Stochastic Optimization from Distributed, Streaming Data in Rate-limited Networks

Matthew Nokleby and Waheed U. Bajwa

Abstract—Motivated by machine learning applications in networks of sensors, internet-of-things (IoT) devices, and autonomous agents, we propose techniques for distributed stochastic convex learning from high-rate data streams. The setup involves a network of nodes—each one of which has a stream of data arriving at a constant rate—that solve a stochastic convex optimization problem by collaborating with each other over rate-limited communication links. To this end, we present and analyze two algorithms—termed distributed stochastic approximation mirror descent (D-SAMD) and accelerated distributed stochastic approximation mirror descent (AD-SAMD)—that are based on two stochastic variants of mirror descent. The main collaborative step in the proposed algorithms is approximate averaging of the local, noisy subgradients using distributed consensus. While distributed consensus is well suited for collaborative learning, its use in our setup results in perturbed subgradient averages due to rate-limited links, which may slow down or prevent convergence. Our main contributions in this regard are: (i) bounds on the convergence rates of D-SAMD and AD-SAMD in terms of the number of nodes, network topology, and ratio of the data streaming and communication rates, and (ii) sufficient conditions for order-optimum convergence of D-SAMD and AD-SAMD. In particular, we show that there exist regimes under which AD-SAMD, when compared to D-SAMD, achieves order-optimum convergence with slower communications rates. This is in contrast to the centralized setting in which use of accelerated mirror descent results in a modest improvement over regular mirror descent for stochastic composite optimization. Finally, we demonstrate the effectiveness of the proposed algorithms using numerical experiments.

I. INTRODUCTION

Machine learning at its core involves solving stochastic optimization (SO) problems of the form

$$\min_{x \in X} \psi(x) \triangleq \min_{x \in X} E_\xi[\phi(x, \xi)]$$

(1)

to learn a “model” $x \in X \subset \mathbb{R}^n$ that is then used for tasks such as dimensionality reduction, classification, clustering, regression, and/or prediction. A primary challenge of machine learning is to find a solution to the SO problem (1) without knowledge of the distribution $P(\xi)$. This involves finding an approximate solution to (1) using a sequence of $T$ samples $\{\xi(t) \in \mathcal{Y}\}_{t=1}^T$, termed training data, that are typically drawn independently from the distribution $P(\xi)$, which is supported on a subset of $\mathcal{Y}$. There are, in particular, two main categorizations of training data that, in turn, determine the types of methods that can be used to find approximate solutions to the SO problem. These are (i) batch training data and (ii) streaming training data.

In the case of batch training data, where all $T$ samples $\{\xi(t)\}$ are pre-stored and simultaneously available, sample average approximation (SAA) (also referred to as empirical risk minimization (ERM)) is commonly employed. In SAA/ERM, one minimizes the empirical average of the “risk” function $\phi(\cdot, \cdot)$ in lieu of the true expectation:

$$x^*_{\text{SAA}} = \arg \min_{x \in X} \frac{1}{T} \sum_{t=1}^T \phi(x, \xi(t))$$.

(2)

In contrast, the samples $\{\xi(t)\}$ in the case of streaming training data arrive one-by-one, cannot be stored in memory for long, and should be processed as soon as possible. In this setting, stochastic approximation (SA) methods—the most well known of which is stochastic gradient descent (SGD)—are preferred over SAA/ERM techniques. The main idea in SA is to update the current iterate using only the most recent data sample(s) and terminate on exhaustion of training samples. Both SAA and SA have a long history in the literature; see [2] for a historical survey of SA methods, [3] for a comparative review of SAA and SA techniques, and [4] for a recent survey of SO techniques.

Despite the success of classical SO methods for machine learning, there has been a renewed interest in development and analysis of new methods due to the advent of fast-streaming and distributed data. Indeed, rapid proliferation of sensing and wearable devices, emergence of the Internet-of-Things (IoT), storage of data across geographically-distributed data centers, etc., are increasingly leading to scenarios in which data samples both continuously arrive at high rates and are distributed across physical entities. The goal of this paper is to provide an approximate solution to the SO problem (1) in this setting of distributed, streaming data. In particular, we focus on geographically-distributed entities (nodes) that collaborate over rate-limited communication links (e.g., wireless links within an IoT infrastructure) and obtain independent streams of training data arriving at a constant rate. The relationship between the rate at which communication takes place between nodes and the rate at which streaming data arrive at individual nodes plays a critical role in this setting. If, for example, data samples arrive much faster than nodes can communicate among themselves, it is difficult for the nodes to exchange

Matthew Nokleby is with Wayne State University, Detroit, MI (email: matthew.nokleby@wayne.edu), and Waheed U. Bajwa is with Rutgers University, Piscataway, NJ (email: waheed.bajwa@rutgers.edu). The work of WUB was supported in part by the NSF under grant CCF-1453073 and by the ARO under grant W911NF-14-1-0295.

1It is important to note here that the distinction between SAA and SA methods is not always as sharp as this discussion might suggest. Indeed, SGD is used routinely for SAA in deep learning not because training data are streaming, but to reduce computation and memory burdens (see, e.g. [1] Chapter 8.1.3).
enough information to enable an SA iteration on existing data in the network before new data arrives, thereby overwhelming the network. In order to address the challenge of distributed SO in the presence of a mismatch between the communications and streaming rates, we propose and analyze two distributed SA techniques, each based on distributed averaging consensus and stochastic mirror descent. In particular, we present bounds on the convergence rates of these techniques and derive conditions—involving the number of nodes, network topology, the streaming rate, and the communications rate—under which our solutions achieve order-optimum convergence speed.

A. Relationship to Prior Work

The idea of solving SO problems using SA methods dates back to the seminal work of Robbins and Monro [5]. Until recently, it was commonly believed that SAA/ERM methods converge faster than SA methods. This changed with [6, 7], which established that SA techniques can in fact outperform SAA/ERM methods for stochastic convex learning. Lan [8] improved on these results further using an accelerated stochastic mirror descent algorithm for stochastic composite optimization. This method, which makes use of noisy subgradients of $\psi(\cdot)$ computed using incoming training samples, satisfies

$$E[\psi(x(T)) - \psi(x^\ast)] \leq O(1) \left[ \frac{L}{T^2} + \frac{M + \sigma}{\sqrt{T}} \right], \quad (3)$$

where $\sigma^2$ denotes variance of the subgradient noise, and $M$ and $L$ denote the Lipschitz constants associated with the non-smooth (convex) component of $\psi$ and the gradient of the smooth (convex) component of $\psi$, respectively. In the case of general stochastic composite optimization, the rate in (3) is in fact the best possible rate for any iterative algorithm. Further assumptions such as smoothness and strong convexity of $\psi(\cdot)$ and/or presence of a structured regularizer term in $\psi(\cdot)$ can remove the dependence of the convergence rate on $M$ and/or improve the convergence rate to $O(\sigma/T)$ [6, 9–11].

The problem of distributed SO goes as far back as the seminal work of Tsitsiklis et al. [12], which presents distributed first-order methods for SO and gives proofs of their asymptotic convergence. Myriad works since then have applied these ideas to other settings, each with different assumptions about the type of data, how the data are distributed across the network, and how distributed units process data and share information among themselves. In order to put our work in context, we review a representative sample of these works.

A setting closely related to [12] is distributed optimization under the shared-memory model, in which distributed processors access optimization variables stored at a central, shared location. Synchronization of reads/writes to memory is a critical issue in this case; recent SO methods such as HogWild [13] and KromMagnon [14] offer convergence guarantees under asynchronous and/or sparse access to memory.

A line of work more closely related to this paper involves incremental (sub)gradient methods, in which a single node takes an optimization step based on its local data and passes that solution to a neighbor, which takes another optimization step based on its data, and so on. The message passing in incremental methods may suppose a ring network topology or use a randomized message-passing rule over an arbitrary topology. Several incremental distributed SO techniques that are based on either SAA/ERM [15–19] or SA [20] have been published. But, since only one node at a time processes data, these methods are unsuitable for SO in rate-limited networks with fast streaming data.

Another related line of work is distributed optimization of linearly separable objective functions. The objective function in this case is the sum of constituent functions, each one of which is available at a single node. This setup is equivalent to SAA/ERM, where each node’s objective function is the empirical risk evaluated over local data. Solutions to this problem minimize the global objective function via iterate, dual, or (sub)gradient averaging [21–25] or via alternating direction method of multipliers or augmented Lagrangian methods [26, 27], often using gossip algorithms or average consensus [28] to average variables.

A final line of work directly addresses distributed SA. In [29], nodes engage in distributed SA using projected stochastic subgradient descent and distributed consensus that averages nodes’ iterates. Similarly, in [30], nodes perform distributed stochastic dual averaging using consensus averaging of the dual variables. Both these works require that nodes engage in a single round of communication (consensus) for each stochastic subgradient sample. Stated differently, these works assume the communications rate is identical to the data streaming rate, which allows the nodes to carry out a consensus round before the next set of subgradient samples arrive in the network. In contrast, this paper allows the communications and streaming rates to be different from each other; in particular, our solutions enable distributed SA even when the communications rate is slower than the streaming rate.

We conclude by discussing two works [31], [32] that are most closely related to this work. In [31], nodes perform distributed SA by forming distributed mini-batch averages of stochastic gradients and using stochastic dual averaging. The main assumption in this work is that nodes can compute exact stochastic gradient averages (e.g., via AllReduce in parallel computing architectures). Under this assumption, it is shown in this work that there is an appropriate mini-batch size for which the nodes’ iterates converge at the optimum (centralized) rate. However, the need for exact averages in this work is not suited to rate-limited (e.g., wireless) networks, in which mimicking the AllReduce functionality can be costly and challenging.

The need for exact stochastic gradient averages in [31] has been relaxed recently in [32], in which nodes carry out distributed stochastic dual averaging by computing approximate mini-batch averages of dual variables via distributed consensus. In addition, and similar to our work, [32] allows for a mismatch between the communications rate and the data streaming rate. Nonetheless, there are four main distinctions between [32] and our work. First, we provide results for stochastic composite optimization, in which the objective function has a non-smooth component, whereas [31], [32] consider a differentiable objective. Second, we study distributed mirror descent, which allows for a limited generalization to non-Euclidean settings, whereas [32] (and [31]) consider only...
Euclidean settings. Third, we explicitly examine the impact of slow communications rate on performance, in particular highlighting the need for large mini-batches and their impact on convergence speed when the communications rate is slow. In [32], on the other hand, the optimum mini-batch size is first derived from [31], after which the communications rate needed to facilitate distributed consensus at the optimum mini-batch size is specified. While it appears to be possible to derive some of our results from a re-framing of the results of [32], it is crucial to highlight the trade-offs necessary under slow communications, which is not done in prior works. Finally, this work presents a distributed accelerated mirror descent approach to distributed SA, whereas [32] (and [31]) consider only standard/“non-accelerated” methods. A somewhat surprising outcome of this is that acceleration substantially improves the convergence rate in rate-limited networks.

B. Our Contributions

In this paper, we present two strategies for distributed SA over networks with fast streaming data and slow communications links: distributed stochastic approximation mirror descent (D-SAMD) and accelerated distributed stochastic approximation mirror descent (AD-SAMD). In both cases, nodes first locally compute mini-batch stochastic subgradient averages to accommodate a fast streaming rate (or, equivalently, a slow communications rate), and then they collaboratively compute approximate network subgradient averages via distributed consensus. Finally, nodes individually employ mirror descent and accelerated mirror descent, respectively, on the approximate averaged subgradients for the next set of iterates.

Our main theoretical contribution is the derivation of bounds on the convergence rates of D-SAMD and AD-SAMD. These bounds involve a careful analysis of the impact of imperfect subgradient averaging on individual nodes’ iterates. In addition, we derive sufficient conditions for order-optimum convergence of D-SAMD and AD-SAMD in terms of the streaming and communications rates, the size and topology of the network, and the data statistics.

A key finding of this paper is that AD-SAMD achieves order-optimum convergence in a larger regime than D-SAMD, thus accommodating slower communications links relative to the streaming rate. By contrast, the convergence speeds of centralized stochastic mirror descent and accelerated stochastic mirror descent differ only in higher order terms.

C. Notation and Organization

We typically use boldfaced lowercase and boldfaced capital letters (e.g., $x$ and $W$) to denote (possibly random) vectors and matrices, respectively. Unless otherwise specified, all vectors are assumed to be column vectors. The transpose operation is denoted by $(\cdot)^T$, while $1$ denotes the vector of all ones. Further, we use $E[\cdot]$ to denote the expectation operation and $\mathbb{R}$ to denote the field of real numbers. The gradient operation is denoted by $\nabla$, while $\otimes$ denotes the Hadamard product. Finally, given two functions $p(r)$ and $q(r)$, we write $p(r) = O(q(r))$ if there exists a constant $C$ such that $\forall r, p(r) \leq Cq(r)$, and we write $p(r) = \Omega(q(r))$ if $q(r) = O(p(r))$.

The rest of this paper is organized as follows. In Section II we formalize the problem of distributed stochastic composite optimization. In Sections III and IV we describe D-SAMD and AD-SAMD, respectively, and also derive performance guarantees for these two methods. We demonstrate the effectiveness of D-SAMD and AD-SAMD via numerical experiments in Section V and we conclude the paper in Section VI. Proofs are provided in the Appendix.

II. Problem Formulation

The objective of this paper is order-optimal, distributed minimization of the composite function

$$
\psi(x) = f(x) + h(x),
$$

where $x \in X \subset \mathbb{R}^n$ and $X$ is convex and compact. The space $\mathbb{R}^n$ is endowed with an inner product $\langle \cdot, \cdot \rangle$ that need not be the usual one and a norm $\|\cdot\|$ that need not be the one induced by the inner product. In the following, the minimizer and the minimum value of $\psi$ are denoted as:

$$
x^* \triangleq \arg \min_{x \in X} \psi(x), \quad \psi^* \triangleq \psi(x^*).
$$

We now make a few assumptions on the smooth ($f(\cdot)$) and non-smooth ($h(\cdot)$) components of $\psi$. The function $f : X \rightarrow \mathbb{R}$ is convex with Lipschitz continuous gradients, i.e.,

$$
\|\nabla f(x) - \nabla f(y)\|_s \leq L \|x - y\|, \quad \forall x, y \in X,
$$

where $\|\cdot\|_s$ is the dual norm associated with $\langle \cdot, \cdot \rangle$ and $\|\cdot\|$:

$$
\|g\|_s \triangleq \sup_{\|x\| \leq 1} \langle g, x \rangle.
$$

The function $h : X \rightarrow \mathbb{R}$ is convex and Lipschitz continuous:

$$
\|h(x) - h(y)\| \leq M \|x - y\|, \quad \forall x, y \in X.
$$

Note that $h$ need not have gradients; however, since it is convex we can consider its subdifferential, denoted by $\partial h(y)$:

$$
\partial h(y) = \{g : h(z) \geq g^T(z - y), \forall z \in X\}.
$$

We further impose a notion of Lipschitz continuity on the subdifferential of $h$ and suppose that, for any $x, y \in X$, there exist $g_x \in \partial h(x)$ and $g_y \in \partial h(y)$ such that

$$
\|g_x - g_y\|_s \leq L \|x - y\|.
$$

Equivalently, there exists a function that chooses a unique element from the subdifferential $\partial h(x)$ that is Lipschitz in $x$. For example, the non-smooth function $h(x) = |x|$ has this property. This condition is usually not needed in standard SA problems, but we need it in the distributed SA setting in order to bound the accumulation of subgradient estimation error.

A. Distributed Stochastic Composite Optimization

Our focus in this paper is on minimizing $\psi(x)$ over a network of $m$ nodes, represented by the undirected graph $G = (V, E)$. To this end, we suppose that nodes minimize $\psi$ collaboratively by exchanging subgradient information with their neighbors at each communications round. Specifically,
each node $i \in V$ transmits a message at each communications round to each of its neighbors, defined as
\[ \mathcal{N}_i = \{ j \in V : (i, j) \in E \}, \]  
where we suppose that a node is in its own neighborhood, i.e. $i \in \mathcal{N}_i$. We assume that this message passing between nodes takes place without any error or distortion. Further, we constrain the messages between nodes to be members of the dual space of $X$ and to satisfy causality; i.e., messages transmitted by a node can depend only on its local data and previous messages received from its neighbors.

Next, in terms of data generation, we suppose that each node $i \in V$ queries a first-order stochastic “oracle” at a fixed rate—which may be different from the rate of message exchange—to obtain noisy estimates of the subgradient of $\psi$ at different query points in $X$. Formally, we use ‘$t$’ to index time according to data-acquisition rounds and define $\{\xi_i(t) \in \mathcal{T} \}_{t \geq 1}$ to be a sequence of independent (with respect to $i$ and $t$) randomly distributed (i.i.d) random variables with unknown probability distribution $P(\xi)$. At each data-acquisition round $t$, node $i$ queries the oracle at search point $x_i(s)$ to obtain the point $G(x_i(s), \xi_i(t))$ that is a noisy version of the subgradient of $\psi$ at $x_i(s)$. Here, we use ‘$s$’ to index time according to search-point update rounds, with possibly multiple data-acquisition rounds per search-point update. The reason for allowing the search-point update index $s$ to be different from the data-acquisition index $t$ is to accommodate the setting in which data (equivalently, subgradient estimates) arrive at a much faster rate than the rate at which nodes can communicate with each other; we will elaborate further on this in the next subsection.

Formally, $G(x, \xi)$ is a Borel function that satisfies the following properties:
\[ E[G(x, \xi)] \triangleq g(x) \in \partial \psi(x), \quad \text{and} \quad E[\|G(x, \xi) - g(x)\|^2_2] \leq \sigma^2, \]
where the expectation is with respect to the distribution $P(\xi)$. We emphasize that this formulation is equivalent to that in which the objective function is $\psi(x) \triangleq E[\phi(x, \xi)]$, and where nodes in the network acquire data point $\{\xi_i(t)\}_{i \in V}$ at each data-acquisition round $t$ that are then used to compute the subgradients of $\phi(x, \xi_i(t))$, which—in turn—are noisy subgradients of $\psi(x)$.

B. Mini-batching for Rate-Limited Networks

A common technique to reduce the variance of the (sub)gradient noise and/or reduce the computational burden in centralized SO is to average “batches” of oracle outputs into a single (sub)gradient estimate. This technique, which is referred to as mini-batching, is also used in this paper; however, its purpose in our distributed setting is to both reduce the subgradient noise variance and manage the potential mismatch between the communications rate and the data streaming rate. Before delving into the details of our mini-batch strategy, we present a simple model to parametrize the mismatch between the two rates. Specifically, let $\rho > 0$ be the fixed ratio between the rate of communications and the rate of data acquisition.

That is, $\rho \geq 1$ implies nodes engage in $\rho$ rounds of message exchanges for every data-acquisition round. Similarly, $\rho < 1$ means there is one communications round for every $1/\rho$ data-acquisition rounds. We ignore rounding issues for simplicity.

The mini-batching in our distributed problem proceeds as follows. Each mini-batch round spans $b \geq 1$ data-acquisition rounds and coincides with the search-point update round, i.e., each node $i$ updates its search point at the end of a mini-batch round. In each mini-batch round $s$, each node $i$ uses its current search point $x_i(s)$ to compute an average of oracle outputs
\[ \theta_i(s) = \frac{1}{b} \sum_{t=(s-1)b+1}^{sb} G(x_i(s), \xi_i(t)). \]  
This is followed by each node computing a new search point $x_i(s+1)$ using $\theta_i(s)$ and messages received from its neighbors.

In order to analyze the mini-batching distributed SA techniques proposed in this work, we will rely on a generalization of the usual averaging property of variances to non-Euclidean norms.

**Lemma 1:** Let $z_1, \ldots, z_n$ be i.i.d. random vectors in $\mathbb{R}^n$ with $E[|z|] = 0$ and $E[\|z\|_2^2] \leq \sigma^2$. There exists a constant $C_\ast \geq 0$, which depends only on $\|\cdot\|$ and $\langle \cdot, \cdot \rangle$, such that
\[ E \left[ \left\| \frac{1}{k} \sum_{i=1}^{k} z_i \right\|_2^2 \right] \leq \frac{C_\ast \sigma^2}{k}. \]

**Proof:** This follows directly from the property of norm equivalence in finite-dimensional spaces.

In order to illustrate Lemma 1, notice that when $\|\cdot\| = \|\cdot\|_1$, i.e., the $l_1$ norm, and $\langle \cdot, \cdot \rangle$ is the standard inner product, the associated dual norm is the $l_\infty$ norm: $\|\cdot\|_\infty = \|\cdot\|_\infty$. Since $\|x\|_\infty^2 \leq \|x\|_2^2 \leq n \|x\|_\infty^2$, we have $C_\ast = n$ in this case. Thus, depending on the norm in use, the extent to which averaging reduces subgradient noise variance may depend on the dimension of the optimization space.

Let $z_i(s) \triangleq \theta_i(s) - g(x_i(s))$. Then, $E[\|z_i(s)\|^2_2] \leq C_\ast \sigma^2/b$. We emphasize that the subgradient noise vectors $z_i(s)$ depend on the search points $x_i(s)$; we suppress this notation for brevity.

C. Problem Statement

Mini-batching induces a performance trade-off: Averaging reduces subgradient noise and processing time, but it also reduces the rate of search-point updates. This trade-off depends on the relationships between the data and the communications rates. In order to carry out distributed SA in an order-optimal manner, we will require that the nodes collaborate by carrying out $r \geq 1$ rounds of averaging consensus on their mini-batch averages $\theta_i(s)$ in each mini-batch round $s$ (see Section III for details). In order to complete the $r$ communication rounds in time for the next mini-batch round, we have the constraint
\[ r \leq b \rho. \]  
If communications is faster, or if the mini-batch rounds are longer, nodes can fit in more rounds of information exchange between each mini-batch round or, equivalently, between each search-point update. But when the mismatch factor $\rho$ is small,
III. DISTRIBUTED STOCHASTIC APPROXIMATION MIRROR DESCENT

In this section we present our first distributed SA algorithm, called distributed stochastic approximation mirror descent (D-SAMD). This algorithm is based upon stochastic approximated mirror descent, which is a generalized version of stochastic subgradient descent. Before presenting D-SAMD, we review a few concepts that underlie mirror descent.

A. Stochastic Mirror Descent Preliminaries

Stochastic mirror descent, presented in [8], is a generalization of stochastic subgradient descent. This generalization is characterized by a distance-generating function \( \omega : X \to \mathbb{R} \) that generalizes the Euclidean norm. The distance-generating function must be continuously differentiable and strongly convex with modulus \( \alpha \), i.e.

\[
\langle \nabla \omega(x) - \nabla \omega(y), x - y \rangle \geq \alpha \|x - y\|^2, \forall x, y \in X. \tag{17}
\]

In the convergence analysis, we will require two measures of the “radius” of \( X \) that will arise in the convergence analysis, defined as follows:

\[
D_\omega \triangleq \sqrt{\max_{x \in X} \omega(x) - \min_{x \in X} \omega(x)}, \quad \Omega_\omega \triangleq \frac{2}{\alpha} D_\omega.
\]

The distance-generating function induces the \textit{prox function}, or the Bregman divergence \( V : X \times X \to \mathbb{R}_+ \), which generalizes the Euclidean distance:

\[
V(x, z) = \omega(z) - \omega(x) + \langle \nabla \omega(x), z - x \rangle. \tag{18}
\]

The prox function \( V(\cdot, \cdot) \) inherits strong convexity from \( \omega(\cdot) \), but it need not be symmetric or satisfy the triangle inequality. We define the \textit{prox mapping} \( P_\omega : \mathbb{R}^n \to X \) as

\[
P_\omega(y) = \arg \min_{x \in X} \omega(y, z - x) + V(x, z). \tag{19}
\]

The prox mapping generalizes the usual subgradient descent step, in which one minimizes the local linearization of the objective function regularized by the Euclidean distance of the step taken. In (centralized) stochastic mirror descent, one computes iterates of the form

\[
x(s + 1) = P_\omega(s)(\gamma_s g(s)),
\]

where \( g(s) \) is a stochastic subgradient of \( \psi(x(s)) \), and \( \gamma_s \) is a stepsize. These iterates have the same form as (stochastic) subgradient descent; indeed, choosing \( \omega(x) = \frac{1}{2} \|x\|^2_2 \) as well as \( \langle \cdot, \cdot \rangle \) and \( \|\cdot\| \) to be the usual ones results in subgradient descent iterations.

One can speed up convergence by choosing \( \omega(\cdot) \) to match the structure of \( X \) and \( \psi \). For example, if the optimization space \( X \) is the unit simplex over \( \mathbb{R}^n \), one can choose \( \omega(x) = \sum_i x_i \log(x_i) \) and \( \|\cdot\| \) to be the \( l_1 \) norm. This leads to \( V(x, z) = D(z|x) \), where \( D(\cdot|\cdot) \) denotes the Kullback-Leibler divergence between \( z \) and \( x \). This choice speeds up convergence on the order of \( O\left(\sqrt{n/\log(n)}\right) \) over using the Euclidean norm throughout. Along a similar vein, when \( \psi \) includes an \( l_1 \) regularizer to promote a sparse minimizer, one can speed up convergence by choosing \( \omega(\cdot) \) to be a \( p \)-norm with \( p = \log(n)/(\log(n) - 1) \).

In order to guarantee convergence for D-SAMD, we need to further restrict the distance-generating function \( \omega(\cdot) \). In particular, we require that the resulting prox mapping be 1-Lipschitz continuous in \( x, y \) pairs, i.e.

\[
\|P_\omega(x) - P_\omega(y)\| \leq \|x - x'\| + \|y - y'\|, \forall x, x', y, y' \in \mathbb{R}^n.
\]

This condition holds whenever the prox mapping is the projection of a 1-Lipschitz function of \( x, y \) onto \( X \). For example, it is easy to verify that this condition holds in the Euclidean setting. However, some important Bregman divergences, including the KL-divergence, result in prox mappings that are not Lipschitz. Consequently, while we present our results in terms of general prox functions, we emphasize that the results do not apply in all cases. One can think of the results primarily in the setting of Euclidean (accelerated) stochastic subgradient descent—for which case they are guaranteed to hold—with the understanding that one can check on a case-by-case basis to see if they hold for a particular non-Euclidean setting.

B. Description of D-SAMD

Here we present D-SAMD in detail, which generalizes stochastic mirror descent to the setting of distributed, streaming data. Nodes in this case carry out iterations similar to stochastic mirror descent as presented in [8], but instead of using local stochastic subgradients associated with the local search points, they carry out approximate consensus to estimate the average of stochastic subgradients across the network. This reduces the subgradient noise at each node and speeds up convergence.

Let \( W \) be a symmetric, doubly-stochastic matrix consistent with the network graph \( G \), i.e., \([W]_{ij} = 0 \) if \((i, j) \notin E\). Also set the constant step size \( 0 < \gamma \leq \alpha/(2L) \)^2 For simplicity, we suppose that there is a predetermined number of data-acquisition rounds \( T \), which leads to \( S = T/b \) mini-batch rounds. We detail the steps of D-SAMD in Algorithm [1].

In D-SAMD, each node \( i \) initializes its iterate at the minimizer of \( \omega(\cdot) \), which is guaranteed to be unique due to strong convexity. At each mini-batch round \( s \), each node \( i \) obtains its mini-batched subgradient and nodes engage in \( r \) rounds of averaging consensus to produce the (approximate) average subgradients \( h_i^s(s) \). Then, each node \( i \) takes a mirror prox step, using \( h_i^s(s) \) instead of its own mini-batched estimate. Finally, each node keeps a running average of its iterates, which is well-known to speed up convergence [33].

\[\text{We note further that it is possible to relax the constraint that the best Lipschitz constant be no larger than unity. This worsens the scaling laws—}
\]

\[\text{in particular, the required communications rate grows in } T \text{ rather than decreases—and we omit this case for brevity’s sake.}\]

\[\text{It is shown in [6] that a constant step size is sufficient for order-optimal performance, so we adopt such a rule here.}\]
Algorithm 1 Distributed stochastic approximation mirror descent (D-SAMD)

Require: Doubly-stochastic matrix \(W\), step size \(\gamma\), number of consensus rounds \(r\), batch size \(b\), and stream of mini-batched subgradients \(\theta_i(s)\).

1: for \(i = 1 : m\) do
2: \(x_i(1) \leftarrow \min_{x \in X} \omega(x)\) \(\triangleright\) Initialize search points
3: end for
4: for \(s = 1 : S\) do
5: \(h^0_i(s) \leftarrow \theta_i(s)\) \(\triangleright\) Get mini-batched subgradients
6: for \(q = 1 : r, i = 1 : m\) do
7: \(h^q_i(s) \leftarrow \sum_{j \in N_i} w_{ij} h^{q-1}_j\) \(\triangleright\) Consensus rounds
8: end for
9: for \(i = 1 : m\) do
10: \(x_i(s+1) \leftarrow P_{s, \hat{x}_i(s)} (\gamma h^s_i(s)) \) \(\triangleright\) Prox mapping
11: \(x^w_i(s+1) \leftarrow \frac{1}{b} \sum_{k=1}^b x_i(k)\) \(\triangleright\) Average iterates
12: end for
13: end for
return \(x^w_i(S+1)\)

C. Convergence Analysis

The convergence rate of D-SAMD depends on the bias and variance of the approximate subgradient averages \(h^q_i(s)\). In principle, averaging subgradients together reduces the noise variance and speeds up convergence. However, because averaging consensus using only \(r\) communications rounds results in approximate averages, each node takes a slightly different mirror prox step and therefore ends up with a different iterate. At each mini-batch round \(s\), nodes then compute subgradients at different search points, leading to bias in the averages \(h^q_i(s)\). This bias accumulates at a rate that depends on the subgradient noise variance, the topology of the network, and the number of consensus rounds per mini-batch round.

Therefore, the first step in bounding the convergence speed of D-SAMD is to bound the bias and the variance of the subgradient estimates \(h^q_i(s)\), which we do in the following Lemma.

Lemma 2: Let \(0 \leq \lambda_2 < 1\) denote the second-largest eigenvalue of \(W\). Define the matrices

\[ H(s) \triangleq [h^1_i(s), \ldots, h^m_i(s)] \]
\[ G(s) \triangleq [g(x_1(s)), \ldots, g(x_m(s))] \]
\[ Z(s) \triangleq [z_1(s), \ldots, z_m(s)] \]

recalling that the subgradient noise \(z_i(s)\) is defined with respect to the mini-batched subgradient \(\theta_i(s)\). Also define

\[ G(s) \triangleq \frac{1}{m} \sum_{i=1}^m g(x_i(s)), \quad G(s) \triangleq [g(s), \ldots, g(s)] \]
\[ Z_i(s) \triangleq \frac{1}{m} \sum_{i=1}^m z_i(s), \quad Z_i(s) \triangleq [z_i(s), \ldots, z_i(s)] \]

where the matrices \(G(s), Z(s) \in \mathbb{R}^{n \times n}\) have identical columns. Finally, define the matrices

\[ E(s) \triangleq G(s) W^r - G(s) \]
\[ Z(s) \triangleq Z(s) W^r - Z(s) \]

of average consensus error on the subgradients and subgradient noise, respectively. Then, the following facts are true. First, one can write \(H(s)\) as

\[ H(s) = \mathcal{C}(s) + E(s) + Z(s) + \hat{Z}(s). \]

Second, the columns of \(\hat{Z}(s)\) satisfy

\[ E[||\hat{Z}_i(s)||^2] \leq \frac{C_b\sigma^2}{mb}. \]

Finally, the \(i\)th columns of \(E(s)\) and \(Z(s)\), denoted by \(e_i(s)\) and \(z_i(s)\), respectively, satisfy

\[ ||e_i(s)||_s \leq \max_{j,k} m^2 \sqrt{C_b \lambda_2^*} ||g_j(s) - g_k(s)||_s \]

and

\[ E[||z_i(s)||^2] \leq \frac{\lambda^2 r^2 \sigma^2}{b}. \]

The next step in the convergence analysis is to bound the distance between iterates at different nodes. As long as iterates are not too far apart, the subgradients computed at different nodes have sufficiently similar means that averaging them together reduces the overall subgradient noise.

Lemma 3: Let \(a_s \triangleq \max_{i,j} ||x_i(s) - x_j(s)||\). The moments of \(a_s\) follow:

\[ E[a_s] \leq \frac{\alpha^m}{L m^{3/2} \sqrt{b}} ((1 + \alpha m^2 \sqrt{C_b \lambda_2^*})^s - 1), \]

\[ E[a_s^2] \leq \frac{\sigma^2}{L m^2} ((1 + \alpha m^2 \sqrt{C_b \lambda_2^*})^s - 1)^2. \]

Now, we can bound the expected gap to optimality.

Theorem 1: For D-SAMD, the expected gap to optimality at each node \(i\) satisfies

\[ E[\psi(x^w_i(S+1))] - \psi^* \leq \frac{2L\Omega^2}{\alpha S} + \sqrt{\frac{2(4M^2 + 2\Delta_2^2)}{\alpha S}} \sqrt{\frac{\alpha \Xi_S D_{\omega}}{2}}. \]

where

\[ \Xi_s \triangleq \frac{\sigma(1 + m^2 \sqrt{C_b \lambda_2^*})}{m^3/2 \sqrt{b}} ((1 + \alpha m^2 \sqrt{C_b \lambda_2^*})^s - 1) \]

and

\[ \Delta_2^2 \triangleq \frac{2\sigma^2 (1 + m^4 C_b \lambda_2^*)}{m^2b} ((1 + \alpha m^2 \sqrt{C_b \lambda_2^*})^s - 1)^2 + 4C_b \sigma^2 / (mb) + 4\lambda_b^2 C_b \sigma^2 m^2 / b \]

quantify the moments of the effective subgradient noise.

The convergence rate proved in Theorem 1 is akin to that provided in [8], with \(\Delta_2^2\) taking the role of the subgradient noise variance. A crucial difference is the presence of the final term involving \(\Xi_s\). In [8], this term vanishes because the intrinsic subgradient noise has zero mean. However, the equivalent gradient error in D-SAMD does not have zero mean in general. As nodes’ iterates diverge, their subgradients differ, and the nonlinear mapping between iterates and subgradients results in noise with nonzero mean.

The critical question is how fast communications needs to be for order-optimum convergence speed, i.e., the convergence speed that one would obtain if nodes had access to other
nodes’ subgradient estimates at each round. After \( S \) mini-batch rounds, the network has processed \( mT \) data samples. Centralized mirror descent, with access to all \( mT \) data samples in sequence, achieves the convergence rate [8]

\[
O(1) \left[ \frac{L}{mT} + \frac{\mathcal{M} + \sigma}{\sqrt{mT}} \right].
\]

The final term dominates the error as a function of \( m \) and \( T \), so in the following corollary we derive conditions under which the convergence rate of D-SAMD matches this term.

Corollary 1: The optimality gap for D-SAMD satisfies

\[
E[\psi(x_{1}^{\text{av}}(t + 1))] - \psi^* = O \left( \frac{\mathcal{M} + \sigma}{\sqrt{mT}} \right),
\]

provided the minibatch size \( b \), the communications rate \( \rho \), the number of users \( m \), and the Lipschitz constant \( \mathcal{M} \) satisfy

\[
b = \Omega \left( \frac{1 + \log(mT)}{\rho \log(1/\lambda_2)} \right), \quad \rho = \Omega \left( \frac{m^{1/2} \log(mT)}{\sigma T^{1/2} \log(1/\lambda_2)} \right)
\]

\[
m = O(\sigma^2 T), \quad \mathcal{M} = O \left( \min \left\{ \frac{1}{m}, \frac{1}{\sqrt{m^2 T}} \right\} \right).
\]

Remark 1: Corollary 1 gives new insights into the interdependence between rate, network topology, and mini-batch size. In [31], a mini-batch size of \( b = O(T^{1/2}) \) is prescribed—which is sufficient whenever gradient averages are perfect—and in [32] the number of imperfect consensus rounds needed to facilitate the mini-batch size \( b \) is derived. By contrast, we derive a mini-batch condition sufficient to drive the effective noise variance to \( O(\sigma^2/(mT)) \) while taking into consideration the impact of imperfect subgradient averaging. This condition depends not only on \( T \) but also on \( m, \rho \), and \( \sigma^2 \)—indeed, for all else constant, the optimum mini-batch size is merely \( \Omega(\log(T)) \). Then, the condition on \( \rho \) essentially ensures that \( b = O(T^{1/2}) \) as specified in [31].

Remark 2: Corollary 1 imposes a strict requirement on \( \mathcal{M} \), the Lipschitz constant of the non-smooth part of \( \psi \). Essentially the non-smooth part must vanish as \( m, T \), or \( \sigma^2 \) becomes large. This is because the contribution of \( h(x) \) to the convergence rate depends only on the number of iterations taken, not on the noise variance. Reducing the effective subgradient noise via mini-batching has no impact on this contribution, so we require the Lipschitz constant \( \mathcal{M} \) to be small to compensate.

IV. ACCELERATED DISTRIBUTED STOCHASTIC APPROXIMATION MIRROR DESCENT

In this section, we present accelerated distributed stochastic approximation mirror descent (AD-SAMD), which distributes the accelerated stochastic approximation mirror descent proposed in [8]. The centralized version of accelerated mirror descent achieves the optimum convergence rate of

\[
O(1) \left[ \frac{L}{T^2} + \frac{\mathcal{M} + \sigma^2}{\sqrt{T}} \right].
\]

Consequently, we will see that the convergence rate of AD-SAMD has \( 1/S^2 \) as its first term. This faster convergence in \( S \) allows for more aggressive mini-batching, and the resulting conditions for order-optimal convergence are less stringent.

A. Description of AD-SAMD

The setting for AD-SAMD is the same as in Section III. We again suppose a distance function \( \omega : X \to \mathbb{R} \), its associated prox function/Bregman divergence \( V : X \times X \to \mathbb{R} \), and the resulting (Lipschitz) prox mapping \( P_{\gamma} : \mathbb{R}^n \to X \).

We again suppose a mixing matrix \( W \in \mathbb{R}^{m \times m} \) that is symmetric, doubly stochastic, and consistent with \( G \). The main distinction between accelerated and standard mirror descent is the way one averages iterates. Rather than simply average the sequence of iterates, one maintains several distinct sequences of iterates, carefully averaging them along the way. This involves two sequences of step sizes \( \beta_s \in [1, \infty) \) and \( \gamma_s \in \mathbb{R} \), which are not held constant. Again we suppose that the number of mini-batch rounds \( S = T/b \) is predetermined. We detail the steps of AD-SAMD in Algorithm 2.

Algorithm 2 Accelerated distributed stochastic approximation mirror descent (AD-SAMD)

**Require:** Doubly-stochastic matrix \( W \), step size sequences \( \gamma_s, \beta_s \), number of consensus rounds \( r \), batch size \( b \), and stream of mini-batched subgradients \( \theta_i(s) \).

1: for \( i = 1 : m \) do
2: \( x_i(1), x_i^{\text{md}}(1), x_i^{\text{ag}}(1) \leftarrow \min_{x \in X} \omega(x) \) \quad \triangleright Initialize search points
3: end for
4: for \( s = 1 : S \) do
5: for \( i = 1 : m \) do
6: \( x_i^{\text{md}}(s) \leftarrow \beta_s^{-1} x_i(s) + (1 - \beta_s^{-1}) x_i^{\text{ag}}(s) \)
7: \( h_i^0(s) \leftarrow \theta_i(s) \) \quad \triangleright Get mini-batched subgradients
8: end for
9: for \( q = 1 : r, i = 1 : m \) do
10: \( h_i^q(s) \leftarrow \sum_{j \in N_i} w_{ij} h_j^{q-1}(s) \) \quad \triangleright Consensus rounds
11: end for
12: for \( i = 1 : m \) do
13: \( x_{i}(s + 1) \leftarrow P_{\gamma_i(s)}(\gamma_i, h_i^r(s)) \) \quad \triangleright Prox mapping
14: \( x_i^{\text{ag}}(s + 1) \leftarrow \beta_s^{-1} x_i(s + 1) + (1 - \beta_s^{-1}) x_i^{\text{ag}}(s) \)
15: end for
16: end for
return \( x_i^{\text{ag}}(S + 1) \)

The sequences of iterates \( x_i(s), x_i^{\text{md}}(s), \) and \( x_i^{\text{ag}}(s) \) are interrelated in complicated ways; we refer the reader to [8] for an intuitive explanation of these iterations.

B. Convergence Analysis

As with D-SAMD, the convergence analysis relies on bounds on the bias and variance of the averaged subgradients. To this end, we note first that Lemma 2 holds for AD-SAMD, where \( H(s) \) has columns corresponding to noisy subgradients evaluated at \( x_i^{\text{md}}(s) \). Next, we bound the distance between iterates at different nodes. This analysis is somewhat more complicated due to the relationships between the three iterate sequences.
Lemma 4: Let
\[
\begin{align*}
    a_s &\triangleq \max_{i,j} \| x_s^i(s) - x_s^j(s) \| \\
    b_s &\triangleq \max_{i,j} \| x_t^i(s) - x_t^j(s) \| \\
    c_s &\triangleq \max_{i,j} \| x_t^{md}(s) - x_t^{md}(s) \|.
\end{align*}
\]
Then, the moments of \(a_s, b_s,\) and \(c_s\) follow:
\[
\begin{align*}
    E[a_s], E[b_s], E[c_s] &= \frac{\sigma}{\sqrt{BL}}((1 + 2\gamma s m^2 \sqrt{C_\sigma L\lambda_2^s} - 1), \\
    E[a^2_s], E[b^2_s], E[c^2_s] &\leq \frac{\sigma^2}{BL^2}((1 + 2\gamma s m^2 \sqrt{C_\sigma L\lambda_2^s} - 1)^2).
\end{align*}
\]
Using this lemma, we bound the expected gap to optimality of the AD-SAMD iterates.

Theorem 2: For AD-SAMD, the expected gap to optimality satisfies
\[
\begin{align*}
    E[\Psi(x^s(S + 1))] - \Psi^* &\leq \frac{8LD^2_{\omega,x}}{\alpha S^2} + 4D_{\omega,x} \sqrt{\frac{4M + \Delta^2_\omega}{\alpha S}} + \sqrt{\frac{32}{\alpha}} D_{\omega,x} \Xi S, \\
    \Xi^2 &= \sigma^2(1 + \sqrt{C_\sigma m^2 \lambda_2^s})((1 + 2\gamma s m^2 \sqrt{C_\sigma L\lambda_2^s} - 1)^2 + \frac{4C_\sigma \sigma^2}{b}(m^2 + \frac{1}{m}),
\end{align*}
\]
where
\[
\Delta^2_\omega = \frac{4\sigma^2(1 + m^4 C_\sigma \lambda_2^s r^2)}{b}((1 + 2\gamma s m^2 \sqrt{C_\sigma L\lambda_2^s} - 1)^2 + \frac{4C_\sigma \sigma^2}{b}(\lambda_2^s m^2 + \frac{1}{m}).
\]
As with D-SAMD, we study the conditions under which AD-SAMD achieves order-optimum convergence speed. The centralized version of accelerated mirror descent, after processing the \(mT\) data samples that the network sees after \(S\) mini-batch rounds, achieves the convergence rate
\[
O(1) \left[ \frac{L}{(mT)^2} + \frac{M + \sigma}{\sqrt{mT}} \right].
\]
This is the optimum convergence rate under any circumstance. In the following corollary, we derive the conditions under which the converge rate matches the second term, which tends to dominate the error.

Corollary 2: The optimality gap satisfies
\[
E[\Psi(x^s(S + 1))] - \Psi^* = O \left( \frac{M + \sigma}{\sqrt{mT}} \right),
\]
provided
\[
\begin{align*}
    b &= \Omega \left( 1 + \frac{\log(mT)}{\rho \log(1/\lambda_2)} \right), \quad \rho = \Omega \left( \frac{m^{1/4} \log(mT)}{\sigma T^{3/4} \log(1/\lambda_2)} \right) \\
    m &= O(\sigma^2 T), \quad M = O \left( \min \left( \frac{1}{m^2} \sqrt{\frac{1}{m \alpha^2 T}} \right) \right).
\end{align*}
\]
Remark 3: The crucial difference between the two schemes is that AD-SAMD has a convergence rate of \(1/S^2\) in the absence of noise and non-smoothness. This faster term, which is often negligible in centralized mirror descent, means that AD-SAMD tolerates more aggressive mini-batching without impact on the order of the convergence rate. As a result, while the condition on the mini-batch size \(b\) is the same in terms of \(\rho\), the condition on \(\rho\) is relaxed by \(1/4\) in the exponents of \(m\) and \(T\). This is because the condition on \(b = O(T^{1/2})\), which holds for standard stochastic SO methods, is relaxed to \(b = O(T^{3/2})\) for accelerated stochastic mirror descent.

V. NUMERICAL EXAMPLE: LOGISTIC REGRESSION

We demonstrate the performance of D-SAMD and AD-SAMD on supervised learning by considering binary logistic regression. A learning machine observes a stream of pairs \(\xi(t) = (y(t), l(t))\) of data points \(y(t) \in \mathbb{R}^d\) and their labels \(l(t) \in \{0, 1\}\), from which it learns a classifier of the form
\[
F(x, x_0, y) = \frac{1}{1 + \exp(-x^T y(t) + x_0)}.
\]
where \(x \in \mathbb{R}^d\) and \(x_0 \in \mathbb{R}\) are regression coefficients. The function \(F(x, x_0, y(t))\) represents the posterior probability that \(l(t) = 1\) given \(y(t)\).

The SO task is to learn the optimum regression coefficients \(x, x_0\) in terms of the framework of this paper, \(T = \mathbb{R}^d \times \{0, 1\}\), \(X = \mathbb{R}^{d+1}\). We use the Euclidean norm, inner product, and distance-generating function to compute the prox mapping. The objective function is the cross-entropy loss
\[
\psi(x) = E_{y,l}[\log(F(x, x_0, y)) + (1 - l) \log(1 - F(x, x_0, y))].
\]
Minimizing \(\psi\) is equivalent to minimizing the KL divergence between the posterior indicated by \(F\) and the true posterior \(p(l|y)\). Minimizing the empirical loss \(\psi\) over a training set is equivalent to performing maximum likelihood estimation of the regression coefficients.

Although it can be used more generally, logistic regression is well-known to follow from the assumptions of Gaussian naive Bayes [34]. Therefore, we consider a synthetic example in which the data follow a Gaussian distribution. For \(l(t) \in \{0, 1\}\), we let \(y(t) \sim \mathcal{N}(\mu(l(t)), \sigma^2 T)\), where \(\mu(l(t))\) is one of two mean vectors, and \(\sigma^2 > 0\) is the noise variance.

For this experiment, we draw the elements \(\mu_0\) and \(\mu_1\) randomly from the standard normal distribution and choose \(\sigma^2 = 1\). We pick \(m = 20\) and draw a graph at random from the Erdős-Rényi distribution with connection probability 0.1. We choose \(W\) to be the Metropolis weights [35] associated with the resulting graph; it turns out here that \(\lambda_2 = 0.9436\). For \(T = 5000\), we choose step-size \(\gamma = 0.005\) and \(b = \log(T m^2)/\left(\rho \log(1/\lambda_2)\right)\), which guarantees that the equivalent gradient noise variance is \(O(\sigma^2/(m T))\).

In Figure [1] we plot the optimality gap for communications rates \(\rho \in \{1, 10\}\). For \(\rho = 1\) the mini-batch size \(b\) is rather large, so D-SAMD and AD-SAMD only update their search points every few data-acquisition rounds. While one can clearly see the advantage of AD-SAMD over D-SAMD, both centralized algorithms converge much more quickly. By contrast, for \(\rho = 10\) the resulting mini-batch size is smaller, and the gap between centralized and decentralized performance is comparable. Because the other parameters are fixed and \(b = O(T^{1/2})\), each of these schemes has an optimality gap of \(O(1/\sqrt{T})\).

\(^4\)The variance \(\sigma^2\) is distinct from the resulting gradient noise variance \(\sigma^2\).
VI. Conclusion

We have presented two distributed schemes, D-SAMD and AD-SAMD, for convex stochastic optimization over networks of nodes that collaborate via rate-limited links. Further, we have derived sufficient conditions for the order-optimum convergence of D-SAMD and AD-SAMD, showing that accelerated mirror descent provides a foundation for distributed SO that better tolerates slow communications links. These results characterize relationships between network communications speed and the convergence speed of stochastic optimization.

A limitation of this work is that we are restricted to settings in which the prox mapping is Lipschitz continuous, which excludes important Bregman divergences such as the Kullbeck-Liebler divergence. Further, the conditions for optimum convergence restrict the non-smooth component of the objective function to be small. Future work includes study of the limits on convergence speed for general divergences and composite objective functions.

APPENDIX

A. Proof of Lemma 1

The first claim follows immediately from the definitions of the constituent matrices. The second claim follows from Lemma 1. To establish the final claim, we first bound the norm of the columns of $E(s)$:

$$E(s) \triangleq G(s)W^r - \overline{G}(s)$$

$$= G(s)(W^r - 1/n1^T),$$

$$= (G(s) - \overline{G}(s))(W^r - 1/n1^T),$$

where the first equality follows from $\overline{G}(s) = G(s)1/n1^T$ by definition, and the second equality follows from the fact that the dominant eigenspace of a column-stochastic matrix $W$ is the subspace of constant vectors, thus the rows of $G(s)$ lie in the null space of $W^r - 1/n1^T$. We bound the norm of the columns of $E(s)$ via the Frobenius norm of the entire matrix:

$$\|e_i(s)\|_* \leq \sqrt{C^*_1 \|E(s)\|_F}$$

$$= \sqrt{C^*_1 \|G(s) - \overline{G}(s))((W^r - 1/n1^T)\|_F}$$

$$\leq \sqrt{C^*_1 m\lambda^2_2 \|G(s) - \overline{G}(s)\|_F}$$

$$\leq \max_{j,k} m^2 \sqrt{C^*_1 m\lambda^2_2 \|g_j(s) - g_k(s)\|_*},$$

where $C^*_1$ bounds the gap between the Euclidean and dual norms, $C^*_2$ bounds the gap in the opposite direction, thus $C^*_* \leq C^*_1C^*_2$. Then, we bound the variance of the subgradient noise. Similar to the case of $E(s)$, we can rewrite

$$Z(s) = Z(s)(W^r - 11^T).$$

We bound the expected squared norm of each column $z_i(s)$ in terms of the expected Frobenius norm:

$$E[\|z_i(s)\|_*^2] \leq C^*_1 E[\|Z(s)(W^r - 11^T)\|_F^2]$$

$$\leq m\lambda^2_2 C^*_1 E[\|Z(s)\|_*^2]$$

$$\leq \lambda^2_2 m^2 C^*_* \sigma^2/b.\]
where $i^*$ is the index that maximizes $\|\tilde{z}_i\|_s$, the final expression (34) gives the recurrence relationship that governs the divergence of the iterates, and it takes the form of a standard first-order difference equation. Recalling the base case $a_0 = 0$, we obtain the solution

$$a_s \leq \sum_{\tau=0}^{s-1} (1 + \alpha m^2 \sqrt{C_s \lambda_2})^{s-1-\tau} \frac{\alpha}{L} \|\tilde{z}_s(\tau)\|_s.$$  

(35)

To prove the lemma, we bound the moments of $a_s$ using the solution (35). First, we bound $E[a_s^2]$

$$E[a_s^2] \leq E \left[ \left( \sum_{\tau=0}^{s-1} (1 + \alpha m^2 \sqrt{C_s \lambda_2})^{s-1-\tau} \frac{\alpha}{L} \|\tilde{z}_s(\tau)\|_s \right)^2 \right],$$

which via a change of variables we can rewrite as

$$E[a_s^2] \leq \left( \frac{\alpha}{L} \right)^2 \sum_{\tau=0}^{s-1} \sum_{q=0}^{s-1} (1 + \alpha m^2 \sqrt{C_s \lambda_2})^{s-1-q} \|\tilde{z}_s(s-1-\tau)\|_s \|\tilde{z}_s(s-1-q)\|_s.$$ 

Then, we apply the Cauchy-Schwartz inequality and (23) to $E[\|\tilde{z}_s(s-1-\tau)\|_s \|\tilde{z}_s(s-1-q)\|_s]$ to obtain

$$E[a_s^2] \leq \frac{\lambda^2}{b} \sigma^2 C_s \|m\| \left( \frac{\alpha}{L} \right)^2 \sum_{\tau=0}^{s-1} (1 + \alpha m^2 \sqrt{C_s \lambda_2})^{s-1} \sum_{q=0}^{s-1} (1 + \alpha m^2 \sqrt{C_s \lambda_2})^q.$$ 

Finally, we apply the geometric sum identity and simplify to obtain

$$E[a_s^2] \leq \frac{\lambda^2}{b} \sigma^2 C_s \|m\| \left( \frac{\alpha}{L} \right)^2 \left( 1 - (1 + \alpha m^2 \sqrt{C_s \lambda_2})^s \right) \left( 1 - (1 + \alpha m^2 \sqrt{C_s \lambda_2})^s \right).$$ 

This establishes the result for $E[a_s^2]$, and the bound on $E[a_s]$ follows a fortiori from Jensen’s inequality.

C. Proof of Theorem 1

The proof involves an analysis of mirror descent, with the subgradient error quantified by Lemmas 2 and 3. As necessary, we will cite results from the proof of Theorem 1 rather than reproduce its analysis.

Define

$$\delta_i(\tau) \triangleq h_i(\tau) - g(x_i(\tau))$$

$$\eta_i(\tau) \triangleq g(\tau) + e_i(\tau) - g(x_i(\tau)) = \delta_i(\tau) - \tilde{z}_i(\tau) - \tilde{z}_i(\tau)$$

$$\zeta_i(\tau) \triangleq \gamma(\delta_i(\tau), x^* - x_i(\tau)).$$

Applying Lemmas 2 and 3 we bound $E[\|\eta_i(\tau)\|_s]$

$$E[\|\eta_i(\tau)\|_s] = E[\|g(\tau) + e_i(\tau) - g(x_i(\tau))\|_s]$$

$$\leq E[\|g(\tau) - g(x_i(\tau))\|_s] + E[\|e_i(\tau)\|_s]$$

$$\leq LE[a_r] + Lm^2 \sqrt{C_s \lambda_2} E[a_r]$$

$$\leq L(1 + m^2 \sqrt{C_s \lambda_2}) E[a_r]$$

$$\leq \frac{\sigma(1 + m^2 \sqrt{C_s \lambda_2})}{m \sqrt{b}} ((1 + \alpha m^2 \sqrt{C_s \lambda_2})^s - 1).$$

Next, we bound the expected value of $\zeta_i(\tau)$

$$E[\zeta_i(\tau)] = \gamma E[(\eta_i(\tau) + \tilde{z}_i(\tau) - \tilde{z}_i(\tau), x^* - x_i(\tau))]$$

$$\leq \gamma E[(\eta_i(\tau), x^* - x_i(\tau))] + \gamma E[\tilde{z}_i(\tau) - \tilde{z}_i(\tau), x^* - x_i(\tau)]$$

$$\leq \gamma E[\|\tilde{z}_i(\tau)\|_s \|x^* - x_i(\tau)\|_s]$$

$$\leq \gamma \Xi \bar{\Omega}_\omega \|x^* - x_i(\tau)\|_s.$$ 

where the first inequality is due to the definition of the dual norm, and the third equality follows because $\tilde{z}_i(\tau)$ and $\tilde{z}_i(\tau)$ have zero mean and are independent of $x^* - x_i(\tau)$.

Next, we bound $E[\|\delta_i(\tau)\|_s^2]$. By Lemma 2

$$E[\|\delta_i(\tau)\|_s^2] = E \left[ \left\| \bar{g}(\tau) + e_i(\tau) - g(x_i(\tau)) \right\|^2 ight],$$

which we can bound via repeated uses of the triangle inequality:

$$E[\|\delta_i(\tau)\|_s^2] \leq 2E[\|\bar{g}(\tau) - g(x_i(\tau))\|_s^2]$$

$$+ 2E[\|e_i(\tau)\|_s^2] + 4E[\|\bar{z}(\tau)\|_s^2 + \|\tilde{z}_i(\tau)\|_s^2].$$

Next, we apply Lemma 2

$$E[\|\delta_i(\tau)\|_s^2] \leq 2L^2(1 + m^2 C_s \lambda_2^2) E[a_r^2] + 4C_s \sigma^2/(mb) + 4\lambda^2 \sigma C_s^2 m^2 / b = \Delta_s^2.$$ 

Finally, we apply Lemma 3

$$E[\|\delta_i(\tau)\|_s^2] \leq 2L^2(1 + m^2 C_s \lambda_2^2) E[a_r^2] + 4C_s \sigma^2/(mb) + 4\lambda^2 \sigma C_s^2 m^2 / b = \Delta_s^2.$$ 

From the proof of Theorem 1 of [8] to the D-SAMD iterates, we have that

$$S\gamma(\psi(x_{i}^{\infty}(S+1)) - \psi^*) \leq D^2 + \frac{2\gamma^2}{\alpha} \left( 4M^2 + \|\delta_i(\tau)\|_s^2 \right).$$

We take the expectation to obtain

$$E[\psi(x_{i}^{\infty}(S+1)) - \psi^*] \leq \frac{D^2}{S\gamma} + \frac{1}{S} \sum_{r=1}^{s} \sqrt{\frac{\alpha \Xi D \omega}{2}} \frac{\Delta_s}{L} + \frac{2\gamma}{\alpha} \left( 4S^2 + \frac{1}{S} \sum_{r=1}^{s} \Delta_s^2 \right).$$

Because $\{\Xi_s\}$ and $\{\Delta_s\}$ are increasing sequences, we can bound the preceding by

$$E[\psi(x_{i}^{\infty}(S+1)) - \psi^*] \leq \frac{D^2}{S\gamma} + \frac{2\gamma}{\alpha} \left( 4M^2 + \Delta_s^2 \right) + \sqrt{\frac{\alpha \Xi S D \omega}{2}} \frac{\Delta_s}{L},$$

Minimizing this bound over $\gamma$ in the interval $(0, \alpha/(2L))$, we obtain the following optimum value:

$$\gamma^* = \min \left\{ \frac{\alpha}{2L}, \sqrt{\frac{\alpha D^2}{2S^2 (4M^2 + 2\Delta_s)}} \right\}.$$
which results in the bound
\[ E[\psi(x_i^{av}(s+1))] - \psi^* \leq \frac{2LD^2}{\alpha S^2} + \sqrt{\frac{2(4M^2 + 2\Delta^2)}{\alpha S}} + \sqrt{\frac{2}{2^L}}. \]

\[ \Delta = \frac{2}{b}. \]

Now, we find the optimum scaling law on the mini-batch size \( b \).

The resulting gap to optimality scales as
\[ \Xi_s = \frac{s(1 + m^2 \sqrt{C_s \lambda_s^2})}{m^3 \sqrt{b}} \leq 2 \sqrt{C_s m/b(1 + \sigma^2 \sqrt{C_s \lambda_s^2})^k}, \]
\[ = 2 \sqrt{C_s m/b} \sum_{k=0}^{s} (s \alpha m^2 \sqrt{C_s \lambda_s^2})^k, \]
\[ \leq 2 \sqrt{C_s m/b} \sum_{k=0}^{s} (\sigma m^2 \sqrt{C_s \lambda_s^2})^k, \]
\[ \leq 2 \sqrt{C_s m/b} (s + 1) (\sigma m^2 \sqrt{C_s \lambda_s^2}), \]
\[ = O(s^2 \sigma \sqrt{m^5/b \lambda_s^2}), \]

where the first inequality is trivial, the next two statements are due to the binomial theorem and the exponential upper bound on the binomial coefficient, and the final inequality is true if we constrain \( s \sigma m^2 \sqrt{C_s \lambda_s^2} \leq 1 \), which is consistent with the order-wise constraints listed in the statement of this corollary.

Recalling that \( r \leq b \rho \), we obtain
\[ \Xi_S = O\left( \frac{S^2 \lambda_S^b \sigma \sqrt{m^5/b}}{\rho} \right). \]

Now, we find the optimum scaling law on the mini-batch size \( b \). To ensure \( s \sigma m^2 \sqrt{C_s \lambda_s^2} \leq 1 \) for all \( s \), we need
\[ b = \Omega \left( \frac{\log(mT)}{\rho \log(1/\lambda_{s})} \right). \]

Next, in order to get optimum scaling we need \( \Xi_S = O(\sigma/\sqrt{mb}) \), which yields a slightly stronger necessary condition:
\[ b = \Omega \left( \frac{\log(mT)}{\rho \log(1/\lambda_{s})} \right). \]

where the first term is necessary because \( b \) cannot be smaller than unity.

Similar approximations establish that
\[ \Delta^2 = \Theta(s^4 \sigma^2 m^5/b \lambda_{s}^2 \sigma + \sigma^2/(mb) + \lambda_{s}^2 \sigma^2 m^2/b). \]

Thus, \( \Xi_S = O(\sigma^2/(mT)) \), as is needed for order-optimal convergence.

The resulting gap to optimality scales as
\[ E[\Psi(x_i^{av}(S+1))] - \Psi^* = O\left( \frac{b}{T} + \sqrt{\frac{\rho M^2}{T}} + \frac{\sigma}{\sqrt{mT}} \right). \]

Substituting \( \Xi_S = O(\sigma^2/(mT)) \) yields
\[ E[\Psi(x_i^{av}(S+1))] - \Psi^* = O\left( \max\left\{ \frac{\log(mT)}{\rho T \log(1/\lambda_{s})}, \frac{1}{T} \right\} + \sqrt{\frac{M^2 \log(mT)}{\rho T \log(1/\lambda_{s})}}, \frac{\sqrt{M^2}}{T} \right) + \frac{\sigma}{\sqrt{mT}}. \]

In order to achieve order optimality, we need the first term to be \( O(\sigma/\sqrt{mT}) \), which requires
\[ m = O(\sigma^2 T), \quad \rho = \Omega \left( \frac{T^{1/2} \log(1/\lambda_{s})}{\sigma} \right). \]

Finally, we need the second term to be \( O(M/\sqrt{mT}) \), which requires
\[ \mathcal{M} = O\left( \min\left\{ \frac{1}{m}, \frac{1}{\sqrt{mT^2}} \right\} \right). \]

This establishes the result.

\[ E. \text{ Proof of Lemma} \]

The sequences \( a_s, b_s, c_s \) are interdependent. Rather than solving the complex recursion directly, we bound the dominant sequence. Define \( d_s = \max\{a_s, b_s, c_s\} \). First, we bound \( a_s \):
\[ a_{s+1} = \max_{i,j} \| \beta_{s+1}^{-1} (x_i(s+1) - x_j(s+1)) + (1 - \beta_{s+1}^{-1}) (x_i^{ag}(s) - x_j^{ag}(s)) \|, \]

which, via the triangle inequality, is bounded by
\[ a_{s+1} \leq \max_{i,j} \beta_{s+1}^{-1} \| P_{x_i(s)}(\gamma_i h_i(s)) - P_{x_j(s)}(\gamma_j h_j(s)) \| + (1 - \beta_{s+1}^{-1}) a_s. \]

Because the prox-mapping is Lipschitz,
\[ a_{s+1} \leq \max_{i,j} \beta_{s+1}^{-1} (\| x_i(s) - x_j(s) \| + \gamma_s \| h_i(s) - h_j(s) \|), \]
\[ + (1 - \beta_{s+1}^{-1}) a_s. \]

Applying the first Lemma yields
\[ a_{s+1} \leq \max_{i,j} \beta_{s+1}^{-1} b_s + (1 - \beta_{s+1}^{-1}) a_s + \beta_{s+1}^{-1} \gamma_s \| e_i(s) - e_j(s) + \tilde{z}_i(s) - \tilde{z}_j(s) \|. \]

and applying the second part and the triangle inequality yields
\[ a_{s+1} \leq \max_{i,j} d_s + 2 \gamma_s m^2 \sqrt{C_s \lambda_s^2} \| g_i(s) - g_j(s) \| + 2 \gamma_s \| \tilde{z}_i(s) \|. \]

Finally, we invoke the Lipschitz continuity of the subgradient and collect terms to obtain
\[ a_{s+1} \leq \max_{i,j} d_s + 2 \gamma_s m^2 \sqrt{C_s \lambda_s^2} c_s + 2 \gamma_s \| \tilde{z}_i(s) \| \leq \max_{i,j} \left( 1 + 2 \gamma_s m^2 \sqrt{C_s \lambda_s^2} d_s + 2 \gamma_s \| \tilde{z}_i(s) \| \right). \]

Next, we bound \( b_s \) via similar steps:
\[ b_{s+1} = \max_{i,j} \| P_{x_i(s)}(\gamma_i h_i(s)) - P_{x_j(s)}(\gamma_j h_j(s)) \| \leq \max_i L_P \| x_i(s) - x_j(s) \| + L_P \gamma_s \| h_i(s) - h_j(s) \|, \]
\[ \leq \max_i b_s + 2 \gamma_s m^2 \sqrt{C_s \lambda_s^2} c_s + \| \tilde{z}_i(s) \|, \]
\[ \leq \max_i (1 + 2 \gamma_s m^2 \sqrt{C_s \lambda_s^2} d_s + 2 \gamma_s \| \tilde{z}_i(s) \|). \]
Finally, we bound $c_z$:

$$c_{s+1} = \max_{i,j} \|\beta_s^{-1}(x_i(s+1) - x_j(s+1)) + (1 - \beta_s^{-1})(x_{i,y}^{s\epsilon}(s) - x_{j,y}^{s\epsilon}(s))\|.$$

We bound this sequence in terms of $a_s$ and $b_s$:

$$c_{s+1} \leq \max_{i,j} \beta_s^{-1}b_{s+1} + (1 - \beta_s^{-1})a_s$$

$$\leq \max_{i} \beta_s^{-1}(1 + 2\gamma_s m^2 \sqrt{C_\gamma L\lambda_2^s})d_s + 2\gamma_s \|z_i(s)\|$$

Solving the recursion yields

$$d_s \leq \left(1 + 2\gamma_s m^2 \sqrt{C_\gamma L\lambda_2^s}\right)^2 + 2\gamma_s \|z_i(s)\|.$$

Comparing the three bounds, we observe that

$$d_{s+1} \leq (1 + 2\gamma_s m^2 \sqrt{C_\gamma L\lambda_2^s})d_s + 2\gamma_s \|z_i(s)\|.$$

Solving the recursion yields

$$d_s \leq \sum_{\tau=0}^{s-1} (1 + 2\gamma_s m^2 \sqrt{C_\gamma L\lambda_2^\tau})^2 + 2\gamma_s \|z_i(s)\|.$$

Following the same argument as in Lemma 3, we obtain

$$E[d_s^2] \leq 4\gamma_s^2 E[\|z_i(s)\|^2] \left(\sum_{\tau=0}^{s-1} (1 + 2\gamma_s m^2 \sqrt{C_\gamma L\lambda_2^\tau})^2\right)^2$$

$$\leq 4\gamma_s^2 m^4 \lambda_2^\tau C_\gamma \frac{\sigma^2}{b} \left(1 + 2\gamma_s m^2 \sqrt{C_\gamma L\lambda_2^\tau}\right)^2 - 1)^2$$

$$= \frac{\sigma^2}{b^2} (((1 + 2\gamma_s m^2 \sqrt{C_\gamma L\lambda_2^\tau}))^2 - 1)^2.$$

The bound on $E[d_s]$ follows a fortiori.

F. Proof of Theorem 2

Similar to the proof of Theorem 1, the proof involves an analysis of accelerated mirror descent, and as necessary we will cite results from Theorem 2. Define

$$\delta_i(\tau) = h_i(\tau) - g(x_i^{md}(\tau))$$

$$\eta_i(\tau) = g(\tau) + e_\tau(\tau) - g(x_i^{md}(\tau)) = \delta_i(\tau) - z_i(\tau)$$

$$\zeta_i(\tau) = \gamma_{\tau\gamma} \delta_i(\tau) - x^\ast(i) - x_i(\tau).$$

Applying Lemmas 2 and 3, we bound $E[\|\eta_i(\tau)\|]$:

$$E[\|\eta_i(\tau)\|] = E[\|g(\tau) + e_\tau(\tau) - g(x_i^{md}(\tau))\|]
\leq LE[\|e_\tau\|] + L\sqrt{C_\gamma m^2 \lambda_2^\gamma} E[\|e_\tau\|]
\leq \frac{\sigma^2}{m \sqrt{\gamma \lambda_2^\gamma}} ((1 + 2\gamma_m m^2 \sqrt{C_\gamma L\lambda_2^\tau}) - 1).$$

By the definition of the dual norm:

$$E[\zeta_i(\tau)] \leq \gamma_{\tau} \frac{\sigma^2}{m \sqrt{\gamma \lambda_2^\gamma}} \|x^\ast - x_i(\tau)\| \leq \gamma_{\tau} \frac{\sigma^2}{m \sqrt{\gamma \lambda_2^\gamma}} \Omega_{\tau} \leq \sqrt{\frac{\alpha \Omega_{\tau}}{2L}}.$$

Similarly, we bound $E[\|\delta_i(\tau)\|^2]$ via Lemma 2

$$E[\|\delta_i(\tau)\|^2] \leq 4E[\|g(\tau) - g(x_i^{md}(\tau))\|^2] + 4E[\|e_\tau(\tau)\|^2] + 4E[\|z_i(\tau)\|^2] + 4E[\|z_i(\tau)\|^2],$$

which is bounded by

$$E[\|\delta_i(\tau)\|^2] \leq 4L^2 (1 + 4m^2 C_\gamma \lambda_2^\gamma) E[\|\epsilon_{\tau\gamma}\|^2] + \frac{4C_\gamma \frac{\sigma^2}{\gamma \lambda_2^\gamma}}{b^m} (\lambda_2^\gamma m^2 + 1 + m^2).$$

Applying Lemma 4, we obtain

$$E[\|\delta_i(\tau)\|^2] \leq \frac{4\gamma_s^2 (1 + 4m^2 C_\gamma \lambda_2^\gamma)}{b} ((1 + 2\gamma_s m^2 \sqrt{C_\gamma L\lambda_2^\gamma}) - 1)^2 + \frac{4C_\gamma \frac{\sigma^2}{\gamma \lambda_2^\gamma}}{b^m} (\lambda_2^\gamma m^2 + 1 + m^2).$$

From the proof of Theorem 2, we observe that

$$\Delta_{\gamma} \leq \Delta_{\gamma}(\lambda_2^\gamma m^2 + 1 + m^2).$$

Taking the expectation yields

$$\Delta_{\gamma}(\lambda_2^\gamma m^2 + 1 + m^2).$$

Solving for the optimal $\gamma$ in the interval $0 \leq \gamma \leq \alpha/(2L)$, we obtain:

$$\gamma^* = \min \left\{ \frac{\alpha}{2L}, \sqrt{\frac{\alpha \Omega_{\tau}}{2L}} \right\}.$$

This gives the bound

$$E[\|\delta_i(\tau)\|^2] \leq \frac{8L^2 D_{\omega,X}}{\alpha (S^2 + 1)} + \frac{16SD_{\omega,X}^2 (4M^2 + \Delta_\gamma^2)}{\alpha (S + 1)^2} + \sqrt{\frac{32}{\alpha}} D_{\omega,X} \Xi_{S}.$$
and
\[ \Delta^2 = O \left( \frac{\sigma^2 + \delta + \lambda^2}{b \log(1/\lambda^2)} + \frac{\sigma^2}{mb} \right). \]

To obtain order optimum convergence, we need \( \Delta^2 = O(\sigma^2/(mb)) \). Combining the above equation with the constraint \( r \leq b\rho \), we obtain the condition
\[ b = \Omega \left( 1 + \frac{\log(mT)}{\rho \log(1/\lambda^2)} \right). \]

This condition also ensures that \( E[\Psi(x_{mT}^\infty(S + 1))] - \Psi^* = O \left( \frac{b^2}{T^2} + \frac{bM^2}{T} + \frac{\sigma}{\sqrt{mT}} \right). \)

In order to make the first term \( O(\sigma/\sqrt{mT}) \), we need
\[ m = O(\sigma^2T), \quad \rho = \Omega \left( \frac{m^{1/4} \log(mT)}{\sigma^{3/4} \log(1/\lambda^2)} \right). \]

To make the second term \( O(\sigma/\sqrt{mT}) \) we need, as before,
\[ M = O \left( \min \left\{ \frac{1}{m}, \frac{1}{\sqrt{m\alpha^2T}} \right\} \right). \]

REFERENCES

[1] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. MIT Press, 2016, http://www.deeplearningbook.org

[2] H. Kushner, “Stochastic approximation: a survey,” Wiley Interdisciplinary Reviews: Computational Statistics, vol. 2, no. 1, pp. 87–96, 2010.

[3] S. Kim, R. Pasupathy, and S. G. Henderson, “A guide to sample average approximation,” in Handbook of Simulation Optimization. Springer, 2015, pp. 207–243.

[4] M. Pereyra, P. Schniter, E. Chouzenoux, J.-C. Pesquet, J.-Y. Tourneret, A. O. Hero, and S. McLaughlin, “A survey of stochastic simulation and optimization methods in signal processing,” IEEE Journal of Selected Topics in Signal Processing, vol. 10, no. 2, pp. 224–241, 2016.

[5] H. Robbins and S. Monro, “A stochastic approximation method,” The annals of mathematical statistics, pp. 400–407, 1951.

[6] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro, “Robust stochastic approximation approach to stochastic programming,” SIAM Journal on Optimization, vol. 19, no. 4, pp. 1574–1609, 2009.

[7] A. Juditsky, A. Nemirovski, and C. Tauvel, “Solving variational inequalities with stochastic mirror-prox algorithm,” Stochastic Systems, vol. 1, no. 1, pp. 17–58, 2011.

[8] G. Lan, “An optimal method for stochastic composite optimization,” Mathematical Programming, vol. 133, no. 1, pp. 365–397, 2012.

[9] C. Hu, W. Pan, and J. T. Kwok, “Accelerated gradient methods for stochastic optimization and online learning,” in Advances in Neural Information Processing Systems, 2009, pp. 781–789.

[10] L. Xiao, “Dual averaging methods for regularized stochastic learning and online optimization,” Journal of Machine Learning Research, vol. 11, no. Oct, pp. 2543–2596, 2010.

[11] X. Chen, Q. Lin, and J. Pena, “Optimal regularized dual averaging methods for stochastic optimization,” in Advances in Neural Information Processing Systems, 2012, pp. 395–403.

[12] J. Tsetsiklis, D. Bertsekas, and M. Athans, “Distributed asynchronous deterministic and stochastic gradient optimization algorithms,” IEEE Trans. Automat. Control, vol. 31, no. 9, pp. 803–812, Sep. 1986.

[13] B. Recht, C. Re, S. Wright, and F. Niu, “Hogwild: A lock-free approach to parallelizing stochastic gradient descent,” in Advances in Neural Information Processing Systems, 2011, pp. 693–701.

[14] H. Mania, X. Pan, D. Papailiopoulos, B. Recht, K. Ramchandran, and M. I. Jordan, “Perturbed iterate analysis for asynchronous stochastic optimization,” arXiv preprint arXiv:1507.06970, 2015.

[15] M. Rabbat and R. Nowak, “Distributed optimization in sensor networks,” in Proc. 3rd Intl. Symp. Information Processing in Sensor Networks (IPSN’04), Berkeley, CA, Apr. 2004, pp. 20–27.

[16] M. G. Rabbat and R. D. Nowak, “Quantized incremental algorithms for distributed optimization,” IEEE J. Select. Areas Commun., vol. 23, no. 4, pp. 798–808, Apr. 2005.

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

[18] A. Nedic and D. P. Bertsekas, “Incremental subgradient methods for nondifferentiable optimization,” SIAM Journal on Optimization, vol. 12, no. 1, pp. 109–138, 2001.

[19] B. Johansson, M. Rabl, and M. Johansson, “A randomized incremental subgradient method for distributed optimization in networked systems,” SIAM J. Optim., vol. 20, no. 3, pp. 1157–1170, 2010.

[20] S. S. Ram, A. Nedić, and V. V. Veeravalli, “Incremental stochastic subgradient algorithms for convex optimization,” SIAM J. Optim., vol. 20, no. 2, pp. 691–717, 2009.

[21] A. Nedic and A. Ozdaglar, “Distributed subgradient methods for multi-agent optimization,” IEEE Trans. Automat. Control, vol. 54, no. 1, pp. 48–61, Jan. 2009.

[22] K. Srivastava and A. Nedic, “Distributed asynchronous constrained stochastic optimization,” IEEE J. Select. Topics Signal Processing, vol. 5, no. 4, pp. 772–790, Aug. 2011.

[23] K. I. Tsianos, S. Lawlor, and M. G. Rabbat, “Push-sum distributed dual averaging for convex optimization,” in Proc. 51st IEEE Conf. Decision and Control (CDC’12), Maui, HI, Dec. 2012, pp. 5453–5458.

[24] A. Mokhtari and A. Ribeiro, “DSA: Decentralized double stochastic averaging gradient algorithm,” J. Mach. Learn. Res., vol. 17, no. 61, pp. 1–35, 2016.

[25] A. S. Bjiral, A. D. Sarwate, and N. Srebro, “On data dependence in distributed stochastic optimization,” arXiv preprint arXiv:1605.04379, 2016.

[26] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, “On the linear convergence of the ADMM in decentralized consensus optimization,” IEEE Trans. Signal Processing, vol. 62, no. 7, pp. 1750–1761, Apr. 2014.

[27] D. Jakovetić, J. M. F. Moura, and J. Xavier, “Linear convergence rate of a class of distributed augmented lagrangian algorithms,” IEEE Trans. Signal Processing, vol. 63, no. 24, pp. 6216–6231, 2015.

[28] A. G. Dimakis, S. Kar, J. M. Moura, M. G. Rabbat, and A. Scaglione, “Gossip algorithms for distributed signal processing,” Proceedings of the IEEE, vol. 98, no. 11, pp. 1847–1864, 2010.

[29] S. Sundhar Ram, A. Nedic, and V. Veeravalli, “Distributed stochastic subgradient projection algorithms for convex optimization,” J. Optim. Theory App., vol. 147, no. 3, pp. 516–545, Jul. 2010.

[30] J. Duchi, A. Agarwal, and M. Wainwright, “Dual averaging for distributed optimization: Convergence analysis and network scaling,” IEEE Trans. Automat. Control, vol. 57, no. 3, pp. 592–606, Mar. 2012.

[31] O. Dekel, R. Gilad-Bachrach, O. Shamir, and L. Xiao, “Optimal distributed online prediction using mini-hatches,” Journal of Machine Learning Research, vol. 13, no. Jan, pp. 165–202, 2012.

[32] K. I. Tsianos and M. G. Rabbat, “Efficient distributed online prediction and stochastic optimization with approximate distributed averaging,” IEEE Transactions on Signal and Information Processing over Networks, vol. 2, no. 4, pp. 489–506, Dec 2016.

[33] B. T. Polyak and A. B. Juditsky, “Acceleration of stochastic approximation by averaging,” SIAM J. Optim. Control and Optimization, vol. 30, no. 4, pp. 838–855, 1992.

[34] C. M. Bishop, Pattern Recognition and Machine Learning. New York, NY: Springer, 2006.

[35] L. Xiao, S. Boyd, and S. Lall, “A scheme for robust distributed sensor fusion based on average consensus,” in Proceedings of the 4th International Symposium on Information Processing in Sensor Networks, 2005, p. 9.