An Asynchronous Mini-Batch Algorithm for Regularized Stochastic Optimization

Hamid Reza Feyzmahdavian, Arda Aytekin, and Mikael Johansson

Abstract

Mini-batch optimization has proven to be a powerful paradigm for large-scale learning. However, the state of the art parallel mini-batch algorithms assume synchronous operation or cyclic update orders. When worker nodes are heterogeneous (due to different computational capabilities or different communication delays), synchronous and cyclic operations are inefficient since they will leave workers idle waiting for the slower nodes to complete their computations. In this paper, we propose an asynchronous mini-batch algorithm for regularized stochastic optimization problems with smooth loss functions that eliminates idle waiting and allows workers to run at their maximal update rates. We show that by suitably choosing the step-size values, the algorithm achieves a rate of the order $\mathcal{O}(1/\sqrt{T})$ for general convex regularization functions, and the rate $\mathcal{O}(1/T)$ for strongly convex regularization functions, where T is the number of iterations. In both cases, the impact of asynchrony on the convergence rate of our algorithm is asymptotically negligible, and a near-linear speedup in the number of workers can be expected. Theoretical results are confirmed in real implementations on a distributed computing infrastructure.

I. Introduction

Many optimization problems that arise in machine learning, signal processing, and statistical estimation can be formulated as *regularized stochastic optimization* (also referred to as *stochastic composite optimization*) problems in which one jointly minimizes the expectation of a stochastic loss function plus a possibly nonsmooth regularization term. Examples include Tikhonov and elastic net regularization, Lasso, sparse logistic regression, and support vector machines [1]–[5].

Stochastic approximation methods such as stochastic gradient descent were among the first algorithms developed for solving stochastic optimization problems [6]. Recently, these methods have received significant attention due to their simplicity and effectiveness (see, *e.g.*, [7]–[13]). In particular, Nemirovski *et. al.* [7] demonstrated that for nonsmooth stochastic convex optimization problems, a modified stochastic approximation method, the *mirror descent*, exhibits an unimprovable convergence rate $\mathcal{O}(1/\sqrt{T})$, where T is the number of iterations. Later, Lan [8] developed a mirror descent algorithm for stochastic composite convex problems which explicitly accounts for the

H. R. Feyzmahdavian, A. Aytekin, and M. Johansson are with the Department of Automatic Control, School of Electrical Engineering and ACCESS Linnaeus Center, Royal Institute of Technology (KTH), SE-100 44 Stockholm, Sweden. Emails: {hamidrez, aytekin, mikaelj}@kth.se.

smoothness of the loss function and achieves the optimal rate. A similar result for the dual averaging method was obtained by Xiao [9].

The methods for solving stochastic optimization problems cited above are inherently *serial* in the sense that the gradient computations take place on a single processor which has access to the whole dataset. However, it happens more and more often that one single computer is unable to store and handle the amounts of data that we encounter in practical problems. This has caused a strong interest in developing *parallel* optimization algorithms which are able to split the data and distribute the computation across multiple processors or multiple computer clusters (see, *e.g.*, [14]–[31] and references therein).

One simple and popular stochastic approximation method is *mini-batching*, where iterates are updated based on the average gradient with respect to multiple data points rather than based on gradients evaluated at a single data at a time. Recently, Dekel *et. al.* [32] proposed a parallel mini-batch algorithm for regularized stochastic optimization problems, in which multiple processors compute gradients in parallel using their own local data, and then aggregate the gradients up a spanning tree to obtain the averaged gradient. While this algorithm can achieve linear speedup in the number of processors, it has the drawback that the processors need to synchronize at each round and, hence, if one of them fails or is slower than the rest, then the entire algorithm runs at the pace of the slowest processor.

In this paper, we propose an *asynchronous* mini-batch algorithm for regularized stochastic optimization problems with smooth loss functions that eliminates the overhead associated with global synchronization. Our algorithm allows multiple processors to work at *different rates*, perform computations *independently* of each other, and update global decision variables using *out-of-date* gradients. A similar model of parallel asynchronous computation was applied to coordinate descent methods for deterministic optimization in [33]–[35] and mirror descent and dual averaging methods for stochastic optimization in [36]. In particular, Agarwal and Duchi [36] have analyzed the convergence of asynchronous mini-batch algorithms for smooth stochastic convex problems, and interestingly shown that bounded delays do not degrade the asymptotic convergence. However, they only considered the case where the regularization term is the indicator function of a compact convex set.

We extend the results of [36] to general regularization functions (like the l_1 norm, often used to promote sparsity), and establish a sharper expected-value type of convergence rate than the one given in [36]. Specifically, we make the following contributions:

- **(i)** For general convex regularization functions, we show that when the constraint set is closed and convex (but not necessarily bounded), the running average of the iterates generated by our algorithm with constant step-sizes converges at rate $\mathcal{O}(1/T)$ to a ball around the optimum. We derive an explicit expression that quantifies how the convergence rate and the residual error depends on loss function properties and algorithm parameters such as the constant step-size, the batch size, and the maximum delay bound $\tau_{\max}$.
- **(ii)** For general convex regularization functions and compact constraint sets, we prove that the running average of the iterates produced by our algorithm with a time-varying step-size converges to the true optimum (without

residual error) at rate

$$\mathcal{O}\left(\frac{(\tau_{\max}+1)^2}{T}+\frac{1}{\sqrt{T}}\right).$$

This result improves upon the previously known rate

$$\mathcal{O}\left(\frac{\tau_{\max}^2\log T}{T}+\frac{\tau_{\max}+1}{T}+\frac{1}{\sqrt{T}}\right)$$

for delayed stochastic mirror descent methods with time-varying step-sizes given in [36]. In this case, our algorithm enjoys near-linear speedup as long as the number of processors is $\mathcal{O}(T^{1/4})$.

(iii) When the regularization function is strongly convex and the constraint set is closed and convex, we establish that the iterates converge at rate

$$\mathcal{O}\left(\frac{(\tau_{\max}+1)^4}{T^2}+\frac{1}{T}\right).$$

If the number of processors is of the order of $\mathcal{O}(T^{1/4})$, this rate is $\mathcal{O}(1/T)$ asymptotically in T, which is the best known rate for strongly convex stochastic optimization problems in a serial setting.

The remainder of the paper is organized as follows. In Section II, we introduce the notation and review some preliminaries that are essential for the development of the results in this paper. In Section III, we formulate the problem and discuss our assumptions. The proposed asynchronous mini-batch algorithm and its main theoretical results are presented in Section IV. Computational experience is reported in Section V while Section VI concludes the paper.

II. Notation and Preliminaries

A. Notation

We let $\mathbb{N}$ and $\mathbb{N}_0$ denote the set of natural numbers and the set of natural numbers including zero, respectively. The inner product of two vectors $x, y \in \mathbb{R}^n$ is denoted by $\langle x, y\rangle$. We assume that $\mathbb{R}^n$ is endowed with a norm $\|\cdot\|$, and use $\|\cdot\|_*$ to represent the corresponding dual norm, defined by

$$\|y\|_* = \sup_{\|x\|\leq 1} \langle x, y\rangle.$$

B. Preliminaries

Next, we review the key definitions and results necessary for developing the main results of this paper. We start with the definition of a *Bregman distance function*, also referred to as a *prox-function*.

Definition 1: *A function $\omega : X \to \mathbb{R}$ is called a distance generating function with modulus $\mu_\omega > 0$ with respect to norm $\|\cdot\|$, if ω is continuously differentiable and μ_ω-strongly convex with respect to $\|\cdot\|$ over the set $X \subseteq \mathbb{R}^n$. That is, for all $x, y \in X$,*

$$\omega(y) \geq \omega(x) + \langle \nabla\omega(x), y - x\rangle + \frac{\mu_\omega}{2}\|y - x\|^2.$$

Every distance generating function introduces a corresponding Bregman distance function

$$D_\omega(x,y) := \omega(y) - \omega(x) - \langle \nabla\omega(x), y - x\rangle.$$

For example, choosing $\omega(x) = \frac{1}{2}\|x\|_2^2$, which is 1-strongly convex with respect to the l_2-norm over any convex set X, would result in $D_\omega(x,y) = \frac{1}{2}\|x-y\|_2^2$. Another common example of distance generating functions is the entropy function

$$\omega(x) = \sum_{i=1}^n x_i \log x_i,$$

which is 1-strongly convex with respect to the l_1-norm over the standard simplex

$$\Delta := \left\{ x \in \mathbb{R}^n \;\middle|\; \sum_{i=1}^n x_i = 1,\ x \geq 0 \right\},$$

and its associated Bregman distance function is

$$D_\omega(x,y) = \sum_{i=1}^n y_i \log \frac{y_i}{x_i}.$$

The main motivation to use a generalized distance generating function, instead of the usual Euclidean distance function, is to design optimization algorithms that can take advantage of the geometry of the feasible set (see, *e.g.*, [7], [37]–[39]).

Remark 1: The strong convexity of the distance generating function ω always ensures that

$$D_\omega(x,y) \geq \frac{\mu_\omega}{2}\|y-x\|^2, \quad \forall x, y \in X,$$

and $D_\omega(x,y) = 0$ if and only if $x = y$.

Remark 2: Throughout the paper, there is no loss of generality to assume that $\mu_\omega = 1$. Indeed, if $\mu_\omega \neq 1$, we can choose the scaled function $\overline{\omega}(x) = \frac{1}{\mu_\omega}\omega(x)$, which has modulus $\overline{\mu}_\omega = 1$, to generate the Bregman distance function.

The following definition introduces *subgradients* of proper convex functions.

Definition 2: *For a convex function $\Psi : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$, a vector $s \in \mathbb{R}^n$ is called a subgradient of Ψ at $x \in \mathbb{R}^n$ if*

$$\Psi(y) \geq \Psi(x) + \langle s, y - x\rangle, \quad \forall y \in \mathbb{R}^n.$$

The set of all subgradients of Ψ at x is called the subdifferential of Ψ at x, and is denoted by $\partial\Psi(x)$.

III. Problem Setup

We consider stochastic convex optimization problems of the form

$$\underset{x}{\text{minimize}}\ \phi(x) := \mathbb{E}_\xi\big[F(x,\xi)\big] + \Psi(x). \tag{1}$$

Here, $x \in \mathbb{R}^n$ is the decision variable, ξ is a random vector whose probability distribution $\mathcal{P}$ is supported on a set $\Xi \subseteq \mathbb{R}^m$, $F(\cdot, \xi)$ is convex and differentiable for each $\xi \in \Xi$, and $\Psi(x)$ is a proper convex function that may be nonsmooth and extended real-valued. Let us define

$$f(x) := \mathbb{E}_\xi\big[F(x,\xi)\big] = \int_\Xi F(x,\xi) d\mathcal{P}(\xi). \tag{2}$$

Note that the expectation function f is convex, differentiable, and $\nabla f(x) = \mathbb{E}_\xi[\nabla_x F(x,\xi)]$ [40]. We use $X^\star$ to denote the set of optimal solutions of Problem (1) and $\phi^\star$ to denote the corresponding optimal value.

A difficulty when solving optimization problem (1) is that the distribution $\mathcal{P}$ is often unknown, so the expectation (2) cannot be computed. This situation occurs frequently in data-driven applications such as machine learning. To support these applications, we do not assume knowledge of f (or of $\mathcal{P}$), only access to a stochastic oracle. Each time the oracle is queried with an $x \in \mathbb{R}^n$, it generates an independent and identically distributed (i.i.d.) sample ξ from $\mathcal{P}$ and returns $\nabla_x F(x,\xi)$.

We also impose the following assumptions on Problem (1).

Assumption 1 (Existence of a minimum): The optimal set $X^\star$ is nonempty.

Assumption 2 (Lipschitz continuity of F): For each $\xi \in \Xi$, the function $F(\cdot,\xi)$ has Lipschitz continuous gradient with constant L. That is, for all $y, z \in \mathbb{R}^n$,

$$\|\nabla_x F(y,\xi) - \nabla_x F(z,\xi)\|_* \le L\|y - z\|.$$

Note that under Assumption 2, $\nabla f(x)$ is also Lipschitz continuous with the same constant L [9].

Assumption 3 (Bounded gradient variance): There exists a constant $\sigma \ge 0$ such that

$$\mathbb{E}_\xi\big[\|\nabla_x F(x,\xi) - \nabla f(x)\|_*^2\big] \le \sigma^2, \quad \forall x \in \mathbb{R}^n.$$

Assumption 4 (Closed effective domain of Ψ): The function Ψ is simple and lower semi-continuous, and its effective domain, $\operatorname{dom}\Psi = \{x \in \mathbb{R}^n \mid \Psi(x) < +\infty\}$, is closed.

Possible choices of Ψ include:

- *Unconstrained smooth minimization*: $\Psi(x) = 0$.
- *Constrained smooth minimization*: Ψ is the indicator function of a non-empty closed convex set $C \subseteq \mathbb{R}^n$, *i.e.*,
$$\Psi(x) = I_C(x) := \begin{cases} 0, & \text{if } x \in C, \\ +\infty, & \text{otherwise.} \end{cases}$$
- *l_1-regularized minimization*: $\Psi(x) = \lambda\|x\|_1$ with $\lambda > 0$.
- *Constrained l_1-regularized minimization*: In this case, $\Psi(x) = \lambda\|x\|_1 + I_C(x)$ with $\lambda > 0$.

Several practical problems in machine learning, statistical applications, and signal processing satisfy Assumptions 1–4 (see, *e.g.*, [2]–[4]). One such example is *l_1-regularized logistic regression* for sparse binary classification. We are then given a large number of observations

$$\big\{\xi_j = (a_j, b_j) \mid a_j \in \mathbb{R}^n,\ b_j \in \{-1,+1\},\ j = 1,\dots,m\big\},$$

drawn i.i.d. from an unknown distribution $\mathcal{P}$, and want to solve the minimization problem (1) with

$$F(x,\xi) = \log\bigl(1+\exp(-b\langle a,x\rangle\bigr),$$

and $\Psi(x) = \lambda\|x\|_1$. The role of l_1 regularization is to produce sparse solutions.

One approach for solving Problem (1) is the *serial mini-batch method* based on the mirror descent scheme [32]. Given a point $x \in \operatorname{dom}\Psi$, a single processor updates the decision variable x by sampling b i.i.d. random variables $\xi_1,\ldots,\xi_b$ from $\mathcal{P}$, computing the averaged stochastic gradient

$$g_{\text{ave}} = \frac{1}{b}\sum_{i=1}^{b}\nabla_x F(x,\xi_i),$$

and performing the composite mirror descent update

$$x \leftarrow \operatorname*{argmin}_z\left\{\langle g_{\text{ave}},z\rangle + \Psi(z) + \frac{1}{\gamma}D_\omega(x,z)\right\},$$

where γ is a positive step-size parameter. Under Assumptions 1–4 and choosing an appropriate step-size, this algorithm is guaranteed to converge to the optimum [32, Theorem 9]. However, in many emerging applications, such as large-scale machine learning and statistics, the size of dataset is so huge that it cannot fit on one machine. Hence, we need optimization algorithms that can be conveniently and efficiently executed in parallel on multiple processors.

IV. An Asynchronous Mini-Batch Algorithm

In this section, we will present an *asynchronous* mini-batch algorithm that exploits multiple processors to solve Problem (1). We characterize the iteration complexity and the convergence rate of the proposed algorithm, and show that these compare favourably with the state of the art.

A. Description of Algorithm

We assume p processors have access to a shared memory for the decision variable x. The processors may have different capabilities (in terms of processing power and access to data) and are able to update x without the need for coordination or synchronization. Conceptually, the algorithm lets each processor run its own stochastic composite mirror descent process, repeating the following steps:

1) Read x from the shared memory and load it into the local storage location $\widehat{x}$;
2) Sample b i.i.d random variables $\xi_1,\ldots,\xi_b$ from the distribution $\mathcal{P}$;
3) Compute the averaged stochastic gradient vector
$$\widehat{g}_{\text{ave}} = \frac{1}{b}\sum_{i=1}^{b}\nabla_x F(\widehat{x},\xi_i);$$
4) Update current x in the shared memory via
$$x \leftarrow \operatorname*{argmin}_z\left\{\langle \widehat{g}_{\text{ave}},z\rangle + \Psi(z) + \frac{1}{\gamma}D_\omega(x,z)\right\}.$$

The algorithm can be implemented in many ways as depicted in Figure 1. One way is to consider the p processors as peers that each execute the four-step algorithm independently of each other and only share the global memory for storing x. In this case, each processor reads the decision vector twice in each round: once in the first step (before evaluating the averaged gradient), and once in the last step (before carrying out the minimization). To ensure correctness, Step 4 must be an atomic operation, where the executing processor puts a write lock on the global memory until it has written back the result of the minimization (cf. Figure 1, left). The algorithm can also be executed in a master-worker setting. In this case, each of the worker nodes retrieves x from the master in Step 1 and returns the averaged gradient to the master in Step 3; the fourth step (carrying out the minimization) is executed by the master (cf. Figure 1, right)

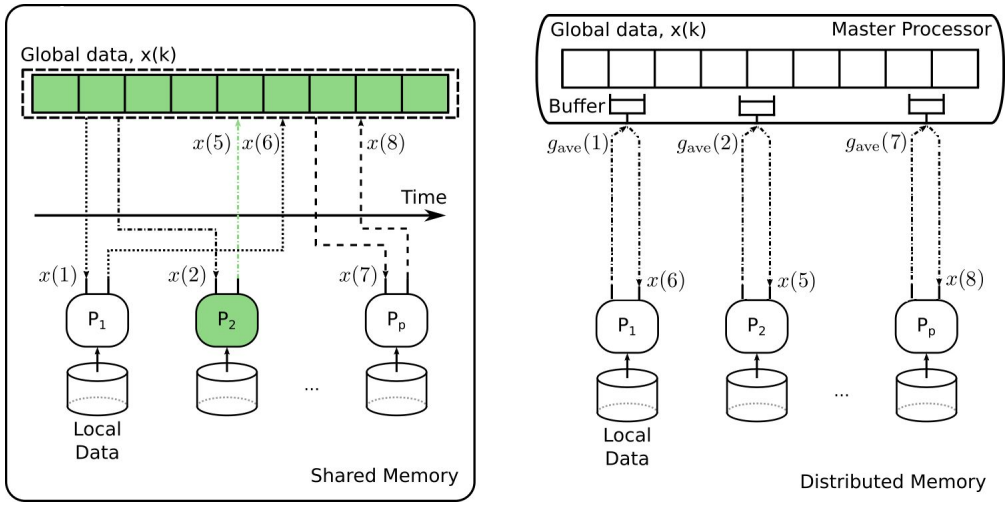

Fig. 1. Illustration of two conceptually different realizations of Algorithm 1: (1) a shared memory implementation (left); (2) a master-worker implementation (right). In the shared memory setting shown to the left, processor P_2 reads $x(2)$ from the shared memory and computes the averaged gradient vector $g_{\text{ave}}(2) = \frac{1}{b}\sum_{i=1}^{b} \nabla_x F\big(x(2), \xi_i\big)$. As the processors are being run without synchronization, $x(3)$ and $x(4)$ are written to the shared memory by other processors while P_2 is evaluating $g_{\text{ave}}(2)$. The figure shows a snapshot of the algorithm at time instance $k = 5$, at which the shared memory is locked by P_2 to read the current x, *i.e.* $x(4)$, to update it using the out-of-date gradient $g_{\text{ave}}(2)$, and write $x(5)$ to the memory. In the master-worker setting illustrated to the right, workers evaluate averaged gradient vectors in parallel and send their computations to buffers on the master processor, which is the sole entity with access to the global memory. The master performs an update using (possibly) out-of-date gradients and passes the updated decision vector x back to the workers.

Independently of how we choose to implement the algorithm, processors may work at different rates: while one processor updates the decision vector (in the shared memory setting) or send its averaged gradient to the master (in the master-worker setting), the others are generally busy computing averaged gradient vectors. The processors

Algorithm 1 Asynchronous Mini-batch Algorithm (running on each processor)

Inputs: positive step-sizes $\{\gamma(k)\}_{k\in\mathbb{N}_0}$; batch size $b \in \mathbb{N}$.

Initialization: $x(0) \in \operatorname{dom} \Psi$; $k = 0$.

repeat

receive inputs $\xi_1, \ldots, \xi_b$ sampled i.i.d. from distribution $\mathcal{P}$;

$$\begin{aligned} &g_{\text{ave}}\big(d(k)\big) \leftarrow \frac{1}{b}\sum_{i=1}^{b} \nabla_x F\big(x(d(k)), \xi_i\big); \\ &x(k+1) \leftarrow \operatorname*{argmin}_z \left\{ \big\langle g_{\text{ave}}\big(d(k)\big), z\big\rangle + \Psi(z) + \frac{1}{\gamma(k)} D_\omega(x(k), z) \right\} \\ &k \leftarrow k+1; \end{aligned} \tag{3}$$

until termination test satisfied

that perform gradient evaluations do not need to be aware of updates to the decision vector, but can continue to operate on stale information about x. Therefore, unlike *synchronous* parallel mini-batch algorithms [32], there is no need for processors to wait for each other to finish the gradient computations. Moreover, the value $\widehat{x}$ at which the average of gradients is evaluated by a processor may differ from the value of x to which the update is applied.

Algorithm 1 describes the p asynchronous processes that run in parallel. To describe the progress of the overall optimization process, we introduce a counter k that is incremented each time x is updated. We let $d(k)$ denote the time at which $\widehat{x}$ used to compute the averaged gradient involved in the update of $x(k)$ was read from the shared memory. It is clear that $0 \leq d(k) \leq k$ for all $k \in \mathbb{N}_0$. The value

$$\tau(k) := k - d(k)$$

can be viewed as the delay between reading and updating for processors and captures the staleness of the information used to compute the average of gradients for the k-th update. We assume that the delay is not too long, *i.e.*, there is a nonnegative integer $\tau_{\max}$ such that

$$0 \leq \tau(k) \leq \tau_{\max}.$$

The value of $\tau_{\max}$ is an indicator of the asynchronism in the algorithm and in the execution platform. In practice, $\tau_{\max}$ will depend on the number of parallel processors used in the algorithm [33]–[35]. Note that the cyclic-delay mini-batch algorithm [36], in which the processors are ordered and each updates the decision variable under a fixed schedule, is a special case of Algorithm 1 where $d(k) = k - p + 1$, or, equivalently, $\tau(k) = p - 1$ for all k.

B. Convergence Rate for General Convex Regularization

The following theorem establishes convergence properties of Algorithm 1 when a constant step-size is used.

Theorem 1: *Let Assumptions 1–4 hold. Assume also that for all $k \in \mathbb{N}_0$,*

$$\gamma(k) = \gamma \in \left(0,\ \frac{1}{L(\tau_{\max}+1)^2}\right). \tag{4}$$

Then, for every $T \in \mathbb{N}$ and any optimizer $x^\star$ of (1)*, we have*

$$\mathbb{E}\big[\phi\big(x_{\text{ave}}(T)\big)\big] - \phi^\star \leq \frac{D_\omega\big(x(0), x^\star\big)}{\gamma T} + \frac{\gamma c \sigma^2}{2b\big(1 - \gamma L(\tau_{\max}+1)^2\big)},$$

where $x_{\text{ave}}(T)$ is the Cesáro average of the iterates, i.e.,

$$x_{\text{ave}}(T) := \frac{1}{T}\sum_{k=1}^{T} x(k).$$

Furthermore, b is the batch size, the expectation is taken with respect to all random variables $\{\xi_i(k) \mid i = 1, \ldots, b,\ k = 0, \ldots, T-1\}$, and $c \in [1, b]$ is given by

$$c = \begin{cases} 1, & \text{if } \|\cdot\|_* = \|\cdot\|_2, \\ 2\max_{\|x\|\leq 1} \omega(x), & \text{otherwise.} \end{cases}$$

Proof: See Appendix A. ∎

Theorem 1 demonstrates that for any constant step-size γ satisfying (4), the running average of iterates generated by Algorithm 1 will converge in expectation to a ball around the optimum at a rate of $\mathcal{O}(1/T)$. The convergence rate and the residual error depend on the choice of γ: decreasing γ reduces the residual error, but it also results in a slower convergence. We now describe a possible strategy for selecting the constant step-size. Let T_ϵ be the total number of iterations necessary to achieve ϵ-optimal solution to Problem (1), that is, $\mathbb{E}\big[\phi\big(x_{\text{ave}}(T)\big)\big] - \phi^\star \leq \epsilon$ when $T \geq T_\epsilon$. If we pick

$$\gamma = \frac{\epsilon}{L\epsilon(\tau_{\max}+1)^2 + c\sigma^2/b}, \tag{5}$$

it follows from Theorem 1 that the corresponding $x_{\text{ave}}(T)$ satisfies

$$\mathbb{E}\big[\phi\big(x_{\text{ave}}(T)\big)\big] - \phi^\star \leq \frac{\epsilon_0}{T}\left(L(\tau_{\max}+1)^2 + \frac{c\sigma^2}{b\epsilon}\right) + \frac{\epsilon}{2},$$

where $\epsilon_0 = D_\omega\big(x(0), x^\star\big)$. This inequality tells us that if the first term on the right-hand side is less than $\epsilon/2$, *i.e.*, if

$$T \geq T_\epsilon := 2\epsilon_0\left(\frac{L(\tau_{\max}+1)^2}{\epsilon} + \frac{c\sigma^2}{b\epsilon^2}\right),$$

then $\mathbb{E}\big[\phi\big(x_{\text{ave}}(T)\big)\big] - \phi^\star \leq \epsilon$. Hence, the iteration complexity of Algorithm 1 with the step-size choice (5) is given by

$$\mathcal{O}\left(\frac{L(\tau_{\max}+1)^2}{\epsilon} + \frac{c\sigma^2}{b\epsilon^2}\right). \tag{6}$$

As long as the maximum delay bound $\tau_{\max}$ is of the order $1/\sqrt{\epsilon}$, the first term in (6) is asymptotically negligible, and hence the iteration complexity of Algorithm 1 is asymptotically $\mathcal{O}(c\sigma^2/b\epsilon^2)$, which is exactly the iteration complexity achieved by the mini-batch algorithm for solving stochastic convex optimization problems in a serial setting [32]. As discussed before, $\tau_{\max}$ is related to the number of processors used in the algorithm. Therefore, if the number of processors is of the order of $\mathcal{O}(1/\sqrt{\epsilon})$, parallelization does not appreciably degrade asymptotic convergence of Algorithm 1. Furthermore, as p processors are being run in parallel, updates occur roughly p times

as quickly and in time scaling as T/p, the processors may compute T averaged gradient vectors (instead of T/p vectors). This means that the near-linear speedup in the number of processors can be expected.

Remark 3: Another strategy for the selection of the constant step-size in Algorithm 1 is to use γ that depends on the prior knowledge of the number of iterations to be performed. More precisely, assume that the number of iterations is fixed in advance, say equal to T_F. By choosing γ as

$$\gamma = \frac{1}{L(\tau_{\max}+1)^2 + \alpha\sqrt{T_F}},$$

for some $\alpha > 0$, it follows from Theorem 1 that the running average of the iterates after T_F iterations satisfies

$$\mathbb{E}\big[\phi\big(x_{\text{ave}}(T_F)\big)\big] - \phi^\star \leq \frac{L(\tau_{\max}+1)^2 D_\omega\big(x(0),x^\star\big)}{T_F} + \frac{1}{\sqrt{T_F}}\left(\alpha D_\omega\big(x(0),x^\star\big) + \frac{c\sigma^2}{2\alpha b}\right).$$

It is easy to verify that the optimal choice of α, which minimizes the second term on the right-hand-side of the above inequality, is

$$\alpha^\star = \frac{\sigma\sqrt{c}}{\sqrt{2bD_\omega\big(x(0),x^\star\big)}}.$$

With this choice of α, we then have

$$\mathbb{E}\big[\phi\big(x_{\text{ave}}(T_F)\big)\big] - \phi^\star \leq \frac{L(\tau_{\max}+1)^2 D_\omega\big(x(0),x^\star\big)}{T_F} + \frac{\sigma\sqrt{2cD_\omega\big(x(0),x^\star\big)}}{\sqrt{bT_F}}.$$

In the case that $\tau_{\max} = 0$, the preceding guaranteed bound reduces to the one obtained in [8, Theorem 1] for the serial stochastic mirror descent algorithm with constant step-sizes. Note that in order to implement Algorithm 1 with the optimal constant step-size policy, we need to estimate an upper bound on $D_\omega\big(x(0),x^\star\big)$, since $D_\omega\big(x(0),x^\star\big)$ is usually unknown.

The following theorem characterizes the convergence of Algorithm 1 with a time-varying step-size sequence when dom Ψ is *bounded* in addition to being closed and convex.

Theorem 2: *Suppose that Assumptions 1–4 hold. In addition, suppose that* dom Ψ *is compact and that* $D_\omega(\cdot,\cdot)$ *is bounded on* dom Ψ. *Let*

$$R^2 = \max_{x,y\in\text{dom }\Psi} D_\omega(x,y).$$

If $\{\gamma(k)\}_{k\in\mathbb{N}_0}$ *is set to* $\gamma(k)^{-1} = L(\tau_{\max}+1)^2 + \alpha(k)$ *with*

$$\alpha(k) = \frac{\sigma\sqrt{c}\sqrt{k+1}}{R\sqrt{b}},$$

then the Cesáro average of the iterates generated by Algorithm 1 satisfies

$$\mathbb{E}\big[\phi\big(x_{\text{ave}}(T)\big)\big] - \phi^\star \leq \frac{LR^2(\tau_{\max}+1)^2}{T} + \frac{2\sigma R\sqrt{c}}{\sqrt{bT}},$$

for all $T \in \mathbb{N}$.

Proof: See Appendix B. ■

The time-varying step-size $\gamma(k)$, which ensures the convergence of the algorithm, consists of two terms: the time-varying term $\eta(k)$ should control the errors from stochastic gradient information while the role of the constant

term $(L(\tau_{\max}+1)^2)$ is to decrease the effects of asynchrony (bounded delays) on the convergence of the algorithm. According to Theorem 2, in the case that $\tau_{\max}=\mathcal{O}(T^{1/4})$, the delay becomes increasingly harmless as the algorithm progresses and the expected function value evaluated at $x_{\text{ave}}(T)$ converges asymptotically at a rate $\mathcal{O}(1/\sqrt{T})$, which is known to be the best achievable rate of the mirror descent method for nonsmooth stochastic convex optimization problems [7].

For the special case of the optimization problem (1) where Ψ is restricted to be the indicator function of a compact convex set, Agarwal and Duchi [36, Theorem 2] showed that the convergence rate of the delayed stochastic mirror descent method with time-varying step-size is

$$\mathcal{O}\left(\frac{LR^2+RG\tau_{\max}}{T}+\frac{\sigma R\sqrt{c}}{\sqrt{Tb}}+\frac{LR^2G^2\tau_{\max}^2 b\log T}{c\sigma^2 T}\right),$$

where G is the maximum bound on $\sqrt{\mathbb{E}[\|\nabla_x F(x,\xi)\|_*^2]}$. Comparing with this result, instead of a asymptotic penalty of the form $\mathcal{O}(\tau_{\max}^2\log T/T)$ due to the delays, we have the penalty $\mathcal{O}(\tau_{\max}^2/T)$, which is much smaller for large T. Therefore, not only do we extend the result of [36] to general regularization functions, but we also obtain a sharper guaranteed convergence rate than the one presented in [36].

C. Convergence Rate for Strongly Convex Regularization

In this subsection, we restrict our attention to stochastic composite optimization problems with strongly convex regularization terms. Specifically, we assume that Ψ is μ_Ψ-strongly convex with respect to $\|\cdot\|$, that is, for any $x,y\in\text{dom }\Psi$,

$$\Psi(y)\geq\Psi(x)+\langle s,y-x\rangle+\frac{\mu_\Psi}{2}\|y-x\|^2,\quad\forall s\in\partial\Psi(x).$$

Examples of the strongly convex function Ψ include:

- *l_2-regularization*: $\Psi(x)=(\rho/2)\|x\|_2^2$ with $\rho>0$.
- *Elastic net regularization*: $\Psi(x)=\lambda\|x\|_1+(\rho/2)\|x\|_2^2$ with $\lambda>0$ and $\rho>0$.

Remark 4: The strong convexity of Ψ implies that Problem (1) has a unique minimizer $x^\star$ [41, Corollary 11.16].

In order to derive the convergence rate of Algorithm 1 for solving (1) with a strongly convex regularization term, we need to assume that the Bregman distance function $D(x,y)$ used in the algorithm satisfies the next assumption.

Assumption 5 (Quadratic growth condition): For all $x,y\in\text{dom }\Psi$, we have

$$D_\omega(x,y)\leq\frac{Q}{2}\|x-y\|^2,$$

with $Q\geq\mu_\omega$.

For example, if $\omega(x)=\frac{1}{2}\|x\|_2^2$, then $D_\omega(x,y)=\frac{1}{2}\|x-y\|_2^2$ and $Q=1$. Note that Assumption 5 will automatically hold when the distance generating function ω has Lipschitz continuous gradient with a constant Q [12].

The associated convergence result now reads as follows.

Theorem 3: *Suppose that the regularization function Ψ is μ_Ψ-strongly convex and that Assumptions 2–5 hold. If $\{\gamma(k)\}_{k\in\mathbb{N}_0}$ is set to $\gamma(k)^{-1} = 2L(\tau_{\max}+1)^2 + \beta(k)$ with*

$$\beta(k) = \frac{\mu_\Psi}{3Q}\big(k+\tau_{\max}+1\big),$$

then the iterates produced by Algorithm 1 satisfies

$$\mathbb{E}\big[\|x(T)-x^\star\|^2\big] \leq \frac{2\left(\frac{6LQ}{\mu_\Psi}+1\right)^2(\tau_{\max}+1)^4}{(T+1)^2} D_\omega\big(x(0),x^\star\big) + \frac{18c\sigma^2Q^2}{b\mu_\Psi^2(T+1)},$$

for all $T\in\mathbb{N}$.

Proof: See Appendix C. ∎

An interesting point regarding Theorem 3 is that for solving stochastic composite optimization problems with strongly convex regularization functions, the maximum delay bound $\tau_{\max}$ can be as large as $\mathcal{O}(T^{1/4})$ without affecting the asymptotic convergence rate of Algorithm 1. In this case, our asynchronous mini-batch algorithm converges asymptotically at a rate of $\mathcal{O}(1/T)$, which matches the best known rate achievable in a serial setting.

V. Experimental Results

We have developed a complete master-worker implementation of our algorithm in C/++ using the Massage Passing Interface libraries (OpenMPI). Although we argued in Section IV that Algorithm 1 can be implemented using atomic operations on shared-memory computing architectures, we have chosen the MPI implementation due to its flexibility in scaling the problem to distributed-memory environments.

We evaluated our algorithm on a document classification problem using the text categorization dataset `rcv1` [42]. This dataset consists of $m\approx 800000$ documents, with $n\approx 50000$ unique stemmed tokens spanning 103 topics. Out of these topics, we decided to classify sports-related documents. To this end, we trained a sparse (binary) classifier by solving the following l_1-regularized logistic regression problem

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & \mathbb{E}_{\{(a_i,b_i)\}}\left[\log\left(1+\exp\left(-b_i\langle a_i,x\rangle\right)\right)\right] + \lambda\|x\|_1 \\ \text{subject to} \quad & \|x\|_2 \leq R\,. \end{aligned}$$

Here, $a_i\in\mathbb{R}^n$ is the sparse vector of token weights assigned to each document, and $b_i\in\{-1,1\}$ indicates whether a selected document is sports-related, or not (b_i is 1 if the document is about sport, -1 otherwise). To evaluate scalability, we used both the training and test sets available when solving the optimization problem. We implemented Algorithm 1 with time-varying step-sizes, and used a batch size of 1000 documents. The regularization parameter was set to $\lambda = 0.01$, and the algorithm was run until a fixed tolerance ϵ was met.

Figure 2 presents the achieved relative speedup of the algorithm with respect to the number of workers used. The relative speedup of the algorithm on p processors is defined as $S(p) = t_1/t_p$, where t_1 and t_p are the time it takes to run the corresponding algorithm (to ϵ-accuracy) on 1 and p processing units, respectively. We observe a near-linear relative speedup, consistent with our theoretical results. The timings are averaged over 10 Monte Carlo runs.

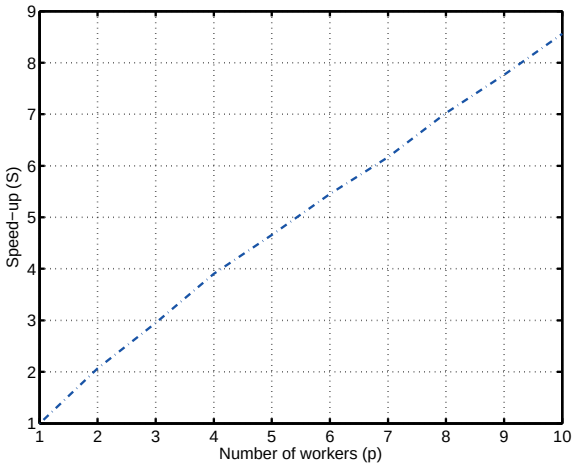

Fig. 2. Speedup of Algorithm 1 with respect to the number of workers.

VI. Conclusions

We have proposed an asynchronous mini-batch algorithm that exploits multiple processors to solve regularized stochastic optimization problems with smooth loss functions. We have established that for closed and convex constraint sets, the iteration complexity of the algorithm with constant step-sizes is asymptotically $\mathcal{O}(1/\epsilon^2)$. For compact constraint sets, we have proved that the running average of the iterates generated by our algorithm with time-varying step-size converges to the optimum at a rate $\mathcal{O}(1/\sqrt{T})$. When the regularization function is strongly convex and the constraint set is closed and convex, the algorithm achieves the rate of the order $\mathcal{O}(1/T)$. We have shown that the penalty in convergence rate of the algorithm due to asynchrony is asymptotically negligible and a near-linear speedup in the number of processors can be expected. Our computational experience confirmed the theory.

Appendix

In this section, we prove the main results of the paper, namely, Theorems 1–3. We first state three key lemmas which are instrumental in our argument.

The following result establishes an important recursion for the iterates generated by Algorithm 1.

Lemma 1: Suppose Assumptions 1–4 hold. Then, the iterates $\{x(k)\}_{k\in\mathbb{N}_0}$ generated by Algorithm 1 satisfy

$$\begin{aligned}\phi\big(x(k+1)\big) - \phi^\star + \frac{1}{\gamma(k)} D_\omega\big(x(k+1), x^\star\big) \leq{}& \frac{1}{2\eta(k)}\big\|e\big(d(k)\big)\big\|_*^2 \\ &+ \big\langle e\big(d(k)\big), x(k) - x^\star\big\rangle + \frac{1}{\gamma(k)} D_\omega\big(x(k), x^\star\big) \\ &+ \frac{L(\tau_{\max}+1)}{2} \sum_{j=0}^{\tau_{\max}} \big\|x(k-j) - x(k-j+1)\big\|^2 \\ &- \frac{1}{2}\left(\frac{1}{\gamma(k)} - \eta(k)\right) \|x(k+1) - x(k)\|^2 \\ &- \frac{\mu_\Psi}{2}\big\|x(k+1) - x^\star)\big\|^2, \end{aligned} \tag{7}$$

where $x^\star \in X^\star$, $\{\eta(k)\}$ is a sequence of strictly positive numbers, and $e(k) := \nabla f(x(k)) - g_{\text{ave}}(k)$ is the error in the gradient estimate.

Proof: We start with the first-order optimality condition for the point $x(k+1)$ in the minimization problem (3): there exists subgradient $s(k+1) \in \partial\Psi(x(k+1))$ such that for all $z \in \text{dom}\,\Psi$, we have

$$\Big\langle g_{\text{ave}}\big(d(k)\big) + s(k+1) + \frac{1}{\gamma(k)}\nabla_{(2)} D_\omega\big(x(k), x(k+1)\big), z - x(k+1)\Big\rangle \geq 0,$$

where $\nabla_{(2)} D_\omega(\cdot,\cdot)$ denotes the partial derivative of the Bregman distance function with respect to the second variable. Plugging the following equality

$$\nabla_{(2)} D_\omega\big(x(k), x(k+1)\big) = \nabla\omega\big(x(k+1)\big) - \nabla\omega\big(x(k)\big),$$

into the previous inequality and re-arranging terms gives

$$\begin{aligned}
\frac{1}{\gamma(k)}\Big\langle \nabla\omega\big(x(k)\big) - \nabla\omega\big(x(k+1)\big), z - x(k+1)\Big\rangle &\leq \Big\langle g_{\text{ave}}\big(d(k)\big) + s(k+1), z - x(k+1)\Big\rangle \\
&= \Big\langle g_{\text{ave}}\big(d(k)\big), z - x(k+1)\Big\rangle \\
&\quad + \Big\langle s(k+1), z - x(k+1)\Big\rangle \\
&\leq \Big\langle g_{\text{ave}}\big(d(k)\big), z - x(k+1)\Big\rangle \\
&\quad + \Psi(z) - \Psi(x(k+1)) - \frac{\mu_\Psi}{2}\big\|z - x(k+1)\big\|^2, \qquad (8)
\end{aligned}$$

where the last inequality used

$$\Psi(z) \geq \Psi\big(x(k+1)\big) + \big\langle s(k+1), z - x(k+1)\big\rangle + \frac{\mu_\Psi}{2}\big\|z - x(k+1)\big\|^2,$$

by the (strong) convexity of Ψ. We now use the following well-known *three point identity* of the Bregman distance function [43] to rewrite the left-hand side of (8):

$$\big\langle \nabla\omega(a) - \nabla\omega(b), c - b\big\rangle = D_\omega(a,b) - D_\omega(a,c) + D_\omega(b,c).$$

From this relation, with $a = x(k)$, $b = x(k+1)$, and $c = z$, we have

$$\Big\langle \nabla\omega\big(x(k)\big) - \nabla\omega\big(x(k+1)\big), z - x(k+1)\Big\rangle = D_\omega\big(x(k), x(k+1)\big) - D_\omega\big(x(k), z\big) + D_\omega\big(x(k+1), z\big).$$

Substituting the preceding equality into (8) and re-arranging terms result in

$$\begin{aligned}
\Psi(x(k+1)) - \Psi(z) + \frac{1}{\gamma(k)} D_\omega\big(x(k+1), z\big) &\leq \Big\langle g_{\text{ave}}\big(d(k)\big), z - x(k+1)\Big\rangle + \frac{1}{\gamma(k)} D_\omega\big(x(k), z\big) \\
&\quad - \frac{1}{\gamma(k)} D_\omega\big(x(k), x(k+1)\big) - \frac{\mu_\Psi}{2}\big\|z - x(k+1)\big\|^2.
\end{aligned}$$

Since the distance generating function $\omega(x)$ is 1-strongly convex, we have the lower bound

$$D_\omega\big(x(k), x(k+1)\big) \geq \frac{1}{2}\|x(k+1) - x(k)\|^2,$$

which implies that

$$\begin{aligned}
\Psi(x(k+1)) - \Psi(z) + \frac{1}{\gamma(k)} D_\omega\big(x(k+1), z\big) &\leq \Big\langle g_{\text{ave}}\big(d(k)\big), z - x(k+1)\Big\rangle + \frac{1}{\gamma(k)} D_\omega\big(x(k), z\big) \\
&\quad - \frac{1}{2\gamma(k)}\|x(k+1) - x(k)\|^2 - \frac{\mu_\Psi}{2}\big\|z - x(k+1)\big\|^2. \qquad (9)
\end{aligned}$$

The essential idea in the rest of the proof is to use convexity and smoothness of the expectation function f to bound $f(x(k+1)-f(z)$ for each $z \in \operatorname{dom} \Psi$ and each $k \in \mathbb{N}_0$. According to Assumption 2, $\nabla F(x,\xi)$ and, hence, $\nabla f(x)$ are Lipschitz continuous with the constant L. By using the L-Lipschitz continuity of ∇f and then the convexity of f, we have

$$\begin{aligned} f(x(k+1)) &\leq f(x(d(k))) + \langle \nabla f(x(d(k))), x(k+1) - x(d(k)) \rangle + \frac{L}{2}\|x(k+1) - x(d(k))\|^2 \\ &\leq f\big(z\big) + \big\langle \nabla f\big(x(d(k))\big), x(k+1) - z \big\rangle + \frac{L}{2}\big\|x(k+1) - x(d(k))\big\|^2, \end{aligned} \tag{10}$$

for any $z \in \operatorname{dom} \Psi$. Combining inequalities (9) and (10), and recalling that $\phi(x) = f(x) + \Psi(x)$, we obtain

$$\begin{aligned} \phi(x(k+1)) - \phi(z) + \frac{1}{\gamma(k)} D_\omega\big(x(k+1), z\big) \leq \big\langle \nabla f\big(x(d(k))\big) - g_{\text{ave}}\big(d(k)\big), x(k+1) - z \big\rangle + \frac{1}{\gamma(k)} D_\omega\big(x(k), z\big) \\ - \frac{1}{2\gamma(k)}\|x(k+1) - x(k)\|^2 - \frac{\mu_\Psi}{2}\big\|z - x(k+1)\big\|^2 \\ + \frac{L}{2}\|x(k+1) - x(d(k))\|^2. \end{aligned}$$

We now rewrite the above inequality in terms of the error $e(d(k)) = \nabla f(x(d(k))) - g_{\text{ave}}(d(k))$ as follows:

$$\begin{aligned} \phi(x(k+1)) - \phi(z) + \frac{1}{\gamma(k)} D_\omega\big(x(k+1), z\big) &\leq \big\langle e\big(d(k)\big), x(k+1) - z \big\rangle + \frac{1}{\gamma(k)} D_\omega\big(x(k), z\big) \\ &\quad - \frac{1}{2\gamma(k)}\|x(k+1) - x(k)\|^2 - \frac{\mu_\Psi}{2}\big\|z - x(k+1)\big\|^2 \\ &\quad + \frac{L}{2}\|x(k+1) - x(d(k))\|^2 \\ &= \underbrace{\big\langle e\big(d(k)\big), x(k+1) - x(k) \big\rangle}_{U_1} \\ &\quad + \big\langle e\big(d(k)\big), x(k) - z \big\rangle + \frac{1}{\gamma(k)} D_\omega\big(x(k), z\big) \\ &\quad - \frac{1}{2\gamma(k)}\|x(k+1) - x(k)\|^2 - \frac{\mu_\Psi}{2}\big\|z - x(k+1)\big\|^2 \\ &\quad + \frac{L}{2}\underbrace{\|x(k+1) - x(d(k))\|^2}_{U_2}. \end{aligned} \tag{11}$$

We will seek upper bounds on the quantities U_1 and U_2. Let $\{\eta(k)\}_{k\in\mathbb{N}_0}$ be a sequence of positive numbers. For U_1, we have

$$\begin{aligned} U_1 &\leq \left| \left\langle \frac{1}{\sqrt{\eta(k)}} e\big(d(k)\big), \sqrt{\eta(k)}\big(x(k+1) - x(k)\big) \right\rangle \right| \\ &\leq \frac{1}{2\eta(k)}\big\|e\big(d(k)\big)\big\|_*^2 + \frac{\eta(k)}{2}\big\|x(k+1) - x(k)\big\|^2, \end{aligned} \tag{12}$$

where the second inequality follows from the Fenchel-Young inequality applied to the conjugate pair $\frac{1}{2}\|\cdot\|^2$ and $\frac{1}{2}\|\cdot\|_*^2$, *i.e.*,

$$\big|\langle a, b\rangle\big| \leq \frac{1}{2}\big\|a\big\|_*^2 + \frac{1}{2}\big\|b\big\|^2.$$

We now turn to U_2. It follows from definition $\tau(k) = k - d(k)$ that

$$U_2 = \big(k - d(k) + 1\big)^2 \left\| \sum_{j=0}^{k-d(k)} \frac{x(k-j) - x(k-j+1)}{k - d(k) + 1} \right\|^2$$
$$= \big(\tau(k) + 1\big)^2 \left\| \sum_{j=0}^{\tau(k)} \frac{x(k-j) - x(k-j+1)}{\tau(k) + 1} \right\|^2 .$$

Then, by the convexity of the norm $\|\cdot\|$, we conclude that

$$U_2 \leq \big(\tau(k) + 1\big) \sum_{j=0}^{\tau(k)} \big\|x(k-j) - x(k-j+1)\big\|^2$$
$$\leq \big(\tau_{\max} + 1\big) \sum_{j=0}^{\tau_{\max}} \big\|x(k-j) - x(k-j+1)\big\|^2, \tag{13}$$

where the last inequality comes from our assumption that $\tau(k) \leq \tau_{\max}$ for all $k \in \mathbb{N}_0$. Substituting inequalities (12) and (13) into the bound (11) and simplifying yield

$$\begin{aligned}\phi(x(k+1)) - \phi(z) + \frac{1}{\gamma(k)} D_\omega\big(x(k+1), z\big) \leq{}& \frac{1}{2\eta(k)} \big\|e\big(d(k)\big)\big\|_*^2 \\ &+ \big\langle e\big(d(k)\big), x(k) - z\big\rangle + \frac{1}{\gamma(k)} D_\omega\big(x(k), z\big) \\ &+ \frac{L(\tau_{\max}+1)}{2} \sum_{j=0}^{\tau_{\max}} \big\|x(k-j) - x(k-j+1)\big\|^2 \\ &- \frac{1}{2}\left(\frac{1}{\gamma(k)} - \eta(k)\right) \|x(k+1) - x(k)\|^2 \\ &- \frac{\mu_\Psi}{2} \big\|z - x(k+1)\big\|^2.\end{aligned}$$

Setting $z = x^\star$, where $x^\star \in X^\star$, completes the proof. ■

The next result follows from Lemma 1 by taking summation of the relations in (7).

Lemma 2: Let Assumptions 1–4 hold. Assume also that $\{\gamma(k)\}_{k\in\mathbb{N}_0}$ is set to

$$\gamma(k) = \frac{1}{\eta(k) + L(\tau_{\max}+1)^2}, \quad k \in \mathbb{N}_0,$$

where $\eta(k)$ is positive for all k. Then, the iterates $\{x(k)\}_{k\in\mathbb{N}_0}$ produced by Algorithm 1 satisfy

$$\begin{aligned}\sum_{k=0}^{T-1} \big(\phi(x(k+1)) - \phi^\star\big) \leq{}& \sum_{k=0}^{T-1} \frac{1}{2\eta(k)} \big\|e\big(d(k)\big)\big\|_*^2 \\ &+ \sum_{k=0}^{T-1} \big\langle e\big(d(k)\big), x(k) - x^\star\big\rangle + \frac{1}{\gamma(0)} D_\omega\big(x(0), x^\star\big) \\ &+ \sum_{k=0}^{T-1} \left(\frac{1}{\gamma(k+1)} - \frac{1}{\gamma(k)}\right) D_\omega\big(x(k+1), x^\star\big) \\ &- \frac{\mu_\Psi}{2} \sum_{k=0}^{T-1} \big\|x(k+1) - x^\star\big\|^2,\end{aligned}$$

for all $T \in \mathbb{N}$.

Proof: Applying Lemma 1 with

$$\eta(k) = \frac{1}{\gamma(k)} - L(\tau_{\max}+1)^2,$$

adding and subtracting $\gamma(k+1)^{-1} D_\omega\big(x(k+1), x^\star\big)$ to the left-hand side of (7), and re-arranging terms, we obtain

$$\begin{aligned}
\phi(x(k+1)) - \phi^\star + \frac{1}{\gamma(k+1)} D_\omega\big(x(k+1), x^\star\big) \leq{}& \frac{1}{2\eta(k)} \big\|e\big(d(k)\big)\big\|_*^2 \\
&+ \big\langle e\big(d(k)\big), x(k) - x^\star \big\rangle + \frac{1}{\gamma(k)} D_\omega\big(x(k), x^\star\big) \\
&+ \left(\frac{1}{\gamma(k+1)} - \frac{1}{\gamma(k)}\right) D_\omega\big(x(k+1), x^\star\big) \\
&+ \frac{L(\tau_{\max}+1)}{2} \sum_{j=0}^{\tau_{\max}} \big\|x(k-j) - x(k-j+1)\big\|^2 \\
&- \frac{L(\tau_{\max}+1)^2}{2} \|x(k+1) - x(k)\|^2 \\
&- \frac{\mu_\Psi}{2} \big\|x(k+1) - x^\star\big\|^2.
\end{aligned}$$

Summing the preceding inequality over $k = 0, \ldots, T-1$, $T \in \mathbb{N}$, yields

$$\begin{aligned}
\sum_{k=0}^{T-1} \big(\phi(x(k+1)) - \phi^\star\big) + \frac{1}{\gamma(T)} D_\omega\big(x(T), x^\star\big) \leq{}& \sum_{k=0}^{T-1} \frac{1}{2\eta(k)} \big\|e\big(d(k)\big)\big\|_*^2 \\
&+ \sum_{k=0}^{T-1} \big\langle e\big(d(k)\big), x(k) - x^\star \big\rangle + \frac{1}{\gamma(0)} D_\omega\big(x(0), x^\star\big) \\
&+ \sum_{k=0}^{T-1} \left(\frac{1}{\gamma(k+1)} - \frac{1}{\gamma(k)}\right) D_\omega\big(x(k+1), x^\star\big) \\
&+ \frac{L(\tau_{\max}+1)}{2} \sum_{k=0}^{T-1} \sum_{j=0}^{\tau_{\max}} \big\|x(k-j) - x(k-j+1)\big\|^2 \\
&- \frac{L(\tau_{\max}+1)^2}{2} \sum_{k=0}^{T-1} \|x(k+1) - x(k)\|^2 \\
&- \frac{\mu_\Psi}{2} \sum_{k=0}^{T-1} \big\|x(k+1) - x^\star\big\|^2 \\
\leq{}& \sum_{k=0}^{T-1} \frac{1}{2\eta(k)} \big\|e\big(d(k)\big)\big\|_*^2 \\
&+ \sum_{k=0}^{T-1} \big\langle e\big(d(k)\big), x(k) - x^\star \big\rangle + \frac{1}{\gamma(0)} D_\omega\big(x(0), x^\star\big) \\
&+ \sum_{k=0}^{T-1} \left(\frac{1}{\gamma(k+1)} - \frac{1}{\gamma(k)}\right) D_\omega\big(x(k+1), x^\star\big) \\
&- \frac{\mu_\Psi}{2} \sum_{k=0}^{T-1} \big\|x(k+1) - x^\star\big\|^2, \qquad (14)
\end{aligned}$$

where the second inequality used the facts

$$\begin{aligned}\sum_{k=0}^{T-1}\sum_{j=0}^{\tau_{\max}}\big\|x(k-j)-x(k-j+1)\big\|^2 &= \sum_{j=0}^{\tau_{\max}}\sum_{k=-j}^{T-j-1}\big\|x(k)-x(k+1)\big\|^2\\ &= \sum_{j=0}^{\tau_{\max}}\sum_{k=0}^{T-j-1}\big\|x(k)-x(k+1)\big\|^2\\ &\leq \sum_{j=0}^{\tau_{\max}}\sum_{k=0}^{T-1}\big\|x(k)-x(k+1)\big\|^2\\ &\leq (\tau_{\max}+1)\sum_{k=0}^{T-1}\big\|x(k)-x(k+1)\big\|^2,\end{aligned}$$

and $x(k)=x(0)$ for all $k\leq 0$. Dropping the second term on the left-hand side of (14) concludes the proof. ■

Lemma 3: Let $\|\cdot\|$ be a norm over $\mathbb{R}^n$ and let $\|\cdot\|_\star$ be its dual norm. Let ω be a 1-strongly convex function with respect to $\|\cdot\|$ over $\mathbb{R}^n$. If $y_1,\ldots,y_b\in\mathbb{R}^n$ are mean zero random variables drawn i.i.d. from a distribution $\mathcal{P}$, then

$$\mathbb{E}\left[\left\|\frac{1}{b}\sum_{i=1}^{b}y_i\right\|_*^2\right]\leq\frac{c}{b^2}\sum_{i=1}^{b}\mathbb{E}\left[\|y_i\|_*^2\right],$$

where $c\in[1,b]$ is given by

$$c=\begin{cases}1, & \text{if } \|\cdot\|_*=\|\cdot\|_2,\\ 2\max_{\|x\|=1}\omega(x), & \text{otherwise.}\end{cases}$$

Proof: The result follows from [44, Lemma B.2] and convexity of the norm $\|\cdot\|_*$. For further details, see [32, §4.1]. ■

A. Proof of Theorem 1

Assume that the step-size $\{\gamma(k)\}_{k\in\mathbb{N}_0}$ is set to

$$\gamma(k)=\gamma=\frac{1}{\eta+L(\tau_{\max}+1)^2},$$

for some $\eta>0$. It is clear that γ satisfies (4). Applying Lemma 2 with $\mu_\Psi=0$, $\gamma(k)=\gamma$ and $\eta(k)=\eta$, we obtain

$$\sum_{k=0}^{T-1}\big(\phi(x(k+1))-\phi^\star\big)\leq\sum_{k=0}^{T-1}\frac{1}{2\eta}\big\|e\big(d(k)\big)\big\|_*^2+\sum_{k=0}^{T-1}\big\langle e\big(d(k)\big),x(k)-x^\star\big\rangle+\frac{D_\omega\big(x(0),x^\star\big)}{\gamma},\tag{15}$$

for all $T\in\mathbb{N}$. Each $x(k)$, $k\in\mathbb{N}$, is a deterministic function of the history $\xi_{[k-1]}:=\{\xi_i(t)\mid i=1,\ldots,b,\ t=0,\ldots,k-1\}$ but not of $\xi_i(k)$. Since $\nabla f(x)=\mathbb{E}_\xi[\nabla_x F(x,\xi)]$, it follows that

$$\mathbb{E}_{|\xi_{[k-1]}}\left[\big\langle e\big(d(k)\big),x(k)-x^\star\big\rangle\right]=0.$$

Moreover, as ξ_i and ξ_j are independent whenever $i \neq j$, it follows from Lemma 3 that

$$\begin{aligned}\mathbb{E}\big[\|e\big(d(k)\big)\|_*^2\big] &= \mathbb{E}\left[\left\|\frac{1}{b}\sum_{i=1}^{b}\big(\nabla f(x(d(k))) - \nabla_x F(x(d(k)),\xi_i)\big)\right\|_*^2\right] \\ &\leq \frac{c}{b^2}\sum_{i=1}^{b}\mathbb{E}\left[\|\nabla f(x(d(k))) - \nabla_x F(x(d(k)),\xi_i)\|_*^2\right] \\ &\leq \frac{c\sigma^2}{b},\end{aligned}$$

where the last inequality follows from Assumption 3. Taking expectation on both sides of (15) and using the above observations yield

$$\sum_{k=1}^{T}\big(\mathbb{E}[\phi(x(k))] - \phi^\star\big) \leq \frac{c\sigma^2}{2\eta b}T + \frac{D_\omega\big(x(0),x^\star\big)}{\gamma}.$$

By the convexity of ϕ, we have

$$\phi\big(x_{\text{ave}}(T)\big) = \phi\left(\frac{1}{T}\sum_{k=1}^{T}x(k)\right) \leq \frac{1}{T}\sum_{k=1}^{T}\phi\big(x(k)\big),$$

which implies that

$$\mathbb{E}\big[\phi(x_{\text{ave}}\big(T)\big)\big] - \phi^\star \leq \frac{c\sigma^2}{2\eta b} + \frac{D_\omega\big(x(0),x^\star\big)}{\gamma T}.$$

Substituting $\eta = \gamma^{-1} - L(\tau_{\max}+1)^2$ into the above inequality proves the theorem.

B. Proof of Theorem 2

Assume that the step-size $\{\gamma(k)\}_{k\in\mathbb{N}_0}$ is chosen such that $\gamma(k)^{-1} = L(\tau_{\max}+1)^2 + \alpha(k)$ where

$$\alpha(k) = \frac{\sigma\sqrt{c}\sqrt{k+1}}{R\sqrt{b}}.$$

Since $\gamma(k)$ is a non-increasing sequence, and $D_\omega(x,y) \leq R^2$ for all $x, y \in$ dom Ψ, we have

$$\sum_{k=0}^{T-1}\left(\frac{1}{\gamma(k+1)} - \frac{1}{\gamma(k)}\right)D_\omega\big(x(k+1),x^\star\big) \leq \left(\frac{1}{\gamma(T)} - \frac{1}{\gamma(0)}\right)R^2.$$

Applying Lemma 2 with $\mu_\Psi = 0$ and $\eta(k) = \alpha(k)$, taking expecation, and using Lemma 3 completely identically to the proof of Theorem 1, we then obtain

$$\sum_{k=1}^{T}\big(\mathbb{E}[\phi(x(k))] - \phi^\star\big) \leq \frac{R^2}{\gamma(T)} + \frac{c\sigma^2}{2b}\sum_{k=0}^{T-1}\frac{1}{\alpha(k)}. \tag{16}$$

Viewing the sum as an lower-estimate of the integral of the function $y(t) = 1/\sqrt{t+1}$, one can verify that

$$\begin{aligned}\sum_{k=0}^{T-1}\frac{1}{\alpha(k)} = \sum_{k=0}^{T-1}\frac{1}{\widetilde{\alpha}\sqrt{k+1}} &\leq \frac{1}{\widetilde{\alpha}}\left(1 + \int_0^{T-1}\frac{dt}{\sqrt{t+1}}\right) \\ &\leq \frac{2\sqrt{T}}{\widetilde{\alpha}},\end{aligned}$$

where $\widetilde{\alpha} = (\sigma\sqrt{c})/(R\sqrt{b})$. Substituting this inequality into the bound (16), we obtain the claimed guaranteed bound.

C. Proof of Theorem 3

Assume that the step-size $\{\gamma(k)\}_{k\in\mathbb{N}_0}$ in Algorithm 1 is set to $\gamma(k)^{-1} = 2L(\tau_{\max}+1)^2 + \beta(k)$, with

$$\beta(k) = \frac{\mu_\Psi}{3Q}\big(k+\tau_{\max}+1\big).$$

We first describe some important properties of $\gamma(k)$ relevant to our proof. Clearly, $\gamma(k)$ is non-increasing, *i.e.*,

$$\frac{1}{\gamma(k)} \leq \frac{1}{\gamma(k+1)}, \tag{17}$$

for all $k\in\mathbb{N}_0$. Since $\gamma(0)^{-1} \leq \gamma(k)^{-1}$, we have

$$2L(\tau_{\max}+1)^2 + \frac{\mu_\Psi \tau_{\max}}{3Q} \leq \frac{1}{\gamma(k)}. \tag{18}$$

Moreover, one can easily verify that

$$\begin{aligned}
\frac{1}{\gamma(k+1)^2} - \frac{1}{\gamma(k)^2} &= \frac{\mu_\Psi}{Q}\left(\frac{4L}{3}(\tau_{\max}+1)^2 + \frac{\mu_\Psi}{3Q}\left(\frac{2}{3}(k+\tau_{\max})+1\right)\right)\\
&\leq \frac{\mu_\Psi}{Q}\left(2L(\tau_{\max}+1)^2 + \frac{\mu_\Psi}{3Q}\big(k+\tau_{\max}+1\big)\right)\\
&= \frac{\mu_\Psi}{Q}\frac{1}{\gamma(k)},
\end{aligned}$$

which implies that

$$\frac{1}{\gamma(k+1)^2} \leq \frac{1}{\gamma(k)}\left(\frac{1}{\gamma(k)} + \frac{\mu_\Psi}{Q}\right), \tag{19}$$

for all $k\in\mathbb{N}_0$. Finally, by the definition of $\gamma(k)$, we have

$$\begin{aligned}
\frac{\gamma(k)}{\gamma(k+\tau_{\max})} &= 1 + \frac{\frac{\mu_\Psi}{3Q}\tau_{\max}}{2L(\tau_{\max}+1)^2 + \frac{\mu_\Psi}{3Q}\big(k+\tau_{\max}+1\big)}\\
&\leq 1 + \frac{\mu_\Psi \tau_{\max}}{6LQ(\tau_{\max}+1)^2},
\end{aligned}$$

and hence,

$$\frac{1}{\gamma(k+\tau_{\max})} \leq \left(1 + \frac{\mu_\Psi\tau_{\max}}{6LQ(\tau_{\max}+1)^2}\right)\frac{1}{\gamma(k)}. \tag{20}$$

We are now ready to prove Theorem 3. Applying Lemma 1 with

$$\eta(k) = \frac{1}{2\gamma(k)}, \quad k\in\mathbb{N}_0,$$

and using the fact

$$D_\omega\big(x(k+1),x^\star\big) \leq \frac{Q}{2}\big\|x(k+1)-x^\star\big\|^2,$$

by Assumption 5, we obtain

$$\begin{aligned}
\phi\big(x(k+1)\big) - \phi^\star + \left(\frac{1}{\gamma(k)} + \frac{\mu_\Psi}{Q}\right) D_\omega\big(x(k+1),x^\star\big) \leq{}& \gamma(k)\big\|e\big(d(k)\big)\big\|_*^2\\
&+ \big\langle e\big(d(k)\big), x(k)-x^\star\big\rangle + \frac{1}{\gamma(k)}D_\omega\big(x(k),x^\star\big)\\
&+ \frac{L(\tau_{\max}+1)}{2}\sum_{j=0}^{\tau_{\max}}\big\|x(k-j)-x(k-j+1)\big\|^2\\
&- \frac{1}{4\gamma(k)}\|x(k+1)-x(k)\|^2.
\end{aligned}$$

Multiplying both sides of this relation by $1/\gamma(k)$, and then using (19), we have

$$\begin{aligned}\frac{1}{\gamma(k)}\big(\phi\big(x(k+1)\big)-\phi^\star\big)+\frac{1}{\gamma(k+1)^2}D_\omega\big(x(k+1),x^\star\big) \le \big\|e\big(d(k)\big)\big\|_*^2 \\ + \frac{1}{\gamma(k)}\big\langle e\big(d(k)\big),x(k)-x^\star\big\rangle + \frac{1}{\gamma(k)^2}D_\omega\big(x(k),x^\star\big) \\ + \frac{L(\tau_{\max}+1)}{2\gamma(k)}\sum_{j=0}^{\tau_{\max}}\big\|x(k-j)-x(k-j+1)\big\|^2 \\ - \frac{1}{4\gamma(k)^2}\|x(k+1)-x(k)\|^2.\end{aligned}$$

Summing the above inequality from $k=0$ to $k=T-1$, $T\in\mathbb{N}$, and dropping the first term on the left-hand side yield

$$\begin{aligned}\frac{1}{\gamma(T)^2}D_\omega\big(x(T),x^\star\big) \le \sum_{k=0}^{T-1}\big\|e\big(d(k)\big)\big\|_*^2 \\ + \sum_{k=0}^{T-1}\frac{1}{\gamma(k)}\big\langle e\big(d(k)\big),x(k)-x^\star\big\rangle + \frac{1}{\gamma(0)^2}D_\omega\big(x(0),x^\star\big) \\ + \frac{L(\tau_{\max}+1)}{2}\sum_{k=0}^{T-1}\sum_{j=0}^{\tau_{\max}}\frac{1}{\gamma(k)}\big\|x(k-j)-x(k-j+1)\big\|^2 \\ - \frac{1}{4}\sum_{k=0}^{T-1}\frac{1}{\gamma(k)^2}\|x(k+1)-x(k)\|^2. \qquad (21)\end{aligned}$$

What remains is to bound the third term on the right-hand side of (21). It follows from (17)–(20) that

$$\begin{aligned}\frac{L(\tau_{\max}+1)}{2}\sum_{k=0}^{T-1}\sum_{j=0}^{\tau_{\max}}\frac{1}{\gamma(k)}\big\|x(k-j)-x(k-j+1)\big\|^2 &= \frac{L(\tau_{\max}+1)}{2}\sum_{j=0}^{\tau_{\max}}\sum_{k=0}^{T-j-1}\frac{1}{\gamma(k+j)}\big\|x(k)-x(k+1)\big\|^2 \\ &\le \frac{L(\tau_{\max}+1)}{2}\sum_{j=0}^{\tau_{\max}}\sum_{k=0}^{T-1}\frac{1}{\gamma(k+j)}\big\|x(k)-x(k+1)\big\|^2 \\ &\overset{(17)}{\le} \frac{L(\tau_{\max}+1)}{2}\sum_{j=0}^{\tau_{\max}}\sum_{k=0}^{T-1}\frac{1}{\gamma(k+\tau_{\max})}\big\|x(k)-x(k+1)\big\|^2 \\ &= \frac{L(\tau_{\max}+1)^2}{2}\sum_{k=0}^{T-1}\frac{1}{\gamma(k+\tau_{\max})}\big\|x(k)-x(k+1)\big\|^2 \\ &\overset{(20)}{\le} \frac{2L(\tau_{\max}+1)^2+\frac{\mu_\Psi\tau_{\max}}{3Q}}{4}\sum_{k=0}^{T-1}\frac{1}{\gamma(k)}\big\|x(k)-x(k+1)\big\|^2 \\ &\overset{(18)}{\le} \frac{1}{4}\sum_{k=0}^{T-1}\frac{1}{\gamma(k)^2}\big\|x(k)-x(k+1)\big\|^2.\end{aligned}$$

Substituting the above inequality into (21), and then taking expectation on both sides (similarly to the proof of Theorems 1 and 2), we have

$$\frac{1}{\gamma(T)^2}\mathbb{E}\big[D_\omega\big(x(T),x^\star\big)\big] \le \frac{c\sigma^2T}{b}+\frac{1}{\gamma(0)^2}D_\omega\big(x(0),x^\star\big). \qquad (22)$$

According to Remark 1,

$$\frac{1}{2}\|x(T)-x^\star\|^2 \le D_\omega\big(x(T),x^\star\big).$$

Moreover, by the definition of $\gamma(k)$,

$$\frac{\mu_\Psi(T+1)}{3Q} \leq \beta(T) \leq \frac{1}{\gamma(T)}.$$

Combing these inequalities with the bound (22), we conclude

$$\mathbb{E}\big[\|x(T)-x^\star\|^2\big] \leq \frac{18c\sigma^2Q^2}{b\mu_\Psi^2(T+1)} + \frac{2\left(\frac{6LQ}{\mu_\Psi}+1\right)^2(\tau_{\max}+1)^4}{(T+1)^2}D_\omega\big(x(0),x^\star\big).$$

The proof is complete.

REFERENCES

[1] K. P. Bennett and O. L. Mangasarian, "Robust linear programming discrimination of two linearly inseparable sets," *Optimization Methods and Software*, vol. 1, no. 1, pp. 23–34, 1992.

[2] R. Tibshirani, "Regression shrinkage and selection via the Lasso," *Journal of the Royal Statistical Society: Series B (Methodological)*, pp. 267–288, 1996.

[3] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 67, no. 2, pp. 301–320, 2005.

[4] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning*. Springer, 2009.

[5] S. Shalev-Shwartz and A. Tewari, "Stochastic methods for l_1-regularized loss minimization," *Journal of Machine Learning Research*, vol. 12, pp. 1865–1892, 2011.

[6] H. Robbins and S. Monro, "A stochastic approximation method," *Annals of Mathematical Statistics*, pp. 400–407, 1951.

[7] 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.

[8] G. Lan, "An optimal method for stochastic composite optimization," *Mathematical Programming*, vol. 133, pp. 365–397, 2012.

[9] L. Xiao, "Dual averaging method for regularized stochastic learning and online optimization," *Advances in Neural Information Processing Systems*, pp. 2116–2124, 2009.

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

[11] S. Ghadimi and G. Lan, "Optimal stochastic approximation algorithms for strongly convex stochastic composite optimization I: A generic algorithmic framework," *SIAM Journal on Optimization*, vol. 22, no. 4, pp. 1469–1492, 2012.

[12] A. Nedić and S. Lee, "On stochastic subgradient mirror-descent algorithm with weighted averaging," *SIAM Journal on Optimization*, vol. 24, no. 1, pp. 84–107, 2014.

[13] D. Needell, R. Ward, and N. Srebro, "Stochastic gradient descent, weighted sampling, and the randomized Kaczmarz algorithm," *Advances in Neural Information Processing Systems*, pp. 1017–1025, 2014.

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

[15] A. Nedić, D. P. Bertsekas, and V. S. Borkar, "Distributed asynchronous incremental subgradient methods," *Studies in Computational Mathematics*, vol. 8, pp. 381–407, 2001.

[16] M. Zinkevich, J. Langford, and A. J. Smola, "Slow learners are fast," *Advances in Neural Information Processing Systems*, pp. 2331–2339, 2009.

[17] M. Zinkevich, M. Weimer, L. Li, and A. J. Smola, "Parallelized stochastic gradient descent," *Advances in Neural Information Processing Systems*, pp. 2595–2603, 2010.

[18] I. Lobel and A. Ozdaglar, "Distributed subgradient methods for convex optimization over random networks," *IEEE Transactions on Automatic Control*, vol. 56, no. 6, pp. 1291–1306, 2011.

[19] K. Tsianos, S. Lawlor, and M. G. Rabbat, "Communication/computation tradeoffs in consensus-based distributed optimization," *Advances in Neural Information Processing Systems*, pp. 1943–1951, 2012.

[20] B. Recht and C. Ré, "Parallel stochastic gradient algorithms for large-scale matrix completion," *Mathematical Programming Computation*, vol. 5, no. 2, pp. 201–226, 2013.

[21] M. Li, L. Zhou, Z. Yang, A. Li, F. Xia, D. G. Andersen, and A. Smola, "Parameter server for distributed machine learning," *Big Learning NIPS Workshop*, 2013.

[22] P. Bianchi and J. Jakubowicz, "Convergence of a multi-agent projected stochastic gradient algorithm for non-convex optimization," *IEEE Transactions on Automatic Control*, vol. 58, no. 2, pp. 391–405, 2013.

[23] B. McMahan and M. Streeter, "Delay-tolerant algorithms for asynchronous distributed online learning," *Advances in Neural Information Processing Systems*, pp. 2915–2923, 2014.

[24] M. Hong, "A distributed, asynchronous and incremental algorithm for nonconvex optimization: An ADMM based approach," *arXiv preprint arXiv:1412.6058*, 2014.

[25] M. Jaggi, V. Smith, M. Takác, J. Terhorst, S. Krishnan, T. Hofmann, and M. I. Jordan, "Communication-efficient distributed dual coordinate ascent," *Advances in Neural Information Processing Systems*, pp. 3068–3076, 2014.

[26] J. Marecek, P. Richtárik, and M. Takác, "Distributed block coordinate descent for minimizing partially separable functions," *arXiv preprint arXiv:1406.0238*, 2014.

[27] R. Zhang and J. Kwok, "Asynchronous distributed ADMM for consensus optimization," *Proceedings of the 31st International Conference on Machine Learning (ICML)*, pp. 1701–1709, 2014.

[28] Y. Zhang and L. Xiao, "Communication-efficient distributed optimization of Self-Concordant empirical loss," *arXiv preprint arXiv:1501.00263*, 2015.

[29] C.-J. Hsieh, H.-F. Yu, and I. S. Dhillon, "PASSCoDe: Parallel asynchronous stochastic dual coordinate descent," *arXiv preprint arXiv:1504.01365*, 2015.

[30] S. J. Wright, "Coordinate descent algorithms," *arXiv preprint arXiv:1502.04759*, 2014.

[31] P. Richtárik and M. Takác, "Parallel coordinate descent methods for big data optimization," *Mathematical Programming*, pp. 1–52, 2015.

[32] O. Dekel, R. Gilad-Bachrach, O. Shamir, and L. Xiao, "Optimal distributed online prediction using mini-batches," *Journal of Machine Learning Research*, vol. 13, no. 1, pp. 165–202, 2012.

[33] F. Niu, B. Recht, C. Ré, and S. J. Wright, "Hogwild!: A lock-free approach to parallelizing stochastic gradient descent." *Advances in Neural Information Processing Systems*, pp. 693–701, 2011.

[34] J. Liu, S. J. Wright, C. Ré, V. Bittorf, and S. Sridhar, "An asynchronous parallel stochastic coordinate descent algorithm," *Proceedings of the 31st International Conference on Machine Learning (ICML)*, pp. 469–477, 2014.

[35] J. Liu and S. J. Wright, "Asynchronous stochastic coordinate descent: Parallelism and convergence properties," *SIAM Journal on Optimization*, vol. 25, no. 1, pp. 351–376, 2015.

[36] A. Agarwal and J. C. Duchi, "Distributed delayed stochastic optimization," *IEEE Conference on Decision and Control*, pp. 5451–5452, 2012.

[37] A. Beck and M. Teboulle, "Mirror descent and nonlinear projected subgradient methods for convex optimization," *Operations Research Letters*, vol. 31, no. 3, pp. 167–175, 2003.

[38] P. Tseng, "Approximation accuracy, gradient methods, and error bound for structured convex optimization," *Mathematical Programming*, vol. 125, no. 2, pp. 263–295, 2010.

[39] J. Duchi, S. Shalev-Shwartz, Y. Singer, and A. Tewari, "Composite objective mirror descent," *In Annual Conference on Learning Theory (COLT)*, 2010.

[40] R. Rockafellar and R. Wets, "On the interchange of subdifferentiation and conditional expectation for convex functionals," *Stochastics: An International Journal of Probability and Stochastic Processes*, vol. 7, no. 3, pp. 173–182, 1982.

[41] H. H. Bauschke and P. L. Combettes, *Convex analysis and monotone operator theory in Hilbert spaces*. Springer, 2011.

[42] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li, "RCV1: A new benchmark collection for text categorization research," *Journal of Machine Learning Research*, vol. 5, pp. 361–397, 2004.

[43] G. Chen and M. Teboulle, "Convergence analysis of a proximal-like minimization algorithm using Bregman functions," *SIAM Journal on Optimization*, vol. 3, no. 3, pp. 538–543, 1993.

[44] A. Cotter, O. Shamir, N. Srebro, and K. Sridharan, "Better mini-batch algorithms via accelerated gradient methods," *Advances in Neural Information Processing Systems*, pp. 1647–1655, 2011.