online inference by networked agents.

Notation

All vectors are column vectors. We employ bold font to emphasize random variables, and regular font for their realizations as well as deterministic quantities. Upper-case letters denote matrices, while Greek letters denote scalar variables. We employ k to index nodes or agents in the network, and i to index time. In this way, $\boldsymbol{x}_{k,i}$ will denote the data available to agent k at time i, modeled as a random variable. When discussing supervised learning problems, $\boldsymbol{x}_{k,i} \triangleq \mathrm{col}\{\boldsymbol{h}_{k,i}, \boldsymbol{\gamma}_{k,i}\}$ will contain both the feature vector $\boldsymbol{h}_{k,i}$ and the associated label $\boldsymbol{\gamma}_{k,i}$.

Unified view

In the networked SIP context, K agents are nodes of a connected network, whose graph is described by a weighted adjacency matrix $C \in \mathbb{R}^{K\times K}$, where $c_{\ell k} \triangleq [C]_{\ell k}$ denotes the strength of the link from node ℓ to node k. We denote by $\mathcal{N}_k$ the neighbors of node k, i.e., those other agents with which k communicates directly and cooperates. With undirected graphs, the graph is also described by its (weighted) Laplacian matrix, $L = \mathrm{diag}\{C\mathbb{1}\} - C$. Here, "$\mathbb{1}$" denotes the vector consisting of all 1s of appropriate size. We illustrate an example of a graph and its adjacency matrix in Figure 2. We further associate with each node a local model $w_k \in \mathbb{R}^M$, which can correspond to unknown parameters describing a random field, parameterizing a channel or filter, or representing a hyperplane or a neural network. For convenience, we define *network-level quantities*, which we denote through calligraphic letters; they aggregate quantities from across the network. In this manner, we can write compactly $\boldsymbol{w} \triangleq \mathrm{col}\{w_k\}$. This notation allows us to highlight a useful relation between the adjacency matrix C, and the Laplacian matrix L, namely, that for undirected graphs

$$\sum_{k=1}^{K}\sum_{\ell=1}^{K} c_{\ell k}\left\| w_k - w_\ell \right\|^2 = \boldsymbol{w}^{\mathsf{T}}\mathcal{L}\boldsymbol{w} \tag{1}$$

where we defined $\mathcal{L} \triangleq L \otimes I_M$. Relation (1) captures through the variational operator $\mathcal{L}$ the weighted variation of the local models w_k across the network, and is fundamental when deriving algorithms for distributed processing over networks, as we illustrate further in the next section.

Unification through stochastic optimization

Suppose we would like each agent in the network to estimate the unknown parameter w_k^o used to generate local observations through the linear model

$$\boldsymbol{\gamma}_{k,i} = \boldsymbol{h}_{k,i}^{\mathsf{T}} w_k^o + \boldsymbol{v}_{k,i}. \tag{2}$$

In a parameter estimation framework, $\boldsymbol{h}_{k,i}$ denotes the local known observation model, $\boldsymbol{v}_{k,i}$ denotes noise, and $\boldsymbol{\gamma}_{k,i}$ are the observations. In a machine learning interpretation, during training, we learn the weights or model w_k^o in (2) from known pairs of input data $\boldsymbol{h}_{k,i}$ and target values $\boldsymbol{\gamma}_{k,i}$. Common other terminology refers to $\boldsymbol{h}_{k,i}$ as a regressor, feature vector, independent variables, or inputs. We may then formulate local estimation or learning problems based on the mean square error (MSE) risk, $J_k(w_k) \triangleq \mathbb{E}\left\| \boldsymbol{\gamma}_{k,i} - \boldsymbol{h}_{k,i}^{\mathsf{T}} w_k \right\|^2$, and pursue

$$w_k^o \triangleq \arg\min_{w_k\in\mathbb{R}^M} J_k(w_k) \quad \text{or equivalently} \quad \min_{\boldsymbol{w}} \sum_{k=1}^{K} J_k(w_k). \tag{3}$$

If, however, we are provided with prior information that the parameters w_k^o vary smoothly as defined by the variational relation (1) over a graph with Laplacian L, we may instead pursue

$$\min_{\boldsymbol{w}} \left\{ \sum_{k=1}^{K} J_k(w_k) + \frac{\eta}{2}\boldsymbol{w}^{\mathsf{T}}\mathcal{L}\boldsymbol{w} \right\}. \tag{4}$$

The regularization term $(\eta/2)\boldsymbol{w}^{\mathsf{T}}\mathcal{L}\boldsymbol{w}$ couples the independent objectives $J_k(w_k)$ and encourages parameterizations w_k that vary smoothly over the graph. It can be verified that the coupled optimization problem (4) corresponds to a maximum a posteriori estimate of the models w_k^o in the linear model

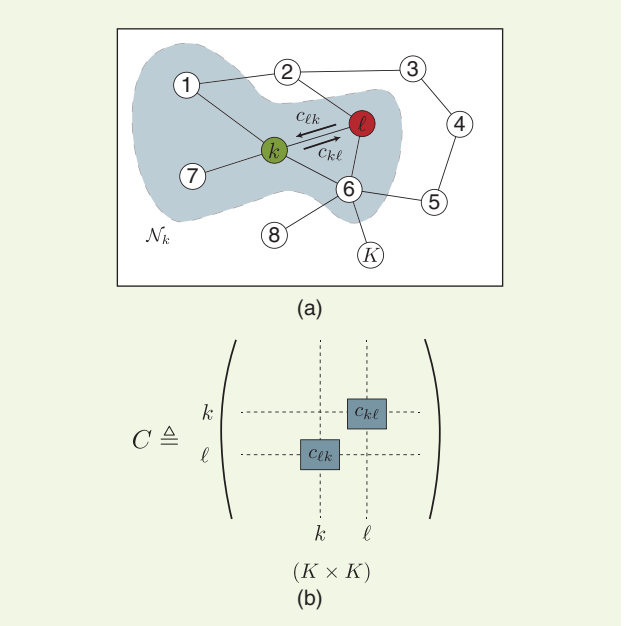


FIGURE 2. The schematic of a general network and its adjacency matrix. (a) Topology of a sample network. (b) Corresponding adjacency matrix.

(2) under a Gaussian–Markov random-field prior. Motivated by applications in wireless sensor networks, least-squares problems of this form were the focus of many of the early works on distributed processing [3], [4], [5], [9], [31].

More generally, with each agent, we associate a local objective function $J_k(w_k) = \mathbb{E}Q(w_k; \boldsymbol{x}_{k,i})$, where $\boldsymbol{x}_{k,i}$ refers to the data that are available to agent k, and $Q(w_k; \boldsymbol{x}_{k,i})$ is a loss function. Setting $Q(w_k; \boldsymbol{x}_{k,i}) \triangleq \left\| \boldsymbol{\gamma}_{k,i} - \boldsymbol{h}_{k,i}^{\mathsf{T}} w_k \right\|^2$ recovers the MSE loss leading to (4). We consider the general class of coupled optimization problems

$$\min_{w \in \mathbb{R}^{KM}} \left\{ \sum_{k=1}^{K} J_k(w_k) + \eta R(w) \right\}, \quad \text{subject to } w \in \Omega. \tag{5}$$

The coupling regularizer $R(w)$ and constraint $w \in \Omega$ encode relationships among local objectives and encourages local cooperation. Letting $R(w) = (1/2) w^{\mathsf{T}} \mathcal{L} w$ and $\Omega = \mathbb{R}^{KM}$ recovers the smoothed aggregate learning problem (4). Although decentralized algorithms for learning and optimization can be developed for general asymmetric-adjacency matrices $C \neq C^{\mathsf{T}}$ [22], [42], for the sake of simplicity, we focus on symmetric-adjacency matrices in this section. We comment on the implications of employing asymmetric combination policies in the "Asymmetric Combination Policies" section.

A common theme in many networked data processing applications is the limited access to the cost function and its gradient.

Stochastic gradient approximations

A common theme in many networked data processing applications is the limited access to the cost function $J_k(\cdot)$ and its gradient $\nabla J_k(\cdot)$ due to the fact that the cost $J_k(\cdot)$ is defined as the expected value of the loss $Q_k(\cdot; \boldsymbol{x}_{k,i})$, and $\boldsymbol{x}_{k,i}$ follows some unknown distribution. As a result, gradient descent algorithms that rely on the use of the true gradient $\nabla J_k(\cdot)$ are replaced by stochastic gradient algorithms, which employ an approximated gradient denoted by $\widehat{\nabla J}_k(\cdot)$. The most common construction for a stochastic gradient approximation is to employ $\widehat{\nabla J}_k(\cdot) \triangleq \nabla Q(\cdot; \boldsymbol{x}_{k,i})$, where $\boldsymbol{x}_{k,i}$ denotes a single sample of the variable $\boldsymbol{x}_k$ obtained at time i. However, other constructions are possible depending on the setting. For example, we may envision a scenario where agent k is provided with several independent samples $\{\boldsymbol{x}_{k,i,b}\}_{b=1}^{B_k}$ at time i, allowing for the minibatch construction $\widehat{\nabla J}_k(\cdot) \triangleq (1/B_k) \Sigma_{b=1}^{B_k} \nabla Q(\cdot; \boldsymbol{x}_{k,i,b})$. Alternatively, one may be faced with a situation where agents may be able to provide a gradient approximation with only some probability π_k, either due to lack of data, slow or delayed updates, or computational failure. Such asynchronous behavior can be modeled via [26]

$$\widehat{\nabla J}_k(\cdot) = \begin{cases} \dfrac{1}{\pi_k} \nabla Q(\cdot; \boldsymbol{x}_{k,i}), & \text{with prob. } \pi_k, \\ 0, & \text{with prob. } 1 - \pi_k. \end{cases} \tag{6}$$

As a final example of commonly used constructions for stochastic gradient approximations, we note perturbed stochastic gradients of the form $\widehat{\nabla J}_k(\cdot) = \nabla Q(\cdot; \boldsymbol{x}_{k,i}) + \boldsymbol{r}_{k,i}$, where $\boldsymbol{r}_{k,i}$ denotes some additional zero-mean noise. Examples of settings where additional noise is added to gradient approximations are plentiful, and include noise added due to quantization, noise used to ensure differential privacy, or noise used to escape from saddle points in nonconvex environments [24]. As we will see in the learning guarantees that we survey later, performance of the algorithms based on stochastic gradient approximations will in some way depend on the quality of $\widehat{\nabla J}_k(\cdot)$. Most commonly, this is quantified through bounds on its variance.

Condition 1 (variance of the gradient approximation)

The gradient approximation $\widehat{\nabla J}_k(\boldsymbol{w}_{k,i-1})$ is required to be unbiased with bounded variance as follows:

$$\mathbb{E}\{\widehat{\nabla J}_k(\boldsymbol{w}_{k,i-1}) \,|\, \boldsymbol{w}_{k,i-1}\} = \nabla J_k(\boldsymbol{w}_{k,i-1}) \tag{7}$$

$$\mathbb{E}\{\| \widehat{\nabla J}_k(\boldsymbol{w}_{k,i-1}) - \nabla J_k(\boldsymbol{w}_{k,i-1}) \|^2 \boldsymbol{w}_{k,i-1}\} \leq \beta_k^2 \left\| w_k^o - \boldsymbol{w}_{k,i-1} \right\|^2 + \sigma_k^2. \tag{8}$$

Here, β_k^2, σ_k^2 denote nonnegative constants, and w_k^o denotes an arbitrary reference point, most commonly, the minimizing argument from (3).

As previously shown in [38], the zero-mean condition (7) can be verified to hold for many popular constructions, including the constructions listed previously. In (7) and (8), we condition on the current iterate $\boldsymbol{w}_{k,i-1}$ and take expectation with respect to the remaining variability in generating the gradient approximation $\widehat{\nabla J}_k(\boldsymbol{w}_{k,i-1})$, which is the data available to agent k at time i. For example, in the case of ordinary stochastic gradient descent $\widehat{\nabla J}_k(\cdot) \triangleq \nabla Q(\cdot; \boldsymbol{x}_{k,i})$, this corresponds to $\boldsymbol{x}_{k,i}$, which is generally assumed to be independent of $\boldsymbol{w}_{k,i-1}$. Variance bounds of the form (8), on the other hand, need to be verified for specific choices of loss functions $Q(\cdot; \boldsymbol{x}_k)$, distributions of the data $\boldsymbol{x}_{k,i}$, and gradient approximations $\widehat{\nabla J}_k(\cdot)$. Nevertheless, the key takeaway is that conditions of this form hold for most processing and learning problems of interest. The resulting constants β_k^2, σ_k^2 quantify the quality of the utilized gradient approximation. In Table 1, we list the relevant quantities for the MSE and logistic loss as examples. It is also useful to note that,

Table 1. Constants in gradient variance bounds for popular loss functions for supervised learning problems with $\boldsymbol{x}_k \triangleq \text{col}\{\boldsymbol{h}_k, \gamma_k\}$ [22]. The quantities σ_v^2 and R_h denote the data statistics $\mathbb{E}\boldsymbol{v}_{k,i}^2$ and $\mathbb{E}\boldsymbol{h}_{k,i}\boldsymbol{h}_{k,i}^{\mathsf{T}}$, respectively.

Loss	Gradient Approximation	β_k^2 (Relative)	σ_k^2 (Absolute)
$\frac{1}{2}\left\| \boldsymbol{\gamma}_{k,i} - \boldsymbol{h}_{k,i}^{\mathsf{T}} w \right\|^2$	$\boldsymbol{h}_{k,i}(\boldsymbol{\gamma}_{k,i} - \boldsymbol{h}_{k,i}^{\mathsf{T}} w)$	$\mathbb{E}\left\| R_h - \boldsymbol{h}_k \boldsymbol{h}_k^{\mathsf{T}} \right\|^2$	$\sigma_v^2 \text{Tr}(R_h)$
$\ln(1 + e^{-\boldsymbol{\gamma}_{k,i}\boldsymbol{h}_{k,i}^{\mathsf{T}} w}) + \frac{\rho}{2}\left\| w \right\|^2$	$\boldsymbol{\gamma}_{k,i}\boldsymbol{h}_{k,i}\left(\frac{1}{1 + e^{\boldsymbol{\gamma}_{k,i}\boldsymbol{h}_{k,i}^{\mathsf{T}} w}}\right) + \rho w$	0	$\text{Tr}(R_h)$

given the constants β_k^2, σ_k^2 for an ordinary gradient approximation, such as those listed in Table 1, one can immediately recover those of the variants listed previously; this is illustrated in Table 2.

We finally note that the current exposition mainly focuses on methods that assume first-order (i.e., gradient-type) information is available or accessible in the construction of the distributed algorithms. Due to intractability of gradient computation in certain applications (for instance, in scenarios where the cost model is not directly available but perhaps may be computed at desired query points via noisy simulations), one can resort to zeroth-order approaches. In this case, noisy and biased gradient estimates obtained from measuring function values using various difference approximations are used in the algorithm design in lieu of exact or unbiased gradients as assumed in the first-order setting (see [38], [43], and the references therein for more details).

Table 2. The quantities $\beta_{k,\text{ord}}^2$, $\sigma_{k,\text{ord}}^2$ correspond to the gradient noise constants of the "ordinary" gradient approximation $\widehat{\nabla J_k}(\boldsymbol{w}_{k,i-1}) = \nabla Q(\boldsymbol{w}_{k,i-1}, \boldsymbol{x}_{k,i})$, and can be read from Table 1. Constants of the variants are provided in terms of the baseline quantities $\beta_{k,\text{ord}}^2$, $\sigma_{k,\text{ord}}^2$. The parameter δ_k corresponds to the Lipschitz constant of the gradient $\nabla J_k(\cdot)$.

$\widehat{\nabla J_k}(\boldsymbol{w}_{k,i-1})$	Unbiased?	β_k^2 (Relative)	σ_k^2 (Absolute)
$\nabla Q(\boldsymbol{w}_{k,i-1}, \boldsymbol{x}_{k,i})$	Yes	$\beta_{k,\text{ord}}^2$	$\sigma_{k,\text{ord}}^2$
$\frac{1}{B}\sum_{b=1}^{B} \nabla Q(\boldsymbol{w}_{k,i-1}, \boldsymbol{x}_{k,i,b})$	Yes	$\frac{\beta_{k,\text{ord}}^2}{B}$	$\frac{\sigma_{k,\text{ord}}^2}{B}$
Asynchronous as in (6)	Yes	$\frac{\beta_{k,\text{ord}}^2}{\pi_k} + \frac{1-\pi_k}{\pi_k}\delta_k^2$	$\frac{\sigma_{k,\text{ord}}^2}{\pi_k}$
$\widehat{\nabla J_k}(\cdot) = \nabla Q(\cdot; \boldsymbol{x}_{k,i}) + \boldsymbol{r}_{k,i}$	Yes	$\beta_{k,\text{ord}}^2$	$\sigma_{k,\text{ord}}^2 + \sigma_{r,k}^2$

Task relationships

As a separate consideration from the choice of the risk functions $J_k(w_k)$, one may consider various frameworks for the relation between individual models w_k, also referred to as *tasks*. In the absence of coupling regularization or constraints, i.e., in the case the regularizer $R(\{w_k\}_{k=1}^K) = 0$ and $\Omega = \mathbb{R}^{KM}$, optimization over the aggregate cost $\Sigma_{k=1}^K J_k(w_k)$ decouples into independent problems $J_k(w_k)$ over local models w_k. These can be pursued in a noncooperative manner.

Perhaps the most commonly studied framework for distributed optimization is that of consensus optimization, where individual models are required to be common, i.e., $w_k = w$, giving rise to

$$\min_{w} \sum_{k=1}^{K} J_k(w). \tag{9}$$

Networked algorithms for (9) can be developed from (5) in several ways, giving rise to different families of algorithms for distributed optimization [38], as we proceed to show.

Penalty-based approaches

We may encourage consensus by penalizing pairwise differences between connected agents, i.e., $R(w) = (1/2)\Sigma_{k=1}^K \Sigma_{\ell\in\mathcal{N}_k} c_{\ell k} \| w_k - w_\ell \|^2$, resulting in

$$\begin{aligned} &\min_{\{w_k\}_{k=1}^K} \left\{ \sum_{k=1}^{K} J_k(w_k) + \frac{\eta}{2} \sum_{k=1}^{K} \sum_{\ell\in\mathcal{N}_k} c_{\ell k} \| w_k - w_\ell \|^2 \right\} \\ &\quad \Longleftrightarrow \min_{w} \left\{ \mathcal{J}(w) + \frac{\eta}{2} w^{\mathsf{T}} \mathcal{L} w \right\} \end{aligned} \tag{10}$$

where, in addition to making use of (5), we defined $\mathcal{J}(w) \triangleq \Sigma_{k=1}^K J_k(w_k)$. It can be verified that, as long as the graph described by C is connected, $(1/2)\Sigma_{k=1}^K \Sigma_{\ell\in\mathcal{N}_k} c_{\ell k} \| w_k - w_\ell \|^2 = 0$, if and only if $w_k = w$ for all k, and hence, (10) is equivalent to (9) in the limit as $\eta \to \infty$. At the same time, this fact implies that for finite η, problems (10) and (9) will, in general, have distinct solutions. It is for this reason that penalty-based methods generally operate with large choices of the penalty parameter η, exhibiting some small bias relative to the exact consensus problem (9), unless $\eta \to \infty$. Applying stochastic gradient descent to (10) results in

$$\boldsymbol{w}_i = (I - \mu\eta\mathcal{L})\boldsymbol{w}_{i-1} - \mu\widehat{\nabla\mathcal{J}}(\boldsymbol{w}_{i-1}). \tag{11}$$

If we set $A \triangleq I - \mu\eta\mathcal{L}$ and return to node-level quantities, we recover the recursion

$$\boldsymbol{w}_{k,i} = \sum_{\ell\in\mathcal{N}_k} a_{\ell k} \boldsymbol{w}_{\ell,i-1} - \mu\widehat{\nabla J}_k(\boldsymbol{w}_{k,i-1}) \tag{12}$$

which corresponds to the decentralized (stochastic) gradient descent algorithm [12], [29] of the "consensus + innovation" type [8]. If we instead, following [44], appeal to an incremental gradient descent argument, where we first take a step relative to the cost $\mathcal{J}(w)$, and subsequently descend along the penalty $(\eta/2)w^{\mathsf{T}}\mathcal{L}w$, we obtain the adapt-then-combine (ATC) diffusion algorithm [3], [21]

$$\boldsymbol{\psi}_{k,i} = \boldsymbol{w}_{k,i-1} - \mu\widehat{\nabla J}_k(\boldsymbol{w}_{k,i-1}) \tag{13}$$

$$\boldsymbol{w}_{k,i} = \sum_{\ell\in\mathcal{N}_k} a_{\ell k} \boldsymbol{\psi}_{\ell,i}. \tag{14}$$

Reversing the order in the argument instead yields the combine-then-adapt variation of diffusion [3], [21].

Imperfect and noisy communication

In the exposition thus far, we have assumed for simplicity that the interagent communication is perfect. In practice, we may have random packet dropouts or link failures and distortions in the data exchanged by agents due to channel noise, quantization, or other forms of compression. There has been extensive research on consensus and diffusion procedures that deal with time-varying or stochastic Laplacian matrices to model issues such as link failures, whereas in other instances, controlled randomization in the communication has been used via random Laplacians as a tool to improve

communication efficiency (see [5], [26], [29], and [30] for a sample of the relevant literature). On the other hand, noise in the observations or communication, either injected as additive communication noise or through quantization and other forms of compression, are handled by carefully designing the mixing parameters, the $a_{\ell k}$'s in (12)–(14), and building on tools from stochastic approximation as in the "Stochastic Gradient Approximations" section [8], [26], or through the use of probabilistic ideas such as dithering [33]. Most of the development in the current article will continue to hold for such imperfect interagent communication through appropriate modifications as discussed earlier.

Primal-dual approaches

As an alternative to penalty-based approaches, one may wish to enforce exact consensus by introducing constraints, such as [38]

$$\min_{\{w_k\}_{k=1}^K} \sum_{k=1}^K J_k(w_k) \quad \text{s.t.} \sum_{k=1}^K \sum_{\ell\in\mathcal{N}_k} c_{\ell k} \left\| w_k - w_\ell \right\|^2 = 0. \tag{15}$$

In contrast to penalty-based formulations, constrained formulations of the consensus problem can no longer be pursued using pure gradient-based algorithms. Instead, constraints are most commonly enforced through dual arguments such as alternating direction method of multipliers (ADMM), dual averaging, or the augmented Lagrangian. Early algorithms involving primal-dual arguments for exact consensus optimization, such as [13], [16], [45], and [46], involved the propagation and communication of dual variables in addition to weight vectors w_k. ADMM-based algorithms [13], [46] generally involve two timescales, where an auxiliary optimization problem is solved in between every outer iteration. Although these methods exhibit appealing convergence properties, their implementation is only practical in situations where the inner optimization problem has a specific structure that allows it to be solved efficiently or in a closed form.

Single timescale primal-dual algorithms [16], [45] instead employ first-order approximations at every, step thus avoiding the need to solve a costly inner optimization problem. As a representative example, we list here the stochastic, first-order augmented Lagrangian strategy from [45]

$$\boldsymbol{w}_i = (I - \mu\eta\mathcal{L})\boldsymbol{w}_{i-1} - \mu\eta\mathcal{B}^{\mathsf{T}}\boldsymbol{\lambda}_{i-1} - \mu\widehat{\nabla\mathcal{J}}(\boldsymbol{w}_{i-1}) \tag{16}$$

$$\boldsymbol{\lambda}_i = \boldsymbol{\lambda}_{i-1} + \mu\eta\mathcal{B}\boldsymbol{w}_{i-1} \tag{17}$$

where $L = B^{\mathsf{T}}B$. An examination of (16) reveals that the augmented Lagrangian-based strategy corrects the "consensus + innovation" algorithm (12) by adding the additional term $-\mu\eta\mathcal{B}^{\mathsf{T}}\boldsymbol{\lambda}_{i-1}$, which compensates the bias induced by the penalty-based formulation (10). Although effective at ensuring an exact consensus, the propagation of dual variables is associated with additional overhead in terms of both computation and communication. Conveniently, dual variables can frequently be eliminated and replaced by a momentum-like term. To illustrate this point, let us consider a variant of (16) and (17), where the primal and dual updates are performed in an incremental manner, allowing the dual update to make use of the most recent primal variable $\boldsymbol{w}_i$, rather than $\boldsymbol{w}_{i-1}$. This results in

$$\boldsymbol{w}_i = (I - \mu\eta\mathcal{L})\boldsymbol{w}_{i-1} - \mu\eta\mathcal{B}^{\mathsf{T}}\boldsymbol{\lambda}_{i-1} - \mu\widehat{\nabla\mathcal{J}}(\boldsymbol{w}_{i-1}) \tag{18}$$

$$\boldsymbol{\lambda}_i = \boldsymbol{\lambda}_{i-1} + \mu\eta\mathcal{B}\boldsymbol{w}_i. \tag{19}$$

After setting $\eta = 1/\mu$ and following [47], we can verify that (18) and (19) are equivalent to

$$\begin{aligned}\boldsymbol{w}_i = {} & 2(I - \mathcal{L})\boldsymbol{w}_{i-1} - (I - \mathcal{L})\boldsymbol{w}_{i-2} \\ & - \mu(\widehat{\nabla\mathcal{J}}(\boldsymbol{w}_{i-1}) - \widehat{\nabla\mathcal{J}}(\boldsymbol{w}_{i-2}))\end{aligned} \tag{20}$$

which is equivalent to the EXTRA algorithm of [15] for appropriately chosen weight matrices. Letting $A \triangleq I - L$, and returning to node-level quantities we obtain

$$\boldsymbol{\phi}_{k,i} = \sum_{\ell\in\mathcal{N}_k} a_{\ell k}\boldsymbol{w}_{\ell,i-1} - \mu\widehat{\nabla J}_k(\boldsymbol{w}_{k,i-1}) \tag{21}$$

$$\boldsymbol{w}_{k,i} = \boldsymbol{\phi}_{k,i} + \sum_{\ell\in\mathcal{N}_k} a_{\ell k}\boldsymbol{w}_{\ell,i-1} - \boldsymbol{\phi}_{k,i-1}. \tag{22}$$

These recursions can again be identified as a bias-corrected version of the "consensus + innovation" recursion (12), but now rely on the momentum term $\sum_{\ell\in\mathcal{N}_k} a_{\ell k}\boldsymbol{w}_{\ell,i-1} - \boldsymbol{\phi}_{k,i-1}$ rather than the propagation of dual variables, as in (17). Making the same incremental gradient adjustment that led to the penalty-based ATC diffusion algorithm (13) and (14), we obtain the exact diffusion algorithm from [44]

$$\boldsymbol{\psi}_{k,i} = \boldsymbol{w}_{k,i-1} - \mu\widehat{\nabla J}_k(\boldsymbol{w}_{k,i-1}) \tag{23}$$

$$\boldsymbol{\phi}_{k,i} = \boldsymbol{\psi}_{k,i} + \boldsymbol{w}_{k,i-1} - \boldsymbol{\psi}_{k,i-1} \tag{24}$$

$$\boldsymbol{w}_{k,i} = \sum_{\ell\in\mathcal{N}_k} a_{\ell k}\boldsymbol{\phi}_{\ell,i}. \tag{25}$$

Exact diffusion is also referred to as D^2 [48] or *NIDS* [60].

Gradient-tracking-based approaches

An alternative to the approaches described previously is based on gradient tracking. Although the initial motivation [19], [49] for the construction was based on the dynamic average consensus algorithm, it has been noted in [50] that gradient-tracking-based algorithms for decentralized optimization can be viewed as a variation of the primal-dual arguments leading to the EXTRA and exact diffusion algorithms described earlier. The interested reader is referred to [38] and [50] for details, and simply list the resulting NEXT [19]/DIGing [50] algorithm

$$\boldsymbol{w}_{k,i} = \sum_{\ell\in\mathcal{N}_k} a_{\ell k}\boldsymbol{w}_{\ell,i-1} - \mu\boldsymbol{g}_{k,i-1} \tag{26}$$

$$\boldsymbol{g}_{k,i} = \sum_{\ell\in\mathcal{N}_k} a_{\ell k}\boldsymbol{g}_{\ell,i-1} + \widehat{\nabla J}_k(\boldsymbol{w}_{k,i}) - \widehat{\nabla J}_k(\boldsymbol{w}_{k,i-1}). \tag{27}$$

Following an incremental construction, on the other hand, analogous to the step from EXTRA to exact diffusion mentioned before, results in an ATC variant of the NEXT/DIGing algorithm, proposed in [49] under the name *Augmented Distributed Gradient Method (Aug-DGM)*

$$\boldsymbol{\psi}_{k,i} = \boldsymbol{w}_{k,i-1} - \mu \boldsymbol{g}_{k,i-1} \tag{28}$$

$$\boldsymbol{w}_{k,i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \boldsymbol{\psi}_{\ell,i} \tag{29}$$

$$\boldsymbol{g}_{k,i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \big(\boldsymbol{g}_{\ell,i-1} + \widehat{\nabla J_\ell}(\boldsymbol{w}_{\ell,i}) - \widehat{\nabla J_\ell}(\boldsymbol{w}_{\ell,i-1}) \big). \tag{30}$$

Decentralized algorithms for consensus optimization based on gradient tracking generally have similar convergence properties to their primal-dual counterparts EXTRA and exact diffusion. One key difference is the fact that exchanges of the gradient estimate $\boldsymbol{g}_{k,i}$, in addition to the local models, result in an increase in communication cost roughly at a factor of two.

Constrained learning

In our discussion thus far, the constraint Ω of (5) has been used to encode equality constraints of the form $\Omega = \Big\{ w \,|\, \Sigma_{k=1}^K \Sigma_{\ell \in \mathcal{N}_k} c_{\ell k} \big\| w_k - w_\ell \big\|^2 = 0 \Big\} = \{ \boldsymbol{w} \,|\, w_k = w \;\forall\, k \}$. This ensures consensus on a common model w. In many applications, we may wish to further constrain the common model w to some set Θ. The variations of most of the algorithms described in the "Unified View" section for constrained optimization and learning were developed and studied by employing Euclidean and proximal projections or penalty functions [14], [51], [52], [53]. These solve the constrained consensus optimization problem $\min_{w \in \Theta} \Sigma_{k=1}^K J_k(w)$. For example, applying the same incremental argument that led to (13) and (14), followed by projection onto Θ, leads to a projected variant of the ATC diffusion or consensus + innovation algorithm (12)–(14), studied in [14] and [53]

$$\boldsymbol{\psi}_{k,i} = \boldsymbol{w}_{k,i-1} - \mu \widehat{\nabla J_k}(\boldsymbol{w}_{k,i-1}) \tag{31}$$

$$\boldsymbol{w}_{k,i} = \mathrm{Proj}_\Theta \Big(\sum_{\ell \in \mathcal{N}_k} a_{\ell k} \boldsymbol{\psi}_{\ell,i} \Big). \tag{32}$$

Similarly, we may introduce projections into primal-dual algorithms to derive projected variants of primal-dual algorithms, such as the PG-EXTRA generalization [51] of the EXTRA algorithm (20).

Multitask learning

Although the pursuit of an optimal average model, as defined in (9), is appropriate in many situations, it is important to recognize that a good average model may perform poorly on any local cost $J_k(\cdot)$. This observation motivates the pursuit of networked multitask learning algorithms [23], where agents aim to learn from one another without forcing an exact consensus. More recently, this area has received attention under the name *personalized federated learning*. Multitask learning is generally achieved using variations of the regularized aggregate problem (5), where the regularization is chosen to match some underlying prior on task relationships (rather than to enforce exact consensus). Solutions can again be pursued using primal or primal-dual approaches. Due to space limitations, a detailed treatment of multitask learning is beyond the scope of this article, so we refer the reader to [23] and the references therein.

Applications

Weather prediction

The task of predicting weather patterns naturally lends itself to networked solutions because 1) measurements tend to be available in dispersed locations, and 2) it is reasonable to believe that weather models ought to be related in adjacent regions, encouraging the diffusion of information as a means of improving performance. To illustrate this fact, here we reproduce a simulation study from [54]. The simulation is based on meteorological data from across the United States, shown in Figure 3(a) and (b). The implementation is based on the regularized learning problem (4), with logistic risks $J_k(w_k) = \mathbb{E} \ln\big(1 + e^{-\boldsymbol{\gamma}_{k,i} \boldsymbol{h}_{k,i}^\mathsf{T} w_k}\big) + \rho \big\| w_k \big\|^2$. The performance is shown in Figure 3(c), where the choice $\eta = 0$ corresponds to a noncooperative implementation, $\eta = \mu^{-1}$ corresponds to the ATC diffusion algorithm (13) and (14), and other choices of η correspond to softer coupling of local models. Due to space limitations, we refer the reader to [54] for a more detailed discussion of the setup and results.

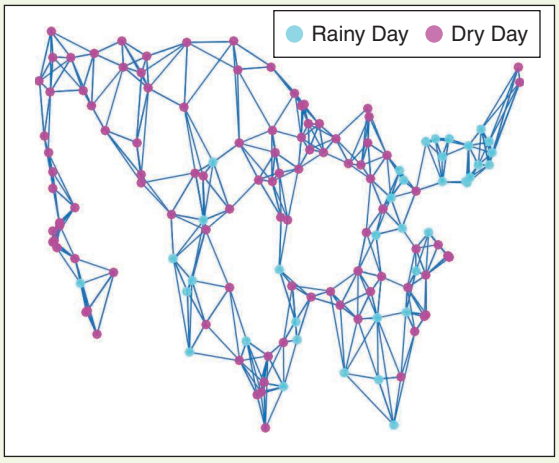
Wide area monitoring in power systems

Wide area monitoring in power transmission systems consists of tracking the overall system state based on measurements obtained at control areas or balancing authorities (nodes or agents in our current exposition). The geographical distribution and practical data sharing limitations among the control areas naturally calls for distributed state estimation algorithms (see [55]), with the goal of monitoring the global system's state while minimizing data exchange among the control areas. In [55], fully distributed approaches for wide area state estimation based on consensus + innovation-type algorithms [see (12)] are proposed, both for dc and ac state estimation. The typical quantities of interest in wide area monitoring are voltage magnitudes and relative angles (phases) at the system buses based on power flow measurements at subsets of transmission lines, and power injection measurements at the system buses. In dc state estimation, the bus voltage magnitudes are typically assumed to be at a nominal 1 per-unit reference value (see [55] for details), and the unknown phase estimation at the buses reduces to a linear least-squares-type formulation, as in (2).

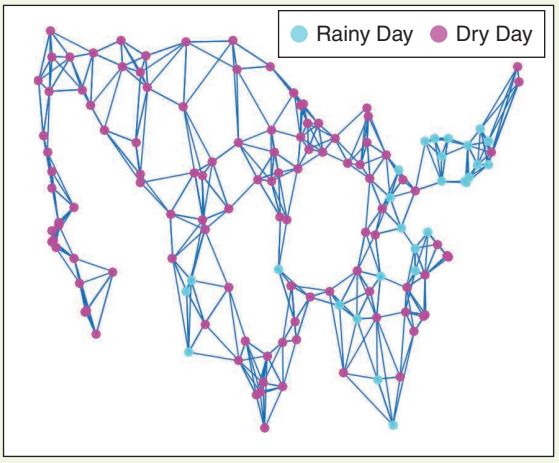
In particular, Figure 4 shows the application of a consensus + innovation approach with decaying step-sizes (taken from [55]) for dc state estimation on an IEEE 118-bus benchmark test system: Figure 4(a) depicts the 118-bus system partitioned into eight control areas that communicate over a connected communication graph (typically, this graph conforms to the physical coupling between the control areas, or is chosen based on geographical proximity); the application essentially consists of reformulating the wide area phase estimation objective as a least-squares cost minimization, with $J_k(w_k)$ of

the form in (2) and (3), and applying the consensus + innovation approach.

In Figure 4(a), we compare the gap (referred to as the *phase angle gap*) between the relative phases obtained by the iterative distributed approach, and those from a hypothetical, fusion-center-based, optimal one-shot least-squares estimator across multiple bus pairs, i.e., for instance, the quantity $g_{1,2}$ denotes the gap between the phase difference between buses 1 and 2 obtained by the distributed approach and that obtained by the centralized estimator. As expected, by the convergence guarantees discussed in the "Task Relationships" section, these gaps converge to zero as the iterations progress. Similar distributed approaches may be used for distributed (nonlinear) ac state estimation where the objective is to estimate both the bus voltages and relative angles. This is performed by resorting to a nonlinear least-squares-type minimization in [53] and [55], and applying a projected variant of the consensus + innovation approach [see the discussion pertaining to (31) and (32)] to deal with certain trigonometric nonlinearities associated with the ac power flow model.



(a)

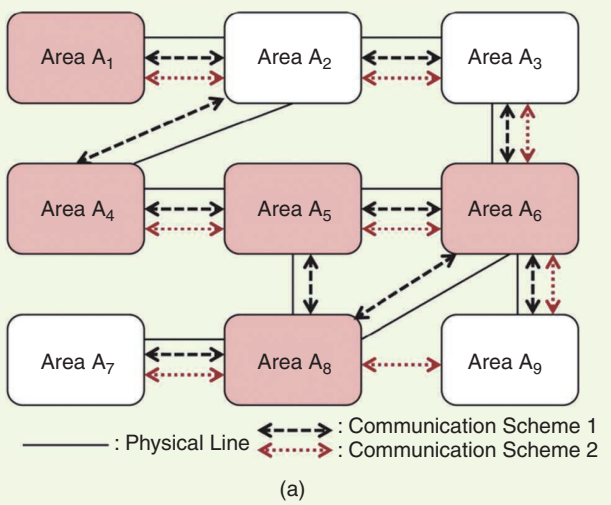


(b)

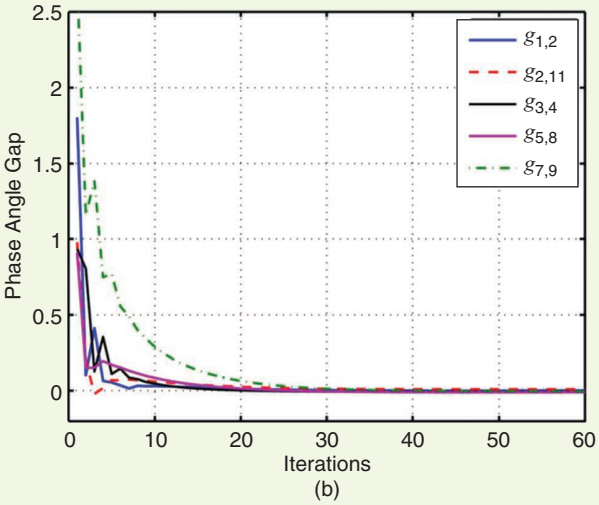
	$\eta = 0$	$\eta = 10$	$\eta = 45$	$\eta = 100$	$\eta = 1{,}000$	$\eta = \mu^{-1}$
Prediction Error	0.309	0.232	**0.225**	0.226	0.228	0.232

(c)

FIGURE 3. Weather prediction using diffusion algorithms (taken from [54]). (a) Actual occurrence of rain. (b) Predicted occurrence of rain. (c) Prediction accuracy as a function of the regularization parameter η of (4).



(a)



(b)

FIGURE 4. Wide area state estimation using consensus + innovation algorithms (taken from [55]). (a) An IEEE 118-bus system partitioned into eight control areas or nodes with possible internode communication patterns. (b) The relative phase angle estimation error at different bus pairs.

Performance guarantees in distributed learning

We now proceed to survey some performance guarantees of algorithms for decentralized learning, with a particular emphasis on stochastic settings. Given space limitations, it is not possible for us to provide a comprehensive survey; instead, we aim to highlight some key insights that have emerged from an extensive body of work over the past two decades, in an attempt to provide the reader with a starting point and guidelines when matching the choice of a learning algorithm to a problem at hand.

Algorithms with vanishing step-sizes can converge asymptotically to the exact minimizer with zero error, albeit at a slower rate than when constant step-sizes are used.

Constant and diminishing step-sizes

Optimization algorithms based on stochastic gradient approximations are subject to persistent noise, and hence, generally, do not converge to exact solutions. This can be remedied by employing a time-varying and diminishing step-size, resulting in slower but exact convergence. We highlight this by comparing the performance guarantees of the primal consensus and diffusion algorithms from [3], [8], and [21], although similar conclusions apply to other decentralized algorithms. For strongly convex costs and using a constant step-size construction, the asymptotic performance of the penalty-based algorithms described in the "Task Relationships" section is given by [22, Example 11.8]

$$\limsup_{i\to\infty} \mathrm{ER}_i = \frac{\mu}{4}\sum_{k=1}^{K} p_k^2\sigma_k^2 + o(\mu). \tag{33}$$

The excess-risk (ER) measures the average suboptimality $\mathrm{ER}_i \triangleq (1/K)\sum_{k=1}^{K}\mathbb{E}J(\boldsymbol{w}_{k,i}) - J(w^o)$. The notation $o(\mu)$ denotes terms that are higher order in the step-size and hence negligible for sufficiently small step-sizes μ. The constants σ_k^2 correspond to the absolute gradient noise variance of (8). The analysis in [22] is performed for general left-stochastic adjacency matrices $A \neq A^\top$ with Perron eigenvector $Ap = p$. For symmetric-adjacency matrices, the weights reduce to $p_k = 1/K$. Convergence to the steady-state value occurs at a linear rate given by $\alpha = 1 - 2\nu\mu + o(\mu)$ [22, eq. (11.139)], where ν denotes the strong-convexity constant of the aggregate cost $J(w)$. If we instead employ a diminishing step-size of the form $\mu_i = 1/i$, it holds asymptotically for large i that [36]

$$\mathrm{ER}_i = \frac{\mu}{4i}\sum_{k=1}^{K} p_k^2\sigma_k^2 + o(\mu) \tag{34}$$

and hence, $\lim_{i\to\infty}\mathrm{ER}_i = 0$. At the same time, we note that convergence using a diminishing step-size occurs at a sublinear rate $O(1/i)$, rather than the linear rate $O((1-2\mu\nu+o(\mu))^i)$.

These remarks reflect a well-known trade-off between convergence rate, asymptotic error, and tracking ability of a learning algorithm [8], [36], [56]. Algorithms with vanishing step-sizes can converge asymptotically to the exact minimizer with zero error, albeit at a slower rate than when constant step-sizes are used. In this latter case, the algorithms approach the minimizer at a faster exponential rate, albeit within an MSE range that is proportional to the step-size parameter. When this parameter is small, as is normally the case, this construction enables the algorithm with a constant step-size to track drifts in the underlying parameter when the statistical properties of the data change with time. Often, implementations in practice may use a combination of vanishing and constant step-sizes. On the other hand, when one is interested in asymptotic convergence of the error to zero, then it is known from statistical learning theory and large-sample asymptotics in parameter estimation, that the $O(1/i)$ rate is optimal for statistically consistent online estimators (i.e., estimators that achieve asymptotically zero error almost surely), where i, the iteration count, coincides with or is proportional to the number of data points or online stochastic gradients sampled during the estimation process. Further, in this online scenario, within the class of statistically consistent estimators, the ones with optimal asymptotic variance, i.e., asymptotically efficient estimators, may be obtained by appropriately tuning the diminishing step-size sequences (see, for example, [8], [36], and [56]); in such scenarios the distributed estimators end up achieving the optimal online-centralized error rates.

Both nonvanishing and vanishing weights algorithms can overcome a lack of knowledge of model parameters or noise statistics, for example, by replacing noise mean and covariance by empirical sample estimates, like distributed recursive least-squares (RLS) [9], still guaranteeing the distributed algorithm's stability and error mean and covariance asymptotic optimality. With vanishing weights, to guarantee optimal asymptotic MSE in the sense of Fisher information rate, algorithm (12) should be augmented by a recursion for the gain of the innovations or data term [56] so that the agents engage on distributed learning to asymptotically recover the optimal gains, while simultaneously carrying out their distributed task with negligible asymptotic information rate loss.

Linear gains in performance

If we consider symmetric-adjacency matrices $A = A^\top$, resulting in $p_k = 1/K$, and homogeneous data profiles $\sigma_k^2 = \sigma^2$, we note that for both constant (33) and diminishing step-sizes (34), the asymptotic ER scales with $O(\mu\sigma^2/K)$. The scaling by the network size K is referred to as *linear gain*. It is consistent with the performance gains that can be expected when fusing raw data in a centralized architecture (see, e.g., [22, Th. 5.1]) and provides motivation for agents to participate in the cooperative learning protocol. Analogous results have been obtained for primal-dual algorithms [57] as well as in the pursuit of second-order stationary points [24] in nonconvex environments.

Penalty-based and primal-dual algorithms

A motivation for considering primal-dual algorithms for decentralized optimization over penalty-based construction is the removal of the bias induced by employing a finite regularization term (10) in place of (15). When exact gradients are employed and no noise is added due to the use of stochastic gradient

approximations, this results in a pronounced difference in performance as primal-dual algorithms are able to converge linearly and exactly, using a constant step-size, in strongly convex environments [15], [16], [44], [46], [50], while penalty algorithms require a diminishing step-size to ensure exact convergence, resulting in a sublinear rate [12].

In the stochastic setting, however, iterates are subjected to additional perturbations induced by the utilization of data-dependent, stochastic gradient approximations. This causes the difference in performance between penalty-based and primal-dual algorithms to be more nuanced. For example, it was shown in [45] that the primal-dual algorithm (16) and (17) exhibits strictly worse performance than penalty-based approaches, such as consensus and diffusion algorithms, when constant step-sizes and stochastic gradient approximations are employed. This is due to the fact that the penalty-based algorithms exhibit lower variance in steady state, which compensates for the additional bias. On the other hand, stochastic variants of exact diffusion [44] and gradient tracking [19] have been shown analytically and empirically to improve upon the performance of their penalty-based counterparts. We illustrate this by reviewing the results in [57] as a case study. For the diffusion algorithms, as an example of a penalty-based algorithm, the authors derived a refined version of the bound (33) of the form

$$\limsup_{i\to\infty} \mathrm{MSD}_i = O\left(\frac{\mu\sigma^2}{K} + \frac{\mu^2\lambda^2\sigma^2}{1-\lambda} + \frac{\mu^2\lambda^2 b^2}{(1-\lambda)^2}\right). \tag{35}$$

The mean square deviation (MSD) of the network is defined as $\mathrm{MSD}_i \triangleq (1/K)\sum_{k=1}^{K}\mathbb{E}\left\|w^o - \boldsymbol{w}_{k,i}\right\|^2$. For δ-smooth and ν-strongly convex $J(\cdot)$, it holds that $(\nu/2)\mathrm{MSD}_i \le \mathrm{ER}_i \le (\delta/2)\mathrm{MSD}_i$. The first term in the performance expression $\mu\sigma^2/K$ corresponds to the performance deterioration from employing stochastic gradient approximations with variance σ^2 and is proportional to the step-size μ. This term is consistent with (33). The other two terms scale with μ^2 and quantify the interplay between the mixing rate of the adjacency matrix $\lambda = \rho(A - (1/K)\mathbb{1}\mathbb{1}^\top)$ and the bias term $b^2 = 1/K\sum_{k=1}^{K}\left\|\nabla J_k(w^o)\right\|^2$. The mixing rate λ measures the level of connectivity of the network and is close to one whenever the adjacency matrix is sparse. The bias term b^2, on the other hand, measures the level of heterogeneity in the network. Both $O(\mu^2)$ terms become negligible as $\mu \to \infty$, but can be significant for very sparse, heterogeneous networks and moderate step-sizes.

For exact diffusion, as an example of a stochastic primal-dual algorithm, on the other hand, we have [44], [57]

$$\limsup_{i\to\infty} \mathrm{MSD}_i = O\left(\frac{\mu\sigma^2}{K} + \frac{\mu^2\lambda^2\sigma^2}{1-\lambda}\right). \tag{36}$$

We note the removal of the term $\mu^2\lambda^2 b^2/(1-\lambda)^2$. As a result, the performance no longer depends on the heterogeneity b^2 and has an improved dependence on the mixing rate λ (Figure 5). A similar improved dependence on network heterogeneity and connectivity has been observed in pursuing first-order stationary points of nonconvex problems as well [48].

The mixing rate λ measures the level of connectivity of the network and is close to one whenever the adjacency matrix is sparse.

Stability gains via incremental constructions

As we saw throughout the algorithm derivations in the "Unification Through Stochastic Optimization" section, the derivations of some decentralized algorithms rely on the use of incremental steps, such as the ATC diffusion algorithm [3], the exact diffusion algorithm [44], and the Aug-DGM algorithm [49]. These variants incorporate incremental steps in comparison to the "consensus + innovation" algorithm [8], [12], the EXTRA algorithm [15], and the DIGing algorithm [50]. It turns out that in many cases, the incremental steps endow the resulting algorithms with improved robustness and stability properties, particularly when employing constant and uncoordinated step-sizes and noisy gradient approximations. Early evidence of this phenomenon appears in [20] and [22], where it was shown that diffusion strategies based on the ATC construction, which is incremental, enjoy a wider stability range than consensus-based constructions. In particular, the stability range for the ATC diffusion algorithm can be independent of the network connectivity (as long as agents are locally stable), while the stability range for the consensus algorithm, in general, depends on the mixing rate of the adjacency matrix [20], [22]. Analogous observations were made in [44] when comparing the stability range of the exact diffusion algorithm to EXTRA.

Asynchronous behavior

The stochastic gradient approximations framework described in the "Stochastic Gradient Approximations" section is general enough to cover a large number of phenomena that may arise in the presence of asynchrony and imperfections, such as intermittent updates (6) or noisy links. The implications of these imperfections on performance follow from relations (33)–(36) after adjusting the gradient variance σ_k^2 according to Table 2. Another form of asynchrony, not directly covered within the gradient approximations framework, refers to time-varying, intermittent communication graphs. Such asynchrony can be more challenging as exchanges that now occur so infrequently can, in principle, result in divergent behavior, particularly for heterogeneous networks. Nevertheless, when properly designed, decentralized algorithms have been shown to be remarkably robust to asynchronous communication policies, including random [26] and deterministically time-varying policies [50]. The takeaway from these studies is that, as long as adjacency matrices are connected in expectation [26], or their union over time is connected [50], information can sufficiently diffuse, and agents can efficiently learn from each other.

Asymmetric combination policies

Most of our discussion thus far has focused on symmetric-adjacency matrices $A = A^\top$. Nevertheless, a decentralized algorithm for optimization and learning can also be deployed

with asymmetric matrices [3], [21], [22], [24], [44]. The effect of such constructions is that certain agents will be able to exert more or less influence over the behavior of the network. To be precise, we associate with the adjacency matrix its Perron eigenvector $Ap = p$, where p_k denotes the entry corresponding to agent k. It can then be shown that most decentralized algorithms will converge to the minimizer of the weighted sum, i.e., $w^o \triangleq \sum_{k=1}^{K} p_k J_k(w)$, where the weights p_k now modulate the relative influence of the cost $J_k(w)$ associated with agent k. For symmetric matrices $A = A^\top$, we have $p_k = 1/K$, and we recover (9). The ability of certain agents to be more or less influential within the network adds a degree of freedom to the design of a multiagent system. In heterogeneous environments, where some agents may have access to data or gradient approximations of higher quality, this can be exploited to improve performance or convergence rate [22]. On the other hand, there may be situations where such behavior is undesirable, and we may wish to minimize the unweighted cost (9) while employing asymmetric network topologies. This can be achieved by effectively rescaling the agent-specific step-sizes to compensate for the Perron weights p_k [44], [58], [59].

Federated learning

Federated learning has emerged in recent years as an umbrella term for architectures that involve a fusion center as well as high levels of asynchrony and heterogeneity. The federated setting can be viewed as a special case of the decentralized algorithms for an appropriately chosen network topologies and asynchrony models. As a result, many algorithms and performance guarantees for federated learning can be recovered from their decentralized counterparts by appropriately specializing the network topology. To illustrate this fact, let us consider the diffusion algorithm (13) and (14) with random adjacency matrix $\boldsymbol{A}$. The resulting behavior at any given agent corresponds precisely to a stochastic variant of the federated averaging algorithm, and the analysis of [26] applies. We may instead consider a deterministic

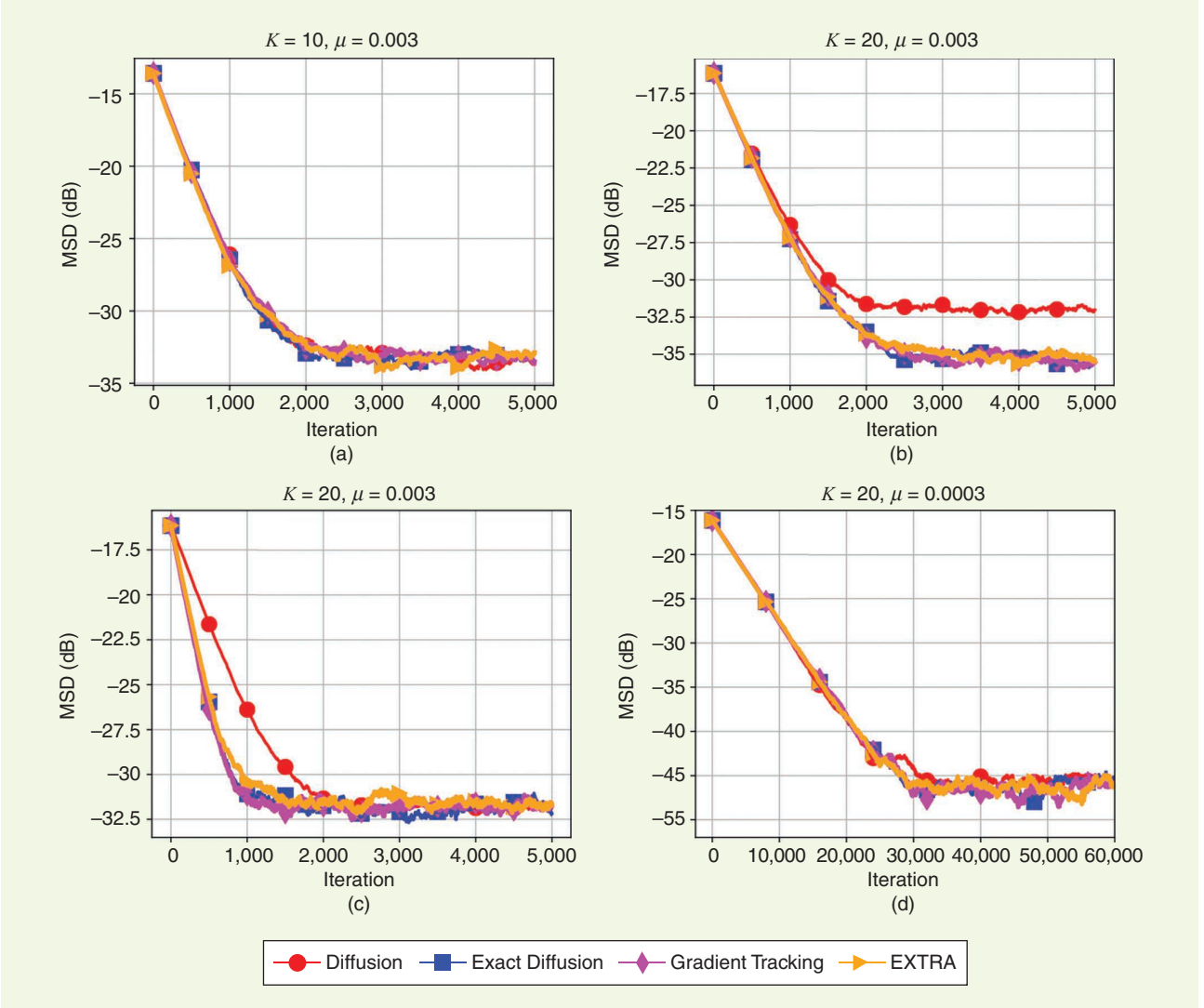


FIGURE 5. An illustration of the benefit of primal-dual algorithms (taken from [57]). Primal-dual algorithms (e.g., EXTRA, gradient-tracking, and exact diffusion) outperform a primal algorithm (diffusion) for a large network size (large K, i.e., λ closer to one) and large step-size. This is the range where $(\mu^2\lambda^2 b^2)/(1-\lambda)^2$ in (35) is nonnegligible.

variant with time-varying A_i, where $A_i = \mathbb{1}\mathbb{1}^\top$ if i is a multiple of $i_o \geq 1$, and $A_i = I$ otherwise. In this case, the arguments of [50] apply. This corresponds to a deterministic variant of federated averaging, where agents interlace multiple local updates with any round of communications. Of course, variants of these constructions are possible, and we refer the reader to [26] and [50] for details.

The federated setting can be viewed as a special case of the decentralized algorithms for an appropriately chosen network.

Conclusion

The ever-increasing need for processing signals and information available at dispersed locations has led to broad research efforts across a number of communities in the past two decades. In this article, we presented a unified view on algorithms for distributed inference and learning through the lens of stochastic primal and primal-dual optimization, and surveyed some common themes in performance, such as the impact of learning rate, network topology, and the benefit of cooperation. The key takeaway from these studies is that in most cases, distributed solutions with appropriately designed cooperation protocols are able to match the performance of centralized, fusion-center-based approaches, while offering scalability, robustness to node-and-link failure, communication efficiency, and no need for the exchange of raw data.

Acknowledgment

A longer version of this manuscript, with examples and illustrative applications, is available at https://arxiv.org/abs/2210.13767.

Authors

Stefan Vlaski (s.vlaski@imperial.ac.uk) received his Ph.D. degree in electrical and computer engineering from the University of California, Los Angeles, USA, in 2019. He is currently a lecturer with Imperial College, SW7 2AZ London, U.K. From 2019 to 2021, he was a postdoctoral researcher with the Adaptive Systems Laboratory, École Polytechnique Fédérale de Lausanne, Switzerland. His research interests include the intersection of machine learning, network science, and optimization. He is a Member of IEEE.

Soummya Kar (soummyak@andrew.cmu.edu) received his Ph.D. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, where he is currently a professor of electrical and computer engineering. From June 2010 to May 2011, he was with the Electrical Engineering Department, Princeton University, Princeton, NJ, as a postdoctoral research associate. His research interests include signal processing in large-scale networked systems, machine learning, and stochastic analysis, with applications in cyberphysical systems. He is a Fellow of IEEE.

Ali H. Sayed (ali.sayed@epfl.ch) is the dean of engineering at École Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland, where he also directs the Adaptive Systems Laboratory. He has served as a distinguished professor and chair of electrical engineering at the University of California, Los Angeles. He served as president of the IEEE Signal Processing Society in 2018 and 2019 and is currently a member of the IEEE Board of Directors. His work has been recognized with several awards, including the 2022 IEEE Fourier Technical Field Award and the 2020 IEEE Wiener Society Award. His research interests include adaptation and learning theories, data and network sciences, and statistical inference. He is a member of the U.S. National Academy of Engineering and The World Academy of Sciences.

José M.F. Moura (moura@ece.cmu.edu) is the Philip L. and Marsha Dowd University Professor at Carnegie Mellon University, Pittsburgh 15213, USA. His patented detector (co-inventor Alek Kavcic) is in more than 60% of computers sold in the last 18 years (4 billion). CMU settled with Marvell its infringement for US$750 million. He was the 2019 IEEE president and CEO. He holds honorary doctorate degrees from the University of Strathclyde and Universidade de Lisboa and has received the Great Cross and Order of Infante D. Henrique. He received the 2023 IEEE Kilby Signal Processing Medal. His research interests include signal processing and data science. He is a Fellow of IEEE, and a fellow of the American Association for the Advancement of Science and the National Academy of Inventors, and a member of the Portugal Academy of Sciences and the National Academy of Engineering.

References

[1] M. Rabbat and R. Nowak, "Distributed optimization in sensor networks," in *Proc. Int. Symp. Inf. Process. Sens. Netw.*, 2004, pp. 20–27, doi: 10.1145/984622.984626.

[2] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proc. IEEE*, vol. 95, no. 1, pp. 215–233, Jan. 2007, doi: 10.1109/JPROC.2006.887293.

[3] C. G. Lopes and A. H. Sayed, "Diffusion least-mean squares over adaptive networks: Formulation and performance analysis," *IEEE Trans. Signal Process.*, vol. 56, no. 7, pp. 3122–3136, Jul. 2008, doi: 10.1109/TSP.2008.917383.

[4] I. D. Schizas, A. Ribeiro, and G. B. Giannakis, "Consensus in ad hoc WSNs with noisy links—Part I: Distributed estimation of deterministic signals," *IEEE Trans. Signal Process.*, vol. 56, no. 1, pp. 350–364, Jan. 2008, doi: 10.1109/TSP.2007.906734.

[5] S. Kar and J. M. F. Moura, "Distributed consensus algorithms in sensor networks with imperfect communication: Link failures and channel noise," *IEEE Trans. Signal Process.*, vol. 57, no. 1, pp. 355–369, Jan. 2009, doi: 10.1109/TSP.2008.2007111.

[6] S. Barbarossa and G. Scutari, "Decentralized maximum-likelihood estimation for sensor networks composed of nonlinearly coupled dynamical systems," *IEEE Trans. Signal Process.*, vol. 55, no. 7, pp. 3456–3470, Jul. 2007, doi: 10.1109/TSP.2007.893921.

[7] 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, doi: 10.1109/JPROC.2010.2052531.

[8] 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, doi: 10.1109/TIT.2012.2191450.

[9] 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, doi: 10.1109/TSP.2009.2033729.

[10] D. Bajovic, D. Jakovetic, J. M. F. Moura, J. Xavier, and B. Sinopoli, "Large deviations performance of consensus+innovations distributed detection with non-Gaussian observations," *IEEE Trans. Signal Process.*, vol. 60, no. 11, pp. 5987–6002, Nov. 2012, doi: 10.1109/TSP.2012.2210885.

[11] V. Matta, P. Braca, S. Marano, and A. H. Sayed, "Diffusion-based adaptive distributed detection: Steady-state performance in the slow adaptation regime," *IEEE Trans. Inf. Theory*, vol. 62, no. 8, pp. 4710–4732, Aug. 2016, doi: 10.1109/TIT.2016.2580665.

[12] A. Nedić and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Trans. Autom. Control*, vol. 54, no. 1, pp. 48–61, Jan. 2009, doi: 10.1109/TAC.2008.2009515.

[13] 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, Jul. 2011, doi: 10.1561/2200000016.

[14] S. S. Ram, A. Nedić, and V. V. Veeravalli, "Distributed stochastic subgradient projection algorithms for convex optimization," *J. Optim. Theory Appl.*, vol. 147, no. 3, pp. 516–545, Jul. 2010, doi: 10.1007/s10957-010-9737-7.

[15] W. Shi, Q. Ling, G. Wu, and W. Yin, "EXTRA: An exact first-order algorithm for decentralized consensus optimization," *SIAM J. Optim.*, vol. 25, no. 2, pp. 944–966, 2015, doi: 10.1137/14096668X.

[16] Q. Ling, W. Shi, G. Wu, and A. Ribeiro, "DLM: Decentralized linearized alternating direction method of multipliers," *IEEE Trans. Signal Process.*, vol. 63, no. 15, pp. 4051–4064, Aug. 2015, doi: 10.1109/TSP.2015.2436358.

[17] D. Jakovetić, J. Xavier, and J. M. F. Moura, "Fast distributed gradient methods," *IEEE Trans. Autom. Control*, vol. 59, no. 5, pp. 1131–1146, May 2014, doi: 10.1109/TAC.2014.2298712.

[18] D. Jakovetić, D. Bajović, J. Xavier, and J. M. F. Moura, "Primal–dual methods for large-scale and distributed convex optimization and data analytics," *Proc. IEEE*, vol. 108, no. 11, pp. 1923–1938, Nov. 2020, doi: 10.1109/JPROC.2020.3007395.

[19] P. Di Lorenzo and G. Scutari, "NEXT: In-network nonconvex optimization," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 2, no. 2, pp. 120–136, Jun. 2016, doi: 10.1109/TSIPN.2016.2524588.

[20] A. H. Sayed, "Adaptive networks," *Proc. IEEE*, vol. 102, no. 4, pp. 460–497, Apr. 2014, doi: 10.1109/JPROC.2014.2306253.

[21] J. Chen and A. H. Sayed, "On the learning behavior of adaptive networks - Part I: Transient analysis," *IEEE Trans. Inf. Theory*, vol. 61, no. 6, pp. 3487–3517, Jun. 2015, doi: 10.1109/TIT.2015.2427360.

[22] A. H. Sayed, "Adaptation, learning, and optimization over networks," *Found. Trends Mach. Learn.*, vol. 7, nos. 4–5, pp. 311–801, Jul. 2014, doi: 10.1561/2200000051.

[23] R. Nassif, S. Vlaski, C. Richard, J. Chen, and A. H. Sayed, "Multitask learning over graphs: An approach for distributed, streaming machine learning," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 14–25, May 2020, doi: 10.1109/MSP.2020.2966273.

[24] S. Vlaski and A. H. Sayed, "Distributed learning in non-convex environments—Part II: Polynomial escape from saddle-points," *IEEE Trans. Signal Process.*, vol. 69, pp. 1257–1270, Jan. 2021, doi: 10.1109/TSP.2021.3050840.

[25] B. Swenson, R. Murray, H. V. Poor, and S. Kar, "Distributed gradient flow: Nonsmoothness, nonconvexity, and saddle point evasion," *IEEE Trans. Autom. Control*, vol. 67, no. 8, pp. 3949–3964, Aug. 2022, doi: 10.1109/TAC.2021.3111853.

[26] X. Zhao and A. H. Sayed, "Asynchronous adaptation and learning over networks—Part I: Modeling and stability analysis," *IEEE Trans. Signal Process.*, vol. 63, no. 4, pp. 811–826, Feb. 2015, doi: 10.1109/TSP.2014.2385046.

[27] M. H. DeGroot, "Reaching a consensus," *J. Amer. Statist. Assoc.*, vol. 69, no. 345, pp. 118–121, Apr. 2012, doi: 10.1080/01621459.1974.10480137.

[28] R. L. Berger, "A necessary and sufficient condition for reaching a consensus using DeGroot's method," *J. Amer. Statist. Assoc.*, vol. 76, no. 374, pp. 415–418, Mar. 2012, doi: 10.1080/01621459.1981.10477662.

[29] J. Tsitsiklis, D. Bertsekas, and M. Athans, "Distributed asynchronous deterministic and stochastic gradient optimization algorithms," *IEEE Trans. Autom. Control*, vol. 31, no. 9, pp. 803–812, Sep. 1986, doi: 10.1109/TAC.1986.1104412.

[30] 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, doi: 10.1109/TIT.2006.874516.

[31] S. Kar and J. M. F. Moura, "Convergence rate analysis of distributed gossip (Linear Parameter) estimation: Fundamental limits and tradeoffs," *IEEE J. Sel. Topics Signal Process.*, vol. 5, no. 4, pp. 674–690, Aug. 2011, doi: 10.1109/JSTSP.2011.2127446.

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

[33] 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, doi: 10.1109/TSP.2009.2036046.

[34] A. H. Sayed and C. G. Lopes, "Adaptive processing over distributed networks," *IEICE Trans. Fundam. Electron., Commun. Comput. Sci.*, vol. E90-A, no. 8, pp. 1504–1510, Aug. 2007, doi: 10.1093/ietfec/e90-a.8.1504.

[35] 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, doi: 10.1109/TSP.2007.913164.

[36] Z. J. Towfic, J. Chen, and A. H. Sayed, "Excess-risk of distributed stochastic learners," *IEEE Trans. Inf. Theory*, vol. 62, no. 10, pp. 5753–5785, Oct. 2016, doi: 10.1109/TIT.2016.2593769.

[37] K. Srivastava and A. Nedić, "Distributed asynchronous constrained stochastic optimization," *IEEE J. Sel. Topics Signal Process.*, vol. 5, no. 4, pp. 772–790, Aug. 2011, doi: 10.1109/JSTSP.2011.2118740.

[38] A. H. Sayed, *Inference and Learning from Data*. Cambridge, U.K.: Cambridge Univ. Press, 2022.

[39] 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, doi: 10.1109/TSP.2008.927480.

[40] F. S. 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, doi: 10.1109/TAC.2010.2042987.

[41] S. Kar and J. M. F. Moura, "Gossip and distributed Kalman filtering: Weak consensus under weak detectability," *IEEE Trans. Signal Process.*, vol. 59, no. 4, pp. 1766–1784, Apr. 2011, doi: 10.1109/TSP.2010.2100385.

[42] C. Xi and U. A. Khan, "Distributed subgradient projection algorithm over directed graphs," *IEEE Trans. Autom. Control*, vol. 62, no. 8, pp. 3986–3992, Aug. 2017, doi: 10.1109/TAC.2016.2615066.

[43] A. K. Sahu and S. Kar, "Decentralized zeroth-order constrained stochastic optimization algorithms: Frank–Wolfe and variants with applications to black-box adversarial attacks," *Proc. IEEE*, vol. 108, no. 11, pp. 1890–1905, Nov. 2020, doi: 10.1109/JPROC.2020.3012609.

[44] K. Yuan, B. Ying, X. Zhao, and A. H. Sayed, "Exact diffusion for distributed optimization and learning – Part II: Convergence analysis," *IEEE Trans. Signal Process.*, vol. 67, no. 3, pp. 724–739, Feb. 2019, doi: 10.1109/TSP.2018.2875883.

[45] Z. J. Towfic and A. H. Sayed, "Stability and performance limits of adaptive primal-dual networks," *IEEE Trans. Signal Process.*, vol. 63, no. 11, pp. 2888–2903, Jun. 2015, doi: 10.1109/TSP.2015.2415759.

[46] D. Jakovetić, J. M. F. Moura, and J. Xavier, "Linear convergence rate of a class of distributed augmented Lagrangian algorithms," *IEEE Trans. Autom. Control*, vol. 60, no. 4, pp. 922–936, Apr. 2015, doi: 10.1109/TAC.2014.2363299.

[47] A. Mokhtari and A. Ribeiro, "DSA: Decentralized double stochastic averaging gradient algorithm," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 2165–2199, Jan. 2016.

[48] H. Tang, X. Lian, M. Yan, C. Zhang, and J. Liu, "D^2: Decentralized training over decentralized data," in *Proc. Int. Conf. Mach. Learn.*, Jul. 2018, vol. 80, pp. 4848–4856.

[49] J. Xu, S. Zhu, Y. C. Soh, and L. Xie, "Augmented distributed gradient methods for multi-agent optimization under uncoordinated constant stepsizes," in *Proc. IEEE Conf. Decis. Contr.*, 2015, pp. 2055–2060, doi: 10.1109/CDC.2015.7402509.

[50] A. Nedić, A. Olshevsky, and W. Shi, "Achieving geometric convergence for distributed optimization over time-varying graphs," *SIAM J. Optim.*, vol. 27, no. 4, pp. 2597–2633, 2017, doi: 10.1137/16M1084316.

[51] W. Shi, Q. Ling, G. Wu, and W. Yin, "A proximal gradient algorithm for decentralized composite optimization," *IEEE Trans. Signal Process.*, vol. 63, no. 22, pp. 6013–6023, Nov. 2015, doi: 10.1109/TSP.2015.2461520.

[52] Z. J. Towfic and A. H. Sayed, "Adaptive penalty-based distributed stochastic convex optimization," *IEEE Trans. Signal Process.*, vol. 62, no. 15, pp. 3924–3938, Aug. 2014, doi: 10.1109/TSP.2014.2331615.

[53] A. K. Sahu, S. Kar, J. M. F. Moura, and H. V. Poor, "Distributed constrained recursive nonlinear least-squares estimation: Algorithms and asymptotics," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 2, no. 4, pp. 426–441, Dec. 2016, doi: 10.1109/TSIPN.2016.2618318.

[54] R. Nassif, S. Vlaski, C. Richard, and A. H. Sayed, "Learning over multitask graphs—Part I: Stability analysis," *IEEE Open J. Signal Process.*, vol. 1, pp. 28–45, Apr. 2020, doi: 10.1109/OJSP.2020.2989038.

[55] L. Xie, D.-H. Choi, S. Kar, and H. V. Poor, "Fully distributed state estimation for wide-area monitoring systems," *IEEE Trans. Smart Grid*, vol. 3, no. 3, pp. 1154–1169, Sep. 2012, doi: 10.1109/TSG.2012.2197764.

[56] S. Kar, J. M. F. Moura, and H. V. Poor, "Distributed linear parameter estimation: Asymptotically efficient adaptive strategies," *SIAM J. Contr. Optim.*, vol. 51, no. 3, pp. 2200–2229, 2013, doi: 10.1137/110848396.

[57] K. Yuan, S. A. Alghunaim, B. Ying, and A. H. Sayed, "On the influence of bias-correction on distributed stochastic optimization," *IEEE Trans. Signal Process.*, vol. 68, pp. 4352–4367, Jul. 2020, doi: 10.1109/TSP.2020.3008605.

[58] A. Nedić and A. Olshevsky, "Distributed optimization over time-varying directed graphs," *IEEE Trans. Autom. Control*, vol. 60, no. 3, pp. 601–615, Mar. 2015, doi: 10.1109/TAC.2014.2364096.

[59] F. Saadatniaki, R. Xin, and U. A. Khan, "Decentralized optimization over time-varying directed graphs with row and column-stochastic matrices," *IEEE Trans. Autom. Control*, vol. 65, no. 11, pp. 4769–4780, Nov. 2020, doi: 10.1109/TAC.2020.2969721.

[60] Z. Li, W. Shi, and M. Yan, "A decentralized proximal-gradient method with network independent step-sizes and separated convergence rates," *IEEE Trans. Signal Process.*, vol. 67, no. 17, pp. 4494–4506, 2019.

SP